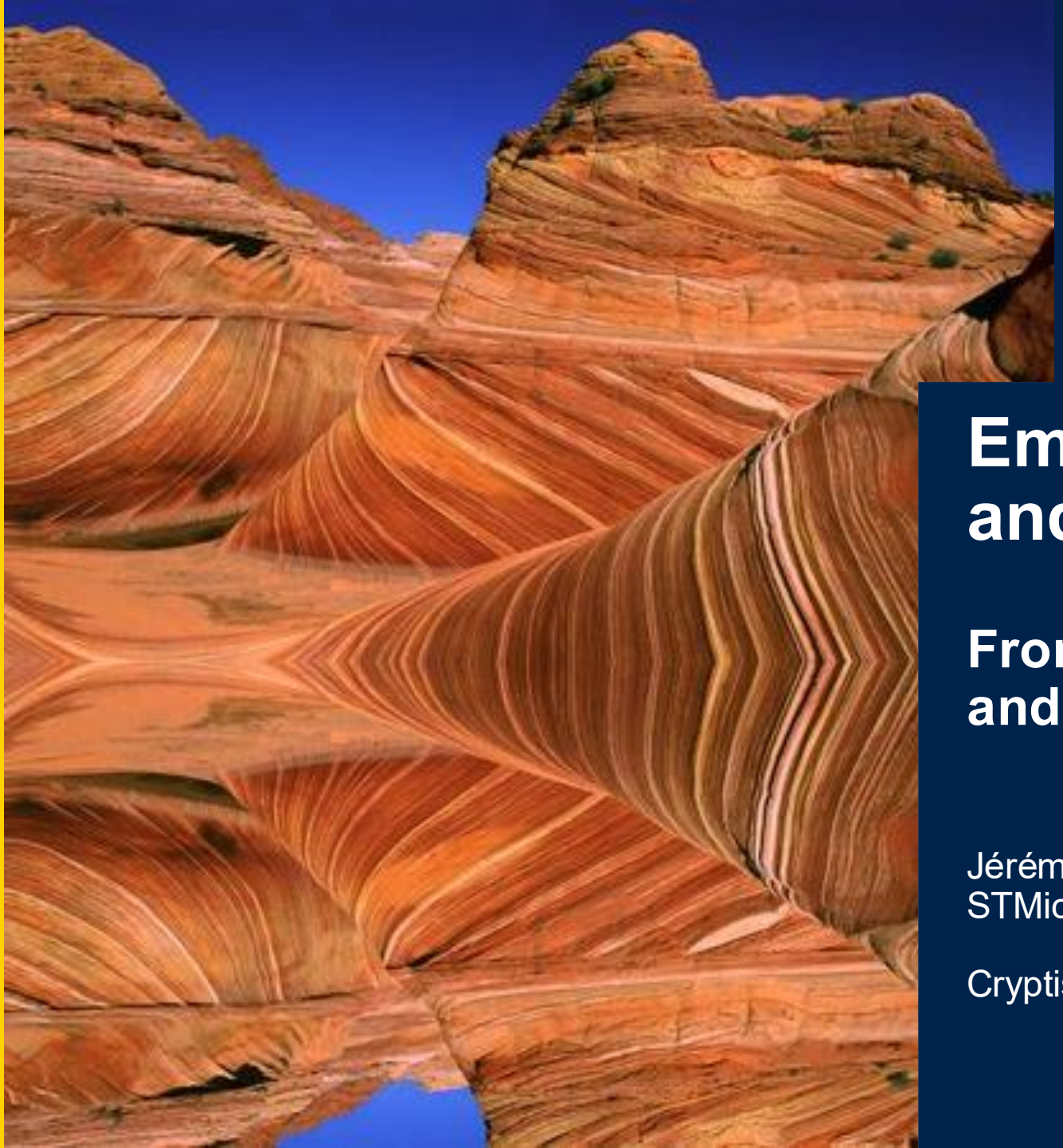


life.augmented

Embedded Cryptography and Side-Channel Analysis

From past to present … and beyond …

Jérémy Dron & Benoît Feix
STMicroelectronics

Cryptis Master2, November 2025.

Agenda

1. Hardware attacks
2. Side-channel is coming
3. A taste of RSA
4. Side-channel methodology
5. Post Quantum Cryptography
6. Resources, nowadays
7. Attack potential and standards

life.augmented

Agenda

Past: from 90's ... Before Kocher ..

- Designing a security product
- Invasive Attacks and Countermeasures

96's: (well?)coming side-channel attacks

- Simple Analysis, explanations, demos
- Differential Analysis, DPA, CPA, demos on DES and AES
- Countermeasures, High Order on Symmetric Algorithm and CMs: demo

Back to RSA

- Differential Analysis: demo
- Countermeasures

Side-channel Methodology

- Demos on t test and reverse analysis

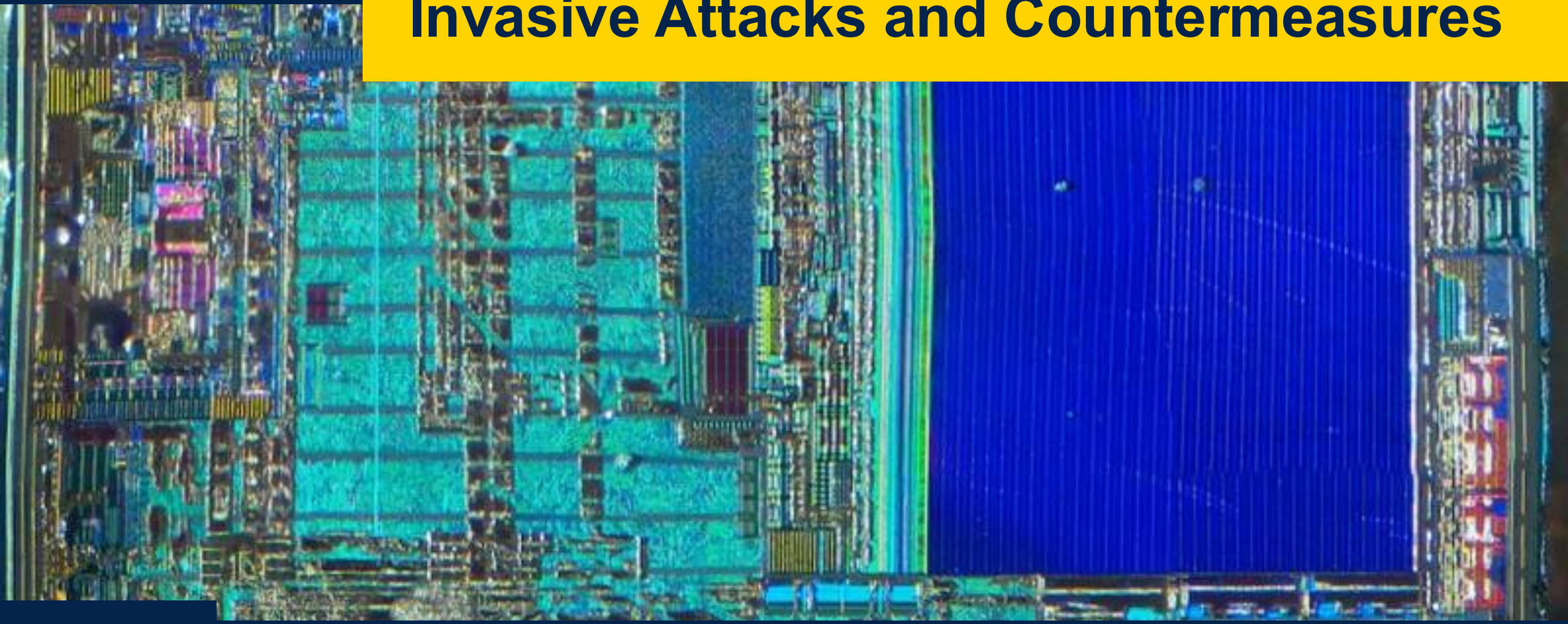
Nowadays

- No more 'confidential' science, open sources and CTF
- Recent attacks

Post Quantum Cryptography

Standard and attack ratings in the industry

Secure Products
Invasive Attacks and Countermeasures



life.augmented

Started with smart cards …

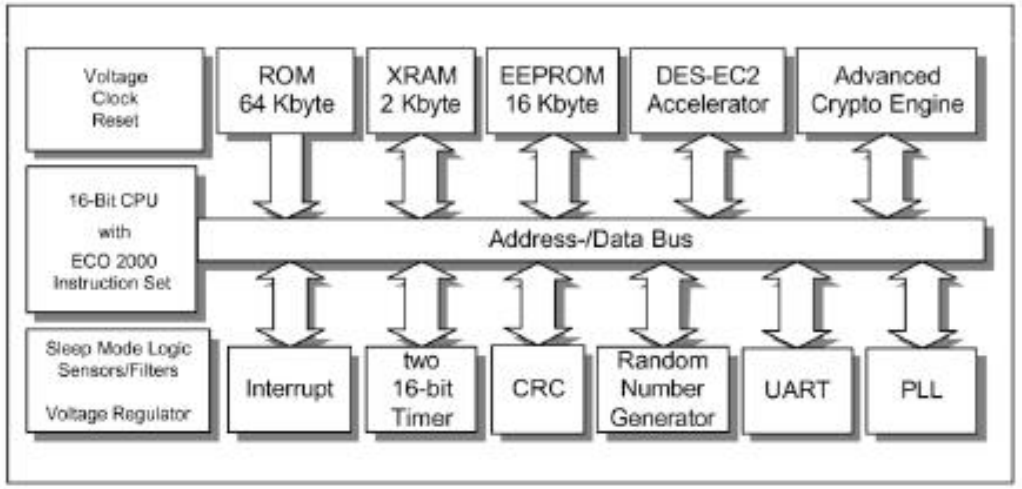


Figure : SLE66 Infineon Block Diagram

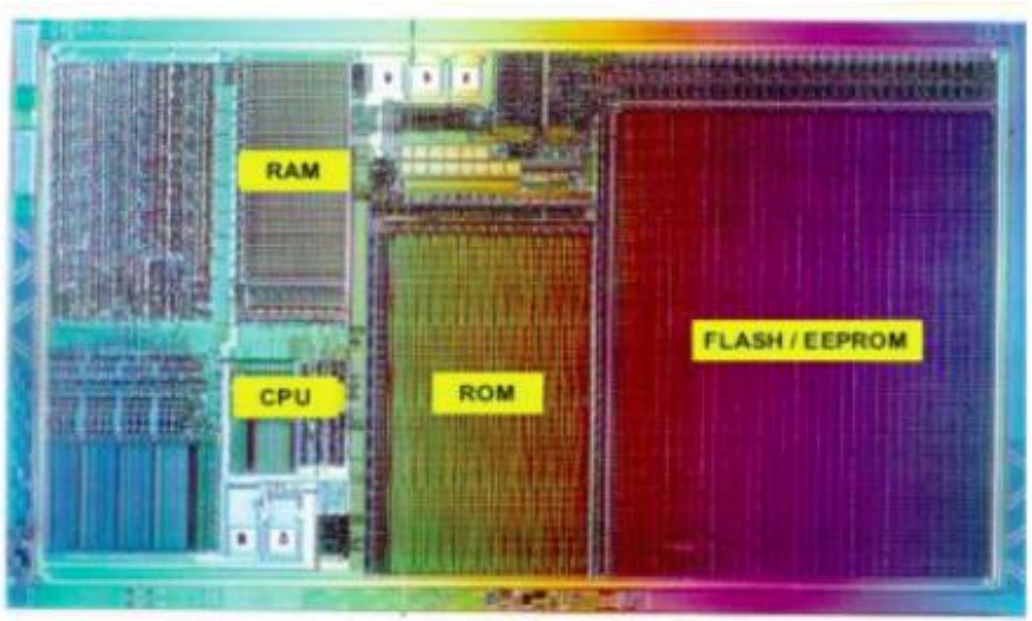
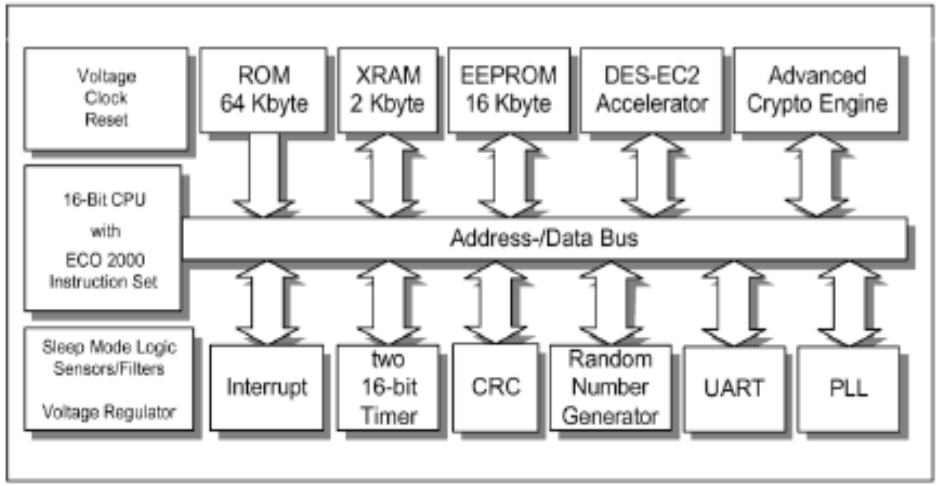
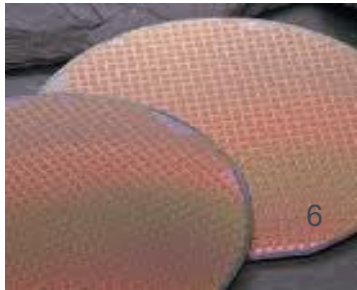


Figure : Smart card IC picture

Integrated Circuit Architecture

- Processor:
  - 8, 16 or 32 bits
  - CISC or RISC
  - Harvard or Von Neumann
- ROM: Read Only Memory
- RAM: Random Access Memory
- EEPROM: Electrically Erasable Programmable Read Only Memory
- Flash: NOR or NAND – close to EEPROM
- RNG: Random Number Generator
- Coprocessors for:
  - TDES, AES
  - CRC, HASH
  - Long Integer Arithmetic (RSA, ECDH, ECDSA)
- In/Out IPs
- MPU/MMU: Memory Management Unit
- Security sensors





life.augmented

Product Conception Phases

Tests
Production

Préparation

Circuit Intégré
Développement

OS
Développement

CRYPTO
Développement

Applications
Développement

Sécurisation

Produit

Personnalisation

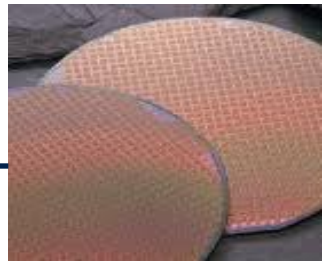
Pré
personnalisation

Pré
personnalisation

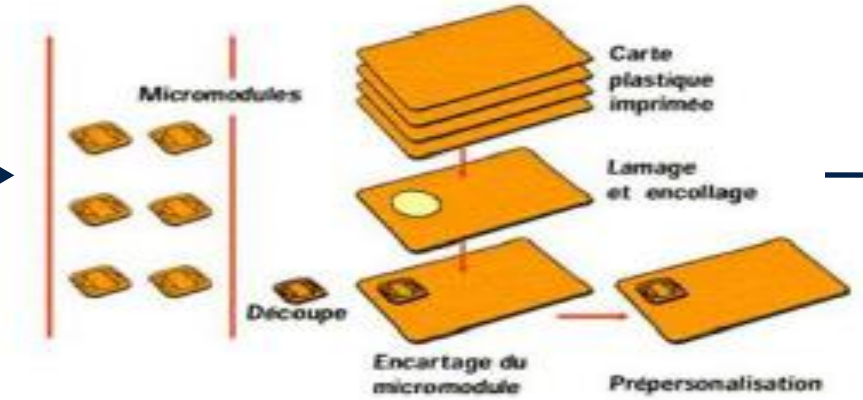
Initialisation

Tests

Fabrication
Photomasks
Fonderie



Different Phases
From User to End-Of-Life cycles



Mise en circulation

- Traitement des commandes
- Mises à jour (modifications données)

Gestion du Parc

- Remplacements
- Vols, oppositions, pertes

Fin de vie

- Désactivation,
- Usure fonctionnelle
- Casse

life.augmented

Standard Crypto Algorithms in Products

- Symmetric
  - DES/TDES: NIST recommendation now TDES EDE3 for 2^{40} use of same key
  - AES 128-192-256
  - SM4, SEED, FEAL-32
- Hash Functions
  - SHA-1, SHA-256
  - RIPEMD 160
  - SHA-3: KECCAK
  - HMAC Functions
  - SM3
- Asymmetric
  - CRT-RSA – RSA SFM (≥ 2048 bit)
  - DSA / ECDSA (i.e. GF-P256)
  - DH / ECDH
  - OBKG, Kgen
  - SM2

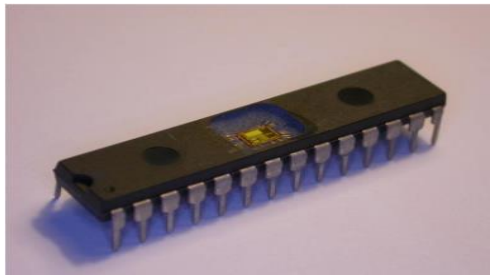
And also …

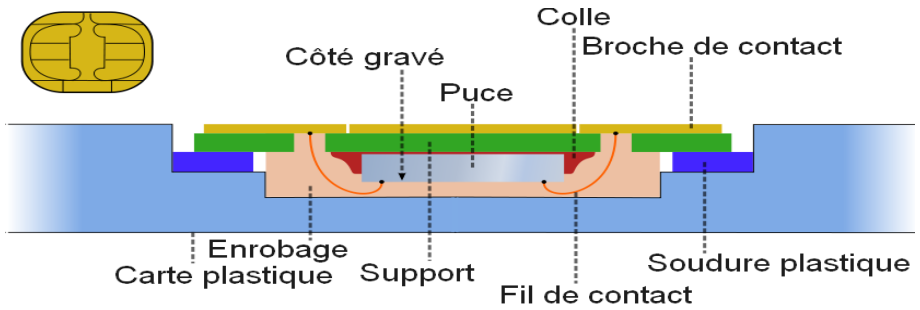
- DRBG, example: AES-CTR based i.e. NIST SP800-90
- Stream Ciphers
- Lightweight crypto
- Post Quantum Cryptography

life.augmented

Reverse Engineering

- Accessing to the circuit
  - Remove plastic or open with a knife or cutter





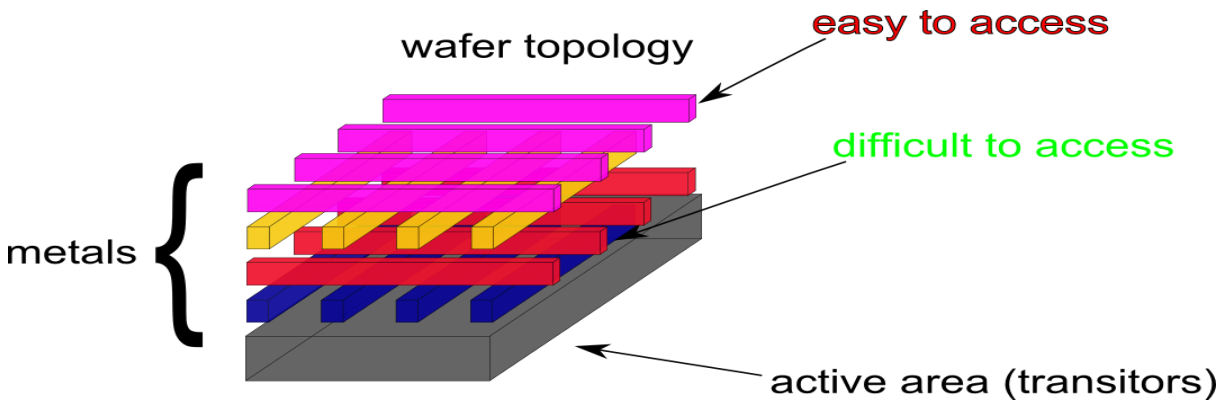
  - Use chemical products to remove (destroy) epoxy (resin) with nitric acid and acetone …
    → We observe now the chip that we can analyze using microscope(s)
  - Reconnect the circuit on another ‘’package’’
    - In order to use it again we reconnect the IO pins, clock pins, VCC pads, etc.
  - We can then analyze the IC when running with several tools

life.augmented

Reverse Engineering

Reverse the IC Layout

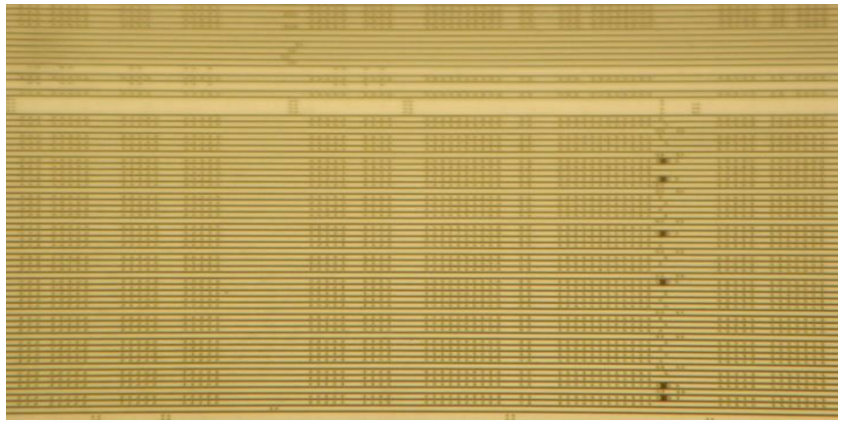
- Re-assemble metal layers
  - Integrated Circuit = SUM of metal layers interconnected each others



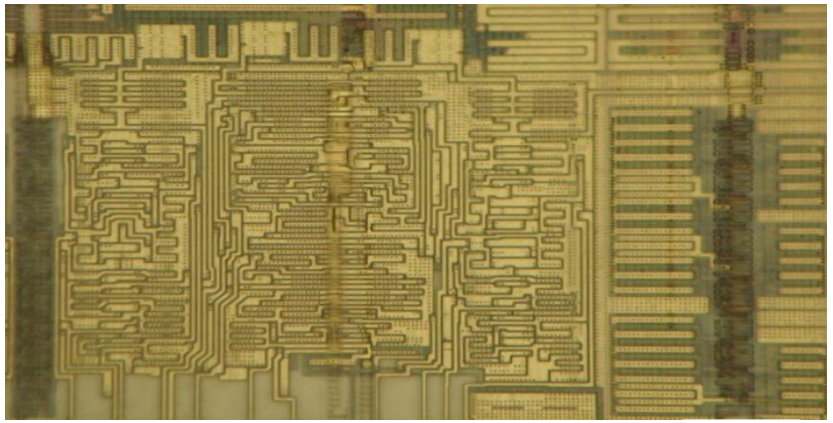
  - Using an optical microscope we can observe and take a picture of each metal layer
    - We start at the higher (upper) layer
    - We remove chemically this layer with fluoric acid or a plasma machine
    - We iterate until we reach the lowest layer (metal 1)

Reverse Engineering

Reverse the Layout



View of top métal



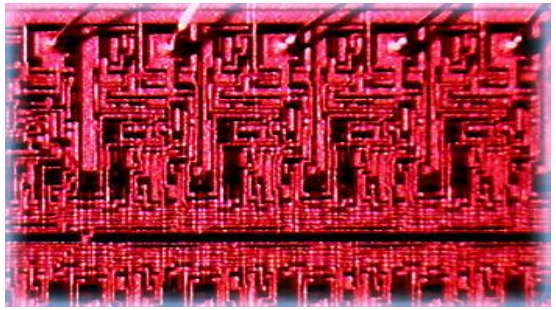
View of 1 metal lower after removal of top metals

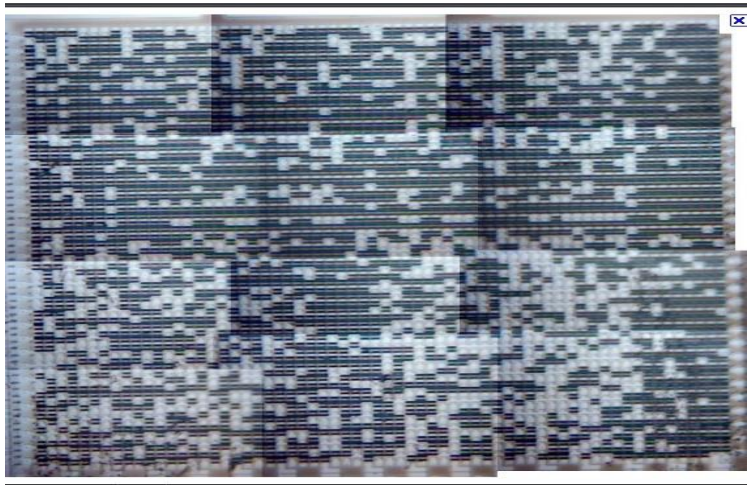
- We can the reconstitute the IC whole structure
- This operation is very complex and time consuming
  - In real life it is faster to focus on area of interest with a precise attack path

life.augmented

Reverse Engineering
Read Memories Content

- Extract (read) the not volatile memories content (data, code).
- RAM, EEPROM, FLASH : difficult but possible
- ROM:
  - Easier
  - Related to the kind of memory (diffused or not)
  - Require different techniques
- Depend to the technological node:
  - 90nm, 65nm
  - 28nm (!)
  - 7nm is the current best techno

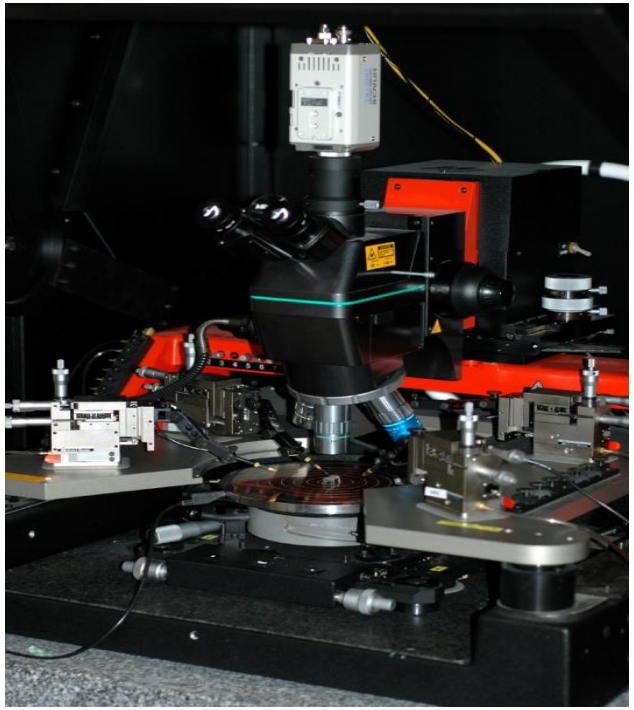


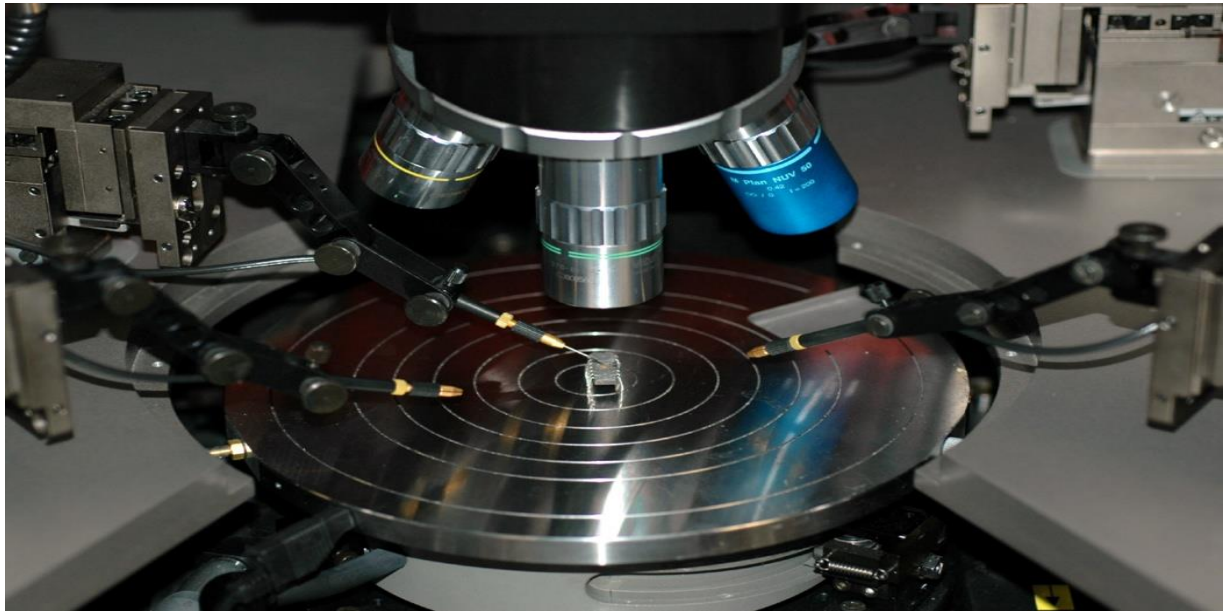


life.augmented

Read/Modify bits with Probing

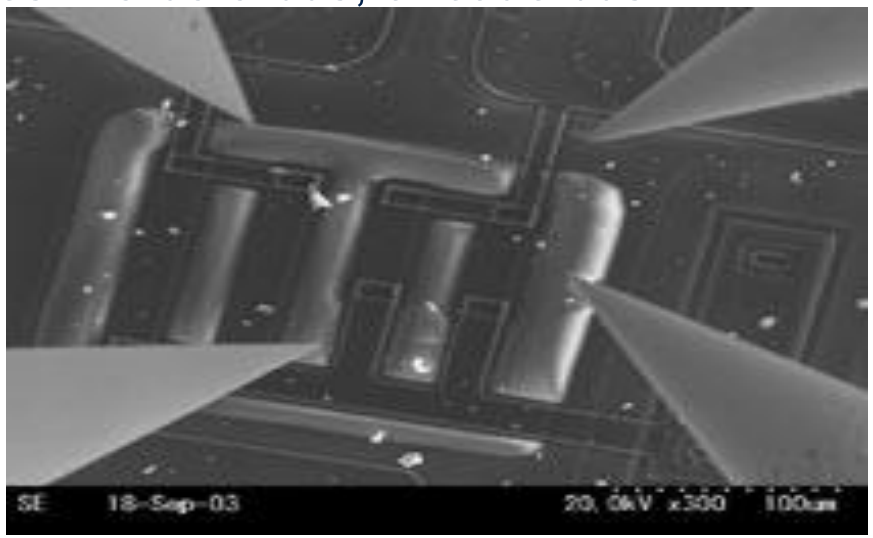
- Require a probe station





Read/Modify bits with Probing

- We position one or more probes on a circuit area to ‘’read’’ or ‘’change/flip’’ a value (bits):
  - For instance, the bit lines in a data bus, or code bus



- For instance, you can read a key value transferred on a bus to a memory or register
- You change the key bits ..
- Etc.

life.augmented

Modify the Circuit
Focused Ion Beam (FIB)

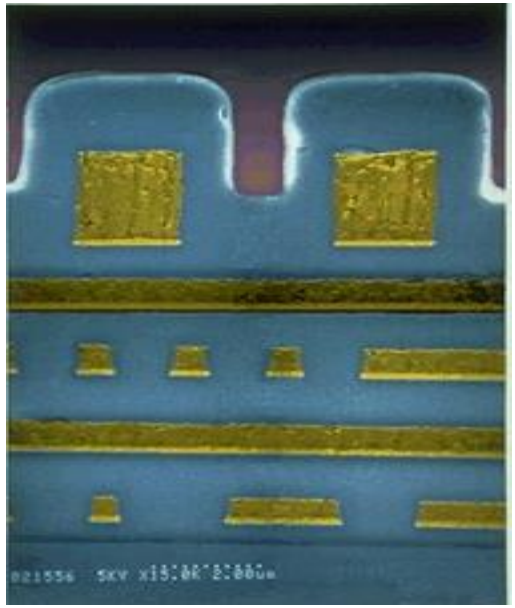
- FIB (Focused Ion Beam) stations are most often used to analyze (and “debug”) failures in circuits

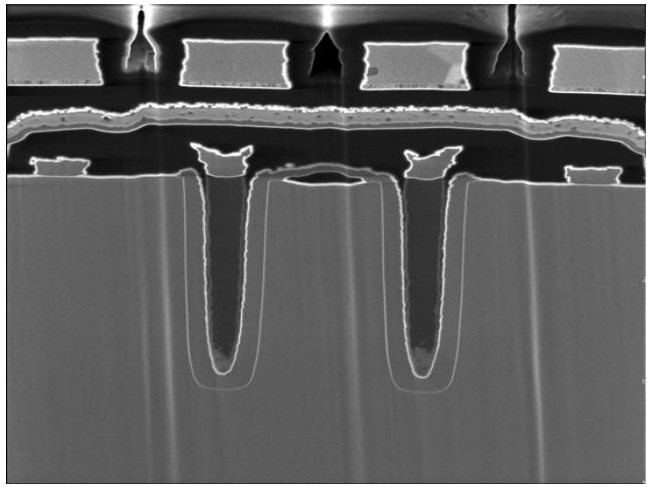
- FIB can
  - ‘’remove” metal, and modify connections
  - He can add metal and then create new connections

- Then an ATTACKER can
  - Access to “hidden” signals at lower metal levels
  - Modify the circuit (behavior)

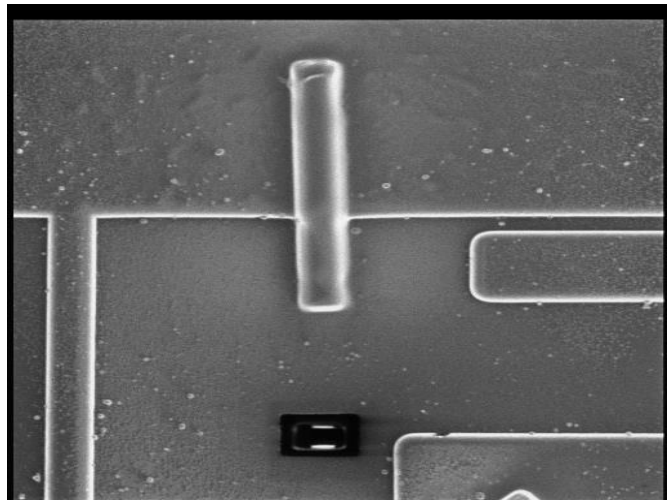
Very expensive .. But low cost for renting !

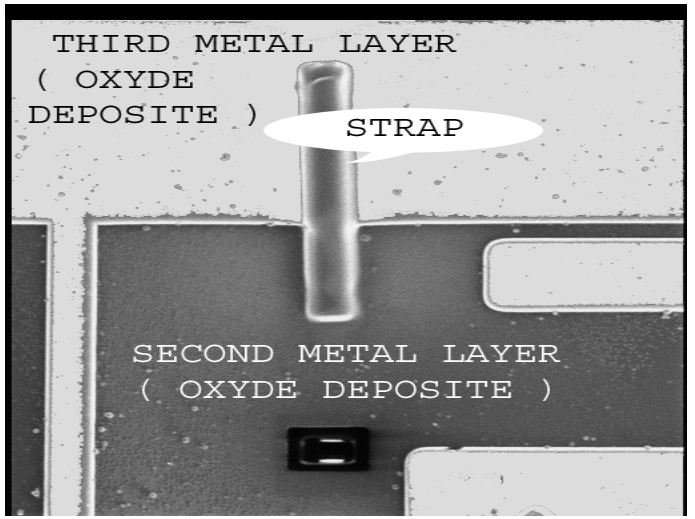
Modify the Circuit
Focused Ion Beam (FIB)





Cross Sections

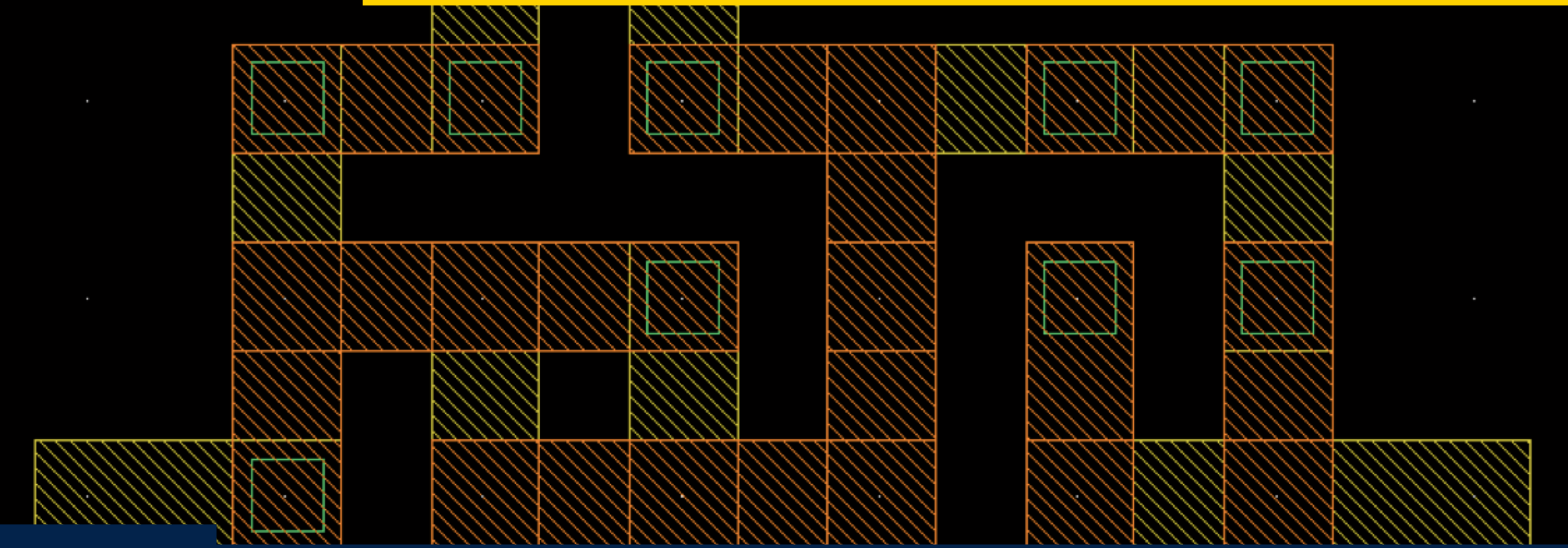




- Very powerful tool
- Can defeat product security
- Expensive
- Can be rent
- Usage to be be monitored

life.augmented

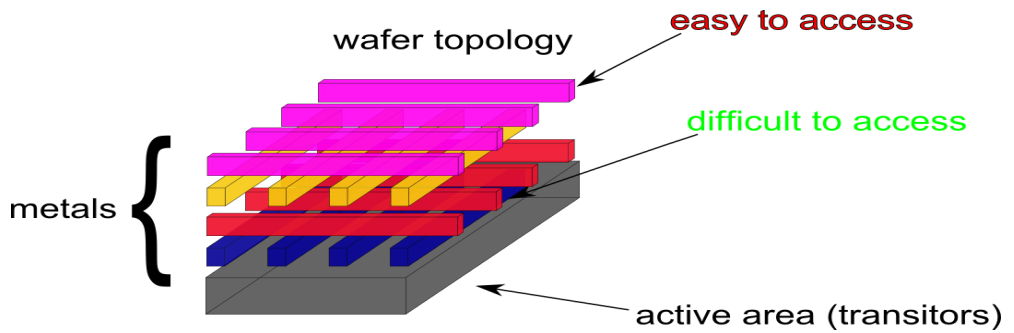
Countermeasures Vs. Invasive Attacks



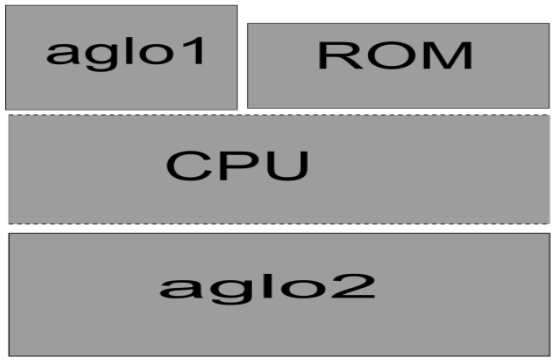
life.augmented

IC Reverse Engineering Countermeasures

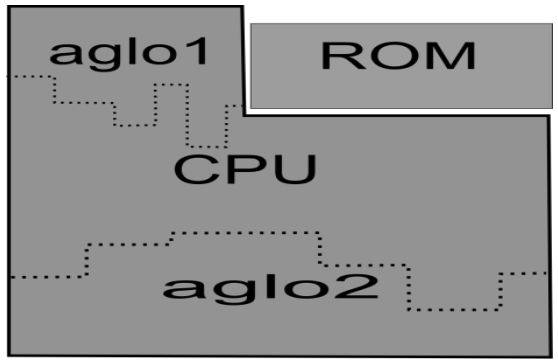
- Make design more complex
  - Embed sensitive lines in lower levels
  - Use thinner technologies (< 65 nm)
  - Make the structure more compact (i.e.. Glue logic)



- Glue Logic
  - Makes the IP and block localization difficult to the attacker



IP easy to locate



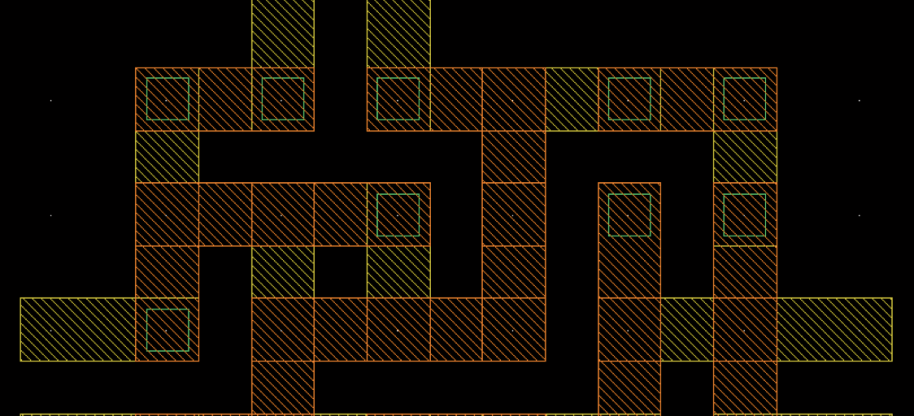
IP more difficult to locate

Reverse Engineering on Memories Countermeasures

- Use memory technology more difficult to reverse
- Apply Scrambling / Permutations on data (addresses)
  - Apply secret permutation to the data bits to make difficult the bit assembly once the memory has been reversed
  - Can be software or hardware dedicated IPs
- Hardware Data Encryption
  - Data in memory are never in plain
  - Encrypted with a cryptographic algorithm
  - Must be very efficient as applied for each READ/WRITE access in memory
    - Decryption for each memory read
    - Encryption for each memory write
    - Same for registers, etc.
  - Difficult to design such algorithms (not a XOR with a hardcoded key please)
  - Mandatory for serious products
- Software Encryption
  - Is recommended to be added at software level for (very) sensitive assets

Probing Countermeasures

- Passive Shield *(insufficient)*
  - Full metal level on the top of the IC to “forbid” the access
- Active Shield/Mesh *(efficient)*
  - Full metal on top of the shield
  - Embed active lines interconnected
  - Touch or Cut a line and you will be detected by the shield sensors connected to the active lines
  - https://hal.science/hal-00721569v2/document
- Be a Clever Designer
  - place sensitive buses, registers (data or configuration) in area difficult to access to a probe
- Integrity Check on data flow
  - Will detect a probe modification on data as integrity will not be coherent
- Memories/Buses/Registers Encryption



FIB Countermeasures

Active Shield/Mesh *(efficient)*

- Full metal on top of the shield
- Embed active lines interconnected
- Touch or Cut a line and you will be detected by the shield sensors connected to the active lines

Be a Clever Designer

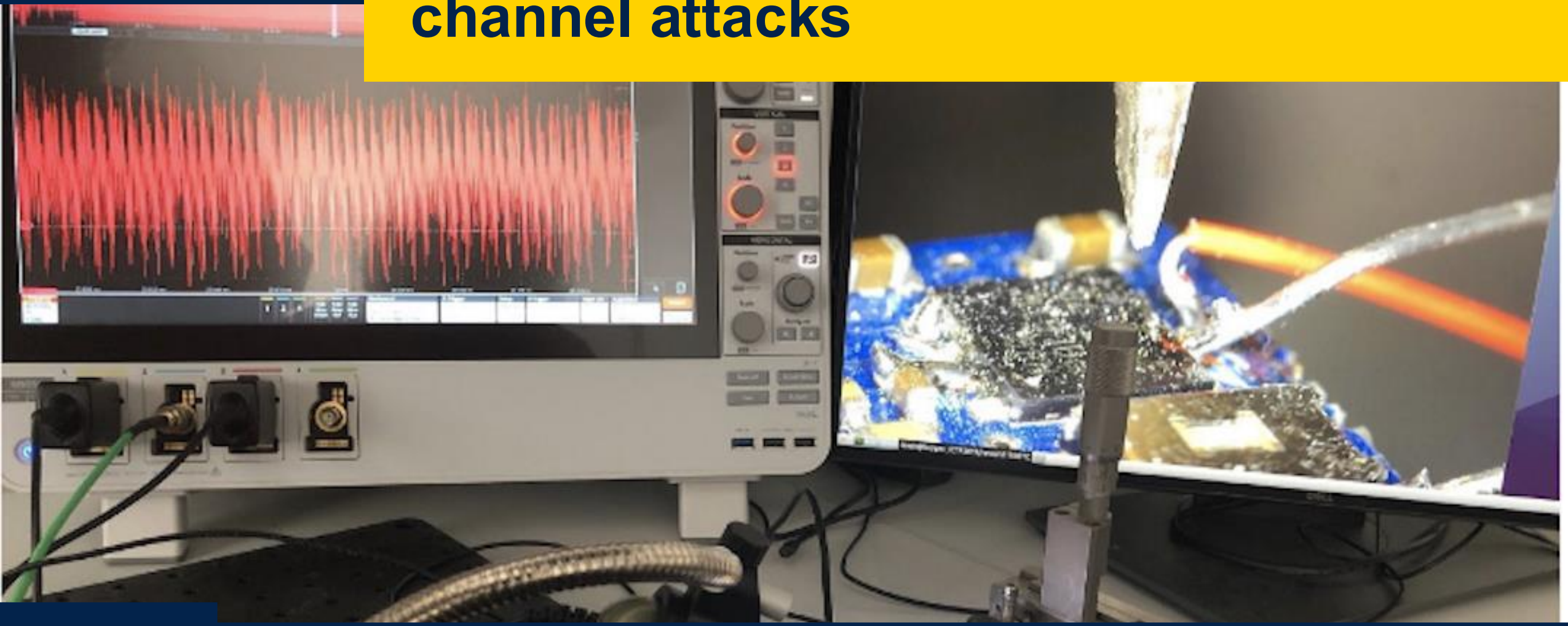
- place sensitive buses, registers (data or configuration) in area difficult to access to a probe

Difficult

- With more and more efforts FIB can counterfeit most of the protection
- Matter of time and money and techno used

life.augmented

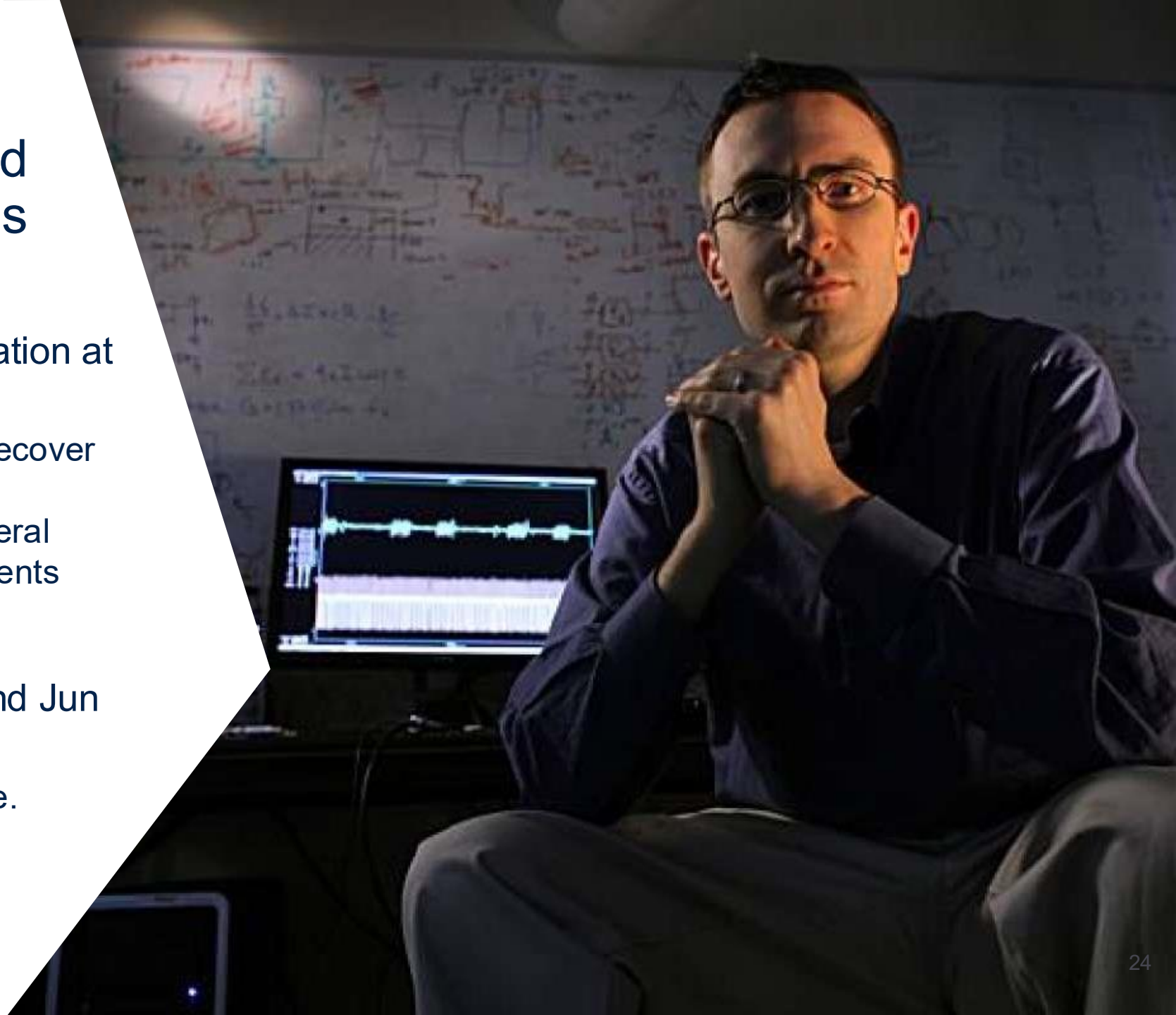
96’s: (well?)coming timing and side-channel attacks



Timing Attack and Simple Power Analysis

- 1996 – P. Kocher first publication at Crypto conference
  - Exploited timing analysis to recover the secret key used in RSA exponentiation thanks to several executions timing measurements

- 1998 – P. Kocher, B. Jaffe and Jun – SPA on RSA.
  - Published on the CRI website.

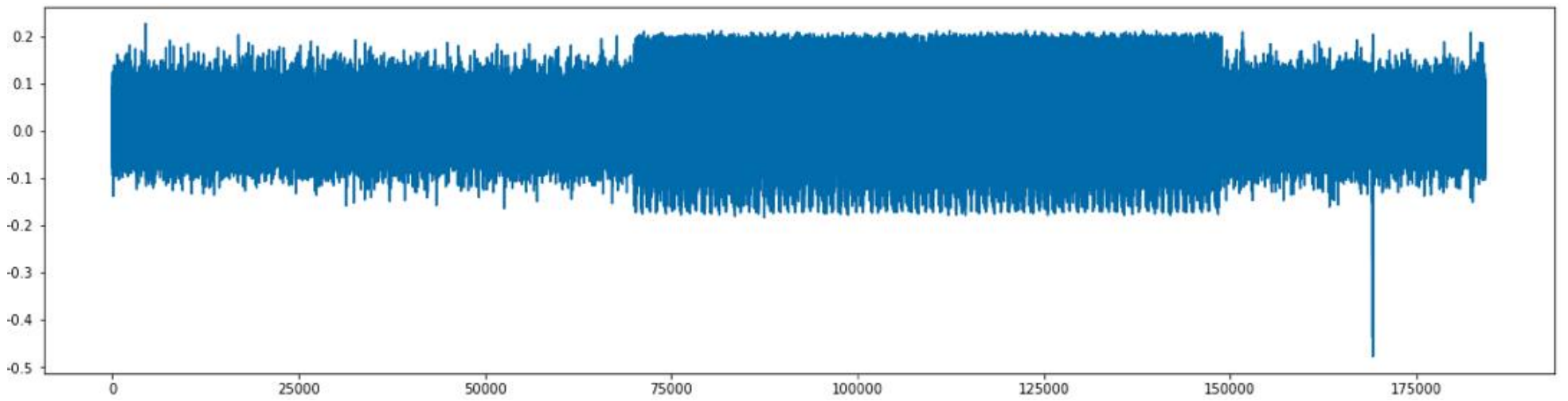


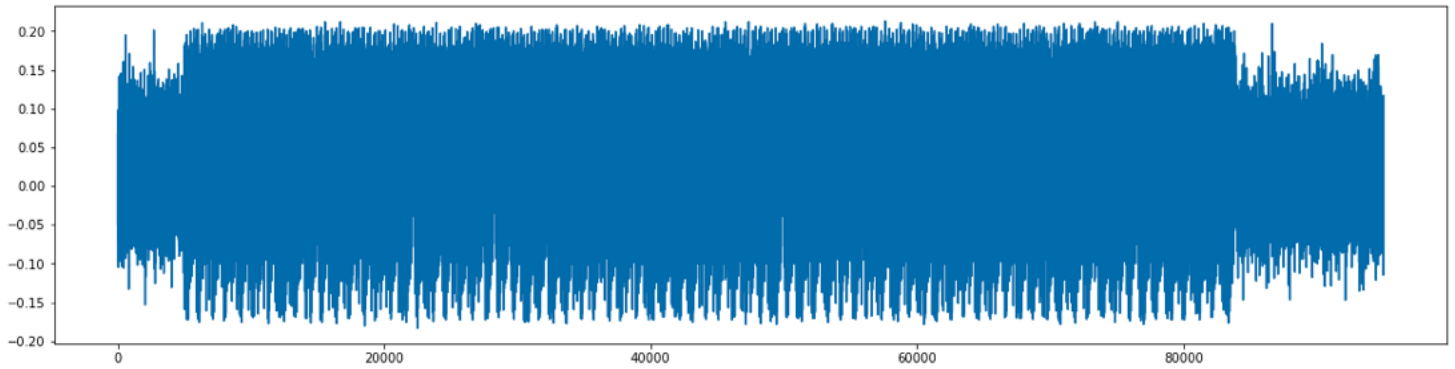
life.augmented

Some side-channel history

- 1999 – P. Kocher, J. Jaffe and B. Jun – first DPA publication on DES
  - Paper: Differential Power Analysis. Crypto 1999.
- 1999 – T.S. Messerges, E.A. Dabbish and R.H Sloan – DPA on RSA
  - Power Analysis of Modular Exponentiation in Smartcards. CHES 1999.
- 2000 – T.S Messerges – Second Order Power Analysis
  - Using Second-Order Power Analysis to Attack DPA Resistant Software. CHES 2000.
- 2003 – E. Brier, C .Clavier and F. Olivier: CPA
  - Correlation Power Analysis with a Leakage Model CHES 2004 (IACR 2003).
- 2008 – B. Gierlichs, L. Batina, P. Tuyls and B. Preneel: MIA
  - Mutual Information Analysis. CHES 2008
- 2010 – A. Moradi, O. Mischke, and T. Eisenbarth: Collision Correlation on AES
  - Correlation-Enhanced Power Analysis Collision Attack. CHES 2010

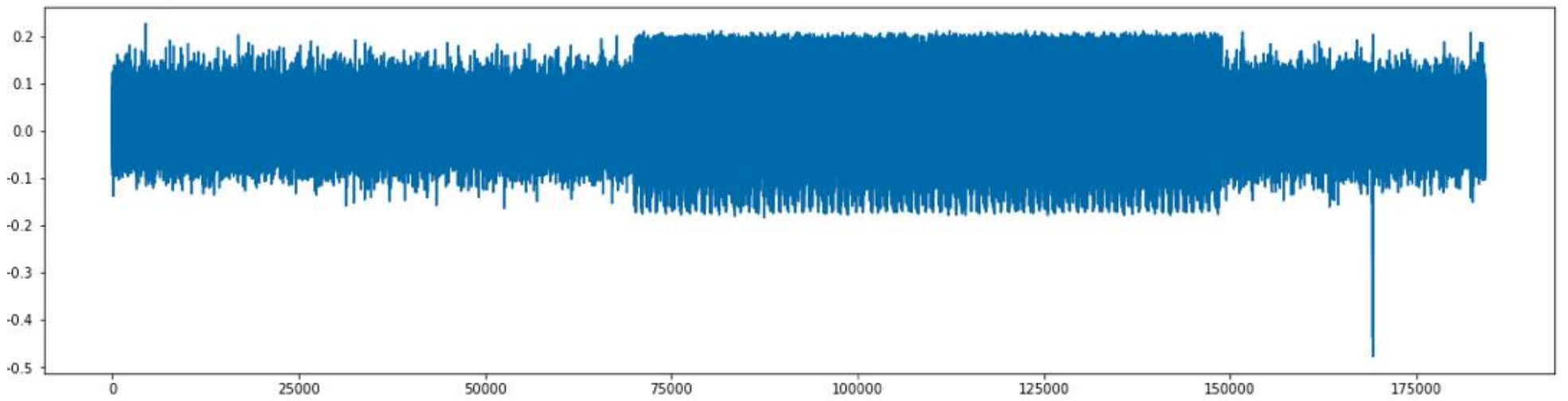
What is this?





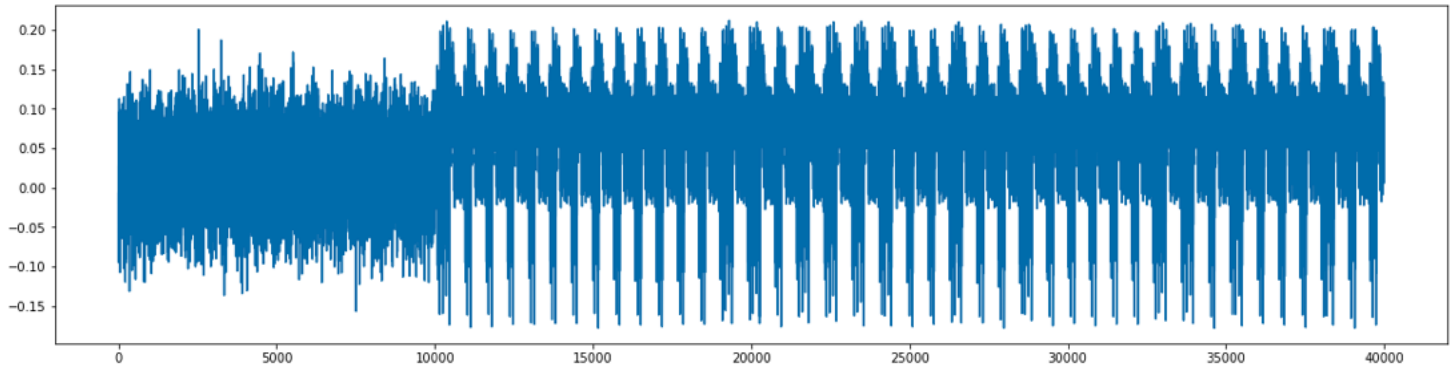
life.augmented

What is this?



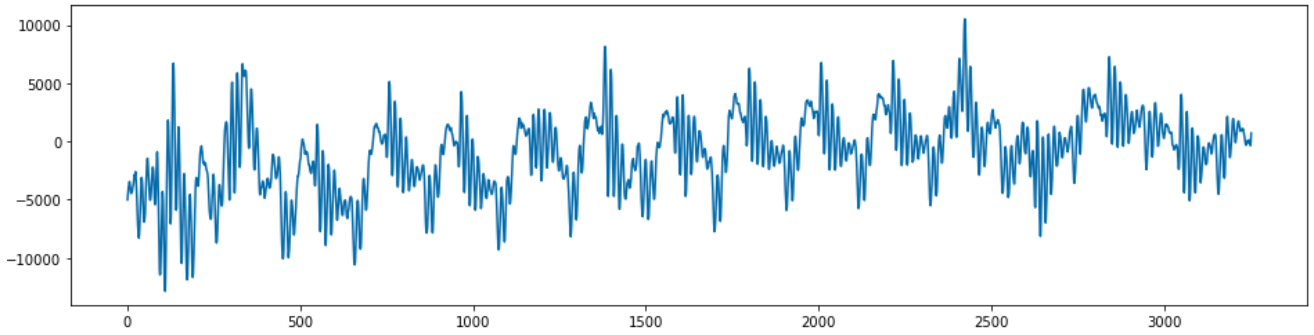
Zooming you identify loop of multiplications

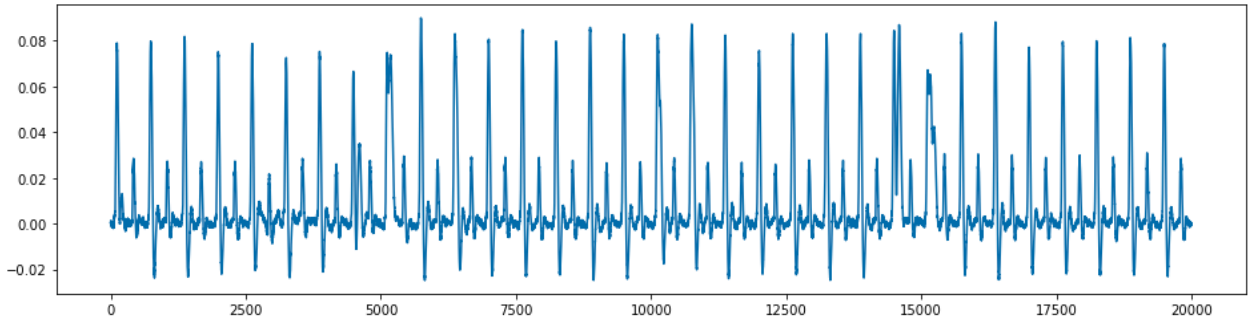
→ RSA exponentiation



life.augmented

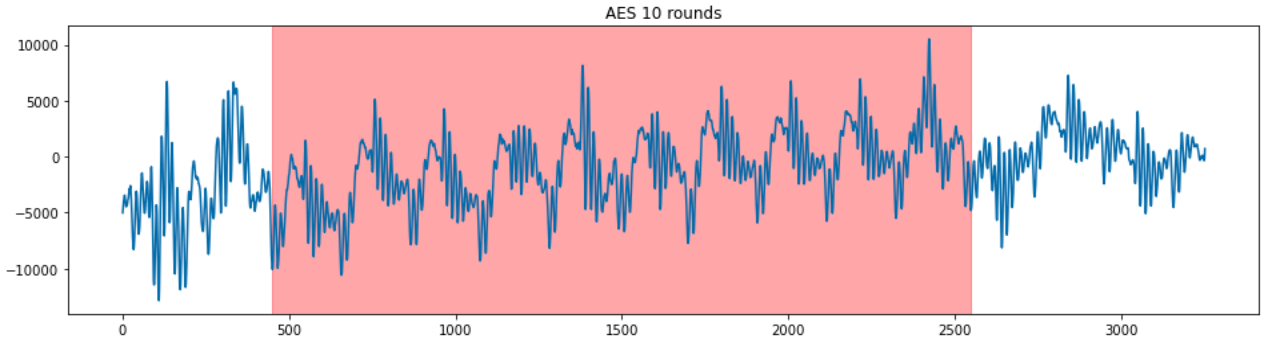
These ones?

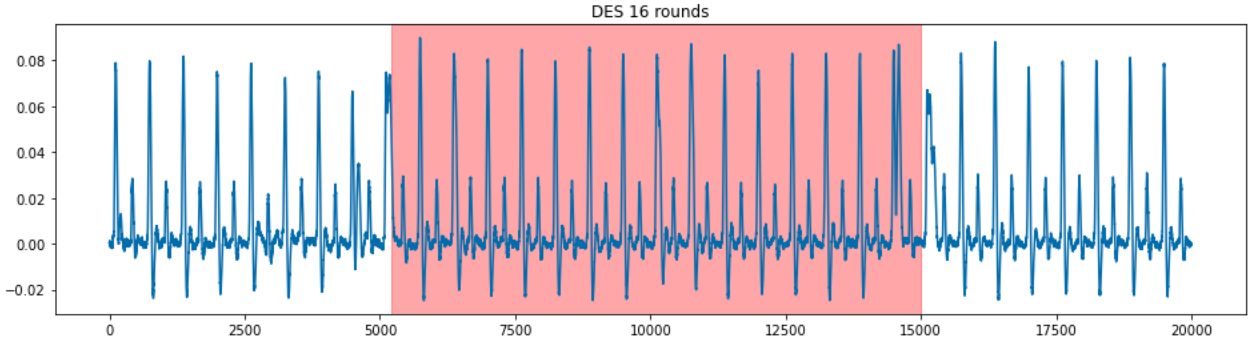




life.augmented

These ones?





life.augmented

Triple DES

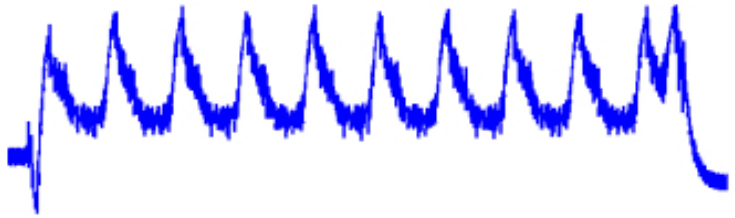
life.augmented

Good AES Example Reading

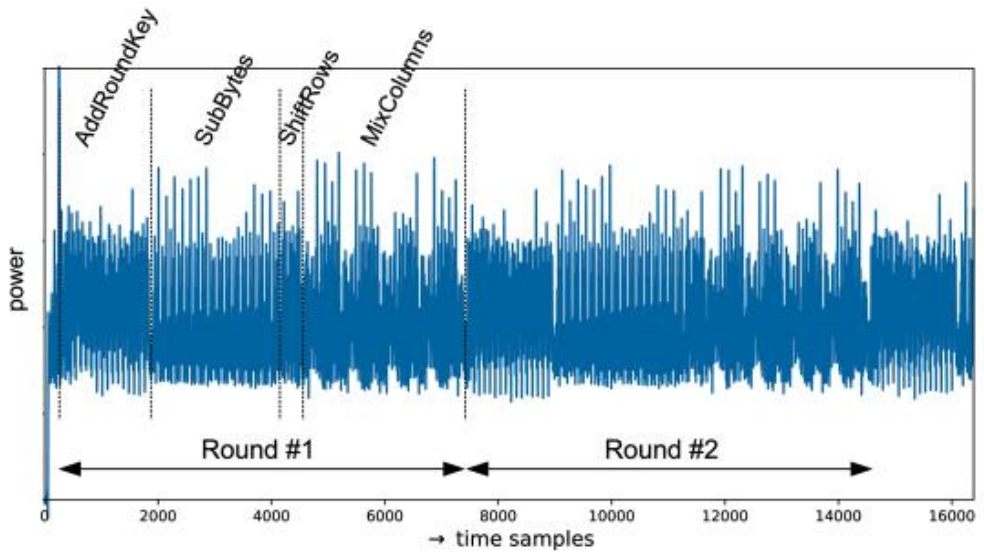
From Francois Durvaux

SCA-Pitaya: A Practical and Affordable Side-Channel Attack Setup for Power Leakage–Based Evaluations

https://dl.acm.org/doi/fullHtml/10.1145/3371393



Instantaneous power consumption of an AES- 128 execution



life.augmented

Side-channel Leakage Origin

- Devices are done using CMOS technology where the device power consumption varies depending on:
  - The type of operation (code) executed
  - The data manipulated by the hardware

- Leakage models mostly considered in literature and practice
  - Hamming weight: power consumption = F(data)
  - Hamming distance: power consumption = F(reference_state XOR data)
  - Single bit or multi-bit
  - Direct value

Remember RSA

Public Key

modulus n
public exponent e
plaintext M

Private Key

secret exponent d

Private Key CRT

prime p
prime q
exponent d_p
exponent d_q
Inverse i_q

Signature Verification

$M \stackrel{?}{=} S^e \bmod n$

Signature Generation

$S = M^d \bmod n$

Signature Generation with CRT mode

$M_p = M \bmod p$
$M_q = M \bmod q$
$S_p = M_p^{dp} \bmod p$
$S_q = M_q^{dq} \bmod q$
$S = [(S_p - S_q) \times i_q \bmod p] \times q + S_q$

life.augmented

Implementing RSA (and ECC)

- CRT RSA: modular exponentiation M^d mod n is replaced by two exponentiations of half size.
- As the complexity of this operation is in $O(\ell^3)$ (with ℓ the bit-length of n) a half size exponentiation is 8 times faster.
- The CRT-RSA is then theoretically about 4 times faster than the non-CRT exponentiation.
- In practice, the efficiency gain will be closer to 3 than 4 due to the reduction and the recombination operations.
- Most of the chip manufacturers include an arithmetic long integer accelerator, also said public key coprocessor, to compute modular multiplication operations more efficiently.
- Depending on the manufacturer, the choice of the modular arithmetic can vary (i.e., Montgomery, Barrett, Quisquater).
- It enables efficient implementations of RSA, CRT-RSA, DSA, DH schemes.
- This coprocessor is also useful to implement efficient elliptic curves operations in ECDSA or ECDH schemes.

Classical Exponentiation Square and Multiply

Input: $m, n \in \mathbb{N}$, $m < n$, $d = (d_{k-1} d_{k-2} \dots d_0)_2$
Output: $m^d \mod n$

1. $a \leftarrow 1$
2. **for** $i = k - 1$ **to** 0 **do**
3. $\quad a \leftarrow a^2 \mod n$
4. $\quad$ **if** $d_i = 1$ **then**
5. $\quad\quad a \leftarrow a \times m \mod n$
6. **return** a

Left-to-right Square-and-Multiply Exponentiation

Input: $m, n \in \mathbb{N}$, $m < n$, $d = (d_{k-1} d_{k-2} \dots d_0)_2$
Output: $m^d \mod n$

1. $a \leftarrow 1$; $b \leftarrow m$
2. **for** $i = 0$ **to** $k - 1$ **do**
3. $\quad$ **if** $d_i = 1$ **then**
4. $\quad\quad a \leftarrow a \times b \mod n$
5. $\quad b \leftarrow b^2 \mod n$
6. **return** a

Right-to-Left Square-and-Multiply Exponentiation

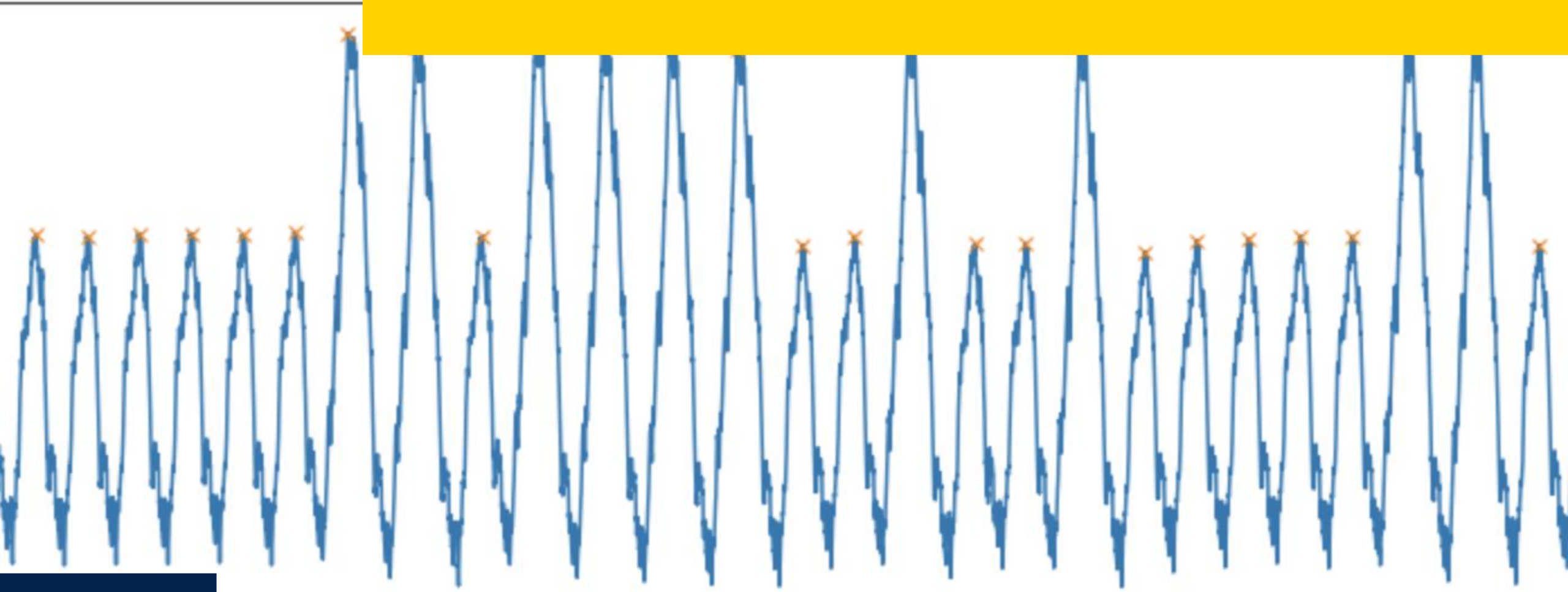
life.augmented

Modular Arithmetic Methods

- The implementation of long integer efficient modular squaring and multiplication operations is of crucial importance for product makers.
- Heavy constraints lie on these operations, especially in the context of embedded devices.
- Several methods can be found in the literature. For instance, some common ones are the following:
  - Montgomery method
  - Barrett method
  - Quisquater method
  - Brickell method
  - Sedlak method
  - Scholar method (Knuth)
- In practice the Montgomery and Barrett methods are the most efficient and most commonly used in most of the products.

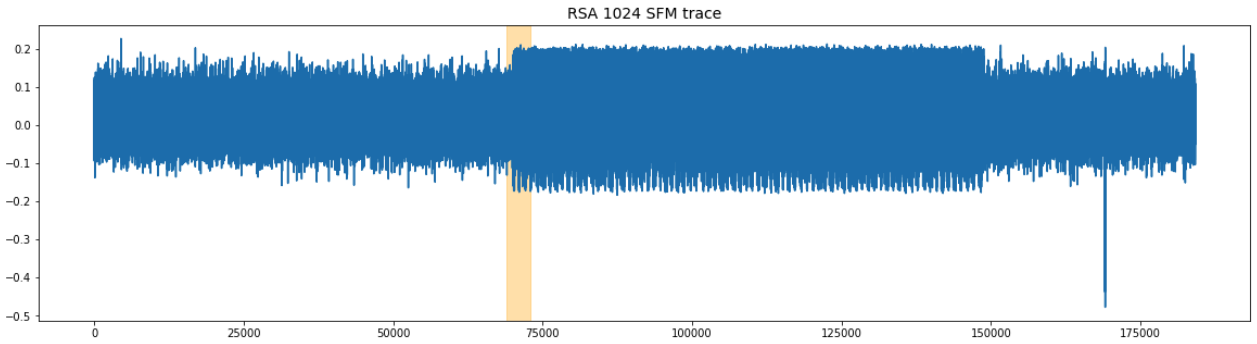
- Barrett, P. (1986). Implementing the Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor. In Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings, pages 311-323.
- Brickell, E. F. (1982). A fast modular multiplication algorithm with application to two key cryptography. In Advances in Cryptology: Proceedings of CRYPTO '82, Santa Barbara, California, USA, August 23-25, 1982, pages 51-60.
- Knuth, D. E. (1997). The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
- Montgomery, P. L. (1985). Modular multiplication without trial division. Mathematics of computation, 44(170):519-521.
- Quisquater, J.-J. (1992). Encoding system according to the so-called RSA method, by means of a microcontroller and arrangement implementing this system. US Patent 5,166,978.
- Sedlak, H. (1987). The RSA cryptography processor. In Advances in Cryptology - EUROCRYPT '87, Workshop on the Theory and Application of Cryptographic Techniques, Amsterdam, The Netherlands, April 13-15, 1987, Proceedings_, pages 95-105.

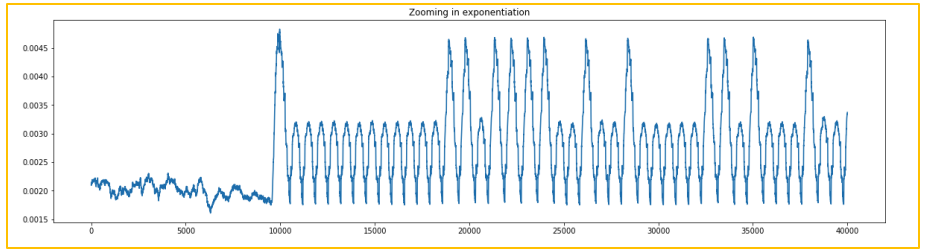
Simple Side Channel Analysis

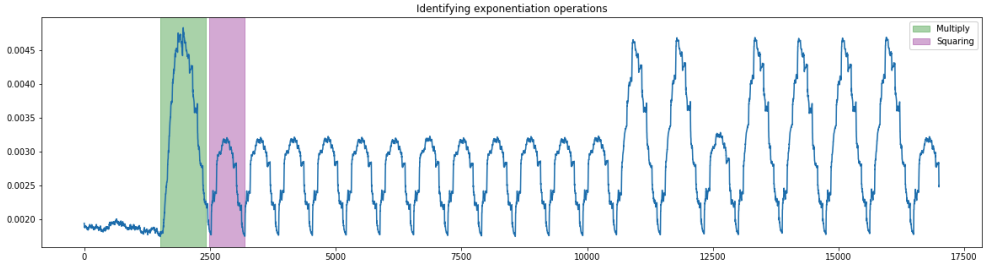


Demo/TP on SSCA on RSA challenge

- See demo on RSA SSTIC challenge 2019







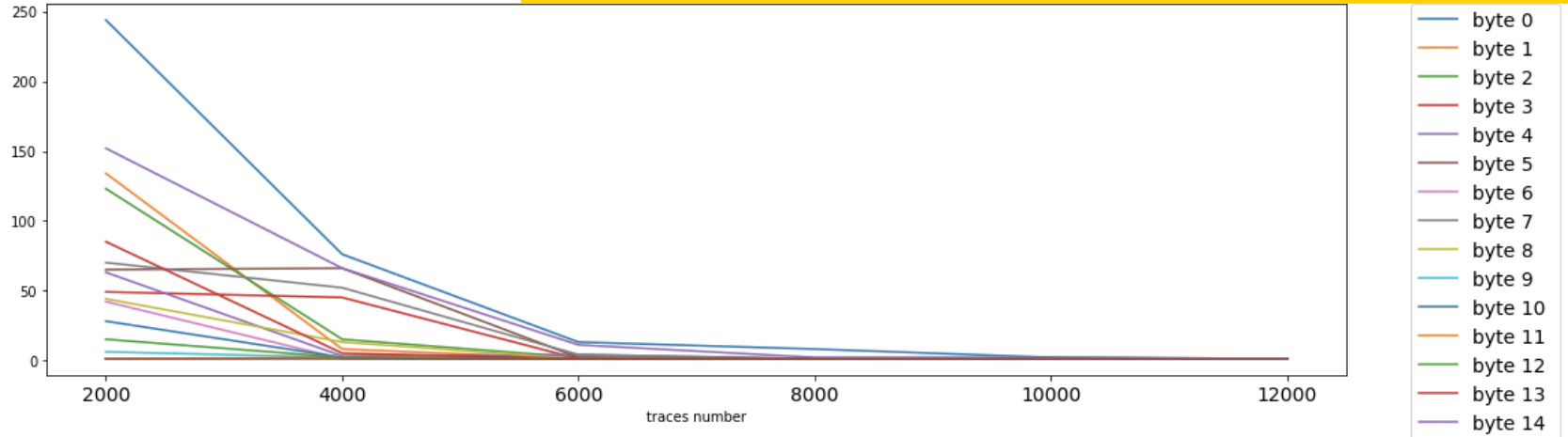
WITHOUT CLUSTERING

WITH CLUSTERING

https://scikit-leaxrn.org/stable/modules/clustering.html

Differential Side Channel Analysis

```
attack_result.plot_attack_ranks(rig
```




```
key_rank_lower, key_rank_est, key_rank_max = attack_result.remaining_entropy(right_key)
```

```
------------------------------------------------------------
Number of traces for full key recovery            =  12000
------------------------------------------------------------
Number of traces for 32-bit remaining entropy:
-- Higher bound entropy  =  6000
-- Lower bound entropy   =  6000
-- Estimated entropy     =  6000
------------------------------------------------------------
```

Guessing Entropy Graph

life.augmented

Tools

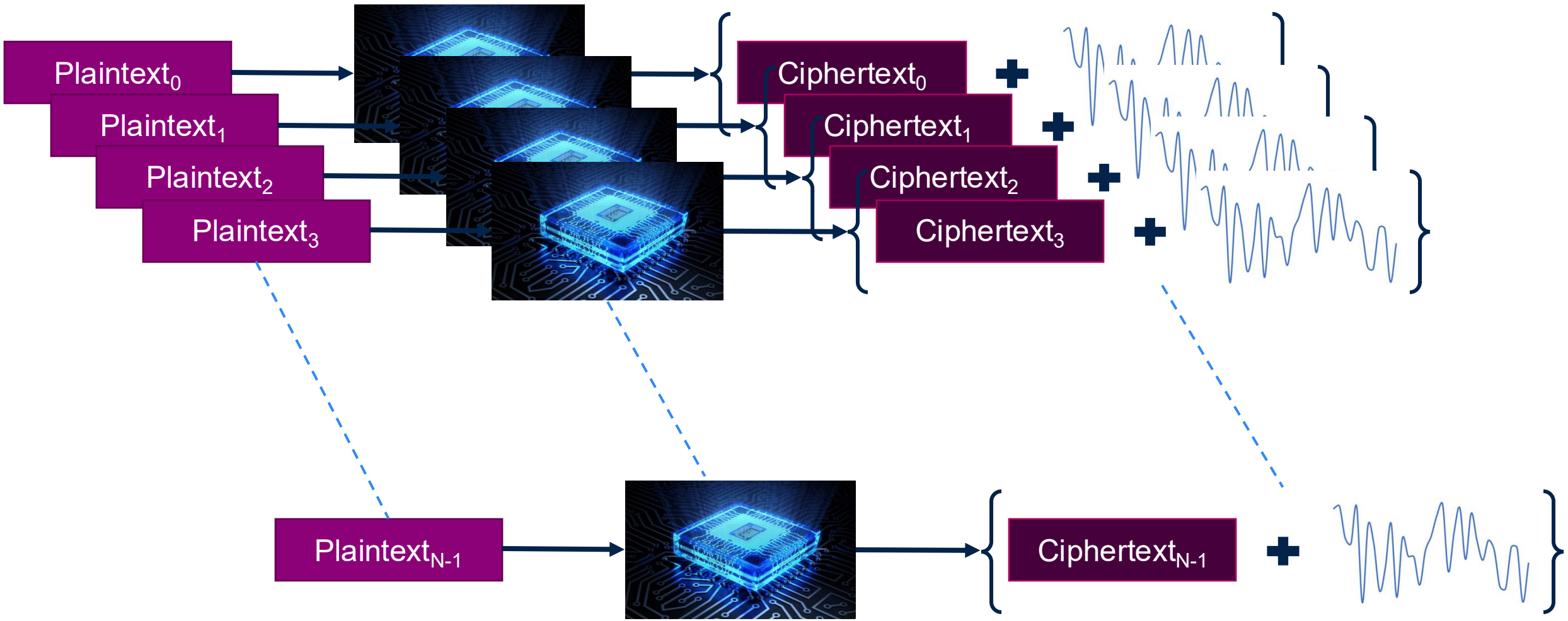
- Distinguisher
  - Statistical tool used to compare simulated set (from guess) and real traces set (from DUT)
  - DPA, CPA, SNR, NICV, MIA, ANOVA …
- Discriminant:
  - Function to apply to sort the distinguisher resulting scores
- Model:
  - Hamming weight, monobit, Hamming distance, value

Differential Side-Channel Analysis

- Run traces collection for cryptographic algorithm
- Run E_K (encryption) of N plaintexts P_0 .. P_{N-1} store related side-channel traces C_0 .. C_{N-1}.

$C_0 = E_K(P_0)$ … $C_{N-1} = E_K(P_{N-1})$

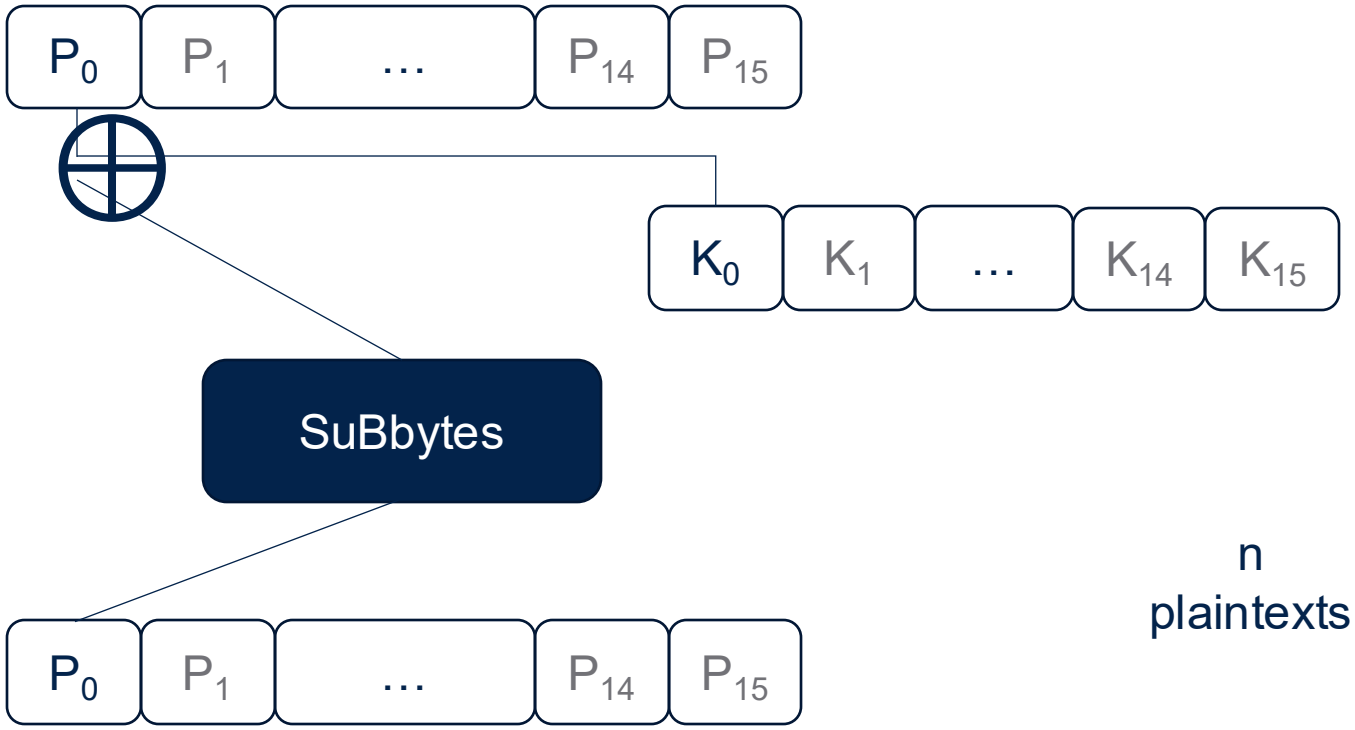
Side-Channel Analysis



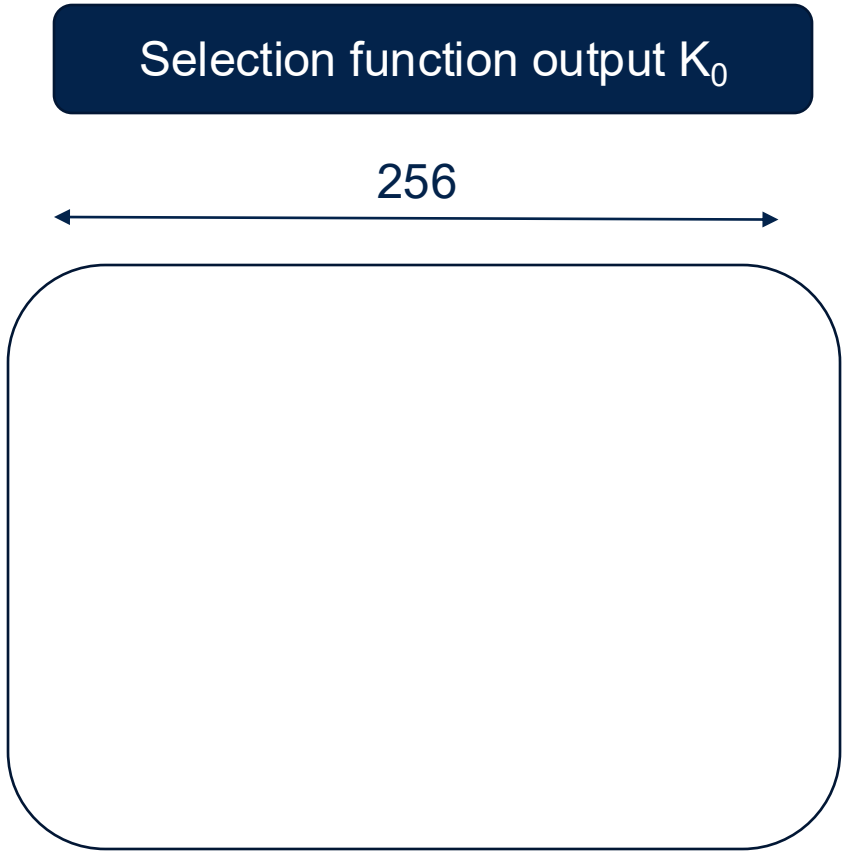
Differential Side-Channel Analysis

- Apply Divide-and-Conquer method on the operations and involved key-bits
  - Guess λ bit of key chunk: for instance, 6-bit for TDES SBoxes, 8-bit for AES ByteSub
  - Break down the problem = full key recovery to sub-problem = recover key per bit-chunk involved in sub operations
  - Define the computation targeted
  - **Selection function** = KeyAddition Substitution …. In first or last rounds

Selection Function
Key Byte 0



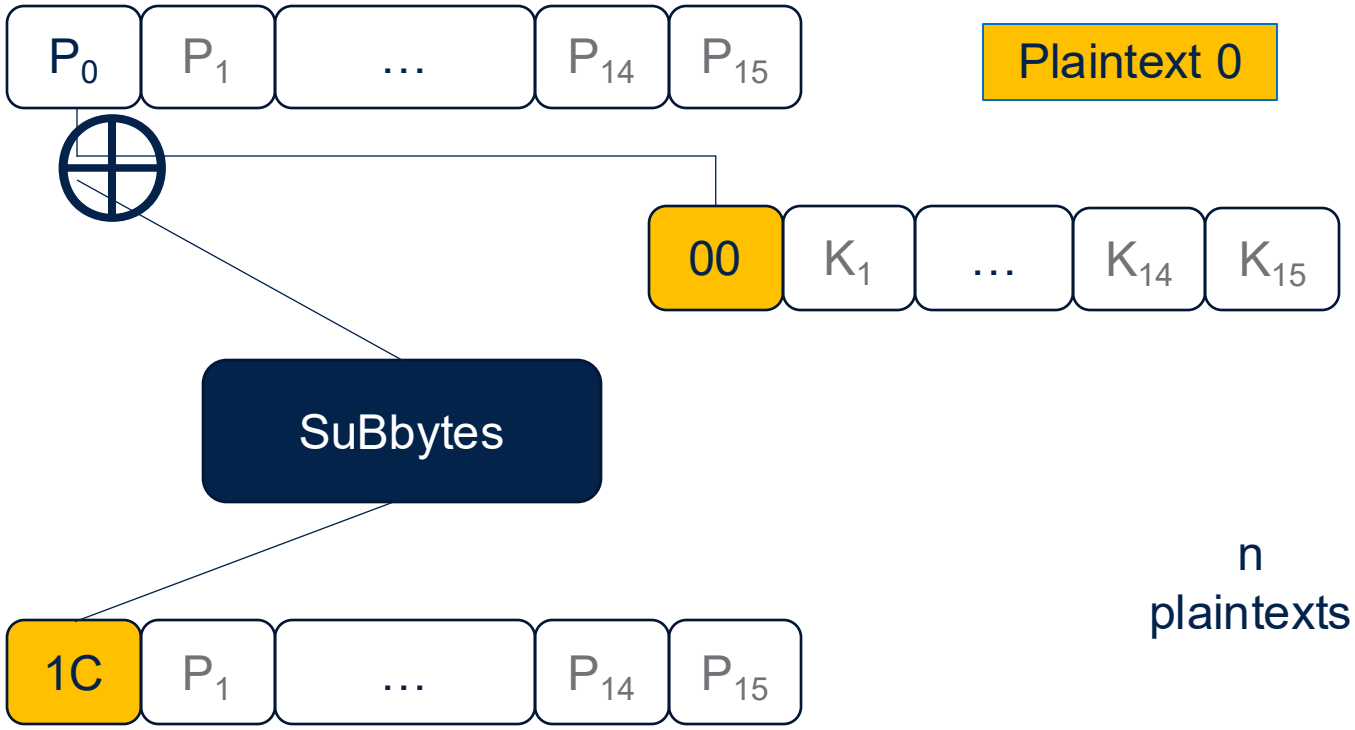
n
plaintexts



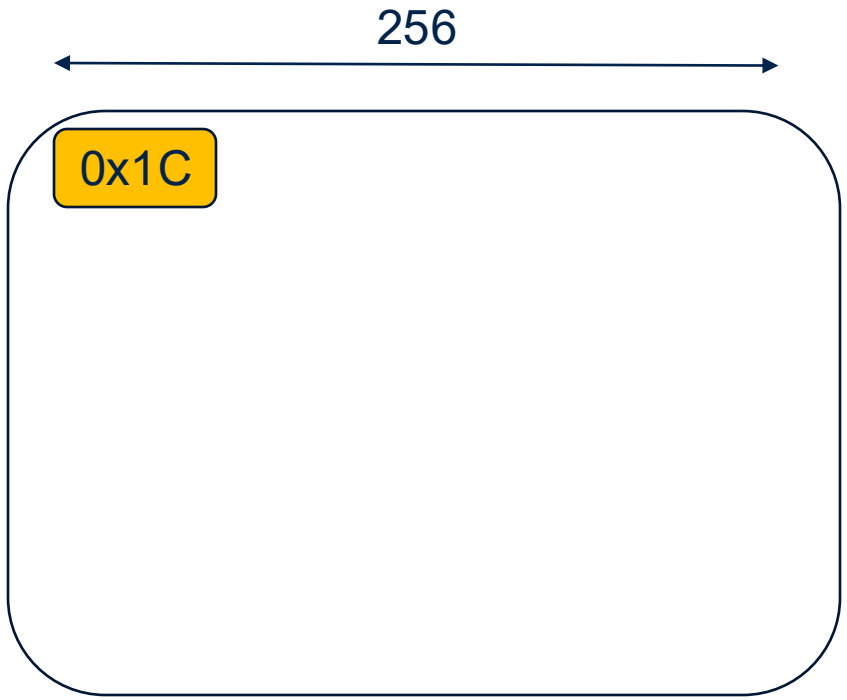
Guess 0 Guess 1 Guess 255

life.augmented

Selection Function
Key Byte 0



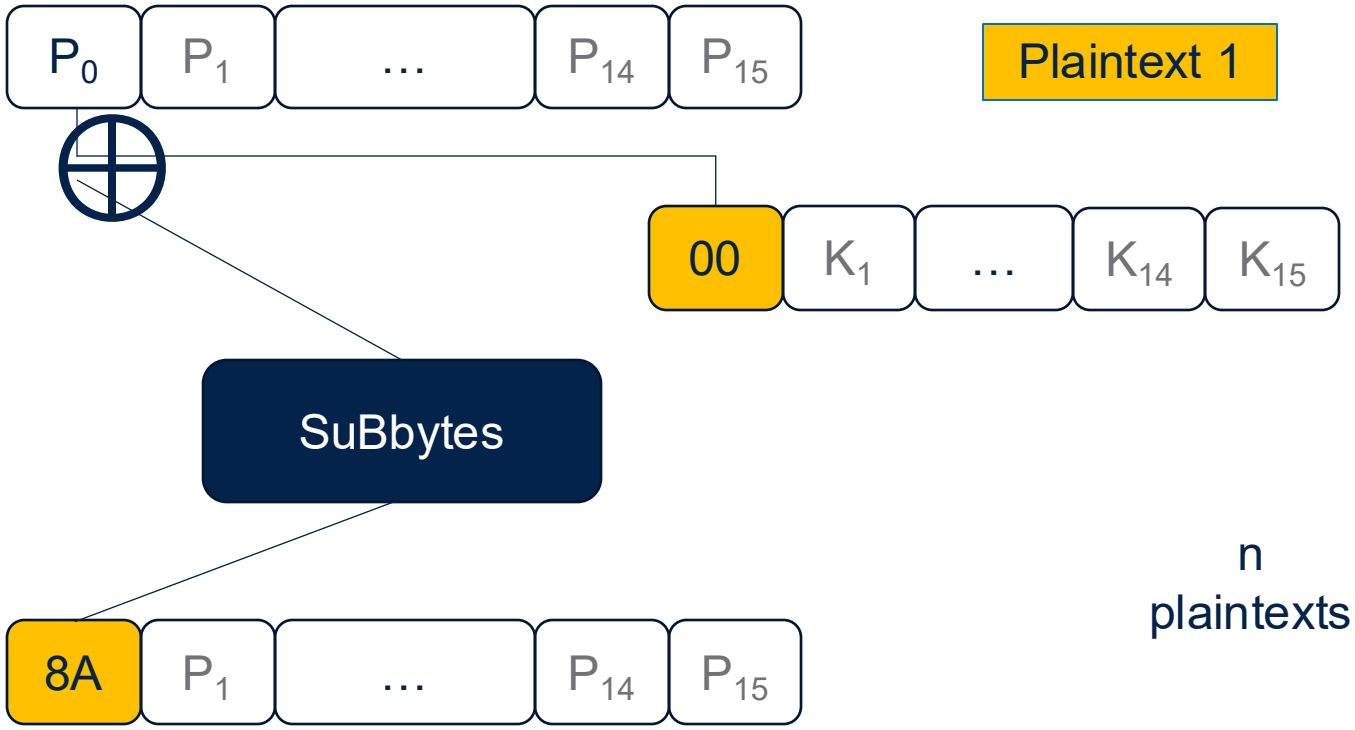
n
plaintexts



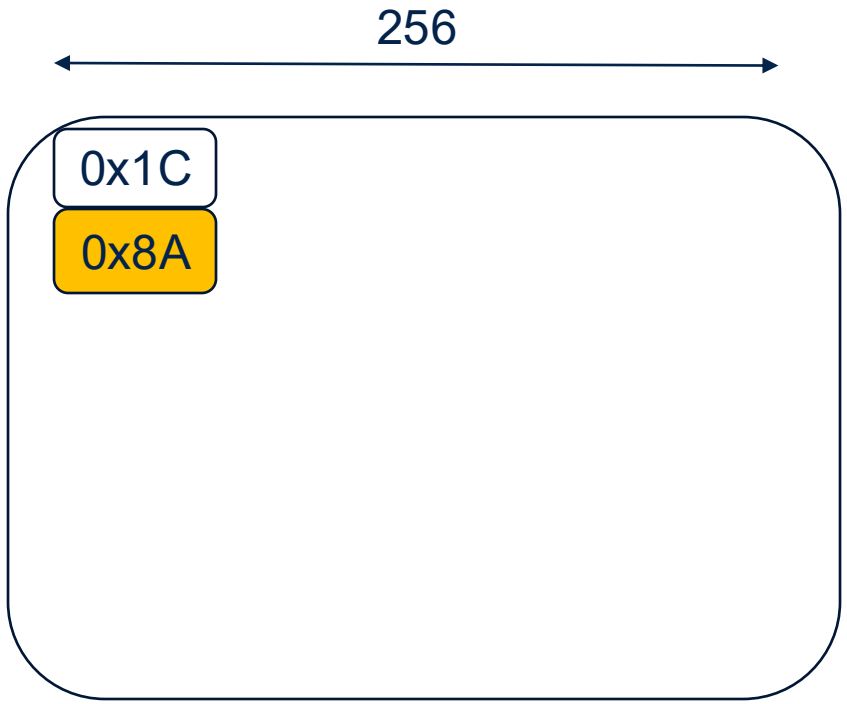
Guess 0 Guess 1 Guess 255

life.augmented

Selection Function
Key Byte 0

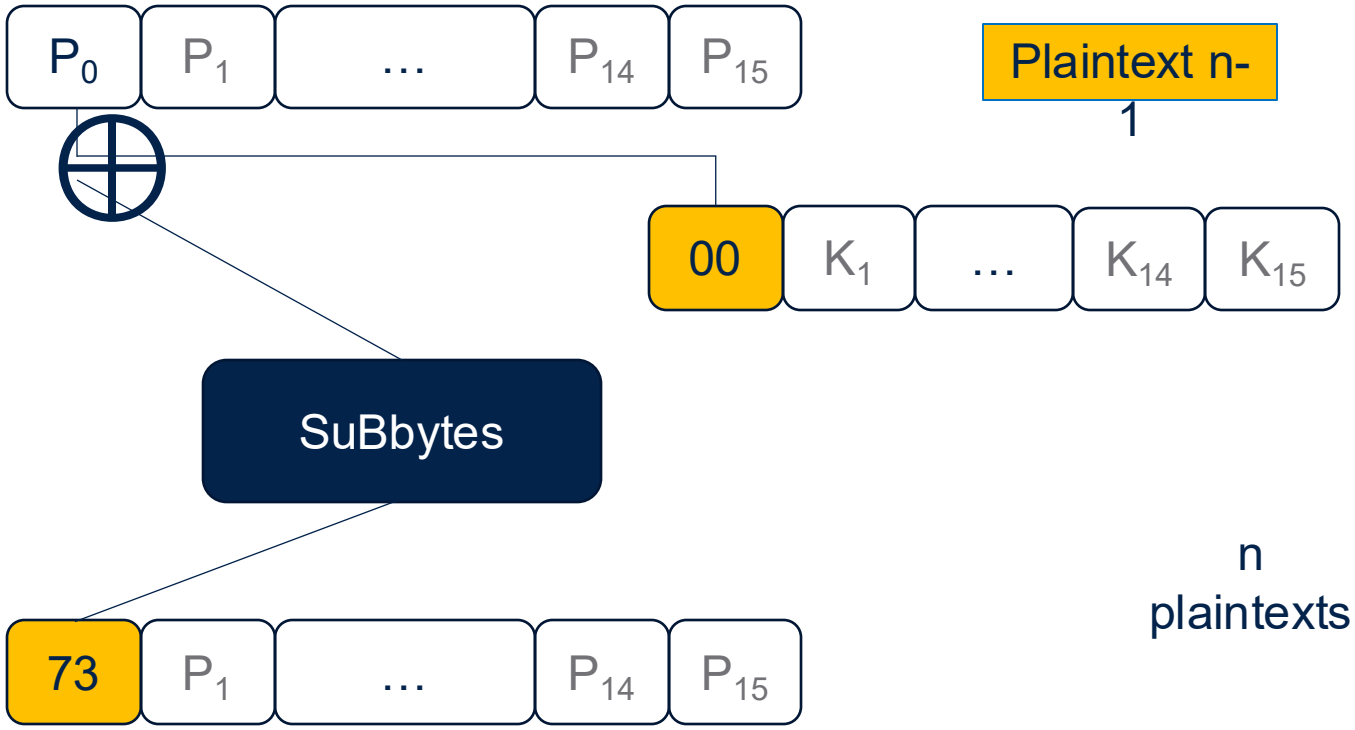


n
plaintexts



Guess 0 | Guess 1 | Guess 255

Selection Function
Key Byte 0



n plaintexts

256

0x1C

0x8A

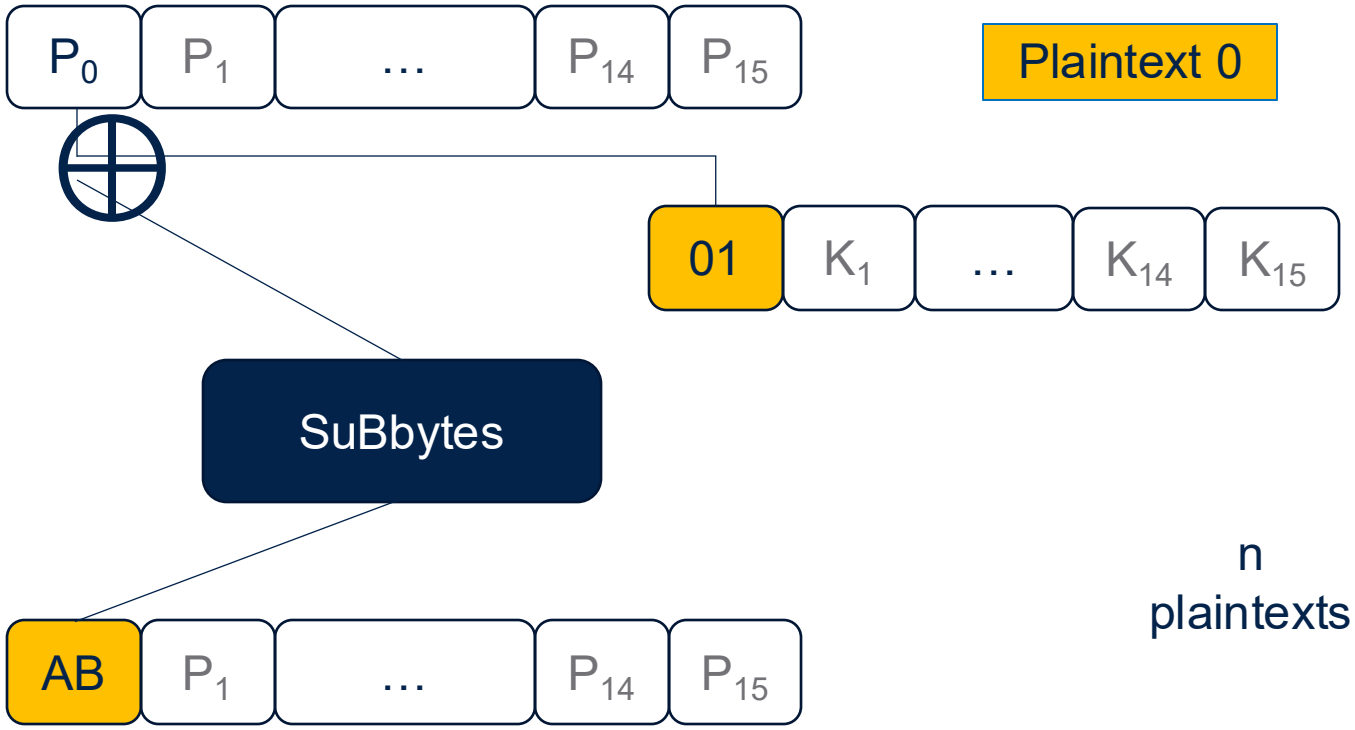
…

0x73

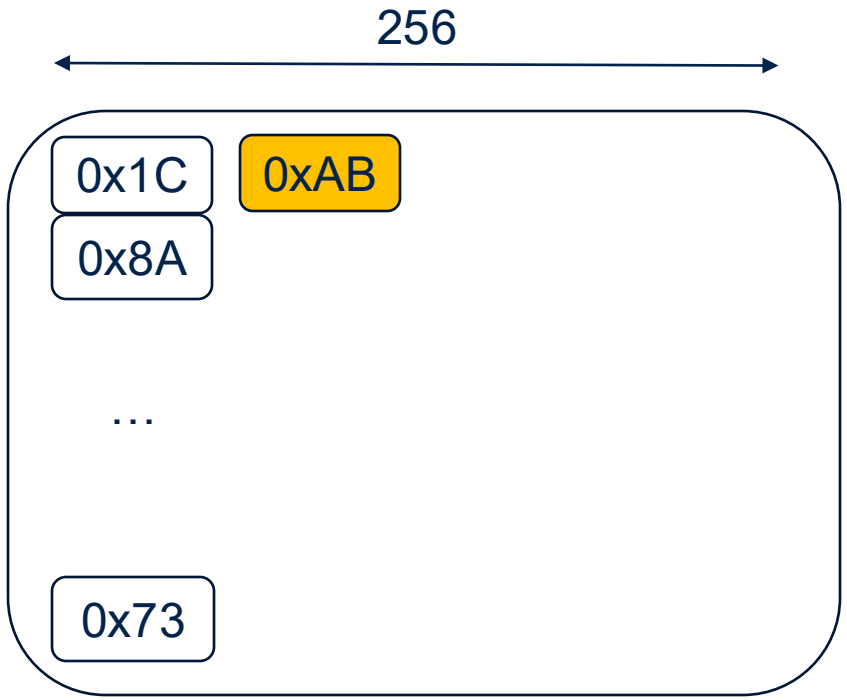
Guess 0 Guess 1 Guess 255

life.augmented

Selection Function
Key Byte 0



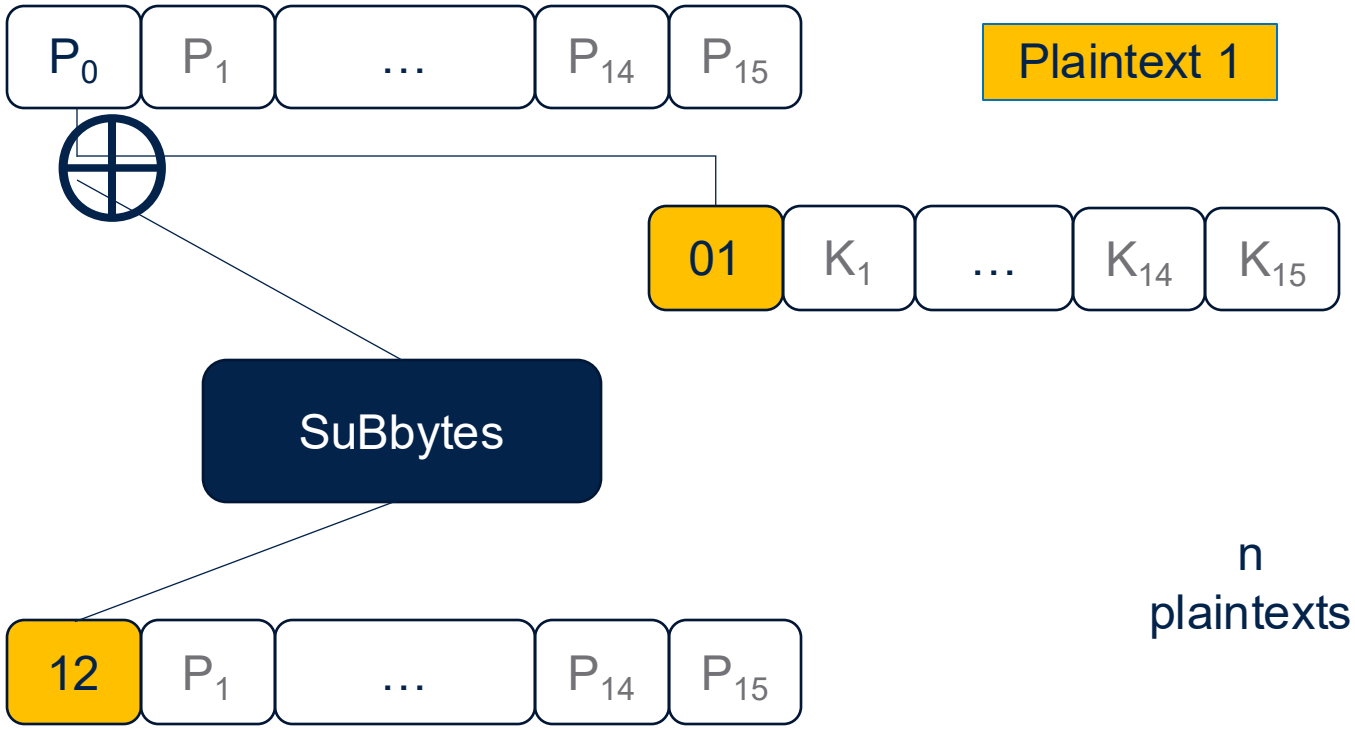
n plaintexts



Guess 0 | Guess 1 | … | Guess 255

life.augmented

Selection Function
Key Byte 0



n plaintexts

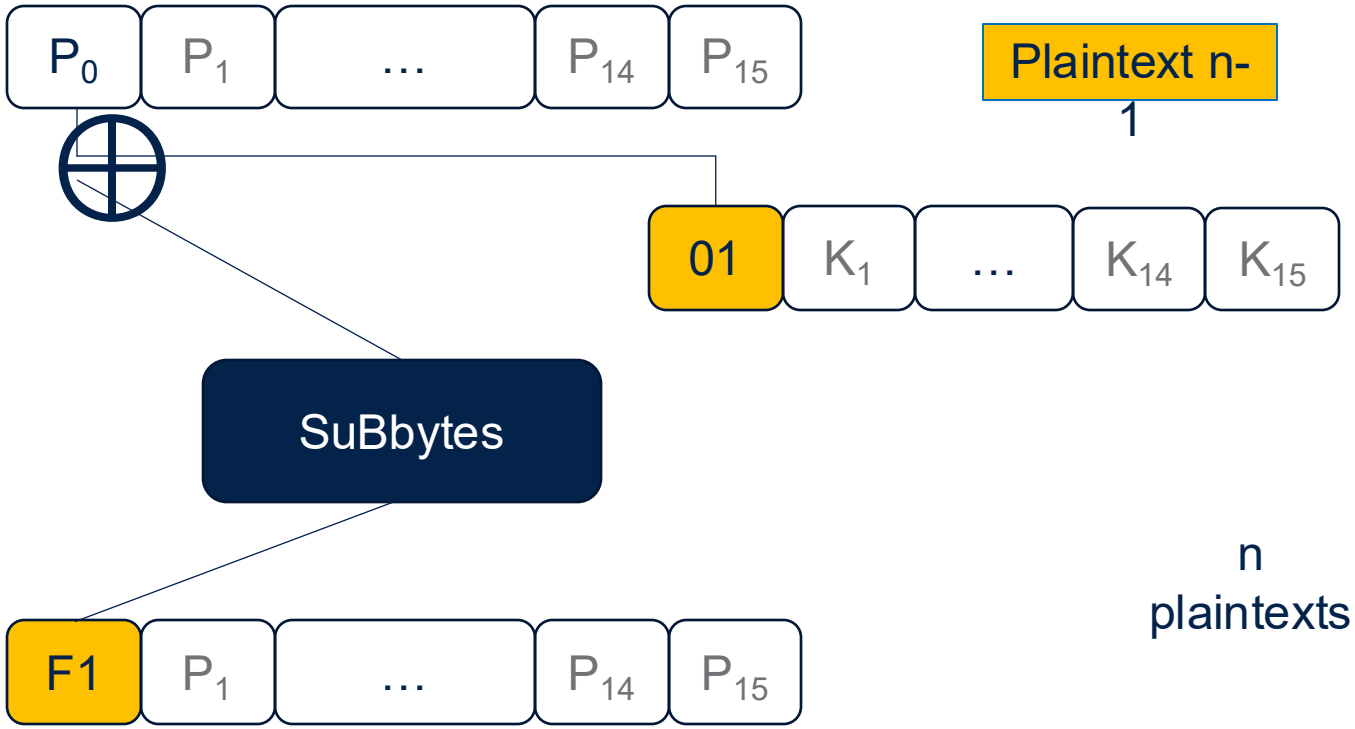
256

0x1C 0xAB
0x8A 0x12
…
0x73

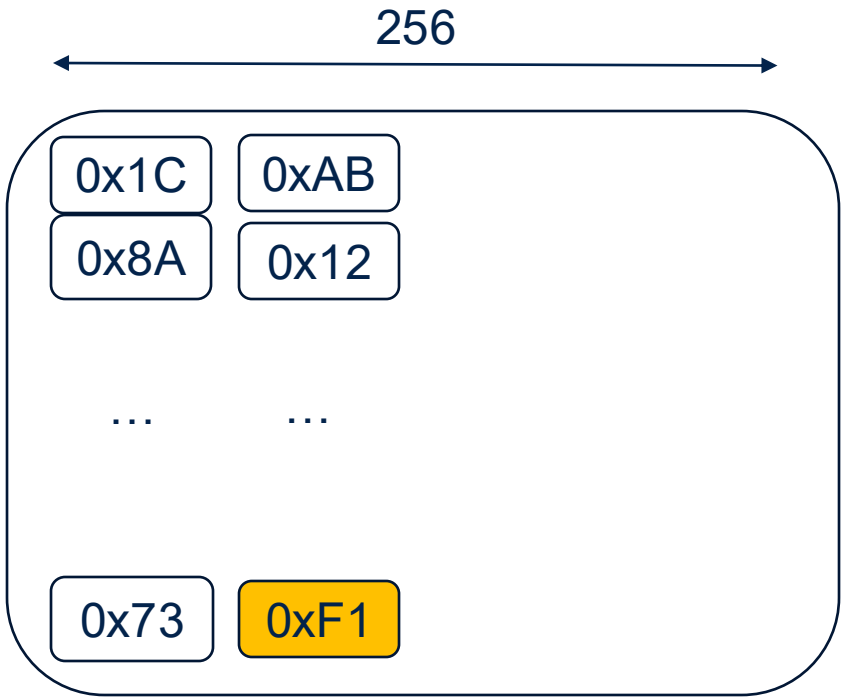
Guess 0 | Guess 1 | … | Guess 255

life.augmented

Selection Function
Key Byte 0



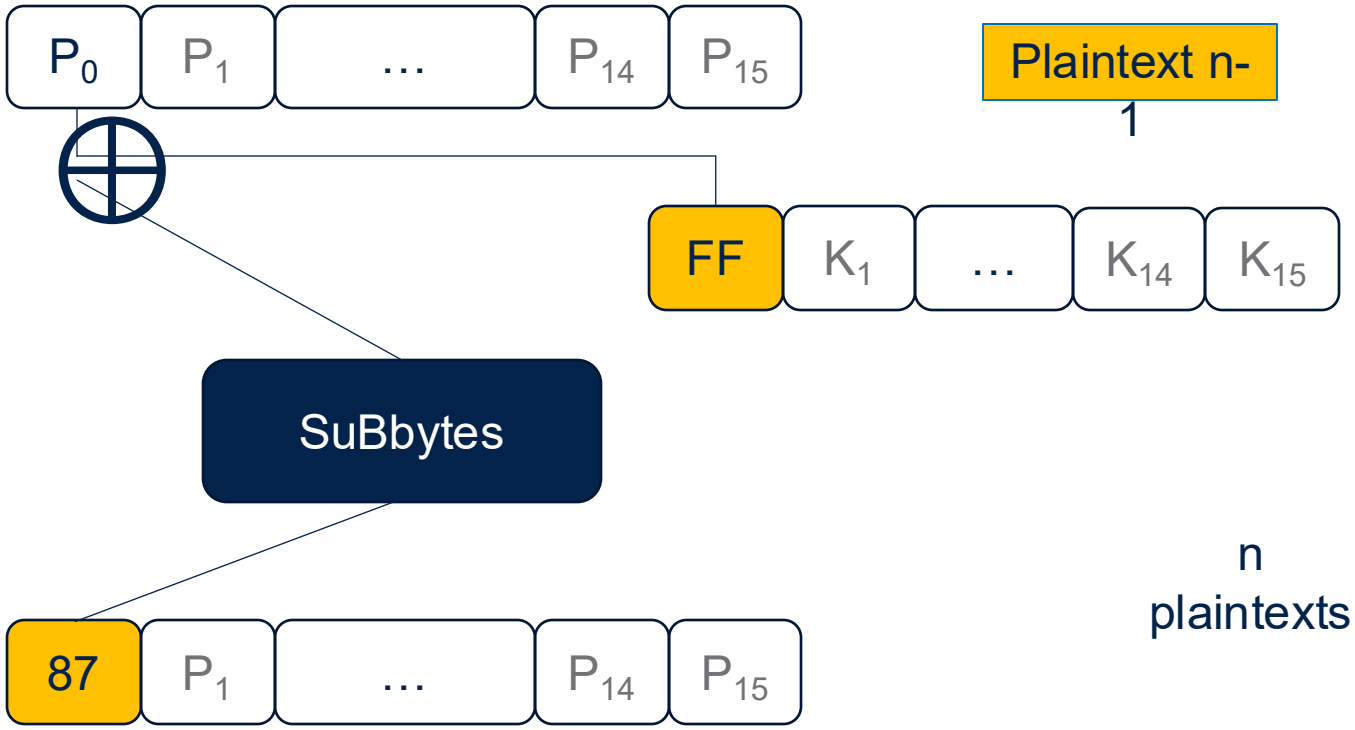
n plaintexts



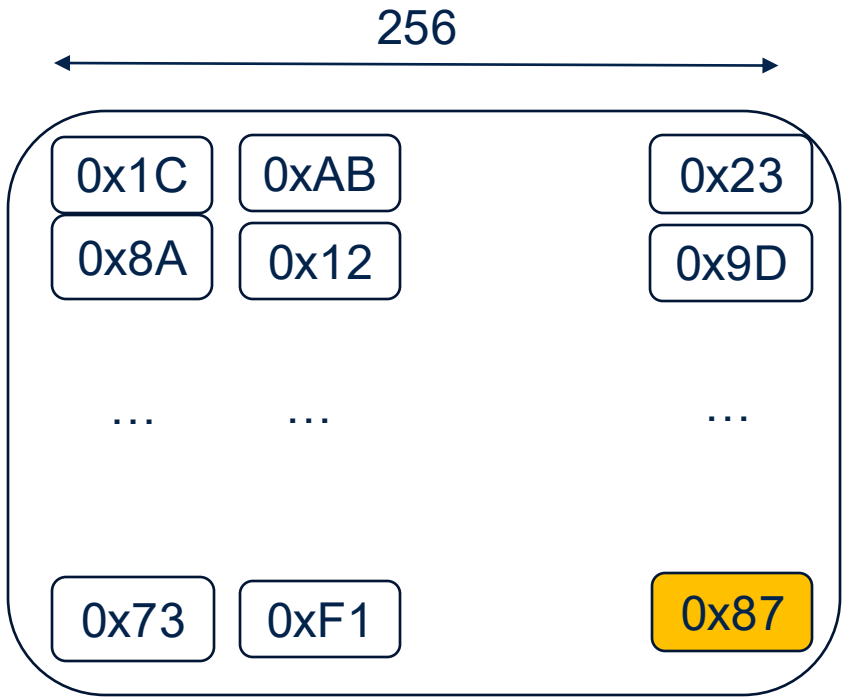
Guess 0 | Guess 1 | … | Guess 255

life.augmented

Selection Function
Key Byte 0



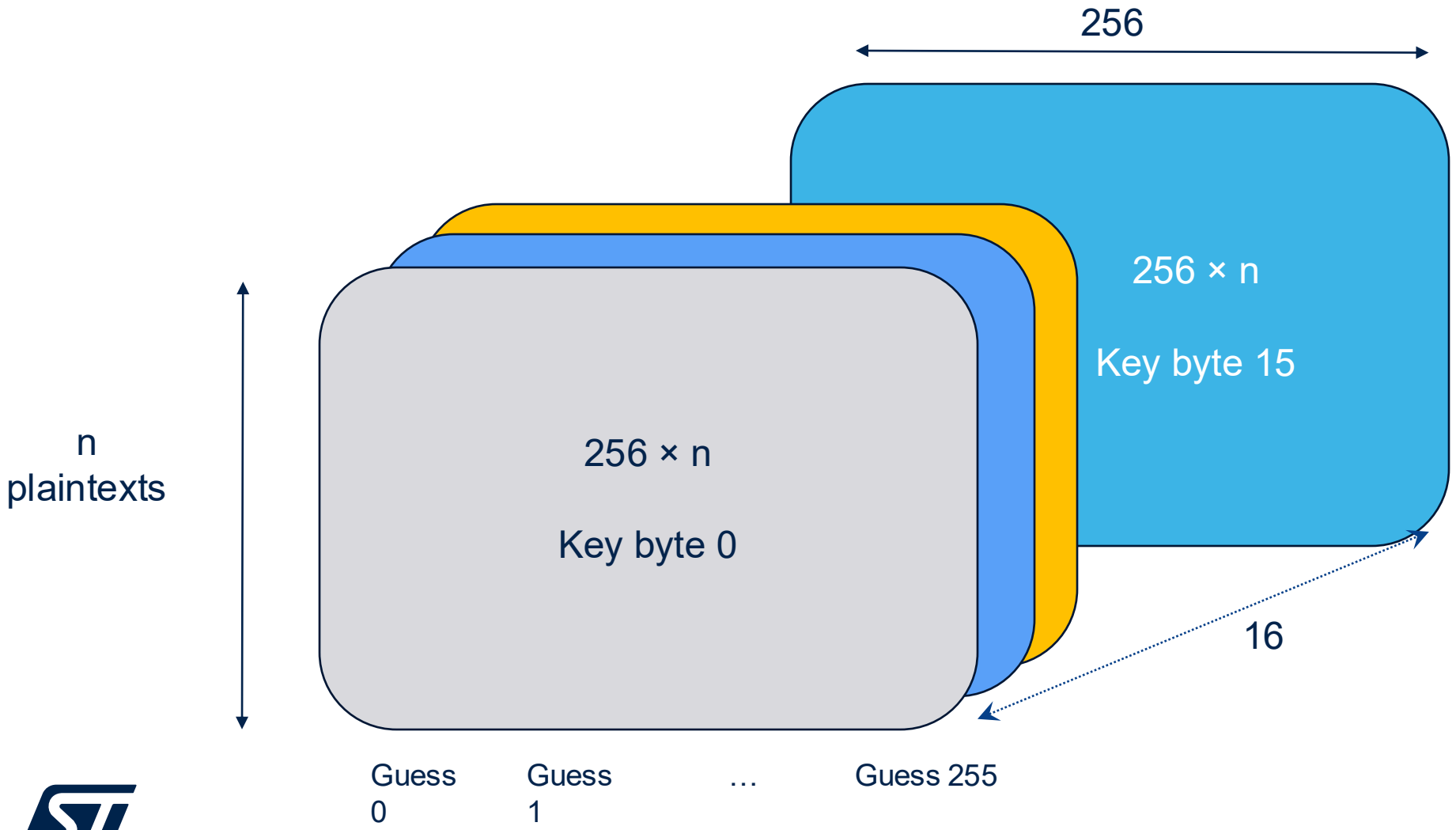
n plaintexts



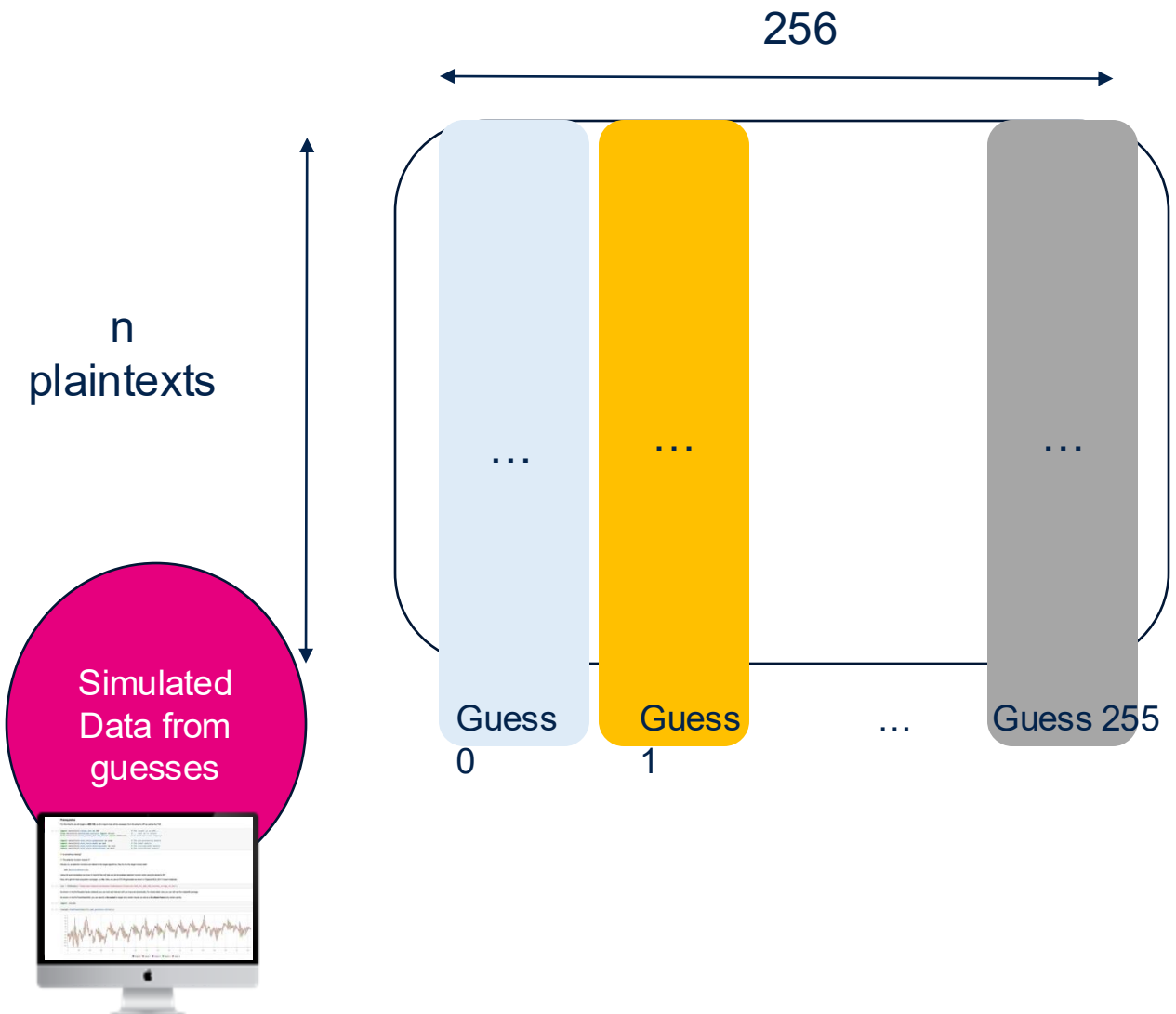
Guess 0 | Guess 1 | … | Guess 255

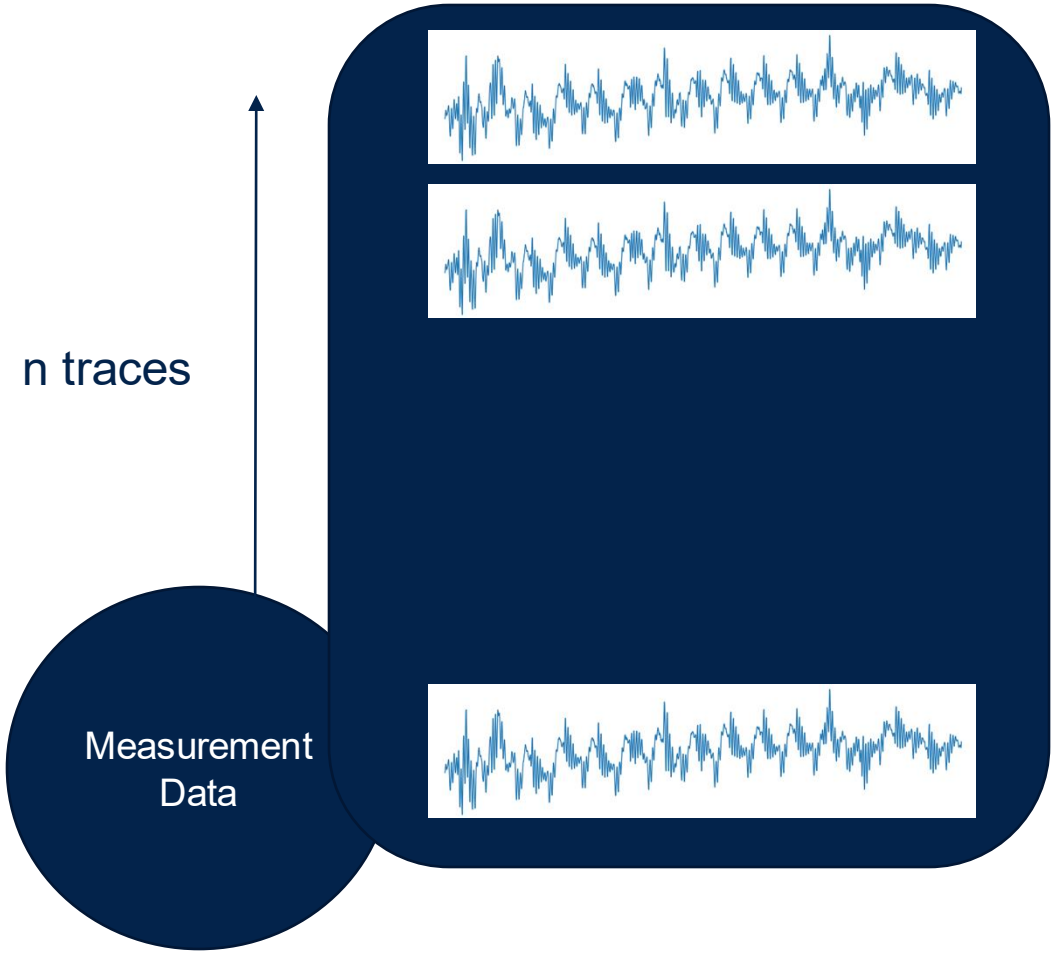
life.augmented

Selection Function



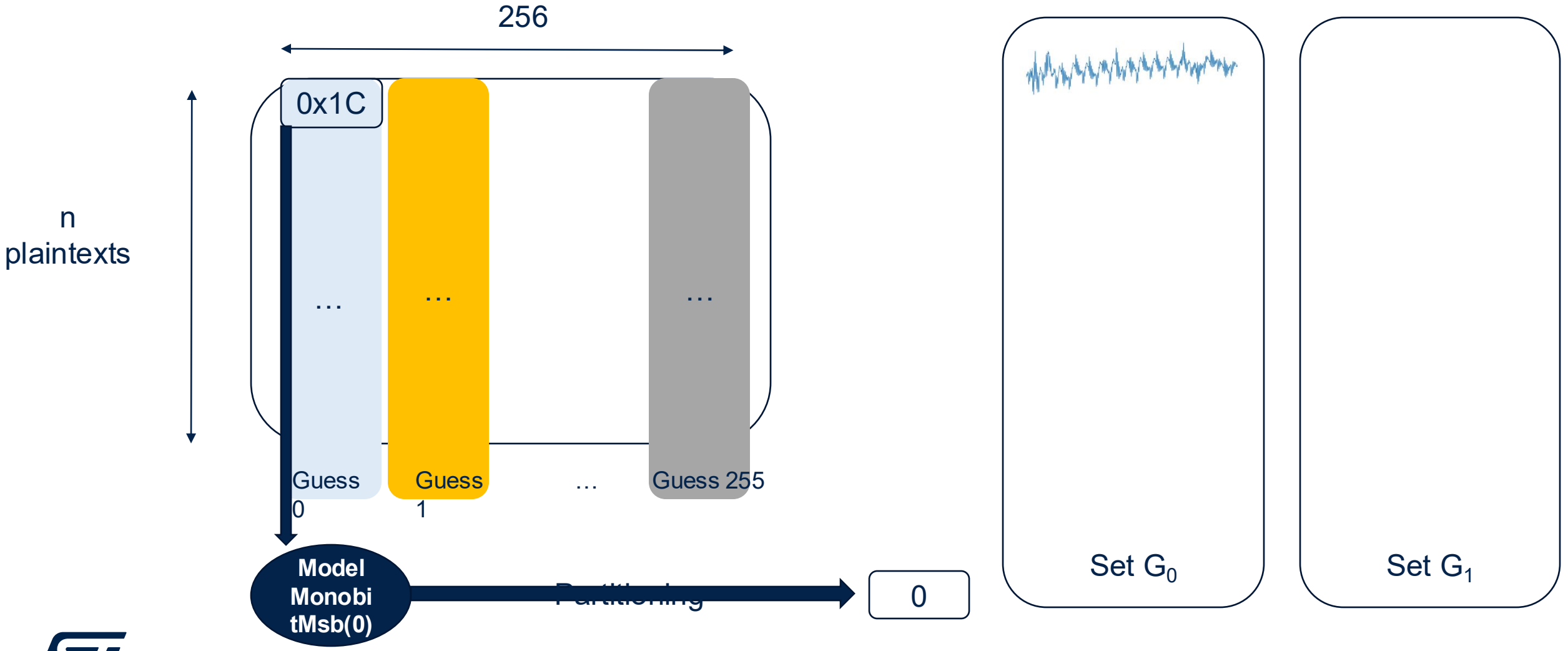
Now use a distinguisher …





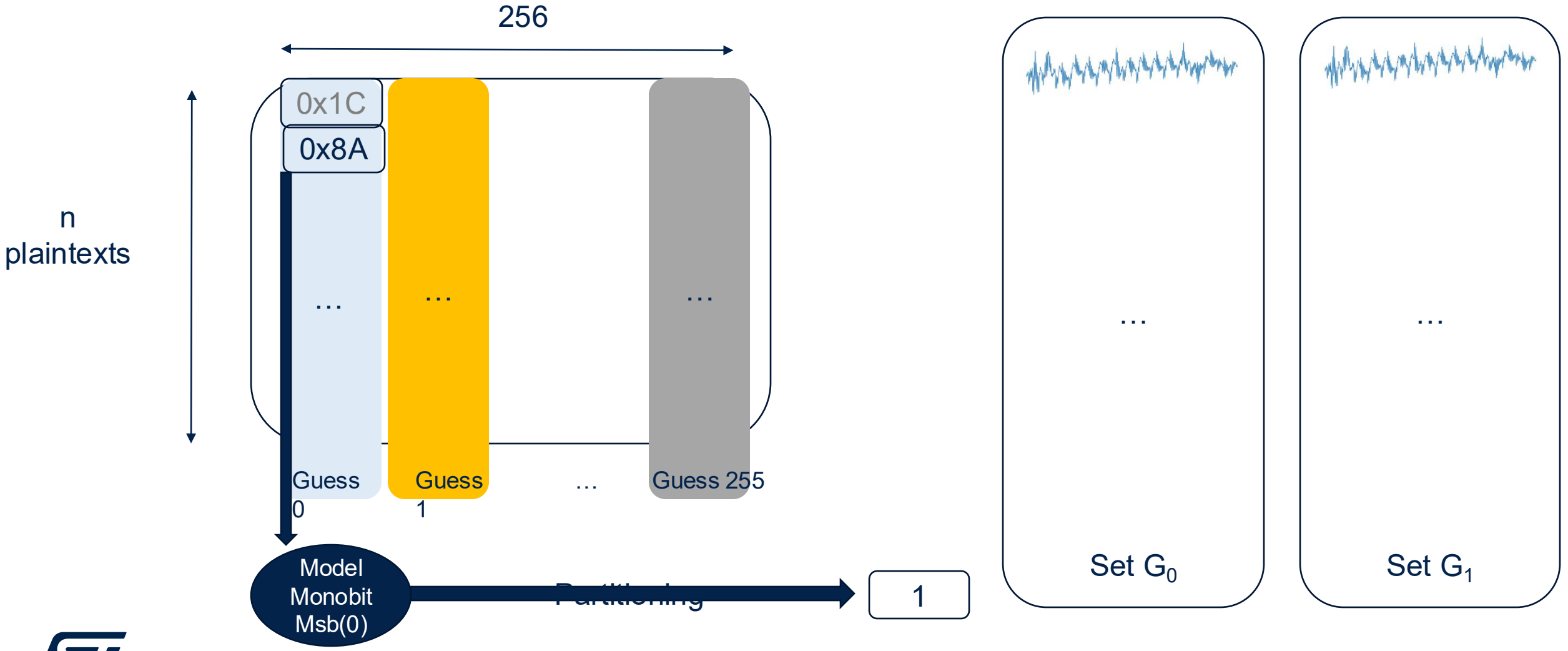
Differential Side-Channel Analysis

DPA for Differential Power Analysis



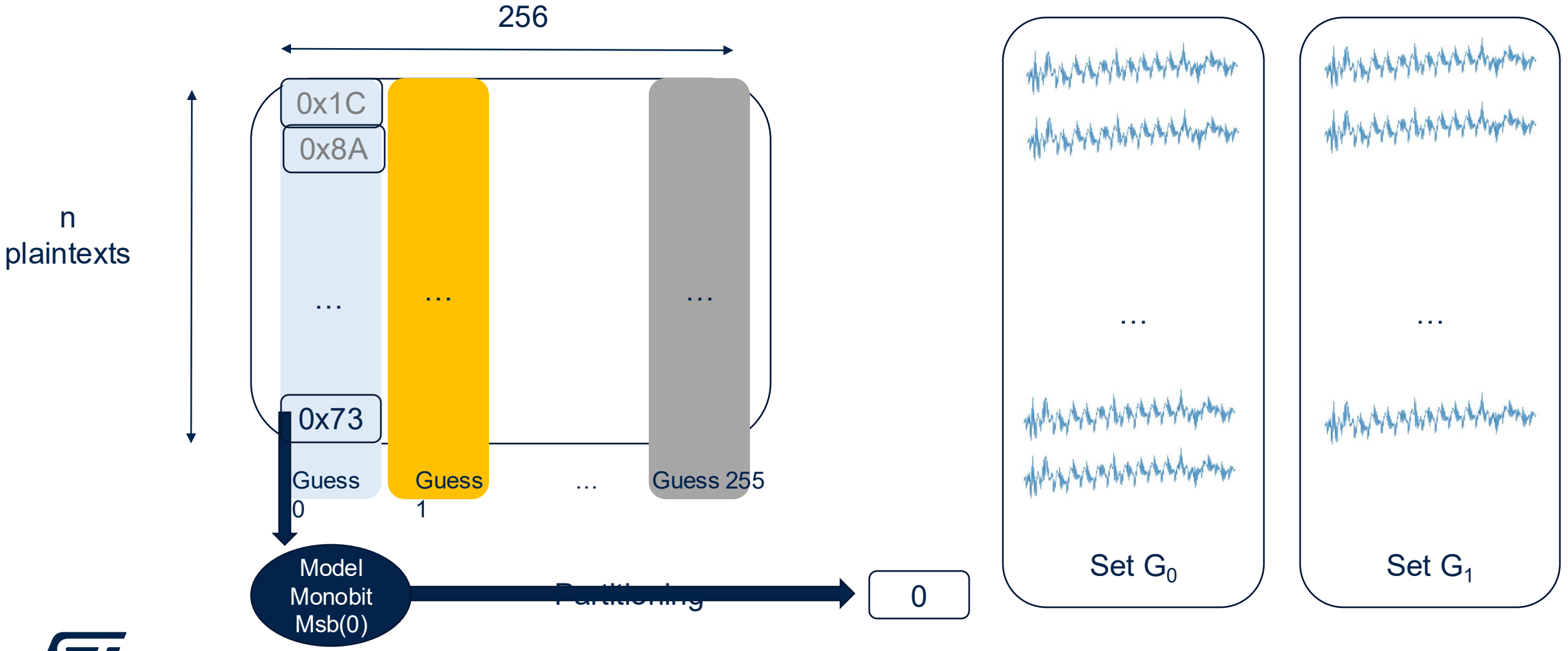
Differential Side-Channel Analysis

DPA for Differential Power Analysis



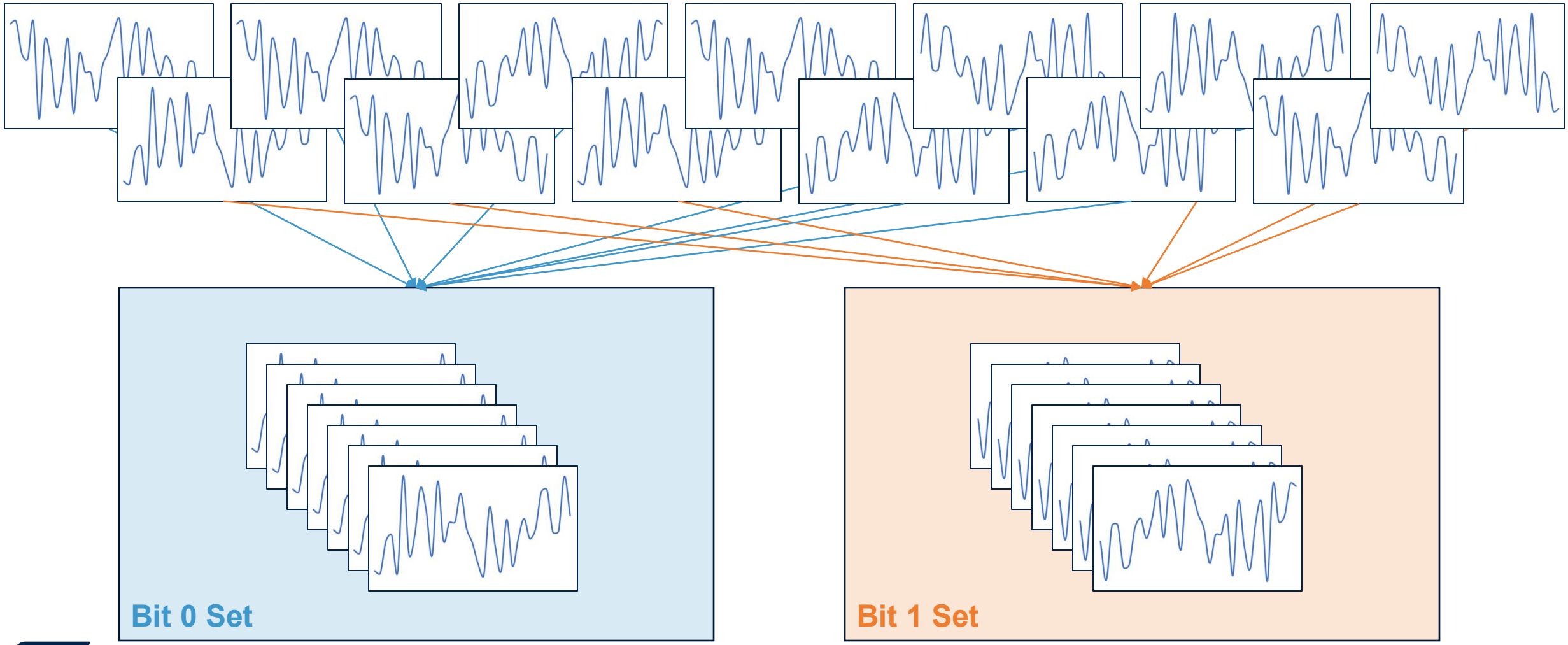
Differential Side-Channel Analysis

DPA for Differential Power Analysis



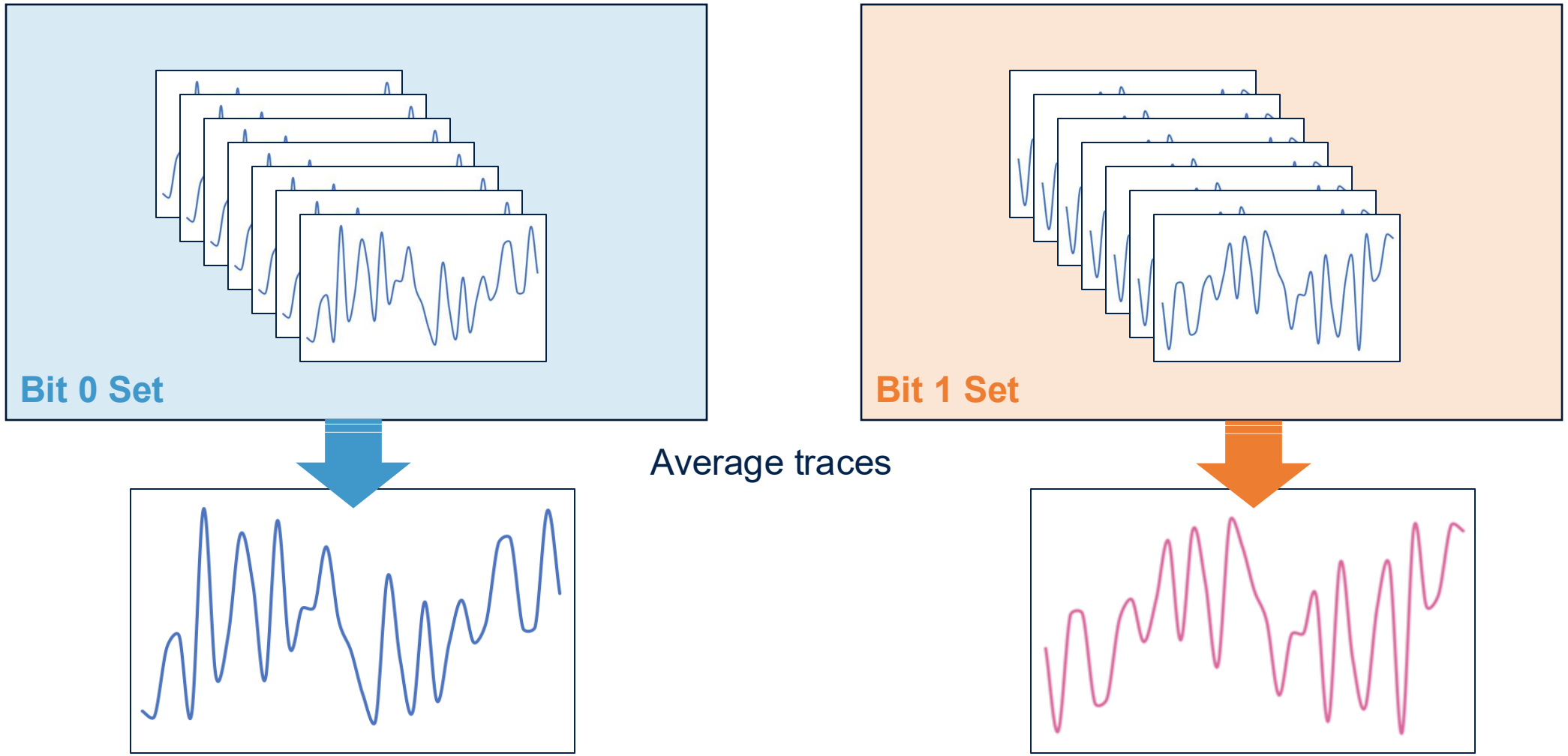
Differential Side-Channel Analysis

DPA for Differential Power Analysis



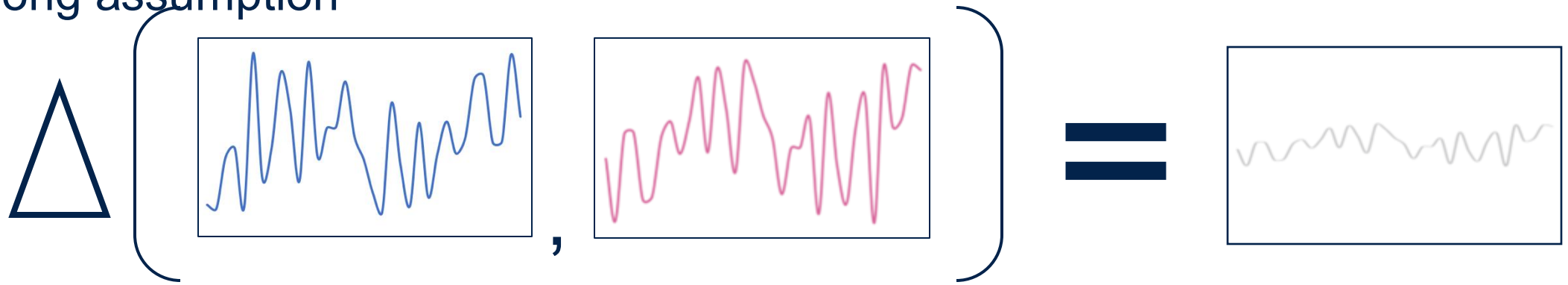
life.augmented

Side-Channel Analysis

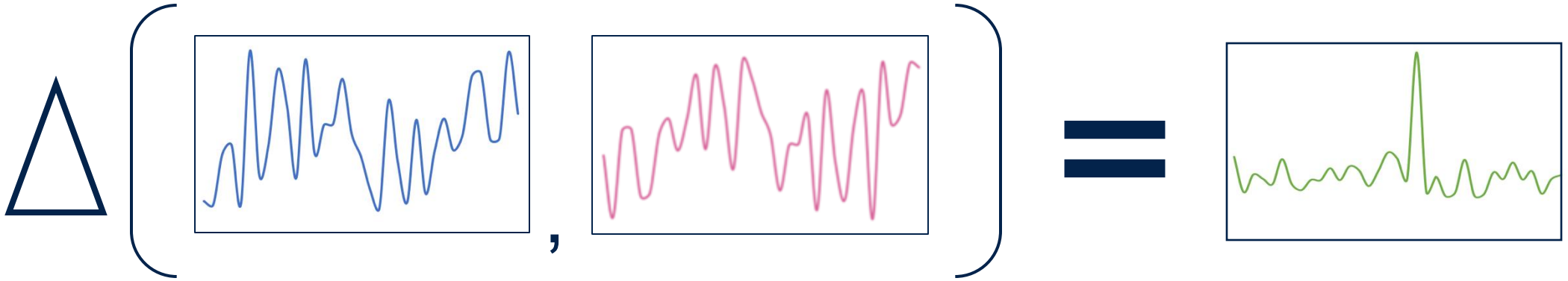


Side-Channel Analysis

- Wrong assumption



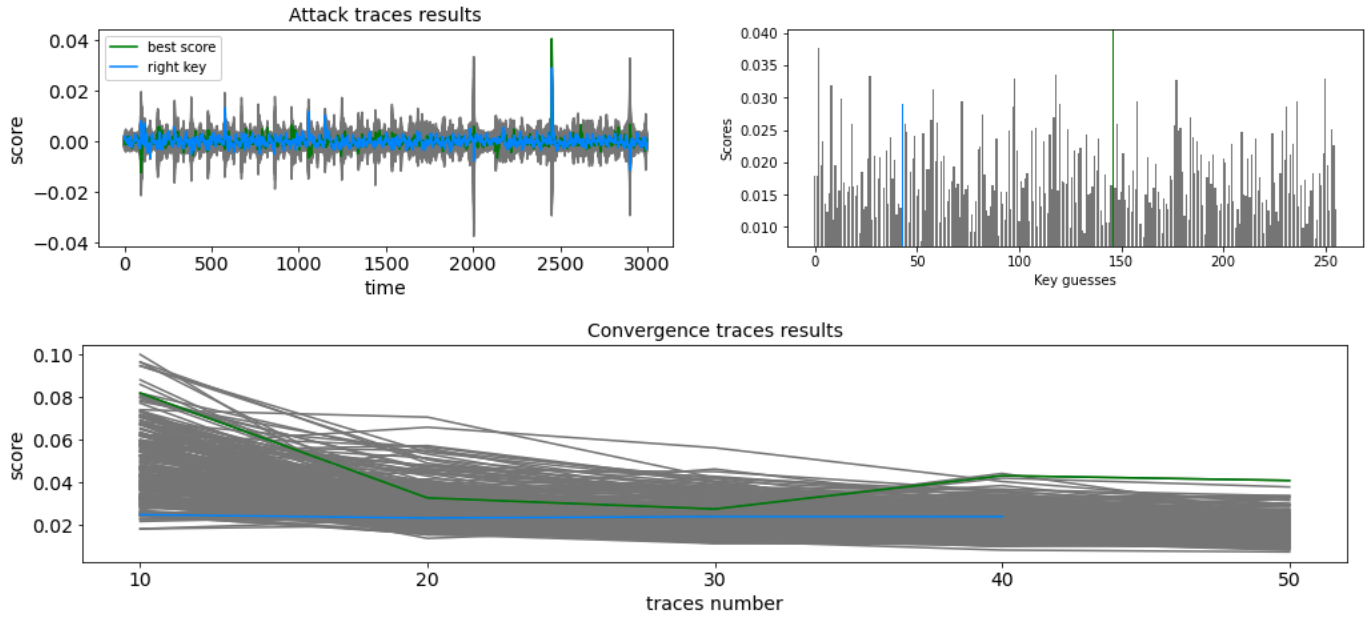
- Good assumption



Differential Side-Channel Analysis
Not always efficient technique

Correct Guess

Wrong Guess



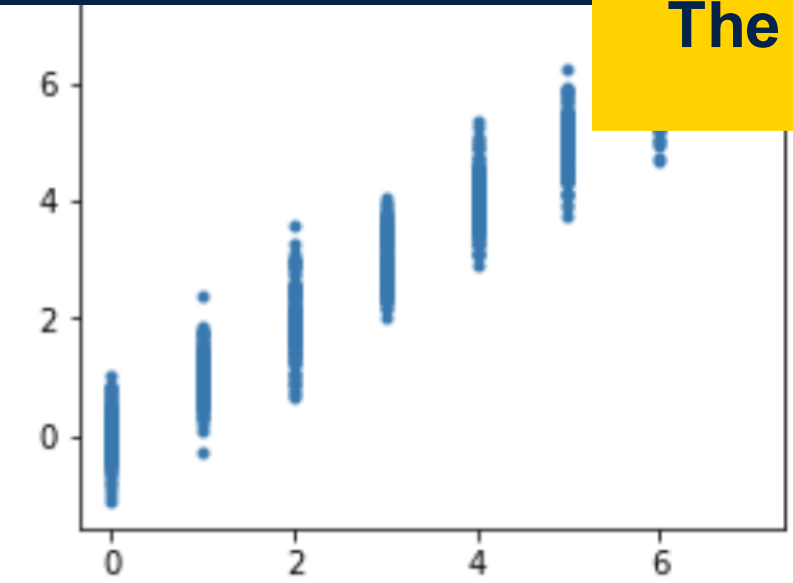
Here DPA is not efficient with this number of traces !!!

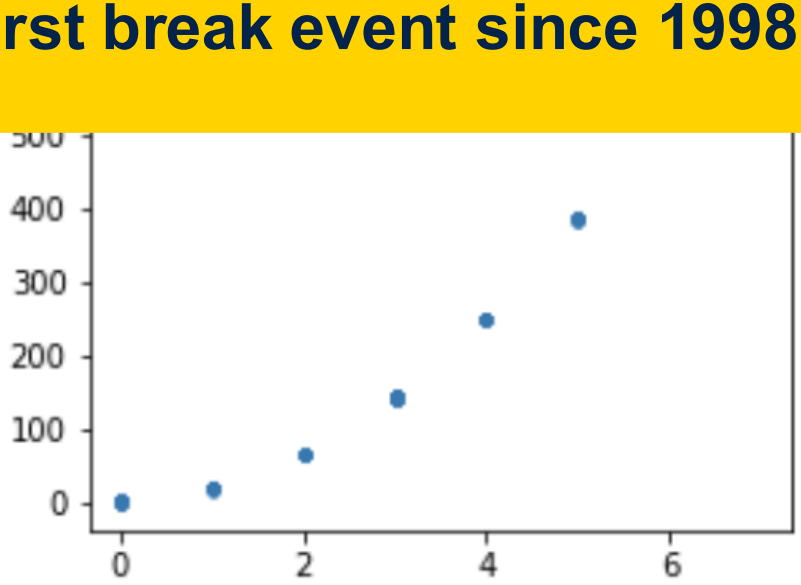
HOW CAN WE IMPROVE?

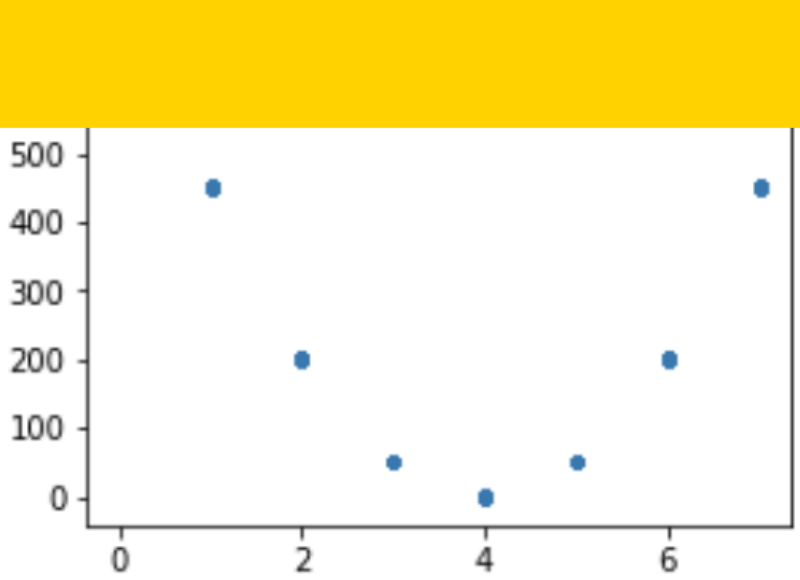
life.augmented

Correlation Side Channel Analysis

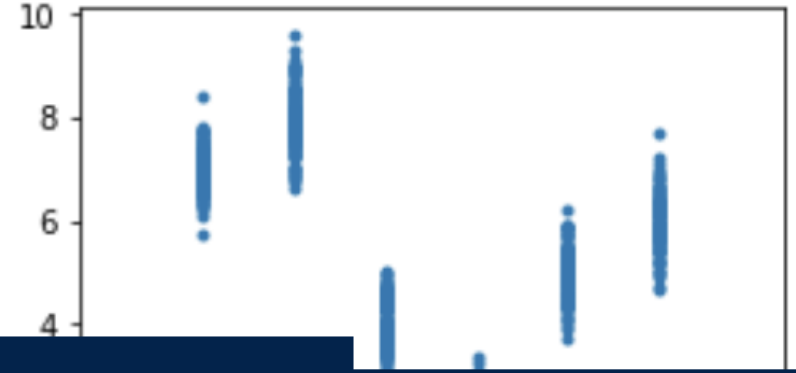
The first break event since 1998.



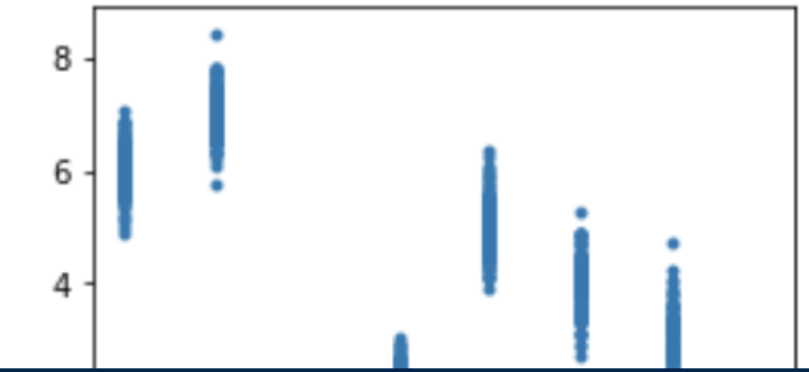




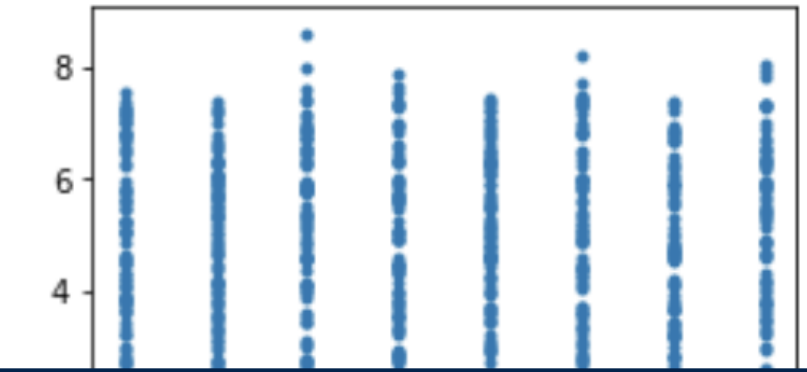
cor(A,E) = -0.09477240442682108



cor(A,F) = -0.47533771427731336



cor(A,G) = -0.0128367141566638 65



life.augmented

Correlation Power Analysis

- Bravais-Pearson Correlation Factor can be used to measure the statistical dependency between two set of data
  - Linear Regression
- Remember we have the following two sets: we want to evaluate if they are
  - Dependant

  or
  - Independent

life.augmented

Correlation Power Analysis

- Two variables (data sets) are linearly related

→ the correlation coefficient will be high

- If the two data sets are not linearly related

→ the correlation coefficient will be low.

- Most cases have shown Linear leakage model in Hamming weight is efficient
- Model:

$$W = a \times HW(data) + b + noise$$

life.augmented

Correlation Power Analysis

Principle

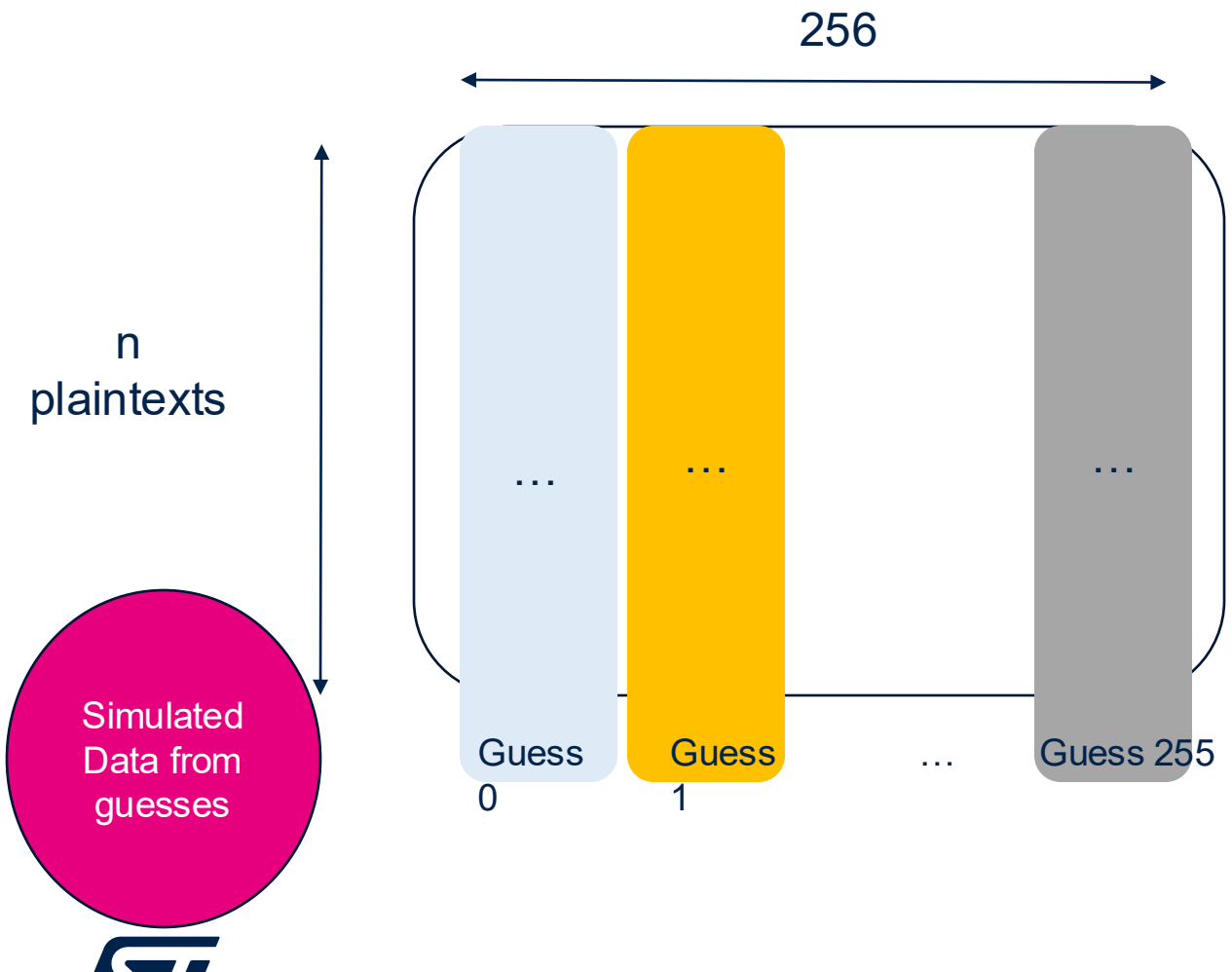
IC power consumption is considered linear in $HW(D \oplus R)$: the hamming distance between a data D and a reference state R,
Model $W = \mu \cdot H(D \oplus R) + \nu$.
The linear correlation factor between W and a curve C is:

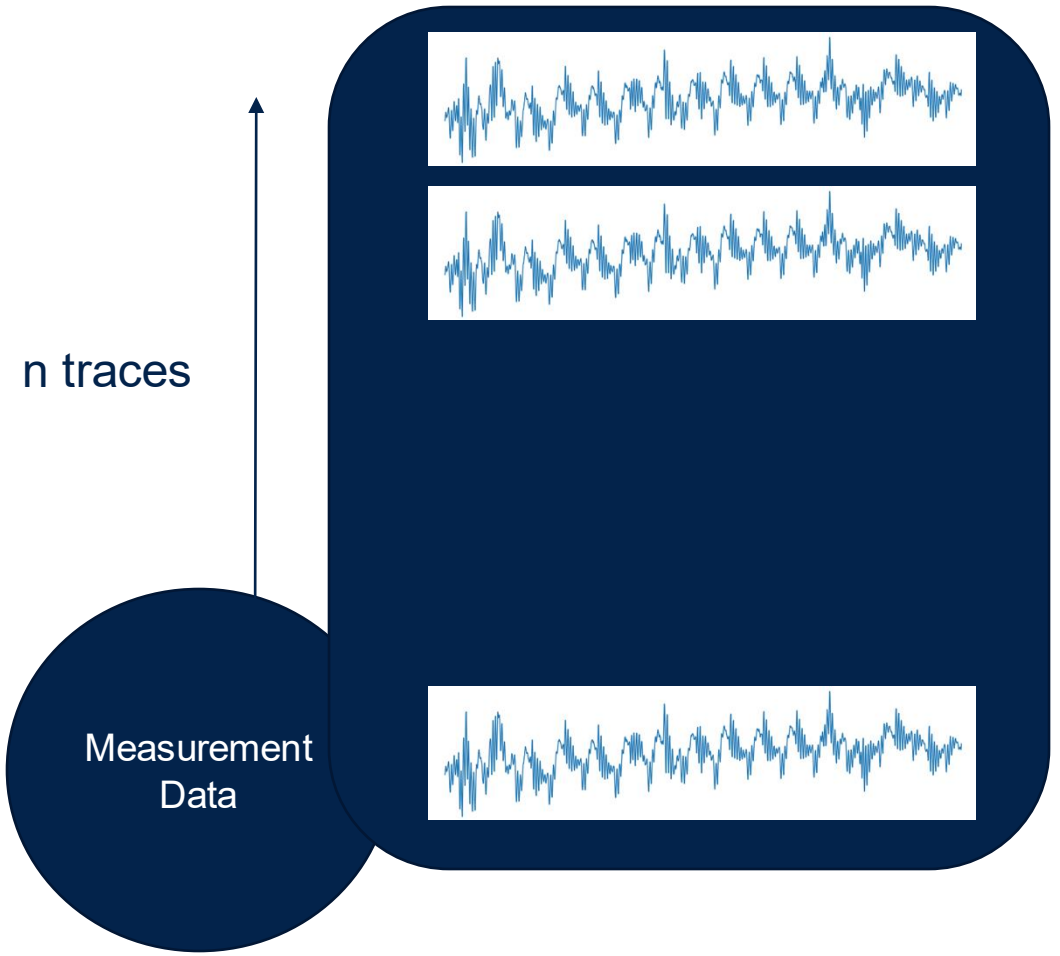
$$\rho_{C,H} = \frac{cov(C,W)}{\sigma_C \cdot \sigma_W} = \frac{cov(C,H)}{\sigma_C \cdot \sigma_H}$$

$$\hat{\rho}_{C,H} = \frac{cov(C,H)}{\sigma_C \sigma_H} = \frac{t\sum(C_i H_{i,R}) - \sum C_i \sum H_{i,R}}{\sqrt{t\sum C_i^2 - (\sum C_i)^2}\sqrt{t\sum H_{i,R}^2 - (\sum H_{i,R})^2}}, i = 1 \ldots t$$

life.augmented

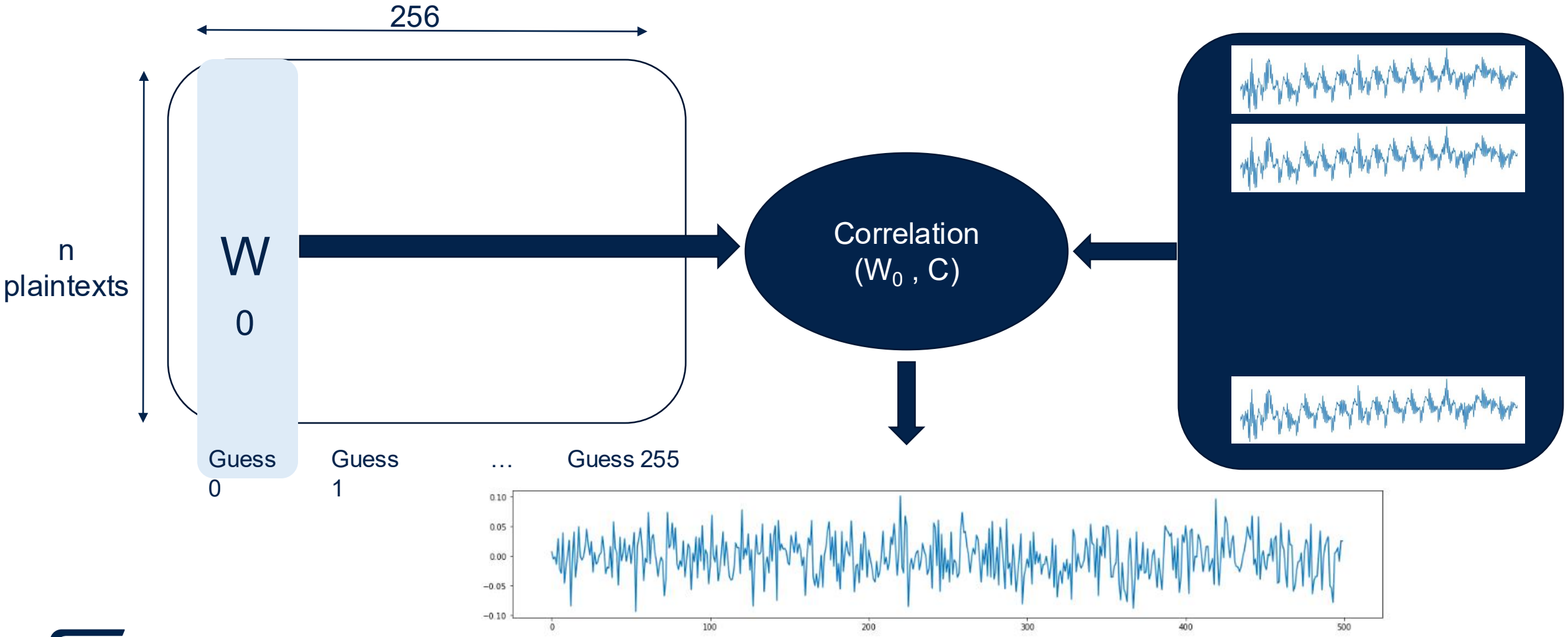
Now use Correlation Analysis



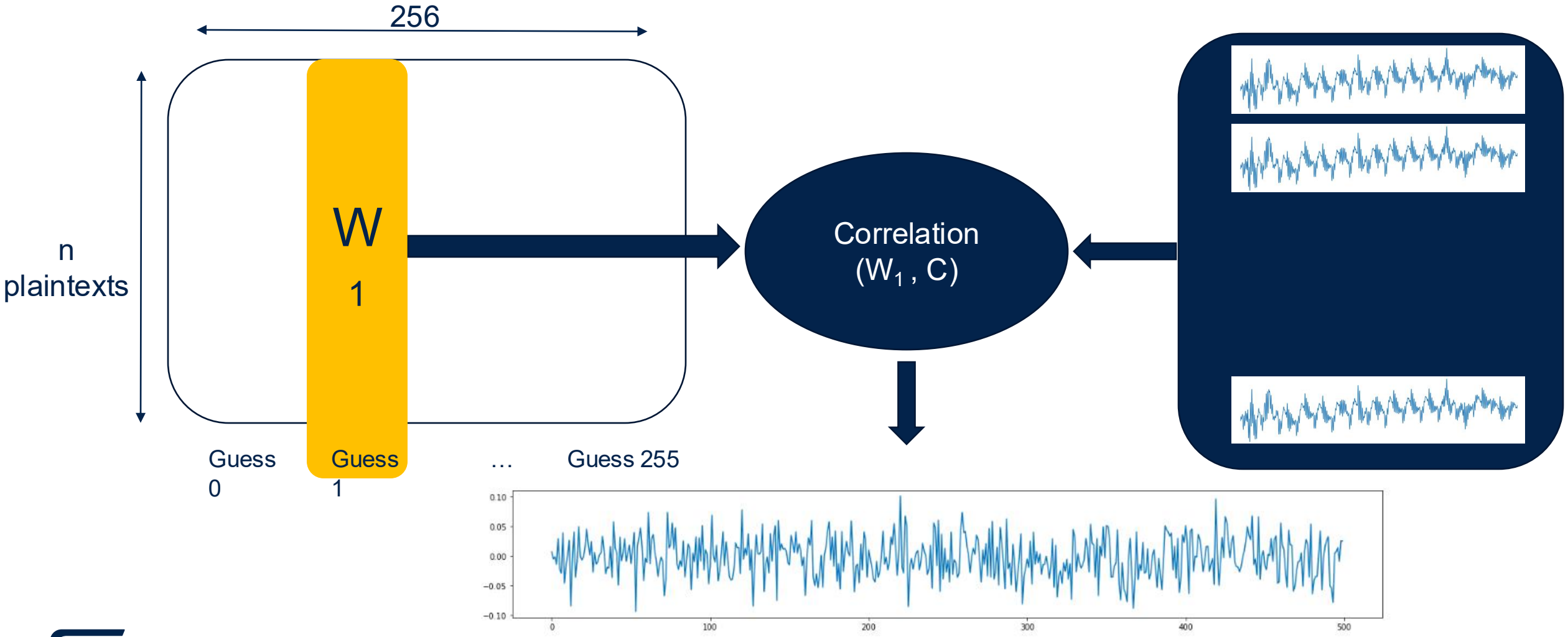


life.augmented

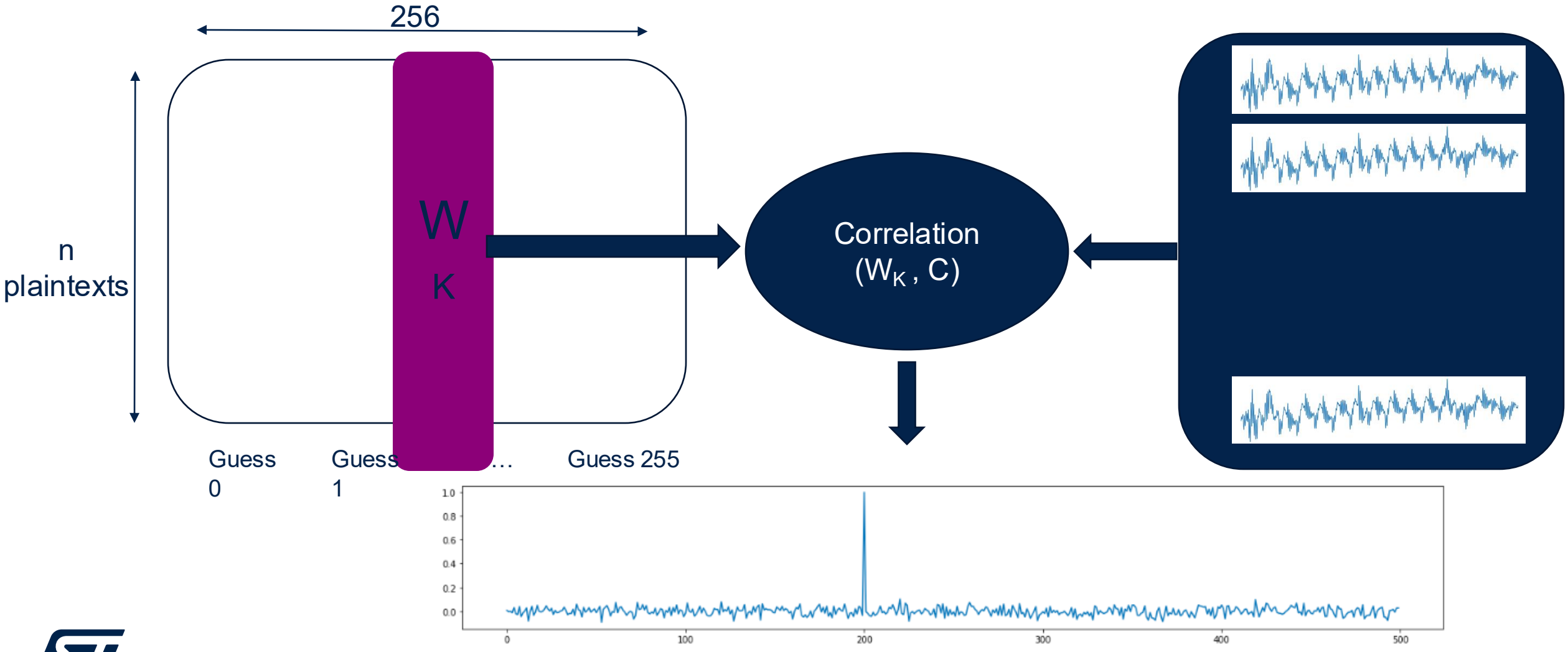
Now use Correlation Analysis



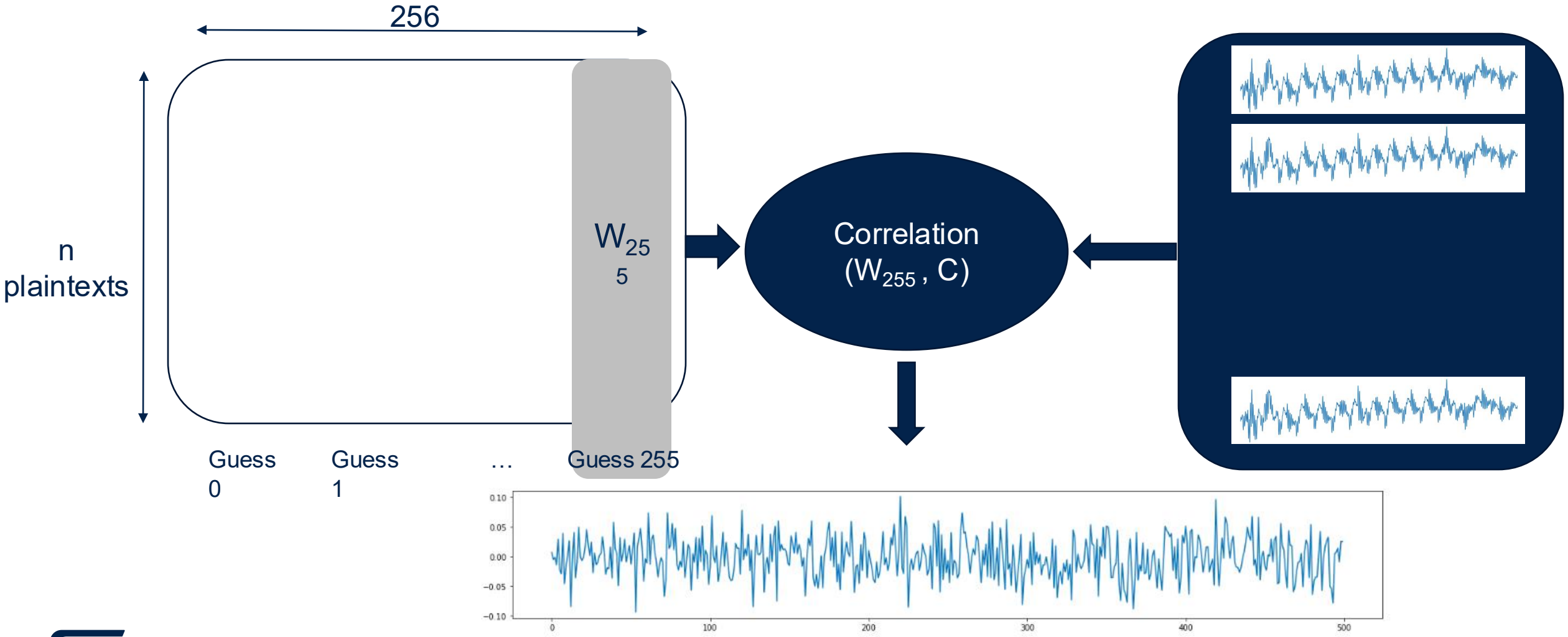
Now use Correlation Analysis



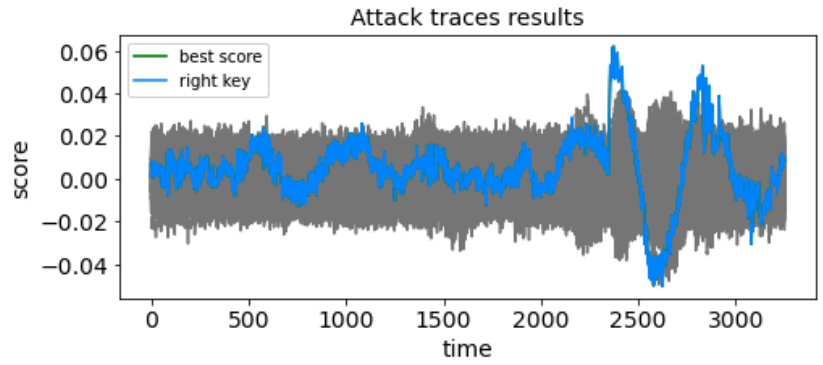
Now use Correlation Analysis

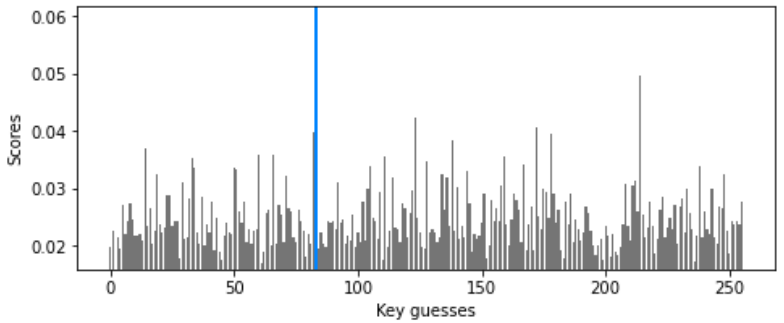


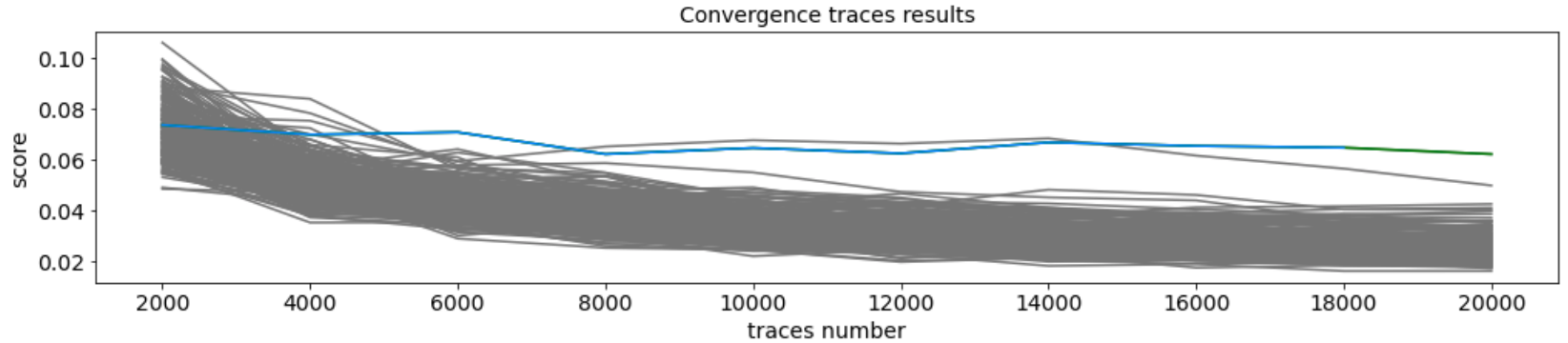
Now use Correlation Analysis



Correlation Attack Result








```
Best guess for byte 0 = 0x53
Correct guess for byte : 0=0x53
```

Correlation Power Analysis

- We can use information from all bits contrarily to DPA – no information loss
  - All operand and values can be used with
    - Mono-bit attacks
    - Hamming weight attack
    - Value itself to be correlated

- This model is linear in the HW or HD
  - But it is close to the real behaviour of a circuit
  - Indeed, the Hamming measures the number of bit flips entre between a state i and the next state i+1
    - Example: writing in a register R0
      - R0 = a
      - R0 ← R0 ⊕ K
      - R0 = b Bit flip a ⊕ b → K

- Requires about 10 times less traces than DPA; no “ghost” peaks

CPA demonstrations on DES and AES

- DPA vs CPA on XMEGA AES Traces
- DSCA on DES DPA contest V1
- DSCA on AES DPA contest V2

Categories of Countermeasures

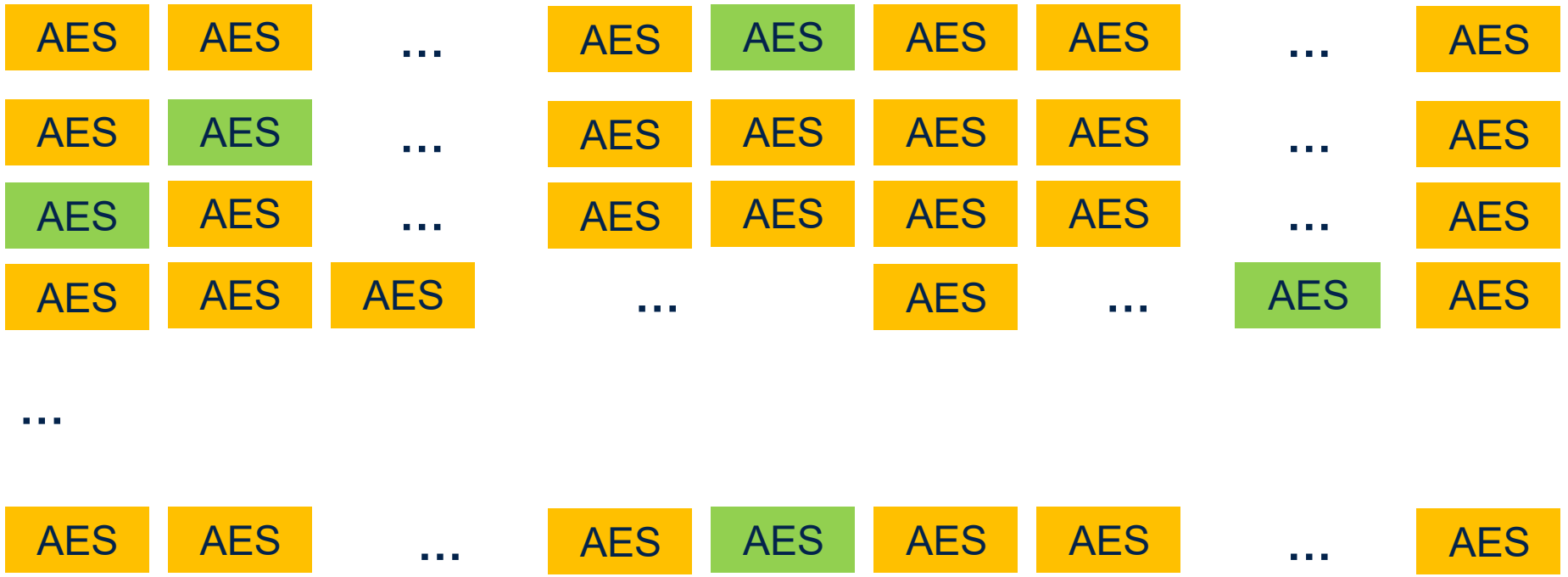
- **Protocol**
  - Paddings
  - Session keys,
  - Fresh re-keying
  - Unknown input and/or output (partially): CTR mode, payment ATC
- **Desynchronize**
  - Hardware de-synchronization: jitter, clock and frequency jitters, random delays
  - Shuffling
  - One amongst N techniques
- **De-correlate**
  - Hardware techniques to balance consumption (dual rail)
  - Masking techniques

One among N countermeasure(s)

One among N countermeasure(s)

- Same input message with fake keys
- Different (random fake input) messages for correct key: *key is manipulated more (!!)*

Attacking: one among N countermeasure(s)?



- Same input message with random fake keys
- **What attack do you recommend?**
  - Ghost peaks?

life.augmented

Attacking one among N countermeasure(s)?

- Same input message with random fake keys
- What attack do you recommend? Ghost peaks?

Cipher texts attacks are more efficient

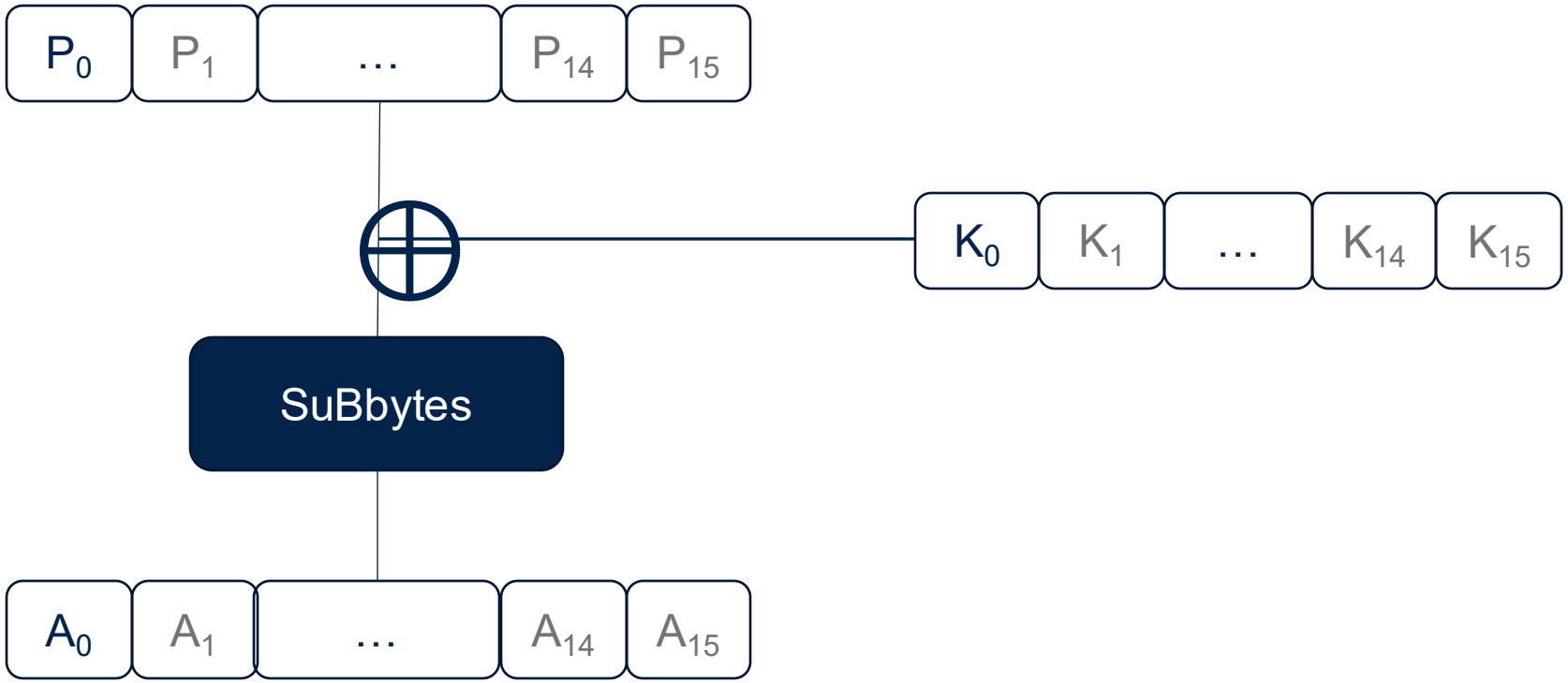
life.augmented

Masking Techniques

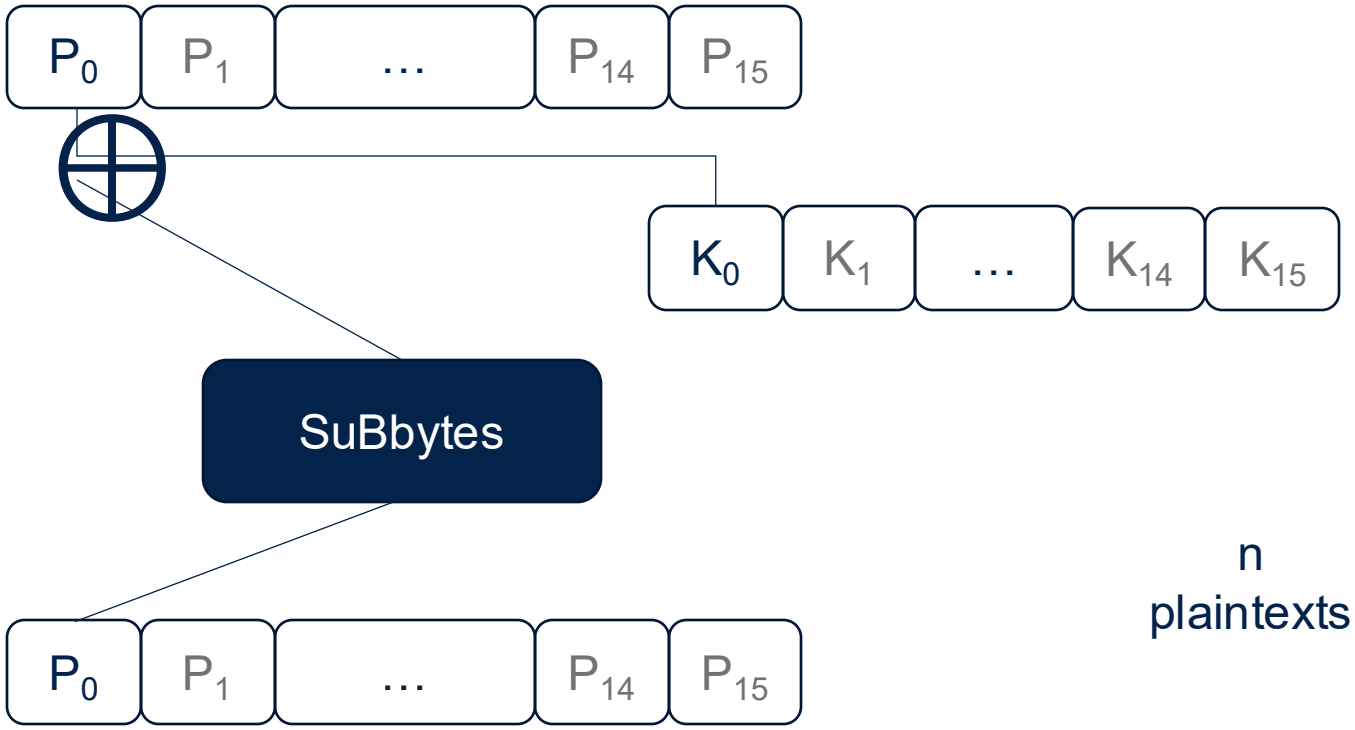
- Data Masking with random values
  - The idea
    - Ai ⊕ K = Bi can be attacked with DPA..
    - Use a random X and replace by
      - Ai ⊕ X = Ci
      - Ci ⊕ K = Di
      - Di ⊕ … → Ei
      - Ei ⊕ X = Bi
  - The bit-flips and IC power consumption changes at each execution
- Several countermeasures based on this principle published
- Can be Boolean (TDES, AES) and/or arithmetic masking (HMAC)

life.augmented

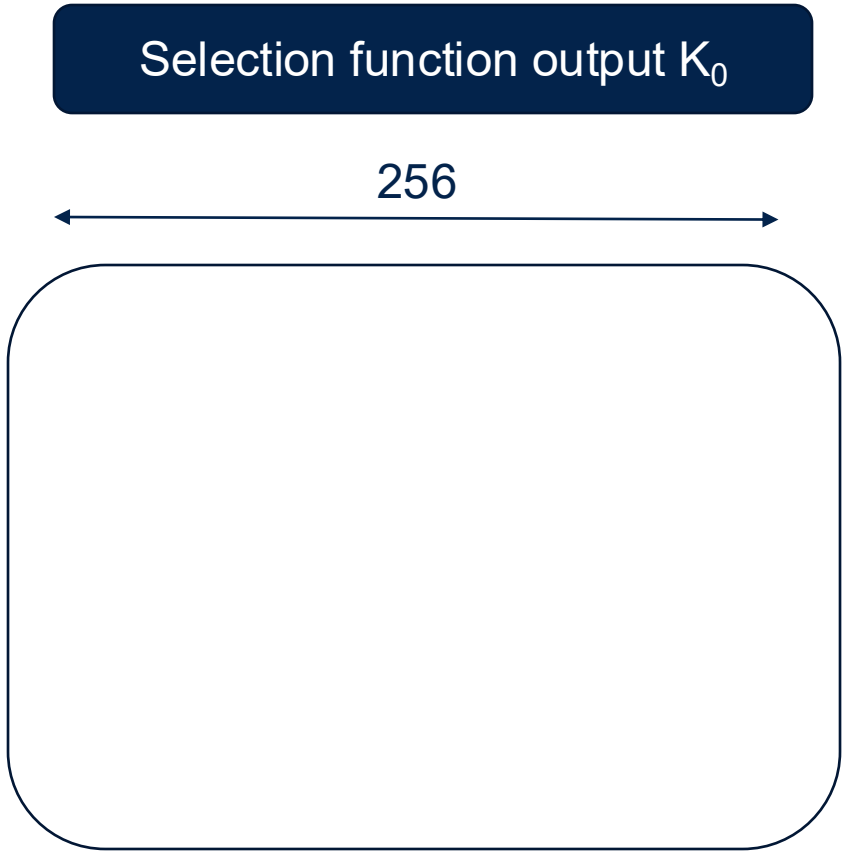
AES first operations



Selection Function
Key Byte 0



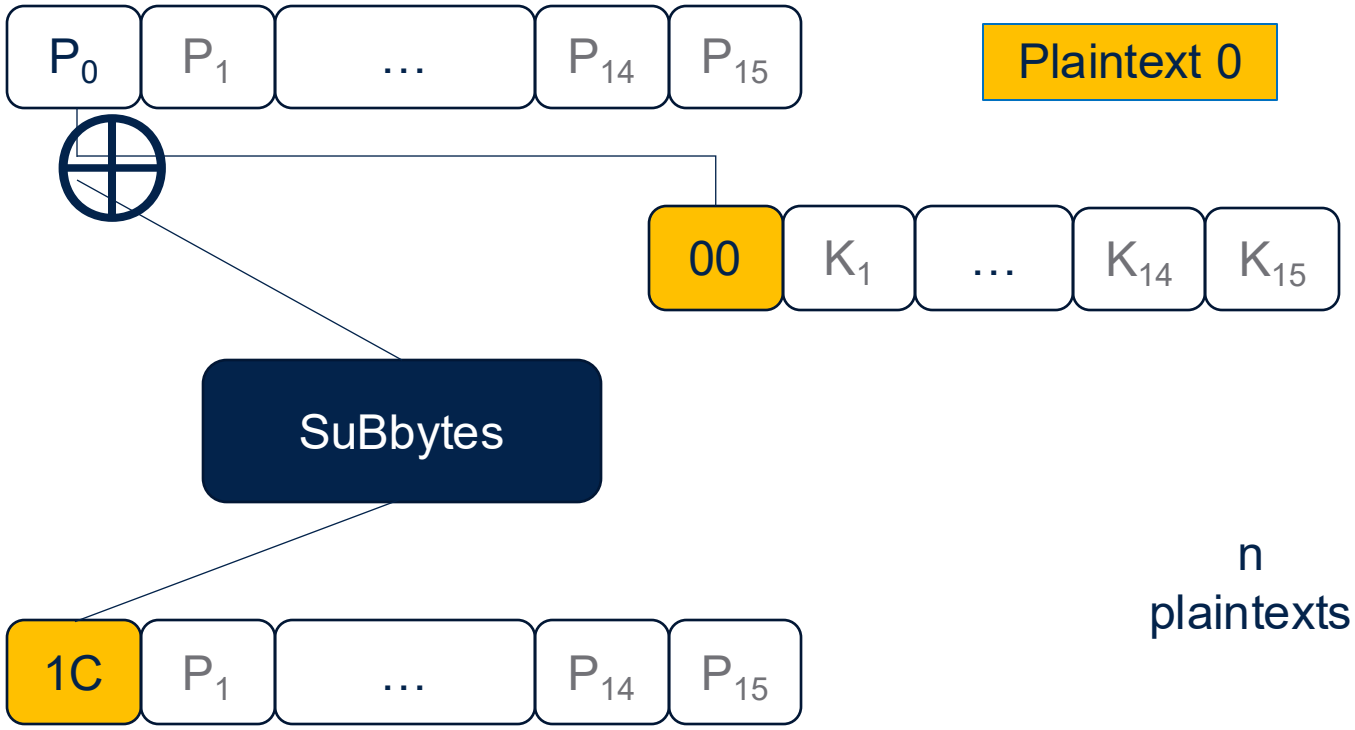
n
plaintexts



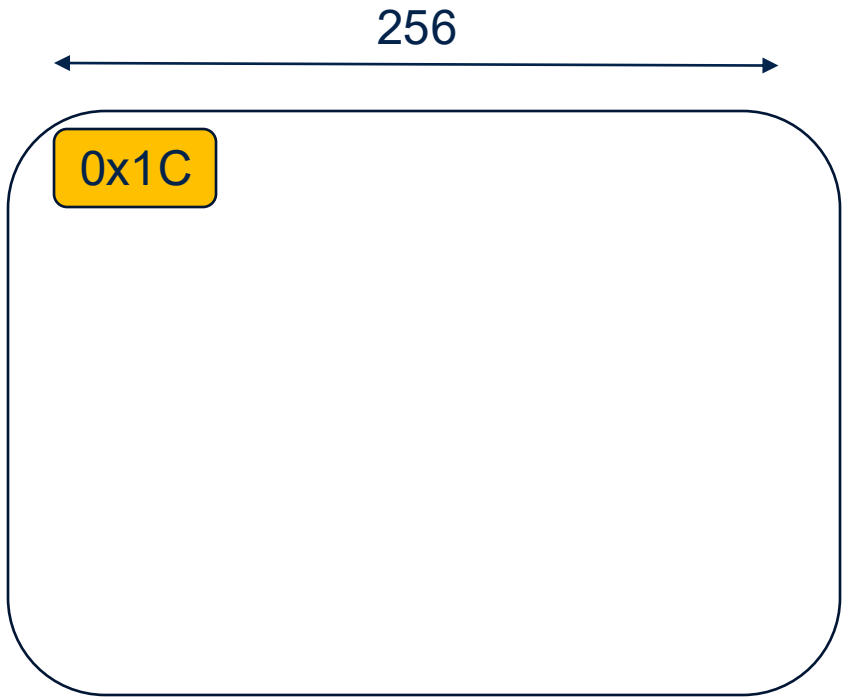
Guess 0 Guess 1 Guess 255

life.augmented

Selection Function
Key Byte 0



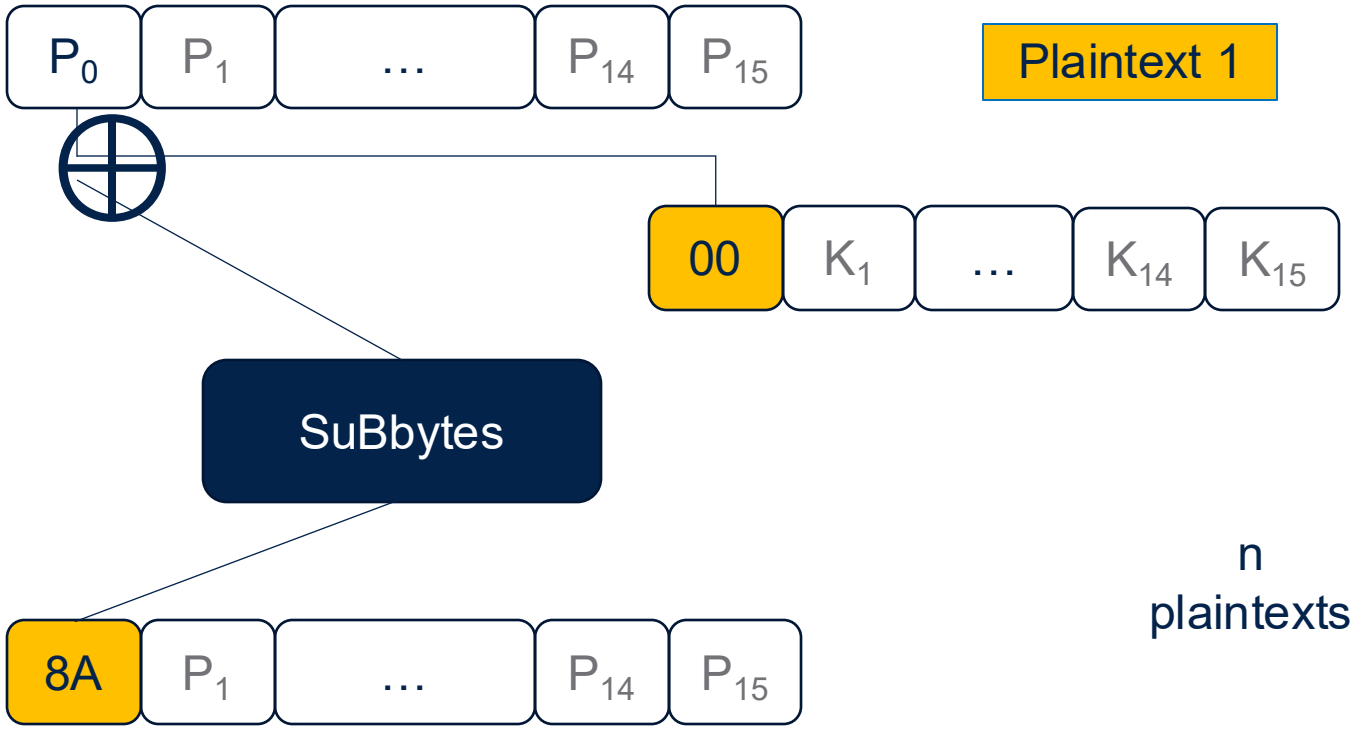
n
plaintexts



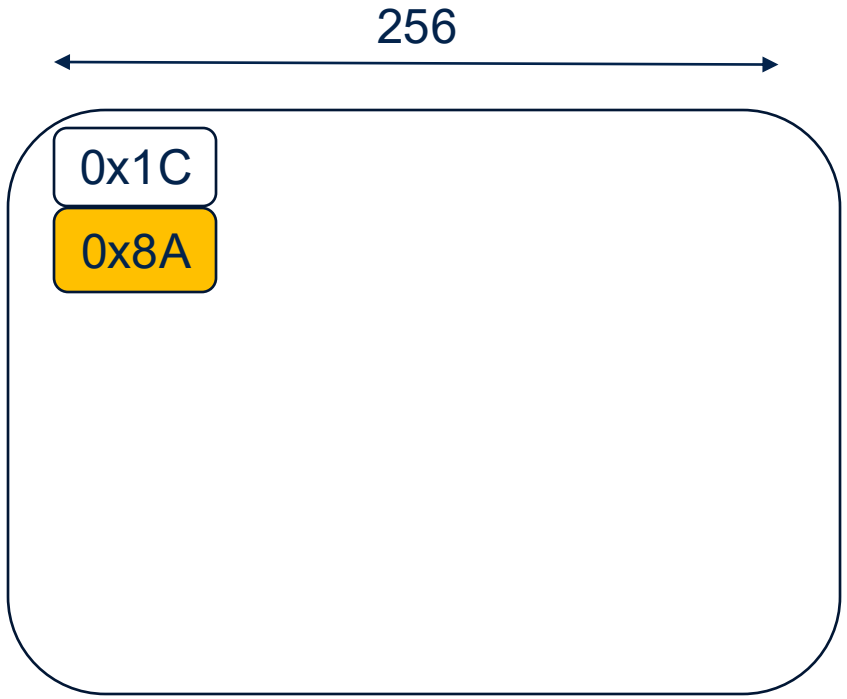
Guess 0 Guess 1 Guess 255

life.augmented

Selection Function
Key Byte 0



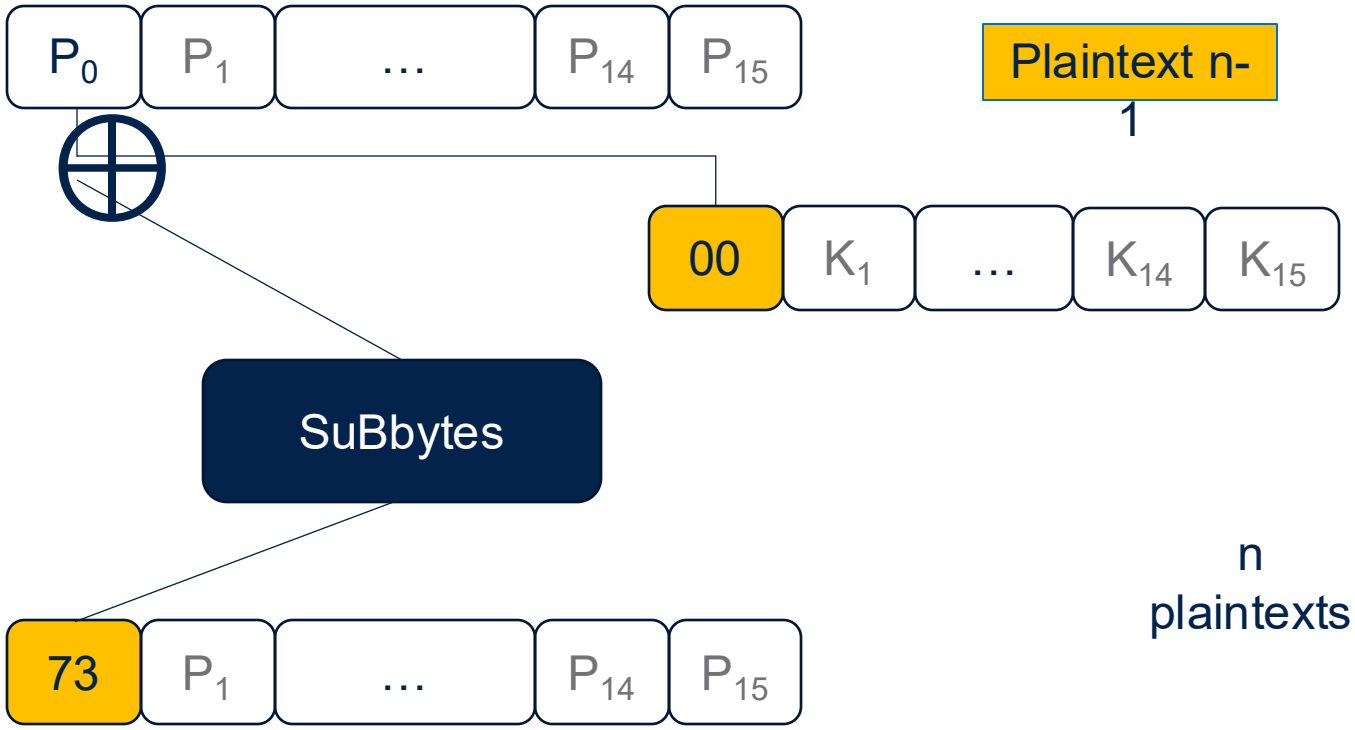
n
plaintexts



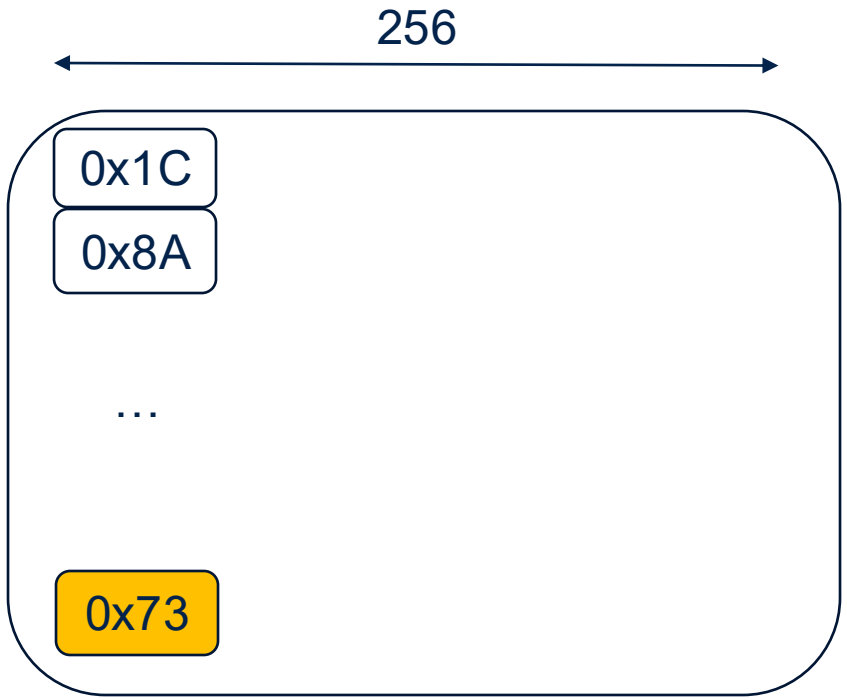
Guess 0 | Guess 1 | Guess 255

life.augmented

Selection Function
Key Byte 0



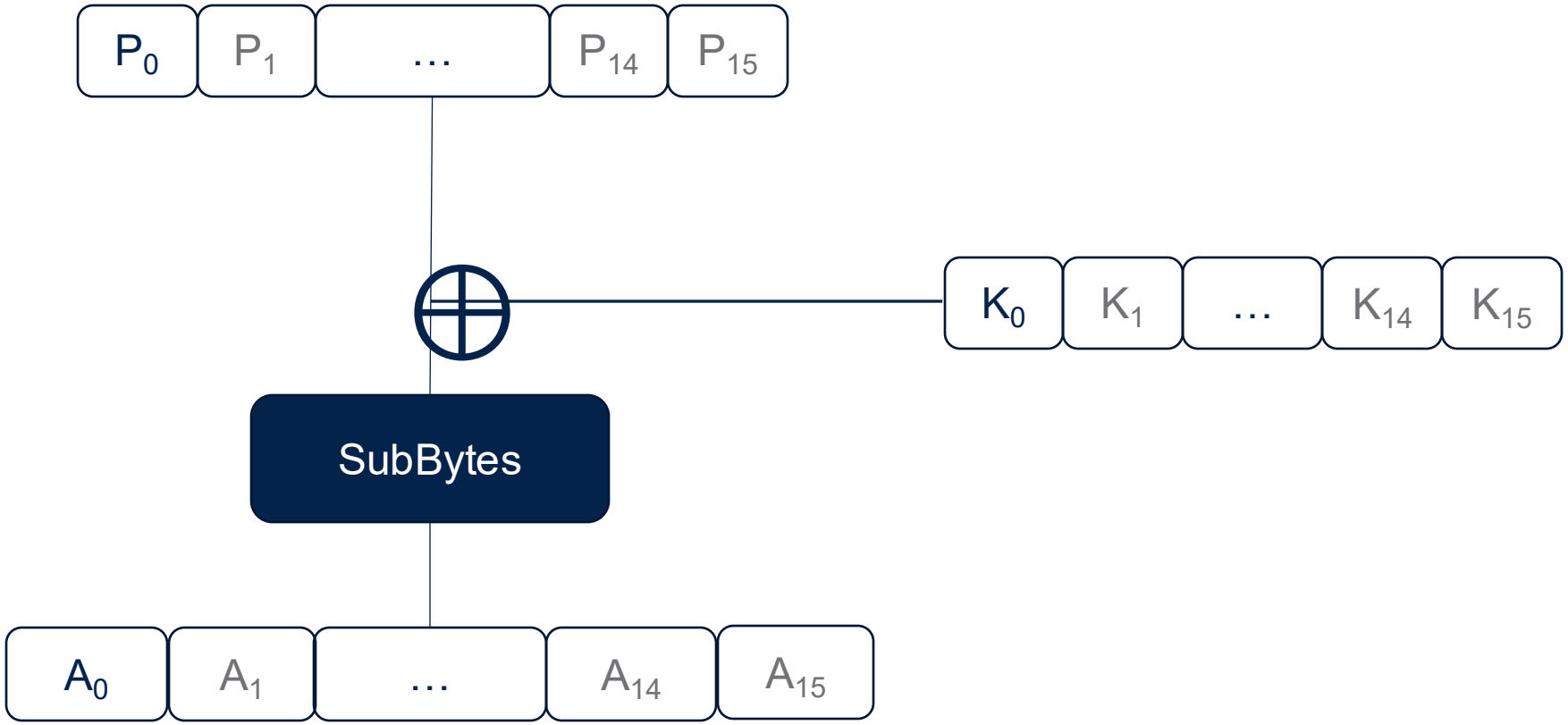
n plaintexts



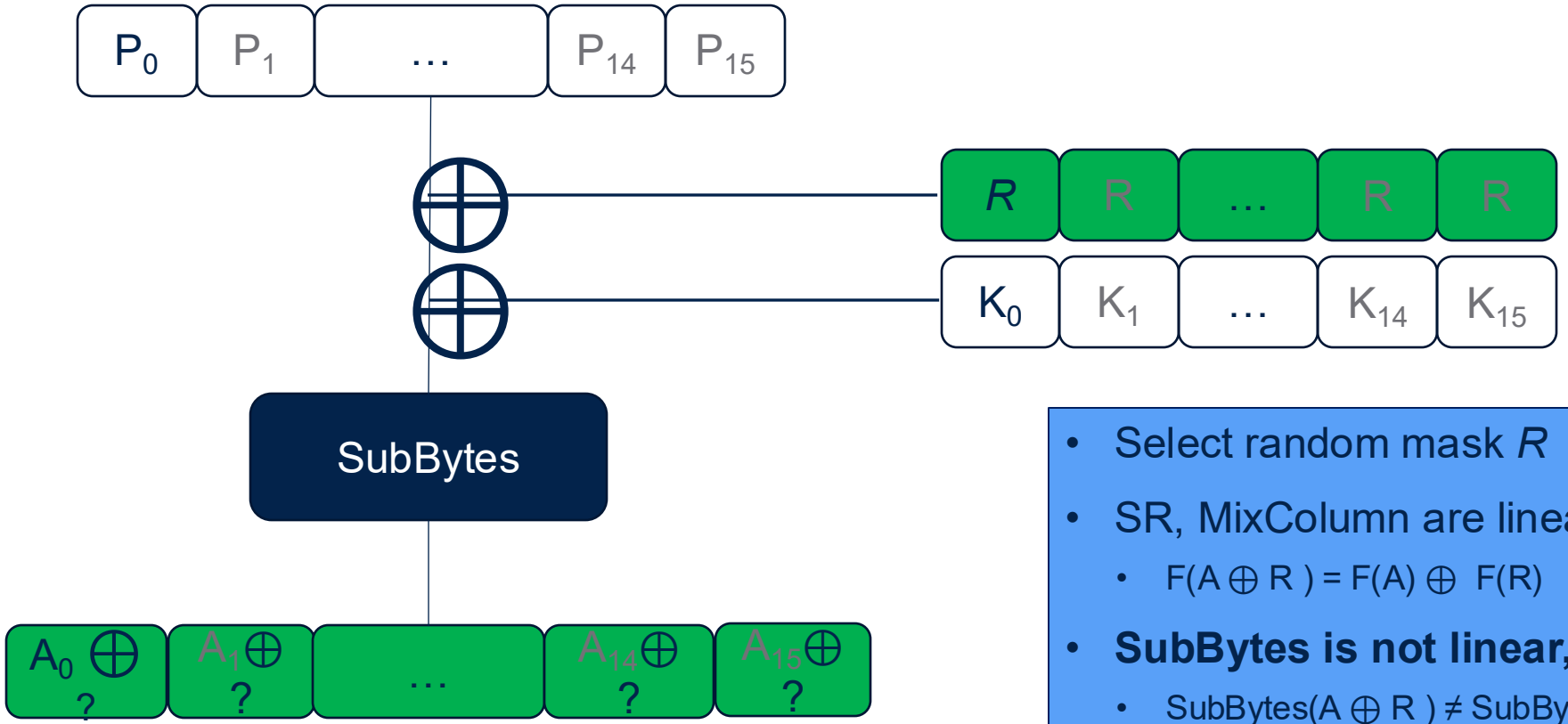
Guess 0 Guess 1 Guess 255

life.augmented

AES



AES side-channel masking

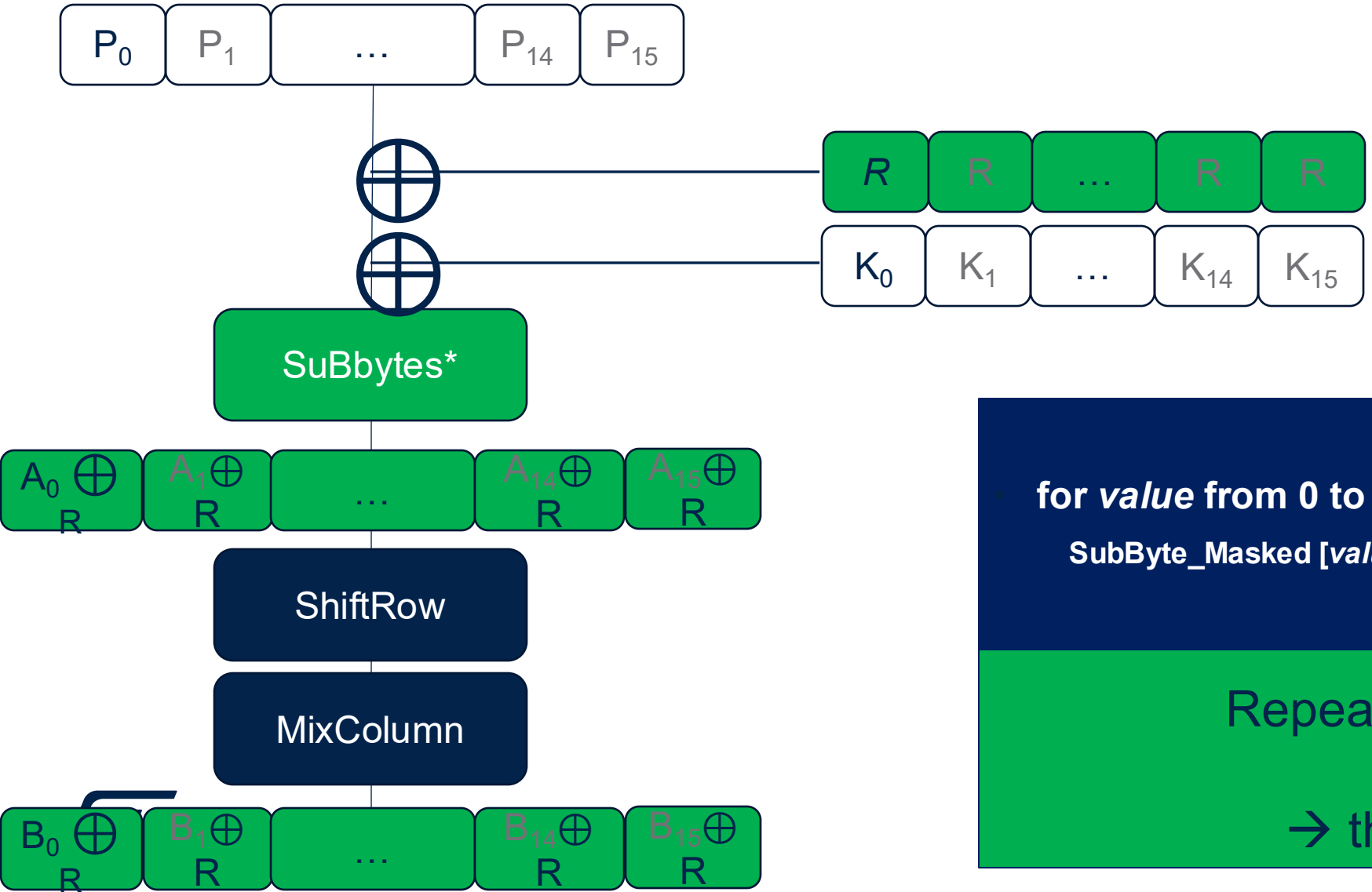


- Select random mask *R*
- SR, MixColumn are linear, easy to mask and unmask
  - F(A ⊕ R) = F(A) ⊕ F(R)
- **SubBytes is not linear, more complex operation**
  - SubBytes(A ⊕ R) ≠ SubBytes(A) ⊕ SubBytes(R)

How to protect the SubBytes ?

life.augmented

AES side channel masking



- **for *value* from 0 to 255**

  SubByte_Masked [*value* ⊕*R*] = SubByte[*value*] ⊕ *R*

Repeat on all Rounds

…

→ then unmask!

More …

- Can be done similarly on T table implementation of AES
- Bigger tables require more memory
- Faster

High Order Side-Channel Analysis

Statistical Analysis of Second Order Differential Power Analysis*

Emmanuel Prouff[1], Matthieu Rivain[2] and Régis Bévan[3]

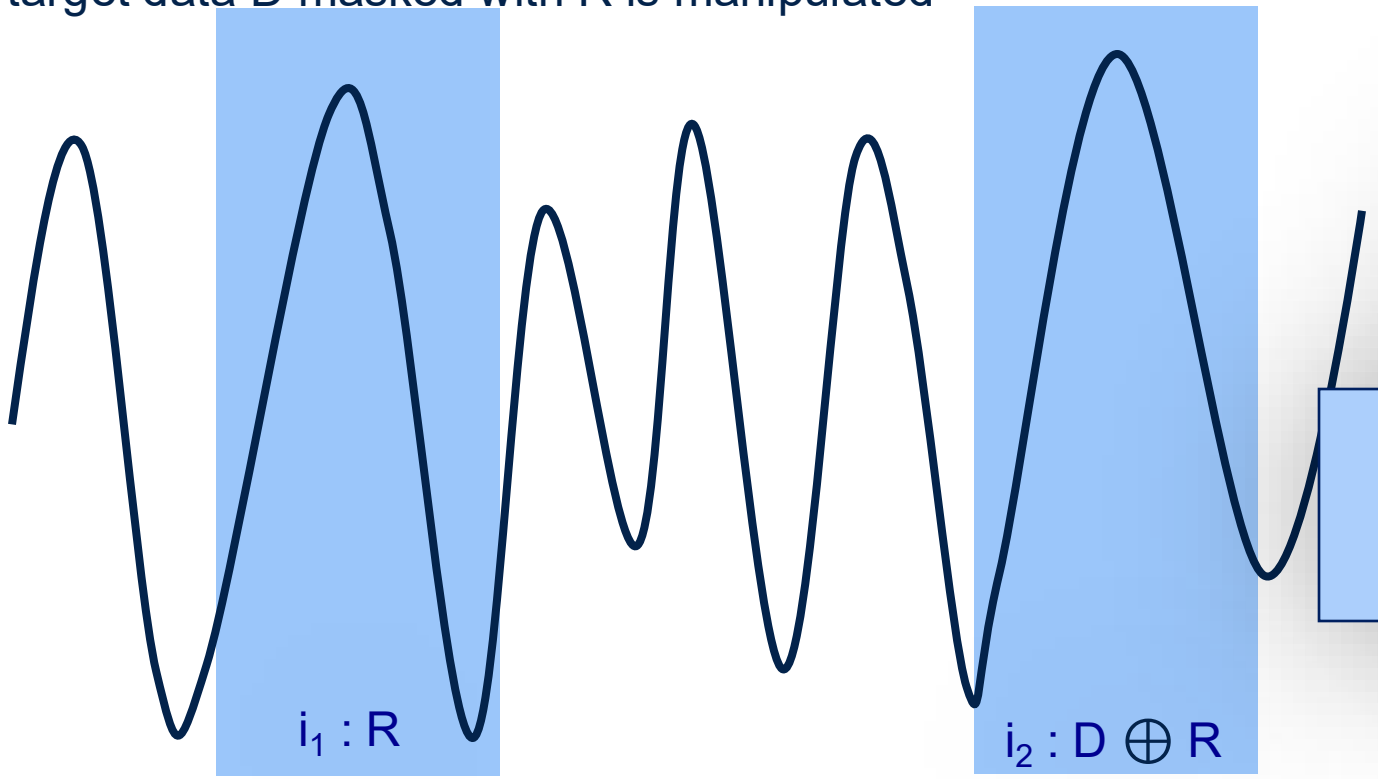
***Abstract*— Second Order Differential Power Analysis (2O-DPA) is a powerful side channel attack that allows an attacker to bypass the widely used masking countermeasure. To thwart 2O-DPA, higher order masking may be employed but it implies an [illegible]verhead. In this context, there is a need to know**

considered in Profiling SCA, as the focused adversary is only able to observe the device behavior and has no *a priori* knowledge of the implementation details. This paper only deals with the set of DPA as it includes a great majority of the attacks encountered

Second (Higher) Order Attacks

Find two instants on the trace where:

- i_1: the mask R is manipulated
- I_2: target data D masked with R is manipulated



$R \oplus (D \oplus R) = D$

$HW(R) \oplus HW(D \oplus R) \neq HW(D)$

$HW(R) \oplus HW(D \oplus R) \approx HW(D)$

life.augmented

Reminder on Second order attacks

- 2nd Order Attack Principle:
  - Find two instants i_1 and i_2 on the trace(s) where the mask *R* is manipulated
  - Combine those two instants i_1 and i_2 to remove the mask influence
  - Perform classical 1st order attack

- Method 1: Combine i_1 and i_2 with Absolute Difference :
  - $T^* = abs(i_1 – i_2)$
  - Correlation (T*, guesses)

- Method 2: Combine i_1 and i_2 with Normalised (Centered) Product
  - E_1 the mean of i_1 and E_2 the mean of i_2
  - $T^* = (i_1 - E_1)*(i_2 - E_2)$
  - Correlate (T*, selection_function_values)

More second order attacks

- Can be difficult to locate R
- Combine SubBytes($P_0 \oplus R$) with SubBytes($P_1 \oplus R$)
  - Requires 2^{16} guesses instead of 2^8
- Combine Round-1- SubBytes($P_0 \oplus R$) with Round-10-SubBytes($P_0 \oplus R$)
- Requires 2^{16} guesses instead of 2^8

- …

Second Order Attack Demonstration

- See notebook demonstration on second order attack.

How to improve resistance

- Diversify the masks
  - One random mask per byte on linear operations
  - Change mask at each round
  - Easy to handle for ShiftRow and MixColumn
- One random byte per SubByte operation in the round
  - Requires 16 masked new SubBytes
- Add Shuffling: random order execution
  - Mainly on SubBytes operations
- Add one AES among N countermeasure
- Implement SubBytes differently
  - Can perform inversion in GF(16) instead of GF(256)
  - Smaller table, more masks, more permutations
- Implement the inversion in GF(256) and mask the operation with a combination of multiplicative (additive and Boolean masking)
- It will make the second order attack theoretically possible but more complex to perform in practice

Infective Computation

- Add integrity or MAC
- If not correct execute but with fake key instead of correct one
- Make the attacker losing time on fake data
- Attacker must not detect the key is fake (you detected him)

Multiplicative Masking Technique

- Several countermeasures (publication) involve multiplicative masking

$$\begin{aligned} A \oplus X &\longrightarrow (A \oplus X) \otimes X = A \otimes X \oplus X^2 \\ A \otimes X \oplus X^2 &\longrightarrow A \otimes X \oplus X^2 \oplus X^2 \end{aligned}$$

  - Boolean mask is now multiplicative
  - Inversion gives:

$$\begin{aligned} A \otimes X &\longrightarrow (A \otimes X)^{-1} = A^{-1} \otimes X^{-1} \\ A^{-1} \otimes X^{-1} &\longrightarrow A^{-1} \otimes X^{-1} \oplus 1 \\ A^{-1} \otimes X^{-1} \oplus 1 &\longrightarrow (A^{-1} \otimes X^{-1} \oplus 1) \otimes X = A^{-1} \oplus X \end{aligned}$$

- Not sensitive to high order attacks

- Sensitive to **ZERO VALUE** attacks

life.augmented

d^{th} order resistant implementations

- Split the data D and key *K* to be protected into d+1 random shares to prevent the product from d^{th} high order side-channel attacks.

- The principle is the following:
  - Generate d random values $r_1, \ldots, r_d$
  - Generate d random values $k_1, \ldots, k_d$
  - Compute $r_0 = D \oplus r_1 \oplus \ldots \oplus r_d$
  - Compute $s_0 = K \oplus s_1 \oplus \ldots \oplus s_d$

- $D \oplus K = (r_0 \oplus s_0) \oplus (r_1 \oplus s_1) \oplus \ldots \oplus (r_d \oplus s_d)$

life.augmented

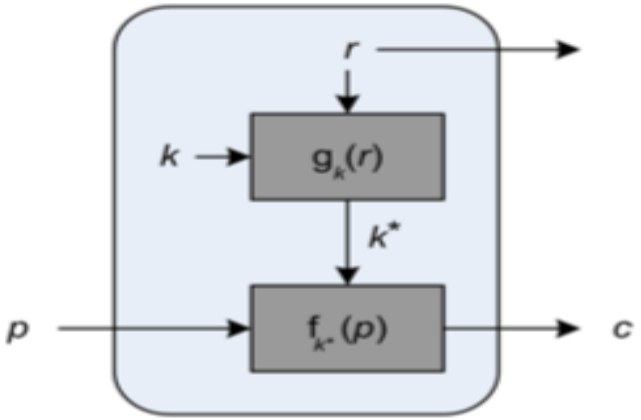
d^{th} order resistant implementations

- ISW (Ishai – Sahai – Wager) scheme
  - Y. Ishai, A. Sahai, and D. Wagner. Private Circuits: Securing Hardware against Probing Attacks. In D. Boneh, editor, Advances in Cryptology – CRYPTO 2003,
  - AND and NOT gates circuit
- M. Rivain and E. Prouff
  - Provably Secure Higher-Order Masking of AES. CHES 2010
- L. Goubin and A. Martinelli
  - Protecting AES with Shamir's Secret Sharing Scheme. CHES 2011.

Fresh Rekeying

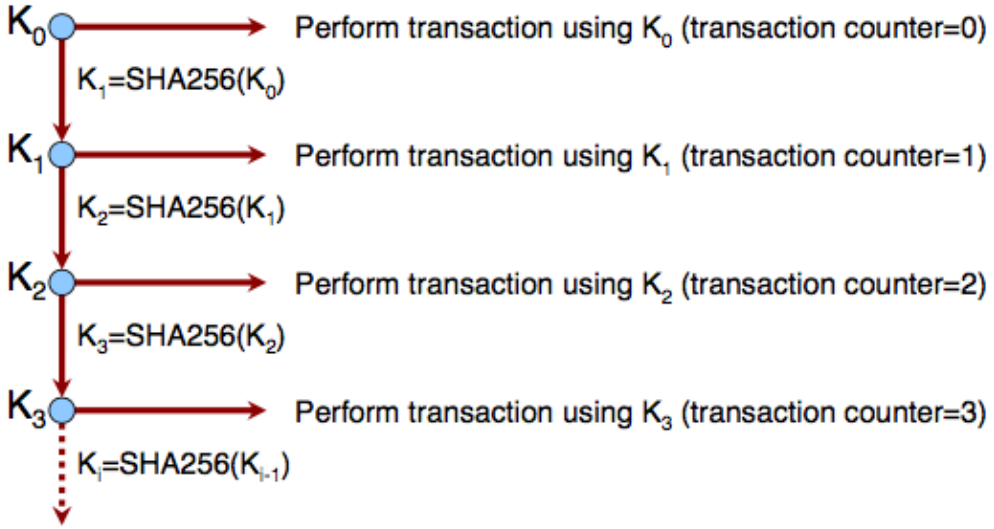
- Provide side-channel resistance by
  - limiting the side-channel attacks paths as different keys are used in the computations.
  - Re-Keying can be done externally via the protocol or internally.
  - We focus here on internal Re-Keying.

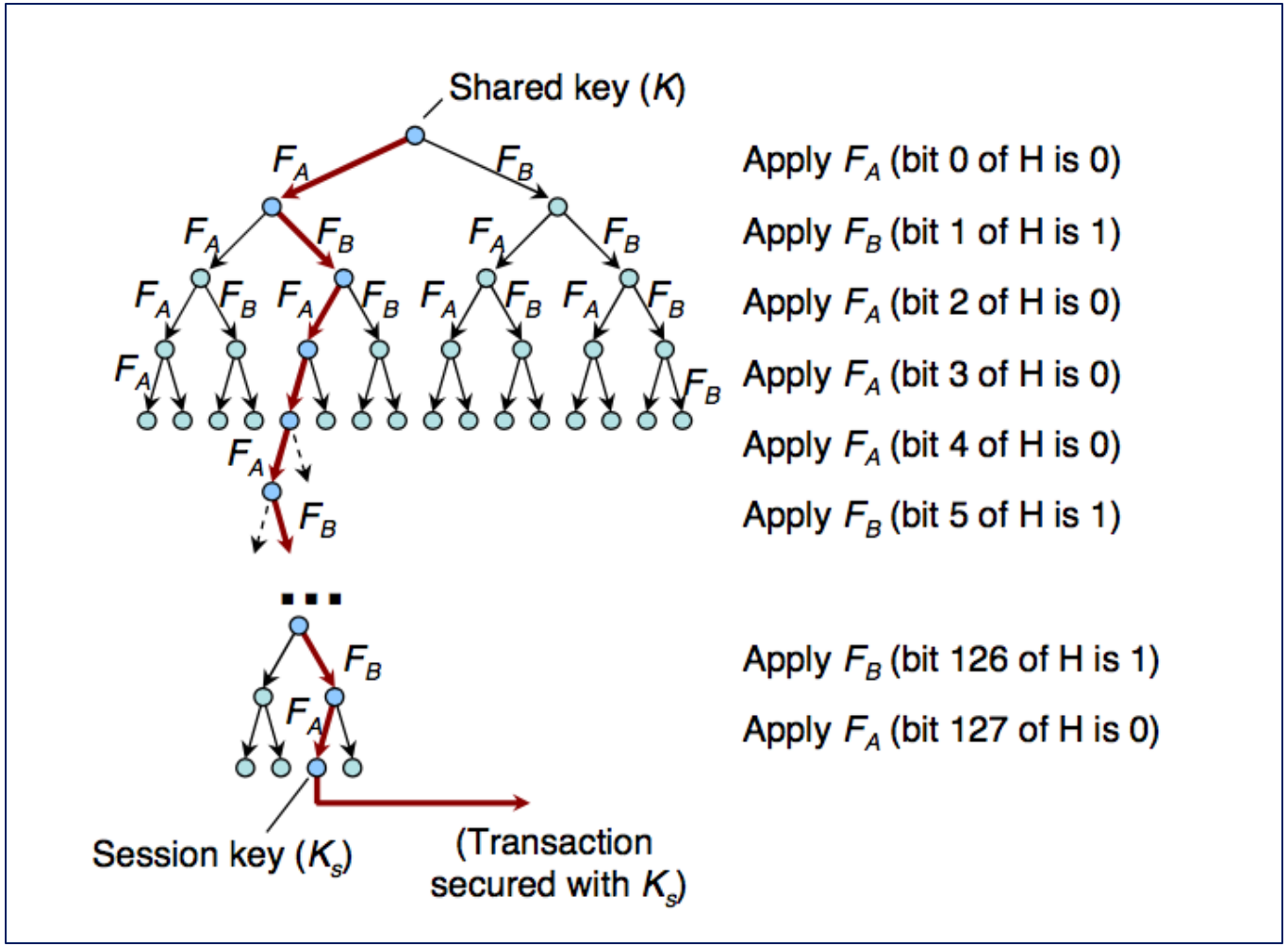
- designing a function *g* to re-key a block cipher *f*
  - with a master and constant values with a random nonce *r*.
  - Then each encryption of a plaintext block is performed with a fresh key *k** such that the ciphertext is $c = f_{k^*}(p)$.
  - It can also be decided to relax the model and to use a fresh key
    - for a given number *w* of plaintext blocks encryptions
    - ensuring the number *L* is small enough to prevent the TOE from profiled and not profiled attacks.



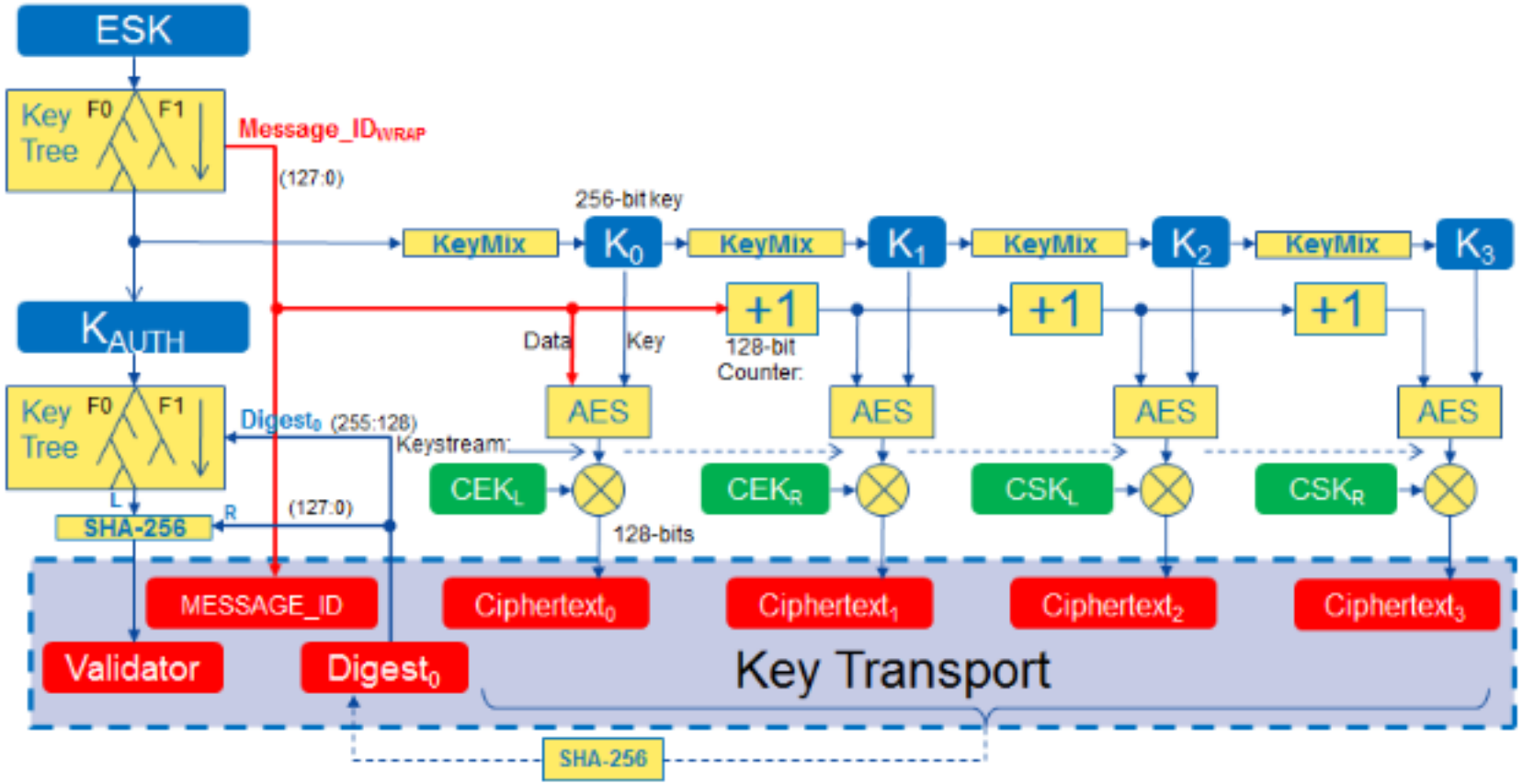
- Make use of functions more resistant to side-channel than others …

Tree Key


K0
Perform transaction using K0 (transaction counter=0)
K1=SHA256(K0)
K1
Perform transaction using K1 (transaction counter=1)
K2=SHA256(K1)
K2
Perform transaction using K2 (transaction counter=2)
K3=SHA256(K2)
K3
Perform transaction using K3 (transaction counter=3)
Ki=SHA256(Ki-1)


Shared key (K)
FA
FB
Apply FA (bit 0 of H is 0)
Apply FB (bit 1 of H is 1)
Apply FA (bit 2 of H is 0)
Apply FA (bit 3 of H is 0)
Apply FA (bit 4 of H is 0)
Apply FB (bit 5 of H is 1)
Apply FB (bit 126 of H is 1)
Apply FA (bit 127 of H is 0)
Session key (Ks)
(Transaction secured with Ks)

Real Example in Secure Boot



Back to RSA

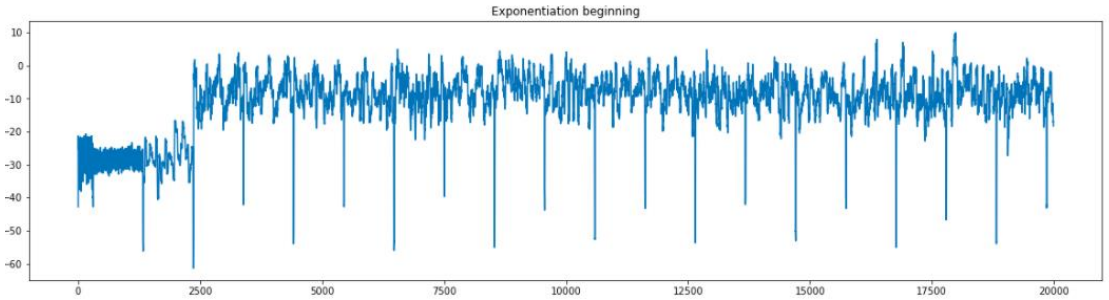
How to protect RSA from SSCA .. Use an atomic exponentiation

- Squaring and multiplication are done using the same operation *Multiply Always* exponentiation

Alg. 1.2.7 Atomic exponentiation

Input: integers m and n with $m < n$, $\ell \cdot t$-bit exponent $d = (d_{\ell \cdot t-1} d_{\ell \cdot t-2} \ldots d_1 d_0)_2$
Output: $\mathsf{Exp}(m,d,n) = m^d \bmod n$

1. $R_0 \leftarrow 1$
2. $R_1 \leftarrow m$
3. $i \leftarrow \ell \cdot t - 1$; $\alpha \leftarrow 0$
4. **while** $i \geqslant 0$ **do**
5. $\quad R_0 \leftarrow \mathsf{ModMul}(R_0, R_\alpha, n)$
6. $\quad \alpha \leftarrow \alpha \oplus d_i$;
7. $\quad i \leftarrow i - 1 + \alpha$
8. **return** R_0



- **Previous simple analysis is not efficient anymore.**

… Or with Regular Implementations

Alg. 1.2.5 Square-and-multiply always regular exponentiation

Input: integers m and n with $m < n$, k-bit exponent $d = (d_{k-1}d_{k-2}\ldots d_1 d_0)_2$
Output: $\mathsf{Exp}(m,d,n) = m^d \bmod n$

1. $R_0 \leftarrow 1$, $R_1 \leftarrow 1$; $R_2 \leftarrow m$
2. **for** $i = k-1$ **down to** 0 **do**
3. $\quad R_1 \leftarrow \mathsf{ModSquare}(R_1, n)$
4. $\quad R_{d_i} \leftarrow \mathsf{ModMul}(R_1, R_2, n)$
5. **return** R_1

Alg. 1.2.6 Montgomery Ladder Exponentiation

Input: $m, n \in \mathbb{N}$, $m < n$, $d = (d_{k-1}d_{k-2}\ldots d_0)_2$
Output: $m^d \bmod n$

1. $R_0 \leftarrow 1$; $R_1 \leftarrow m$
2. **for** $i = k-1$ **to** 0 **do**
3. $\quad R_{1-d_i} \leftarrow \mathsf{ModMul}(R_0, R_1, n)$
4. $\quad R_{d_i} \leftarrow \mathsf{ModSquare}(R_{d_i}, n)$
5. **return** R_0

… Not enough to prevent DSCA …

- See demo notebook for DSCA on Exponentiation

Blinding

- Multiplicative message blinding

  $M^* = r^e \bmod N$ r a large random value, e the public exponent

- Additive message blinding

  $M^* = M + r_1.N \bmod r_2.\mathrm{N}$ r_1, r_2 large random values

There are still possible attacks

- Cross Correlation
- Distinguishing Squaring from Multiplying operations
- The Doubling (Squaring) Attack

Add the *exponent blinding* countermeasure

Euclidean division exponent blinding

$d = r.d_1 + d_2$ r a large random value > 128 bits

→ $M^d = (M^r)^{d1} \times M^{d2} \bmod N$

Additive exponent blinding

$d^* = d + r.\varphi(N)$ r *a* large random value > 128 bits, φ *the Euler function*

Exponent split

$d_1 = d - r$ r *a* large random value > 128 bits

→ $M^d = M^{d1} \times M^r \bmod N$

Combining Several Countermeasures

- Example combining:
  - Atomic exponentiation
  - Additive message blinding
  - Additive exponent blinding

Alg. 1.2.8 Blinded exponentiation

Input: integers m and n with $m < n$, $\ell \cdot t$-bit exponent $d = (d_{\ell \cdot t-1} d_{\ell \cdot t-2} \ldots d_1 d_0)_2$, a security parameter λ

Output: $\mathsf{Exp}(m,d,n) = m^d \bmod n$

1. $r_1 \leftarrow random(1, 2^\lambda - 1)$
2. $r_2 \leftarrow random(1, 2^\lambda - 1)$
3. $r_3 \leftarrow random(1, 2^\lambda - 1)$
4. $\bar{n} \leftarrow r_2 \cdot n$
5. $R_0 \leftarrow 1 + r_1 \cdot n \bmod \bar{n}$
6. $R_1 \leftarrow m + r_1 \cdot n \bmod \bar{n}$
7. $\bar{d} \leftarrow d + r_3 \cdot \varphi(n)$
8. $i \leftarrow \ell \cdot t + \lambda - 1$; $\alpha \leftarrow 0$
9. **while** $i \geqslant 0$ **do**
10. $\quad R_0 \leftarrow \mathsf{ModMul}(R_0, R_\alpha, \bar{n})$
11. $\quad \alpha \leftarrow \alpha \oplus d_i$;
12. $\quad i \leftarrow i - 1 + \alpha$
13. $R_0 \leftarrow R_0 \bmod n$
14. **return** R_0

Single Trace Attack on blinded RSA with Clustering techniques

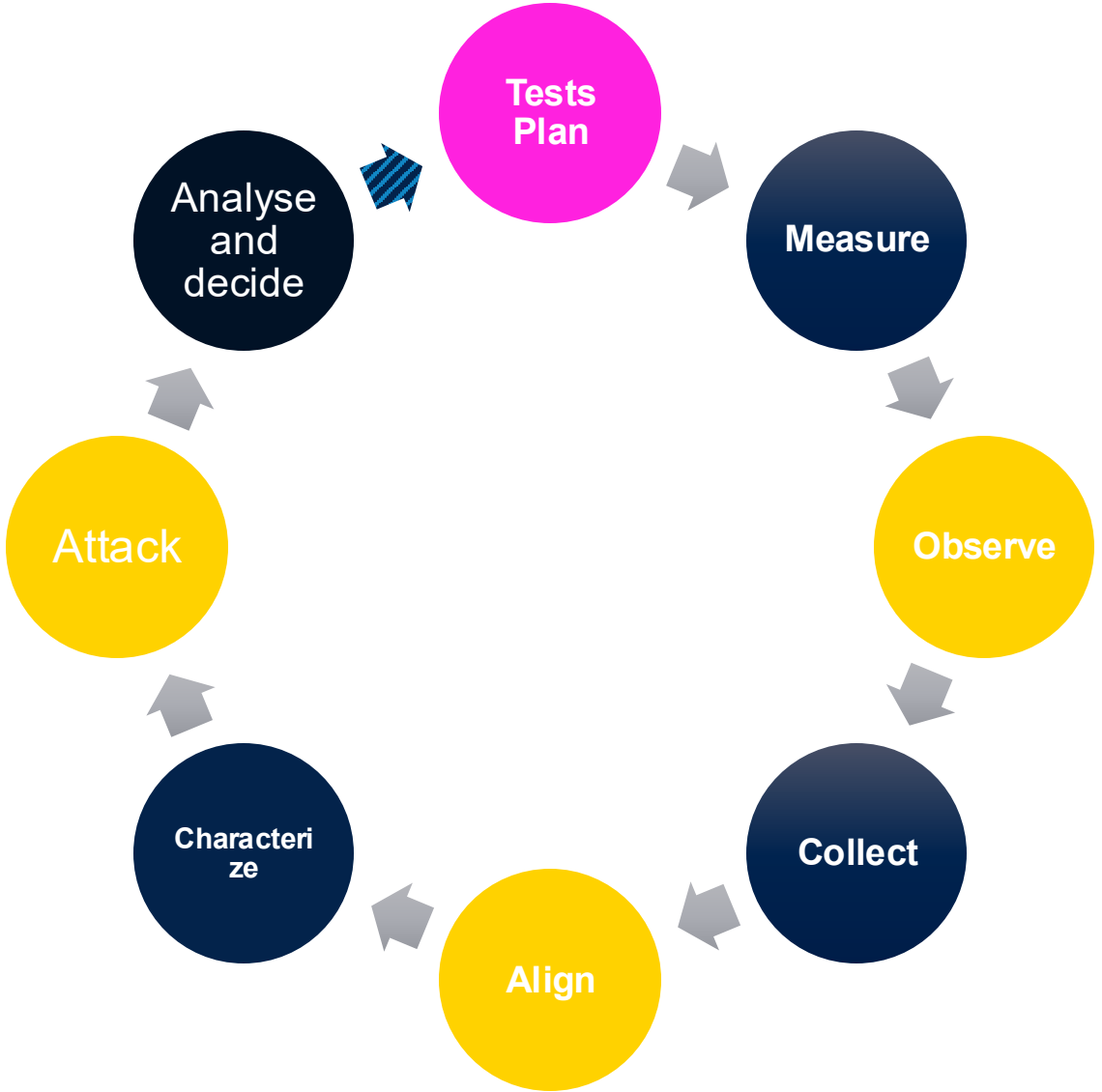
- See demo on RSA SMA

WITH
CLUSTERING

https://scikit-leaxrn.org/stable/modules/clustering.html

Side-Channel Methodology

Side Channel Methodology



Side Channel Attacks Methodology
Step 1: Tests Plan

- Evaluation mode
  - Knowledge on the product is given for the evaluation
  - Source code, IC design, cryptographic operations and IPs, etc. information given
- Vulnerability Analysis
  - Identify potential attack if information available
- Test Plan
  - Define tests based on Vulnerability Analysis if it was performed.

life.augmented

Side Channel Attacks Methodology
Step 2: Measurement

- Measurement set-up
  - Setup the measurement
    - Select probe, location
    - Sampling rate,
    - EM (preferably often)
    - Triggering
  - Identify the operation you are targeting
  - Find the best signal for the operation targeted,
- Cartography can help

Side Channel Attacks Methodology
Step 3: Locate / Process with Few traces Observation

- **What is the knowledge you can get to locate and identify?**

- Observe the trace
  - What can you identify?
  - Seeing the algorithm rounds? rounds operation?

- Chosen input
  - Depending on the leakage model of the device for the targeted operation, fixing a certain number of bytes to a constant (i.e., 0) can make appear specific patterns in the trace to locate the interesting area in the trace or even to recover secret.

- Signal Processing
  - Apply signal processing techniques to observe potential loops, identify better the operations related to the computations
  - Investigate better signal processing that will be helpful for aligning the traces once collected

Side Channel Attacks Methodology
Step 4: Collect traces

- Define setup
  - Sampling to use
  - Define Area to collect in the computation for the chosen attack path (first or last rounds for example)
  - Setup for trace collection

- Run the traces collection
  - Collect traces and store
  - 100K? 1M? 10M?
  - Efficient equipment required (up to 1M per day)

life.augmented

Side Channel Attacks Methodology
Step 5: Align

- Alignment
  - Identify which signal processing techniques to combine to align traces
  - Run the alignment(s)

- Critical Path
  - Alignment can determine the success or failure of the attacks
  - Major part of the global effort

Example: observe then align

- See observation notebook on aligned and not aligned traces

Side Channel Attacks Methodolo
Step 5: Characteri

- **What is the knowledge you can get to characterize?**

- Input – Output
  - Knowing the computation input and/or output (plaintext or ciphertext) can help
  - Perform SCA tests on input **to locate the beginning of the targeted operation** in the trace(s)
  - Perform SCA tests on output **to locate the end of the targeted operation** in the trace(s)
  - Require a first traces collection to be done

- Chosen input → TVLA Test Vector Leakage Assessment i.e., T-test …
  - You can select specific values as input **to get more knowledge on the operations and find leakages points, T test**
  - Fixing a certain number of bytes to a constant (i.e., 0) can reduce the algorithmic noise from one operation on the targeted one
  - Depending on the leakage model of the device for the targeted operation, fixing a certain number of bytes to a constant (i.e. 0) can make appear specific patterns in the trace to locate the interesting area in the trace or even to recover secret.

Side Channel Attacks Methodolo
Step 6: Characterize

- **What is the knowledge you can get to characterize?**

- Key Knowledge → *Leakage Detection on Data Computation with Key Knowledge*
  - Knowing the key means you can characterize your implementation (at each step, each round)
    - You can test to correlate each intermediate value of the computation with the trace
    - Answer: YES there is leakage in the implementation → might lead to an attack exploitation
    - Answer: NO there is no obvious leakage in the implementation

- Chosen Key→ *Leakage Detection on Key Manipulation with Key Knowledge*
  - Knowing the key allows to **characterize the key schedule** (T-test, reverse with varying keys)
  - Can be used to build templates on the key manipulation

- **Characterization results provide first test conclusion.**

life.augmented

Side Channel Attacks Methodology
Step 7: Attack

No characterization knowledge

- Run the attacks
- Test different leakages models
- Test all selection functions
- Log results
- Observe results
- Can use thresholds to sort the result at a first stage
  - score value higher to a threshold
  - Ratio between best score and next higher ones.

With characterization knowledge

- Use characterization results to run appropriate attacks
- Tests related attack paths
  (model, selection functions..)
- Log results
- Observe results coherency with characterization

life.augmented

Side Channel Attacks Methodology
Step 8: Analyse and Decide

- Conclude on testing and evaluation
- Or relaunch different tests if there is a plan.

life.augmented

Welsch t-test

- Most used test is the student T-test

- Requires the evaluator can send chosen messages
  - Using random and fix plaintext sets
  - Variance based test
  - Identify 1st order leakages

- Can be turned to higher order leakages detection

TVLA and t test

A testing methodology for side-channel resistance validation

Gilbert Goodwill, Benjamin Jun, Josh Jaffe, Pankaj Rohatgi: Cryptography Research Inc.

Keywords: side-channel testing, leakage analysis, t-test

The initial publication dealing with using the Welch's t test for TVLA is the following one in 2008:
https://csrc.nist.gov/csrc/media/events/non-invasive-attack-testing-workshop/documents/08_goodwill.pdf

Cryptology ePrint Archive

Paper 2017/138

How (not) to Use Welch's T-test in Side-Channel Security Evaluations

François-Xavier Standaert

Abstract

The Test Vector Leakage Assessment (TVLA) methodology is a qualitative tool relying on Welch's T-test to assess the security of cryptographic implementations against side-channel attacks. Despite known limitations (e.g., risks of false negatives and positives), it is sometimes considered as a pass-fail test to determine whether such implementations are "safe" or not (without clear definition of what is "safe"). In this note, we clarify the limited quantitative meaning of this test when used as a standalone tool. For this purpose, we first show that the straightforward application of this approach to assess the security of a masked implementation is not sufficient. More precisely, we show that even in a simple (more precisely, univariate) case study that seems best suited for the TVLA methodology, detection (or lack thereof) with Welch's T-test can be totally disconnected from the actual security level of an implementation. For this purpose, we put forward the case of a realistic masking scheme that looks very safe from the TVLA point-of-view and is nevertheless easy to break. We then discuss this result in more general terms and argue that this limitation is shared by all "moment-based" security evaluations. We conclude the note positively, by describing how to use moment-based analyzes as a useful ingredient of side-channel security evaluations, to determine a "security order".

In our simple (single-bit secret) case, the TVLA methodology works by collecting Q_0 (resp. Q_1) traces corresponding to the secret value $X = 0$ (resp. $X = 1$) and stores them in vectors $\bar{L}_0$ (resp. $\bar{L}_1$). In order to capture higher-order security, and following what was done in [4, 6, 7, 27], we then process these vectors by removing their mean (so that we next estimate central moments) and raise them to a power o, that we will denote as the attack order. This leads to vectors $\bar{L}'_0$ (resp. $\bar{L}'_1$) of which the samples equal (e.g., for $\bar{L}'_0$):

$$\bar{L}'_0(i) = \left(\bar{L}_0(i) - \hat{\mathsf{E}}(\bar{L}_0)\right)^o,$$

with $\hat{\mathsf{E}}$ the sample mean operator and for $1 \leq i \leq Q_0$. Based on these leakage vectors, the TVLA methodology computes Welch's T statistic as follows:

$$\Delta = \frac{\hat{\mathsf{E}}(\bar{L}'_0) - \hat{\mathsf{E}}(\bar{L}'_1)}{\sqrt{\frac{\hat{\mathsf{var}}(\bar{L}'_0)}{Q_0} + \frac{\hat{\mathsf{var}}(\bar{L}'_1)}{Q_1}}},$$

with $\hat{\mathsf{var}}$ the sample variance operator. The side-channel literature usually assumes this T statistic to be significant when a threshold of 5 is passed.[2]

Welsch t test

- T-test examines the leakage of the Device Under Test (DUT) independent of its underlying architecture.
- Gives a level of confidence to conclude that the DUT has an exploitable leakage or not.
- However, it provides no information
  - about the easiness/hardness of an attack which can exploit the leakage,
  - nor about an appropriate intermediate value and the hypothetical model.
- Can easily and rapidly report that the DUT fails to provide the desired security level:
  - e.g., due to a mistake in the design engineering
  - or a flaw in the countermeasure.
- T-test analysis requires to acquire at least two (better more) sets of traces:
  - Set A provides random data
  - Set B provides fix or semi-fix data
  - The targeted data could be any of the algorithm inputs (plaintext, key, mask ...).

- [GJJR] G. Goodwill, B. Jun, J. Jaffe, P. Rohatji - A testing methodology for side-channel resistance validation. https://csrc.nist.gov/csrc/media/events/non-invasive-attack-testing-workshop/documents/08_goodwill.pdf
- [SM15] Tobias Schneider and Amir Moradi. Leakage Assessment Methodology - a clear roadmap for side-channel evaluations. CHES-2015 - https://eprint.iacr.org/2015/207.pdf
- [BCDJKKLMRS] G. Becker, J. Cooper, E. DeMulder, G. Goodwill, J. Jaffe, G. Kenworthy, T. Kouzminov, A. Leiserson, M. Marson, P.Rohatgi and S. Saa. Test Vector Leakage Assessment (TVLA) methodology in practice

t test

A t-statistic trace T based on the Welch t-test is computed on these two sets (i.e. for each sample):

$$T = \frac{\mu_A - \mu_B}{\sqrt{\frac{\sigma_A^2}{Na} + \frac{\sigma_B^2}{N_B}}}$$

with μ_A mean of set A, σ_A sandart deviation of set A (i.e. σ_A^2 is variance) and N_A number of traces in A.

The leaking moments are those where the T value exceeds the confidence threshold C.

- For a number of traces > 100, the pass fail criteria is C > 4.5 or C < -4.5 with a confidence of 99.999%

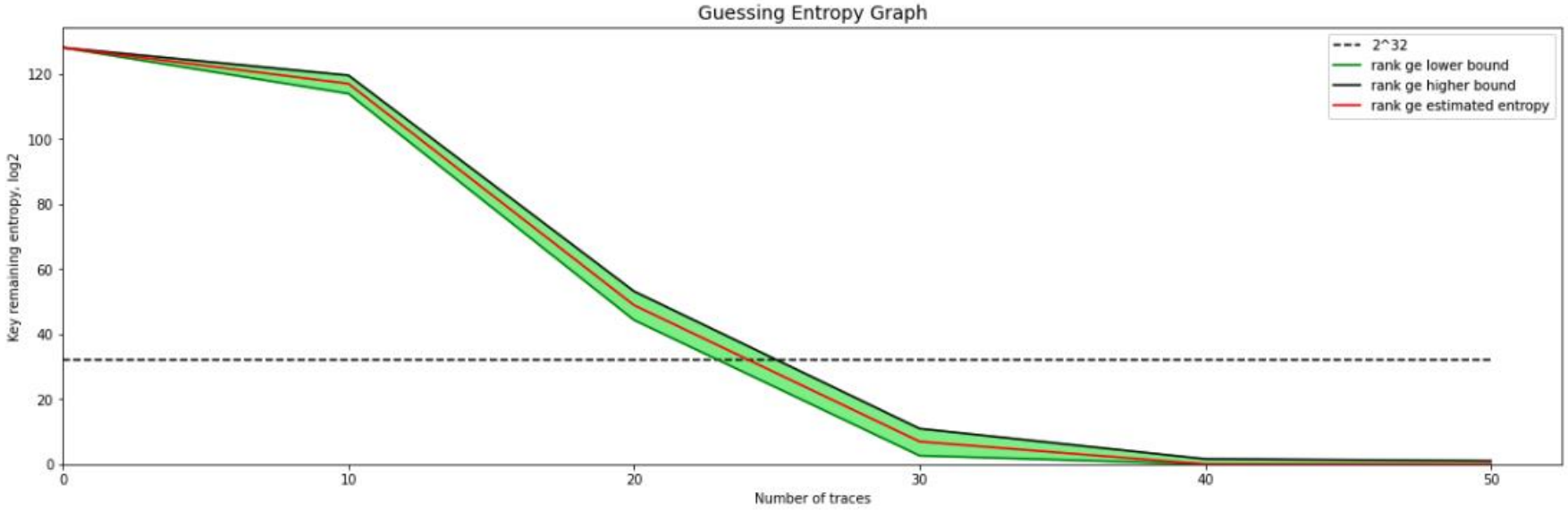
life.augmented

TP of t-test on AES

See demo/TP notebook on AES

Attack and Remaining Key Entropy

- How to evaluate the results of a side-channel attack
  - Do we really need the full key to be recovered?
  - Remaining entropy is the right cursor !
  - https://eprint.iacr.org/2014/920.pdf

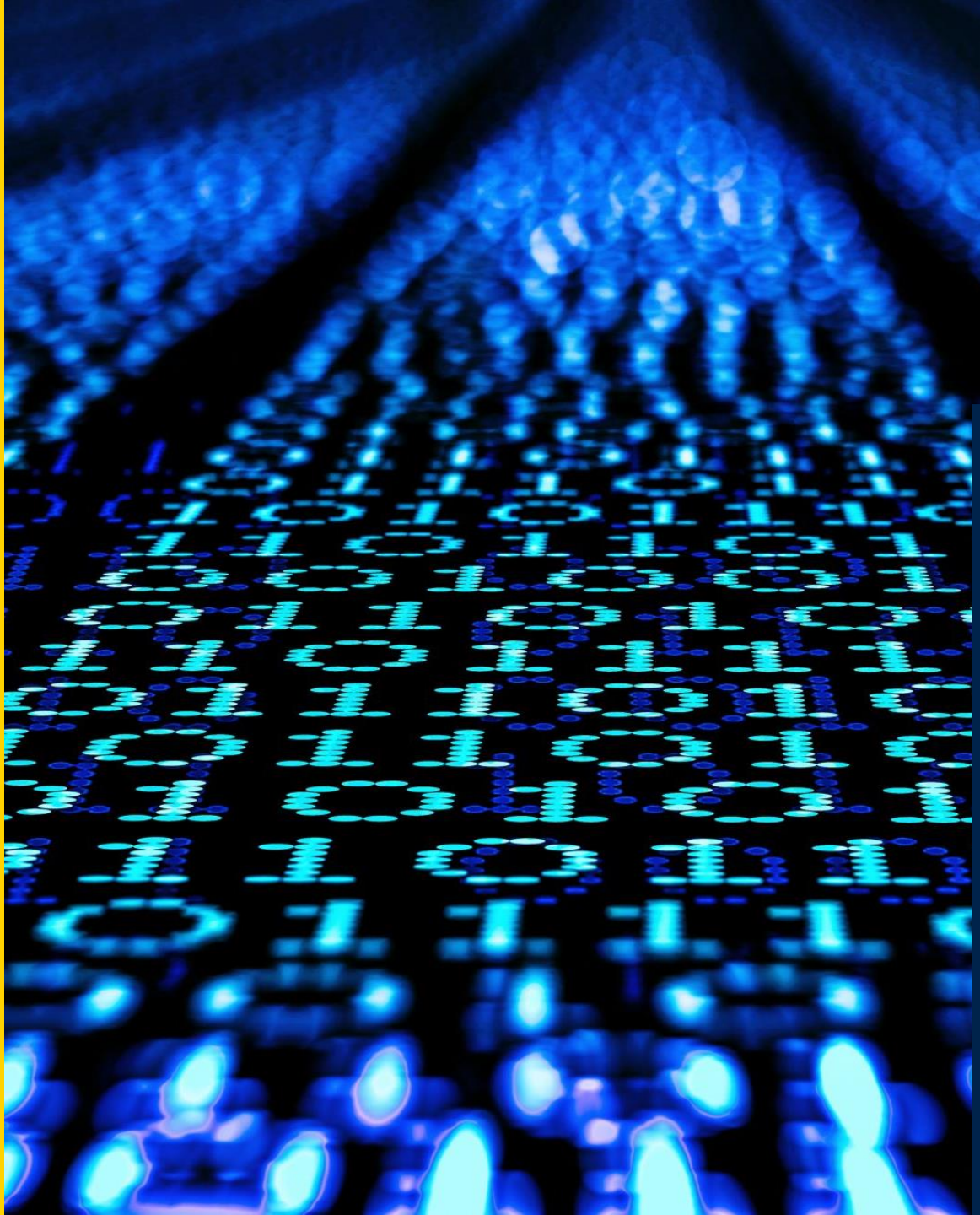


Simpler and More Efficient Rank Estimation for Side-Channel Security Assessment

Cezary Glowacz[1], Vincent Grosso[2], Romain Poussier[2], Joachim Schüth[1], François-Xavier Standaert[2].

[1] T-Systems GEI GmbH, Security Consulting & Engineering, Bonn, Germany.
[2] ICTEAM/ELEN/Crypto Group, Université catholique de Louvain, Belgium.

Abstract. Rank estimation algorithms allow analyzing the computational security of cryptographic keys for which adversaries have obtained partial information thanks to leakage or cryptanalysis. They are particularly useful in side-channel security evaluations, where the key is known by the evaluator but not reachable with exhaustive search. A first instance of such algorithms has been proposed at Eurocrypt 2013. In this paper, we propose a new tool for rank estimation that is conceptually simpler and much more efficient than this previous proposal. It allows approximating the key rank of (128-bit, 256-bit) symmetric keys with very tight bounds (i.e. with less than one bit of error), almost instantaneously and with limited memory. It also scales nicely to larger (e.g. asymmetric) key sizes, for which the previous algorithm was hardly applicable.



life.augmented

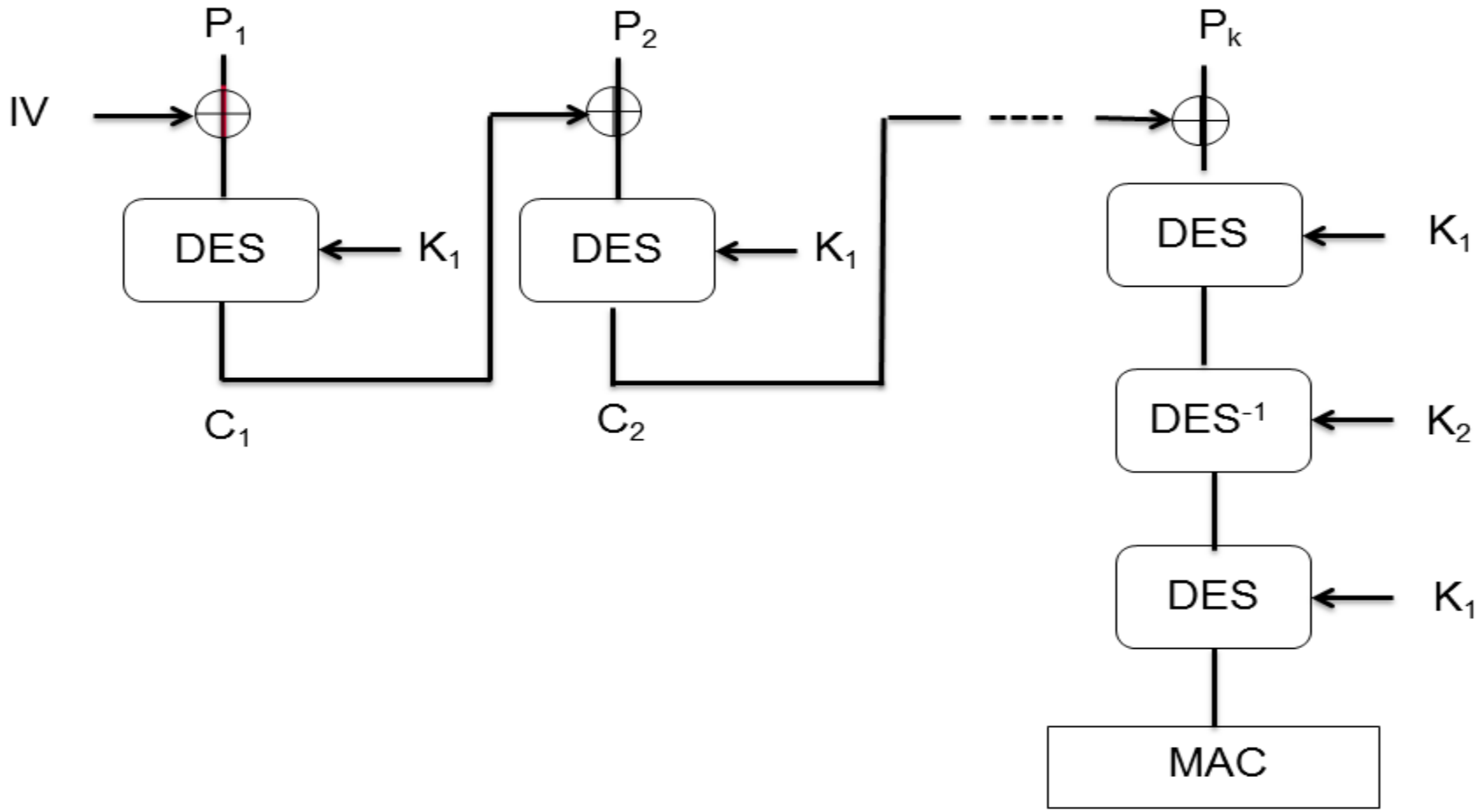
Cryptographic Protocol vs..
Cryptographic Algorithm …

Sometimes there is still space somewhere around the algorithm for something different

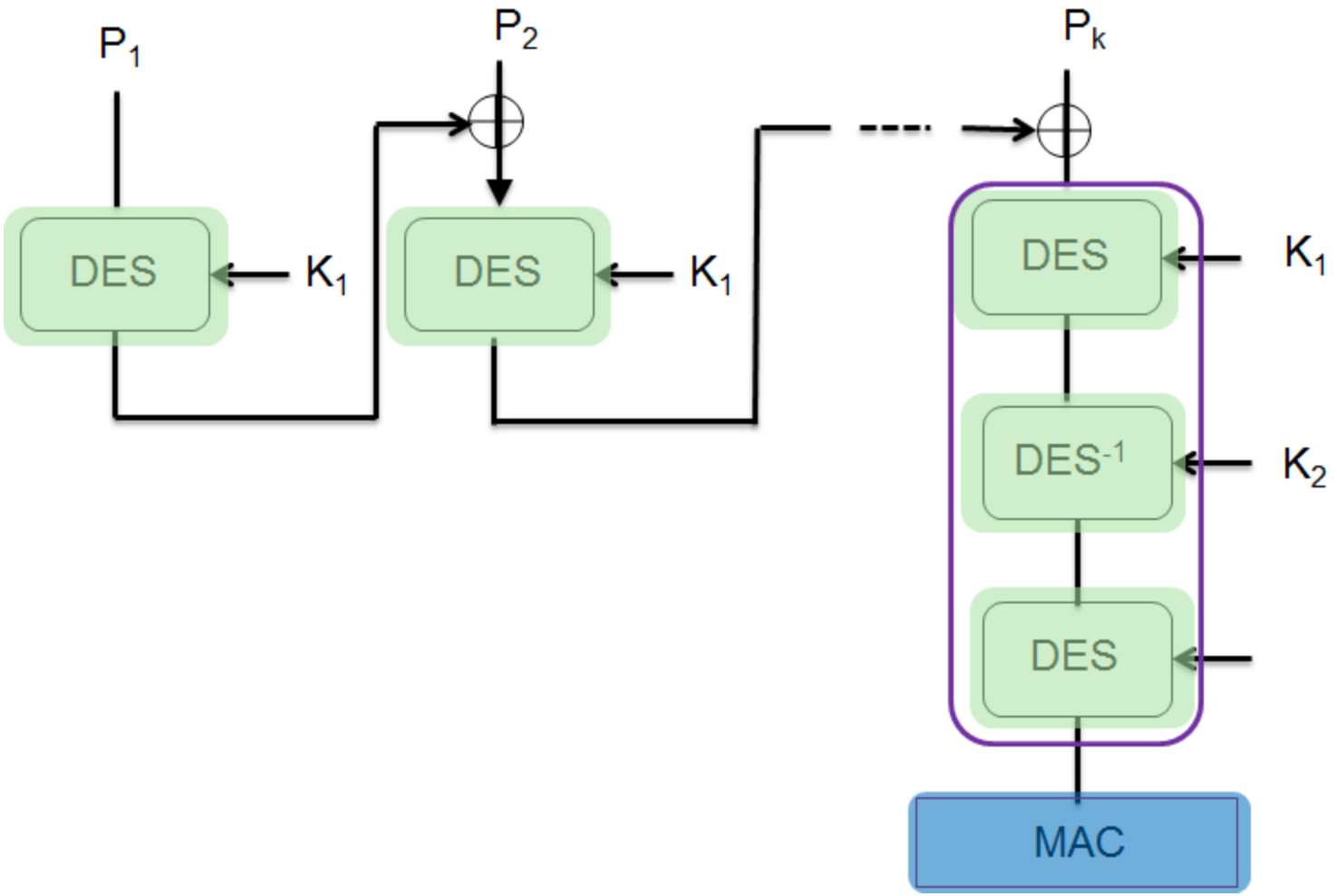
Side-Channel Attack on the MAC Algo3 2015 (only…)

- Developers MUST prevent their product from side-channel attacks
  - First and higher order(s)
  - Cryptographic engineers develop strong DES and AES
- These algorithms are used in protocols
  - Weakness can be in the protocol
  - MAC Algo3 is an ISO protocol
    - DES CBC chaining mode
    - End with a TDES operation

MAC Algo3

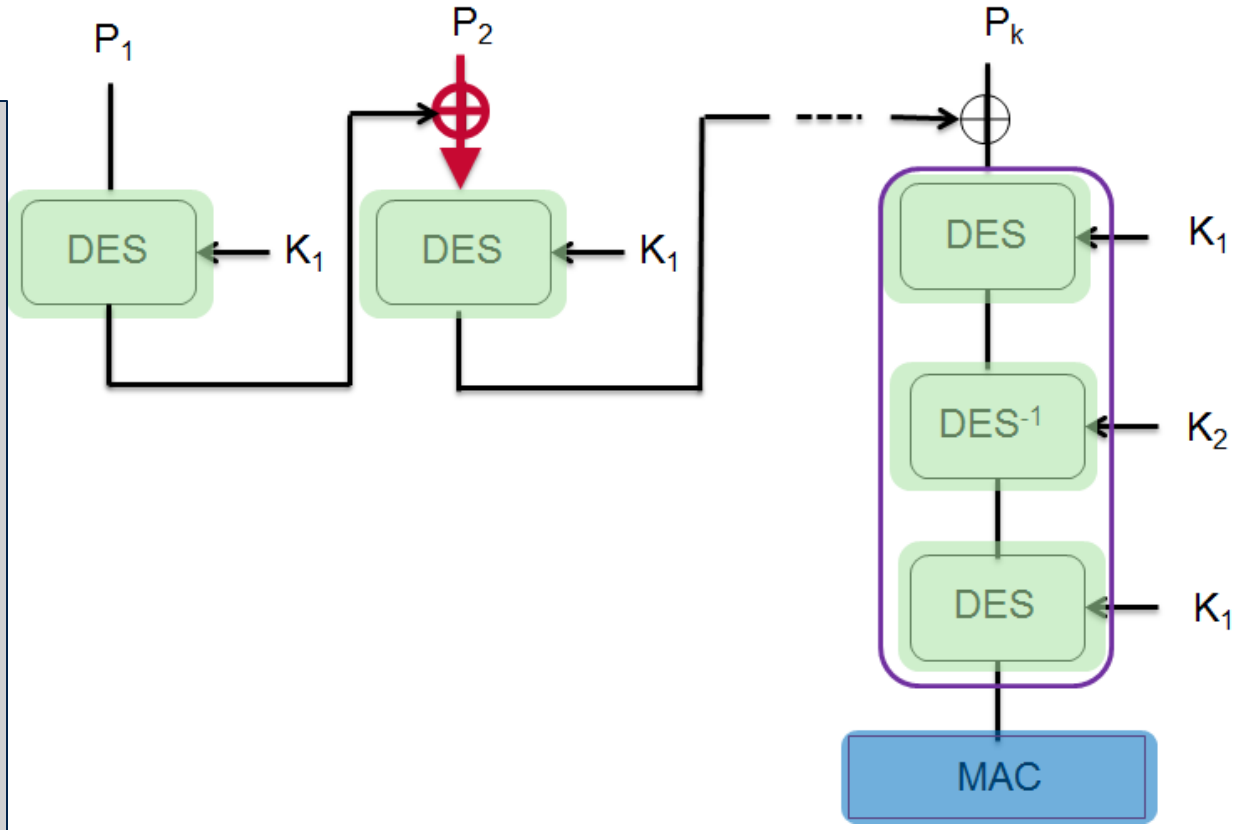


MAC Algo3



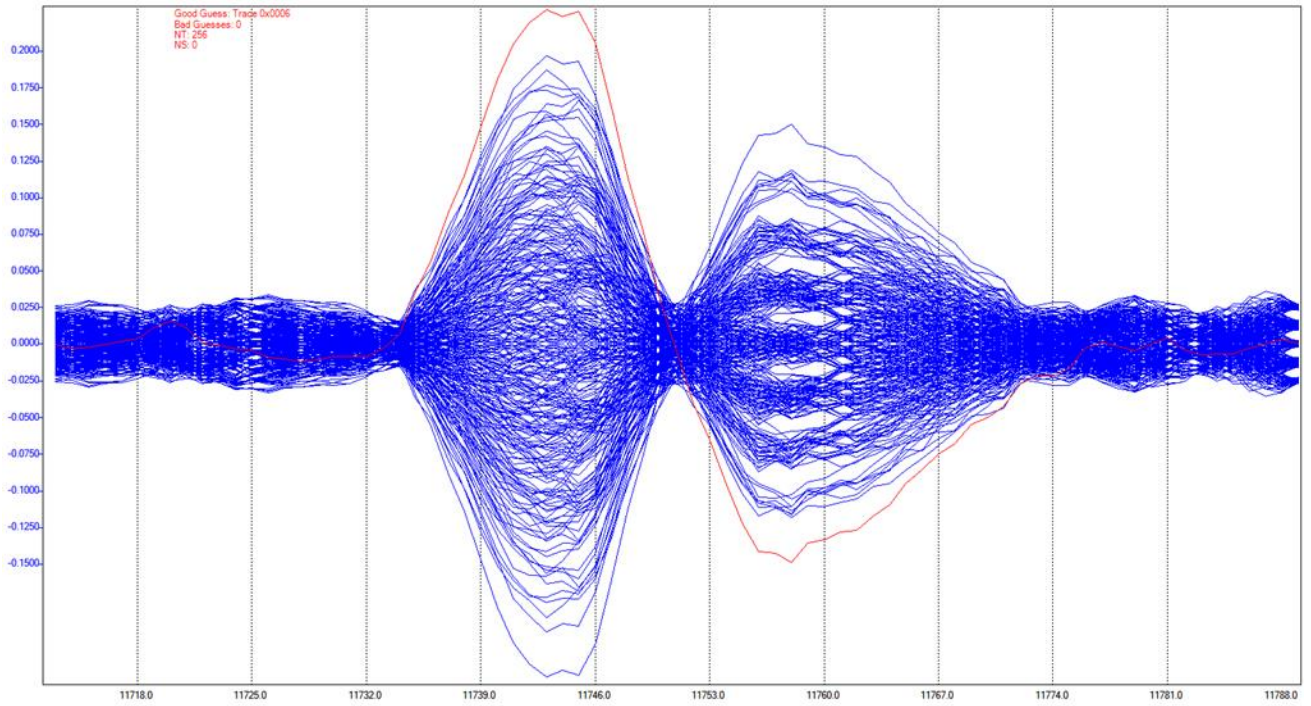
Side-Channel Attack on MAC Algo3

- Set P_1 to constant, i.e. 0
- Execute MA3
  - with random values for P_2: $P_{2,0}$ to $P_{2,n}$
  - Collect execution traces C_0 … C_n
- Perform CPA on P_2 xor C_1
  - C_1 = DES(0, K_1)
- Recover the value C_1
- Brute force C_1 and recover K_1
- Brute force MAC and recover K_2.



life.augmented

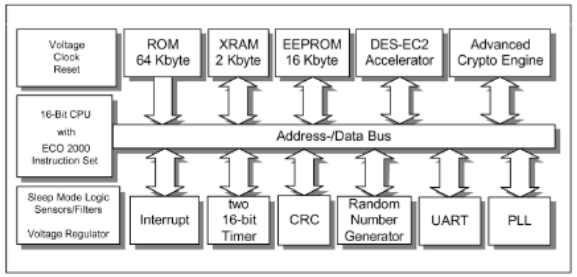
Attack Result



Today?
More and more side-channel attacks in real life.

Side-channel
Target any Kind of Integrated Circuit

Smartcards, Secure Elements, FPGA, SoC FPGA, complex SoC in mobiles, computers …



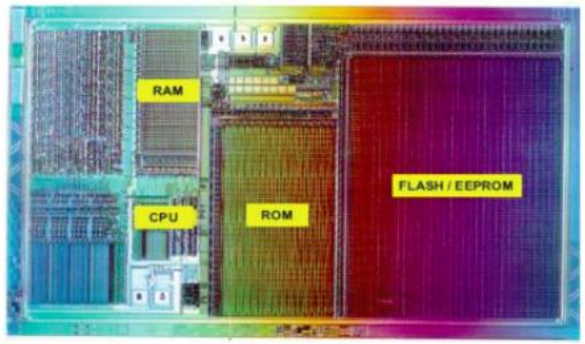
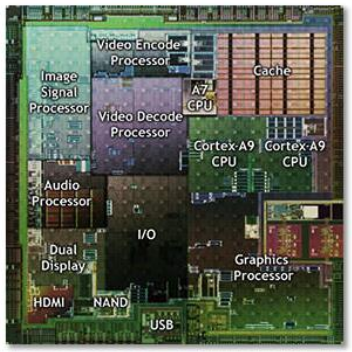
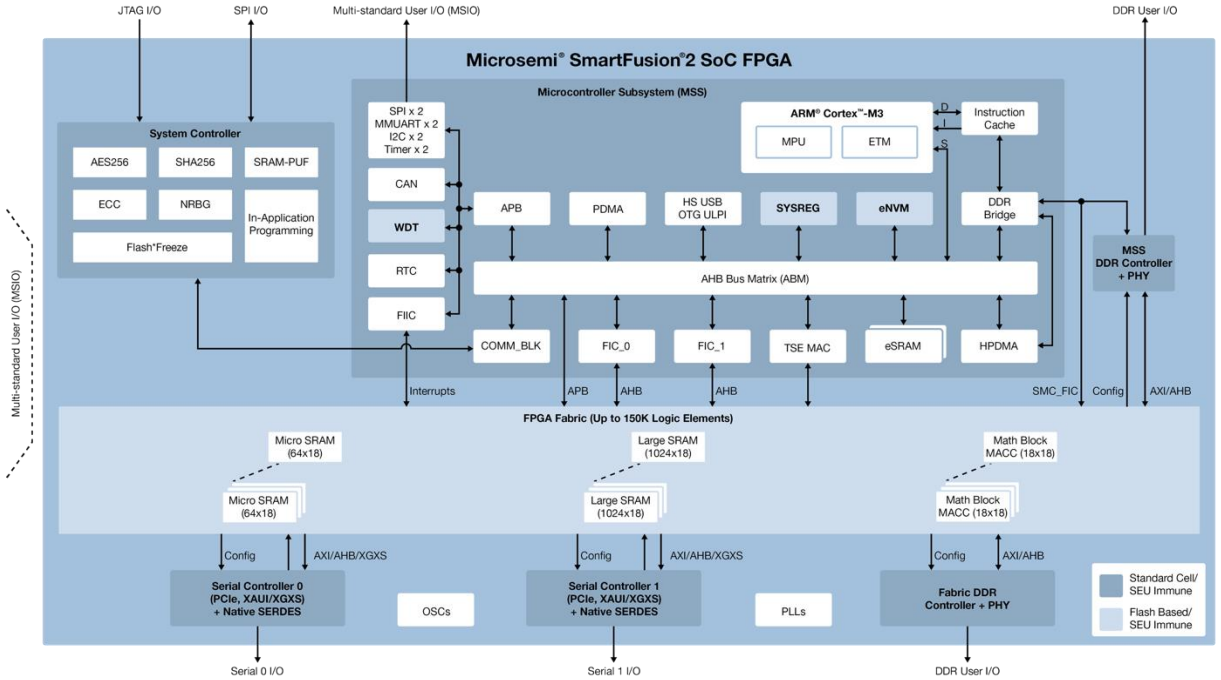
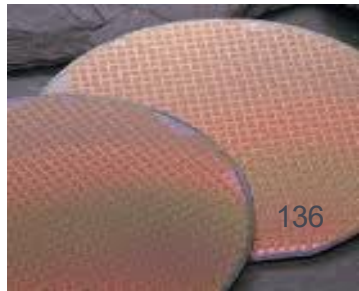


Figure : Smart card IC picture



Nvidia Tegra 2 SoC





life.augmented

Wide Range of Products Nowadays

- Smart cards: contact, contactless or dual interface
- RFID Tags
- Tokens, USB keys, SSD secure storage
- Set-top-Box
- FPGAs
- Systems on Chips (SoC)
- Mobile Phone
- Military Products

Attack categories

- Measurement: power, RF, EM
- Timings Attacks
- Simple Side-Channel Analysis
- Differential Side-Channel Analysis
- Advanced Side-Channel Attacks:
  - Horizontal,
  - High Order
- Profiled vs. Not Profiled Attacks
  - Template Attacks
  - Deep Learning

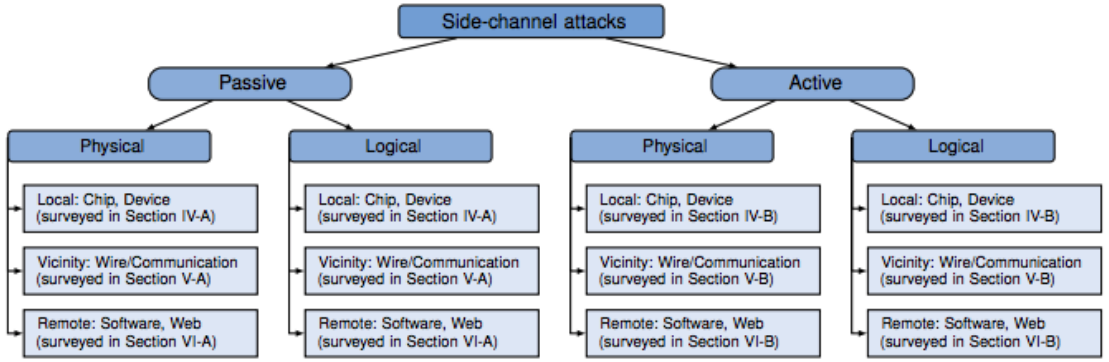


Fig. 6. Proposed classification system for side-channel attacks: (1) passive vs active, (2) physical properties vs logical properties, (3) local attackers vs vicinity attackers vs remote attackers.

Example of side-channel attack classification proposed in:

Systematic Classification of Side-Channel Attacks: A Case Study for Mobile Devices *Raphael Spreitzer, Veelasha Moonsamy, Thomas Korak, and Stefan Mangard*

life.augmented

Open Sources

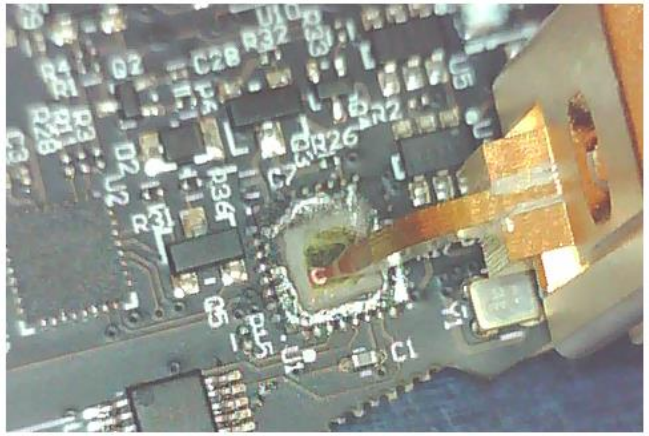
- Lascar https://donjon.ledger.com/about/
- Scared https://github.com/eshard/scared
- Side-Channel marvels: https://github.com/SideChannelMarvels
- REASSURE project https://reassure.eu
- SCALib https://github.com/simple-crypto/SCALib

- Languages:
  - From C, C++, Matlab, Maple
  - to Python, Numpy, Scipy, Jupyter Lab, Julia, Sage

life.augmented

A side journey to Titan

- https://ninjalab.io/a-side-journey-to-titan



A Side Journey to Titan

Side-Channel Attack on the Google Titan Security Key

(Revealing and Breaking NXP's P5x ECDSA Implementation on the Way)

Victor Lomne and Thomas Roche

NinjaLab

161 rue Ada, 34095 Montpellier, France

firstname@ninjalab.io

January 7, 2021

2024: EUCLEAK

ninjalab.io/news/

EUCLEAK

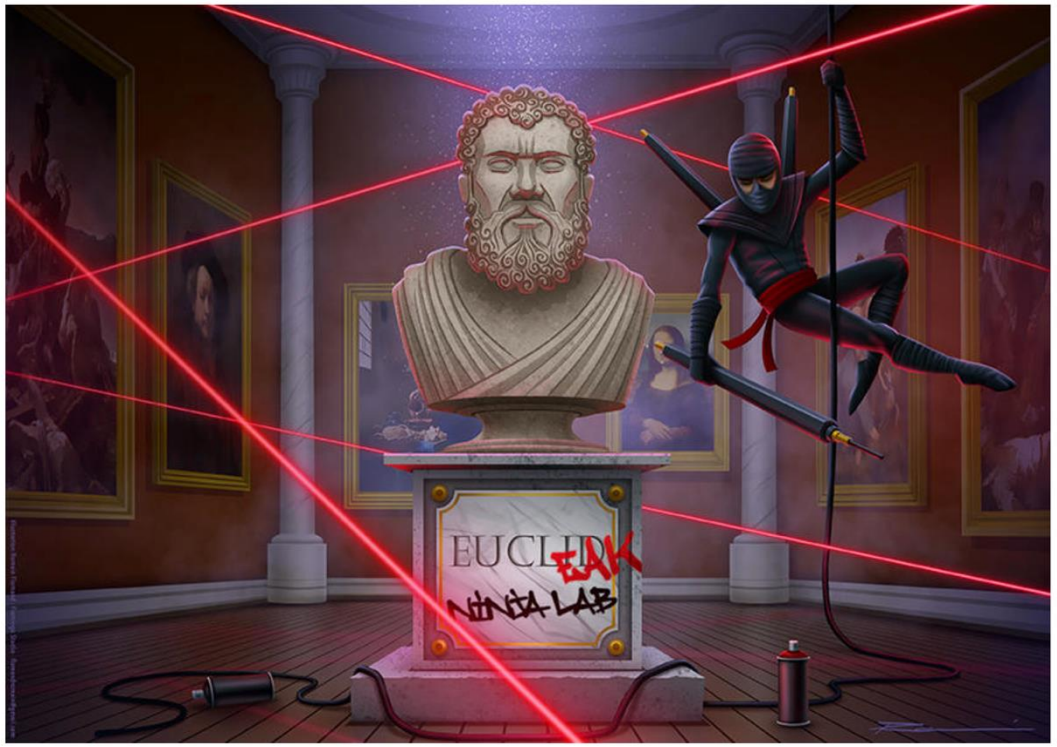


Illustration Romain Flamand – Flamingo Studio – flamandromain@gmail.com

We discovered a side-channel vulnerability in the `YubiKey 5Ci`. More precisely, we were able to extract the full long term ECDSA secret key linked to a FIDO account from the YubiKey. Furthermore our side-channel journey showed that the vulnerability applies to all `YubiKey 5 Series` and more generally to all Infineon security microcontrollers (including TPMs). Find more details and the full write-up of our work

ECDSA secret key recovery from partial scalar leakage

- From full scalar k recovery the secret key d can be recovered with:

$$d = (s \times k - h) \times r^{-1} \bmod n$$

- It might happen only few bits of *k* are recovered: *d* can not be recovered with this formula.
- It is still possible when several consecutive bits of the scalar are recovered for several ECDSA signatures to recover *d* using lattices reduction.
- See papers below.

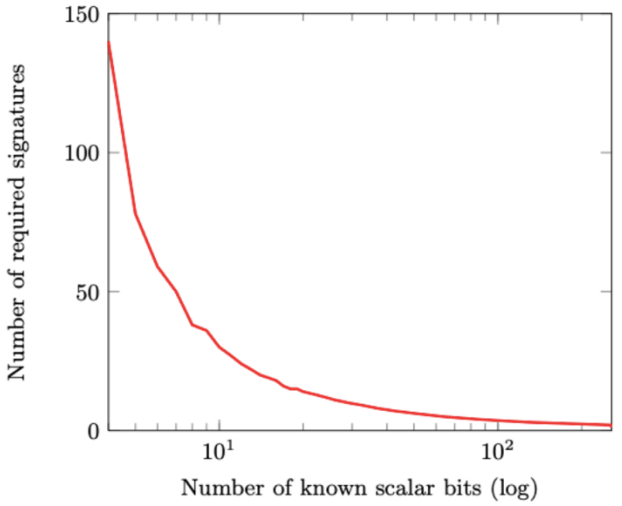


Fig. from [Poussier et al.]

[Howgrave-Graham and Smart, 2001] Howgrave-Graham, N. and Smart, N. P. (2001). Lattice attacks on digital signature schemes. *Des. Codes Cryptogr.*, 23(3):283–290.

[Nguyen and Shparlinski, 2003] Nguyen, P. Q. and Shparlinski, I. E. (2003). The insecurity of the elliptic curve digital signature algorithm with partially known nonces. *Des. Codes Cryptogr.*, 30(2):201–217.

[Medwed and Oswald, 2008] Medwed, M. and Oswald, E. (2008). Template attacks on ECDSA. In *Information Security Applications, 9th International Workshop, WISA 2008, Jeju Island, Korea, September 23-25, 2008, Revised Selected Papers*, pages 14–27.

[Poussier et al., 2017] Poussier, R., Zhou, Y., and Standaert, F. (2017). A sys- tematic approach to the side-channel analysis of ECC implementations with worst-case horizontal attacks. *IACR Cryptology ePrint Archive*, 2017:629.

SideLine attacks

- https://eprint.iacr.org/2020/1127.pdf

- REMOTE side channel attack on device !

- And also now remote fault attack on devices.

SideLine: How Delay-Lines (May) Leak Secrets from your SoC

Joseph Gravellier[1], *Jean-Max Dutertre*[1], *Yannick Teglia*[2] *and Philippe Loubet Moundi*[2]

[1]*Mines Saint-Etienne, CEA-Tech, Centre CMP, Gardanne, France*
{joseph.gravellier, dutertre}@emse.fr
[2]*Thales, La Ciotat, France*
{yannick.teglia, philippe.loubet-moundi}@thalesgroup.com

Post Quantum Cryptography

COMPUTING

IBM Raises the Bar with a 50-Qubit Quantum Computer

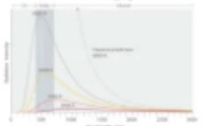
Researchers have built the most sophisticated quantum computer yet, signaling progress toward a powerful new way of processing information.

By Will Knight

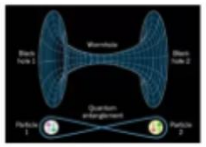
November 10, 2017

Theoretical Foundations

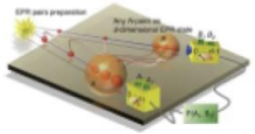
1900 — Planck's Quantum Hypothesis



1935 — The EPR Paradox



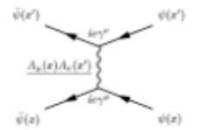
1964 — Bell's Inequality



1970 — Birth of Quantum Information Theory

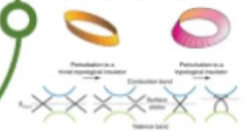
1980 — First Conference on Physics and Computation

1981 — Feynman's Quantum Computer Proposal



Emergence

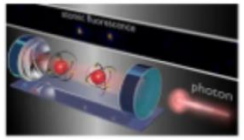
1982 — Discovery of Topological Quantum order



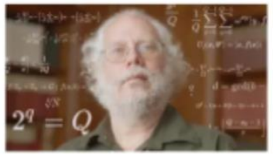
Development

1984 — Quantum Cryptography (BB84 Protocol) By IBM

1985 — Deutsch's Universal Quantum Computer



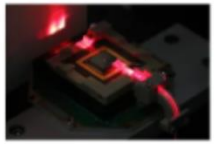
1994 — Shor's Algorithm



1994 — Grover's Algorithm

1996 — DiVincenzo Criteria For Quantum Computer

2000 — First Trap Ion Quantum Computer



Race

2004 — Circuit QED Demo.

Ongoing Advancements

2007 — The Transmon Superconducting Qubit

2007 — D-Wave One Quantum Annealer

D-Wave The Quantum Computing Company

2013 — Rigetti Computing

rigetti

2016 — Microsoft Station Q

Microsoft

2019 — Google Quantum Supremacy

Google

2020 — IBM Quantum Roadmap

IBM Q

2021 — Company Booming

IONQ Honeywell PsiQuantu XANADU

2022 — Quantumpedia's Founding

https://quantumpedia.uk/a-brief-history-of-quantum-computing-e0bbd05893d0

The Colourized Version of Solvay Conference 1927 (Source: Rare Historical Photos)

(Pre-Quantum) Cryptography : TODAY

Cryptography	Algorithms	Strength
Symmetric	AES-128 /AES-192 / AES-256 AES-GCM Chacha20 HMAC-SHA-256. Poly1305 SHA-2 / SHA-3 / Salsa20	2^{128} to 2^{256}
Asymmetric	RSA / DSA 2048-4096 ECDH / ECDSA P-256 / P-521 ECIES - EdDSA	2^{128} to 2^{256}

Schor's algorithm

- 1994 Schor designed quantum algorithm
- Capable to solve big integer factorization and discrete logarithm resolution mathematical problem in polynomial time compared to GNFS that is sub-exponential time.
- RSA, DH, DSA, ECDH, ECDSA security defeated when quantum computers with sufficient qubits allows Schor's algorithm execution.

Algorithms for Quantum Computation: Discrete Logarithms and Factoring

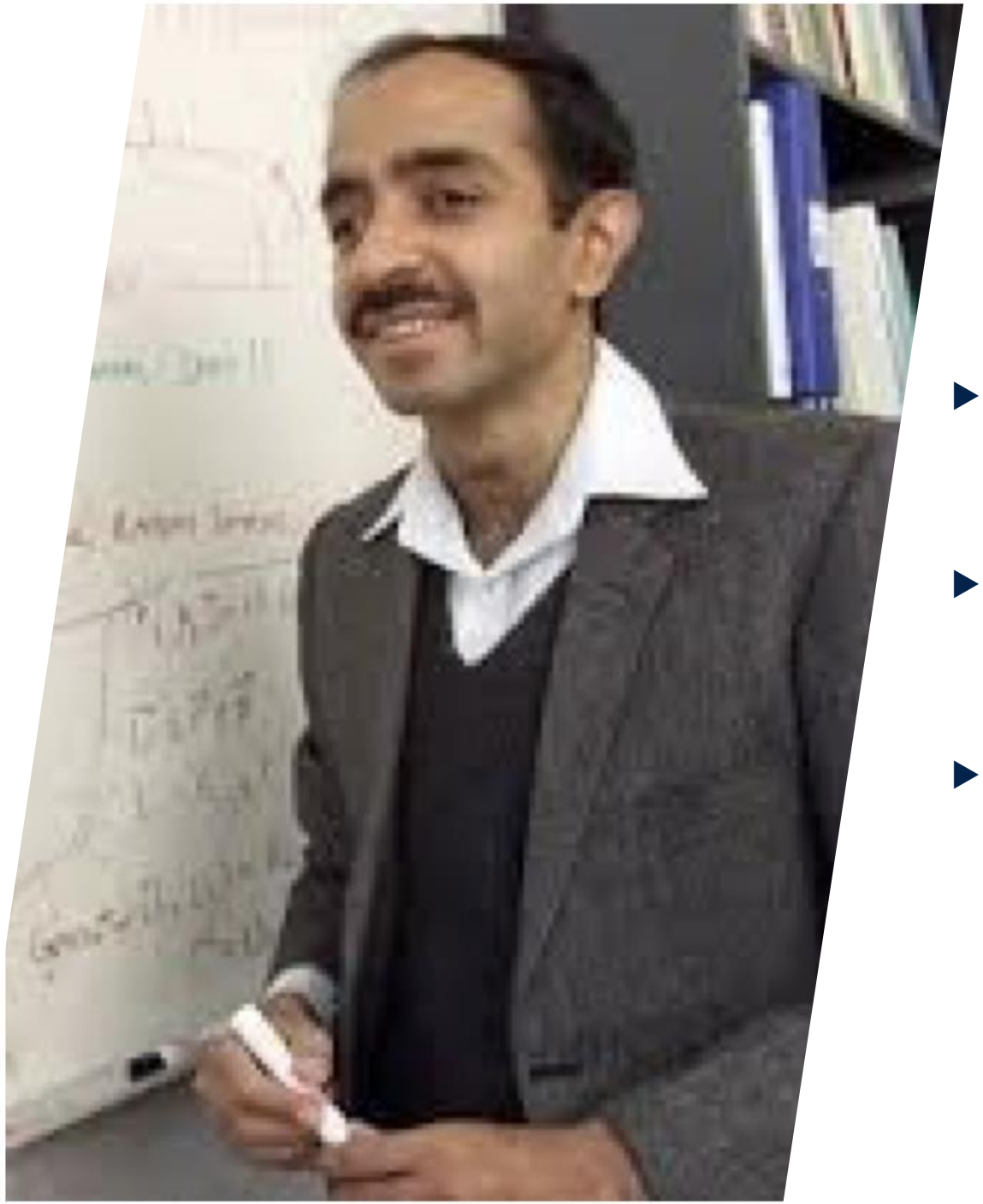
Peter W. Shor
AT&T Bell Labs
Room 2D-149
600 Mountain Ave.
Murray Hill, NJ 07974, USA

Abstract

A computer is generally considered to be a universal computational device; i.e., it is believed able to simulate any physical computational device with a cost in computation time of at most a polynomial factor. It is not clear whether this is still true when quantum mechanics is taken into consideration. Several researchers, starting with David Deutsch, have developed models for quantum mechanical computers and have investigated their compu-

[1, 2]. Although he did not ask whether quantum mechanics conferred extra power to computation, he did show that a Turing machine could be simulated by the reversible unitary evolution of a quantum process, which is a necessary prerequisite for quantum computation. Deutsch [9, 10] was the first to give an explicit model of quantum computation. He defined both quantum Turing machines and quantum circuits and investigated some of their properties.

The next part of this paper discusses how quantum com-



Grover's algorithm

- 1996 Grover designed quantum algorithm to speed up symmetric cryptography algorithm key recovery
- Offering a quadratic speedup compared to classical algorithms, which require approximately N steps.
- Grover's algorithm is less dramatic than that of Shor's algorithm, it still demonstrates the potential of quantum computing to outperform classical computing in specific tasks..

Quantum Computer is coming …
Quantum Computer is there?

MIT Technology Review

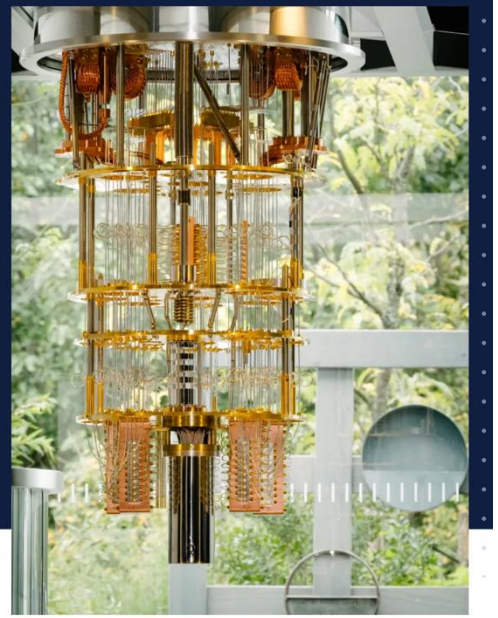
Featured Topics Newsletters Events Podcasts SIGN IN SUBSCRIBE

COMPUTING

IBM Raises the Bar with a 50-Qubit Quantum Computer

Researchers have built the most sophisticated quantum computer yet, signaling progress toward a powerful new way of processing information.

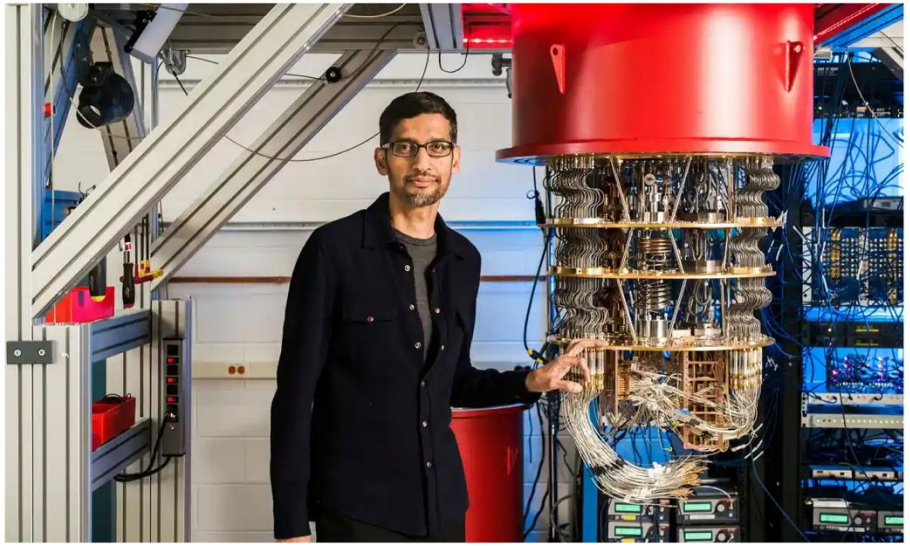
By Will Knight November 10, 2017



IBM established a landmark in computing Friday, announcing a quantum computer that handles 50 quantum bits, or qubits. The company is also making a 20-qubit system available through its cloud computing platform.

Google claims it has achieved 'quantum supremacy' – but IBM disagrees

Task that would take most powerful supercomputer 10,000 years 'completed by quantum machine in minutes'



Sundar Pichai, pictured with the Sycamore Quantum processor, compared the feat to building the first rocket to reach space. Photograph: Reuters

For Google, it was a historic announcement: a declaration that it had won the race to achieve "quantum supremacy" - the moment that a sophisticated quantum computer performed a task that stumped even the most powerful standard computer in the world.

life.augmented

Quantum Computer is coming …
Quantum Computer is there?

MIT Technology Review

Featured Topics Newsletters Events Podcasts SIGN IN SUBSCRIBE

COMPUTING

IBM Raises th Quantum Cor

Researchers have built t computer yet, signaling processing information.

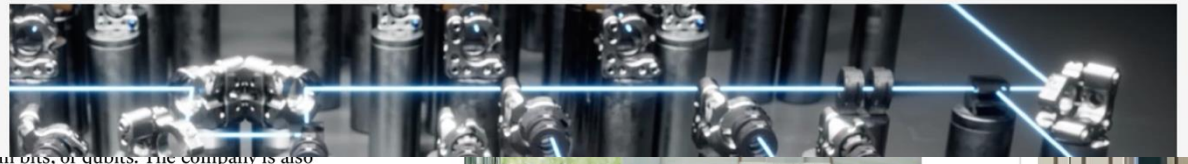
By Will Knight

IBM established a landmark computer that handles 50 quantum bits, or qubits. The company is also making a 20-qubit system available through its cloud computing platform.

China unveils faster light-based quantum computer

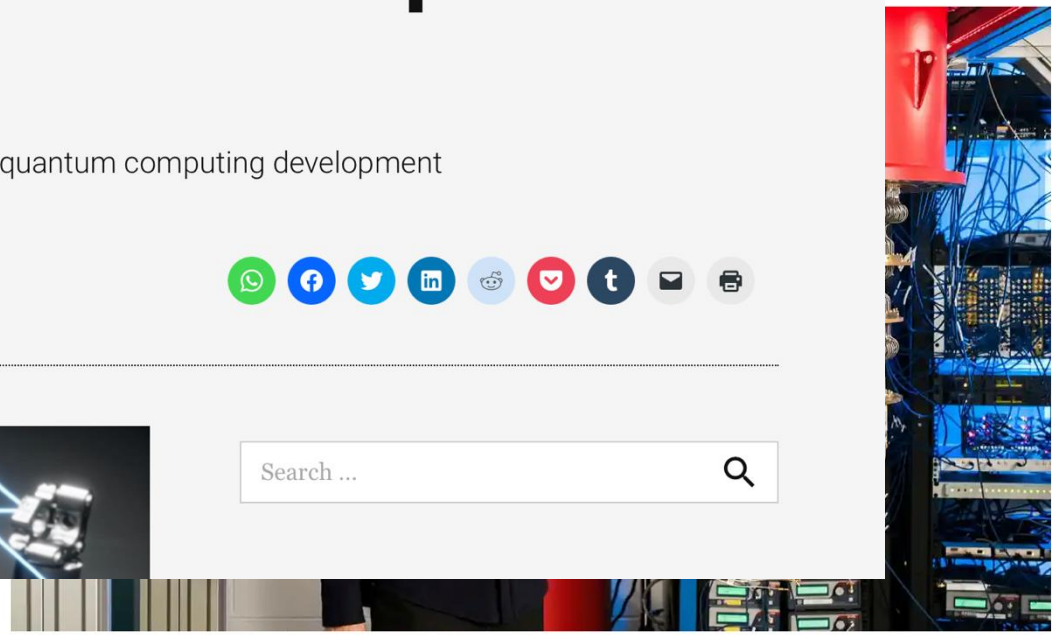
Critics say entry is keyed more to the record books than to long-haul universal quantum computing development

By JEFF PAO
OCTOBER 13, 2023



Search ...

Google claims it has achieved 'quantum supremacy' – but IBM disagrees



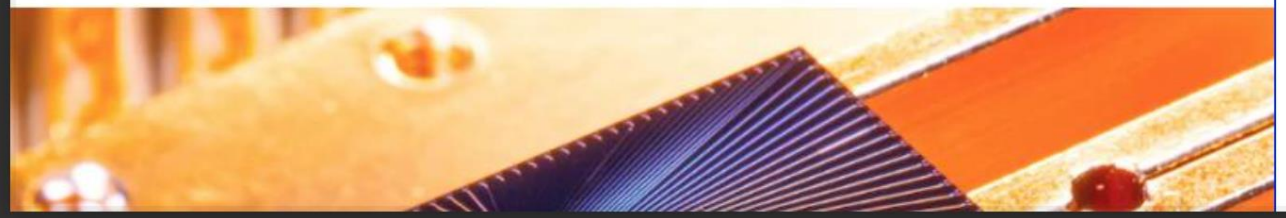
Sundar Pichai, pictured with the Sycamore Quantum processor, compared the feat to building the first rocket to reach space. Photograph: Reuters

For Google, it was a historic announcement: a declaration that it had won the race to achieve "quantum supremacy" - the moment that a sophisticated quantum computer performed a task that stumped even the most powerful standard computer in the world.

life.augmented

Premium

Quantum computing could end encryption within five years, says Google boss



Mr Pichai said a combination of artificial intelligence and quantum would "help us tackle some of the biggest problems we see", but said it was important encryption evolved to match this.

"In a five to ten year time frame, quantum computing will break encryption as we know it today."

This is because current encryption methods, by which information such as texts or passwords is turned into code to make it unreadable, rely upon the fact that classic computers would take billions of years to decipher that code.

Quantum computers, with their ability to be


?
CHAOS

(Pre-Quantum) Cryptography : TODAY

Cryptography	Algorithms	Strength
Symmetric	AES-128 /AES-192 / AES-256 AES-GCM Chacha20 HMAC-SHA-256. Poly1305 SHA-2 / SHA-3 / Salsa20	2^{128} to 2^{256}
Asymmetric	RSA / DSA 2048-4096 ECDH / ECDSA P-256 / P-521 ECIES - EdDSA	2^{128} to 2^{256}

Cryptography with Quantum Computers

Cryptography	Algorithms	Strength
Symmetric	~~AES-128 /AES-192 /~~ AES-256 AES-GCM Chacha20 HMAC-SHA-256. Poly1305 SHA-2 / SHA-3 / Salsa20	~~2^{128} to 2^{256}~~ **128**
Asymmetric	~~RSA / DSA 2048-4096~~ ~~ECDH / ECDSA P-256 / P-521~~ ~~ECIES - EdDSA~~	~~**2^{128} to 2^{256}**~~ **0**

life.augmented

Cryptography with Quantum Computers

Cryptography	Algorithms	Strength
Symmetric	~~AES-128 /AES-192 /~~ AES-256 AES-GCM Chacha20 HMAC-SHA-256. Poly1305 SHA-2 / SHA-3 / Salsa20	~~2^{128} to 2^{256}~~ **128**
Asymmetric	RSA / DSA 2048-4096 ECDH / ECDSA P-256 / P-521 ECIES - EdDSA	~~**2^{128} to 2^{256}**~~ **0**

NEED FOR NEW POST QUANTUM RESISTANT ALGORITHMS FOR ASYMMETRIC CRYPTOGRAPHY

life.augmented

Post Quantum Cryptography Timeline

- 1994: Shor's quantum algorithm. 1996: Grover's quantum algorithm.
  Many subsequent papers on quantum algorithms: see quantumalgorithmzoo.org.
- 2003: Daniel J. Bernstein introduces term Post-quantum cryptography.
- 2006: First International Workshop on Post-Quantum Cryptography. PQCrypto 2006, 2008, 2010, 2011, 2013, 2014, 2016, 2017, 2018, 2019, 2020, 2021, (soon) 2022.
- 2015: NIST hosts its first workshop on post-quantum cryptography.
- 2016: NIST announces a standardization project for post-quantum systems.
- 2017: Deadline for submissions to the NIST competition.
- 2017 – 69 candidates, lots of cryptanalysis, mostly on immature systems (13 broken).
- 2019: Second round of NIST competition begins.
- 2019 – 26 candidates, more cryptanalysis (2 broken).
- 2020: Third round of NIST competition begins.
- 2020 – Focus on 15 candidates; even two established systems crumble.
- ~~2021~~ 2022 ~~"not later than the end of March"~~ July NIST announces first selections.
- 2023/2024?: NIST issues post-quantum standards.

Post Quantum Cryptography Timeline

- 1994: Shor's quantum algorithm. 1996: Grover's quantum algorithm.
  Many subsequent papers on quantum algorithms: see quantumalgorithmzoo.org.
- 2003: Daniel J. Bernstein introduces term Post-quantum cryptography.
- 2006: First International Workshop on Post-Quantum Cryptography. PQCrypto 2006, 2008, 2010, 2011, 2013, 2014, 2016, 2017, 2018, 2019, 2020, 2021, (soon) 2022.
- 2015: NIST hosts its first workshop on post-quantum cryptography.
- 2016: NIST announces a standardization project for post-quantum systems.
- 2017: Deadline for submissions to the NIST competition.
- 2017 – 69 candidates, lots of cryptanalysis, mostly on immature systems (13 broken).
- 2019: Second round of NIST competition begins.
- 2019 – 26 candidates, more cryptanalysis (2 broken).
- 2020: Third round of NIST competition begins.
- 2020 – Focus on 15 candidates; even two established systems crumble.
- ~~2021~~ 2022 ~~"not later than the end of March"~~ July NIST announces first selections.
- 2023/2024?: NIST issues post-quantum standards.

Post Quantum Cryptography

Cryptography under the assumption that the attacker has a quantum computer.

Major categories:

- **Code-based** encryption: McEliece cryptosystem has survived since 1978. Short ciphertexts and large public keys. Security relies on hardness of decoding error-correcting codes.
- **Hash-based** signatures: very solid security and small public keys. Require only a secure hash function (hard to find second preimages).
- **Isogeny-based** encryption: new kid on the block, promising short keys and ciphertexts and non-interactive key exchange. Security relies on hardness of finding isogenies between elliptic curves over finite fields.
- **Lattice-based** encryption and signatures: possibility for balanced sizes. Security relies on hardness of finding short vectors in some (typically special) lattice.
- **Multivariate-quadratic** signatures: short signatures and large public keys. Security relies on hardness of solving systems of multivariate equations over finite fields.

Warning: These are categories of mathematical problems;
individual systems may be totally insecure if the problem is not used correctly.

Post Quantum Cryptographic Selected Algorithms

The winners:

- Kyber, a public-key encryption system based on structured lattices
- Dilithium, a public-key signature scheme based on structured lattices
- Falcon, a public-key signature scheme based on structured lattices
- SPHINCS+, a public-key signature scheme based on hash functions

Schemes advancing to round 4, so maybe more winners later:

- BIKE, a public-key encryption system based on codes
- Classic McEliece, a public-key encryption system based on codes
- HQC, a public-key encryption system based on codes

https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022

2022: "Although this new post-quantum toolbox may seem handy for developers, the maturity level of the post-quantum algorithms presented to the NIST process should not be overestimated. Many aspects lack cryptanalytical hindsight or are still research topics, e.g. analysis of the difficulty of the underlying problem in the classical and quantum computation models, dimensioning, integration of schemes in protocols and more importantly the design of secure implementations. This situation will probably last some time after the publication of NIST standards. **Acknowledging the immaturity of PQC is important: ANSSI will not endorse any direct drop-in replacement of currently used algorithms in the short/medium term.** However, this immaturity should not serve as an argument for postponing the first deployments." (emphasis added)

- Industry is performing R&D to implement and secur
  - Hardware coprocessor for NTT
- **Challenge on hardware resources**: **RAM – execution time – KEY STORAGE**
- **Crystals-Dilithium (Signature)**
  - Module-LWE
  - NTT easily used

NIST Level	2	3	5
Public Key (bytes)	1213	1952	2592
Secret Key (bytes)	2288	3664	4400
Signature (bytes)	2420	3293	4595

- **COMPARE with**

life.augmented

- 512 bytes for key length and signature size with RSA 4096

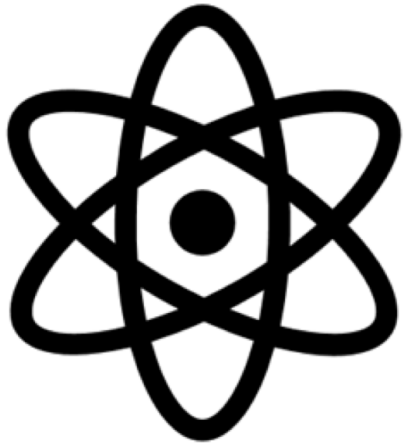
Conclusion on PQC

Post-Quantum Cryptography (PQC) = Cryptography that is **not** vulnerable to attacks by quantum algorithms when bigger quantum computers are built.

Symmetric cryptography (AES, SHA2, SHA3) is not vulnerable. There is no need for Quantum RNGs. Most of the current RISC-V Crypto Extensions (Zk*) are fine.

RSA (factoring) and Elliptic Curve (ECDL) cryptography has to go. Quantum algorithms (Shor's and Regev's) break these very efficiently -- polynomial time, increasing key size not much help.

After 7 years of analysis, newer PQC / Quantum-Secure (lattice-based, code-based, hash-based, ..) signatures and key establishment standards are being rolled out into applications.



Attack Potential and JIL Rating

Joint Interpretation Library

Attack Factors

- For each step, the evaluator considers 6 factors according to the CEM:
  - Amount of time
  - Level of expertise
  - Level of knowledge of the TOE
  - Level of opportunity
  - Equipment required
  - Open Sample required or not

Factors	Identification	Exploitation
Elapse time		
Expertise		
Knowledge of the TOE		
Access to the TOE		
Open Sample		
Equipment		
Sub-total		
TOTAL		

- For each factor a value is provided in a table
- The attack potential is the sum of all values

life.augmented

Elapsed Time

- Two steps are considered: preparation and realization…

	Identification	Exploitation
< one hour	0	0
< one day	1	3
< one week	2	4
< one month	3	6
> one month	5	8
Not practical	*	*

Table 1: Rating for Elapsed Time

Expertise definition

- Expertise also depends on the tools required to perform the attack: Optical Microscope, Chemistry (etching, grinding), Micro-probing station, Laser Cutter, Radiation, Plasma (etching, grinding), Focused Ion Beam (FIB), Scanning Electron Microscope, (SEM), Scanning Force Microscope (SFM)

	Definition according to CEM	**Detailed definition to be used in smartcard evaluations**
a) Experts	Familiar with implemented • Algorithms • Protocols • Hardware structures • Principles and concepts of security	Familiar with • Developers knowledge namely algorithms, protocols, hardware structures, principles and concepts of security and • Techniques and tools for the definition of new attacks
b) Proficient	Familiar with • security behaviour	Familiar with • security behaviour, classical attacks
c) Laymen	No particular expertise	No particular expertise

Table 2: Definition of Expertise

Extent of expertise **(in order of spread of equipment or smartcard related knowledge)**	
Equipment: The level of expertise depends on the degree to which tools require experience to drive them • Oscilloscope • Optical Microscope • Chemistry (etching, grinding), Microprober • Laser Cutter, Radiation • Plasma (etching, grinding), Focused Ion Beam (FIB) • Scanning Electron Microscope (SEM) • Atomic Force Microscope (AFM)	**Knowledge:** The level of expertise depends on knowledge of • Common Product information • Common Algorithms, Protocols • Common Cryptography • Differential Power Analysis (DPA), Differential Fault Analysis (DFA), Electromagnetic Analysis (D/EMA) • Reverse Engineering • Smartcard specific hardware structures • Principles and concepts of security • Developers knowledge

Table 3: Extent of expertise

Expertise Rating

- If several experts are required for the same domain, the highest expertise level is considered for each step
- The level “Multiple Experts” is designed for applicable when several experts of different domains are required (e.g. hardware and crypto)

	Identification	Exploitation
Layman	0	0
Proficient	2	2
Expert	5	4
Multiple Expert	7	6

Table 4: Rating for Expertise

Knowledge of the TOE

- The knowledge of the TOE can be gained through unclassified or unprotected information, or collusion with a developer.
- The following classification is to be used:
  - Public: information in the public domain,
  - Restricted: information from development (specifications, guidance)
  - Sensitive: HLD and LLD information.
  - Critical: Implementation representation (Design and Source Code).
  - Very Critical: Data requiring also heavy specific tools
- Information discovered by Reverse Engineering will be rated as Id step…

	Identification	Exploitation
Public	0	0
Restricted<	2	2
Sensitive	4	3
Critical	6	5
Very critical hardware design	9	na

Table 5: Rating for Knowledge of TOE

life.augmented

Access to the TOE

- Availability of samples (in terms of time and cost) and number of samples needed to succeed
- Reasons to need more than one sample:
  - the attack succeeds only with some probability,
  - The attacker needs to collect information from several copies of the TOE.

	Identification	Exploitation
< 10 samples	0	0
< 30 samples	1	2
< 100 samples	2	4
> 100 samples	3	6
Not practical	*	*

Table 6: Rating for Access to TOE

Equipment

- Equipment category considers price and availability:
  - Standard
  - Specialized (expensive equipment found in universities)
  - Bespoke: very expensive or difficult to keep confidential
  - Multiple Bespoke: several different bespoke equipment
- A value is given for all equipment used in IC business but this must always be reevaluated with technical progress…

	Identification	Exploitation
None	0	0
Standard	1	2
Specialized (1)	3	4
Bespoke	5	6
Multiple Bespoke	7	8

Table 9: Rating for Equipment

Access to Open Samples

- When the evaluator attack uses « open samples » or classified information, this « advantage » is considered in the rating…
- This Open Sample rating sometimes replaces the discovery step, and may give a lower result

(Identification phase only)	PUBLIC or not required	RESTRICTED	SENSITIVE	CRITICAL
Open Samples	0	2	4	6
Samples with known secret	0	2	4	6

Rating the Attack

Range of values	TOE resistant to attackers with attack potential of:
0-15	No rating
16-20	Basic
21-24	Enhanced-Basic
25-30	Moderate
31 and above	High

Table 11: Rating of vulnerabilites for CC v3

- Sum the values for each factor, in preparation and exploitation, for each successful attack
- Compute the rating for both paths with / without access to open samples, and keep the lowest
- The lowest rating gives the resistance of the TOE

Final Table

Factors	Identification	Exploitation
Elapsed time		
< one hour	0	0
< one day	1	3
< one week	2	4
< one month	3	6
> one month	5	8
Not practical	*	*
Expertise		
Layman	0	0
Proficient	2	2
Expert	5	4
Multiple Expert	7	6
Knowledge of the TOE		
Public	0	0
Restricted	2	2
Sensitive	4	3
Critical	6	5
Very critical hardware design	9	NA
Access to TOE		
< 10 samples	0	0
< 30 samples	1	2
< 100 samples	2	4
> 100 samples	3	6
Not practical	*	*
Equipment		
None	0	0
Standard	1	2
Specialized (1)	3	4
Bespoke	5	6
Multiple Bespoke	7	8
Open samples (rated according to access to open samples)		
Public	0	NA
Restricted	2	NA
Sensitive	4	NA
Critical	6	NA

Table 10: Final table for the rating factors

life.augmented

Rating
Example with a CEMA on AES

- AES in a secure element
- CEMA successful.

Factors	Identification	Exploitation
Elapse time	2 (< 1 week)	3 (< 1 day)
Expertise	5 (expert)	2 (Proficient)
Knowledge of the TOE	2 (restricted)	0 (public)
Access to the TOE	0 (<10 samples)	0 (<10 samples)
Open Sample	0	0
Equipment	3 (specialized)	4 (specialized)
Sub-total	12	9
TOTAL	21	

- The product is resistant to an ***Enhanced Basic*** *potential attack*:VAN.3
- In that case certification is a FAIL as AVA VAN.5 was the mandatory requirement.

life.augmented

Questions ?

life.augmented


THANKS

Our technology starts with You

Find out more at www.st.com

life.augmented