I Have Got to Warn You, It Is a Learning Robot: Using Deep Learning Attribution Methods for Fault Injection Attacks

Karim M. Abdellatif

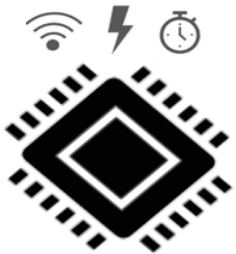
Whoami

Hardware Wallet Manufacturer

Donjon
Ledger's Security Research Team

Hardware attacks

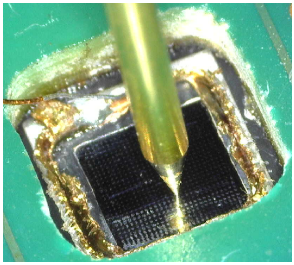
- **Fault injection**: Perturbing the chip during sensitive operations:
  - Power and clock glitches
  - Electromagnetic fault injection (EMFI)
  - Body biasing injection (BBI)
  - Laser fault injection (LFI)
- **Side-channel**: Investing leakages such as EM, power, or time to perform:
  - Simple power analysis (SPA)
  - Differential power analysis (DPA)
  - Profiling attacks



Motivation

- Working on black-box fault injection evaluations takes a lot of time.
- A lot of parameters should brute-forced:
  - Example: BBI or laser fault injection require tuning the following parameters: pulse power, pulse width, **vulnerable timing moments**, and XY point.
- Identifying vulnerable timing moments is one of the big challenges, especially under the case of countermeasures that require injecting multiple faults.

Having reverse engineering tools would be very useful in such evaluations.



(BBI attack [1])

[1] Donjon, "Breaking A Recent SoC's Hardware AES Accelerator Using Body Biasing Injection", HW.io 2022.

Outline

DEEP LEARNING IN HARDWARE SECURITY

DL-based SCAs

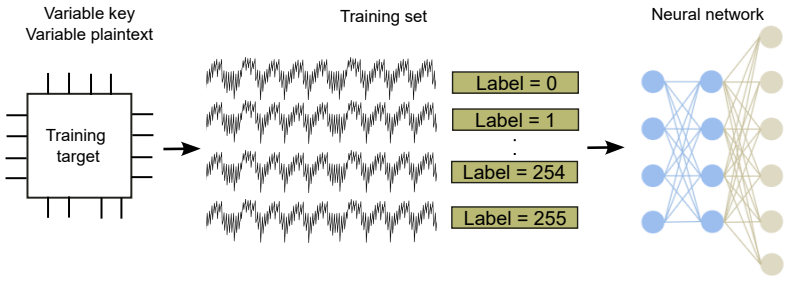
- DL-based SCAs [2]
  - Several devices for learning and test
  - Better efficiency in case of countermeasures [3]
- DL-based leakage detection [4]
  - It uses DL attribution methods to detect POIs.
  - Better than classical statistical techniques in case of countermeasures

[2] H. Maghrebi, T. Portigliatti, and E. Prouff. "Breaking cryptographic implementations using deep learning techniques", SPACE 2016.

[3] E. Cagli, C. Dumas, and E. Prouff "Convolutional neural networks with data augmentation against jitter-based countermeasures: Profiling attacks without pre-processing", CHES 2017

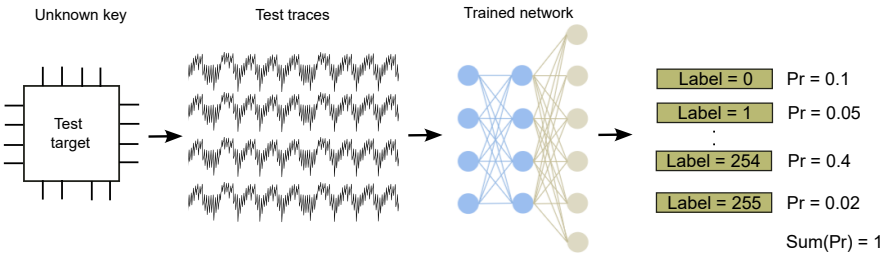
[4] L. Masure et *al*, Gradient Visualization for General Characterization in Profiling Attacks, IACR.

DL-based SCAs



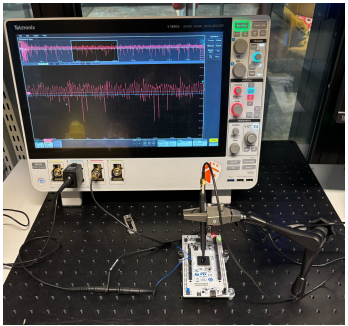
Profiling phase

DL-based SCAs

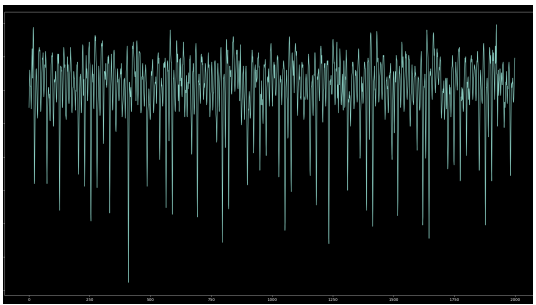


Test phase

Running AES-128 (first round) on a 32-bit MCU.



EM setup for STM32U5 - Donjon



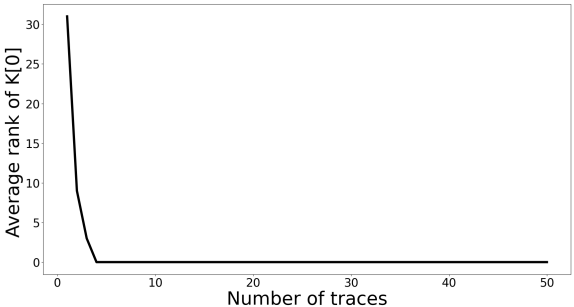
EM signal

MLP-based example

```
def mlp_model(sample_len, range_outer_layer):
    model = Sequential()
    model.add(Dense(20, input_dim=sample_len, activation=tf.nn.relu))
    model.add(Dense(10, activation=tf.nn.relu))
    model.add(Dense(range_outer_layer, activation=tf.nn.softmax))
    model.compile(
        optimizer="adam",
        loss="categorical_crossentropy",
        metrics=["accuracy"],
    )
    return model
```

Few traces to attack unknown key

- 500K traces for profiling
- 1K traces for test
- Labels on Sbox output

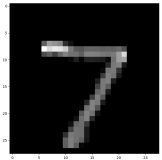


DEEP LEARNING ATTRIBUTION METHODS

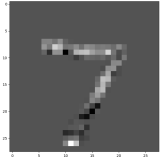
Attribution methods

- Such methods are used to interpret and understand the decisions made by deep neural networks.
- Identify which input features (e.g., pixels in an image) are most influential in the model's predictions.

Gradient-Based Methods, **Activation-Based Methods**, ...



Input challenge (predict 7)



Attribution result

It lies in tracing back the contributions of input nodes to the final prediction.

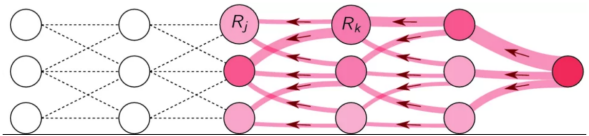


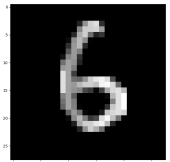
Illustration of the LRP procedure [5]

$$R_j = \sum_k \frac{z_{jk}}{\sum_j z_{jk}} R_k \qquad (1)$$

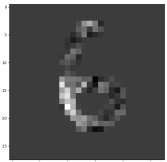
where j and k denote neurons in consecutive layers, and $z_{jk} = a_j w_{jk}$ is the activation of the neuron j multiplied by the weight between neuron j and neuron k.

[5] S. Bach et al. 'On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation.' PloS one 10.7 (2015).

MNIST example

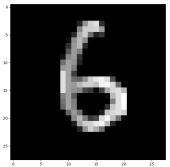


Input challenge

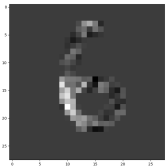


LRP

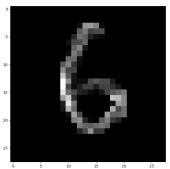
Other methods



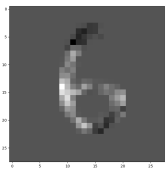
Input challenge



LRP



Taylor



Input

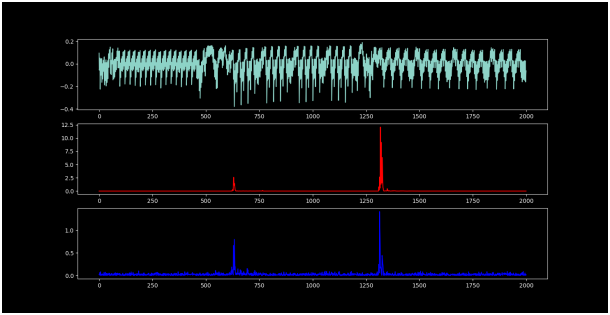
Previous work [6] exists in the side-channel domain.

Advantages: detecting leakage points in case of countermeasures (ex: masking and jitter) unlike SNR or T-test.

[6] B. Hettwer et al., " neural network attribution methods for leakage analysis and symmetric key recovery", SAC-2019.

Practical example - AES

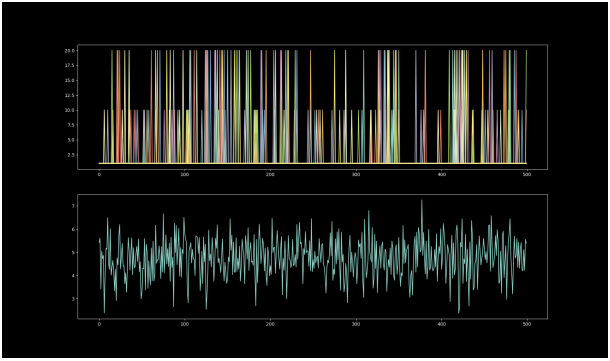
1. Collecting power traces from an AES
2. Profiling on *Sbox[0]*
3. Reverse-engineering using attribution methods
4. Comparing it with SNR/NICV



Difference between SNR and LRP

Another example

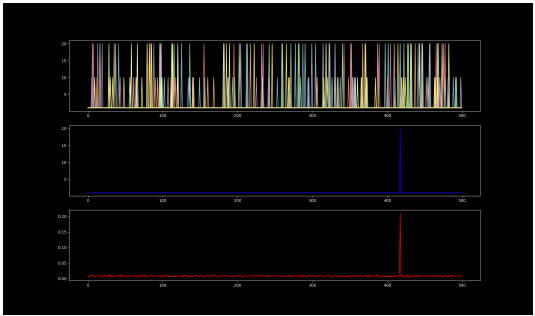
1. Two different values with jitter
2. T-test fails!



Upper: desynchronized traces, below: T-test

Using attribution methods

1. Two different values are randomized.
2. Profiling on the two labels of them
3. Timing is very well detected.
4. Advantages:
   - Decision scalability (one trace)
   - It can defeat countermeasures.

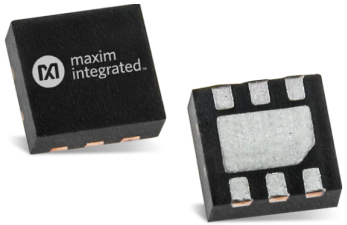


Sample trace and LRP

PRACTICAL CHALLENGE: DS28C36 FROM ANALOG DEVICES

Security features [7]

- ECC-256 computation engine
- FIPS 180 SHA-256 computation engine
- TRNG with NIST SP 800-90B compliant entropy source with function to read out
- 17-Bit one-time settable, non-volatile decrement-only counter with authenticated read
- **8Kbit of EEPROM for user data, keys, and certificates**
- The full data sheet is not available and this required some reverse to find the available commands and their parameters.

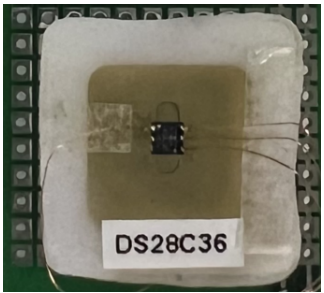


[7] https://www.analog.com/media/en/technical-documentation/data-sheets/DS28C36.pdf

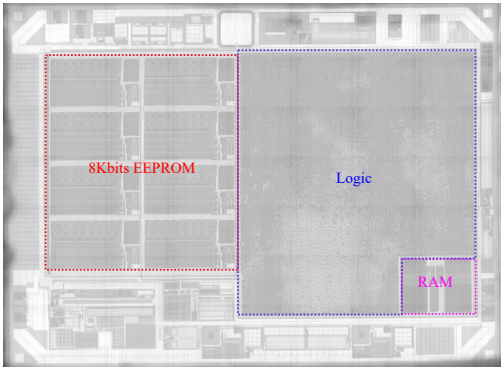
EEPROM organization

Page	Description
0 to 15	User pages
16 to 21	Public keys (x and y)
22 to 24	Private keys
25 to 26	Secret pages
27	Counter
28 to 29	Random
30 to 31	RAM buffer

Sample preparation

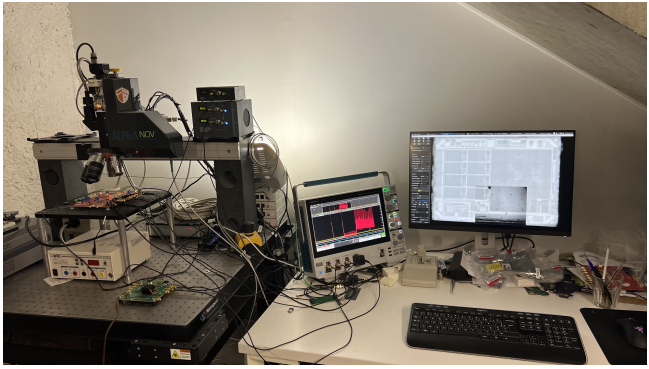


Decapped chip



Infrared backside image

- An infrared pulsed laser source and a microscope for focusing
- A Scaffold[8] board
- A Tektronix MSO44 oscilloscope
- DUT: DS28C36

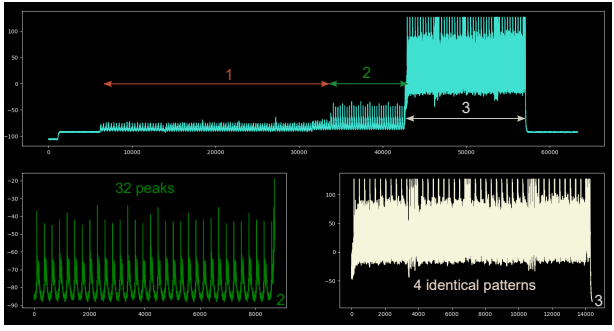


Setup

[8] O. Heriveaux. Scaffold. https://github.com/Ledger-Donjon/scaffold

Read page command

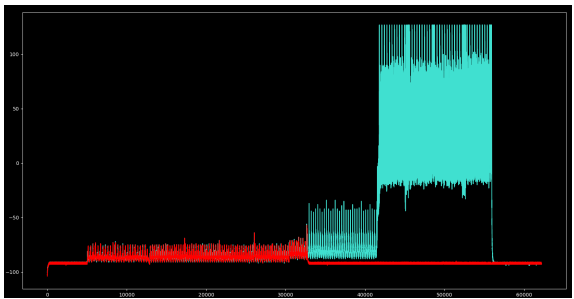
```
write_data(page_number, data)
read_page(page_number)
save_power_trace()
```



Power consumption in case of unprotected page

Unprotected and protected page

```
write_data(page_number, data)
read_page(page_number)
save_power_trace()
lock_page(page_number)
read_page(page_number)
save_power_trace()
```



Protected and unprotected

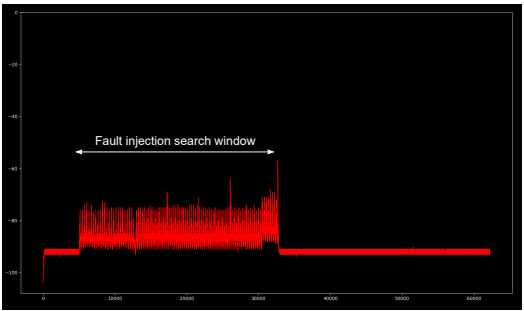
Attack scenario

Step 1:

```
write_data(page_number, data)
lock_page(page_number)
```

Step 2:

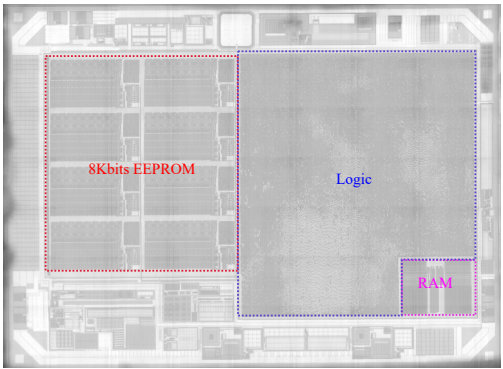
```
while True:
    prepare_fault() # single pulse
    chip_restart()
    read_page(page_number)
    save_log()
    move_laser()
```



Power consumption when the page is locked

Page configuration (bit or bits) can be:

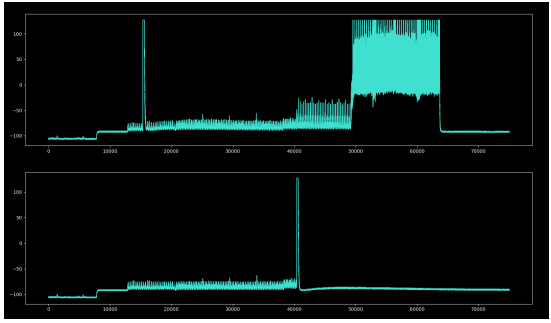
- Stored in the EEPROM
- Stored in eFuses
- Manipulated in the logic
- Temporarily stored in RAM



IR image

Number	Chip response	Note
0	2155ffffffffffffffffffffffffffff ffffffffffffffffffffffffffffffff	Locked
1	ffffffffffffffffffffffffffffffff ffffffffffffffffffffffffffffffff	Timeout
2	NACK	I2C communication error
3	21aab8289516978a7b25eb1d8a317f6c6a 71718b4d47de4754ac32a1d1c5adb7d324	Public key slot
4	21aa208cfc9a7dc7fcdb5437775fea79aa 2c95f5795ed2bfe883082a2ada0585694f	Needs to be investigated

- Correct read page trace for response 3
- In case of response 4, no EEPROM read
- It seems to be a RAM, or RNG content



Difference between response 3 (upper) and response 4 (below)

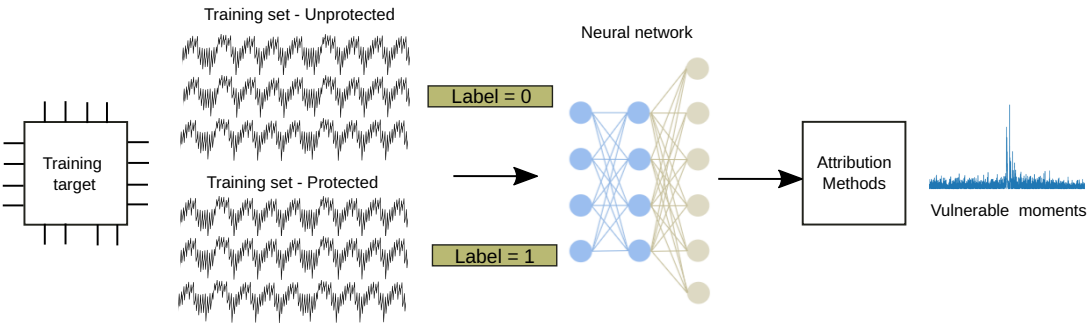
- The chip seems to be protected against single fault attacks? (black box evaluation)
- **Reverse-engineering the Read Page command is the only way to understand clearly.**
- **How can we do that? deep learning?**

APPLYING DL ATTRIBUTION METHODS INTO FAULT INJECTION

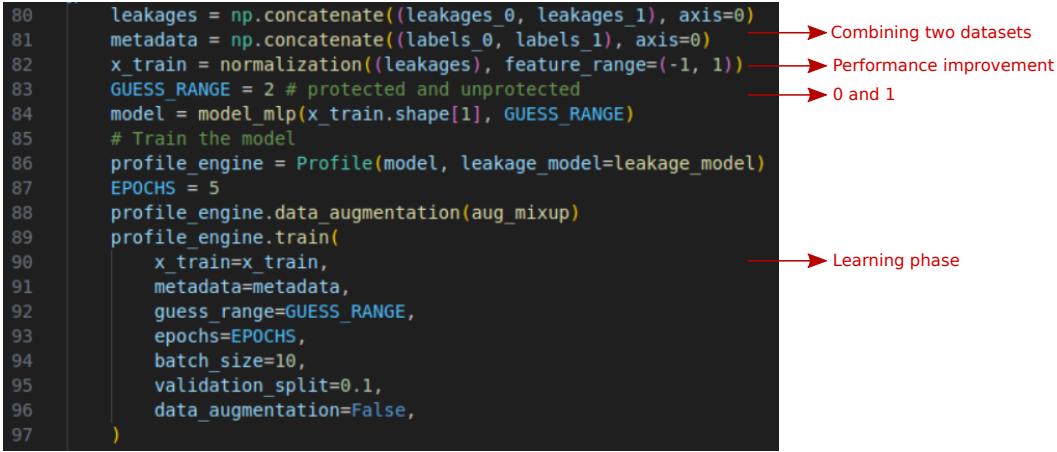
Application on read page command

- We will apply the DL attribution methods, which are used in SCAs to detect sensitive operations, into fault injection (FI).
- The main purpose is to detect when sensitive bits are processed.
- More precisely, we will try to locate on the power consumption trace, the manipulation of the page protection bit/bits.
  - The first set is collected when the page is unlocked (50K traces).
  - The second set is collected when the **same page** is locked (50K traces).

Methodology



Methodology

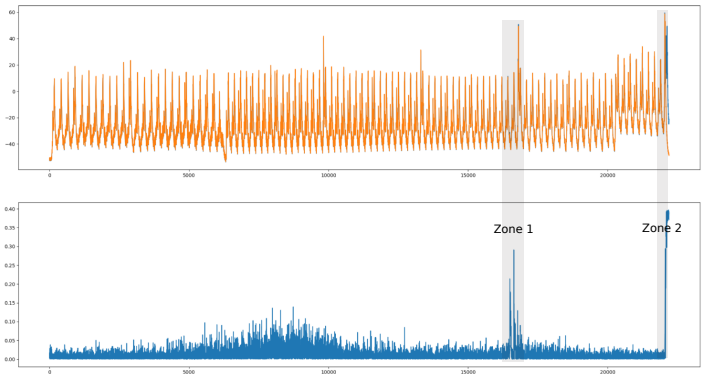


```
115     model_wo_sm = innvestigate.model_wo_softmax(model)
116     gradient_analyzer = innvestigate.analyzer.LRP(model_wo_sm)
117     trace_sample = x_test[0]
118     trace = trace_sample.reshape(1, x_test.shape[1])
119     vis_trace = gradient_analyzer.analyze(trace)[0]
```

LRP

Result

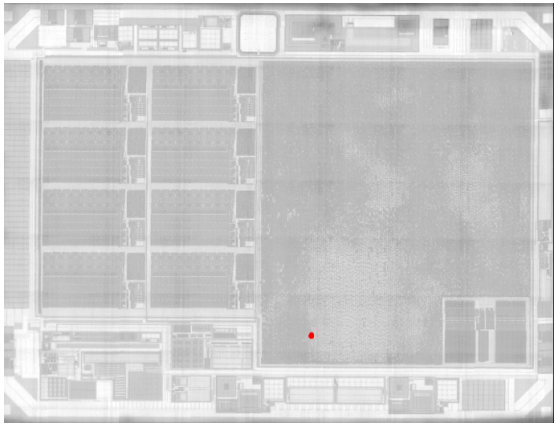
- Two zones
- Protected against single fault attacks



LRP result

Successful faults

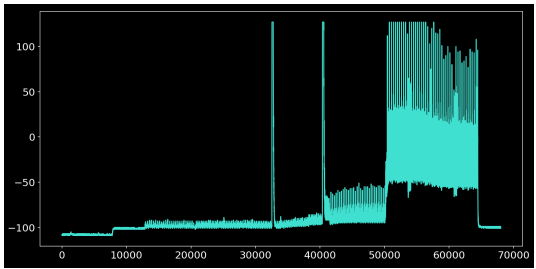
- Scanning the chip with double pulses
- Scanning the logic area
- Successful fault



Vulnerable spot

Double fault attack

- Fixing the laser beam on the correct location
- Success rate close to 99%



Power trace in case of a successful fault

This confirms the efficiency of DL attribution methods.

User pages vs other pages

- The presented attack is applicable on all the user pages.
- It isn't applicable on permanent-protected pages used for P256 curve private keys.
    - The chip passed a fixed unidentified value for these pages.

TOOLING

- Latest side-channel attacks using deep learning
- Leakage detection using deep learning attribution methods

ScadI: Open source tool - Donjon

Clone and investigate!

CONCLUSION

Conclusion

- DL attribution methods involved in this work, can be used when performing fault injection attacks in black box context.
- Manufacturers **must** consider such technique for testing countermeasures in addition to leakage detection techniques to detect vulnerable timing moments.
- Using double verification against fault injection attacks is not efficient enough if it is used alone.
- Manufacturers must at minimum combine it with strong hardware and/or software jitter as an additional countermeasure.

THANK YOU. QUESTIONS?

Karim M. Abdellatif, PhD
e-mail: karim.abdellatif@ledger.fr