14/03/2024

Side-Channel Analysis and Attacks

Davide Bellizia (davide.bellizia@telsy.it)

Outline

1. Physical Attacks
2. Side-Channel Analysis
3. Anatomy of a Trace
4. Advanced Encryption Standard
5. Detection and Analysis
6. CPA Attack
7. Countermeasures

Outline

1. **Physical Attacks**
2. Side-Channel Analysis
3. Anatomy of a Trace
4. Advanced Encryption Standard
5. Detection and Analysis
6. CPA Attack
7. Countermeasures

Physical Attacks

Cryptography and cryptographic techniques only do not guarantee the security of data, information and devices.

Other factors impact on the design of a **secure system**:

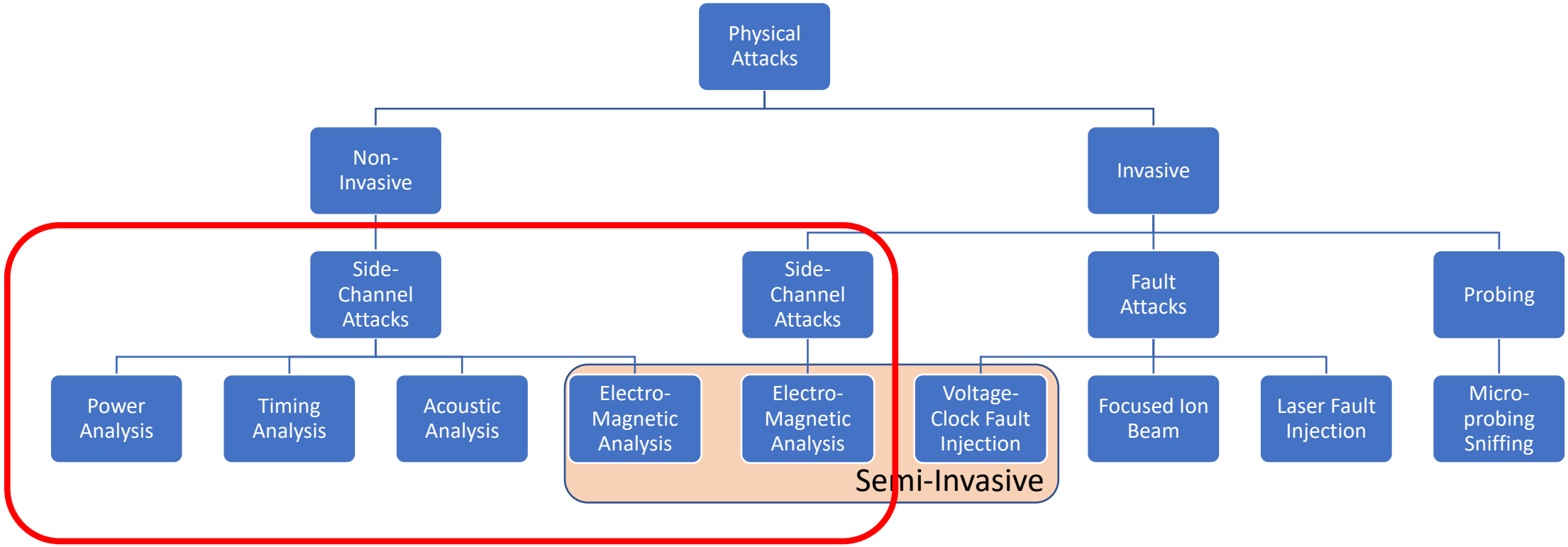
- A good understanding of a (worst case) threat model
- Hardware implementation (and its interaction with software layers)
- Software Implementation (and its interaction with the hardware)

Physical attacks represent a major threat to the security of information and to the security of the devices themselves. Why?

- Relatively low cost
- Realistic (and some times surprisingly short) execution time

Clearly, the definition of security of a device cannot be limited to the guarantees provided by the cryptographic algorithms only, but it strongly depends on how they have been implemented.

Taxonomy of Physical Attacks

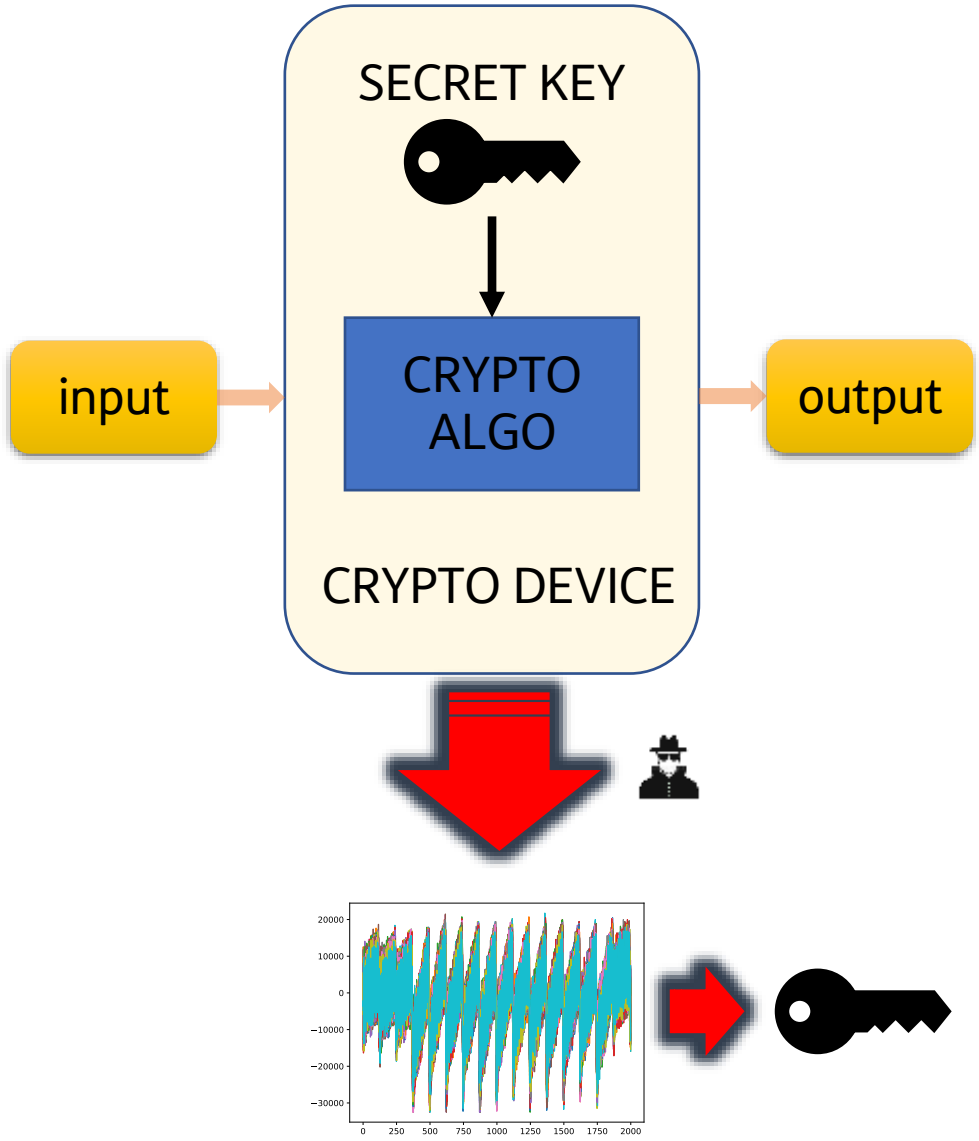


Outline

1. Physical Attacks
2. **Side-Channel Analysis**
3. Anatomy of a Trace
4. Advanced Encryption Standard
5. Detection and Analysis
6. CPA Attack
7. Countermeasures

Side-Channel Analysis

- Attacks based on **Side-Channel Analysis (SCA)** are a powerful class of **passive physical attacks** that exploit the dependency between **(measureable) physical emissions** and manipulated data processed within an electronic device.
- Typical physical emissions exploited for SCA:
  - Power consumption [KJB99]
  - Electro-magnetic emission [QS01]
  - Execution time [Koc96]
  - Acoustic emission [GST14]
  - Light emission [VIL95]
- They have been introduced in literature in 1996 by Paul Kocher [Koc96], but they were already exploited for military purposes (e.g., Tempest attack).
- Their importance has become critical even in the design of (and not limited to) new cryptographic algorithm, as well in the regards of their physical implementation.



SCA: Power Analysis

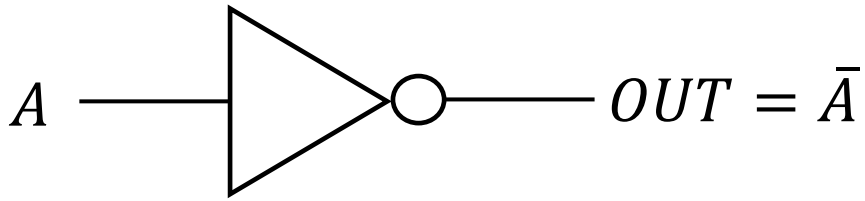
One of the most exploited side-channel is the **power consumtpion** in **CMOS circuits**:

$$P_{total}(data, t) = P_{phy\ noise}(t) + P_{algo\ noise}(t) + P_{data}(data, t)$$

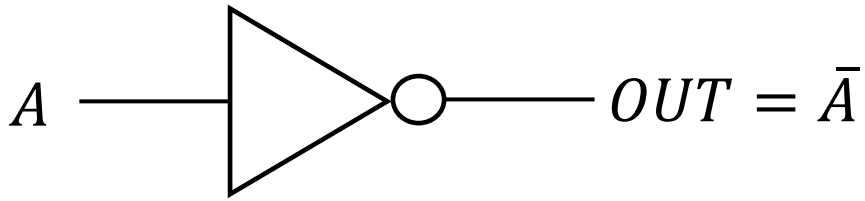
- $P_{phy\ noise}(t)$ is the physical noise generated by the device and the measuring setup.
- $P_{algo\ noise}(t)$ is the algorithmic noise, generated by non-targeted operations.
- $P_{data}(data, t)$ it is the data-dependent power consumption which is the most important component for our attack&analysis to recover and exfiltrate information

Let's see a simple side-channel analysis excercise on the simplest CMOS gate: the **CMOS inverter**.

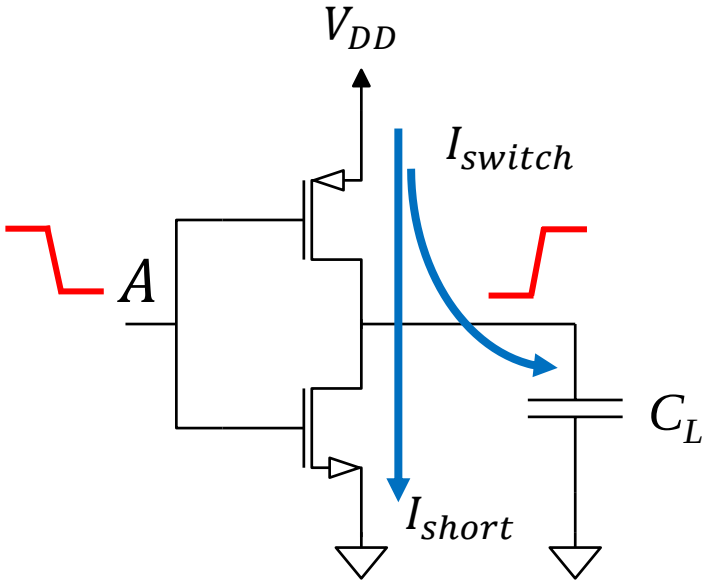
SCA: Power Analysis



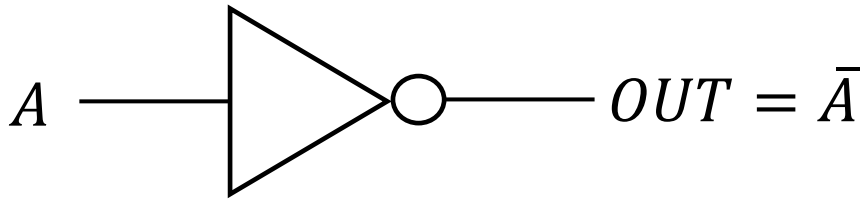
SCA: Power Analysis



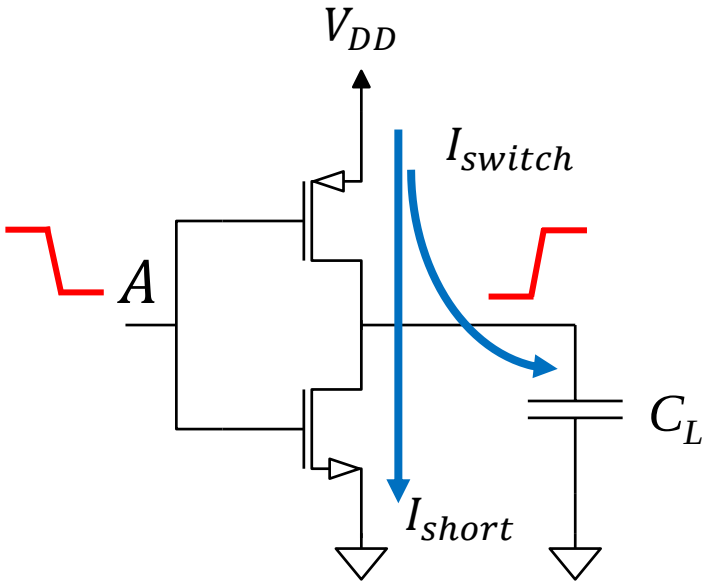
$OUT: 0 \rightarrow 1$



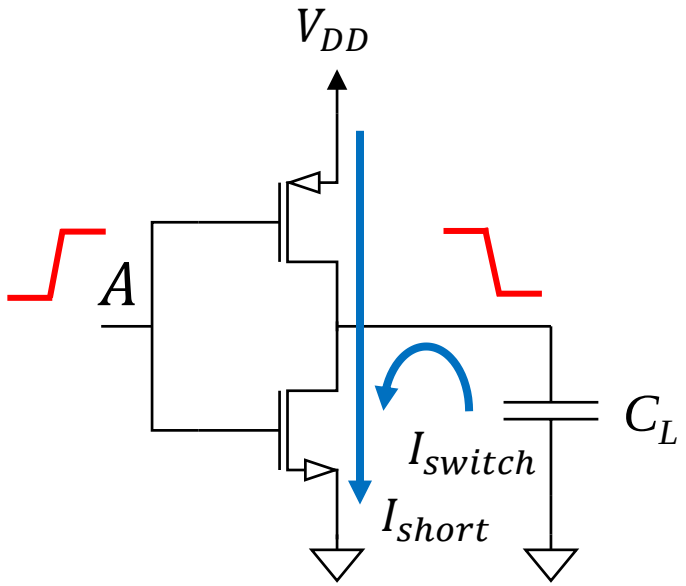
SCA: Power Analysis



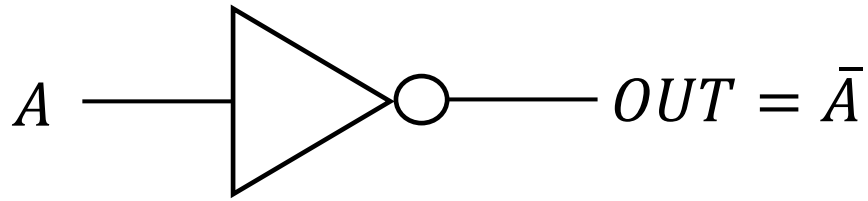
OUT: 0 → 1



OUT: 1 → 0

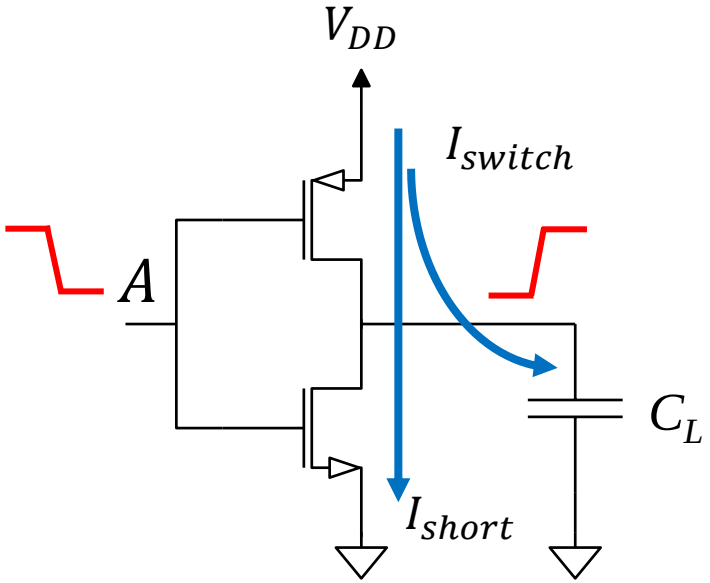


SCA: Power Analysis

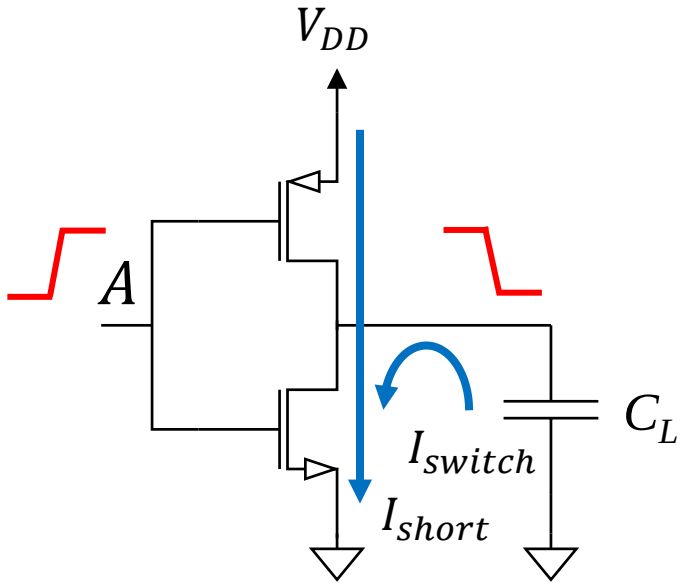


OUT Transition	Measured current on V_{DD}
$0 \rightarrow 1$	$I_{switch} + I_{short}$
$1 \rightarrow 0$	I_{short}

$OUT: 0 \rightarrow 1$



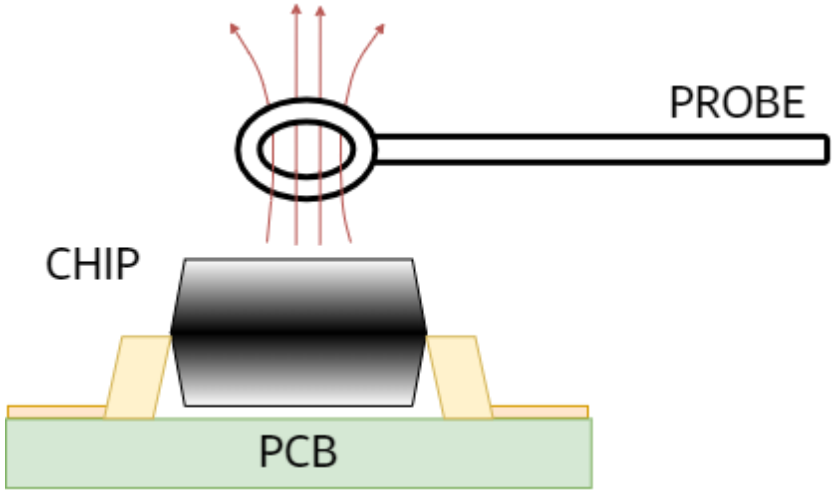
$OUT: 1 \rightarrow 0$

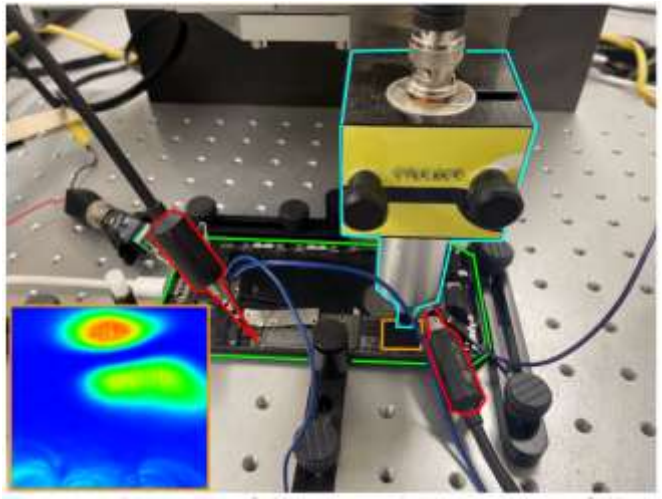


SCA: Electro-Magnetic Analysis

Whenever the logic state of an internal signal of a device changes, an electro-magnetic signal is emitted and it can be measured in the proximity of the chip.

In general, such transitions generate measurable variations of the eletromagnetic field, that allows an attacker to exfiltrate information.





From [HA22] G.Haas, A. Aysu, 'Apple vs. EMA: Electromagnetic Side channel Attacks on Apple CoreCrypto' on an Apple A10 SoC (in orange), mounted on an iPhone 7 (in green).

Compared to the power analysis, the **electro-magnetic analysis** has a **locality** feature and, in general, it is possibly to focus the analysis on *interesting* and/or *specific* parts of the device under test/attack. This usually yields to an **electro-magnetic cartography** of the device.

SCA: Timing Analysis

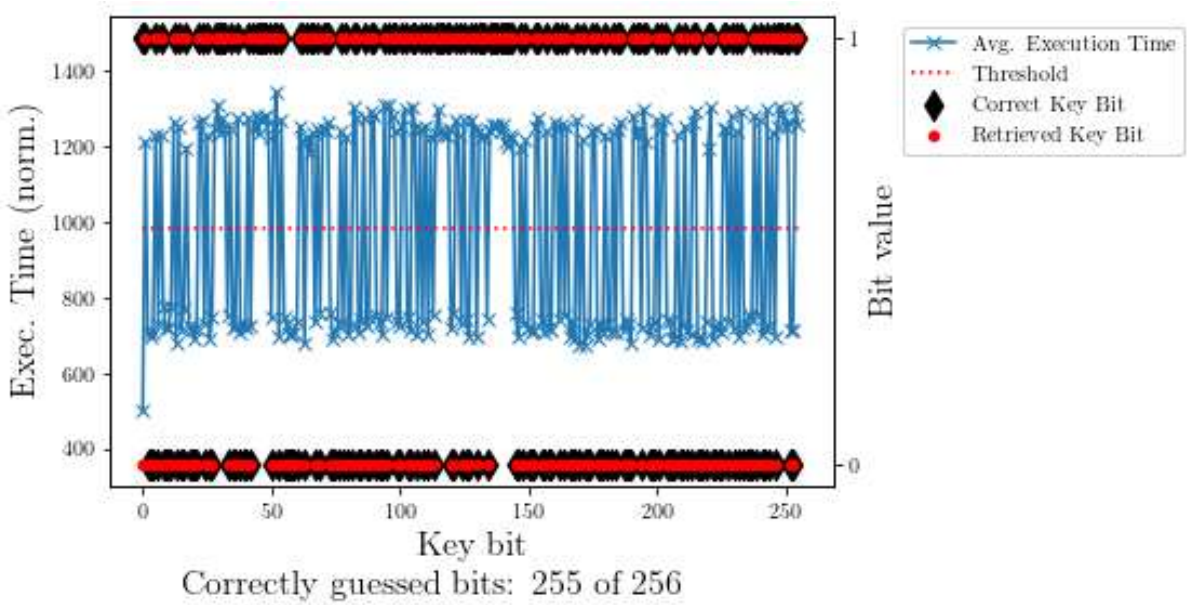
There exist critical vulnerabilities related to the instruction flow and/or to the execution time. This is possible whenever:

- Operation flow depends on data
- Execution time depends on data

A classical example is the naive implementation of the **modular exponentiation** algorithm, which is one of the core operation within RSA.

Algorithm: Modular Exponentiation Square&Multiply
Input: m,e k-bit integers, N k-bit prime
Ouput: m^e mod N

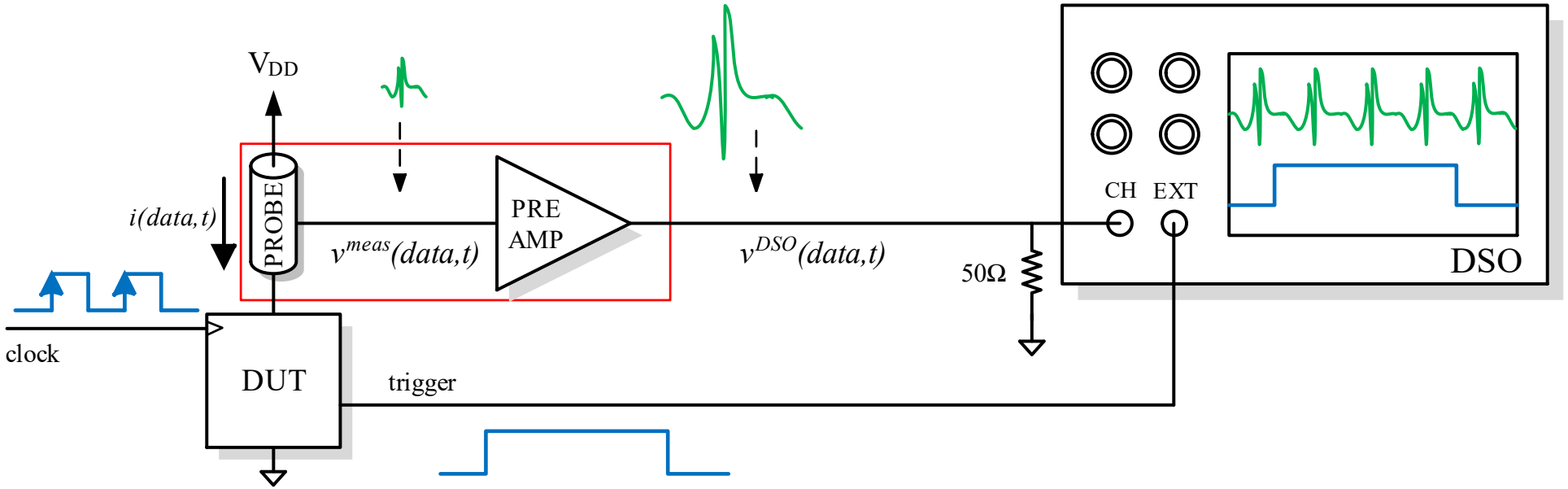
```
1  e_binary = binary(e,k)
2  tmp=1
3  for i in 0 to k-1 do
4      tmp = tmp^2 mod N
5      if e_binary[i] = 1 then
6          tmp = (tmp × m) mod N
7      end
8  done
9  return tmp;
```



Example of a timing analysis attack on a Python implementation of the Square&Multiply algorithm running on an Intel i7 CPU.

Classical SCA Measurement Setup

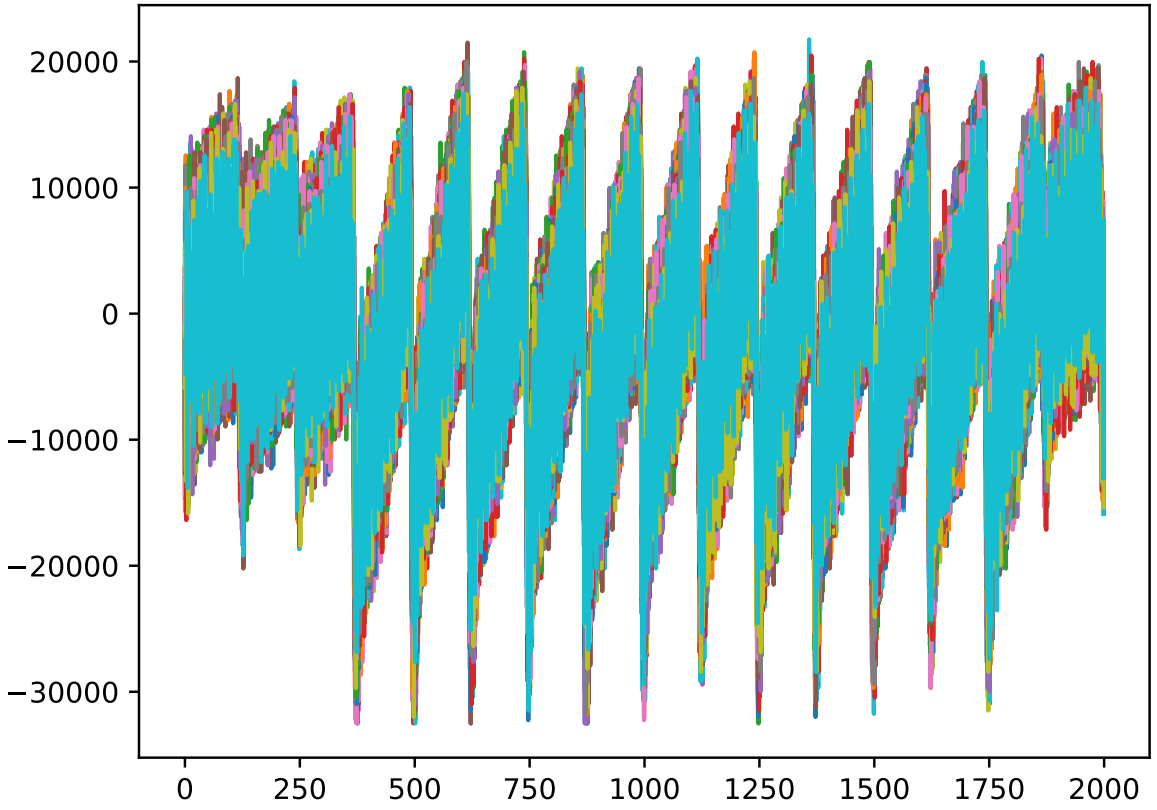
The measurement setup is system that is able to collect and store side-channel emissions [BUS21].



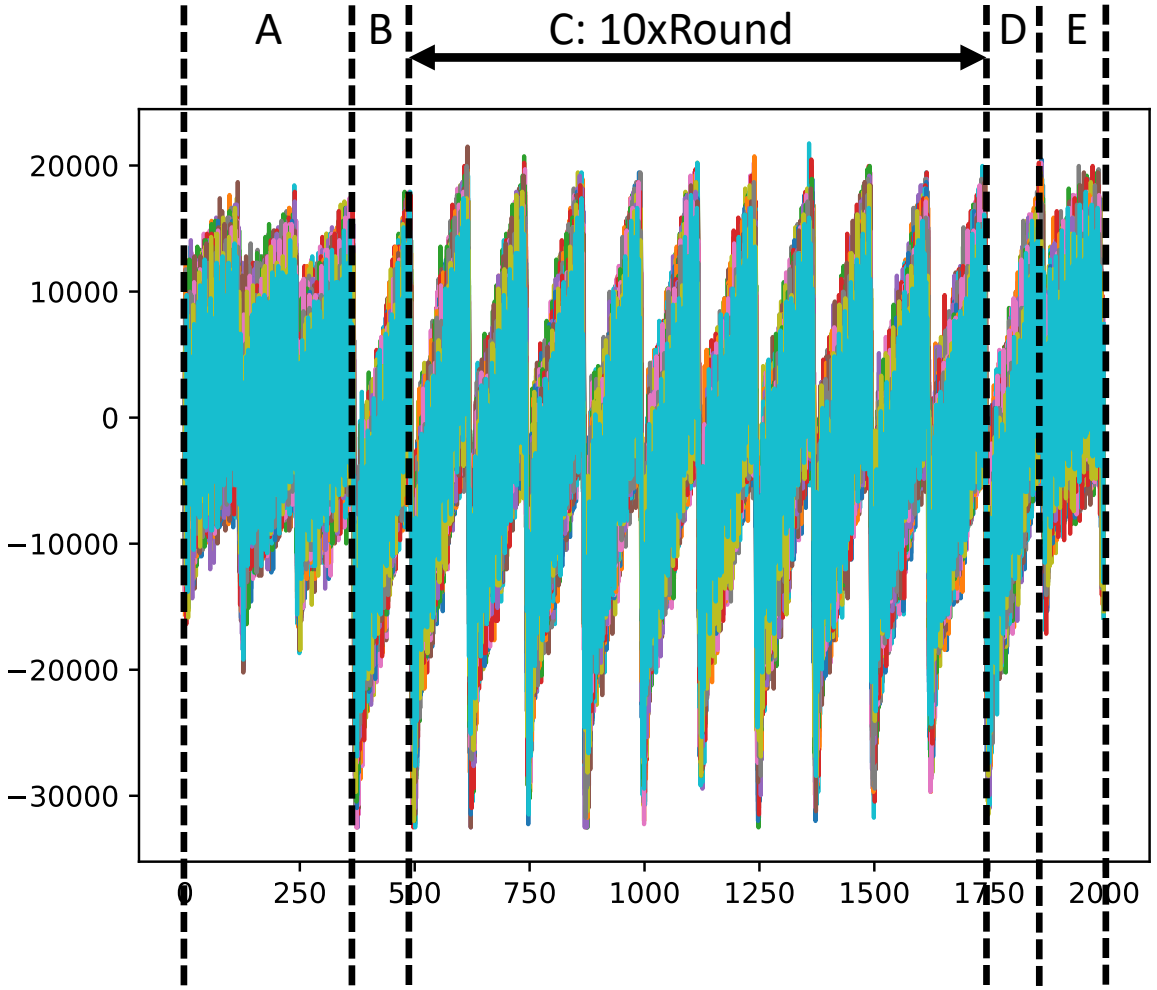
Outline

1. Physical Attacks
2. Side-Channel Analysis
3. **Anatomy of a Trace**
4. Advanced Encryption Standard
5. Detection and Analysis
6. CPA Attack
7. Countermeasures

Anatomy of a Trace



Anatomy of a Trace



Visual Inspection

- A: Receiving input from the controller
- B: plaintext loading in the core AES
- C: AES-128 (10-round)
- D: offload of the ciphertext
- E: other activity

Outline

1. Physical Attacks
2. Side-Channel Analysis
3. Anatomy of a Trace
4. **Advanced Encryption Standard**
5. Detection and Analysis
6. CPA Attack
7. Countermeasures

Advanced Encryption Standard (AES)

The **Advanced Encryption Standard** (**AES**) is a **block** cipher proposed by Vincent Rijmen and John Daemen, and standardised by the National Institute of Standard and Technology (NIST).

World-wide adopted as a building block for complex cryptographic systems, the AES operates on 16-byte data blocks, organized as 4x4-byte matrix, also called the state. It is based on 4 main operations (generically called round), iterated *N* times:

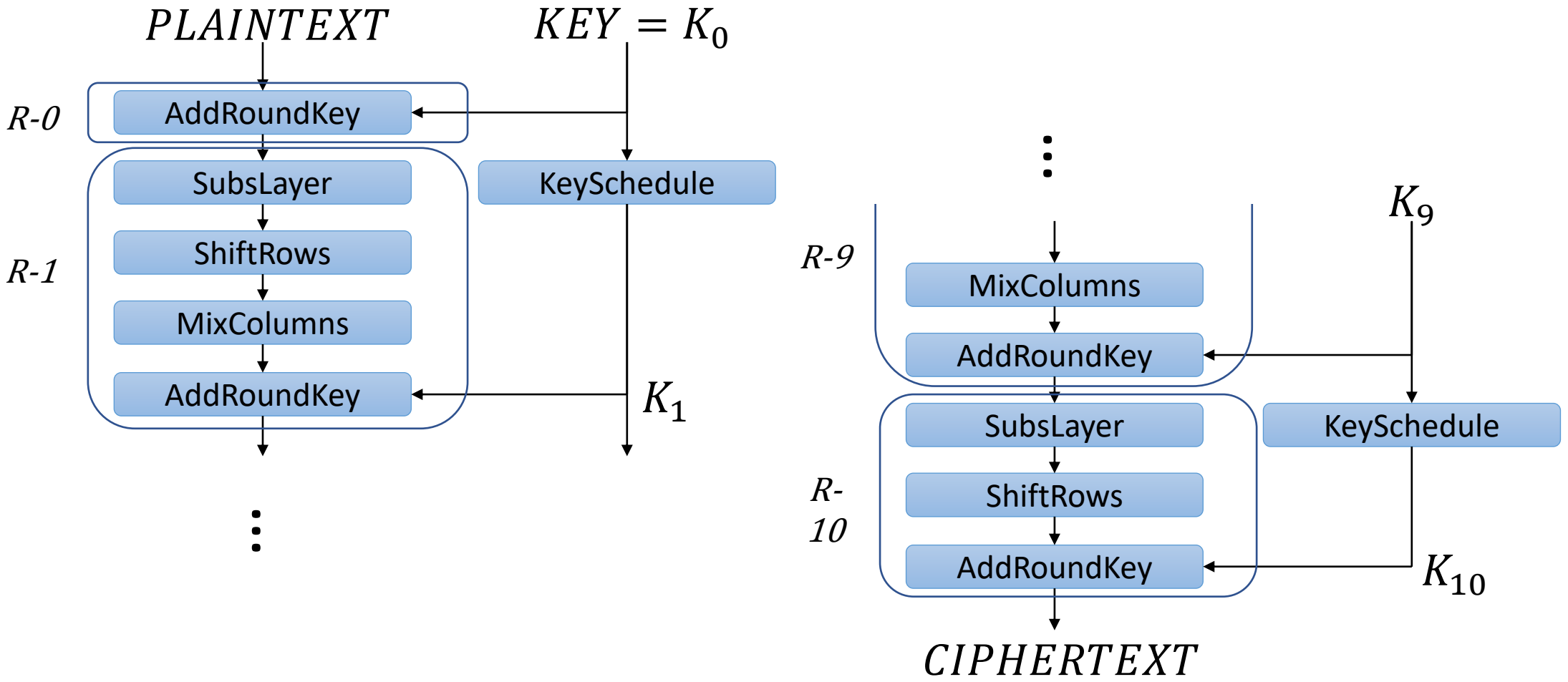
- **AddRoundKey**: binary XOR between the round key and the state
- **SubsByte**: non-linear substitution of the 16-byte, operating at byte level by means of a 8x8 S-box function (can be seen as a look-up table)
- **ShiftRows**: row rotation of the bytes of the state matrix
- **MixColumns**: linear transformation which operates on the columns of the state matrix

A **KeyScheduler** generates round keys starting from the input key.

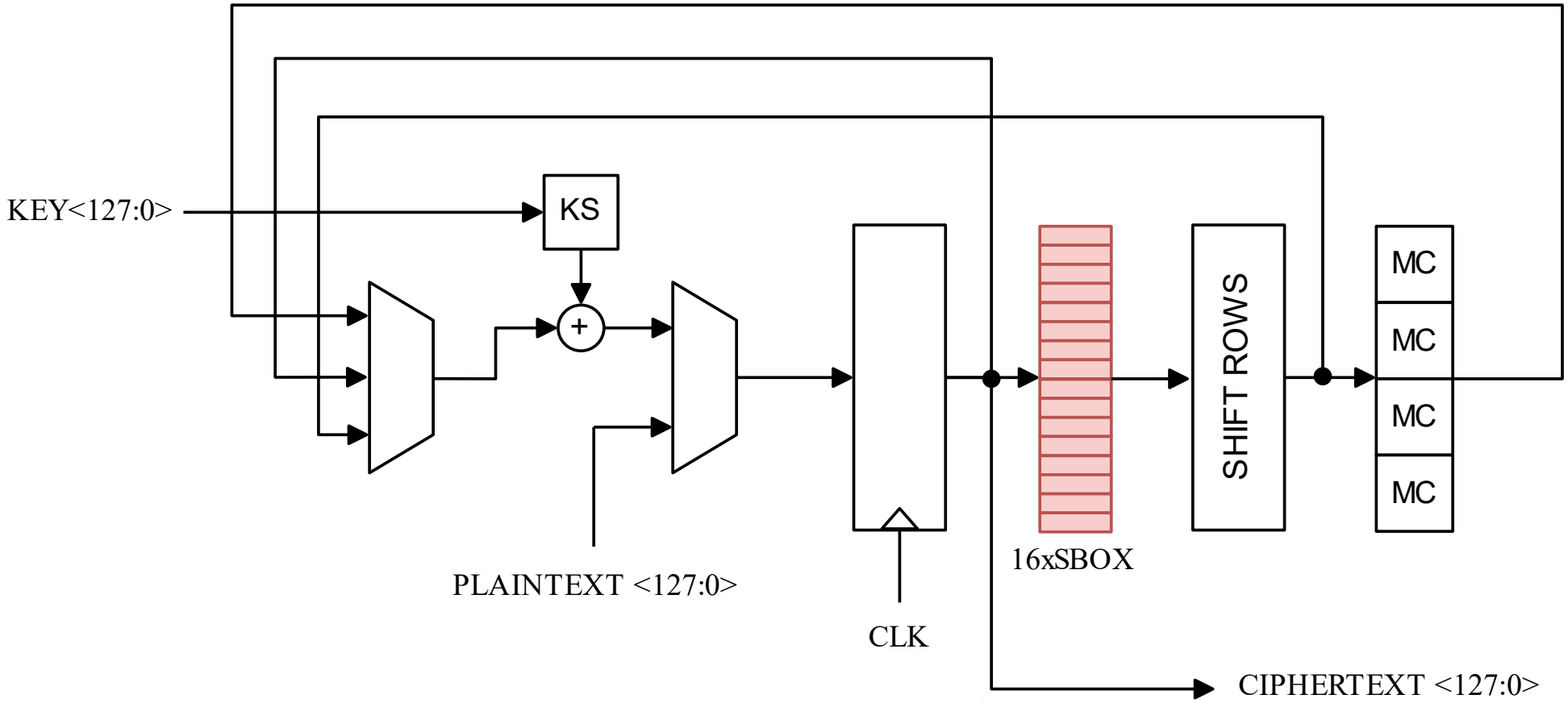
Depending on the key length, there are 3 variants of the AES block cipher, corresponding to 3 different level of security:

- AES-128: 128-bit key (*N=10* rounds)
- AES-192: 192-bit key (*N=12* rounds)
- AES-256: 256-bit key (*N=14* rounds)

Case Study:
AES GoogleVault



Case Study:
AES GoogleVault



Outline

1. Physical Attacks
2. Side-Channel Analysis
3. Anatomy of a Trace
4. Advanced Encryption Standard
5. **Detection and Analysis**
6. CPA Attack
7. Countermeasures

TVLA: Detection and Analysis

- The **detection** has the goal to verify if an implementation (either hardware or software) shows any sign of information leakage, without explicitly extracting the information itself and without verifying any concrete possibility of exploitation.

- A standard methodology that is widely used for detection is the **Test Vector Leakage Assessment** (TVLA) [CDG+13] based on the Student's T-test:

$$t = \frac{E[l_A] - E[l_B]}{\sqrt{\frac{Var[l_A]}{N_A} + \frac{Var[l_B]}{N_B}}}$$

- The TVLA is a pass/fail test, where the device under test is stimulated with **two classes** of input data (e.g., fixed plaintext vs random plaintext with a fixed and given key), and the aforementioend statistical difference is evaluated on the collected traces.

- If this «difference» exceeds (in absolute value) a certain threshold in a point (in time) of the trace, that point of the trace can be identified as a **possible leaky point**.

TVLA: Detection and Analysis

- Normally, the widely adopted threshold used in the TVLA is +/- 4.5, also called as the **critical value**.

- The choice of this particular value has the goal of minimizing **Type I Errors**, also called **False Positives**:
  - **False Positives** are points in time identified as leaky even if they are not leaky points.
  - The probability of not finding any leakage even if an information leakage exists within the impolementation is usually related to the **significance level α**, in turn related to the critical value.

- In order to keep the occurrence of False Positives low and within an acceptable level, the commonly adopted significance level α is 0,00001.

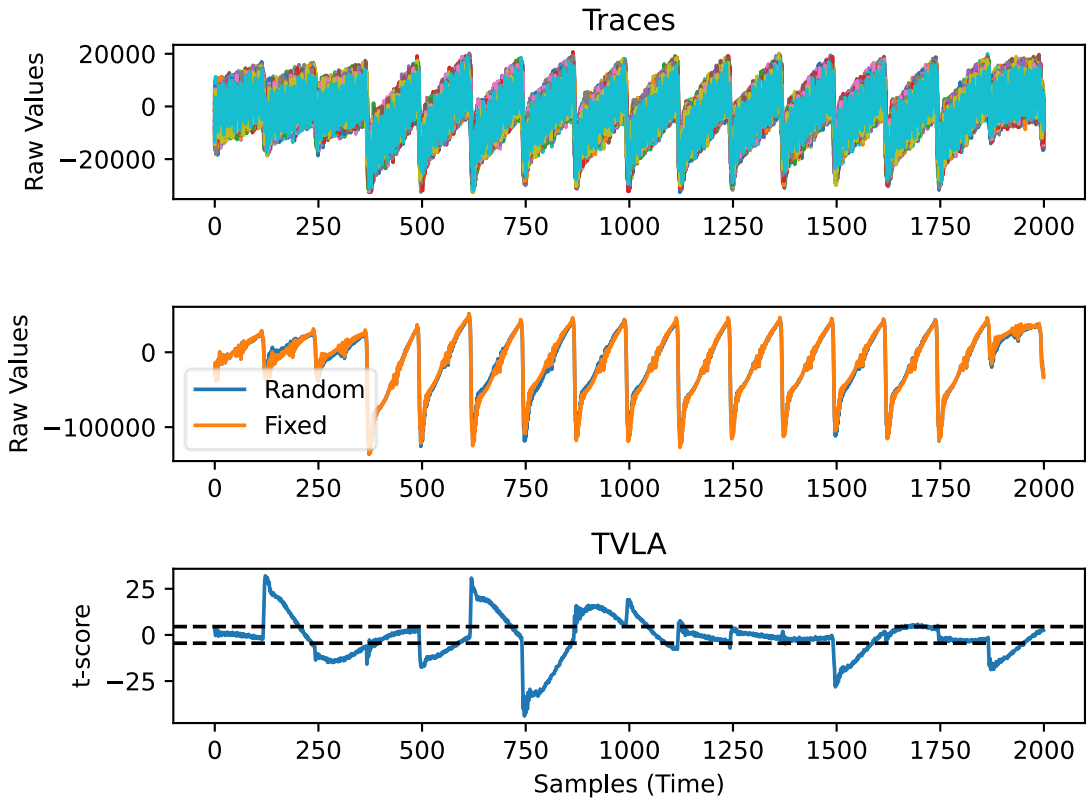
- This implies that if the device does not show any leakage, the probability that t value is in the range (-4.5,.+4.5) is 0,99999.

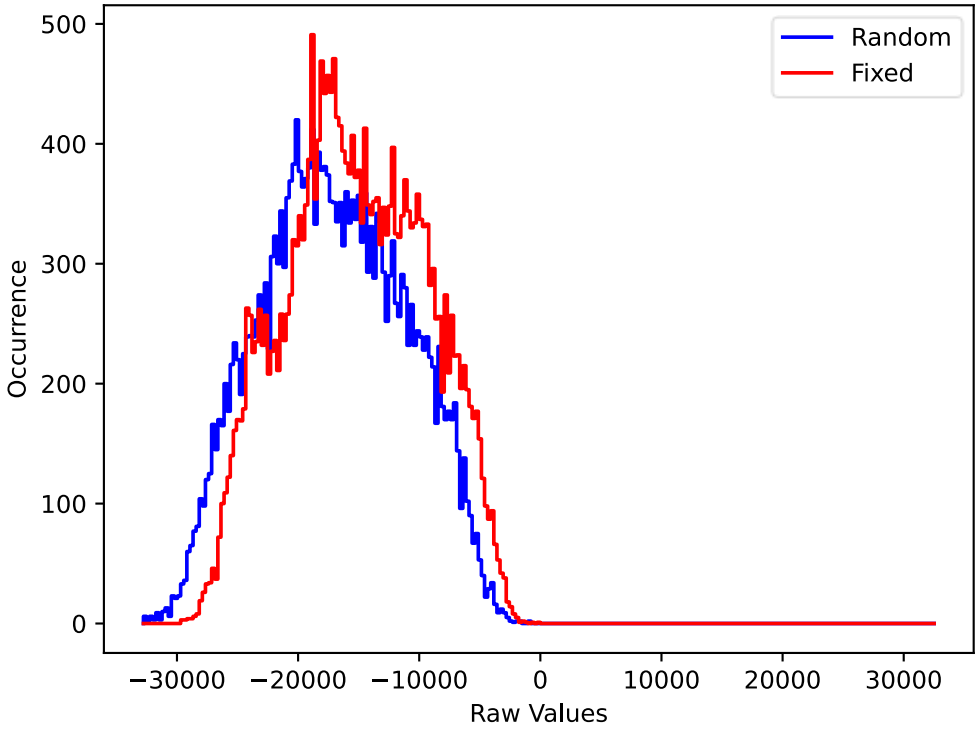
TVLA: Detection and Analysis

- TVLA experiment mode: **fixed vs random**
- Fixed 128-bit key: 0x00112233445566778899AABBCCDDEEFF
- **Class A** is a **fixed** 128-bit plaintext: 0x000102030405060708090A0B0C0D0E0F
- **Class B** is a **random** 128-bit plaintext
- Threshold value: +/-4.5 (α=0,00001)

TVLA: Detection and Analysis

TVLA T-test Fixed vs Random



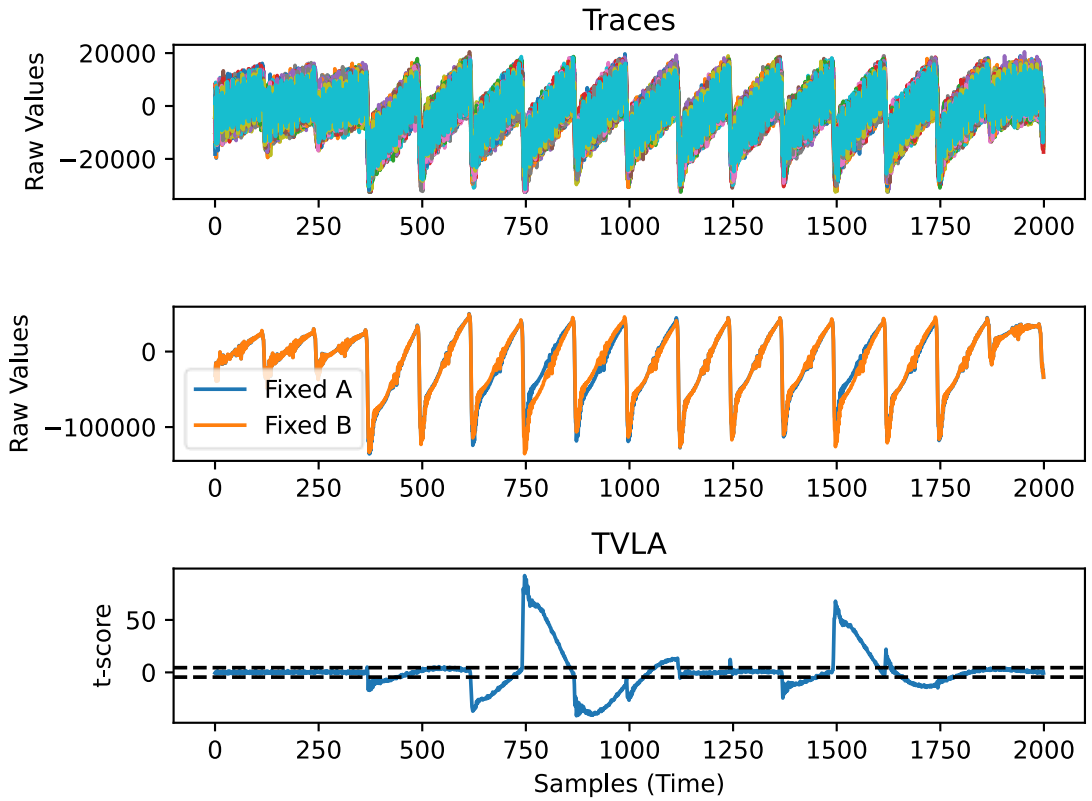


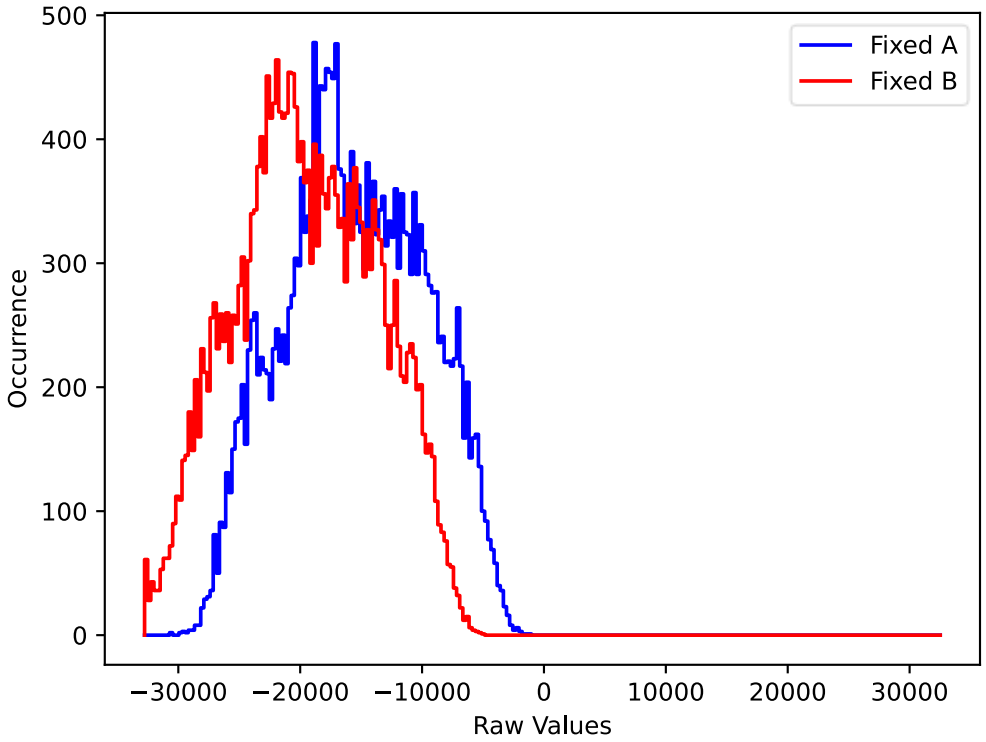
TVLA: Detection and Analysis

- TVLA experiment mode: **fixed vs fixed**
- Fixed 128-bit key: 0x00112233445566778899AABBCCDDEEFF
- **Class A** is a **fixed** 128-bit plaintext: 0x000102030405060708090A0B0C0D0E0F
- **Class B** is a **fixed** 128-bit plaintext: 0xFF0102030405060708090A0B0C0D0E0F
- Threshold value: +/-4.5 (α=0,00001)

TVLA: Detection and Analysis

TVLA con T-test Fixed vs Fixed

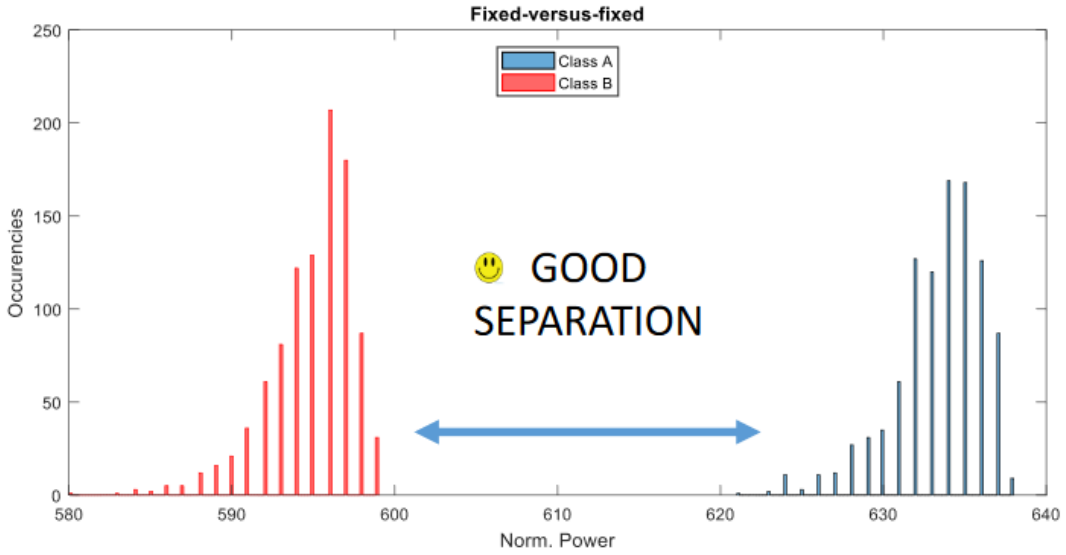


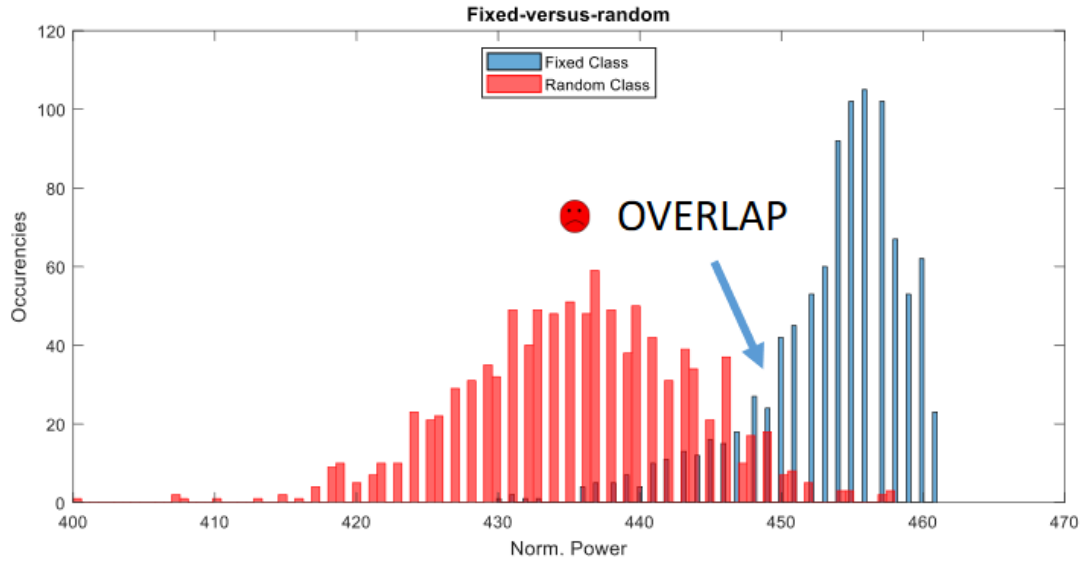


TVLA: Detection and Analysis

- In general, fixed-versus-fixed experiments shows *better* results and higher t values compares to fixed-versus-random, due to more **distinguishable distributions**.

- For an unprotected implementation, the obsderved noise in the fixed-versus-random is generally $2P_{phys\ noise} + P_{alg\ noise}$. In fixed-versus-fixed experiments, most of the algorithmic noise is canceled out (if no masking is applied, data within the implementation do not vary), and the total noise shrinks down to physical noise only, hence $2P_{phys\ noise}$.





TVLA on a software implementation of AES-128 software running on an Atmel ATMega328P microcontroller.

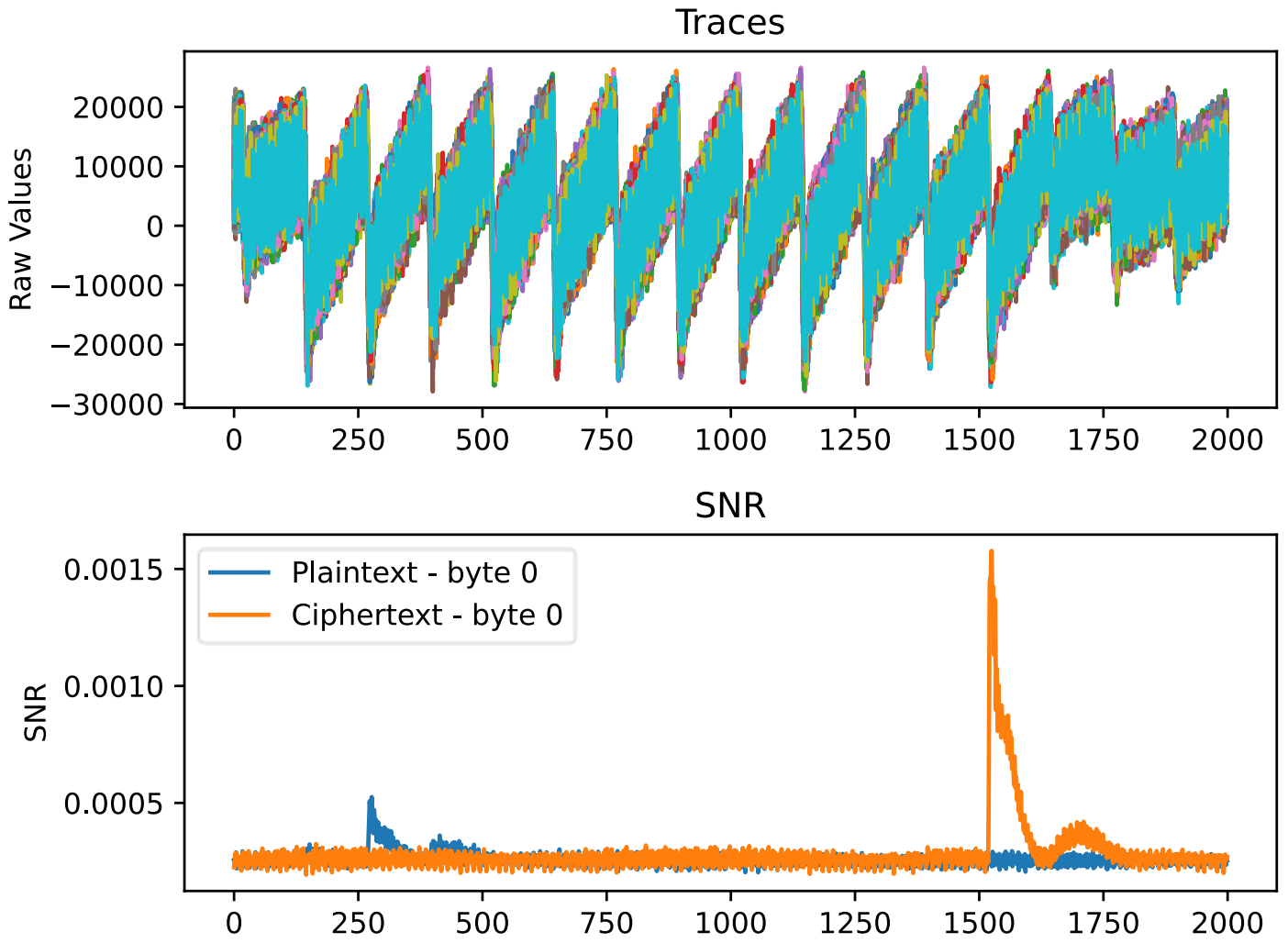
SNR: Detection and Analysis

- Another analysis methodology is based on the evaluation of the **signal-to-noise ratio** (SNR), proposed by S.Mangard in 2004 [Man04].
- The definition of the SNR in the context of the SCA is quite similar to the one adopted in electronics and telecommunication regarding the **quality of a signal**.

$$SNR = \frac{Var[E_i[l_i]]}{E[Var_i[l_i]]}$$

- In this case, the «power» of the signal has to be inteded as the deterministic part of the side-channel emission (ideally noise-free) that is inteded to be used to recover information, for example 8-bit of sub-key, and noise as everything that is not targeted by the analysis/attack (physical and algorithmic noise).
- The **classification** of the traces (note the *i* subscript) can be done following different approaches, depending on the evaluation needs (e.g., target variable value, value of the consumption model of the target variable).
- In the case of **grey/white-box model** of the implementation, it is possible to use this methodology to identify *interesting* points of a side-channel trace.

SNR: Detection and Analysis



Outline

1. Physical Attacks
2. Side-Channel Analysis
3. Anatomy of a Trace
4. Advanced Encryption Standard
5. Detection and Analysis
6. **CPA Attack**
7. Countermeasures

Correlation Power Analysis

The **correlation power analysis** (CPA) [BCO04] is an analysis methodology that can be also used to recover the value of a secret variable by means of statistical comparison between an hypothetical model of emission and the actual emission collected on a device.

The CPA is based on the esitimation of the **Pearson's correlation coefficient**, a statistical tool widely used to evaluate the linear corrleation between to random variables.

$$\rho[t] = \frac{\sigma_{LH}[t]}{\sigma_L[t] \cdot \sigma_H[t]} = \frac{\sum_{i=0}^{N-1}[(l_i[t] - \overline{l[t]}) \cdot (h_i - \bar{h})]}{\sqrt{\sum_{i=0}^{N-1}(l_i[t] - \overline{l[t]})^2 \cdot \sum_{i=0}^{N-1}(h_i - \bar{h})^2}} \quad , \qquad -1 \leq \rho \leq 1$$

The random variable *L* represents the *N* side-channel traces at each time instant t. The random variable H represents the hypothetical leakage value following a certain model of the *N* traces.

Correlation Power Analysis

To build a side-channel modelization *h* suitable for CPA we need three ingredients:

- Something we know (e.g. plaintext or ciphertext) that we can control, *x*
- Knowledge of the algorithm that is running on the device (and it sub-functions), *a(.)*
- A suitable physical model of the emission *f(.)*

$$h = f(a(k_{guess}, x))$$

For a physical model *f(.)* we mean the theoretical or experimental relation between an intermediate variable of the implementation and a feature (for example, the magnitude) of the corresponding physical emission. Classical models adopted for power and electro-magnetic analysis are:

- **Hamming Weight** (HW): sum of the number of 1s in a variabile
- **Hamming Distance** (HD): sum of the number of bits that differ in two variables of the same length
- **Estimated Average & Variance** of the physical emission, based on experimental observations in a controlled experiment

Correlation Power Analysis

In an attack scenario like this, we are interested in recovering 8-bit at a time, so we need to consider all 2^8=256 possible sub-keys (**dividi-et-impera**) for each input we provide to the device under attack. For each possible sub-key we need to compute the hypothetical value of the emission in the regards of the given input we provide, using all three ingredients we discussed so far.

Hence, the vector $h_i \in H, i = 0 \dots (N-1)$ is a matrix $\boldsymbol{H}$ of 256 vectors H_k, where $h_{k,i} \in H_k$. The computation (time-wise) of the Pearson's correlation coefficent becomes:

$$\rho_k[t] = \frac{\sigma_{LH_k}[t]}{\sigma_L[t] \cdot \sigma_{H_k}[t]} = \frac{\sum_{i=1}^{N}[(l_i[t] - \overline{l[t]}) \cdot (h_{k,i} - \overline{h_k})]}{\sqrt{\sum_{i=1}^{N}(l_i[t] - \overline{l[t]})^2 \cdot \sum_{i=1}^{N}(h_{k,i} - \overline{h_k})^2}}$$

The outcome of the CPA attack is the sub-key that maximizes the absolute value $\rho_k$among all possible sub-keys. In other words:

$$k_{guess} = \max_k\{|\rho_k|\}$$

Correlation Power Analysis

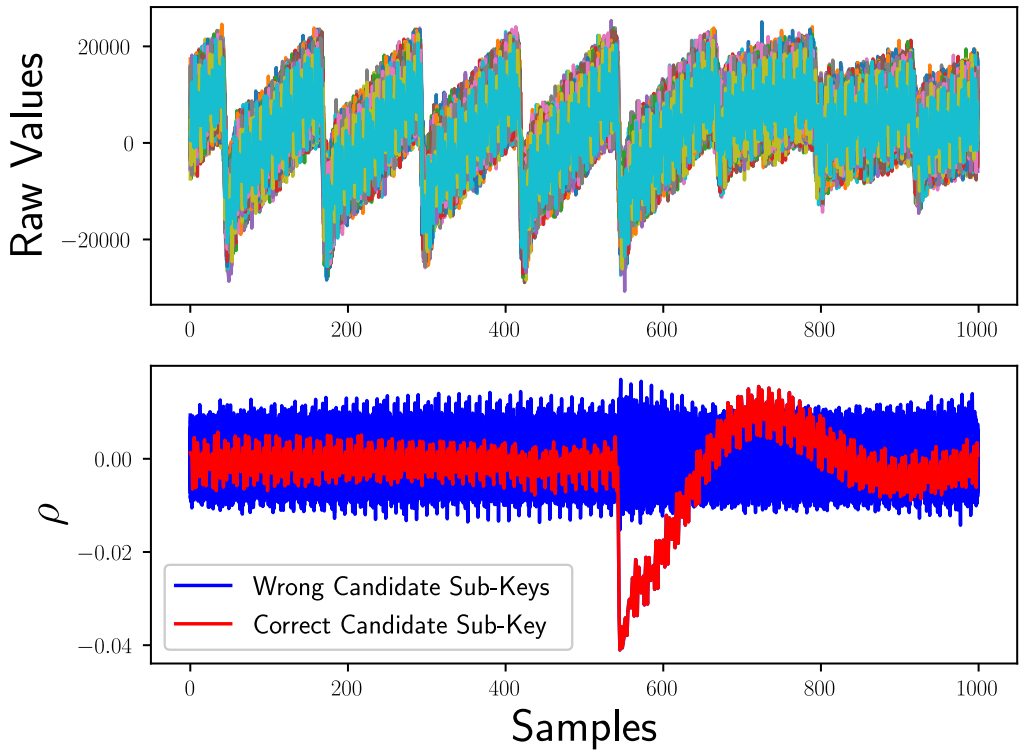
Attack scenario

- AES-128 core hardware implementation configured in encryption, running on FPGA
- Fixed 128-bit key (randomly generated at the beginning of the experiment)
- Core voltage: 1.0V
- Clock Frequency: 12MHz
- Target round: last round

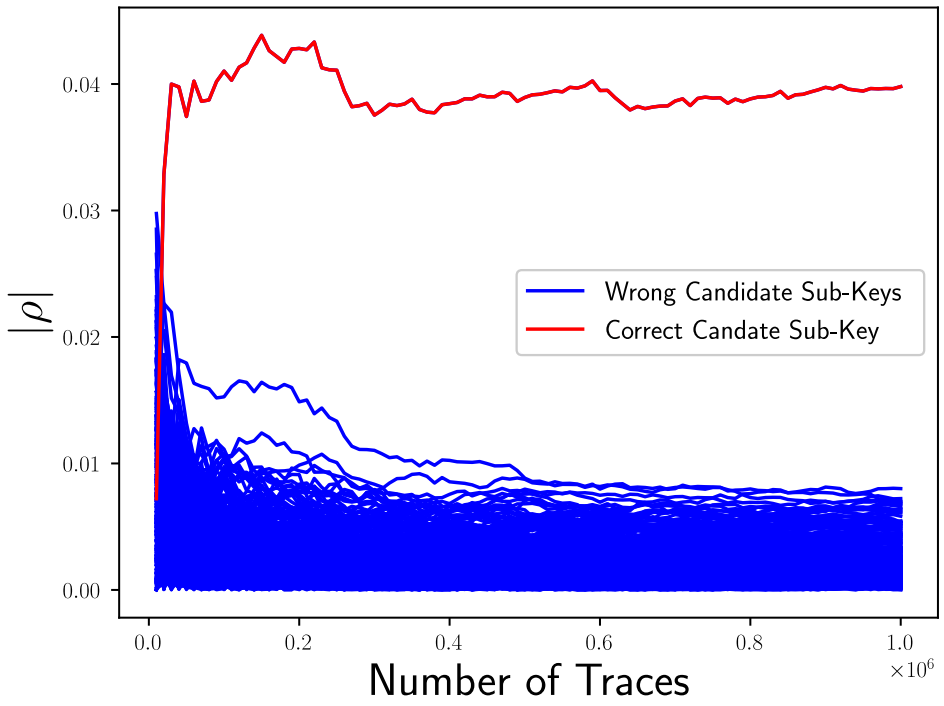
Time required on a standard laptop running Ubuntu 21:

- Collecting data: ~4 minutes
- Elaboration to extract the full 128-bit key: ~2minutes

Correlation Power Analysis

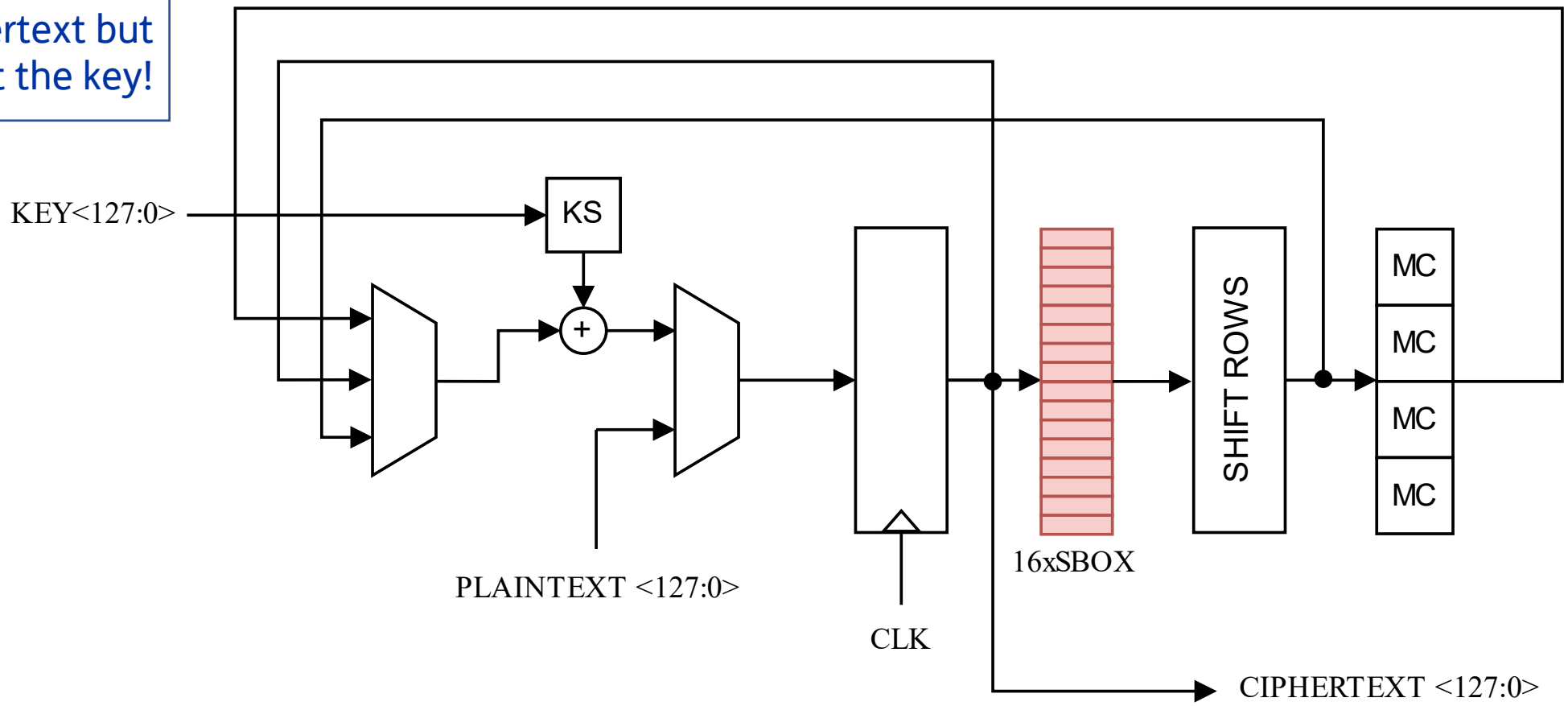


106 traces!



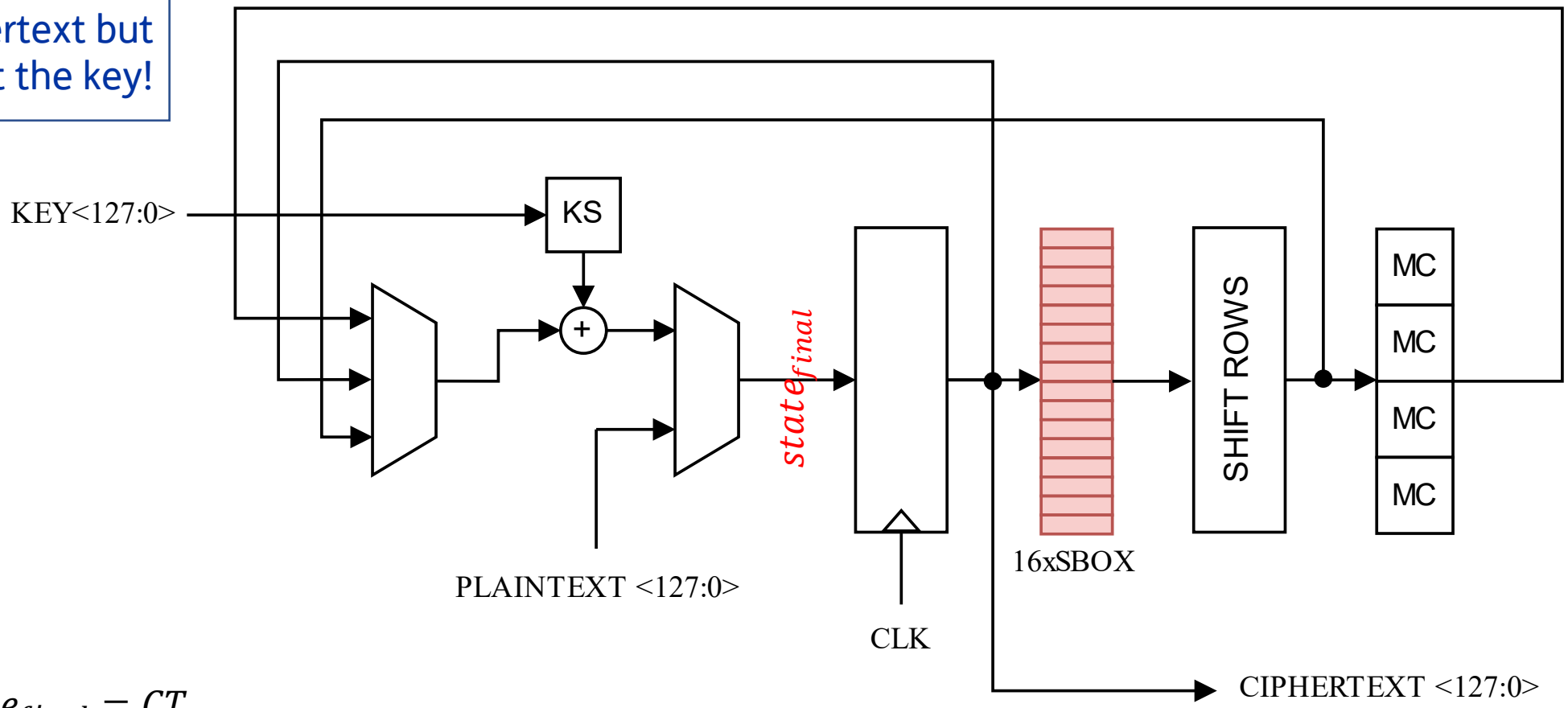
Correlation Power Analysis

We know the ciphertext but not the key!



Correlation Power Analysis

We know the ciphertext but not the key!

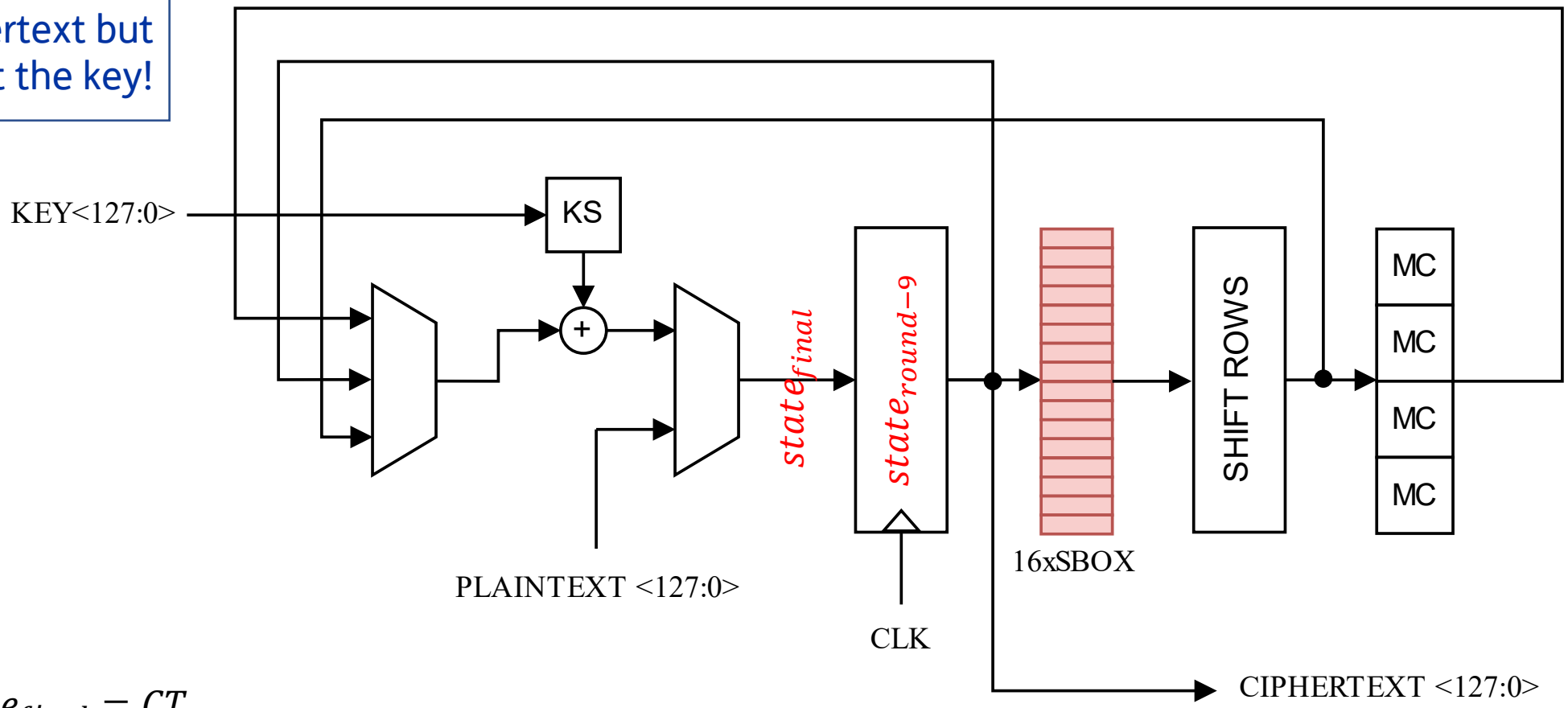


$$state_{final} = CT$$

$$state_{round-9} = SHIFTROWS^{-1}(SBOX^{-1}(CT \oplus K_{10}))$$

Correlation Power Analysis

> **!** We know the ciphertext but not the key!

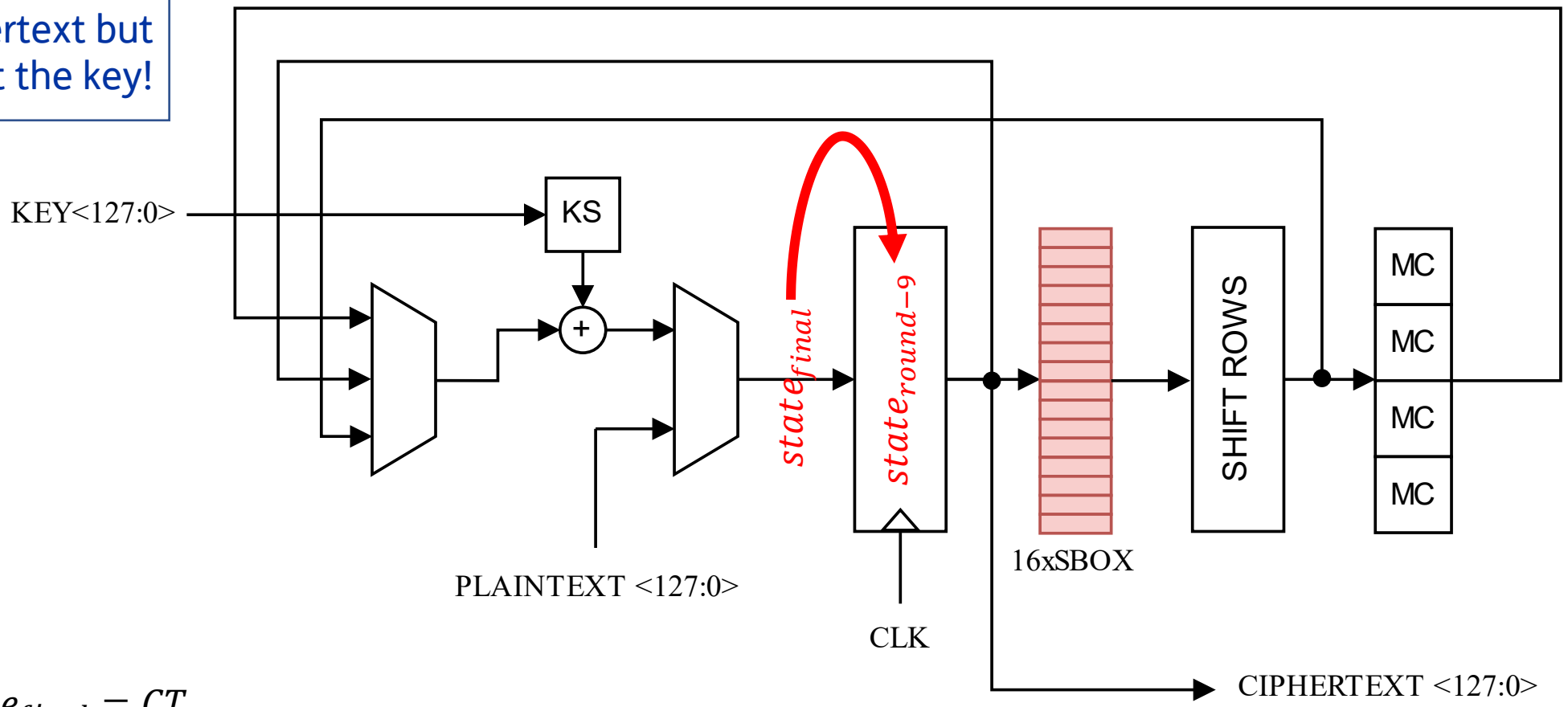


$$state_{final} = CT$$

$$state_{round-9} = SHIFTROWS^{-1}(SBOX^{-1}(CT \oplus K_{10}))$$

Correlation Power Analysis

We know the ciphertext but not the key!

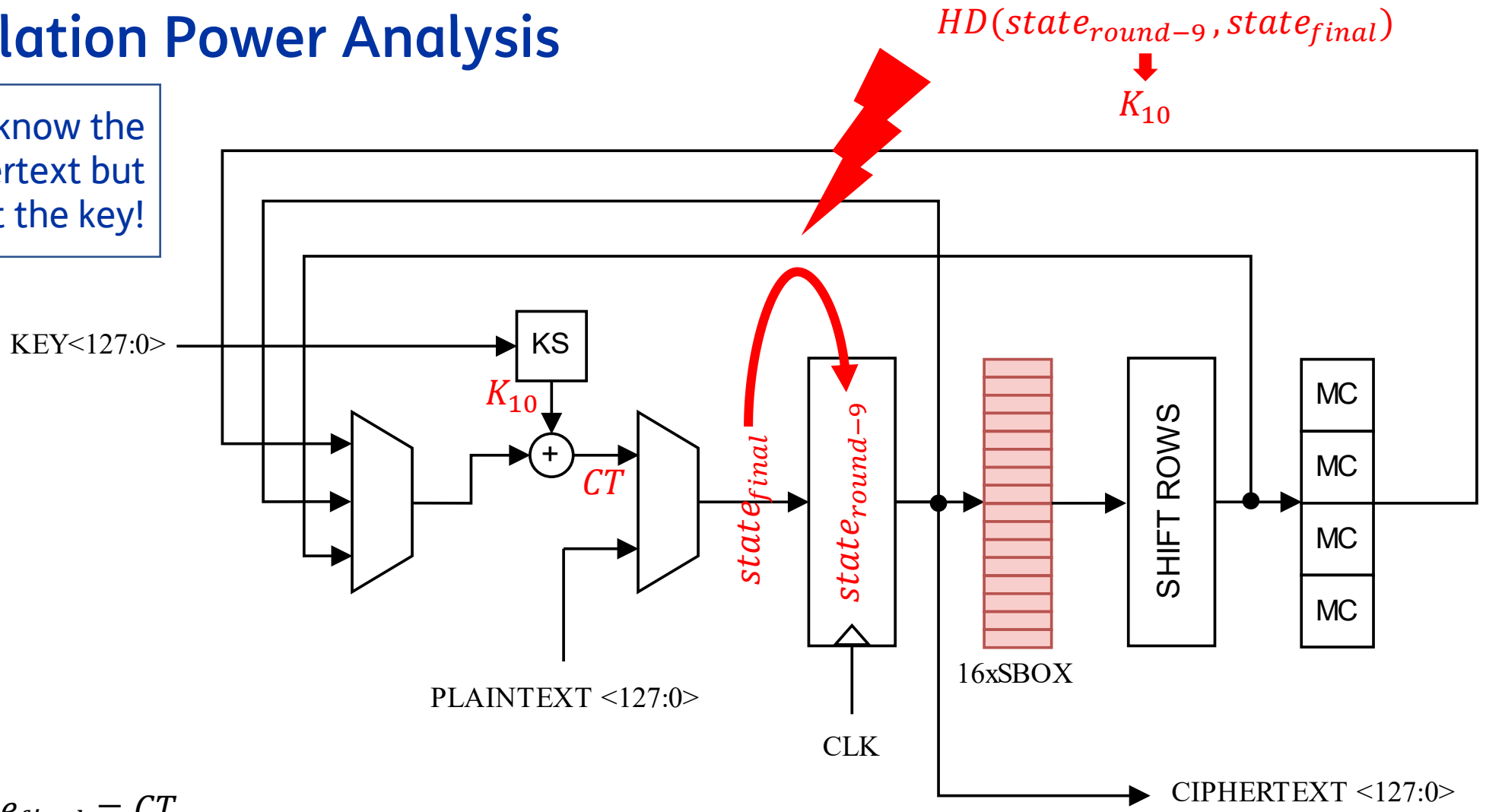


$$state_{final} = CT$$

$$state_{round-9} = SHIFTROWS^{-1}(SBOX^{-1}(CT \oplus K_{10}))$$

Correlation Power Analysis

> **!** We know the ciphertext but not the key!



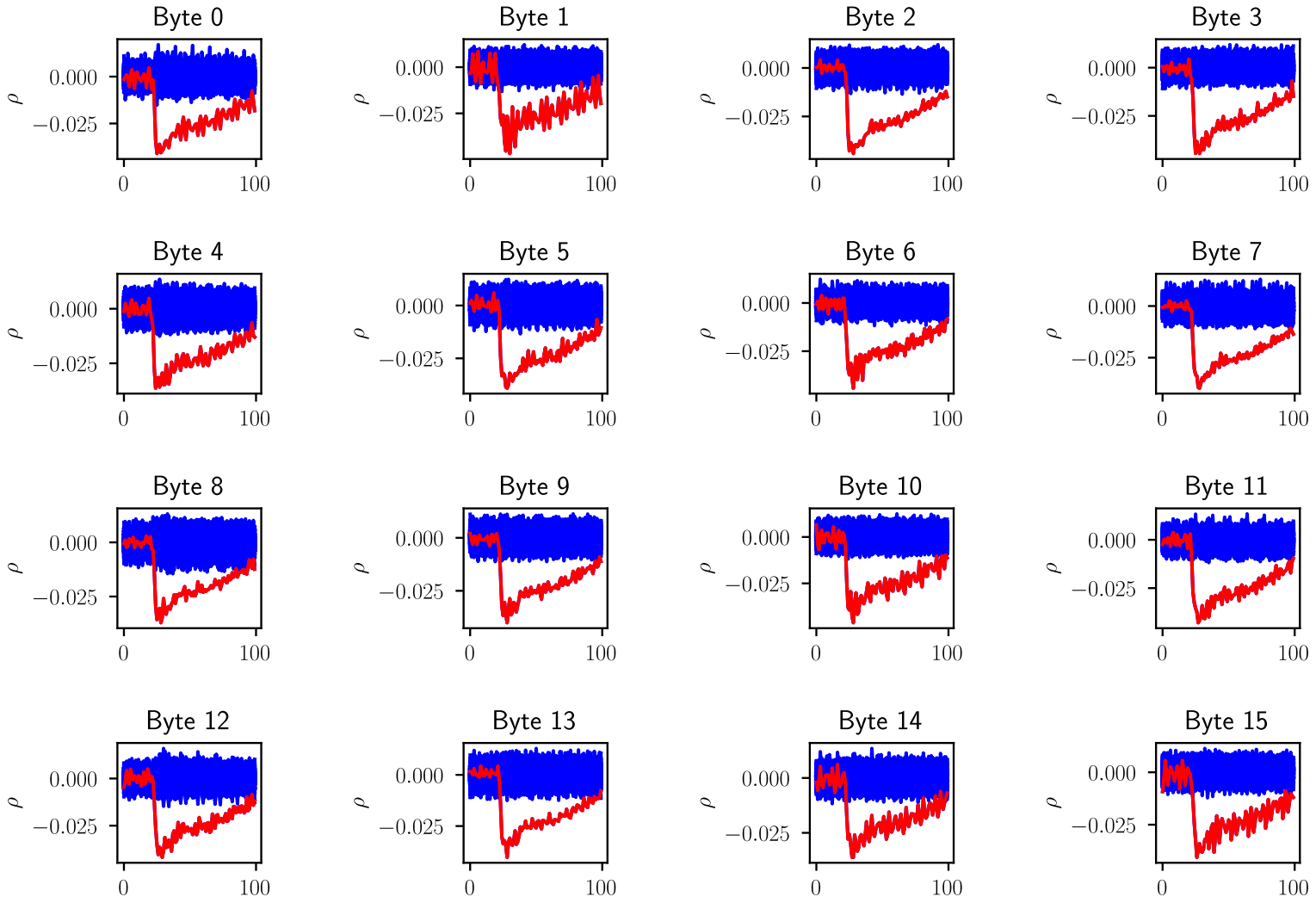
$$state_{final} = CT$$

$$state_{round-9} = SHIFTROWS^{-1}(SBOX^{-1}(CT \oplus K_{10}))$$

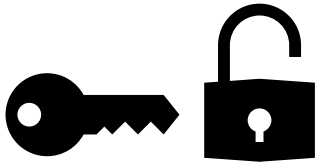
BACKWARD UNROLL OF THE KEY SCHEDULER

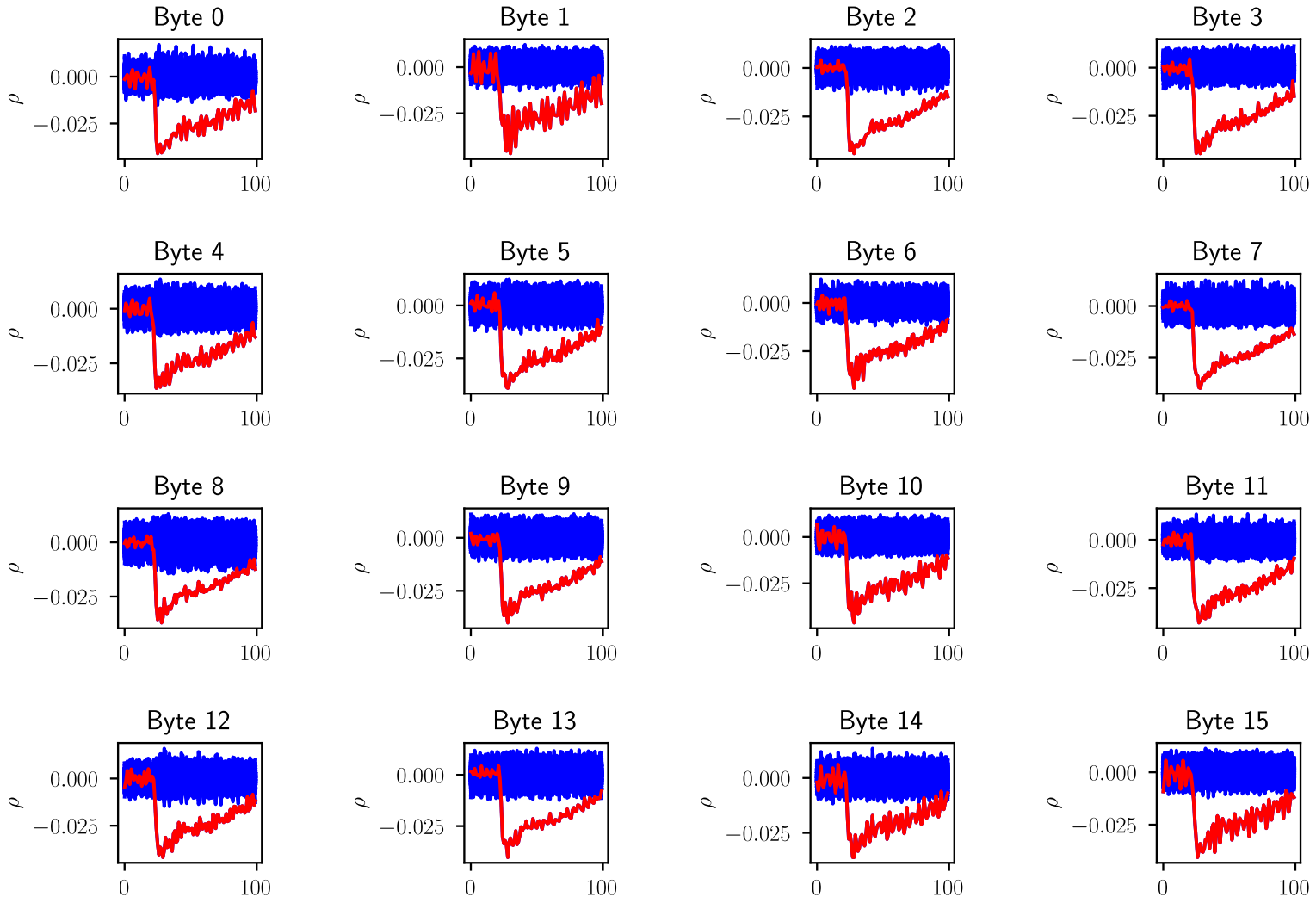
$$KS^{-1}(K_{10}) \rightarrow K_0 = KEY$$

Correlation Power Analysis

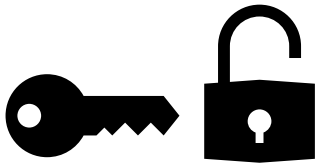


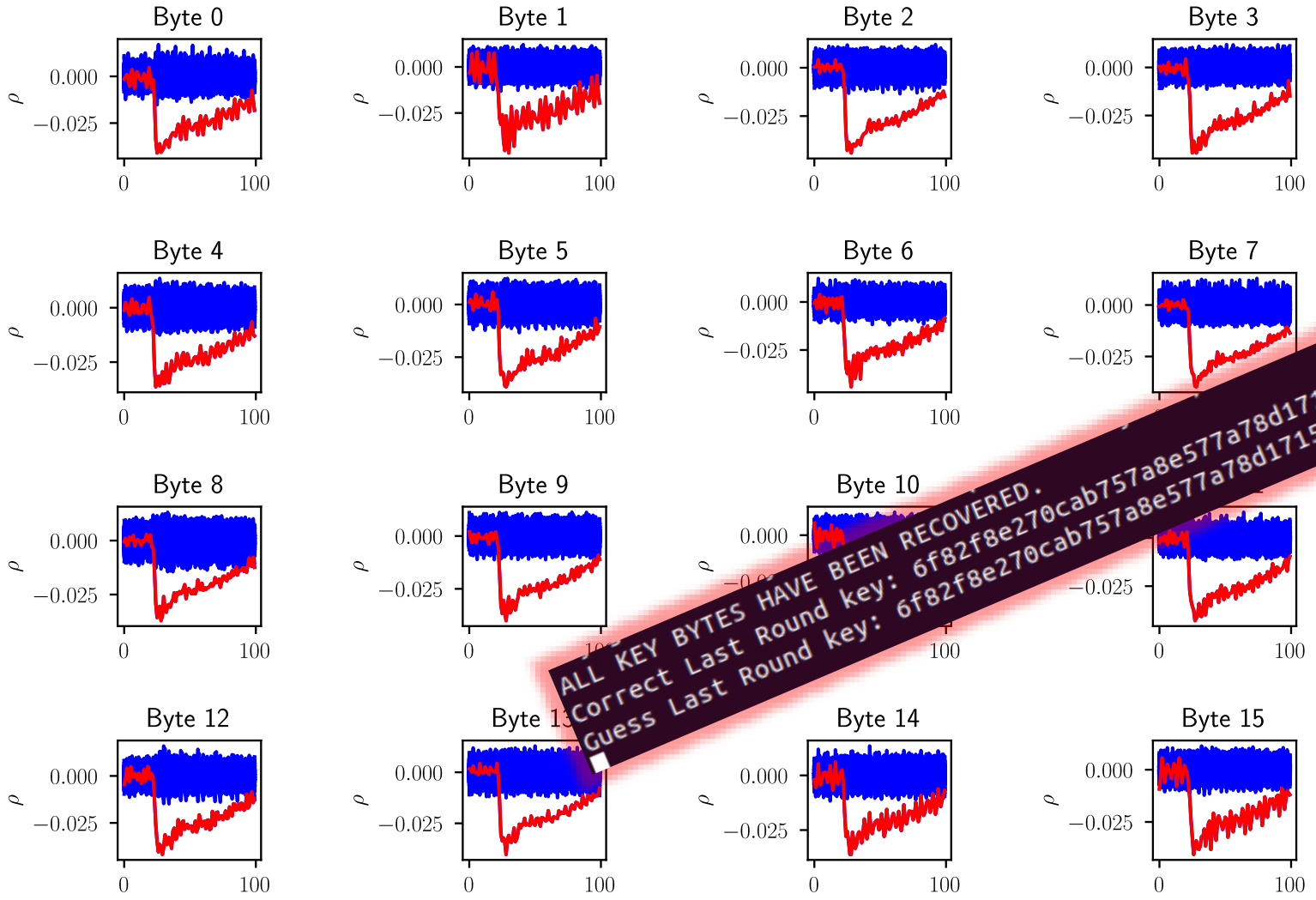
Correlation Power Analysis





Correlation Power Analysis





Outline

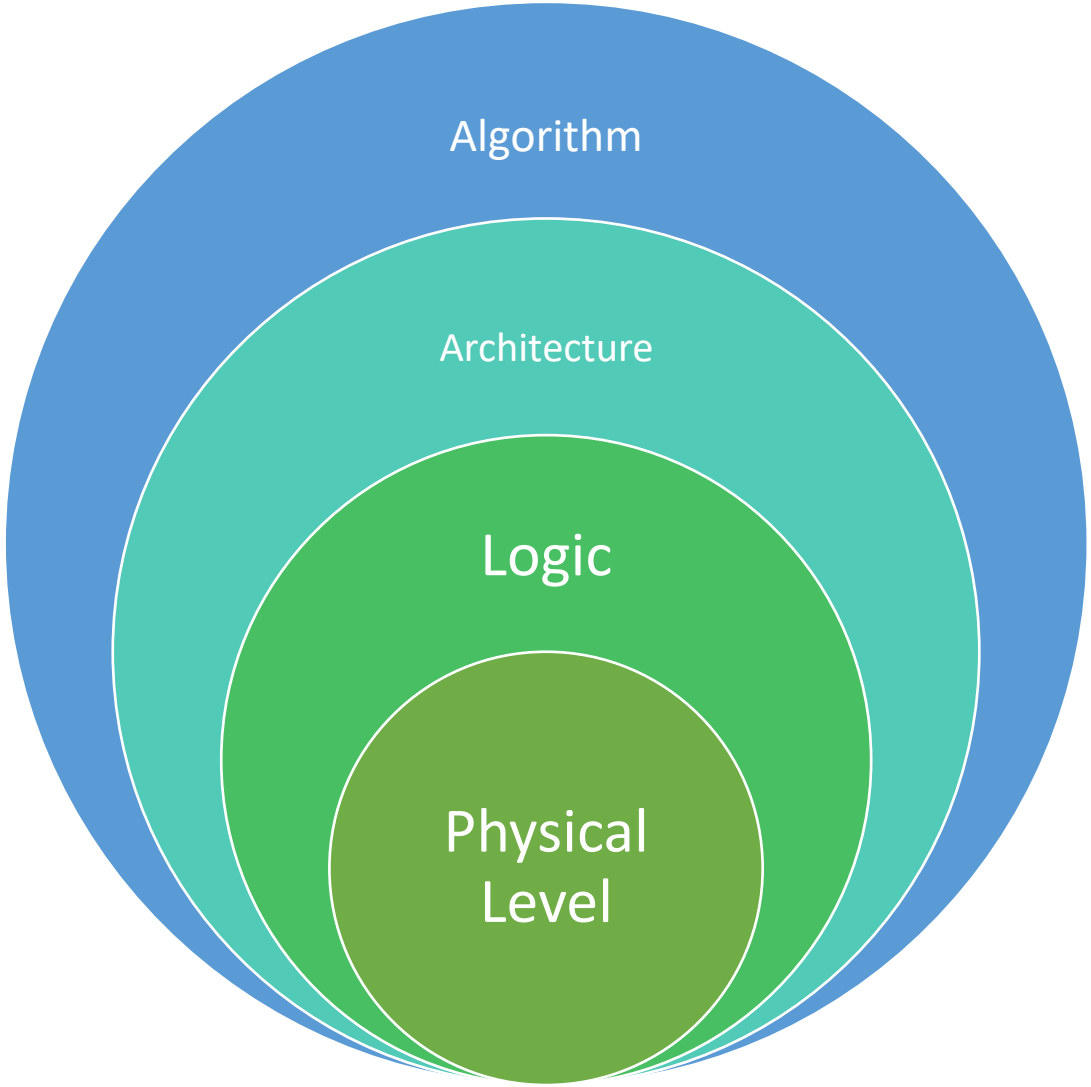
1. Physical Attacks
2. Side-Channel Analysis
3. Anatomy of a Trace
4. Advanced Encryption Standard
5. Detection and Analysis
6. CPA Attack
7. **Countermeasures**

Countermeasures

The presence of these physical vulnerabilities have led to the introduction of several approaches to their **mitigation** by means of **countermeasures**.

Countermeasures can be applied at different **level of abstractions**. Their effectiveness and implementation effort are tightly related with the level of abstraction.

In general, the lower the abstraction layer, the higher will be the cost and the effort for the implementation.



Countermeasures

Countermeasure can be divided in two main categories:

- Hiding
- Masking

They aim at **reducing the SNR**, and sometimes they are used in synergy and it is not trivial to distinguish them.

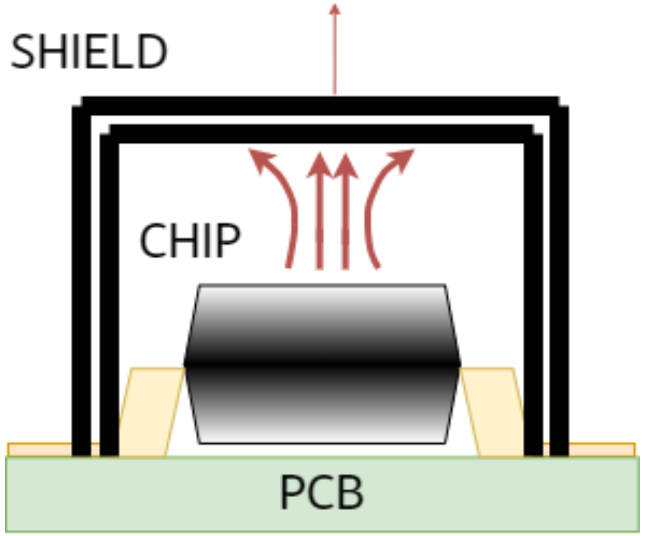
Hiding-based countermeasures aim at reducing the **power of the signal** (hence the numerator of the SNR).

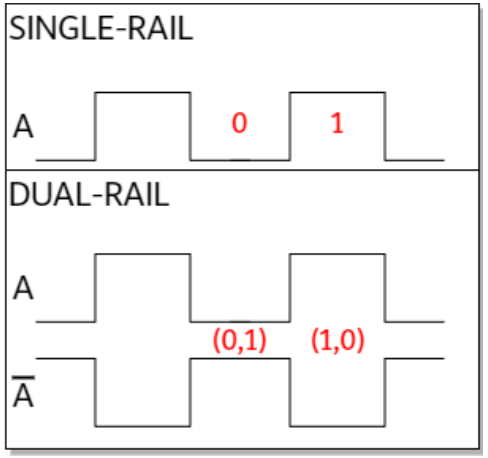
Masking-based countermeasures aim at increasing the **power of the noise** (hence the denominator of the SNR), by introducing external randomness in the computation.

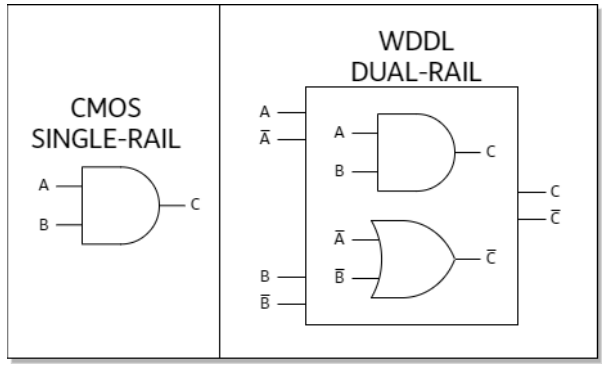
Physical Level Countermeasures

They aim at attenuating the signal emitted by the devices, by:

- Building physcal «barriers» to limit the emission
  - Shielding (...think about a Faraday-cage inside or around your chip)
  - On-chip filters for the power supply
- Different approach to the elaboration at physical level → this usually requires a different approach to the entire design of a chip.
  - Dual-rail logic families







Logic Level Countermeasures

One of the most common approach in mitigating side-channel attacks is based on **masking**. Normally, masking at logic level is based on **secret sharing** and **multi-party** computation. A secret variable is splitted in shares, by means of randomness. Knowing individual (or not completely) the share set does not reveal information about the secret variable. One of the main methodology to implement masking is based on **Boolean Masking**:

$$variabile\ x \rightarrow \{(x_1, x_2): x_1 = x \oplus r_1, x_2 = r_1\}$$
$$x = x_1 \oplus x_2$$

It offers many pros:

- Formal verification
- Easily scalable (in general): complexity vs protection order trade-off
- Suitable for both FPGA and ASIC
- Suitable for software implementations

Architectural Level Countermeasures

At this level, it is possible to reduce information leakage in many ways:

- Time-domain disalignment of operations:
  - Shuffling bit-level
  - Shuffling operation-level
  - Clock randomization
- De-correlation by means of parallel processing of random/dummy data
- Forcing in-order execution to avoid leakage from speculative execution
- Constant-time implementation and data-independent flow
- Noise-generator generating noise during operations that manipulate sensible data

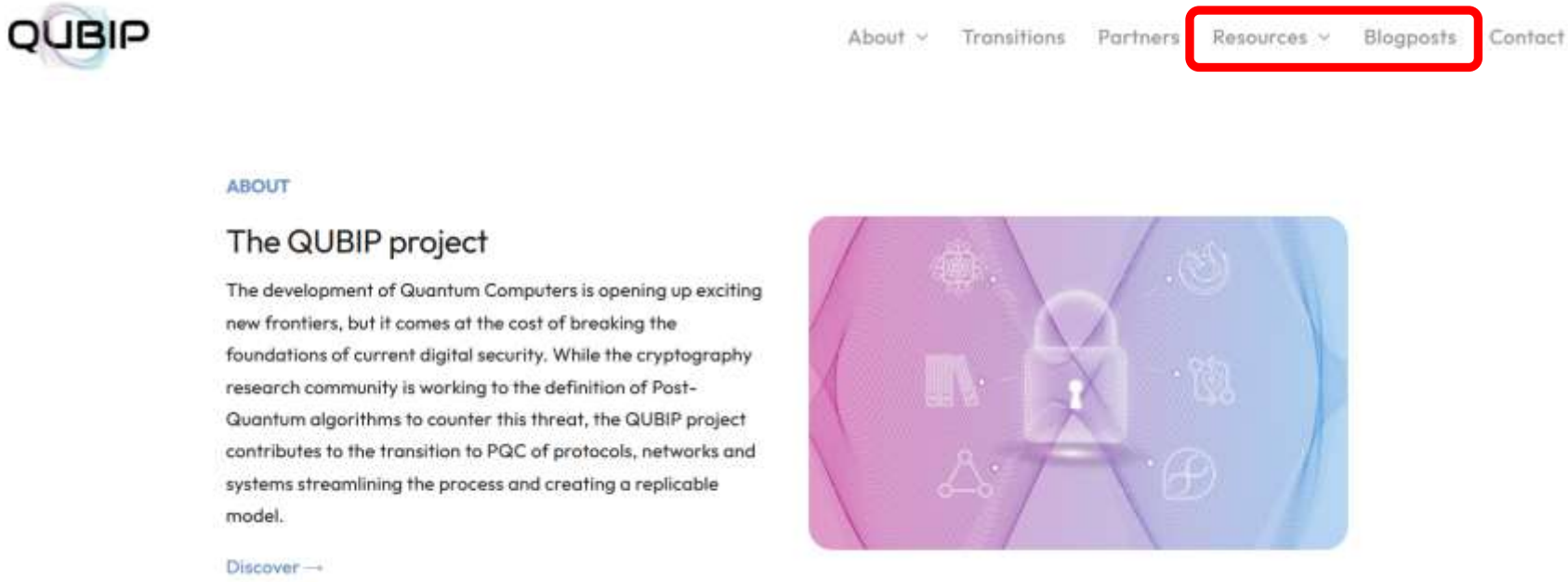
Algorithmic Level Countermeasures

- Many newer cryptographic algorithms and cryptosystems have been frequently designed considering side-channel resilience.

- NIST has been requiring side-channel resilience as a requirement for algorithms to be standardized:
  - Competition for Authenticated Encryption: Security, Applicability, and Robustness (CAESAR)
  - Light-Weight Cryptography (LWC)
  - Post-Quantum Cryptography (PQC)

- Some design approaches allows to strengthen the side-channel resilience [BOC+20]
  - Low-grade non-linear functions to efficiently adopt Boolean masking
  - Two-pass techniques for authenticated encryption
  - Ephemeral keys to limit the manipulation of sensible key material
  - Etc.

QUBIP



- Implementation of PQC algorithms will require to withstand side-channel attackers.
- The Digital Manufacturing IoT Pilot Demonstrator within the project will face the issue from an IoT perspective
  - FPGA implementation of PQC accelerators with side-channel protections
  - More difficult to protect PQC primitives compared to traditional ones
- Visit us at **qubip.eu**!!!

References (1/2)

[BCO04] E. Brier, C. Clavier and F. Olivier, "Correlation power analysis with a leakage model", in Proc. of CHES 2004.

[BKR11] A. Bogdanov, D. Khovratovich and C. Rechberger, "Biclique Cryptanalysis of the Full AES", in Proc. of ASIACRYPT 2011.

[BOC+20] D. Bellizia et al., "Mode-level vs. implementation-level physical security in symmetric cryptography: a practical guide through the leakage-resistance jungle", in Proc. of CRYPTO 2020.

[BUS21] D. Bellizia, B. Udvarhelyi and F.-X. Standaert, "Towards a Better Understanding of Side-Channel Analysis Measurements Setups", in Proc. of CARDIS 2021.

[CDG+13] J. Cooper, E. De Mulder, G. Goodwill et al., "Test Vector Leakage Assessment (TVLA) Methodology in Practice", International Cryptographic Module Conference 2013.

[GST14] D. Genkin, A. Shamir, and E. Tromer. "RSA key extraction via low-bandwidth acoustic cryptanalysis", in Proc. of CRYPTO 2014.

[HA22] G. Haas and A. Aysu, "Apple vs. EMA: Electromagnetic Side channel Attacks on Apple CoreCrypto", DAC 2022.

[Koc96] P.C. Kocher, “Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems”, in Proc. of CRYPTO 1996.

References (2/2)

[KJB99] P.C. Kocher, J. Jaffe and B. Jun, "Differential Power Analysis", in Proc. of CRYPTO ‘99.

[Man04] S. Mangard, "Hardware Countermeasures against DPA? A Statistical Analysis of Their Effectiveness", CT-RSA 2004, pp. 222-235

[QS01] J.J. Quisquater and D. Samyde, "Electromagnetic analysis (ema): Measures and counter-measures for smart cards“, in Smart Card Programming and Security, Springer Berlin Heidelberg, pp. 200-210.

[VIL95] Villa, S., A.L. Lacaita, and A. Pacelli. "Photon emission from hot electrons in silicon.“ in Physical Review B 52.15 (1995).

Thanks for your attention!
Q&A