Radboud University

AI for hardware security: Friend or Foe

AMSec Workshop: Systems Security
February 4, 2025

Lejla Batina

Institute for Computing and Information Sciences
Radboud University
lejla@cs.ru.nl

Outline

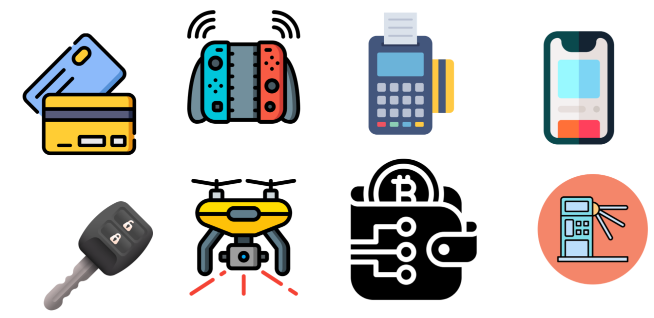
Embedded crypto and side-channel analysis (SCA)

AI and Side-channel analysis

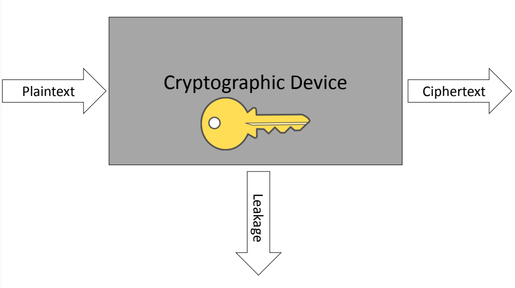
Side-channel analysis of AI implementations

Embedded crypto and side-channel analysis (SCA)

Known challenge: embedded crypto devices



Side-channel analysis (SCA)= Greybox scenario



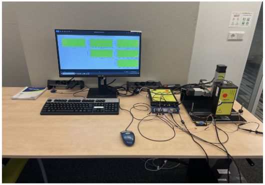
Greybox = SCA adversary in the wild:

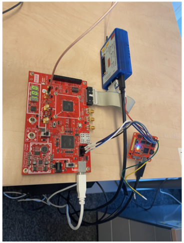
- Crypto is **implemented on a real device** such as a microcontroller, FPGA, ASIC
- Adversary can measure and process *physical quantities* in the device's vicinity
- Adversary's goal: secret key, message recovery, IP, etc.

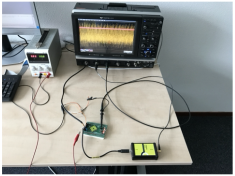
Whitebox = Security evaluator:

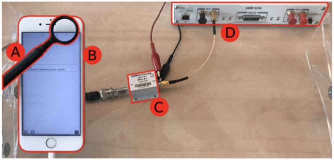
- Algorithms and implementation details are (partially) known
- Adversary's goal: secret key or message recovery by observing input/output pairs while trying all attacks possible

SCA/FI setups @CESCAlab



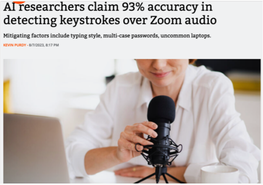






Relevance

August, 2023



AI researchers claim 93% accuracy in detecting keystrokes over Zoom audio

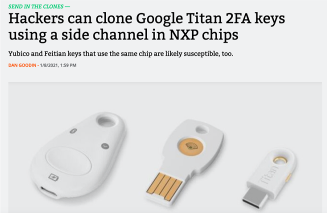
Mitigating factors include typing style, multi-case passwords, uncommon laptops.

KEVIN PURDY - 8/7/2023, 8:17 PM

September, 2024

October, 2019

January, 2021



SEND IN THE CLONES —

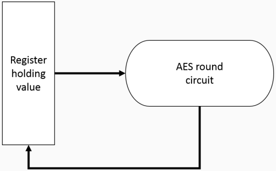
Hackers can clone Google Titan 2FA keys using a side channel in NXP chips

Yubico and Feitian keys that use the same chip are likely susceptible, too.

DAN GOODIN - 1/8/2021, 1:59 PM

Power side-channel: Modeling the leakage

- The Hamming distance model counts the number of $0 \rightarrow 1$ and $1 \rightarrow 0$ transitions
- Example: Assume a hardware register R storing the result of an AES round. The register initially contains value v_0 and gets overwritten with value v_1



- The power consumption because of the register transition $v_0 \rightarrow v_1$ is related to the number of bit flips that occurred
- Thus it can be modeled as $\text{HammingDistance}(v_0, v_1) = \text{HammingWeight}(v_0 \oplus v_1)$

Differential Power Analysis (DPA)

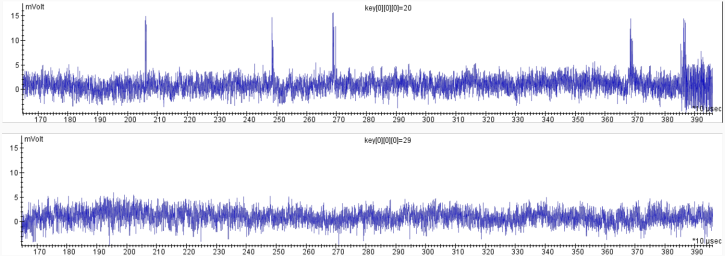


Figure: **Distance of means test plotted over time for the correct and a wrong key.**

- The most popular side-channel attack on crypto implementations
- Aims at recovering the secret key by using a large number of power measurements (traces) collected for known inputs or outputs
- Nowadays often combined/replaced with a leakage evaluation methodology such as Test Vector Leakage Assessment (TVLA)

Test Vector Leakage Assessment (TVLA)

- It is using Welch's t-test to differentiate between two sets of measurements, one with fixed inputs and the other with random inputs
- Leakage assessment of a device is very important for the semiconductor and the security evaluation industries
- Number of attacks to check the device's resistance against keeps on growing
- Various attackers' models possible but security evaluation often goes for the strongest adversary

Countermeasures against DPA: main idea

Goal: break the link between the actual data and power consumption

- Masking: power consumption remains dependent on the data on which computation is performed but not the **actual** data
- Hiding: power consumption is independent of the intermediate values and of the operations

Masking

Boolean masking: a dth-order (Boolean) masking scheme splits an internal sensitive value v into $d+1$ shares $(v_0, v_1, ..., v_d)$, as follows:

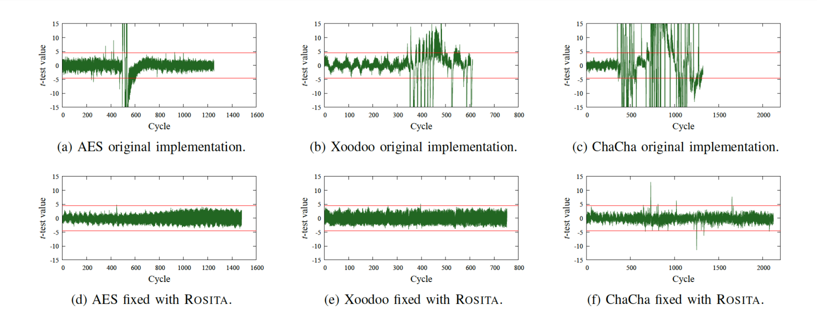
$$v = v_0 \oplus v_1 \oplus \cdots \oplus v_d$$

Probing-secure scheme. We refer to a scheme that uses certain families of shares as $d-$th order probing-secure iff any set of at most d intermediate variables is independent from the sensitive values.

Consequently, the leakage of up to d values does not disclose any information to the attacker.

Masking in practice: unintended interactions between values in the processor cause leakage in 1st order (caused often by transitional effects and glitches).

TVLA on 1st-order secure implementations



(a) AES original implementation.

(b) Xoodoo original implementation.

(c) ChaCha original implementation.

(d) AES fixed with ROSITA.

(e) Xoodoo fixed with ROSITA.

(f) ChaCha fixed with ROSITA.

The slowdowns of the "fixes" for ChaCha, Xoodoo and AES are 61% (1 322 vs. 2 122 cycles), 18% (637 vs. 753 cycles) and 15% (1 285 vs 1 479).

M. A. Shelton, N. Samwel, L. Batina, F. Regazzoni, M. Wagner, Y. Yarom: Rosita: Towards Automatic Elimination of Power-Analysis Leakage in Ciphers. NDSS 2021.

AI and Side-channel analysis

AI entering Crypto and Security

Security applications with AI

- AI in Security market is expected to exceed US $ 61.30 Bil. by 2027
- ML applications: image recognition, natural languages, robotics, . . .
- ML in IoT devices: image and speech recognition
- AI in small devices manipulating our data and affecting our privacy

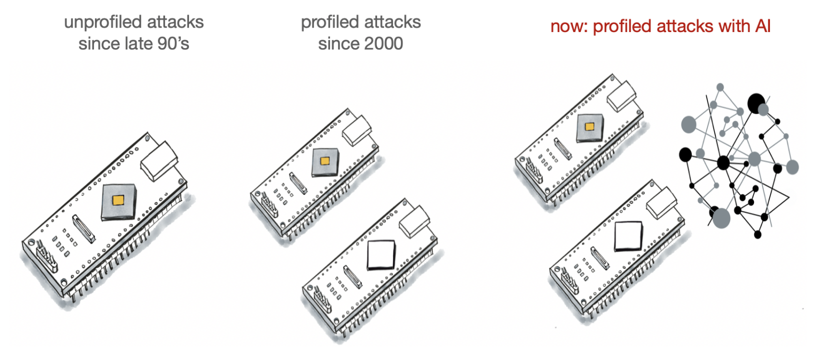
AI in cryptography

- Privacy-preserving AI
- AI for cryptanalysis
- AI-assisted SCA and SCA of AI

AI and Side-channel Analysis (SCA): How it all started

- Machine learning (ML) for SCA was introduced 15 years ago:
  - ML improving DPA attacks (collaboration with Data Science@RU)
  - ML for attack preparation (collaboration with Riscure)
- Deep learning in SCA:
  - neural nets for profiled attacks
  - defeating countermeasures e.g. attacking higher-order masking
  - leakage assessment/simulators (first AI-based simulator, ABBY developed @CESCAlab)
  - TEMPEST-like techniques e.g. screen gleaning
- Attacks on AI:
  - SCA for reverse engineering neural net (NN) implementations
  - SCA for input recovery from NN implementations
  - Cryptographic attacks for NN parameters recovery

Side-channel evaluations

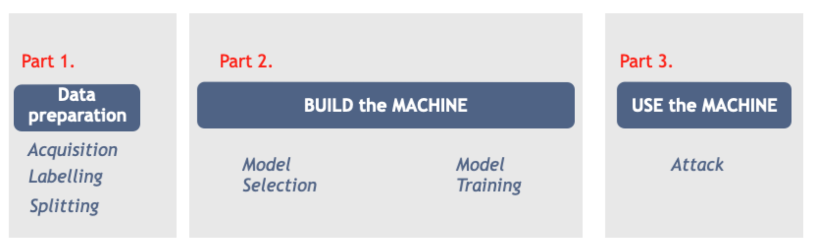


unprofiled attacks
since late 90's

profiled attacks
since 2000

now: profiled attacks with AI

Workflow machine learning



Part 1.

Data preparation

Acquisition

Labelling

Splitting

Part 2.

BUILD the MACHINE

Model Selection

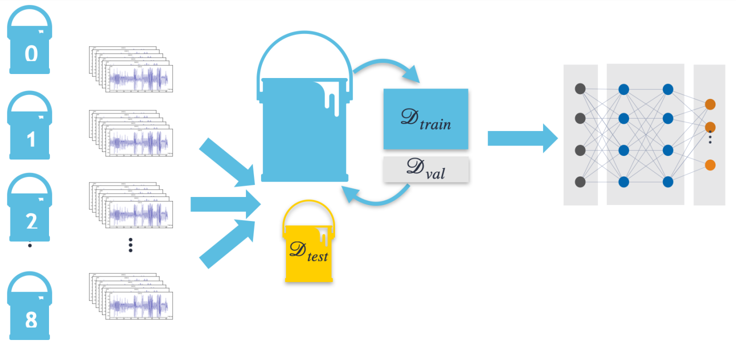
Model Training

Part 3.

USE the MACHINE

Attack

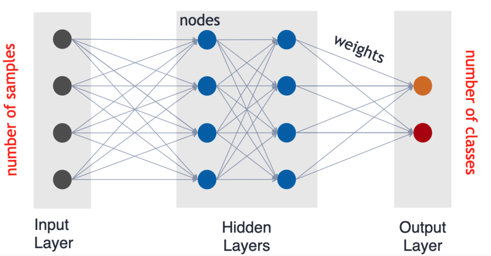
Data preparation for SCA



- Over-fitting: the model performance is very good on the training data, and on testing data is poor;
- Under-fitting: when the model does not produce accurate results on the training data;

A simple multilayer perceptron (MLP) architecture

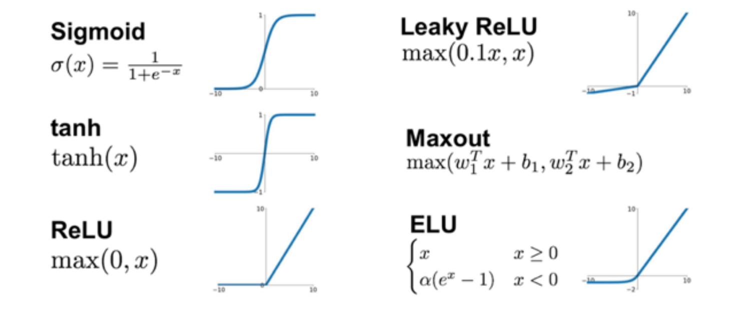
MLP architecture a series of layers formed of connected neurons. The strength of the connection between two neurons is determined by the associated weight.



- In SCA, we use relatively small networks and simple arch.: MLP and CNN
- During training the value of the weights and biases are adjusted

Activation functions

We also need to deal with non-linear functions.



Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$

tanh

$\tanh(x)$

ReLU

$\max(0, x)$

Leaky ReLU

$\max(0.1x, x)$

Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

Side-channel analysis of AI implementations

Motivation for SCA to reverse engineer NNs

- Well-trained models are valuable for certain industries
- In some cases parameters and other training details are considered IP
- Neural nets are being deployed on various platforms from low-end processors e.g. ARM Cortex-M, to FPGAs, GPUs etc.
- This makes the NN architectures and their parameters target for adversaries

New field: Physical SCA on NN implementations

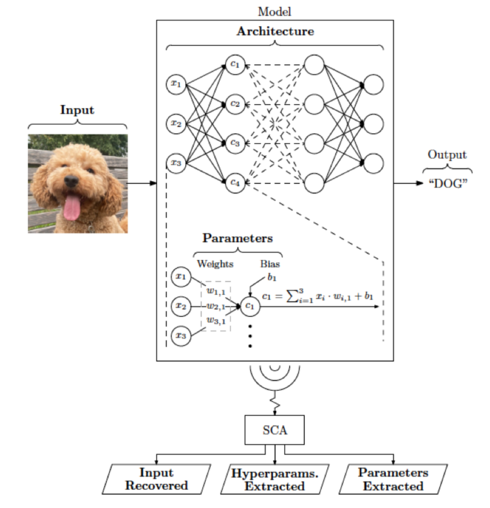


Figure: Physical SCA on a Multi-Layer Perception (MLP) model for image classification.

Threat model and attacker's capabilities

Goal: Recover the architecture of a pre-trained NN model executed on an embedded device while running inference using only side-channel information

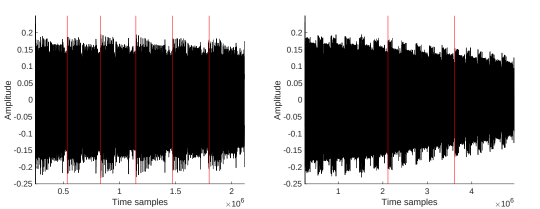
Threat model:

- Adversary can query the model with known/chosen inputs and passively observe side-channel information corresponding to the executed inference
- No specific assumption on the type of inputs or its source, as we work with real numbers

The attacker wants to learn information about:

- layers
- neurons
- activation functions
- weights

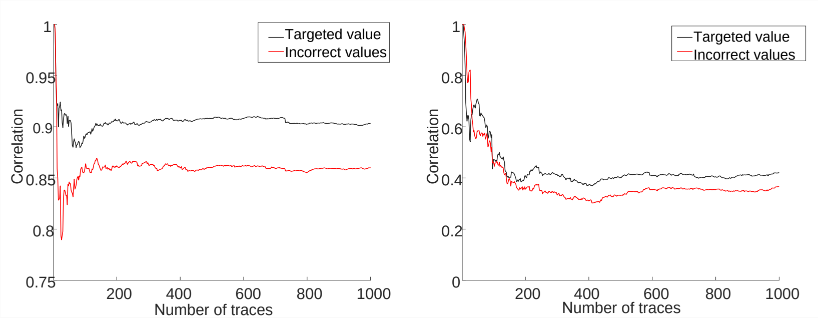
SEMA on hidden layers



(a) One hidden layer with 6 neurons

(b) 3 hidden layers (6,5,5 neurons each)

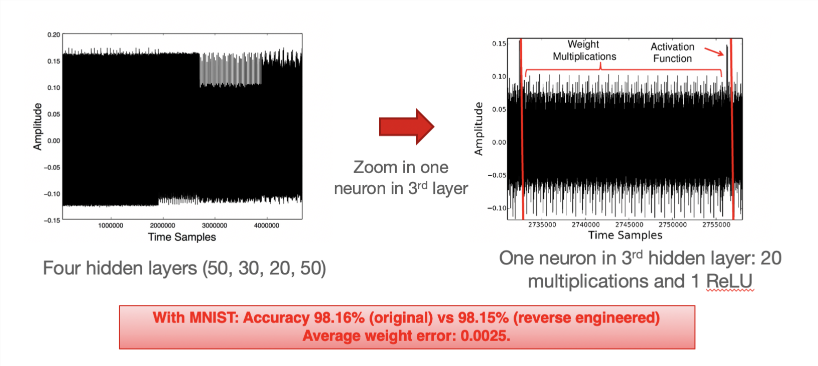
Weights recovery via DPA



(a) First byte recovery (sign and 7-bit exponent)

(b) Second byte recovery (lsb exponent and mantissa)

Results for MLP on ARM Cortex M-3



Four hidden layers (50, 30, 20, 50)

Zoom in one neuron in 3rd layer

One neuron in 3rd hidden layer: 20 multiplications and 1 ReLU

With MNIST: Accuracy 98.16% (original) vs 98.15% (reverse engineered)
Average weight error: 0.0025.

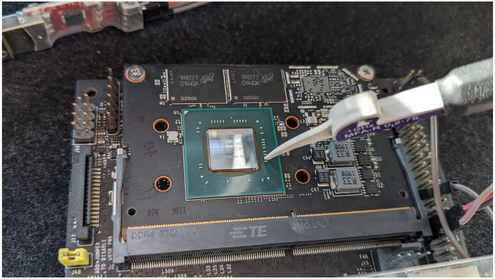
Lejla Batina, Shivam Bhasin, Dirmanto Jap, Stjepan Picek: CSI NN: Reverse Engineering of Neural Network Architectures Through Electromagnetic Side Channel. USENIX Security Symposium 2019: 515-532.

BarraCUDA: Attacking Nvidia GPUs

BarraCUDA: What is new

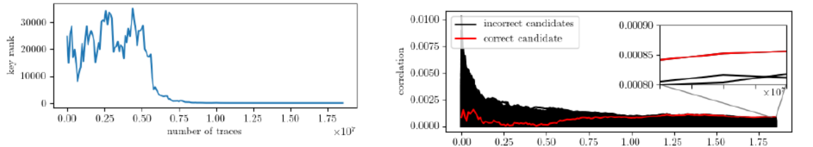
- Reverse engineering the closed source TensorRT library
- Using known SCA techniques (DEMA) to find the best location for the EM probe and the time when a specific neuron is evaluated
- Performed parameter extraction from the EfficientNet model running on an industry-strength Jetson Nano device
- Attack has a large complexity, which required developing a special CUDA-based attack implementation
- The attack requires 11-12 days (traces collection and alignment), and the parameters are recovered in 5-6 min per weight

Setup and the Threat model



- Goal: recover the trade secrets encoded in NNs parameters, from an ML model running on e.g. edge device
- Attacker learned the architecture details by some of known techniques
- Attacker has a physical access to the device and can monitor EM during the inference for known inputs

Results for weight recovery



(c) Key rank vs. number of traces using *HD* for the ninth weight in the first layer with value of -0.7705.

(d) Correlation vs. number of traces (10^7) using *HD* for the ninth weight in the first layer with value of -0.7705.

Key rank and correlations of the 9th weight in the first layer in the real-world CNN architecture.

P. Horvath, L. Chmielewski, L. Weissbart, L. Batina, Y. Yarom. BarraCUDA: GPUs do Leak DNN Weights, USENIX Security 2025, to appear.

BarraCUDA: Contributions

- Proprietary implementations of neural nets on GPU are vulnerable to parameter extraction using SCA
- Recovered weights and biases of real-world networks from Nvidia Jetson Nano and Nvidia Jetson Orin Nano
- Developed a CUDA-based implementation of DEMA to execute the attack an order of magnitude faster for large datasets (millions of traces)
- The attack on Jetson Orin Nano requires only 1 day for trace collection and 1 day for trace alignment with an input batch size of 1 (5 days for batch size 16)

AI in Crypto/Security: Where do we go from here

- Point 1: “Provably” secure implementations are regularly broken
- Point 2: AI-assisted SCA attacks are in general more powerful, than the “old” methods
- Point 3: SCA on NNs implementations (using AI) can recover architecture, inputs etc. and the protection is not straightforward
- SCA and AI are getting more and more intertwined
  - AI in leakage detection and assessment (now mandated by the security evaluation bodies)
  - AI-assisted fault analysis
  - Neural-aided cryptanalysis: unclear if it could break state-of-the-art algorithms

Thank you for your attention!

https://cescalab.cs.ru.nl/