



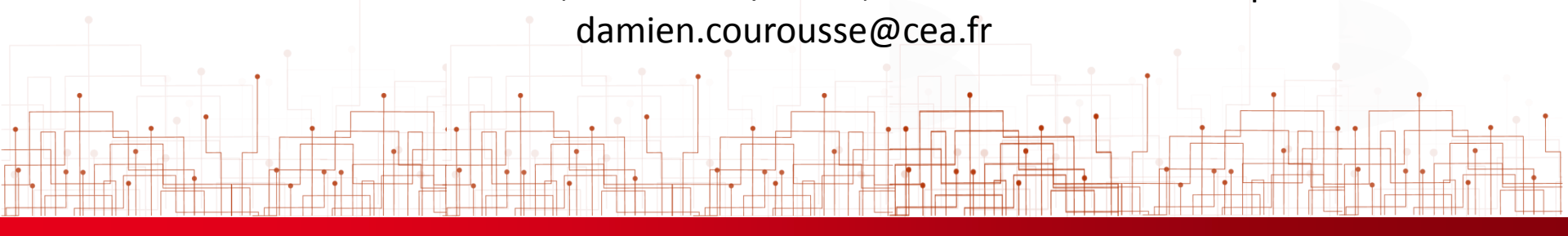
FROM RESEARCH TO INDUSTRY

cea tech

# Side-Channel Attacks

SSPREW 2018

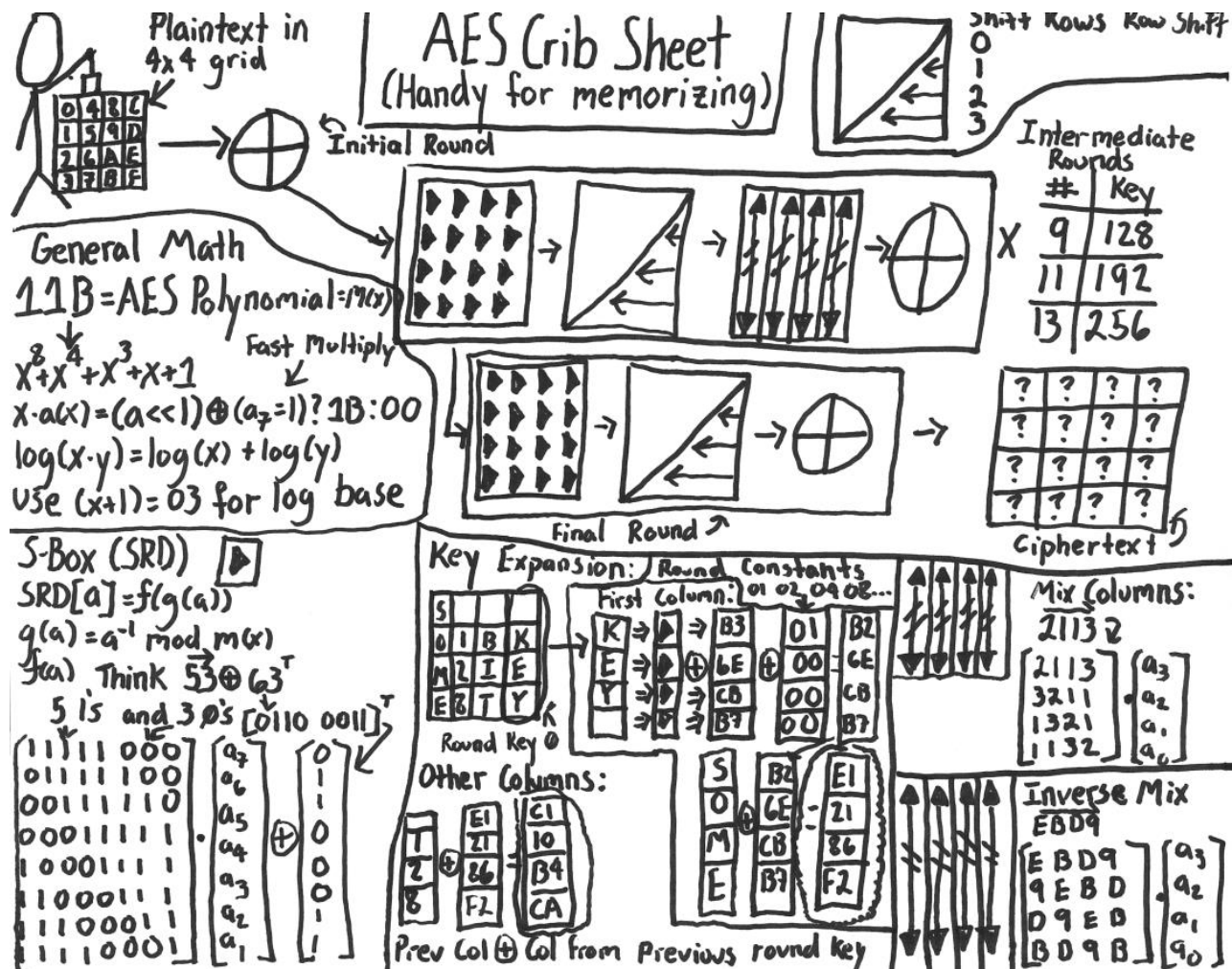
Damien Couroussé, CEA – LIST / LIALP; Grenoble Université Alpes  
[damien.courousse@cea.fr](mailto:damien.courousse@cea.fr)



- **Unless explicit mention at the bottom of the page, these slides are distributed under the Creative Common Attribution 3.0 License**
  - You are free:
    - to share—to copy, distribute and transmit the work
    - to remix—to adapt the work
  - under the following conditions:
    - Attribution: You must attribute the work (but not in any way that suggests that the author endorses you or your use of the work) as follows:  
“Courtesy of Damien Couroussé, CEA France”

The complete license text can be found at  
<http://creativecommons.org/licenses/by/3.0/legalcode>

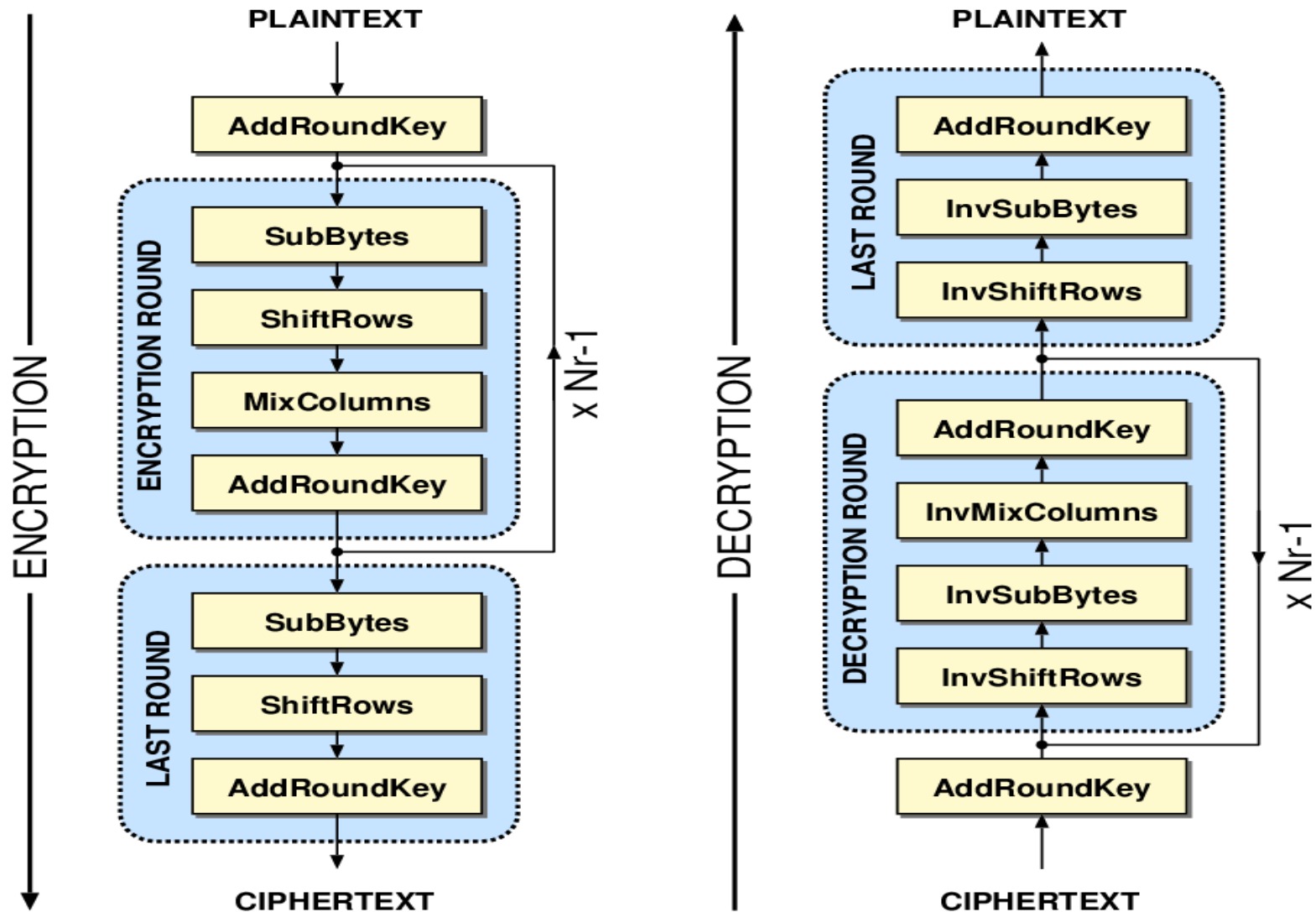
# AES, TIME AFTER TIME (BUT SO USEFUL...)



A Stick Figure Guide to the Advanced Encryption Standard --

<https://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html>

# AES, TIME AFTER TIME (BUT SO USEFUL...)



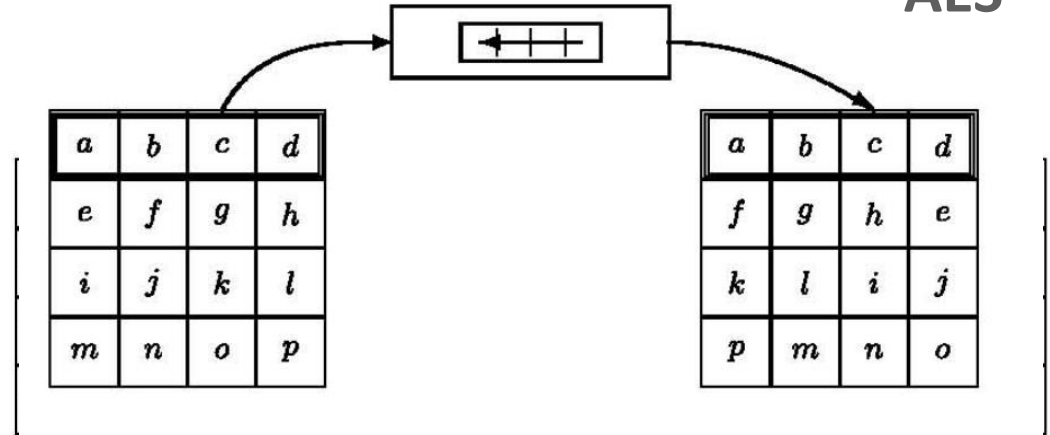
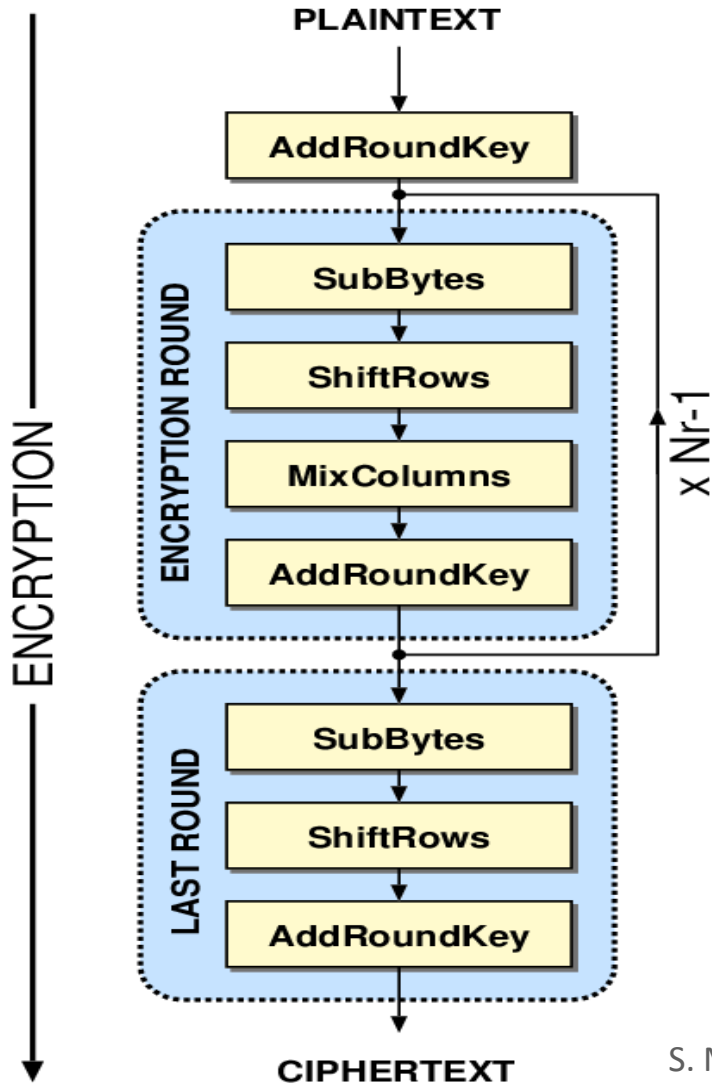


Figure B.6. ShiftRows operates on the rows of the state.

Fig

on.

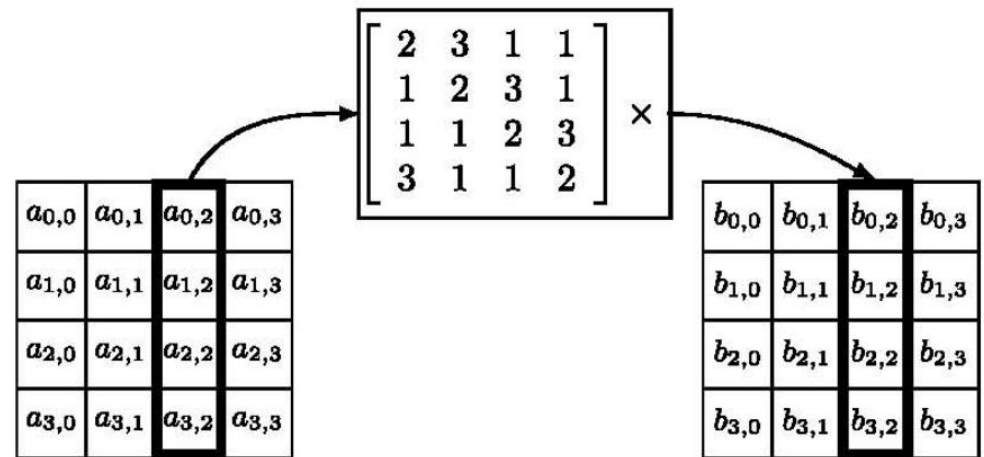


Figure B.7. MixColumns operates on the columns of the state.

S. Mangard, E. Oswald, and T. Popp, Power analysis attacks: Revealing the secrets of smart cards, vol. 31. Springer, 2007.

## Foot-Shooting Prevention Agreement

I, \_\_\_\_\_ , promise that once  
Your Name

I see how simple AES really is, I will  
not implement it in production code  
even though it would be really fun.

This agreement shall be in effect  
until the undersigned creates a  
meaningful interpretive dance that  
compares and contrasts cache-based,  
timing, and other side channel attacks  
and their countermeasures.

X \_\_\_\_\_  
Signature Date



# BESTIARY OF EMBEDDED SYSTEMS

## ... IN NEED FOR SECURITY CAPABILITES

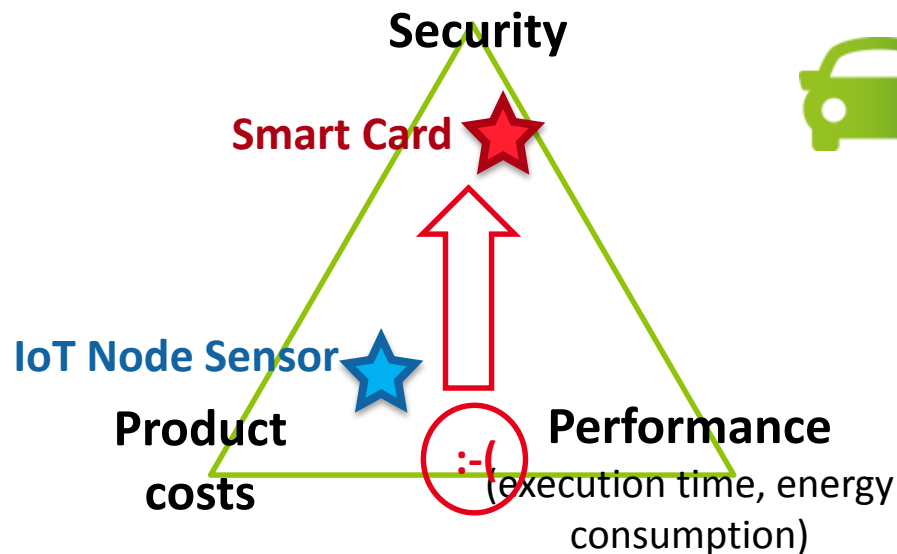
Smart Card

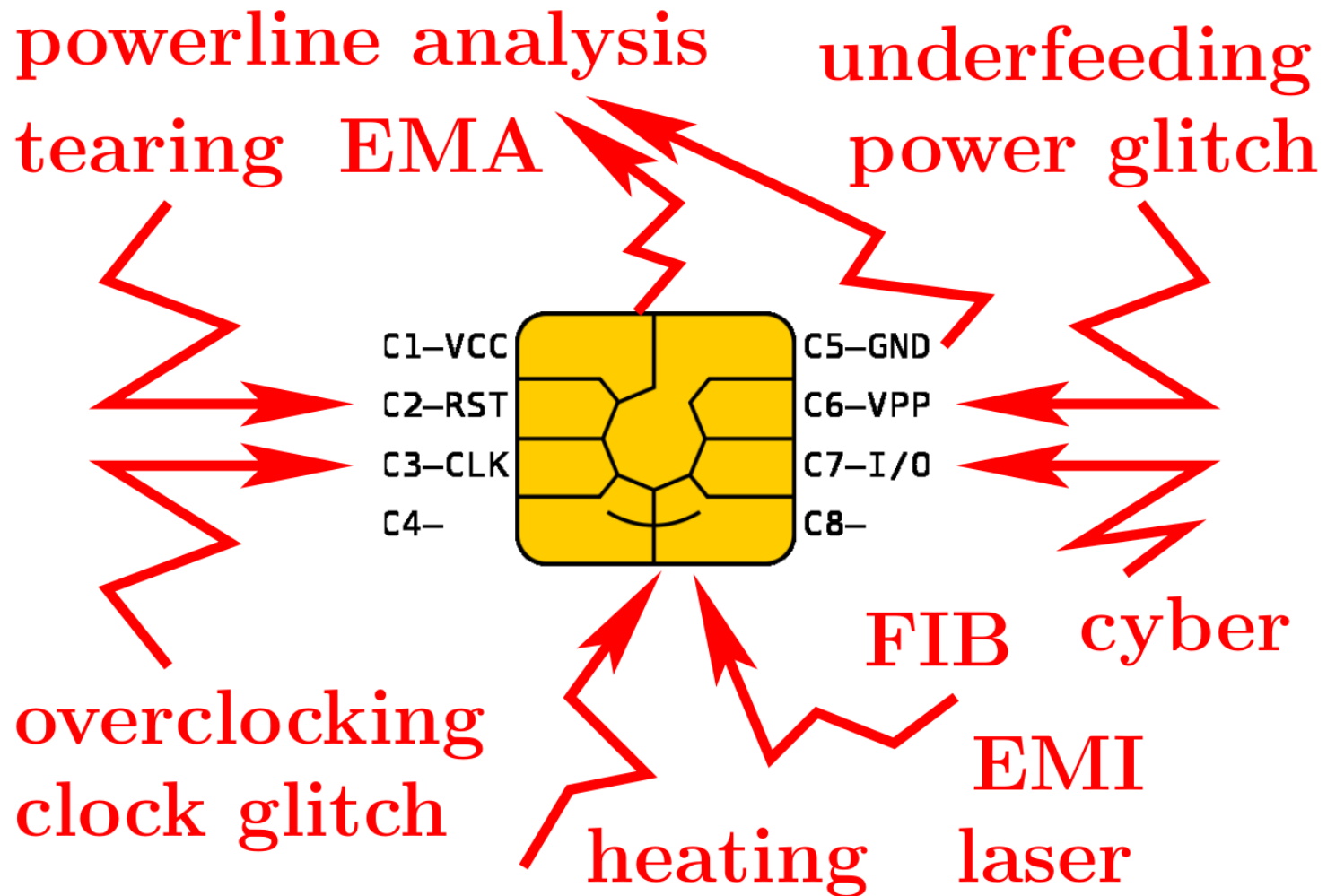


Secure Element inside...



... And many other things





Courtesy of Sylvain Guilley 2015, Télécom ParisTech - Secure-IC



# PHYSICAL ATTACKS: WHY ALL THE FUSS?

**Cryptography** is used to secure communications

- **Encrypted data** can be safely sent over an untrusted communication channel
- Cannot recover the encrypted information without the **key**

**Cryptanalysis** studies the mathematical properties of cryptographic algorithms, and provides a “practical” confidence in security bounds.

- Security bounds are expressed in terms of **attack complexity**

**Physical attacks** are the only (effective) way to break cryptography nowadays.

- Sometimes considered as part of cryptanalysis
- But quite different research communities

In your pocket, you have an “embedded” system secured against physical attacks!

- **Cryptanalysis**

Out of the scope of this talk

- **Reverse engineering**

Hardware inspection: decapsulation, physical abrasion, chemical etching, visual inspection, etc.

Software inspection: debug, memory dumps, code analysis, etc. [see lectures past in the week]

- **Passive attacks: side-channel attacks**

Observations: electromagnetic, electrical / power, acoustic, execution time, etc. [you are here]

- **Active attacks: fault attacks**

Laser or other lights illumination, under/over-voltage, clock glitches, electromagnetic perturbations, etc. [next lecture]

- **Logical attacks**

[see past lectures this weeks]

Sometimes considered as a « solved » issue in High Security products.

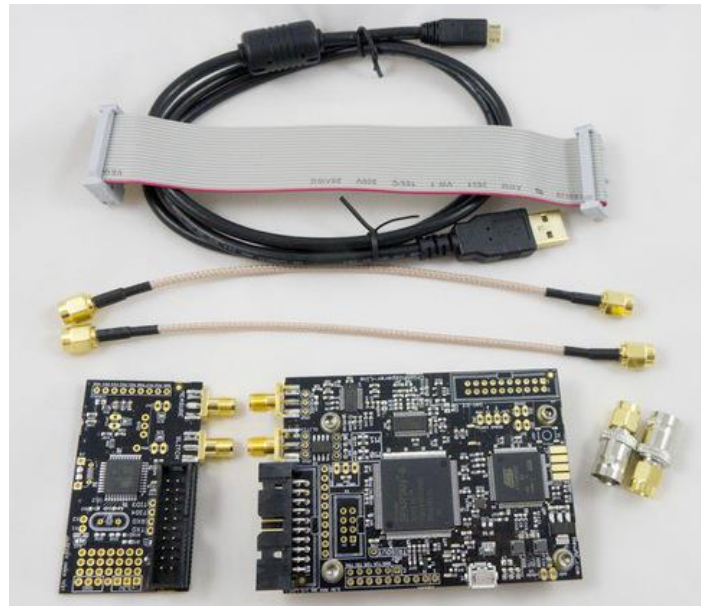
An attacker proceeds in two steps:

1. Global analysis of the target, looking for potential weaknesses or known vulnerabilities – this step is not considered in the littérature.
2. Focused attack on a target

## « PHYSICAL ATTACKS IS SCI-FI »

Physical attacks are considered (by software hackers) as not practical

- Require dedicated HW attack benches, can be quite expensive, especially for fault injection (laser benches)
- We also find low cost ones
  - E.g. *The ChipWhisperer*, starting at ~ 300€
- Require human expertise, but more than other attacks



<https://newae.com/tools/chipwhisperer>

### IoT Goes Nuclear: Creating a ZigBee Chain Reaction

*“Adjacent IoT devices will infect each other with a worm that will rapidly spread over large areas”*

- **Philips Hue Smart lamp**

- ZigBee protocol

- **Uploading malicious firmware with OTA update**

- Discovered the hex command code for OTA update
- Firmware is protected with a single global key! Using symmetric crypto (AES-CCM).

- **Attack path**

- Get access to the key → **side-channel attack with power analysis**
- Sign a malicious firmware
- Take over bulbs by: plugging a bulb, war-driving around in a car, war-flying with a drone
- Request OTA update
- The malicious firmware can request OTA update to its neighbours to spread.

### IoT Goes Nuclear: Creating a ZigBee Chain Reaction

Eyal Ronen(✉)\*, Colin OFlynn†, Adi Shamir\* and Achi-Or Weingarten\*  
PRELIMINARY DRAFT, VERSION 0.91

\*Weizmann Institute of Science, Rehovot, Israel

{[eyal.ronen](mailto:eyal.ronen@weizmann.ac.il),[adi.shamir](mailto:adi.shamir@weizmann.ac.il)}@weizmann.ac.il

†Dalhousie University, Halifax, Canada  
[coflynn@dal.ca](mailto:coflynn@dal.ca)

Other interesting read: N. Timmers and A. Spruyt, “Bypassing Secure Boot using Fault Injection,” presented at the Black Hat Europe 2016, 04-Nov-2016.

# SIDE-CHANNEL ATTACKS FOR REVERSE ENGINEERING

- Reverse-engineering from side-channel analysis
  - Even simpler on interpreters → read the instructions executed from an SPA analysis, i.e. reading directly on side-channel traces (see later in this presentation)
- SCARE attacks: recovering looking tables with side-channel analysis
- FIRE attacks: using fault attacks

T. Eisenbarth, C. Paar, and B. Weghenkel, “Building a Side Channel Based Disassembler,” in Transactions on Computational Science X, Springer, Berlin, Heidelberg, 2010, pp. 78–99.

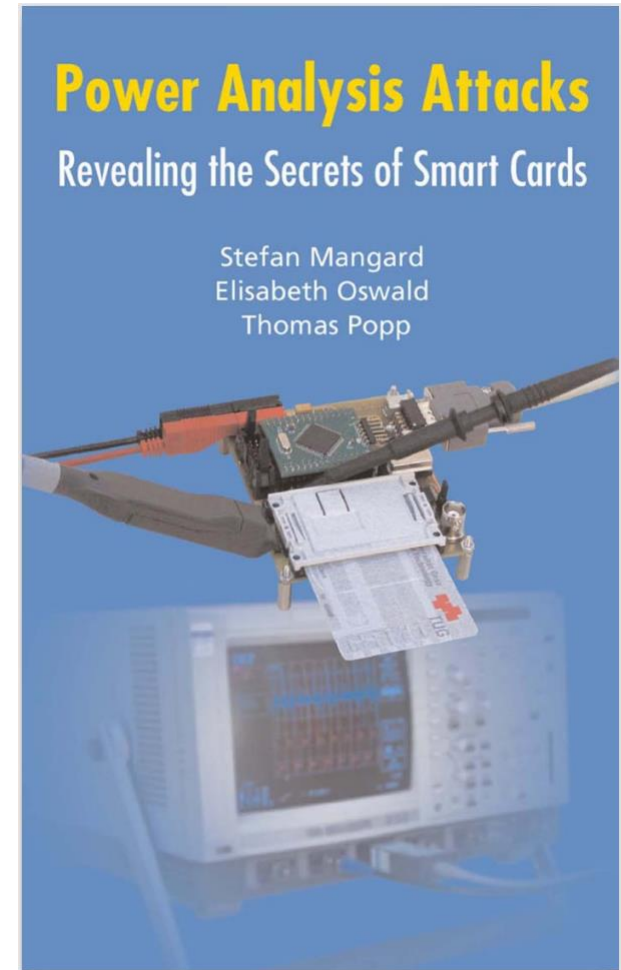
M. S. Pedro, M. Soos, and S. Guilley, “FIRE: Fault Injection for Reverse Engineering,” in Information Security Theory and Practice. Security and Privacy of Mobile Devices in Wireless Communication, 2011, pp. 280–293.

C. Clavier, “An Improved SCARE Cryptanalysis Against a Secret A3/A8 GSM Algorithm,” in Information Systems Security, 2007, pp. 143–155.

## The most comprehensive book about side-channel attacks

- Excellent introduction to side-channel attacks
- Published in 2007: does not cover recent attacks and countermeasures

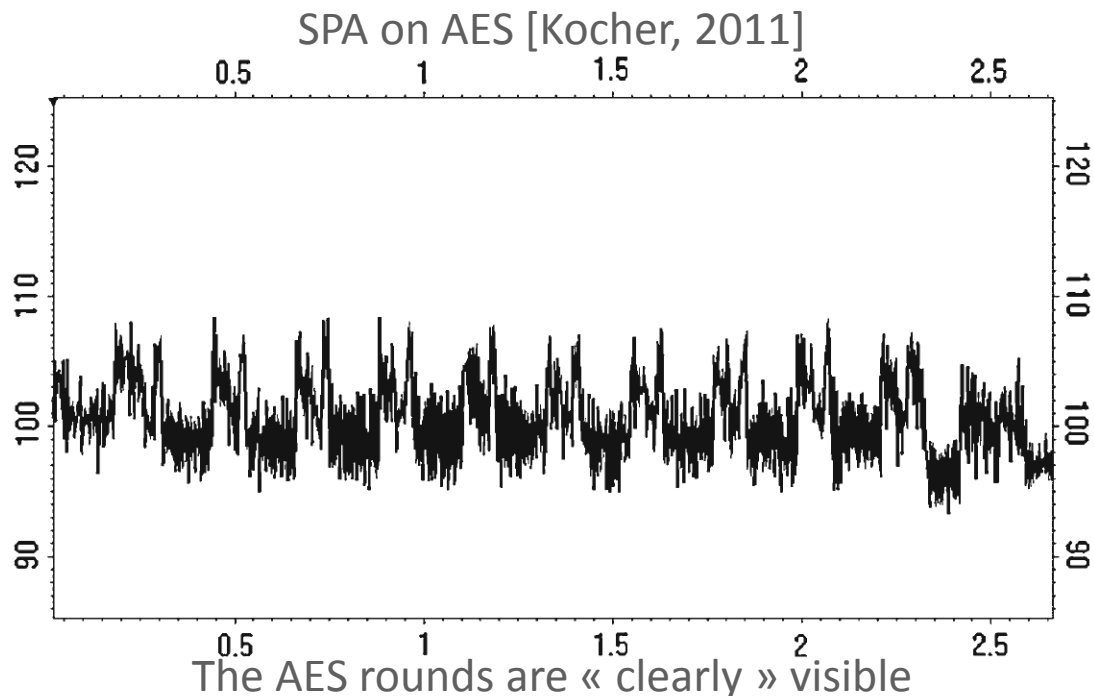
S. Mangard, E. Oswald, and T. Popp, Power analysis attacks: Revealing the secrets of smart cards, vol. 31. Springer, 2007.



# SIMPLE POWER ANALYSIS (SPA)

Direct interpretation of power consumption measurements

Extraction of information by inspection of single side-channel traces



- Nature of the algorithm
- Structure of the algorithm
  - Number of executions
  - Number of iterations
  - Number of sub-functions
  - nature of instructions executed (memory accesses...)
- Etc.

Illustration of SPA in the wild: C. O’Flynn, “A Lightbulb Worm? A teardown of the Philips Hue,” presented at the Black Hat, 2016. cf. slides ~60 to 70

P. Kocher, J. Jaffe, and B. Jun, “Differential Power Analysis,” in Advances in Cryptology — CRYPTO’ 99, vol. 1666, M. Wiener, Ed. Springer Berlin Heidelberg, 1999, pp. 388–397.

P. Kocher, J. Jaffe, B. Jun, and P. Rohatgi, “Introduction to differential power analysis,” Journal of Cryptographic Engineering, vol. 1, no. 1, pp. 5–27, 2011.



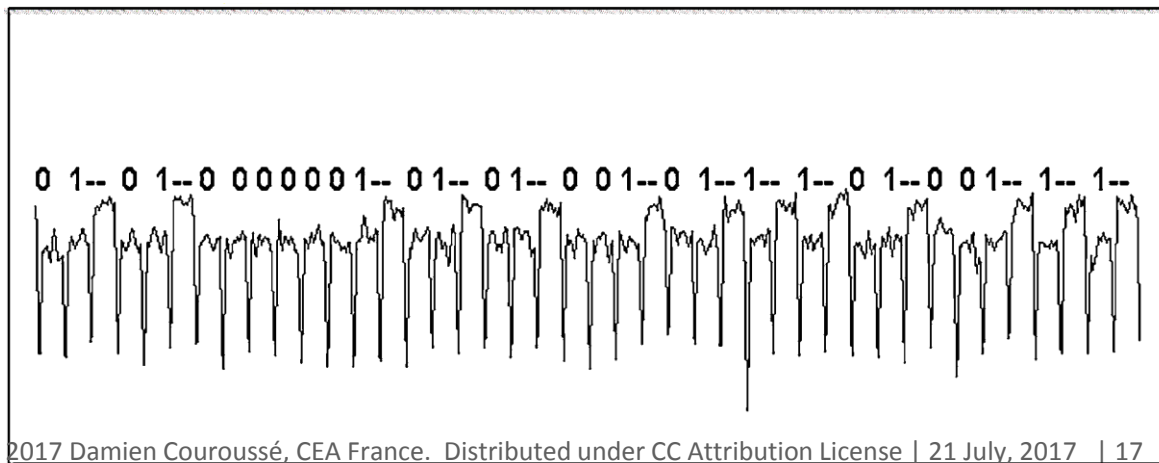
## SPA on RSA [Kocher, 2011]

```
-- Computing  $c = b^e \bmod m$   
-- Source: https://en.wikipedia.org/wiki/Modular\_exponentiation
```

```
function modular_pow(base, exponent, m)  
  if modulus = 1 then return 0  
  Assert :: (m - 1) * (m - 1) does not overflow base  
  result := 1  
  base := base mod m  
  while exponent > 0  
    if (exponent mod 2 == 1):  
      result := (result * base) mod m  
    exponent := exponent >> 1  
    base := (base * base) mod m  
  return result
```

Direct access to key contents:

- bit 0 = square
- bit 1 = square, multiply



## Finding a needle in a haystack...

- Relationship between the different components of power consumption:

$$P_{\text{total}} = P_{\text{operations}} + P_{\text{data}} + P_{\text{noise}}$$

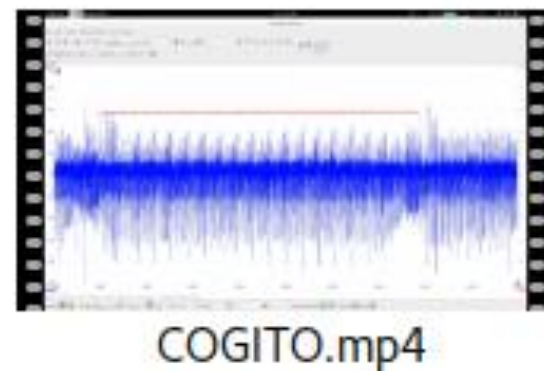
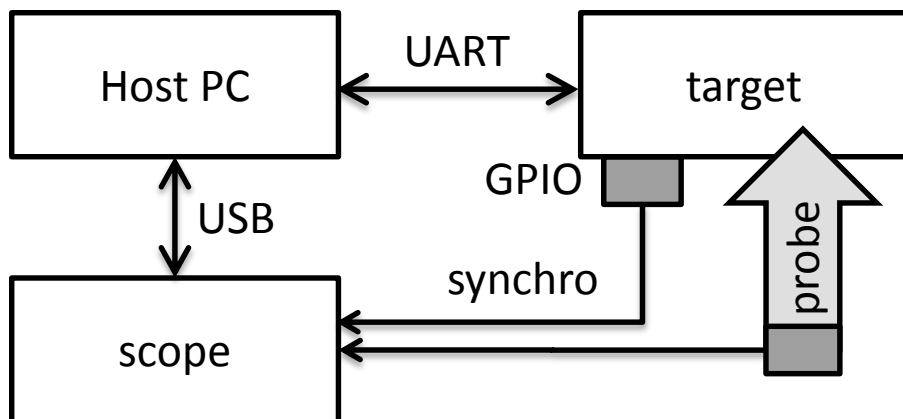
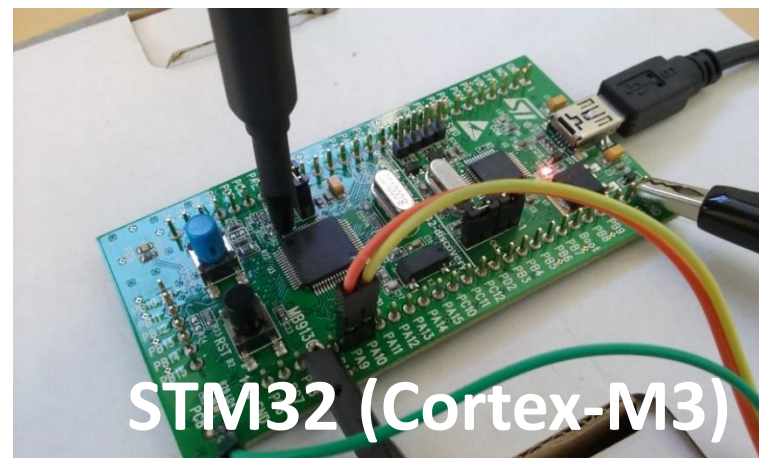
$$P_{\text{total}} = P_{\text{exploitable}} + P_{\text{switching.noise}} + P_{\text{electronic.noise}} + P_{\text{const}}$$

needle                      haystack

- Power signal: a static and a dynamic component.
  - Static component: power consumption of the gate states  $\rightarrow a * HW(\text{state})$
  - Dynamic component: power consumption of transitions in gate states  
 $\rightarrow b * HD(\text{state}[i], \text{state}[i-1])$
- Other needles & stacks
  - Electromagnetic emissions,
  - Execution time,
  - Acoustic emissions,
  - Etc.

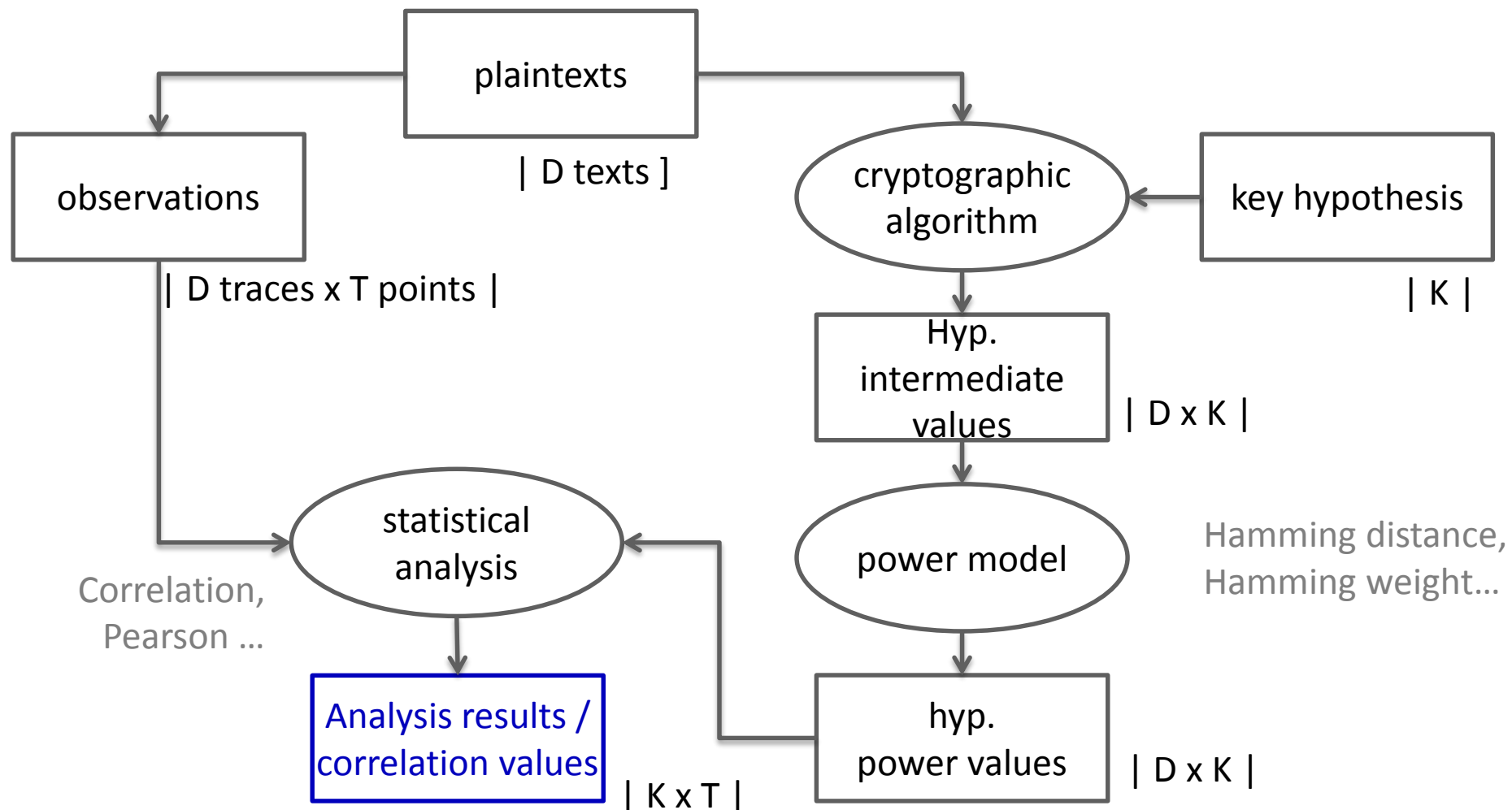
## CPA – MEASUREMENT SETUP

- **Target:** STM32 – ARM Cortex-M3 @ 24MHz, 128KB flash, 8KB RAM
- The AES key is fixed
- A GPIO trigger is used to facilitate the trace measurements
- The attacker either knows the plaintext or the ciphertext (public data)
- Text chosen attack:
  - Generate D random plaintexts
  - Ask the cipher text to the target
  - Record the EM trace during encryption
- **Do the computation analysis!**

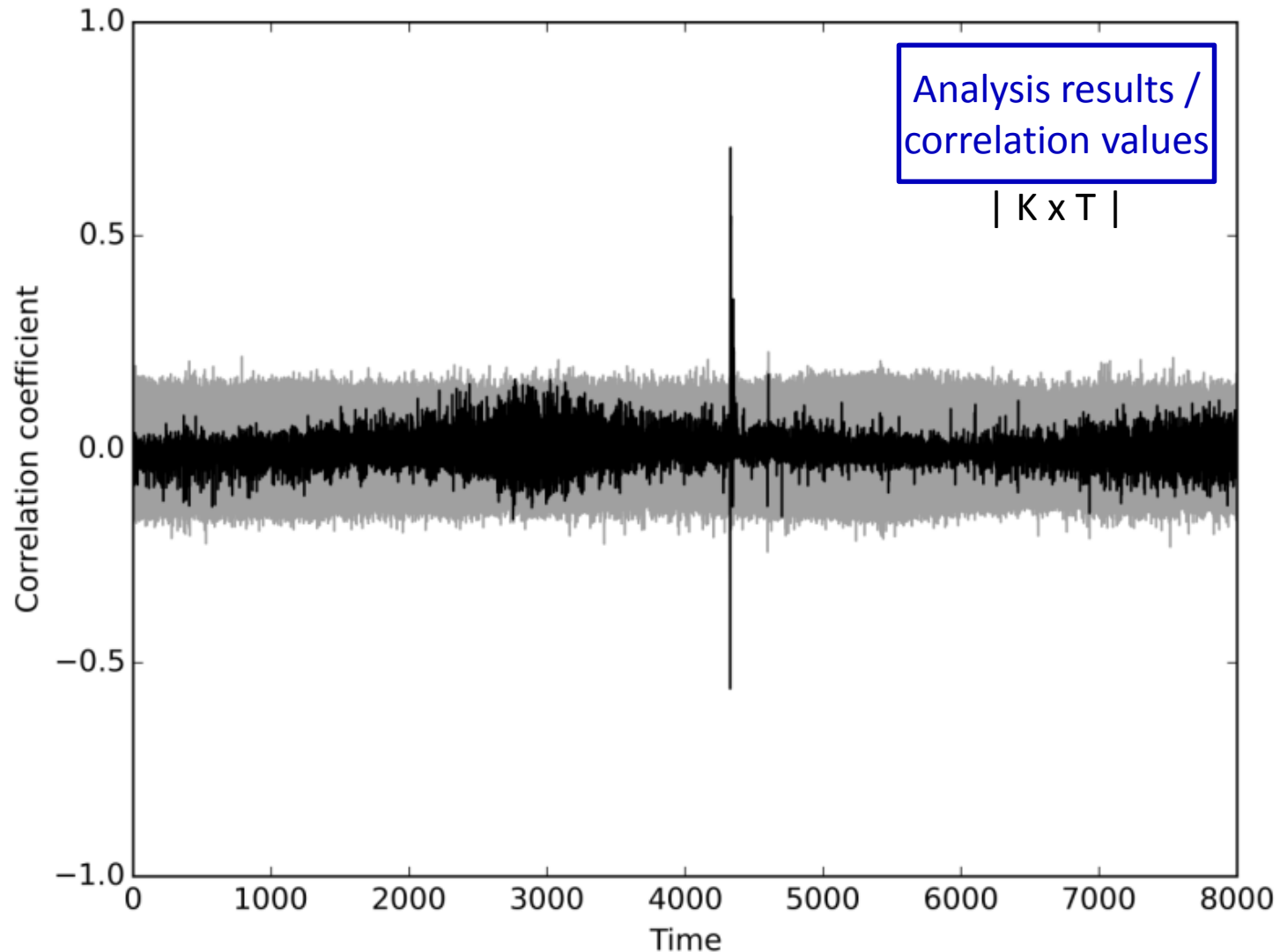


m: plaintext -> controlled by the attacker or observable

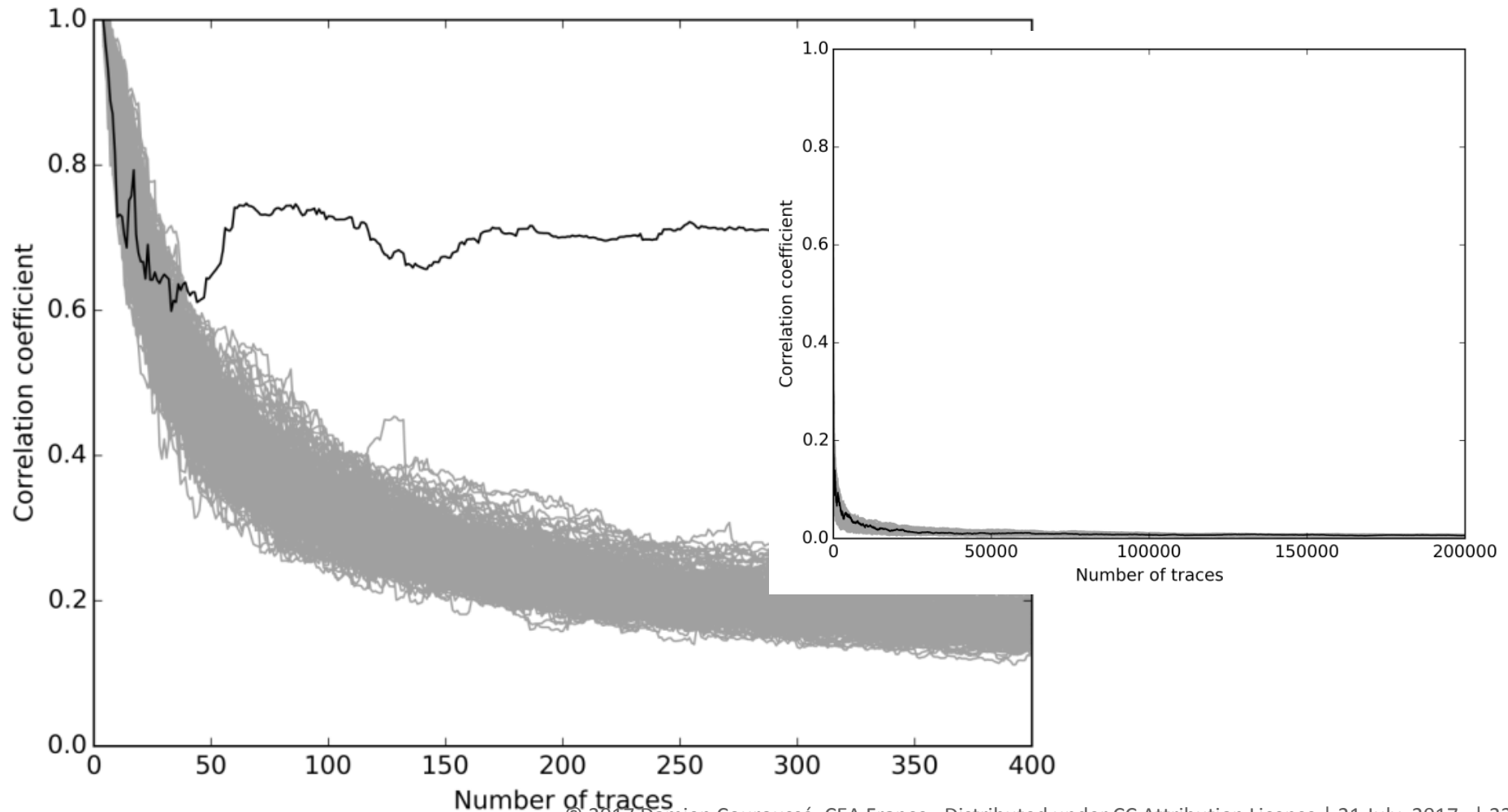
k: cipher key -> unknown to the attacker



## Plot of the correlation matrix, after D side-channel acquisitions



Plot of the maximum value in each row,  
For all  $d$  acquisitions in  $[1; D]$  (Or: plot for  $t$  fixed in  $T$ )



# ESTIMATING THE SUCCESS OF AN ATTACK

**Success rate: success probability of a successful attack**

$$SR = \Pr[ A(E_{k_0}, L) = k_0 ]$$

A side-channel attack

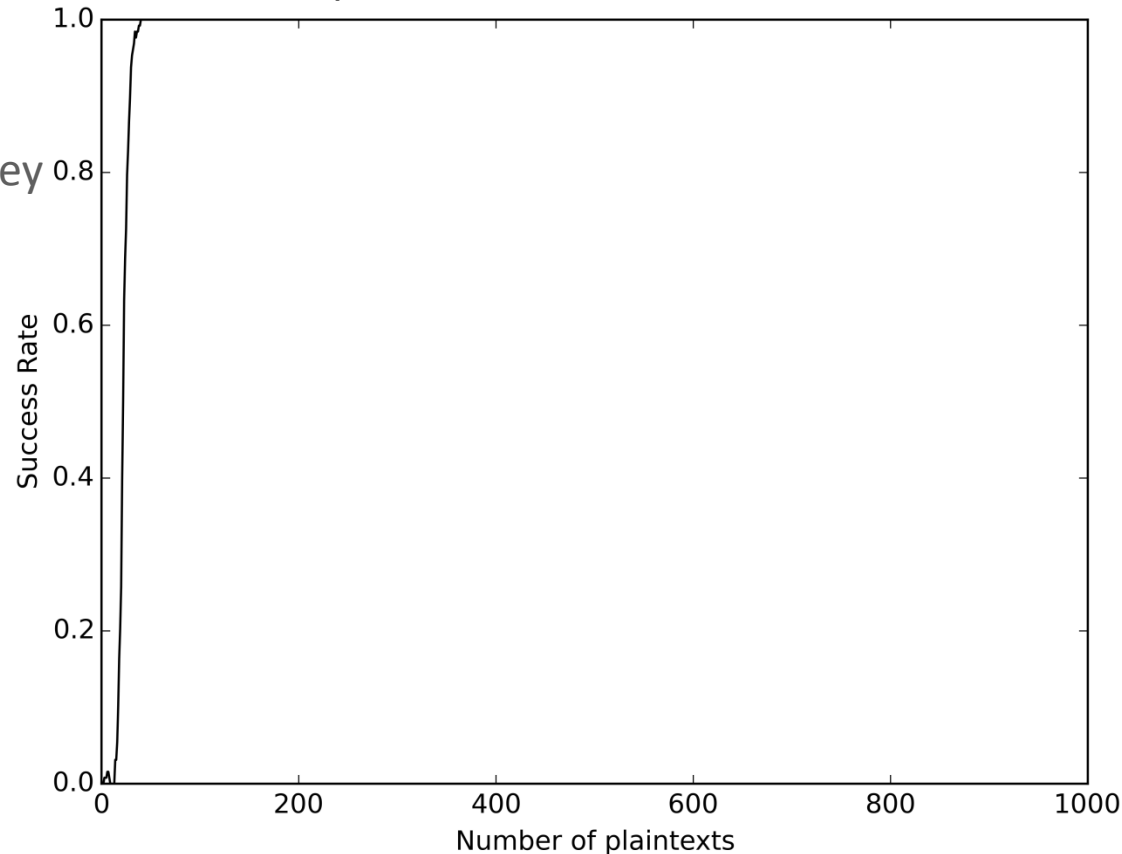
$k_0$  correct key

$E_{k_0}$  encryption with correct key

L leakage

n-order success rate?

Empirical evaluation at first order



F.-X. Standaert, T. Malkin, and M. Yung, “A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks,” in Eurocrypt, 2009, vol. 5479, pp. 443–461.



- **CPA / DPA ... attacks do not constitute a security evaluation.**
- **Playing the role of the attacker is great, but the attacker**
  - is focused on a potential vulnerability
  - Follows a specific attack path
- **Starting from the previous attack, we could change**
  - The hypothetical intermediate values: output of 1st SubBytes, output of 1st AddRoundKey, input of the 10th SubBytes...
  - The power model: Hamming Weight, Hamming Distance, no power model...
  - The distinguisher: Pearson Correlation, Mutual Information...
  - There are many other attacks!
- **Our evaluation target is very “leaky” (less than 50 traces is often enough)**
  - Unprotected components executed on more complex targets (i.e. ARM Cortex A9) will require 100.000 to  $10^6$  traces.
  - What about attacking a counter-measure in this case?
- **As a security designer, you need to cover all the possible attack passes**

## TLVA: Test Leakage Vector Assessment

- Exploit Welch's t-test to assess the amount of information leakage
- Extract two populations of side-channel observations (traces)
- Test the null hypothesis: the two populations are not statistically distinguishable → no information leakage

$$t = \frac{\mu_0 - \mu_1}{\sqrt{\frac{s_0^2}{n_0} + \frac{s_1^2}{n_1}}},$$

$t > 4.5 \rightarrow$  confidence of 99.999% that the null hypothesis is rejected



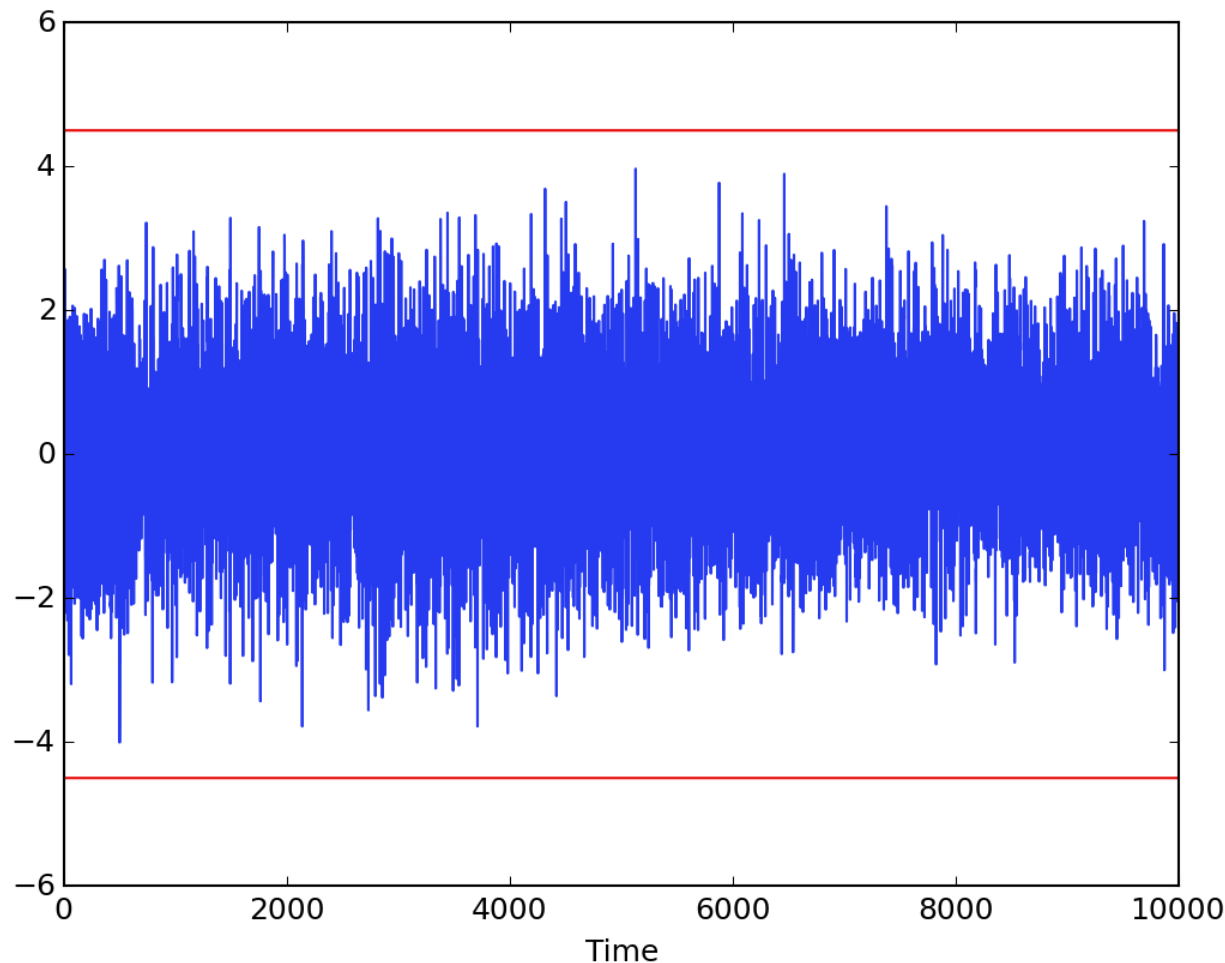
G. Goodwill, B. Jun, J. Jaffe, and P. Rohatgi, "A testing methodology for side-channel resistance validation," in NIST non-invasive attack testing workshop, 2011.

D. B. Roy, S. Bhasin, S. Guilley, A. Heuser, S. Patranabis, and D. Mukhopadhyay, "Leak Me If You Can: Does TVLA Reveal Success Rate?," 1152, 2016.

T. Schneider and A. Moradi, "Leakage Assessment Methodology - a clear roadmap for side-channel evaluations," 207, 2015.

- Exploit
- Extract
- Test the distinguishing

$$t = \frac{\mu}{\sqrt{\frac{s}{n}}} \quad \text{T-TEST}$$



## hypothesis

ation," in NIST

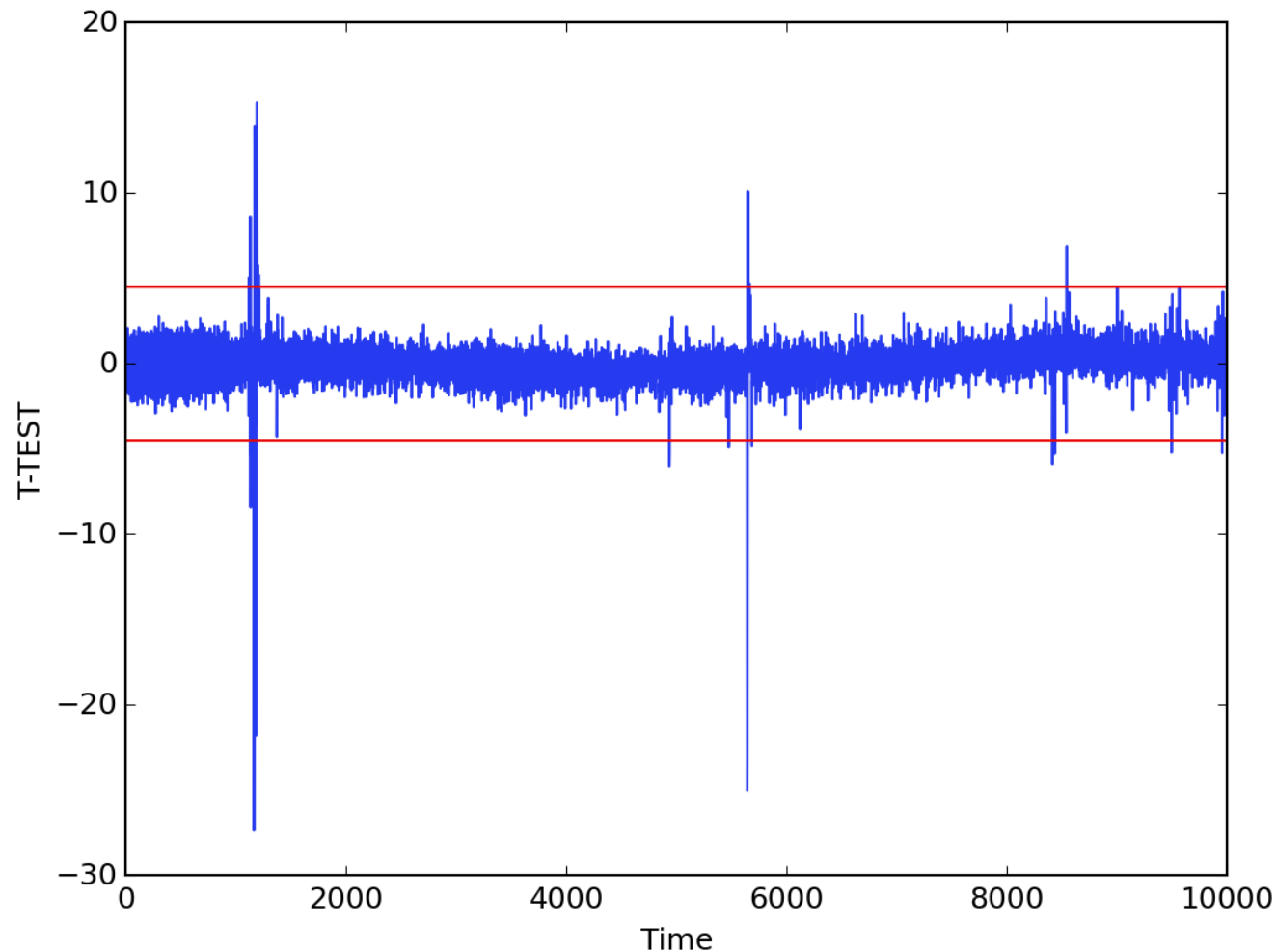
G. Goodwill, B.  
non-invasive a

D. B. Roy, S. Bhattach, J. Gurney, A. Heuser, J. Papanicolaou, and D. Mukhopadhyay, "Leak Me If You Can: Does TVLA Reveal Success Rate?," 1152, 2016.

T. Schneider and A. Moradi, "Leakage Assessment Methodology: a clear roadmap for side-channel evaluations," © 2017, Damien Couroussé, CEA France. Distributed under CC Attribution License | 21 July, 2017 | 26

$$Q_0 = \{T_i \mid \text{target bit}(D_i) = 0\}, \quad Q_1 = \{T_i \mid \text{target bit}(D_i) = 1\}.$$

$$Q_0 = \{T_i \mid \text{target byte}(D_i) = x\}, \quad Q_1 = \{T_i \mid \text{target byte}(D_i) \neq x\}.$$

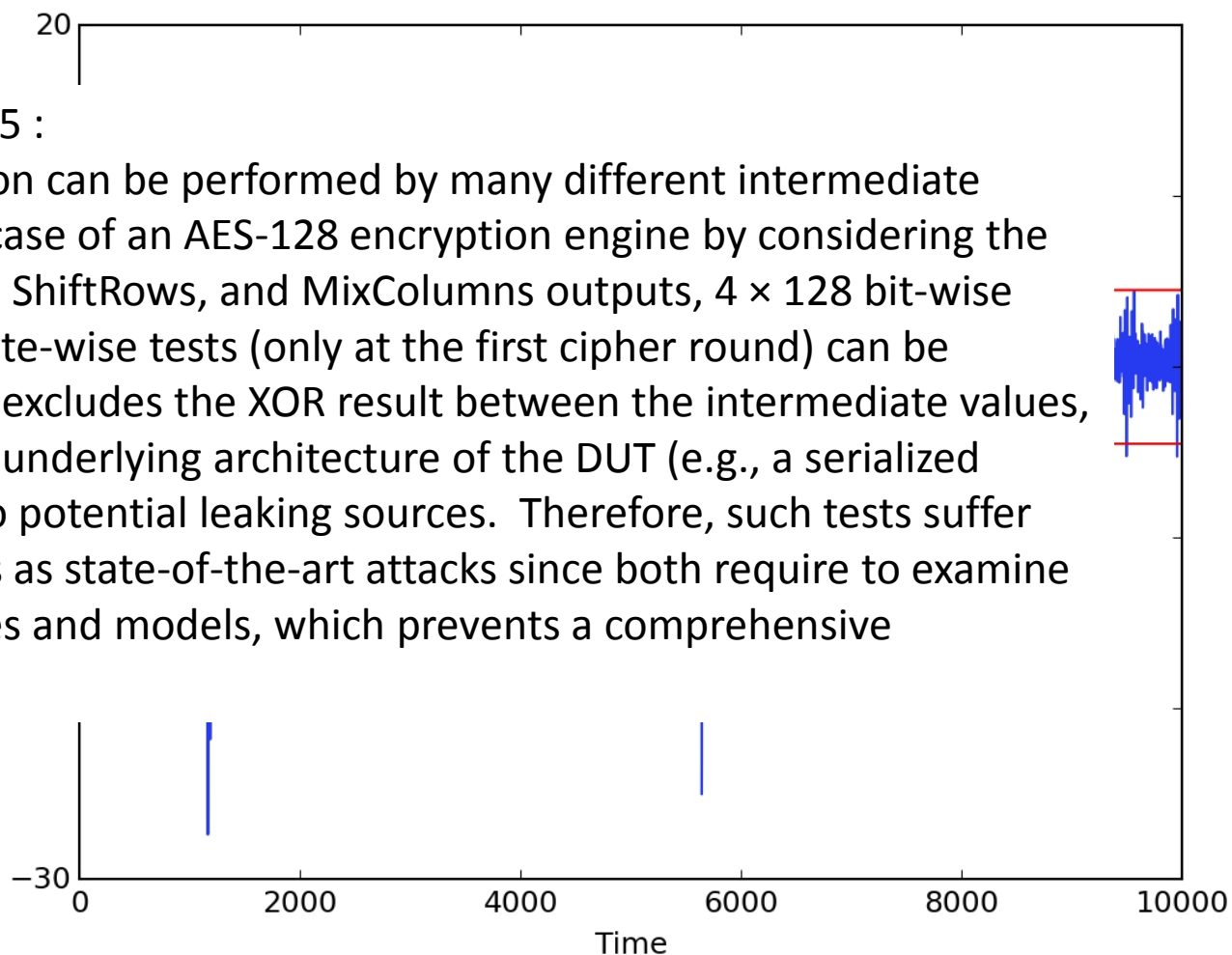


$$Q_0 = \{T_i \mid \text{target bit}(D_i) = 0\}, \quad Q_1 = \{T_i \mid \text{target bit}(D_i) = 1\}.$$

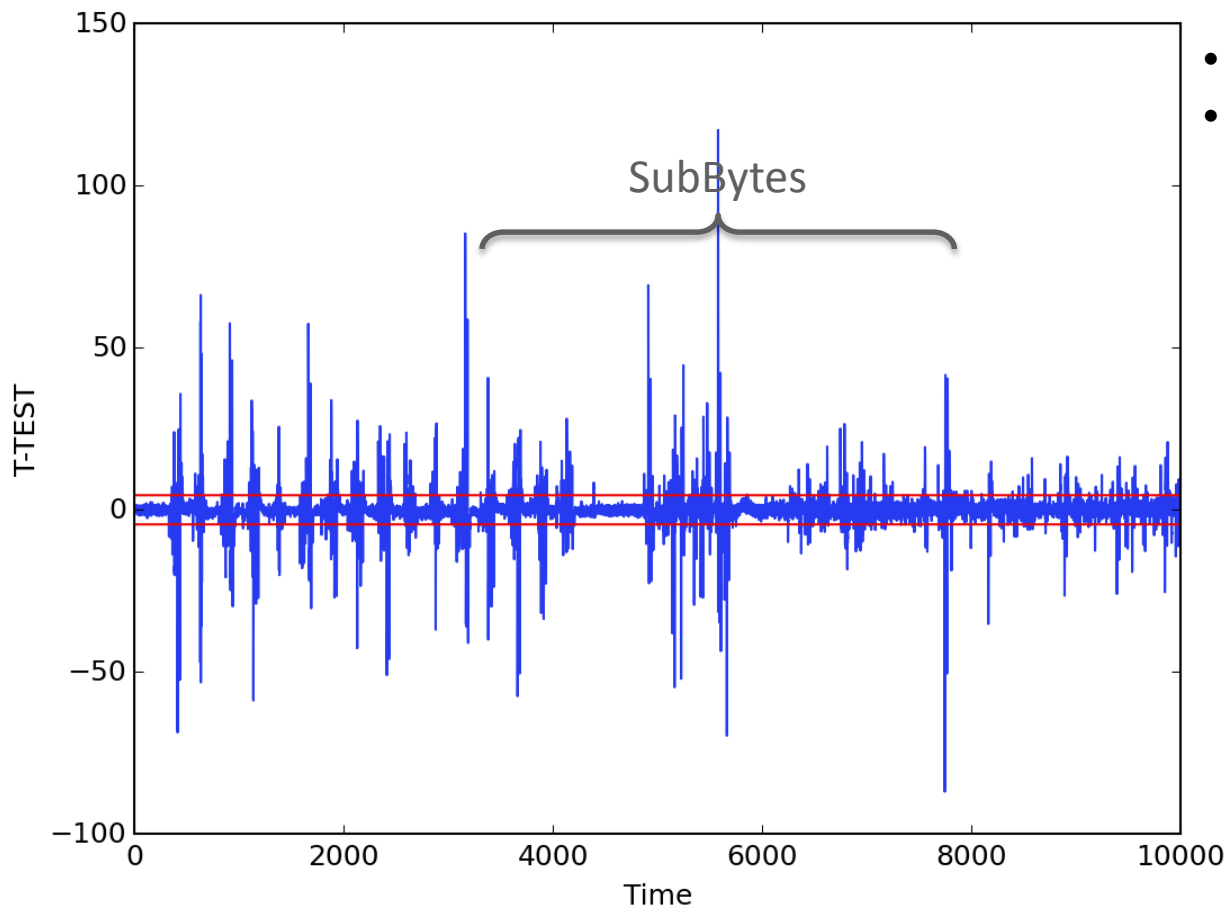
$$Q_0 = \{T_i \mid \text{target byte}(D_i) = x\}, \quad Q_1 = \{T_i \mid \text{target byte}(D_i) \neq x\}.$$

Schneider et Moradi 2015 :

«Therefore, the evaluation can be performed by many different intermediate values. For example, in case of an AES-128 encryption engine by considering the AddRoundKey, SubBytes, ShiftRows, and MixColumns outputs,  $4 \times 128$  bit-wise tests and  $4 \times 16 \times 256$  byte-wise tests (only at the first cipher round) can be conducted. This already excludes the XOR result between the intermediate values, which depending on the underlying architecture of the DUT (e.g., a serialized architecture) may lead to potential leaking sources. Therefore, such tests suffer from the same weakness as state-of-the-art attacks since both require to examine many intermediate values and models, which prevents a comprehensive evaluation. »



*Q0: fixed input plaintext | Q1: random input plaintext*



- Reveals information leakage
- Does not pronounce about the exploitability of this leakage

## “VERTICAL” ATTACKS AND LIMITS OF THE T-TEST

- **We can build a set of traces from an unprotected AES implementation so that the t-test sees no information leakage**
- **How (not) to Use Welch’s T-test in Side-Channel Security Evaluations [1]**
  - TLVA: determine the security order
  - Noise level: test independently

[1] F.-X. Standaert, ‘How (not) to Use Welch’s T-test in Side-Channel Security Evaluations’, 138, 2017.



# **COUNTER-MEASURES AGAINST SIDE-CHANNEL ATTACKS**

**MASKING & MASKING**

## MASKING

In a masked implementation, **each intermediate value  $v$  is concealed** by a random value  $m$  that is called mask:  
 **$Vm = v * m$** . The mask  $m$  is generated internally, i.e. inside the cryptographic device, and varies from execution to execution. Hence, it is not known by the attacker.

[DPA book]



- **Boolean masking:** operator  $*$  is *xor*
- **Arithmetic masking:** operator  $*$  is the modular addition or the modular multiplication

Objective: **each masked variable is statistically independent of the secret  $v$** .  
A (first-order) CPA attack can recover a (first-order) masked variable, but this knowledge is not sufficient to recover the secret value.

Masking countermeasures are applied at the **algorithmic level**.

Our following discussions will be based on the parallel implementation of a masking scheme such as described in [2]. More precisely, we will consider the simplest example where all the shares are in  $\text{GF}(2)$  (generalizations to other fields follow naturally). In this setting, we have a sensitive variable  $x$  that is split into  $m$  shares such that  $x = x_1 \oplus x_2 \oplus \dots \oplus x_m$ , with  $\oplus$  the bitwise XOR. The first  $m - 1$  shares are picked up uniformly at random:  $(x_1, x_2, \dots, x_{m-1}) \xleftarrow{\mathcal{R}} \{0, 1\}$ , and the last one is computed as  $x_m = x \oplus x_1 \oplus x_2 \oplus \dots \oplus x_{m-1}$ .

Denoting the vector of shares  $(x_1, x_2, \dots, x_m)$  as  $\bar{x}$ , we will consider an adversary who observes a single leakage sample corresponding to the parallel manipulation of these shares. A simple model for this setting is to assume this sample to be a linear combination of the shares, namely:

$$L_1(\bar{x}) = \left( \sum_{i=1}^m \alpha_i \cdot x_i \right) + N,$$

F.-X. Standaert, “How (not) to Use Welch’s T-test in Side-Channel Security Evaluations,” 138, 2017.

The goal of hiding countermeasures is to make the **power consumption** of cryptographic devices **independent of the intermediate values** and **independent of the operations** that are performed. There are essentially two approaches to achieve this independence.

1. the **power consumption is random**.
2. **consume an equal amount of power** for all operations and for all data values.

[DPA book]

Hiding countermeasures aim at **breaking the observable relation** between **the algorithm** (operations and intermediate variables) and **observations**.



**Information leakage:** information related to secret data and secret operations “sneaks” outside of the secured component (via a side channel)

**Hiding:** “reducing the SNR”, where

- Signal -> information leakage
- Noise -> everything else
- Temporal dispersion: spread leakage at different computation times
  - Shuffle independent operations
  - Insert «dummy» operations to randomly delay the secret computation
- Spatial dispersion:
  - Move the leaky computation at different places in the circuit
    - E.g. use different registers
  - Modify the “appearance” of information leakage
    - E.g. use different operations

**In practice, a secured product combines masking and hiding countermeasures.**

# Side-Channel Attacks

SSPREW 2018

Damien Couroussé, CEA – LIST / LIALP; Grenoble Université Alpes  
[damien.courousse@cea.fr](mailto:damien.courousse@cea.fr)



Centre de Grenoble  
17 rue des Martyrs  
38054 Grenoble Cedex



Centre de Saclay  
Nano-Innov PC 172  
91191 Gif sur Yvette Cedex

