Introduction to physical attacks against embedded devices

Hélène Le Bouder et Ronan Lashermes

TAF Cyber et IoT
UE sécurité de l'IoT et des systèmes embarqués

2023

helene.le-bouder@imt-atlantique.fr

Safety VS Security

Safety

protects against unintentional accidents as:

- human mistake,
- perturbations,
- failures.

Security

protects against malicious behaviour as:

- spy,
- tampering,
- destruction,
- usurpation (identity fraud).

Cryptology

The **cryptology** (science of secrets) regroups the **cryptography** and the **cryptanalysis**.

Modern cryptography:

- **confidentiality**: limits data access to authorized persons;
- **integrity**: ensures that the information cannot be altered;
- **authentication**: validates the origin of some data;
- **non-repudiation**: allows a person to take part in a contract without the possibility to denounce it later.

AES and Galois field $\mathbb{F}_{2^8}$

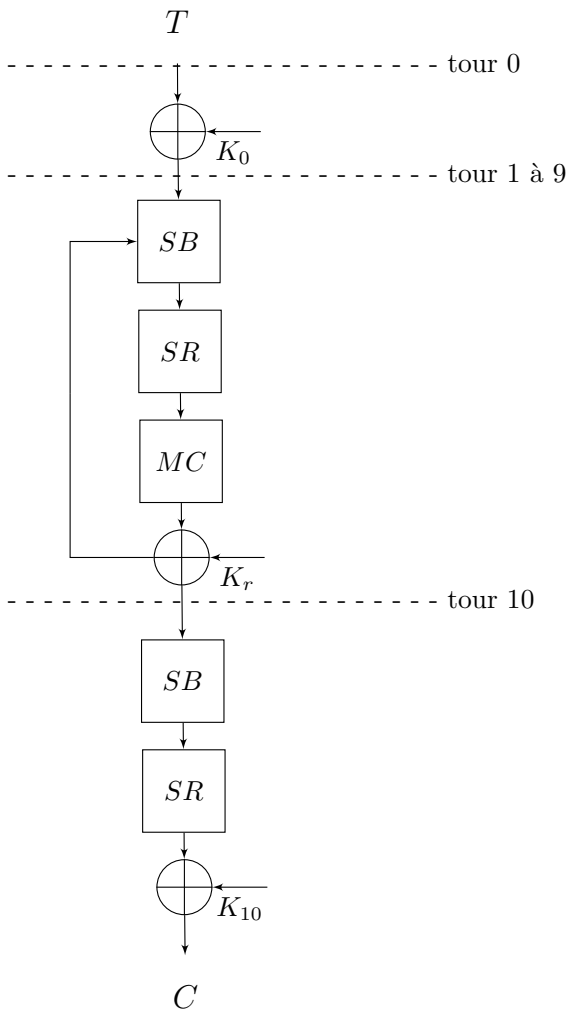
- AES(Advanced Encryption Standard).
- AES is the current block cipher standard.
- The AES work on the Galois field $\mathbb{F}_{2^8}$ with:

$$P(X) = X^8 + X^4 + X^3 + X + 1.$$

as irreducible polynomial .
The intern laws are: $\oplus$ and $\otimes$

AES [1]

- Mapping the plaintext M of 128 bits into an array of $4 \cdot 4 = 16$ bytes, called the State.
- Exists in different key size versions: 128, 192 or 256 bits.
- Substitution Permutation Network.
- The round function is composed of:
  - SubBytes: non-linear transformations, working independently on individual bytes of the State.
  - ShiftRows: a byte-shifting operation on each row of the State.
  - MixColumns: a linear matrix multiplication on $GF(2^8)$, applied on each column.
  - AddRoundKey: $\oplus$ a xor with the round-key.



SubBytes AES

$$SB(68) = 45$$

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

illustration : http://freestudy9.com/aes-transformation-function/

MixColumns AES

- The **MixColumn** is a linear transformation in $GF(2^8)$ to the column of the state.
- $\otimes$ between a column vector of the state and a matrix.

$$\begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix}$$

Figure: Matrix of **MixColumn** *MC*

$$1 \otimes a = a \quad .$$

$$2 \otimes a = \begin{cases} 2 \cdot a, & \text{if } a < 80. \\ FF\&((2 \cdot a) \oplus 1B), & \text{else.} \end{cases}$$

$$3 \otimes a = 2 \otimes a \oplus a \quad .$$

AES: Key generation

- The **master key** K is the round key 0, K_0.
- K_{r+1} is computed with the previous round key K_r,
- according the function **KeyExpansion** :

$$\begin{cases} K_{r+1}^{l,0} = SB\left(K_r^{(l+1) \bmod 4,3}\right) \oplus rcon(l,r) \oplus K_r^{l,0} & \forall l \in [\![0,3]\!] \\ K_{r+1}^{l,c} = K_r^{l,c} \oplus K_{r+1}^{l,c-1} & \forall l \in [\![0,3]\!] \text{ and } c \in [\![1,3]\!] \end{cases} \quad (1)$$

with SB the SubBytes function and $rcon$ a constant matrix of size 4×10.

A good link to summarize :
https://www.youtube.com/watch?v=H2LlHOw_ANg

Enigme

illustration: Pixabay

Different abstraction levels

1. Algorithm: the theory.
2. Programming language (C, java, python, etc.).
3. Architecture and micro-architecture.
4. Logic device.
5. Transistors.

Physical attacks are at level: logic device / transistors.

illustration: pixabay

Physical attacks

- Even if an encryption algorithm is proved secure mathematically, its implementation can open the gate to physical attacks.
- They exploit the fact that some physical states of a device depend on intermediate values of the computation.
- Two families two main types of physical attach
  1. **side channel analysis** (SCA)
  2. **fault injections attacks** (FIA)

Targets

the main target of physical attacks are small devices because of the below features

- small devices
- embedded device
- IoT
- physical access

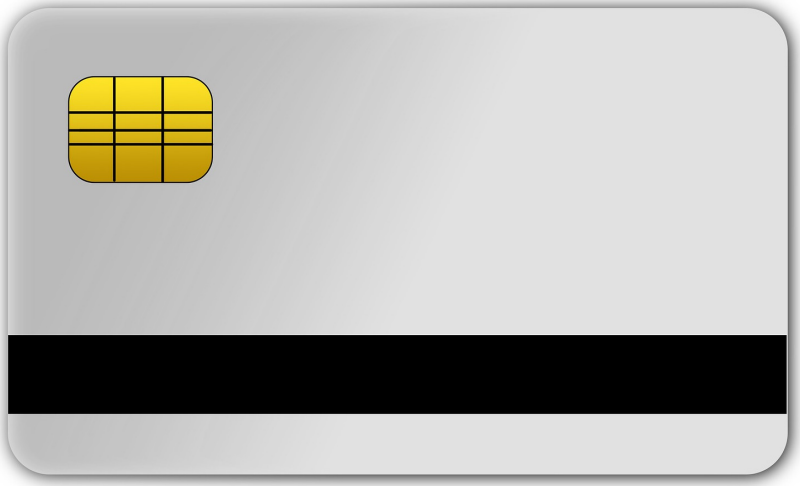
Embedded device features of embedded devices

1. specific function, Process limitation
2. space constraints, memory limitation
3. energy constraints. Battery limitation

Targets

Example: smart card

1. years 70-80.
2. dads: Kunitaka Arimura, Roland Moreno, Helmut Gröttrup and Jürgen Dethloff



Smart carts is a great target for physical attacks

illustration: Pixabay

Use cases

we attack the physical device to find passwords, pins, … in order to …

Find secret

- Cryptanalysis.
- Find a code PIN.

In order to

- steal personnal data,
- control a system,
- misuse identity
- privilege elevation ...

Use cases

It's a use case of physical attacks

Reverse engineering

The activity of studying an device to determine its inner workings.

- industrial spy.

Leakage analysis

Someone put something hidden somewhere

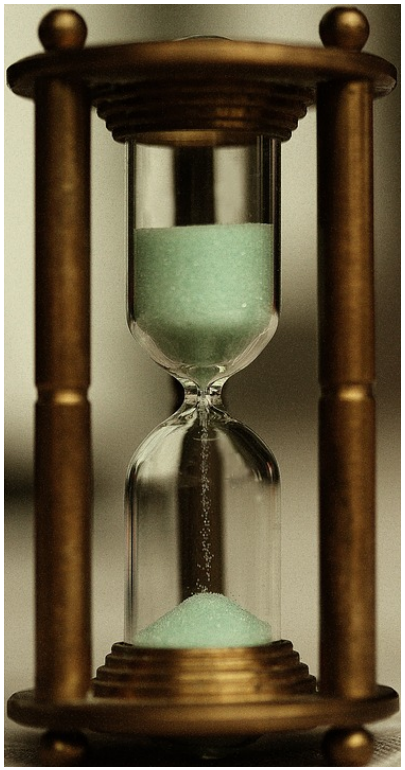
- SCA exploit the fact that some physical states of a device depend on intermediate values of the computation, called leakage.

Examples of physical leakage

- Timing [2]. more calculation needs more time
- Temperature [3].
- Power consumption [4].
- Electromagnetic radiations [5].
- Photons emission [6].

Timing

the first attack was a timing attack using Timing



Time as a leakage

- First leakage exploited.
- **Idea**: measure the execution time.
- Example: during a conditional branching.
- Exmaple of target: PIN code comparison.

illustrations: Pixabay

Code PIN

Code is for auth with crypto (with signature)

authentication

combines 2 of the 3 elements

- something we have, smart card
- something we know, the code pin
- something we are. the biometry

code PIN (Personnal Identifiant Number)

- confidential code
- to authenticate user of a smart card.
- smart card = login (we have)
- code PIN = password (we know)

Exercise

1. How many 4-digit PIN codes are there? 10^4 = 10000
2. If we test digit by digit, how many possibilities are there? 4*10 = 40
3. Find the weakness of this code PIN comparison algorithm.

pin.c

```
1 bool compare_arrays(const uint8_t* a, const uint8_t*
  ↪  b, size_t len) {
2     for(size_t i = 0; i < len; i++) {
3         if (a[i] != b[i]) {
4             return false;
5         }
6     }
7     return true;
8 }
```

4. Suggest a new version.

Correction

- How many 4-digit PIN codes are there?
  10^4

Correction

- How many 4-digit PIN codes are there?
  10^4
- If we test digit by digit, how many possibilities are there?
  $10 \cdot 4$

Correction

- How many 4-digit PIN codes are there?
  10^4
- If we test digit by digit, how many possibilities are there?
  $10 \cdot 4$
- Find the weakness of this code PIN comparison algorithm.
  The comparison is not in constant time.

Correction

- How many 4-digit PIN codes are there?
  10^4
- If we test digit by digit, how many possibilities are there?
  $10 \cdot 4$
- Find the weakness of this code PIN comparison algorithm.
  The comparison is not in constant time.
- Suggest a new version.

Correction

Fake good idea

pin.c

```
1  bool compare_arrays(const uint8_t* a, const uint8_t* b,
   ↪  size_t len) {
2      uint8_t result = true;
3      uint8_t  fake = true;
4      for(size_t i = 0; i < len; i++) {
5          if(a[i] != b[i]){
6              result = false;
7          }
8          else {
9              fake = false;
10         }
11     }
12     return result;
13 }
```

Correction

Correction for the exercise.

pin.c

```
1 bool compare_arrays(const uint8_t* a, const uint8_t* b,
  ↪  size_t len) {
2     uint8_t result = 0;
3     for(size_t i = 0; i < len; i++) {
4         result |= a[i] ^ b[i];
5     }
6     return result == 0;
7 }
```

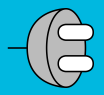
we can also write this line as:
result = a[i] = b[j]

Correction

Real good correction

pin.c

```
1  bool compare_arrays(const uint8_t* a, const uint8_t* b,
   ↪  size_t len) {
2      uint8_t encoded_candidate[HMAC_SHA256_DIGEST_SIZE];
3      hmac_sha256(encoded_candidate, a, len, secret_key,
   ↪  KEY_LEN);
4
5      uint8_t result = 0;
6      for(size_t i = 0; i < HMAC_SHA256_DIGEST_SIZE; i++) {
7          result |= encoded_candidate[i] ^ b[i];
8      }
9      return result == 0;
10 }
```

Power consumption 

That's what we do in March

Principe

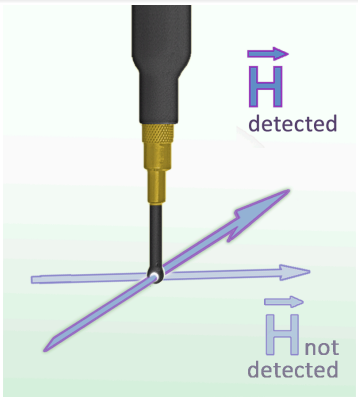
- Connection on the device.
- An integrated device is made up of a set of logic gates.
- The instantaneous device power consumption is the sum of the power consumptions of each of its logic gates.
- Possibility to distinguish a transition from 0 to 1 from a transition from 1 to 0.

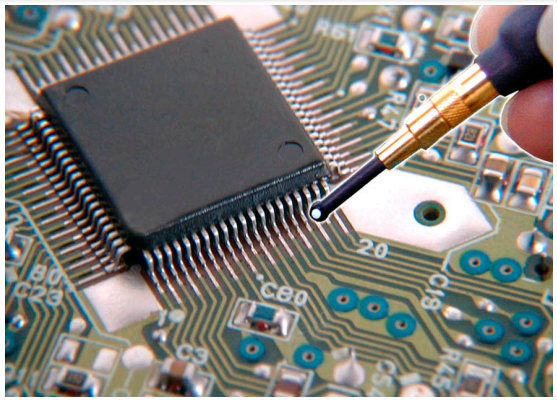
I can use a voltmeter to measure the electro magnetics

Electromagnetic radiations

Principe

- The variation of the power consumption is accompanied by a variation of the magnetic field.
- An EM probe is used for the measurements.
- Advantage: the device cannot detect that it is being observed.
- Weakness: less precise measurements.





illustrations: Langer EMV Tecknik `https://www.langer-emv.com/en/index`

Electromagnetic radiations

EMA, side channel bench of laboratory high security (LHS) d'INRIA Rennes.

Simple power analysis (SPA) [7, 8]

Idea

determine directly, from an observation of the power consumption, during a normal execution of an algorithm, information on the calculation performed or the data manipulated.

Example: Algorithm Square and Multiply

$$x^e \mod n$$

```
1:  procedure SQUARE AND MULTIPLY(x,e,n)
2:      r = 1
3:      b = (e)_2
4:      for i bit de poids fort au bit de poids faible do
5:          if b[i] = 1 then
6:              r = r^2 · x mod n
7:          else
8:              r = r^2 mod n
9:          end if
10:     end for
11:     return r
12: end procedure
```

Example: Algorithm Square and Multiply

long is for square
short is for mulitpication

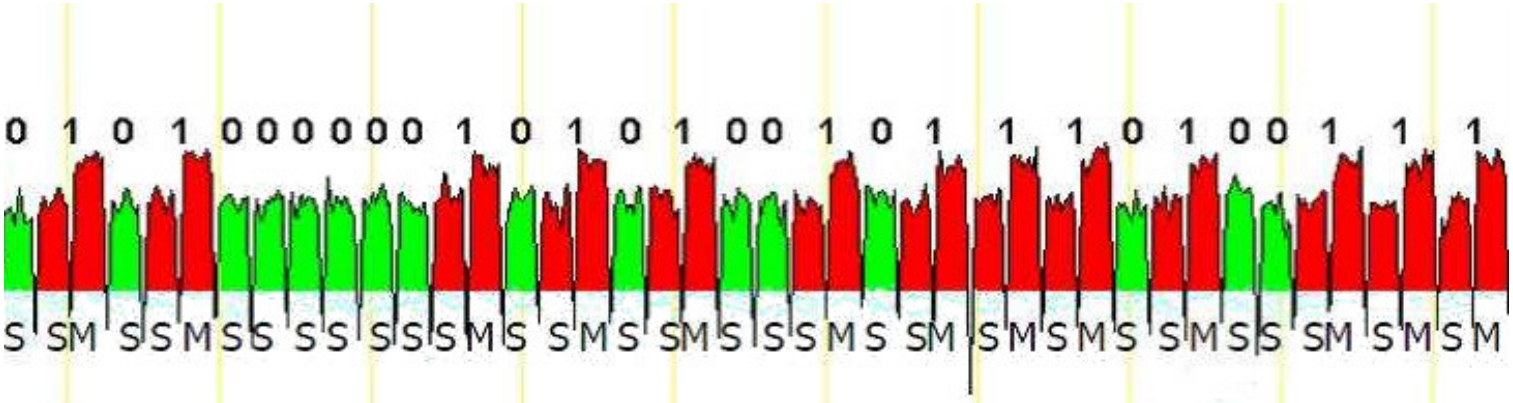


illustration from Cryptography Research, Inc.

Attacks by characterization or template attacks [9]

Advantages and weakness

- Most effective attacks.
- Need a device of the target

profiling or learning phase,

you need 2 devices one for attacking one for computation

- Characterization of the profiling device.
- The attacker can choose and change secrets.
- Learning significant leakage and noise characteristics.

attack or inference phase

- Get traces on the targeted device
- Confront the target traces with profiling traces.

Étape 1: Campaign, the target

what is the target? the secret key of an AES
but first round key

- Starting to define precisely the target $\mathcal{K} = \hat{k}$.
- Example: first round key of AES.
- **Divide and conquer**.
  Split the target in small pieces.
  Idea: make guess on the different part of the secret.
- Example: one **byte** AES round key (256 possible values).

we attack byte by byte

illustration: pixabay

Step 1: Campaign, the observables

Acquisitions de données et mesures

- A **campaign** collects data and measurements, called **observables**
- Examples:power consumption, cipher text, ...

illustration: pixabay

Step 1: Campaign, attack path

- The **attack path** $\mathcal{R}$ is a relation between observables (O_S, O_R) and the target $\mathcal{K} = \hat{k}$.
  $\mathcal{R}(\hat{k}, O_S) = O_R$
  It is composed by:
  - algorithm functions,
  - **physical function(s)**.

Physical function

- It represent a physical behaviour.
- It doesn't has a mathematical expression.
- It is called**leakage function** in SCA.

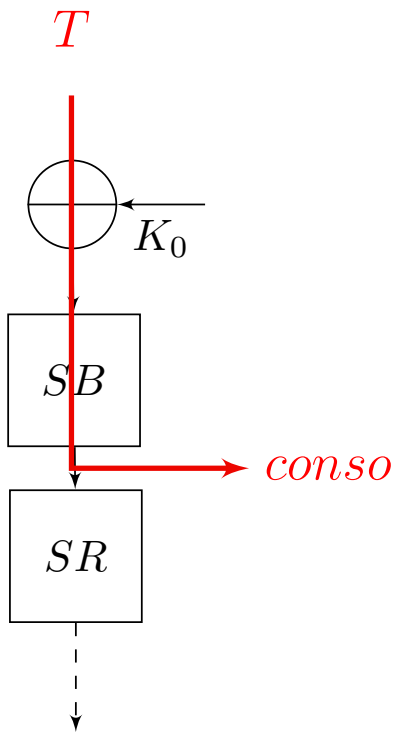
In my attack path I have my secret with my function of algo and phy func

Example (see in TP)

Example on AES: The correlation power analysis (CPA) [10])

- $\mathcal{K} = K_0^{l,c}$, a byte of the first round AES key.
- $O_S = T$, plain text
- $O_R = pow$, power consumption
- $\mathcal{R}(T, K) = f(SB(T \oplus K_0)) = pow$;
- with f the leakage function power consumption.

this is the physical function



Step 2: Predictions, model

Definition

A **model** is a mathematical function allowing to approach a physical function.

Examples

- *HW*, Hamming weight
- *HD*, Hamming distance

Step 2: Predictions, guesses

Theoretical attack path

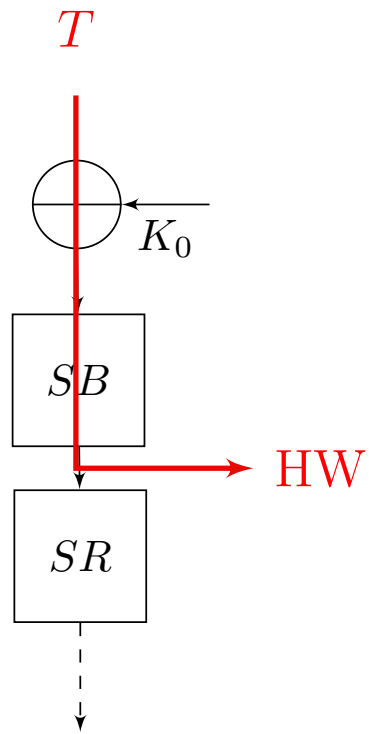
- The second step consist in build **predictions**.
- Choosing one or sevralmodels m to replace the physical function(s) in the attack path.
- The new relation is the theoretical attack path.
- With a divide and conquer approach, it is possible to compute a prediction $P_{m,k}$
  - for each guess k,
  - and each observable O_S,
- $\mathcal{R}_m(k, O_s) = P_{m,k}$.

Example (see in TP)

Example on AES: The correlation power analysis (CPA) [10])

- Hamming weight replaces the physical function f.
- For each first round AES key byte:

$$\begin{aligned} \mathcal{R}_m(k, O_s) &= \mathcal{R}_{HW}(k, T) \\ &= HW(SB(T \oplus k)) \\ &= P_{HW,k} \quad . \end{aligned}$$



Step 3: Confrontation

distinguisher

- To confront the prediction $P_{m,k}$ with observables O_R, a statistic tool is used **distinguisher**;
- It brings out one guess k_d.
- If $\hat{k} = k_d \Rightarrow$, the attack is a success.
- The distinguisher

Examples of distinguisher

sieve [11], counter [12], difference of mean [13], correlation [10], entropy [14], mutual information [15], principal component [16],linear discriminant [17]...

Example (see in TP)

Pearson Correlation

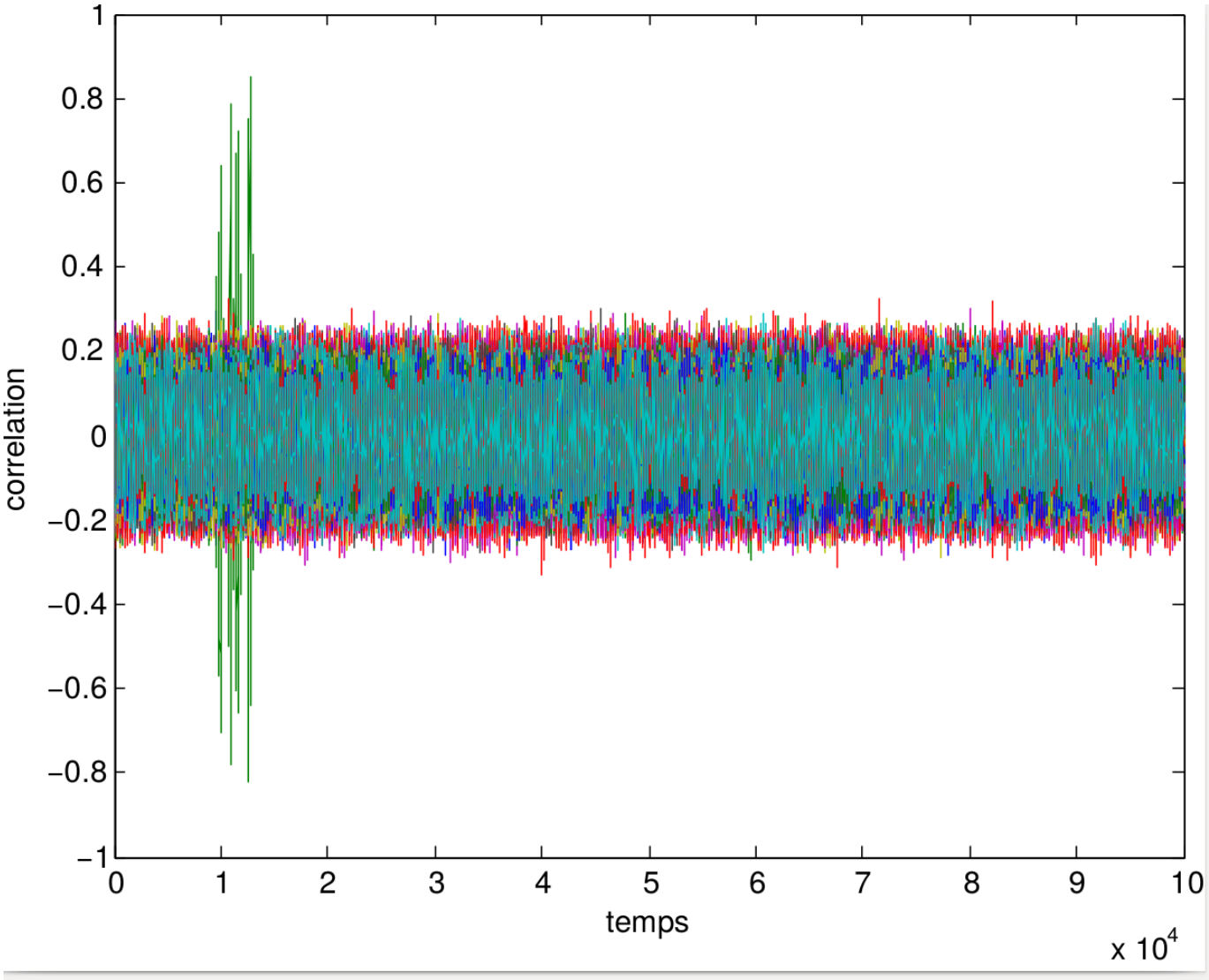
- The Pearson correlation between tow variables $O_R = X$ and $P_{m,k} = Y$ is defined as:

$$\mathrm{Cor}(X, Y) = \frac{\mathrm{Cov}(X, Y)}{\sqrt{\mathrm{Var}(X) \cdot \mathrm{Var}(Y)}} \quad .$$

Correlation Power Analysis [10]

- In our example the prediction matrix $P_{m,k}$ is correlated with the matrix of traces O_R.
- One curve is obtained for each guess.
- The returned guess k_d is the one with the maximum pic value.

Example (see in TP)



Example (see in TP)

Results in the curves

- A curve = one guess key byte.
- 256 curves.
- Here the correct key is associated to the green trace.
- The **points of interest** are the moment when the secret is manipulated.
- Pertinence of the model.
  Here, the correlation is 0.8, it is a good model.

Noise and desynchronisation

- Adding factice instructions/operations.
- Using analogue noise generators.
- Using a variable period clock.

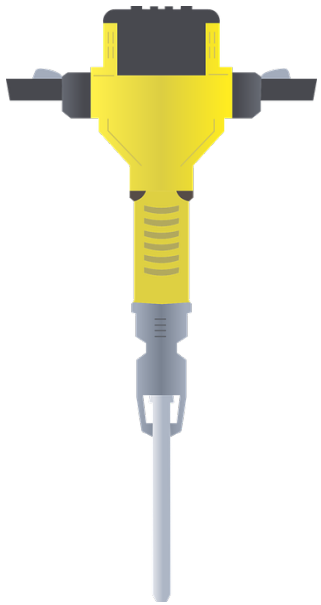
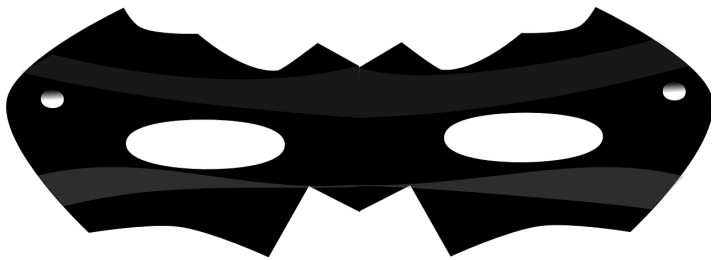


illustration: Pixabay

Mask

Let $f : \mathbb{F}_{2^m} \Rightarrow \mathbb{F}_{2^l}$ be a function which needs protection.
If it cannot possible to avoid an information leakage on x during $f(x)$ computation:

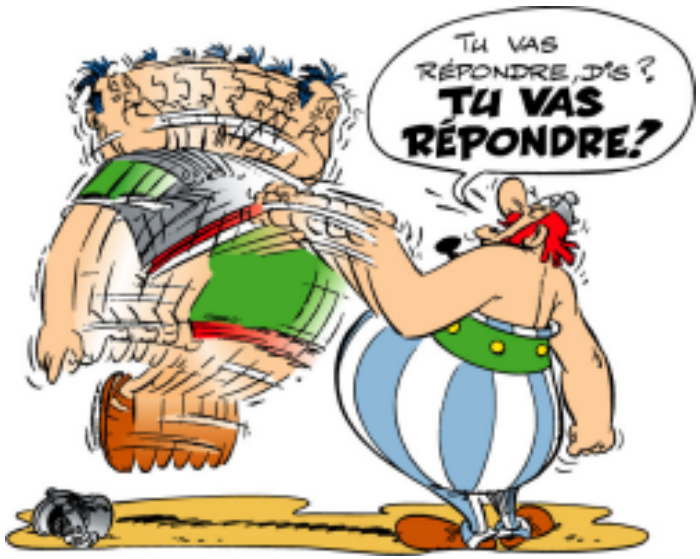
- if f is **linear**,
  we can use a random mask m:
  $f(x) = f(x \oplus m) \oplus f(m)$.
- if f is **not linear**.
  we can generates a function f_m such as:
  $f_m(x \oplus m) = f(x) \oplus m$.



Origin

illustrations : Pixabay

Physical fault



The **fault injection attacks** (FIA) disrupt the normal operation of a device, with the aim of obtaining information or hijacking it.

Examples of way to inject physical faults

- Laser fault injection [18].
- Electromagnetic fault injection [19].
- Clock glitch [20].
- Power glitch [21].
- ...

illustrations: Pixabay

Laser fault injection

Laser

- The energy of the light radiation is absorbed by the silicon of the circuit.
- Pairs electron hole are then created along the light beam.
- They can lead to the appearance of a photoelectric current at the level of a transistor.
- A voltage peak is propagated in combinatorial logic blocks.

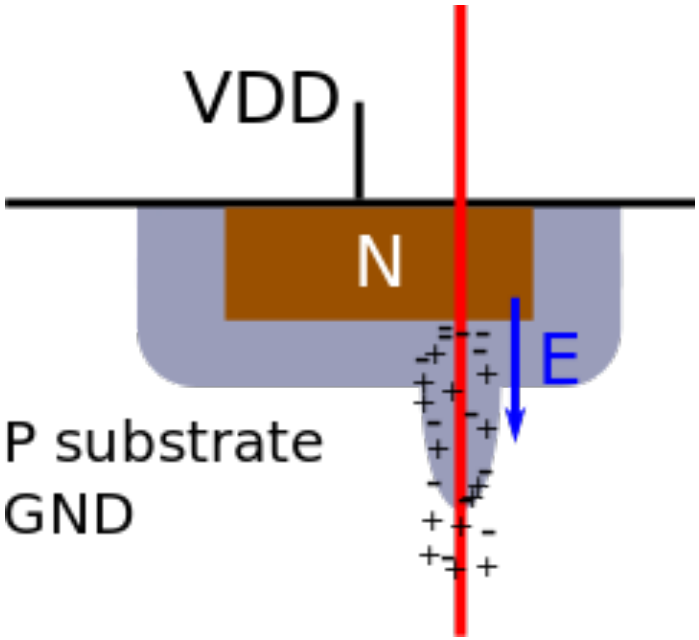
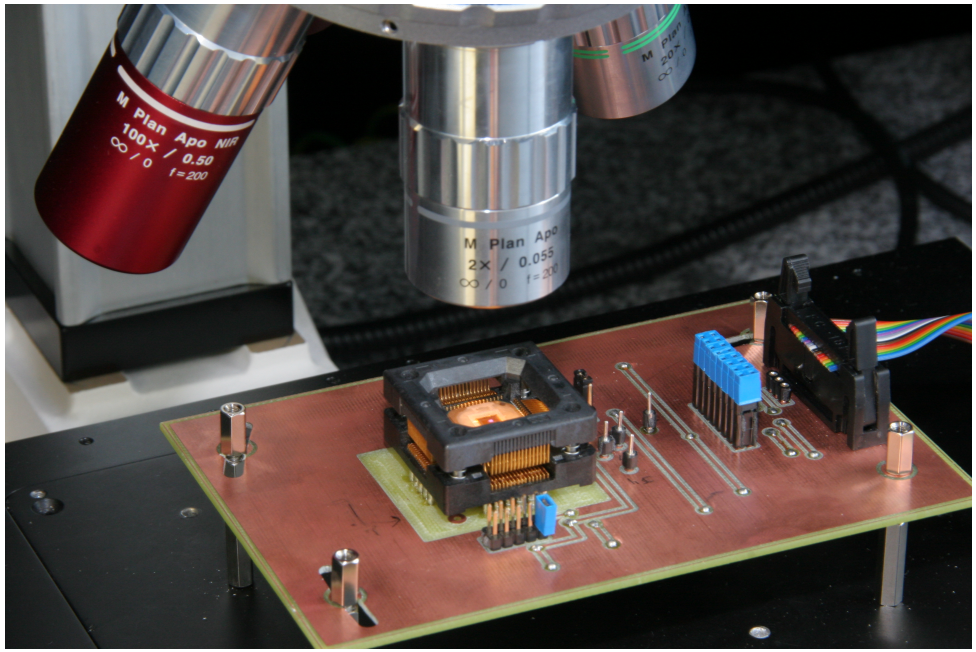


illustration: PhD of Roscian [18]

Laser fault injection



Micropacs laser bench, at "centre de Microélectronique de Provence", of "L'École des Mines de Saint-Étienne"

the most expensive device for testing

MINES Saint-Étienne

Micro-PackS CIM PACA

Electromagnetic fault injection

unlike laser, where you could inject the attack where exactly you wanted, EM is not that exact, but way less expensive

EM

- The attacked circuit is placed under a near-field antenna.
- A very strong current to flow for a very short time.
- By electromagnetic coupling, part of this energy is transmitted in the metal at the surface of the chip.
- The generated power in the chip can therefore generate errors.

Electromagnetic fault injection

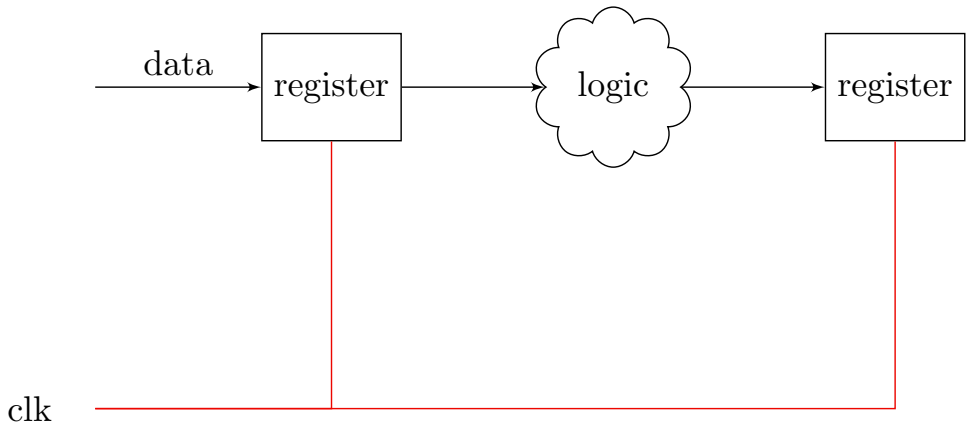
Faustine, of laboratory high security of INRIA Rennes [22].

Clock glitch

that's the attack that we are going to imply

Internal clock

- A synchronous circuit is composed by logic doors and registers managed by a common **internal clock** denoted `clk`.
- At each rising edge of the clock, data are updated in the registers.

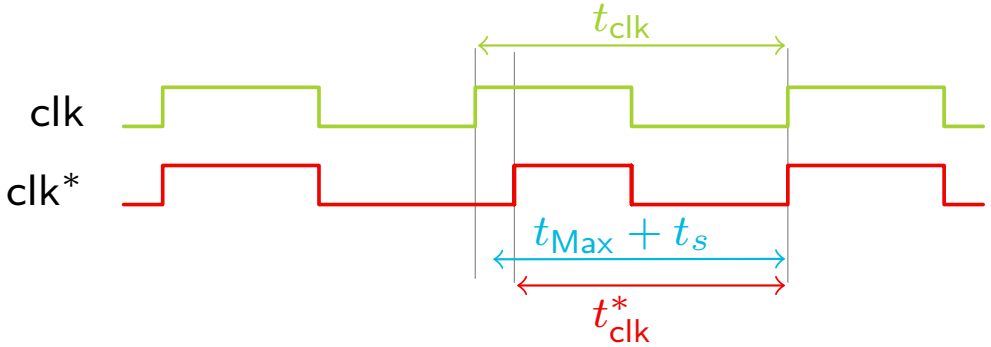


Clock glitch

Internal clock

- t_{clk}: clock period.
- t_{Max}: the time at which the last change in value of the data at the input of the registers takes place before being stable.
- t_s : the time during the data must remain stable in registers to be sampled correctly.

$$t_{clk} > t_{Max} + t_s$$



Clock glitch

- In TP we will use clock glitch.
- We will xor a signal with the internal clock.

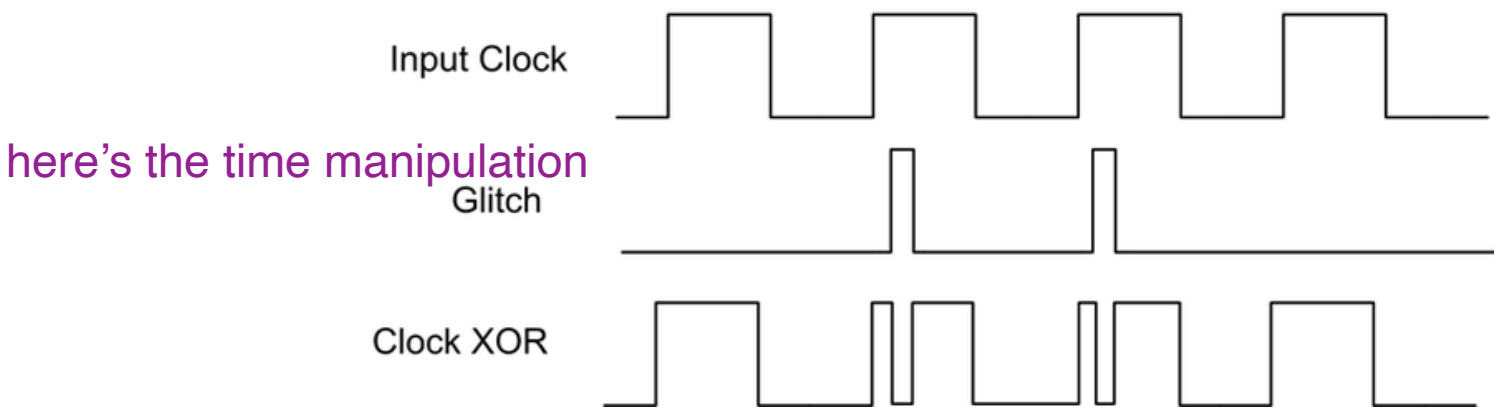


illustration: wiki ChipWhisperer [23]

Effects of Attack

- **No influence.**
- **Modifications of data.**
  - **Bonding**: at the binary or assembler level, this is the setting to 0 or 1 of some bits in the memory or in a register.
  - **Bit-flip**: this pattern represents a change in the assignment value of a variable or an intermediate value.
- **Modification of the instructions.**
  - **Instruction jump**: the instruction is not executed.

In fault injection we can't change the program

  - **Instruction replacement**: the instruction is replaced by another.
    Example: to modify a conditional test before a connection
- And sometimes...

Effects

Destroy the device !

illustration: Pixabay

DFA by Piret-Quisquater [24]

Fault model:

- 1 faulty byte in one input column of MixColumns.
- The model of the error is the function $\oplus e$.
- $4 \cdot 255 = 1020$ possible faults (4 bytes positions, 255 non-zero values).

for 4 byte we have
256-1 = 255
1 is for all zero

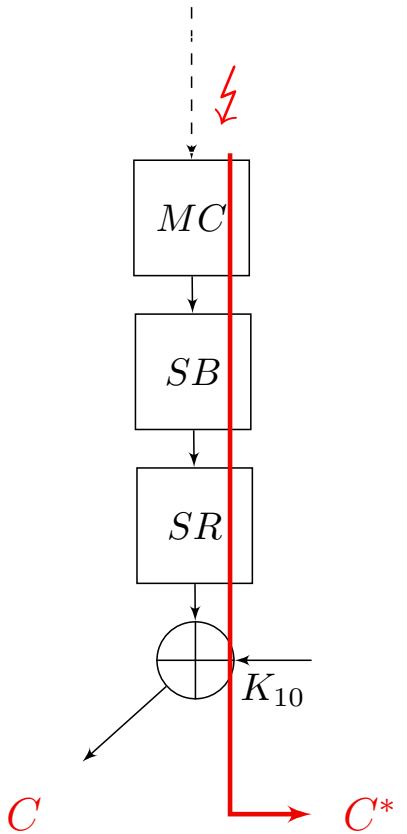
DFA by Piret-Quisquater [24]

Propagation of the fault

- 1020 possible faults at output of MixColumns.
- 1020 possible faults at output of final AddRoundKey.
- We can retrieve a part of the key with doing guesses on the 4 bytes key according to the faulty column.

e = error

$$\begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} e \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 2e \\ e \\ e \\ 3e \end{pmatrix}$$



DFA by Piret-Quisquater [24]

The attacker has the cipher text and the faulty text (which is 4 bytes here) C, $C*$
If error is in II we keep that
otherwise we eliminate it.

Sieve

1. For each guess $\hat{k}$ (among the 2^{32} possibilities) of the K_{10} bytes $0, 7, 10$ and 13.
2. Compute:

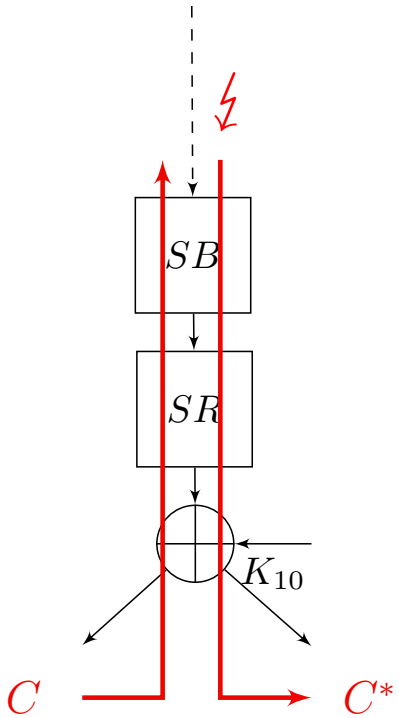
$$\Delta = \left(SB^{-1} \circ SR^{-1}\left(K_{10} \oplus C\right)\right) \oplus \left(SB^{-1} \circ SR^{-1}\left(K_{10} \oplus C^{*}\right)\right)$$

3. If Δ is equal to one of the 1020 fault possibilities, keep the guess.
4. While different guesses are possibles, start gain with an other pair(C, C^{*}).

DFA non uniform error value analysis (NUEVA) [14]

Fault model and campaign

- The fault has a statistical bias.
- Fault at the input of the last SubBytes.
- Repeat the process to have n pairs of (C, C^*).

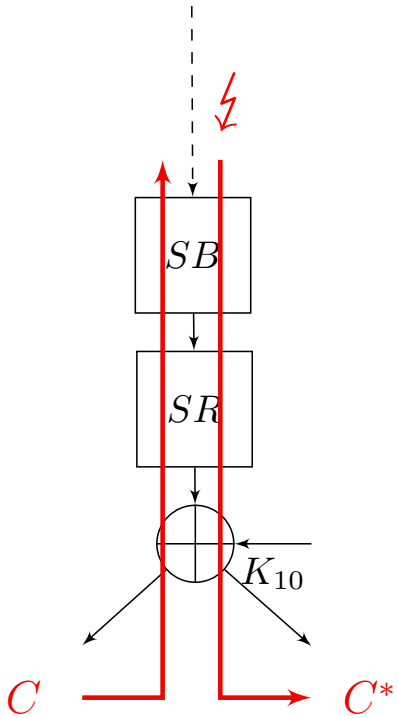


DFA non uniform error value analysis (NUEVA) [14]

Predictions:

- Divide and conquer approach byte per byte.
- For each guess $\hat{k}$ on a faulty byte $K_{10}[i]$ and each pair (C, C^*), compute the error. The error model is a $\oplus e$

$$e = \left(Sbox^{-1}(\hat{k} \oplus C)\right) \oplus \left(Sbox^{-1}(\hat{k} \oplus C^*)\right)$$



DFA non uniform error value analysis (NUEVA) [14]

n / $\hat{k}$	0	1	$\cdots$	255
0	$e_{0,0}$	$e_{0,1}$	$\cdots$	$e_{0,255}$
1	$e_{1,0}$	$e_{1,1}$	$\cdots$	$e_{1,255}$
2	$e_{2,0}$	$e_{2,1}$	$\cdots$	$e_{2,255}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-1$	$e_{n-1,0}$	$e_{n-1,1}$	$\cdots$	$e_{n-1,255}$

DFA non uniform error value analysis (NUEVA) [14]

Distinguisher

- In the table with n lines and 256 columns:
- Only one column corresponds to the real effect of our fault.
- It is the correct byte key guess.
- For a wrong guess, by property of confusion in the of SubBytes function, the distribution is random.
- Compute the **Shannon entropy** of each distribution.

$$H(X) = -\sum_{x \in \mathcal{X}} P(X = x) \cdot \log_2(P(X = x)) \quad .$$

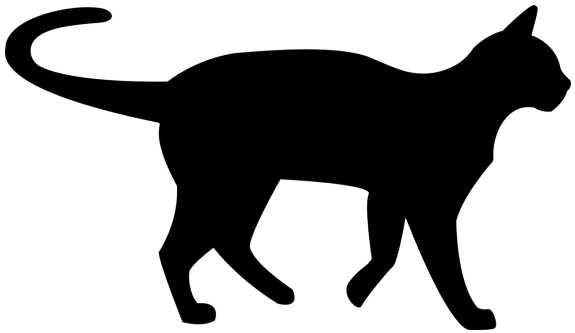
- Return the guess with smallest entropy.
- If the attack success it is the correct guess.

Vérification du code

Redundancy

- spatial,
- temporal,
- detectors/correctors codes.

Comparison between the results.



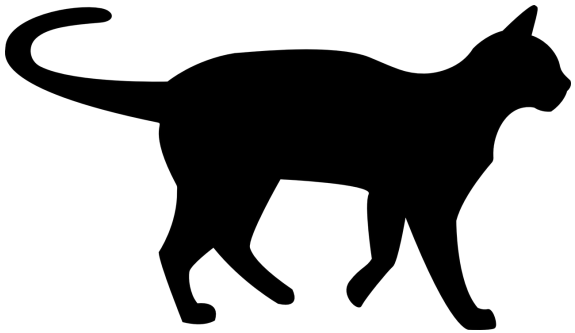


illustration: Pixabay

Protection of the chip

Shield

- metallic piste
- protection electromagnetic passive,
- or active: data is sent to these pistes and checked.

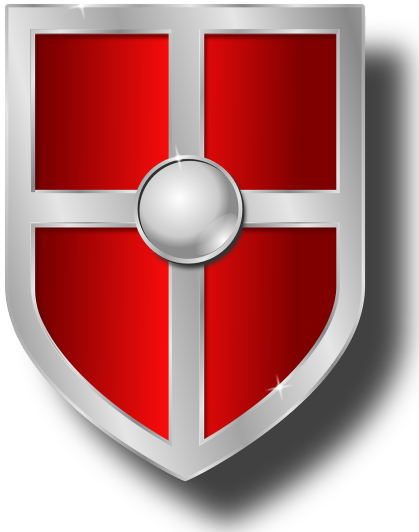
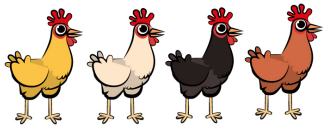


illustration: Pixabay

Les acteurs

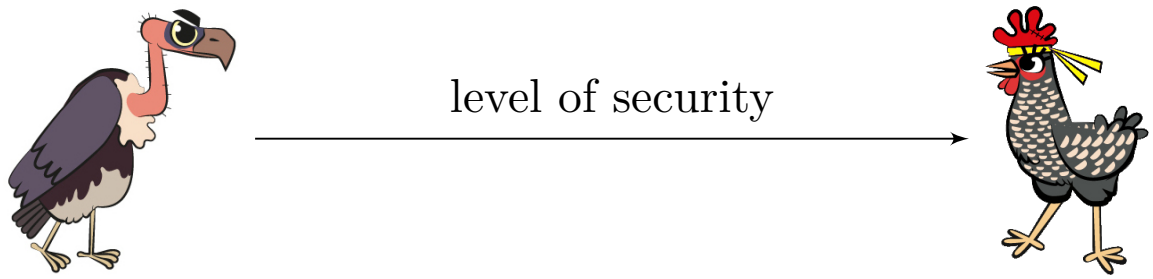
- The product for a specific utilization (ex smart cart for payment) 
- The customer of the products(ex: a bank) 
- The final clients or user sof the product 
- The insurer 
- The product maker 
- A material CESTI  to test the material
- The ANSSI   defence of the state we have follow the rules

In France ANSSI is for defence and the Army of France is for offense

illustrations: studios MoonCat

The different steps

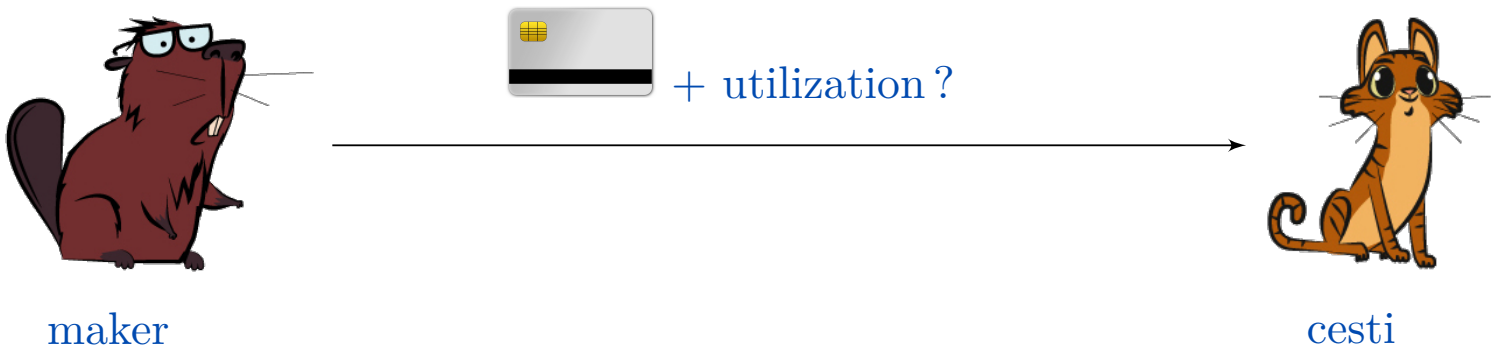
- The insurer asks a level of security.



illustrations: studios MoonCat

The different steps

- The maker asks a quote for the evaluation of pair (product, utilisation) to the cesti.



maker ask the cesti to test the device with specific config

illustrations: studios MoonCat

The different steps

- The CESTI suggests an evaluation for a level of certification given by ANSSI

illustrations: studios MoonCat

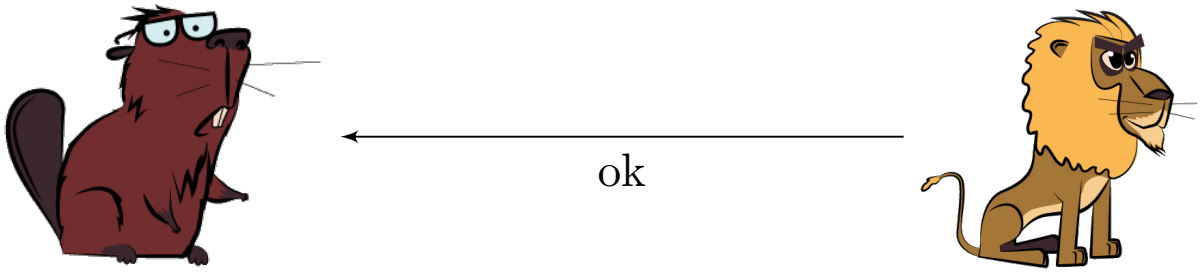
The different steps

- The maker sends this proposition to the ANSSI.

illustrations: studios MoonCat

The different steps

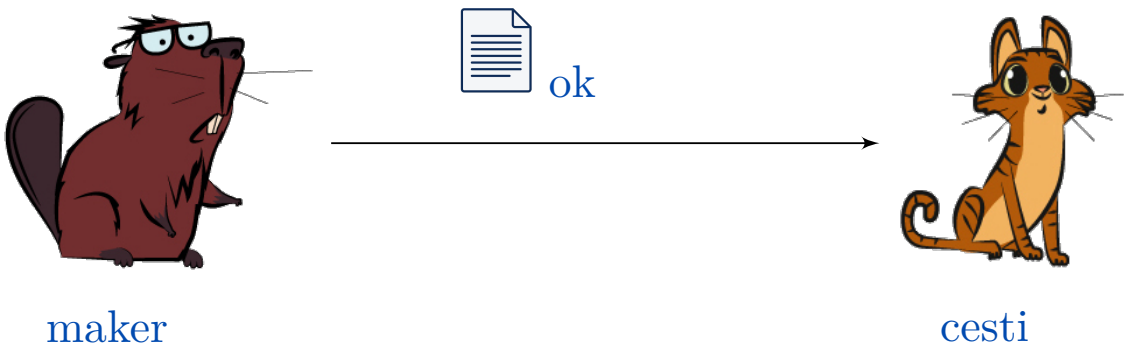
- ANSSI checks.



illustrations: studios MoonCat

The different steps

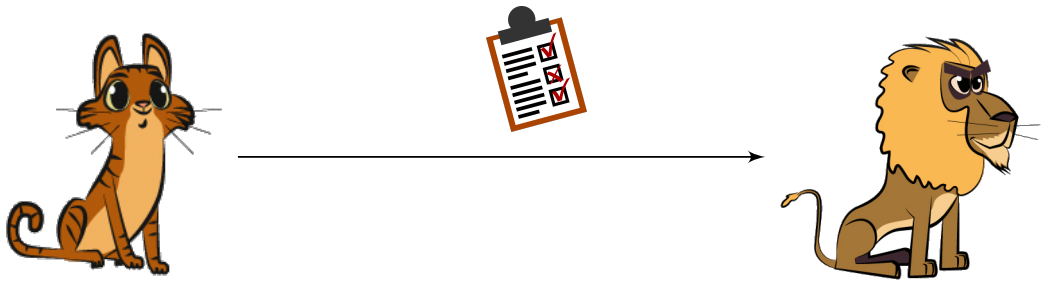
- The maker and the CESTI make a deal.



illustrations: studios MoonCat

The different steps

- The CESTI makes the product evaluation. All results are send to the ANSSI.



illustrations: studios MoonCat

The different steps

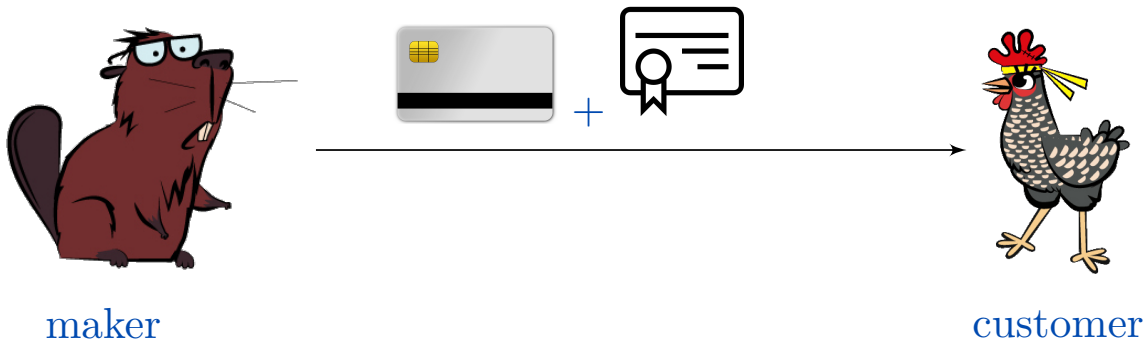
- The ANSSI check the compliance of the evaluation, then sends a certificate to the maker.

ANSSI gives the certificate to maker and not CESTI

illustrations: studios MoonCat

The different steps

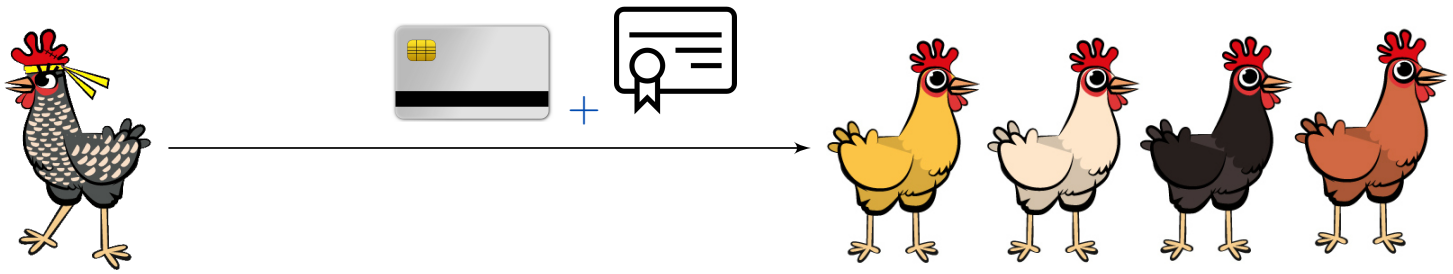
- The maker sends a product with a certificate.



illustrations: studios MoonCat

The different steps

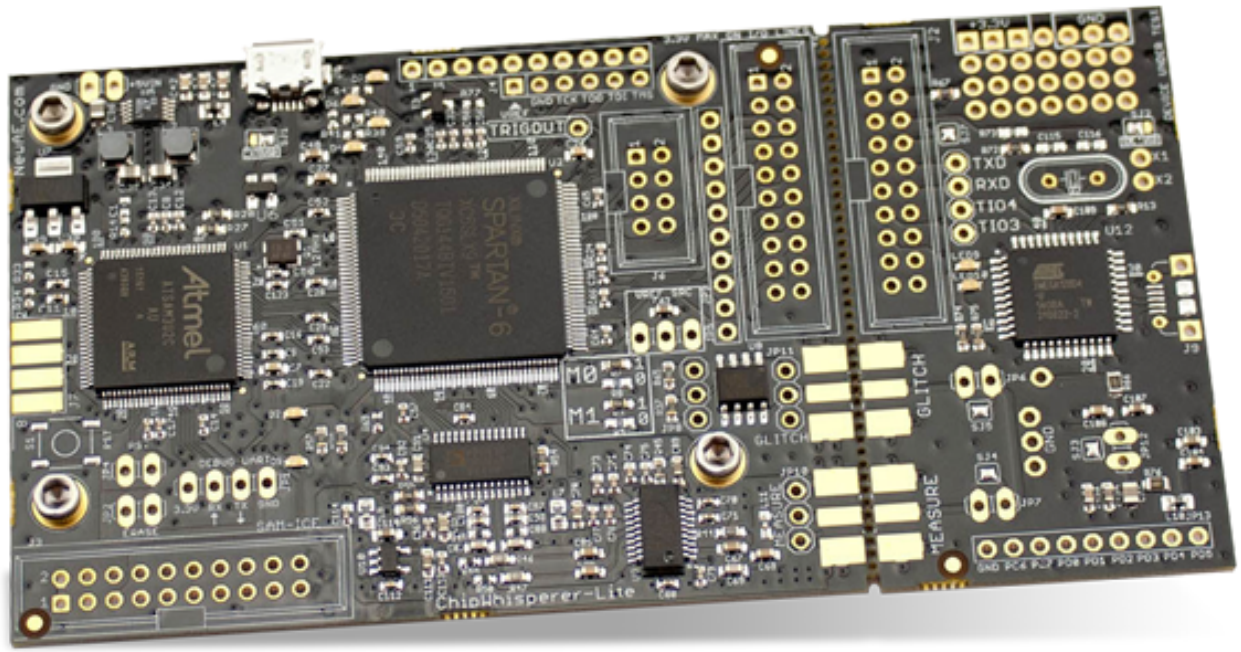
- The customer delivers the products to the final clients/ users.



customer users/final clients

illustrations: studios MoonCat

We use a ChipWhisperer-Lite from NewAE Technology [23].



La ChipWhisperer-Lite:

- is composed by a targeted device (programming in C): a ATxmega128D4,
- A FPGA use to realise attacks;
- communication in python.

The analysis of data is with programming language: julia.

You have to realize 2 attacks:

1. CPA,
2. DFA.

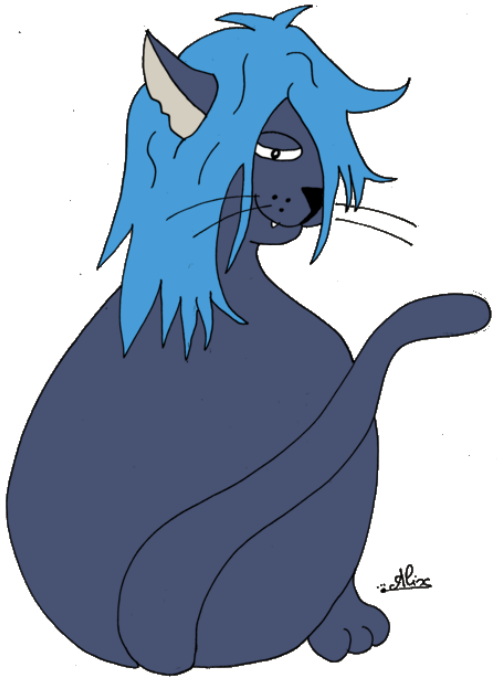
Conclusion:

- Physical attacks are a real threat.
- Do not implement your security in software, if your device can be a target of physical attacks.
- Use dedicated hardware devices.

Fun references: [25, 26, 27].

Reference: [4, 18, 20, 28, 29, 30, 31, 32, 33, 34, 35, 36].

Do you have any quesstion?



References I

NIST.

Specification for the Advanced Encryption Standard.
FIPS PUB 197, 197, 2001.

Paul Kocher.

Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS and Other Systems.
In *Advances in Cryptology - Crypto'96*, pages 104–113, New-York, 1996. Springer-Verlag.

Michael Hutter and Jörn-Marc Schmidt.

The temperature side channel and heating fault attacks.
In *International Conference on Smart Card Research and Advanced Applications*, pages 219–235. Springer, 2013.

Stefan Mangard, Elisabeth Oswald, and Thomas Popp.

Power analysis attacks: Revealing the secrets of smart cards, volume 31.
Springer, 2008.

Jean-Jacques Quisquater and David Samyde.

Electromagnetic analysis (EMA): Measures and counter-measures for smart cards.
In *Smart Card Programming and Security*, pages 200–210. Springer, 2001.

A Schlosser, Dmitry Nedospasov, J Kramer, Susanna Orlic, and Jean-Pierre Seifert.

Simple photonic emission analysis of aes photonic side channel analysis for the rest of us.
Cryptographic Hardware and Embedded Systems-CHES, pages 41–57, 2012.

References II

Paul Kocher, Joshua Jaffe, Benjamin Jun, et al.

Introduction to differential power analysis and related attacks.
1998.

Thomas S Messerges, Ezzy A Dabbish, and Robert H Sloan.

Power analysis attacks of modular exponentiation in smartcards.
In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 144–157. Springer, 1999.

Suresh Chari, Josyula R Rao, and Pankaj Rohatgi.

Template attacks.
In *Cryptographic Hardware and Embedded Systems-CHES 2002*. Springer, 2003.

Eric Brier, Christophe Clavier and Francis Olivier.

Correlation Power Analysis with a Leakage Model.
In *CHES*, pages 16–29, 2004.

Christophe Giraud.

DFA on AES.
In H. Dobbertin and V. Rijmen and A. Sowa, editor, *Advanced Encryption Standard - AES*, volume 3373 of *Lecture Notes in Computer Science*. Springer, 2005.

Eli Biham and Adi Shamir.

Differential Fault Analysis of Secret Key Cryptosystems.
In *CRYPTO*, 1997.

References III

Paul Kocher, Joshua Jaffe, and Benjamin Jun.

Differential power analysis.

In *Annual International Cryptology Conference*, pages 388–397. Springer, 1999.

Ronan Lashermes, Guillaume Reymond, Jean-Max Dutertre, Jacques Fournier, Bruno Robisson and Assia Tria.

A DFA on AES Based on the Entropy of Error Distributions.

In *FDTC*, 2012.

Benedikt Gierlichs, Lejla Batina, Pim Tuyls, and Bart Preneel.

Mutual information analysis.

In *CHES, Proceedings*. Springer, 2008.

Youssef Souissi, Maxime Nassar, Sylvain Guilley, Jean-Luc Danger and Florent Flament.

First Principal Components Analysis: A New Side Channel Distinguisher.

In *ICISC*, 2010.

Suresh Balakrishnama and Aravind Ganapathiraju.

Linear Discriminant Analysis - A Brief Tutorial.

Institute for Signal and Information Processing, Mississippi State University, 1998.

Cyril Roscian.

Cryptanalyse physique de circuits cryptographiques à l'aide de sources LASER.

PhD thesis, Ecole Nationale Supérieure des Mines de Saint-Etienne, 2013.

References IV

Pierre Bayon, Lilian Bossuet, Alain Aubert, Viktor Fischer, François Poucheret, Bruno Robisson, and Philippe Maurine.
Contactless electromagnetic active attack on ring oscillator based true random number generator.
In *International Workshop on Constructive Side-Channel Analysis and Secure Design*, pages 151–166. Springer, 2012.

Loïc Zussa.

Étude des techniques d'injection de fautes par violation de contraintes temporelles permettant la cryptanalyse physique de circuits sécurisés.
PhD thesis, Saint-Etienne, EMSE, 2014.

Loic Zussa, Jean-Max Dutertre, Jessy Clédiere, Bruno Robisson, Assia Tria, et al.
Investigation of timing constraints violation as a fault injection means.
In *27th Conference on Design of Circuits and Integrated Systems (DCIS), Avignon, France*, 2012.

Ouest France.

À rennes, ils craquent vos processeurs avec des ondes...
https://www.ouest-france.fr/bretagne/rennes-35000/rennes-ils-craquent-vos-processeurs-avec-des-ondes-5665553.

NewAE.

Chipwhisperer by newae technology inc.
https://chipwhisperer.readthedocs.io/en/latest/.

References V

Gilles Piret and Jean-Jacques Quisquater.

A differential fault attack technique against spn structures, with application to the aes and khazad.
In *International workshop on cryptographic hardware and embedded systems*, pages 77–88. Springer, 2003.

Hélène Le Bouder.

Des attaques informatiques utilisant la physique.
https://interstices.info/jcms/p_91127/des-attaques-informatiques-utilisant-la-physique.

Arnaud Tisserand.

Piratage de cartes à puces: comprendre comment ça marche.
http://www.slate.fr/story/152207/piratage-cartes-puces-comprendre-comment-marche.

Ronan Lashermes.

Attaques par faute.

Vincent Grosso.

Towards side-channel secure block ciphers.
PhD thesis, Catholic University of Louvain, Louvain-la-Neuve, Belgium, 2015.

Lionel Riviere.

Sécurité des implémentations logicielles face aux attaques par injection de faute sur systemes embarqués.
PhD thesis, PhD thesis, Telecom Paris Tech, 2015.

Christophe Clavier.

De la sécurité physique des crypto-systemes embarqués.
These de doctorat, Université de Versailles Saint-Quentin, 7:5, 2007.

References VI

Hélène Le Bouder.

Un formalisme unifiant les attaques physiques sur circuits cryptographiques et son exploitation afin de comparer et de rechercher de nouvelles atatques.
PhD thesis, Saint-Etienne, EMSE, 2014.

Julien Iguchi-Cartigny Ahmadou Al Khary Séré, Jean-Louis Lanet.

Carte à puce javacard : Protection du code contre les attaques en faute.
http://docplayer.fr/2810588-Carte-a-puce-java-card-protection-du-code-contre-les-attaques-en-faute.html.

Nicolas Moro.

Security of assembly programs against fault attacks on embedded processors.
PhD thesis, Université Pierre et Marie Curie - Paris VI, 2014.
https://tel.archives-ouvertes.fr/tel-01147122.

Jessy Clédière.

Cours sécurité et canaux auxiliaires. introduction aux attaques par fautes.
https://www-polsys.lip6.fr/~renault/SCAM2/PDF/perturbations.pdf.

Pablo Rauzy.

Sécurité et systèmes embarqués. les injections de fautes.
pablo.rauzy.name/teaching/sese.

Ingrid Exurville.

Détection non destructive de modification malveillante de circuits intégrés.
PhD thesis, EMSE, 2015.