Université de Rennes

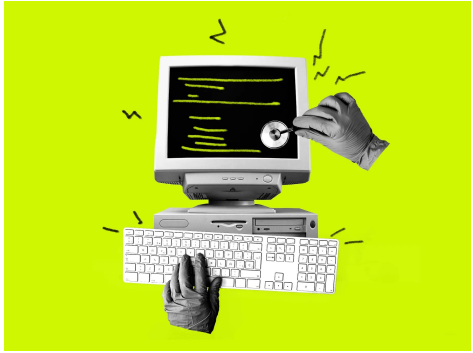
# Side-channel Project

## Influence of Pre-treatments on Side-channel Attacks

*Fabien Guihard – Thomas Feuilletin – Ulysse Le Huitouze – Evenn Resnais – Théo Mainguet*

*May 2024 - M1 Cyber*

# What is a Side-channel Attack?



- Uses physical measurements to break a cryptosystem.
- Various physical information sources are possible:

➢ Power consumption → The most common

➢ Electromagnetic emissions

➢ Acoustic signals

➢ Thermal imaging

# The Power Consumption source

- These recordings are referred to as "traces".
- Traces are measurable physical outputs recorded from a cryptographic device during the operation of encrypting or decrypting data.

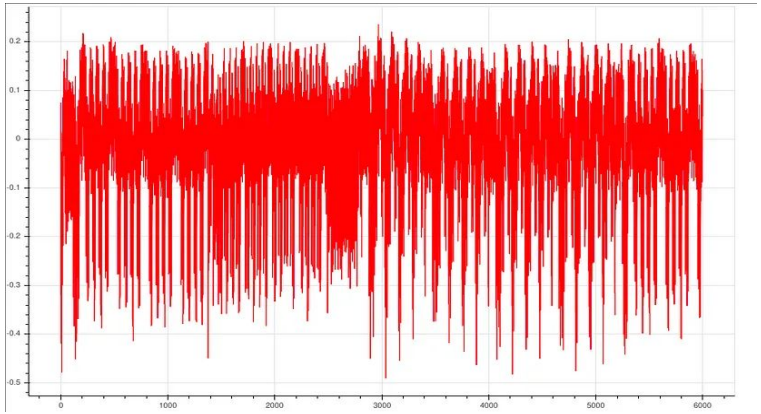



Fig. 1. Example of AES trace

➔ It is hard to differentiate the AES operations.

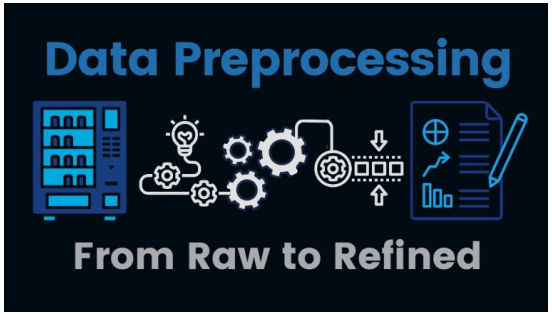
# Exploring Two Fundamental Side Channel Attacks

- Correlation Power Analysis (CPA):
  - ➔ Identifies direct relationships between key guesses and power traces using correlation coefficients.

- Linear Regression Analysis (LRA):
  - ➔ Uses regression to predict key bits from power consumption changes.

# Why is Preprocessing Critical in SCAs?

- Noise Reduction
- Improved Attack Performance
- Enhanced Analysis
    - ★ A topic barely addressed in current scholarly discussion!



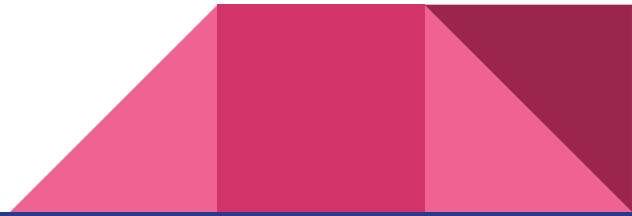
# Generic Attack Flow

1 - Pre-processing on the observations

2 - Pre-processing on the predictions for all estimations

3 - Processing on the predictions and observations

4 - Return a key candidate



# Power Models

- Hamming weight

- Hamming distance

# Implemented Attacks

- CPA with lascar library
- CPA in bare numpy
- CPA in bare numpy in accumulation mode
- LRA in bare numpy

# About lascar library


- Fast
- Black box
- Lack of documentation

# CPAs in bare numpy

- Slow (at first)
- Separated in 2 parts:
  - Computing the model for all hypothesis with data
  - Correlating the model with the leakages

# LRAs in bare numpy

- Slower than CPA
- Separated in parts:
  - Computing the model matrix
  - Quantify the error between the model and the leakages

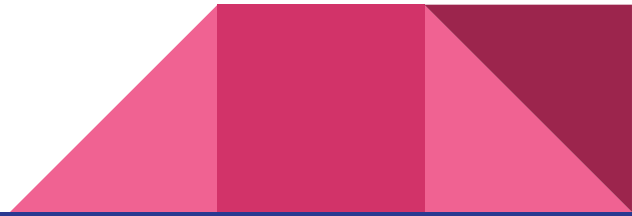
# Experiences

What did we use to achieve those attacks

- Datas
- Metrics
- Pre-treatments

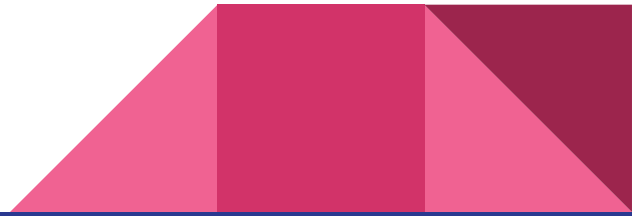
# Datas

- First classic traces
  - 1000 traces
  - Software
    - Software capture to get the traces
    - Hamming weight
- Extended AES HD Dataset
  - 500 000 traces
  - Github
  - Hardware
    - Physical capture
    - Hamming distance



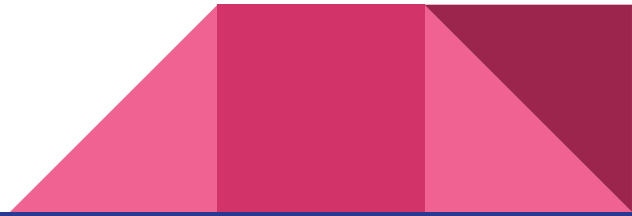
# Metrics

- Rank evolution
  - Tracks the positional change of the correct key among all candidates, evaluated over successive analyses
  - A rapidly decreasing rank indicates an attack increasing accuracy.
  - Helps in evaluating the efficiency of attack strategies and preprocessing techniques.
  - Software dataset
  - Hardware dataset



# Pre-treatments 1/3

- LRA
  - increase of traces overtime
- CPA
  - traces accumulation



# Pre-treatments 2/3

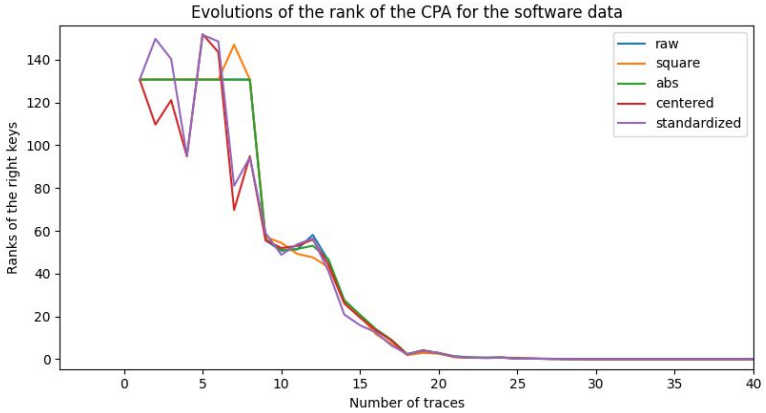
- Raw
  - Raw traces use to compare with pre-treatments
- Square
  - Applies squaring to each trace
- Absolute value
  - Converts all data points in each trace to their absolute values, to focus on the magnitude of fluctuations rather than their direction

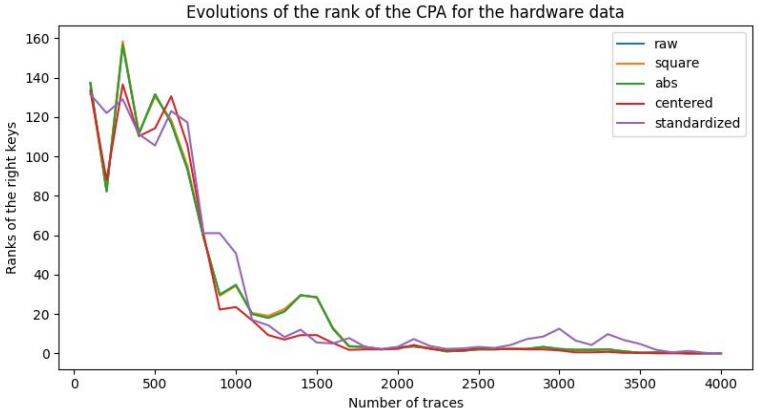
# Pre-treatments 3/3

- Centered
  - Subtracts the mean of each trace from all its values, effectively centering the trace around zero
- Standardized
  - Center the traces by subtracting the mean
  - reduce them by dividing their gap

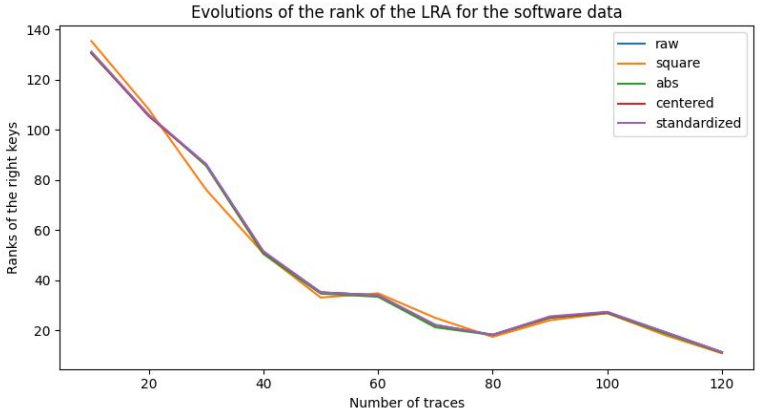
# Results

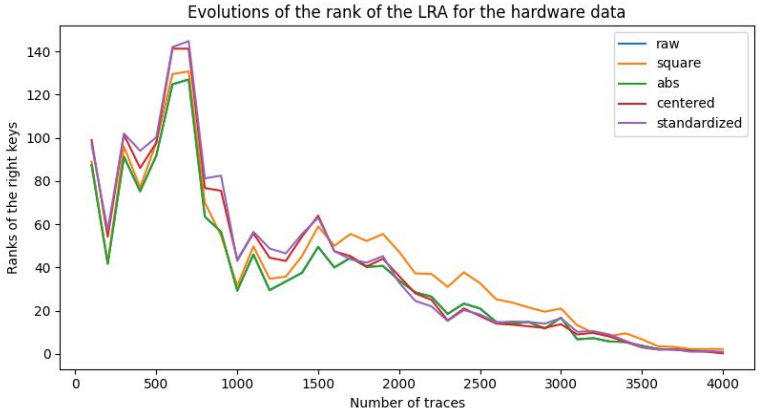
# Ranks evolution for CPA



# Ranks evolution for LRA



# What’s next ?

- Repeat the experiment with random traces
- Look for pre-traitement influence at byte level
- Try more pre-traitements

# Thank you!