Attacks & Countermeasures in Real Life

Benoit Feix

Master Cryptis - Limoges University

29 November 2017

Agenda

- What is a security product today?
- Invasive Attacks and Countermeasures (SE / SoC)
- Side-Channel Attacks and Countermeasures:
  - Examples on DES, AES and RSA
- Combined Attacks and Countermeasures
- A new area for side-channel gamers: White-Box cryptography
- Security Evaluation and Certification Schemes (kézaco?)
- … and so what?

WTF is a Security Product

Some Products

- 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

Product Conception Phases

Tests
Production

Préparation

Circuit Intégré
Développement

OS
Développement

CRYPTO
Développement

Applications
Développement

Sécurisation

Produit

Personnalisation

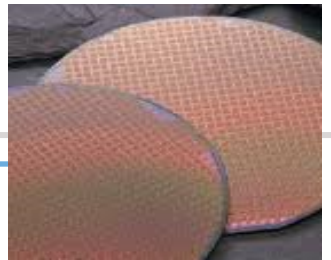
Pré
personnalisation

Pré
personnalisation

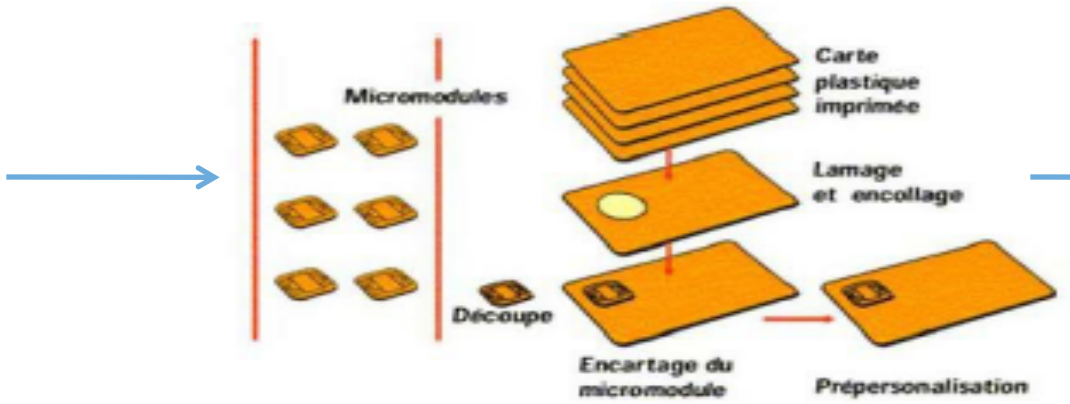
Initialisation

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

Integrated Circuit (Secure Element) Architecture

Smart cards, System-On-Chips, Mobile phones, tablets, PCs, etc.

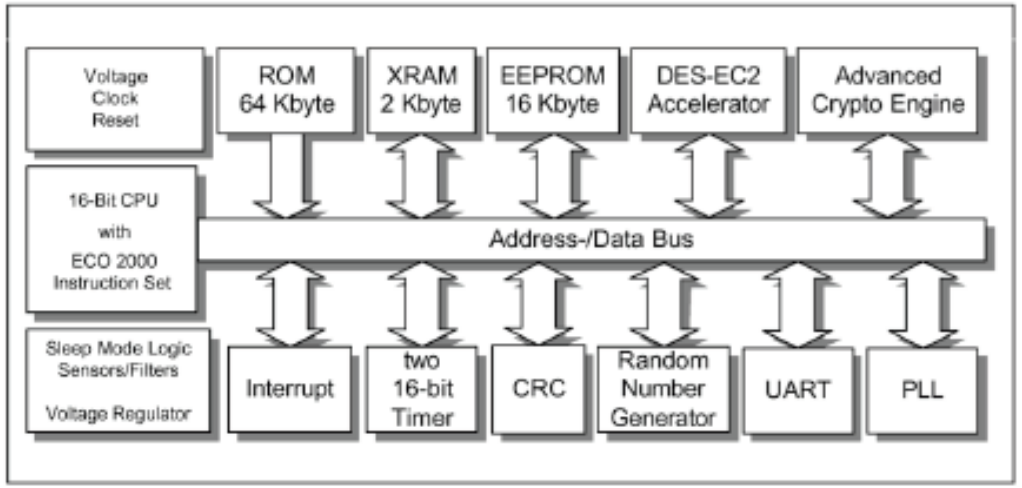


Figure : SLE66 Infineon Block Diagram

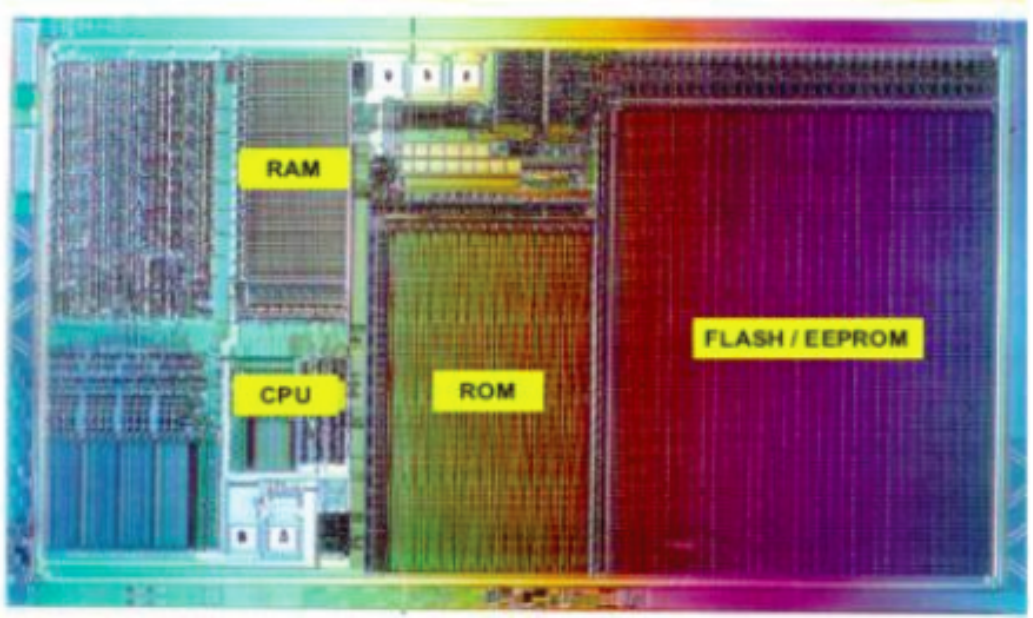
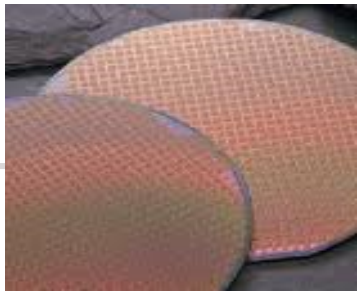
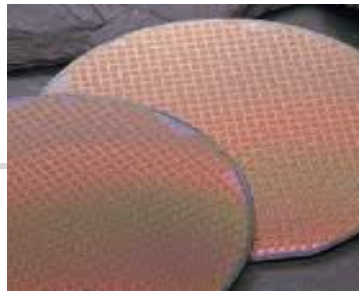


Figure : Smart card IC picture



Integrated Circuit (Secure Element) 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:
  - DES
  - AES
  - CRC
  - Long Integer Arithmetic (RSA, ECDH, ECDSA)
- In/Out IPs
- MPU/MMU: Memory Management Unit
- Security sensors



Standard Crypto Algorithms

- Symmetric
  - DES/TDES: NIST recommendation now TDES EDE3 for 2^{40} use of same key
  - AES 128-192-256
- Hash Functions
  - SHA-1, SHA-256
  - RIPEMD 160
  - SHA-3: KECCAK

  → **HMAC Functions**
- Asymmetric
  - CRT-RSA – RSA SFM
  - DSA / ECDSA
  - DH / ECDH
  - OBKG
- Stream Ciphers

Operating System

- Two kinds of operating systems for SE
  - Closed OS (dedicated) (disappearing)
    - Mostly mono application, dedicated to unique usage like payment cards, health cards, mobile SIM card
    - Now with technology improvement it is disappearing
  - Open OS
    - Not dedicated to a specific usage, they can install all kind of application after product is manufactured: Java Cards, or MULTOS based.

Operating System Functionalities

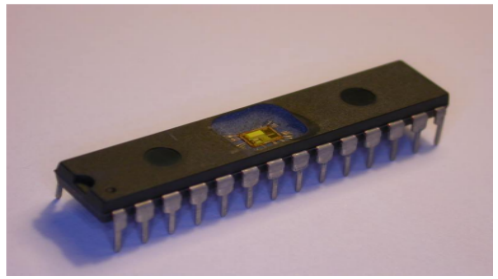
- Data In/Out management with the related communication protocols
- Memory Management in different working areas
- Basic Functions: utils and services for the applicative codes
  - Secure management of critical operations like data transfers and copies
  - Cryptogram - session keys - random generation – CRC – computations
  - Secure counter values management
  - Authentication
  - Protocols, APDU commands …
- Management of access controls: read, write, RW for each memory area
- Application Secure Loading
- Multi-applicative mode management
- Lifecycle secure management
  - Initialisation
  - Pre-personalization
  - Personalisation
  - Mode User
  - End Of Life

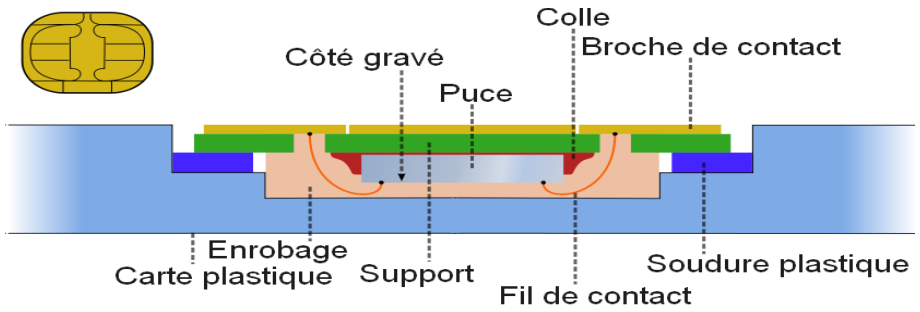
(Designing) A Security Product

Invasive Attacks and Countermeasures

Reverse Engineering

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



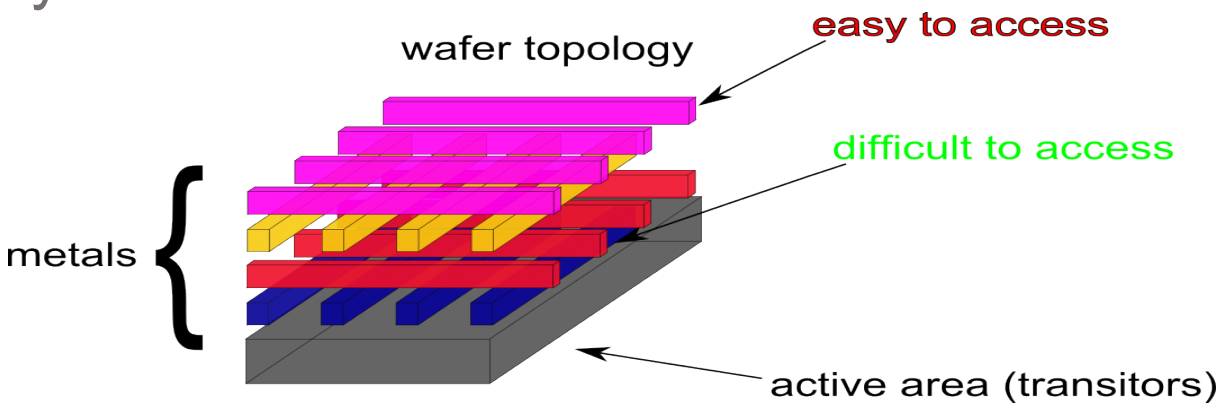


  - Use chemical products to remove (destroy) epoxy (resin) with nitric acid and acetone …
    - →We observe now the chip that we can analyze using microscope(s)
- Reconnect the circuit on another ‘’package’’
  - In order to use it again we reconnect the IO pins, clock pins, VCC pads, etc.
  - We can then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

Reverse Engineering

Reverse the IC Layout

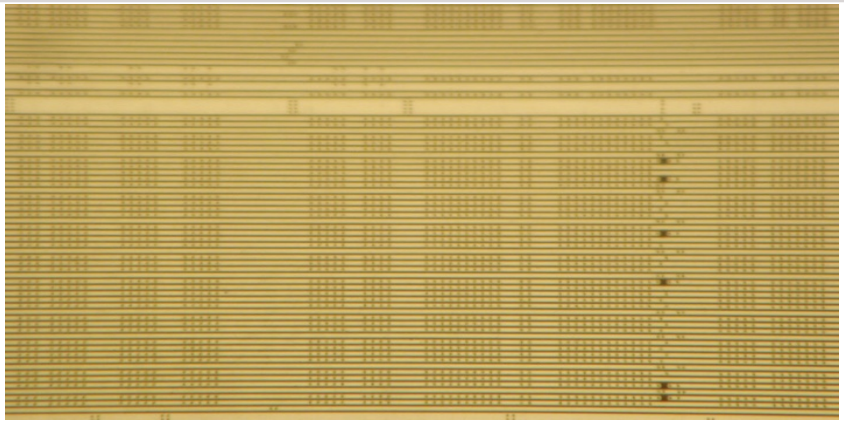
- Re-assemble metal layers
  - Integrated Circuit = SUM of metal layers interconnected each others



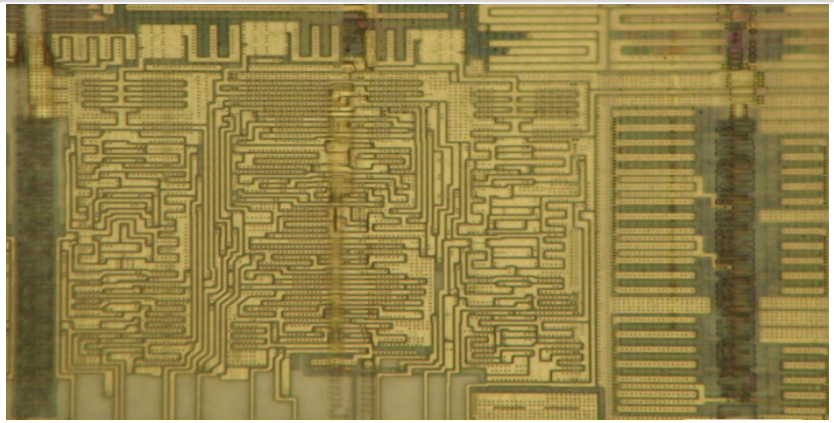
  - Using an optical microscope we can observe and take a picture of each metal layer
    - We start at the higher (upper) layer
    - We remove chemically this layer with fluoric acid or a plasma machine
    - We iterate until we reach the lowest layer (metal 1)

Reverse Engineering

Reverse the Layout



View of top métal

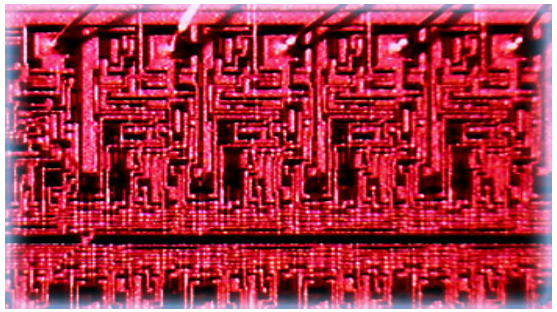


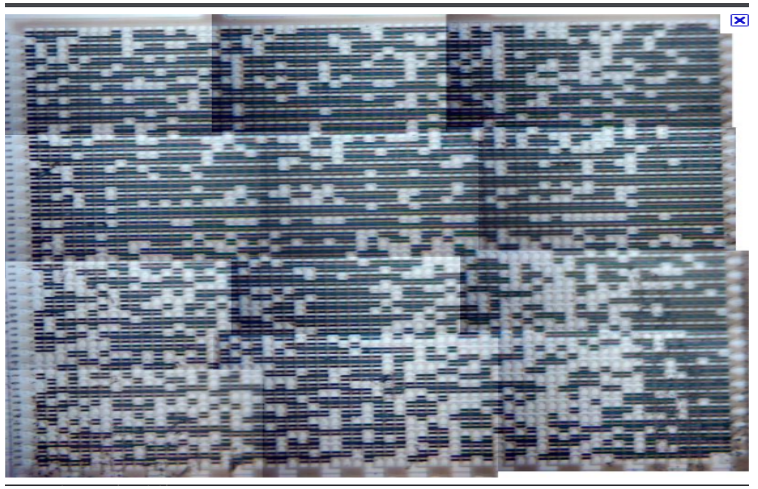
View of 1 metal lower after removal of top metals

- 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

Reverse Engineering
Read Memories Content

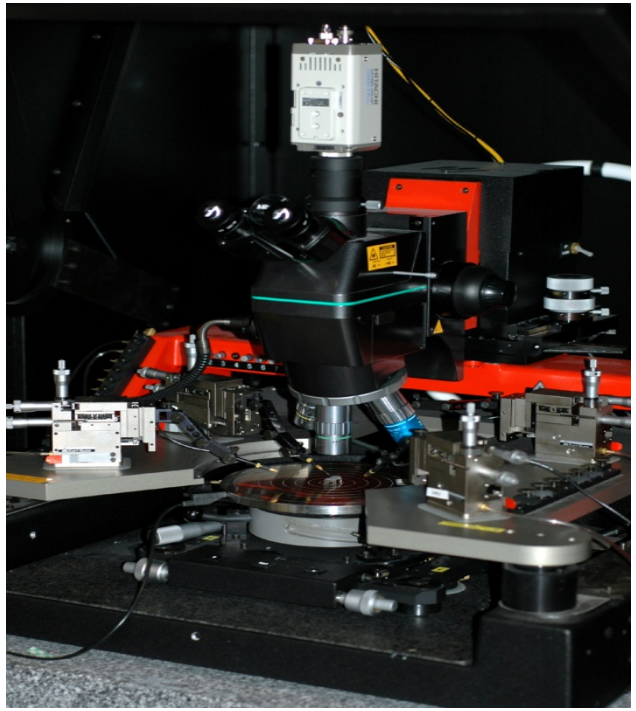
- Extract (read) the non volatile memories content (data, code).
- RAM, EEPROM, FLASH : difficult but possible
- ROM:
  - Easier
  - Related to the kind of memory (diffused or not)
  - Require different techniques
- Depend to the technological node:
  - 90nm, 65nm
  - 28nm (!)
  - 10nm is the current best techno

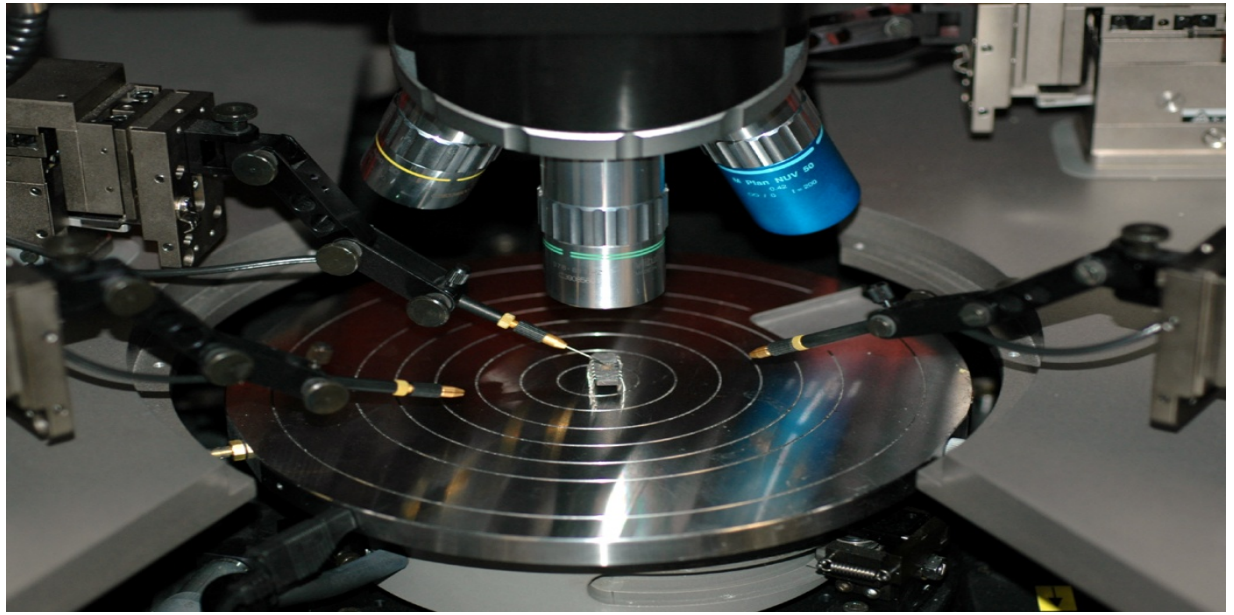




Read/Modify bits with Probing

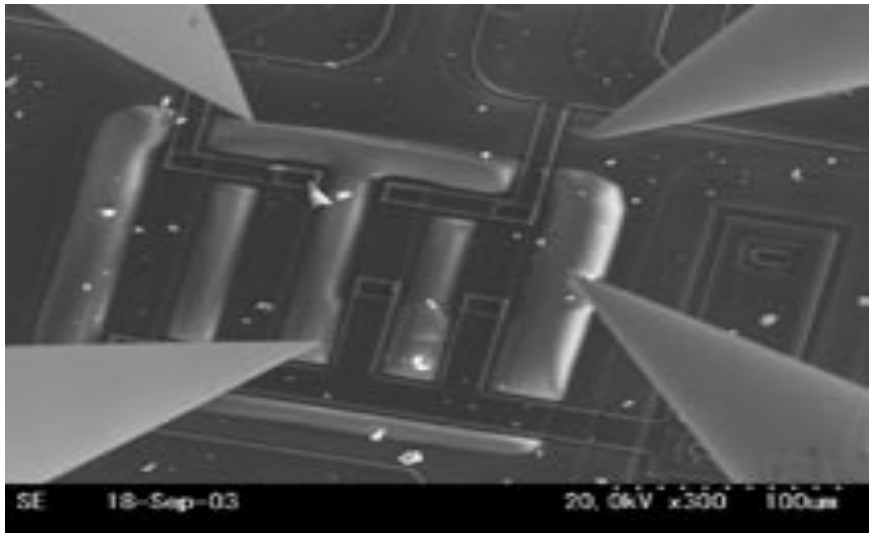
- Require a probe station





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.

Modify the Circuit
Focused Ion Beam (FIB)

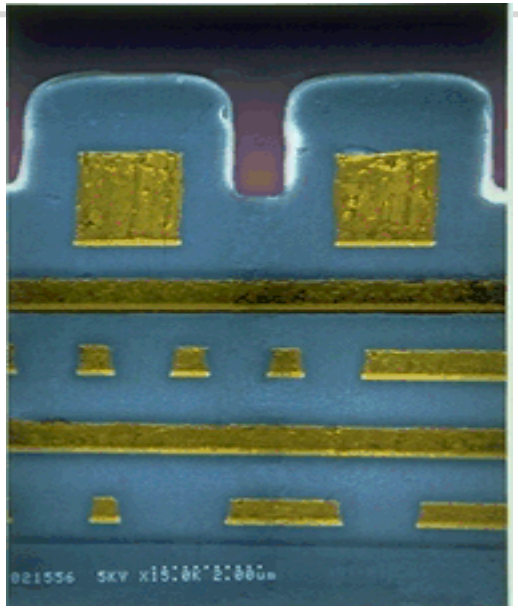
- FIB (Focused Ion Beam) stations are most often used to analyze (and “debug”) failures in circuits

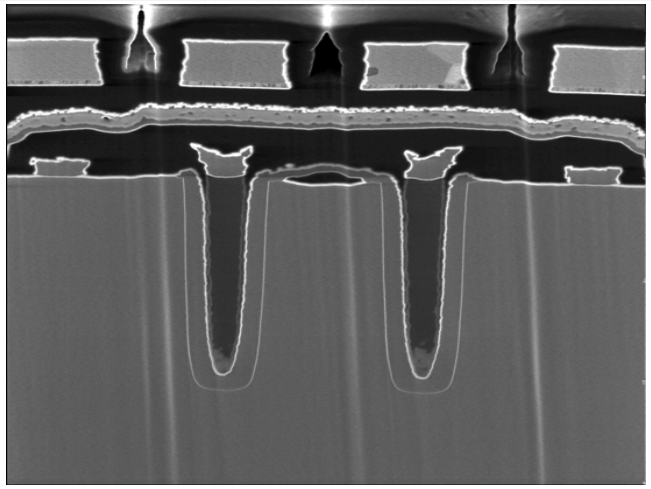
- FIB can
  - ‘’remove’’ metal, and modify connections
  - He can add metal and then create new connections

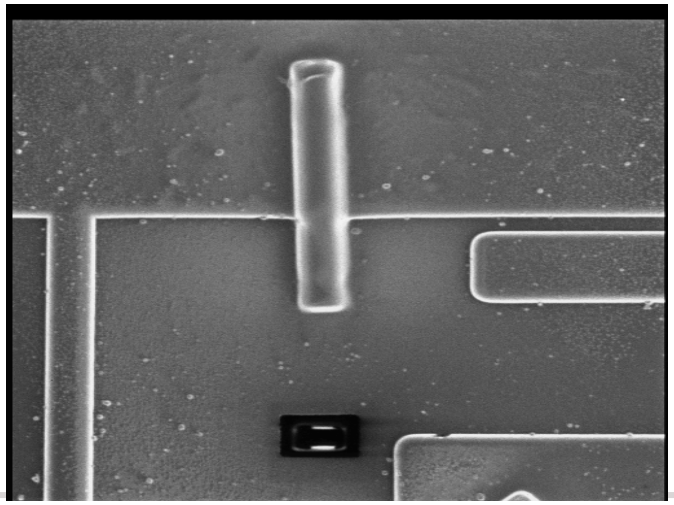
- Then an ATTACKER can
  - Access to “hidden” signals at lower metal levels
  - Modify the circuit (behavior)

- Very expensive .. But low cost for renting !

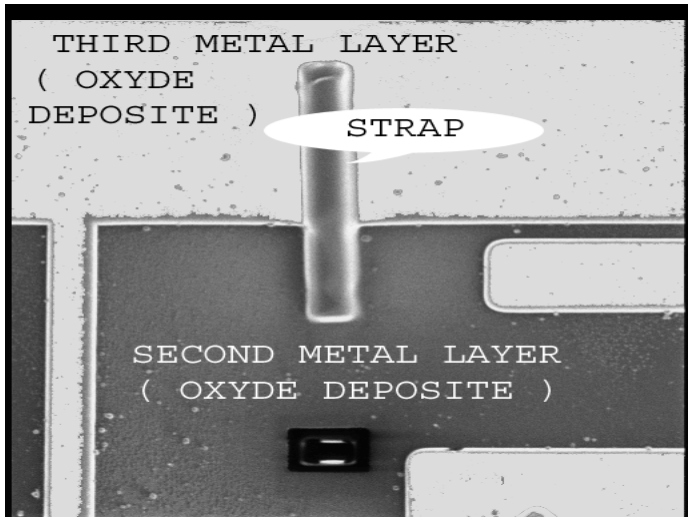
Modify the Circuit Focused Ion Beam (FIB)







Cross Sections

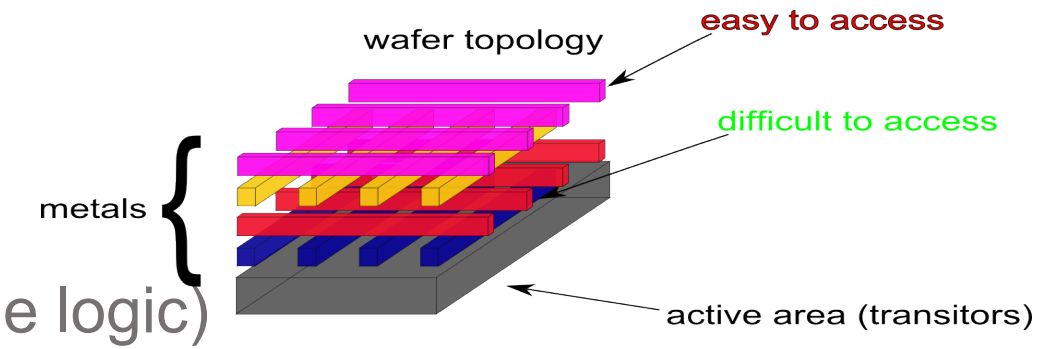


- Very powerful tool
- Can defeat product security
- Expensive
- Can be rent
- Usage to be be monitored

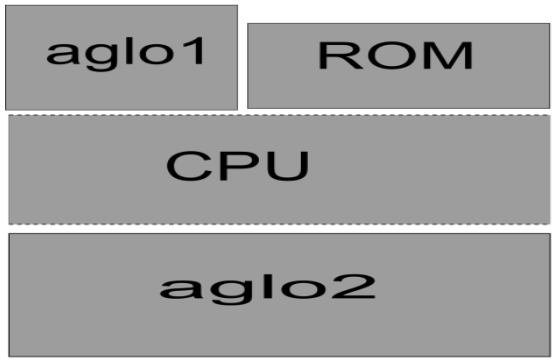
Countermeasures
Vs.
Invasive Attacks

IC Reverse Engineering Countermeasures

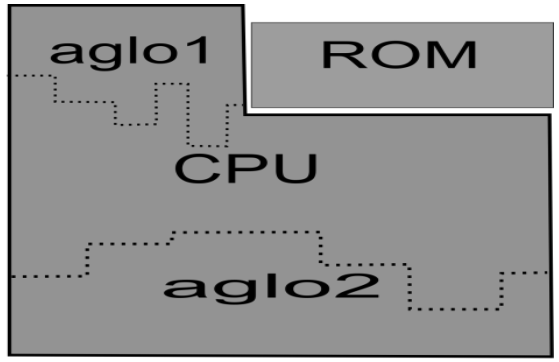
- Make design more complex
  - Embed sensitive lines in lower levels
  - Use thinner technologies (< 65 nm)
  - Make the structure more compact (i.e.. Glue logic)



- 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
- 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

Remember …

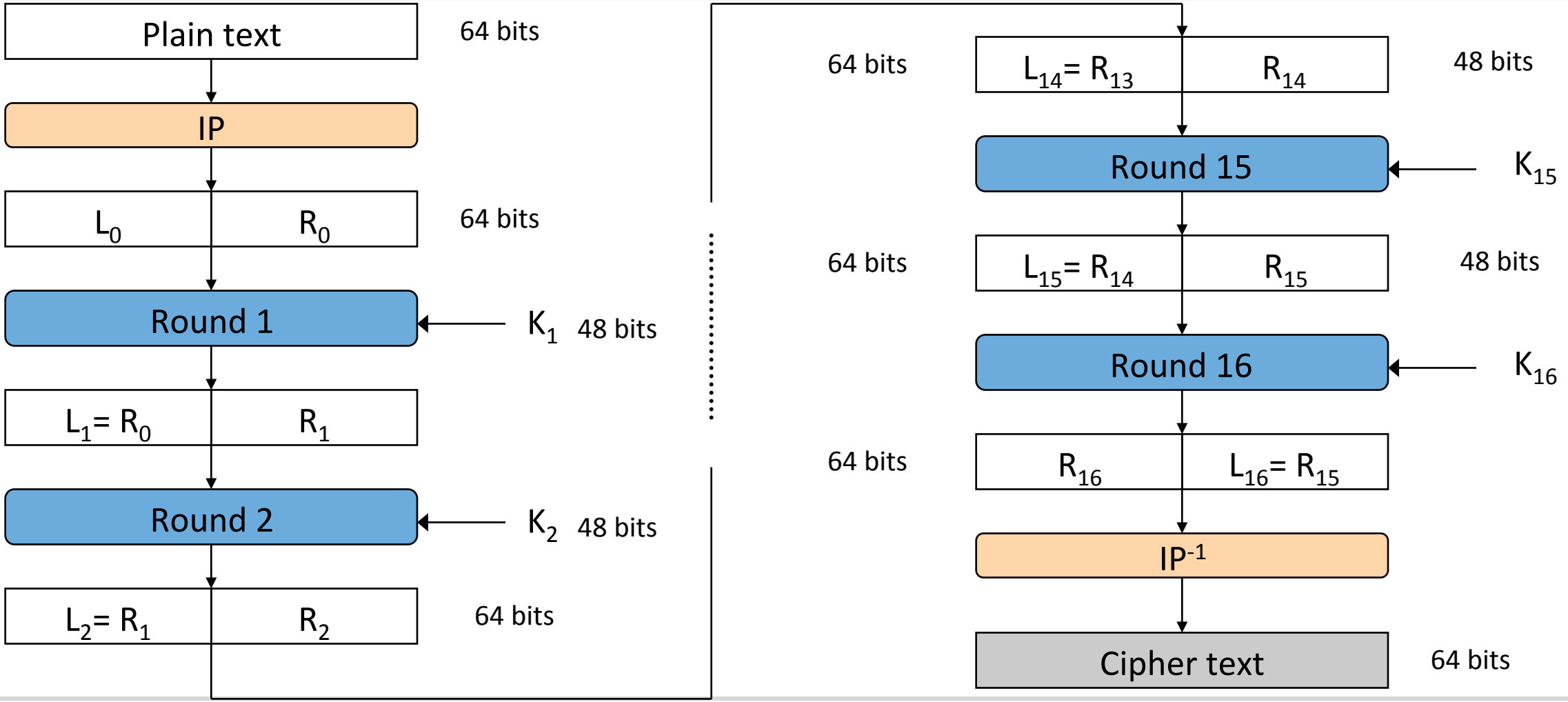
Symmetric Cryptography …

Data Encryption Standard

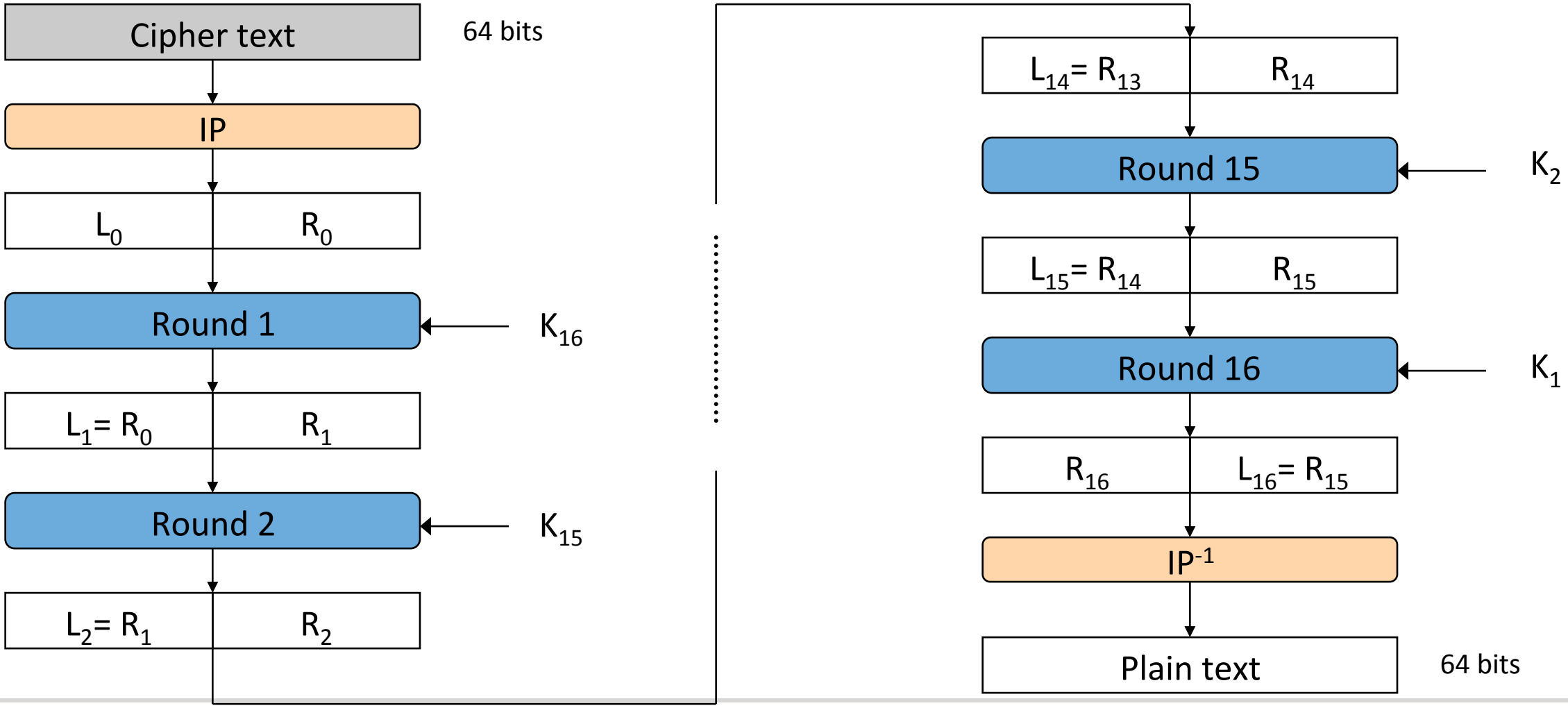
- Block Cipher designed by IBM
- Selected as standard for encryption by the NIST in 1976: cf. DES_FIPS_PUB_46-3
- Feistel scheme with 16 rounds
- Input = 64-bit plaintext
- Output = 64-bits cipher text
- Using a 64-bit secret key

(Initial proposal was ‘Lucifer’ a 128-bit key block cipher)

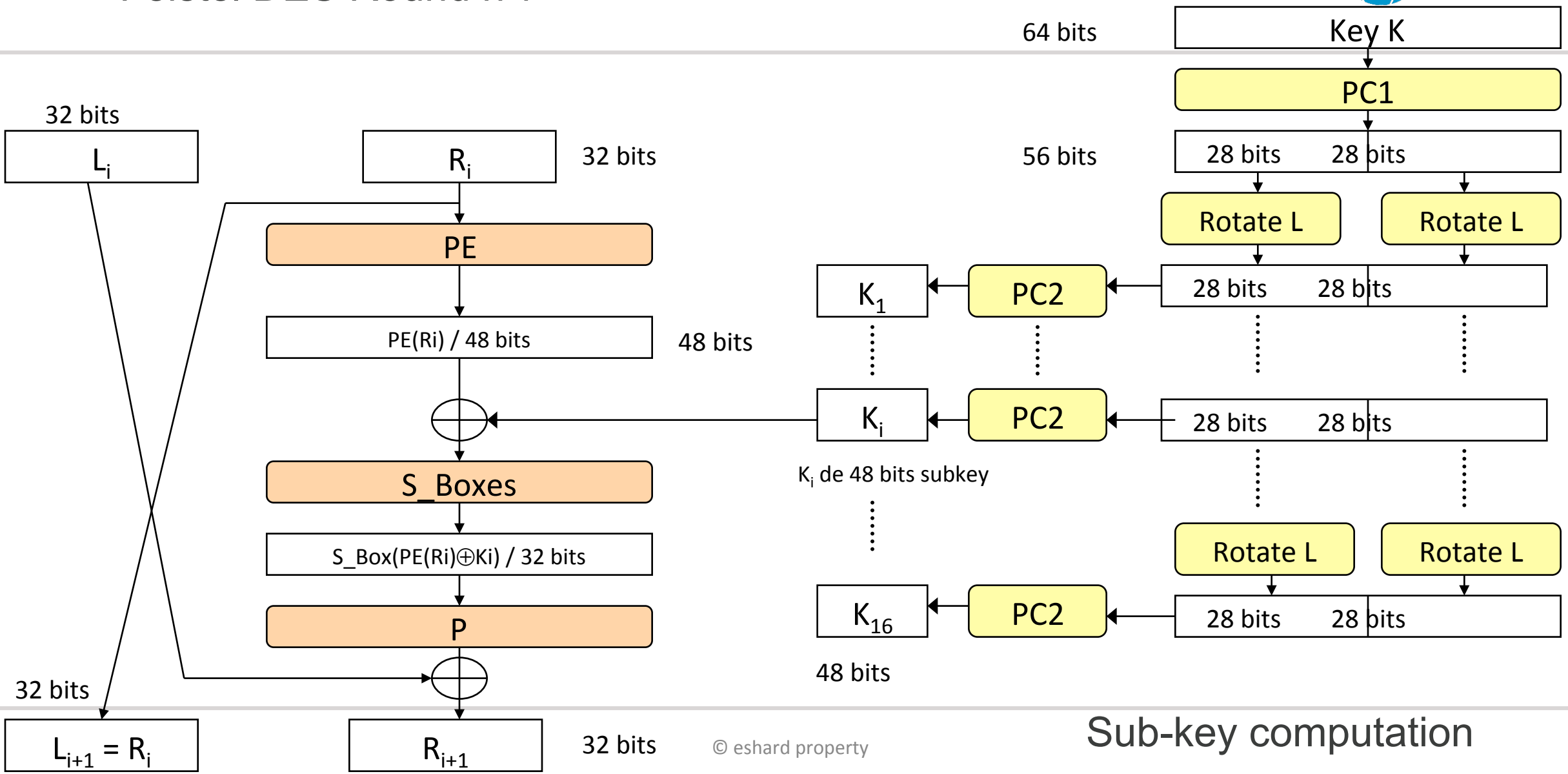
DES Encryption



DES Decryption

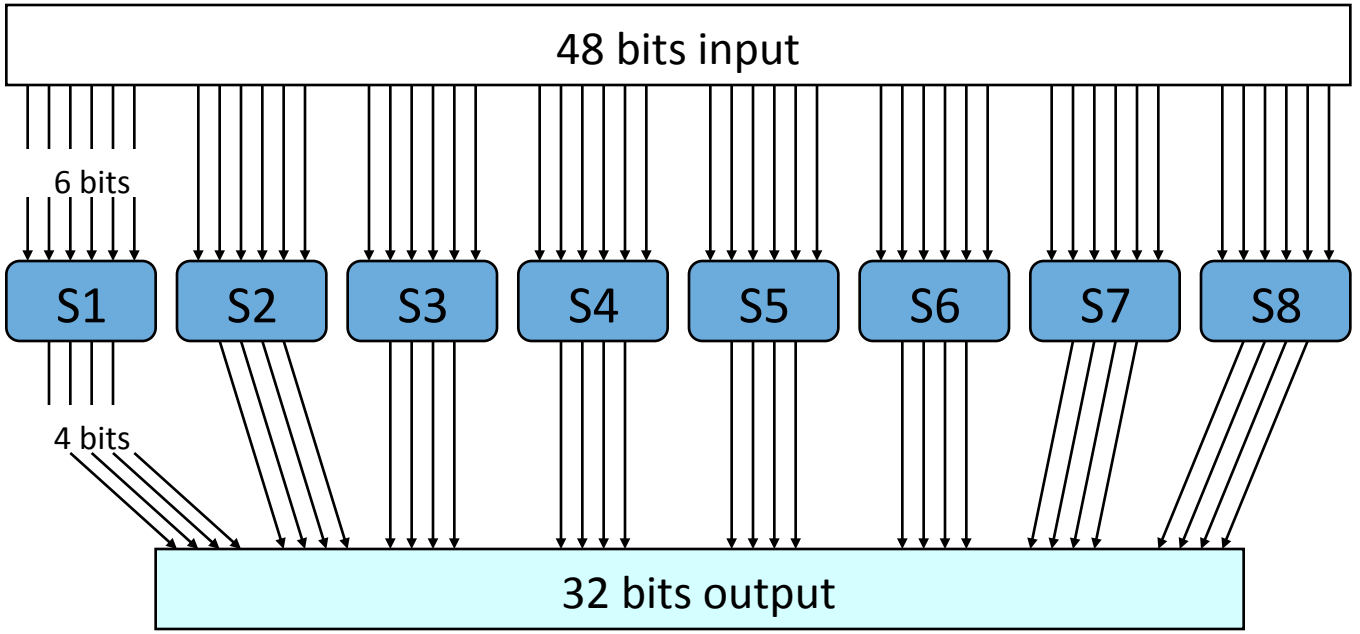


Feistel DES Round n°i



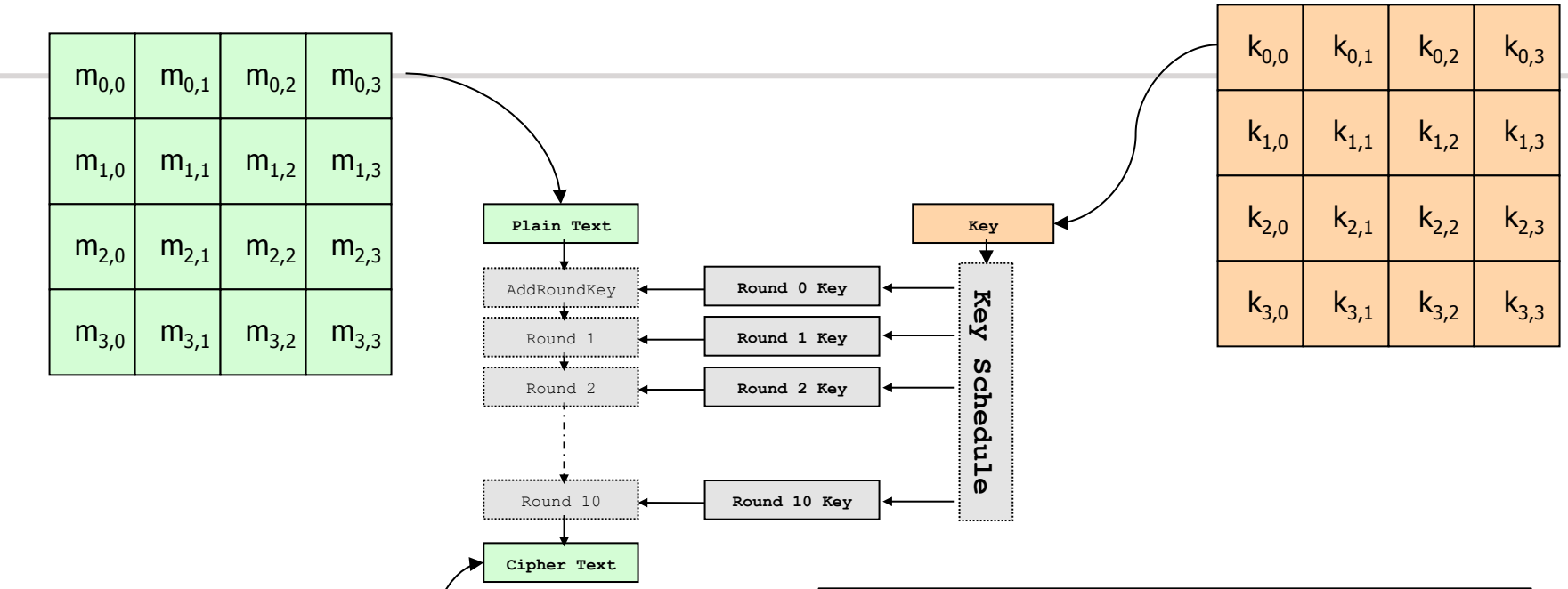
Sub-key computation

DES S(ubstitutions)_Boxes DES



Si are substitution of 6 bits values by 4 bits values.

AES Data Structure



Intermediate data are seen as 4 x 4 byte tables:

- plain text
- round key
- internal state
- cipher text

Remember …

Public Key Cryptography …

Public Key Cryptography

- Public Key Cryptography introduced in 1976 by W. Diffie and M.E. Hellman.
  - Key Exchange Protocol
- Second major invention is RSA crypto-systems by R.L. Rivest, A. Shamir and L. Adleman in 1978.
  - Encryption and Signature usage
- T. Elgamal designs the T. Elgamal crypto-system in 1985
  - Encryption and Signature usage
  - Base of the Digital Standard Signature (DSA for Digital Standard Algorithm – FIPS 186-3)
- Elliptic Curve Cryptosystems independently introduced by N. Koblitz and V.S. Miller in 1986.
  - ECDSA standard, ECDH, ECES

- In 1997, it was publicly disclosed that asymmetric key algorithms were (also) developed by J.H. Ellis, C. Cocks, and M. Williamson at the Government Communications Headquarters (GCHQ) in the UK in 1973.

- Necessitate a Public Key Infrastructure (PKI) to manage public keys certificates and their life cycle.

Public Key in Embedded Devices

- Most used asymmetric cryptosystem is still RSA, ECC more and more present
- Key size are now bigger than 1024 bits

- RSA CRT
  - using the Chinese Remainder theorem allow increasing the calculation (~ 3 times faster than Straight Forward Method) (IN theory: 4)
  - It requires knowledge of all secret keys: prime factors of the module.

- ECDSA as defined by FIPS 186-3 for signature.

- DH and ECDH for Key establishment.

Public Key in Embedded Devices

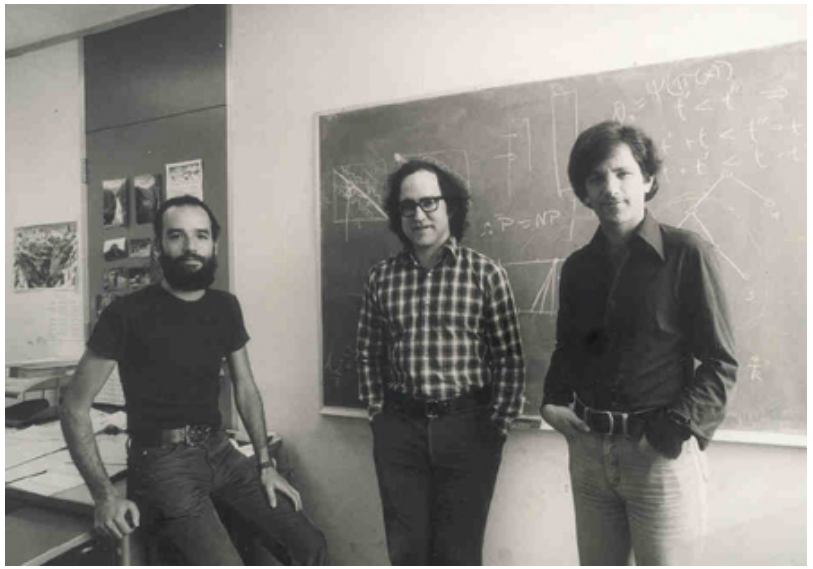
- Device Architecture
  - For years CPU core were 8-bit architecture –
    - did not allow efficient multiplications and reductions,
  - Needed accelerator(s) for these calculations – Done.
  - Today still 8-bit cores, 16-bit cores, more and more 32-bit cores (ARM) … but high efficiency requirement for bigger RSA keys (1536-2048),
  - ECC are more and more spread and also need accelerators.

- Different Coprocessors for long arithmetic calculation
  - Barrett reduction based,
  - Montgomery reduction and multiplication based,
  - Quisquater algorithm based,
  - Sedlak arithmetic,

  … + different architectures choices

→ Many different coprocessors available among manufacturers: Infineon, NXP, STM, Samsung, Broadcom, Qualcomm, Intel… (not exhaustive) …

RSA
SFM and CRT Methods



- **RSA SFM**

Signature *S* of message *m*:

Secret key: d

$S = m^d \bmod n$ (modular exponentiation)

Signature Verification:

Public key: n, e

$m' = S^e \bmod n$ (modular exponentiation)

If m' = m → OK

- **RSA CRT**

Signature *S* of message *m*:

Secret key: d_p, d_q, p, q, i_q

$S_p = m^{dp} \bmod p$

$S_q = m^{dq} \bmod q$

$S = S_q + (S_p - S_q) \times i_q \bmod p) \times q$

Signature Verification:

Public key: n, e

$m' = S^e \bmod n$ (modular exponentiation)

If m' = m → OK

Parameters:

modulus ***n*** = $p \times q$

p, q prime numbers of equivalent sizes – ***secrets****.*

e *public key, small --* ***e****=3 or* $2^{16}+1$

d $= e^{-1} \bmod \varphi(n) = e^{-1} \bmod (p-1)(q-1)$

$i_q = q^{-1} \bmod p$

Naïve Exponentiation

Algorithm 3.1 Exponentiation from left to right

INPUT: integers m and n such that $m < n$, k-bit exponent $d = (d_{k-1}d_{k-2}\dots d_1 d_0)_2$
OUTPUT: ModExp$(m,d,n) = m^d \mod n$

Step 1. $a = 1$

Step 2. for i from $k-1$ to 0 **do**

$a = a \times a \mod n$

If $d_i = 1$ Then $a = a \times m \mod n$

Step 3. Return(a)

Side-Channel Attacks

Some fundamental side-channel attacks publications

- P. C. Kocher. *Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and other systems.* CRYPTO 1996.
- P. C. Kocher, J. Jaffe and B. Jun. *Differential Power Analysis.* CRYPTO'99.
- T. S. Messerges, E. A. Dabbish and R. H. Sloan. *Power Analysis Attacks of Modular Exponentiation in Smartcards.* CHES 1999
- C. Walter.*Sliding Windows Succumbs to Big Mac Attacks.* CHES 2001
- P-A. Fouque and F.Valette. *The Doubling Attack - Why upwards is better than downwards.* CHES 2003.
- B. Chevallier-Mames, M. Ciet and M. Joye. *Low-cost Solutions for Preventing Simple Side-Channel Analysis: side-channel atomicity.* IEEE 2004
- E. Brier, C. Clavier and F. Olivier. *Correlation Power Analysis with a Leakage Model* CHES 2004
- ...

Side-channel bench

Passive Attacks

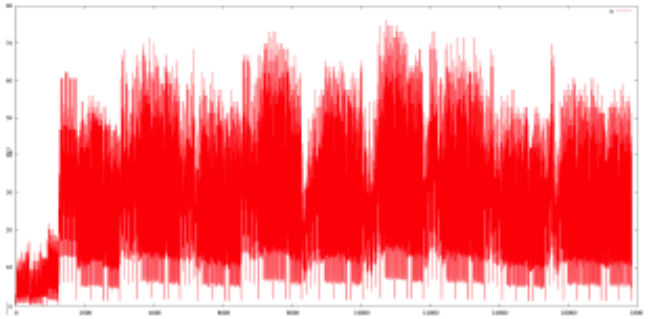
Computer

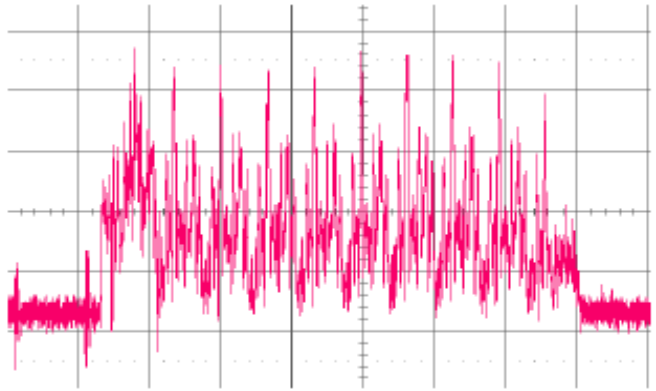
Card Reader

Oscilloscope

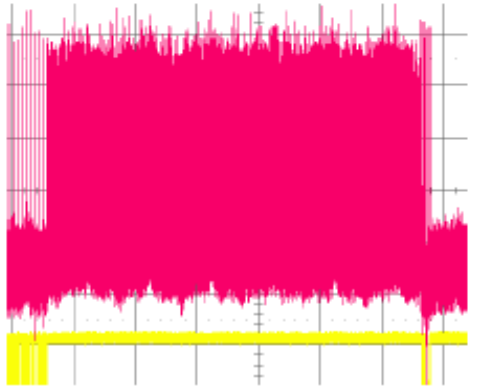
Side-Channel Traces Examples

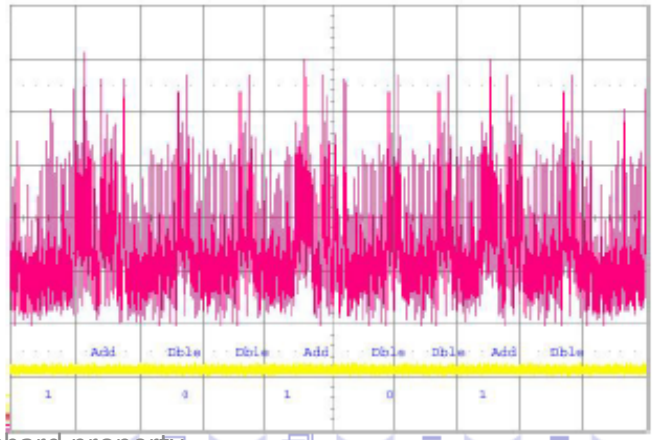
AES Traces





Exponentiation (Scalar) Traces





DPA Attack Example
Demo DPA on DES

- This is the first DPA attack published by Kocher, Jaffe and Jun.

- Notebook presented performs the following tasks:
  - Collect traces from DES executions
  - Perform first observation on traces
  - Run original DPA attack from Kocher et al.
  - Recover the DES round 1 sub-key
  - Brute force to recover the missing 8-bit

- Let’s move to esDynamic notebook

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:

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

- Consider a set of C_j curves: j power traces of exponentiations $Exp(m_j, d, n) = m_j^d \mod n$,
- D_j: function of m_j and secret bits of d,
- Linear correlation factor $\rho_{C,H}$: can be used to correlate each power curve C_j with $H(D_j \oplus R)$ and validate guesses on d,
- Maximum correlation value = correct guess for bits of d.

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, yes!
  - 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

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

Statistical Tests

- Attack consists in measuring the dependency between
  - a sample S1 of data = physically measures traces … is a set of values
  - a sample S2 of estimated data

- Lot of statistical tests can be used for that
  - Bravais-Pearson (CPA) is linear regression of order 1
  - Pearson Rank
  - Kendall Tau
  - Mutual Information Analysis
  - Euclidean Distance

- In practice they are more or less sensitive to noise but they can be used
- CPA has been observed to be one of the best still today

CPA Attack Example
Demo CPA on AES

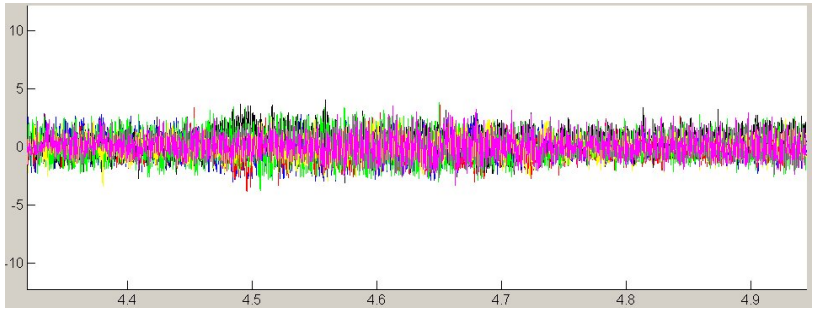
- Notebook presented performs the following tasks:
  - Collect traces from AES executions
  - Perform first observation on traces
  - Run CPA attack
  - Recover the AES key

- Let’s move to esDynamic notebook

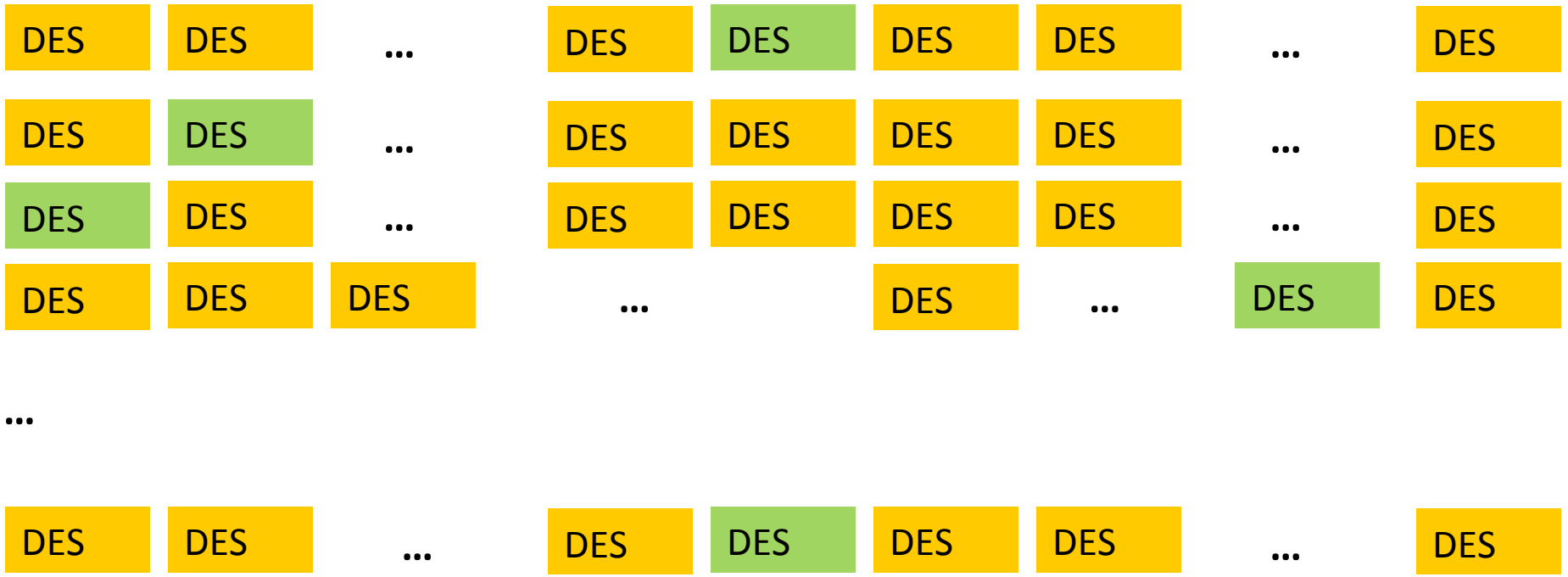
Countermeasures vs. Side-Channel Attacks in Symmetric Cryptography

Categories of Countermeasures

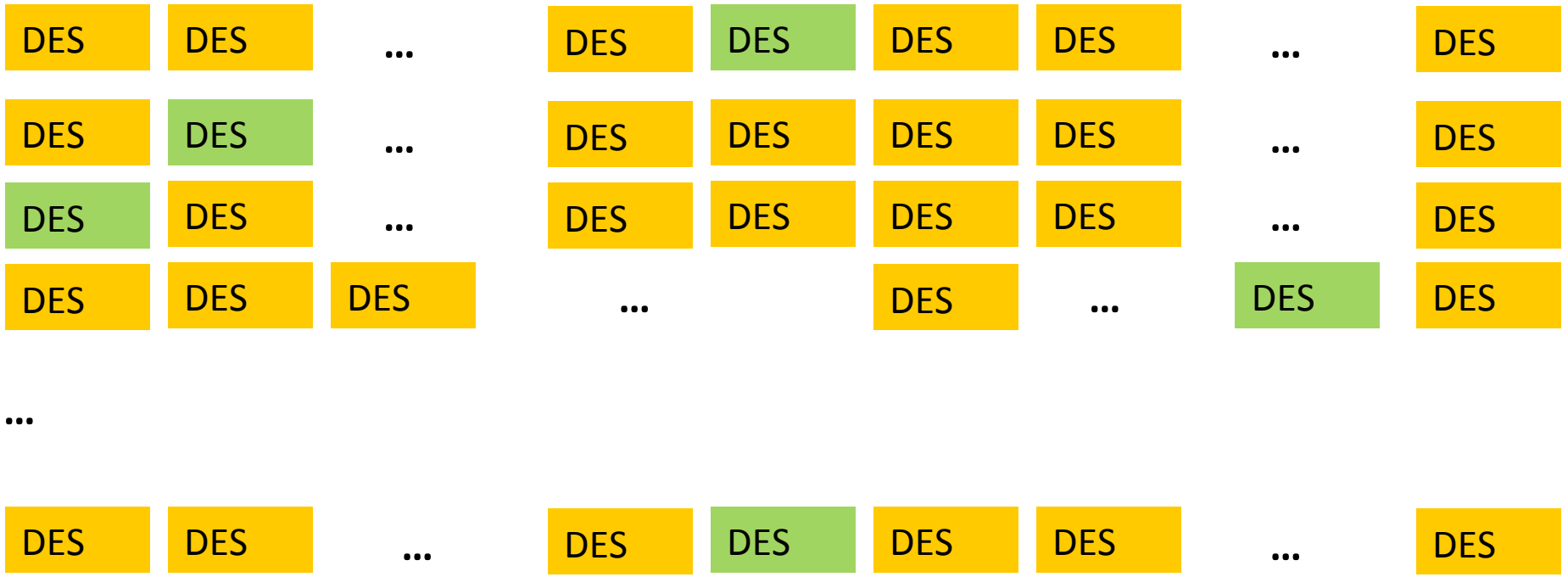
- Protocol
  - Paddings
  - Session keys
  - Fresh re-keying
  - Unknown input and/or output (partially)
- Desynchronize
  - Hardware de-synchronization: jitter, clock and frequency jitters, random delays
  - Shuffling
  - One amongst N techniques
- De-correlate
  - Hardware techniques to balance consumption (dual rail)
  - Masking techniques
  - VS CPA. use value and its complement (i.e. DES)



One among N countermeasures



One among N countermeasures



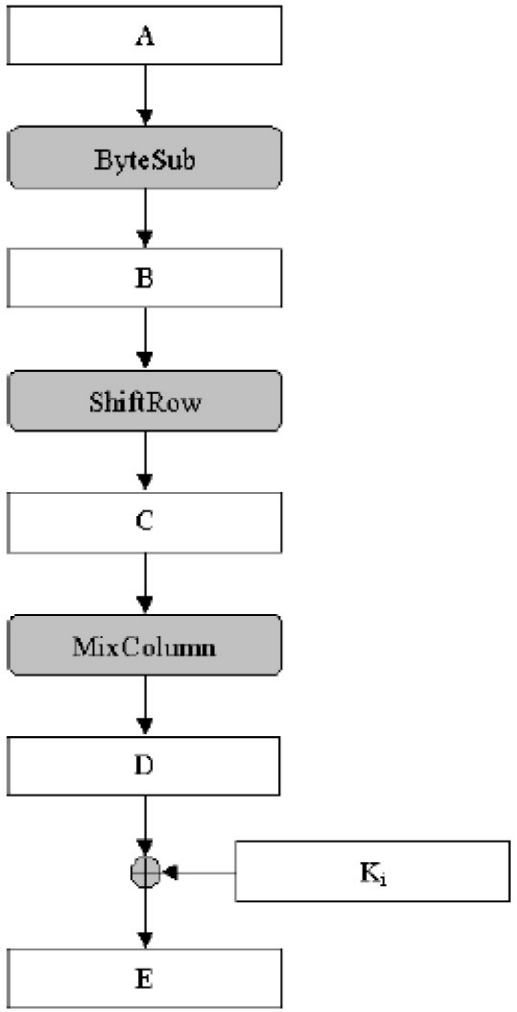
- Same input message with random fake keys keys
- Different (random fake input) messages for correct key: key is manipulated more (!!)
- Cipher texts attacks are more efficient

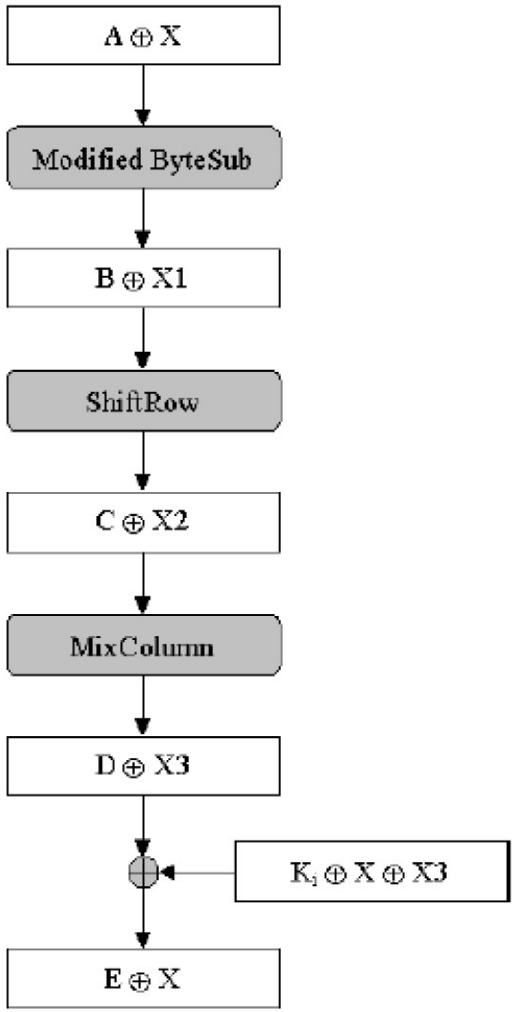
Masking Techniques

- 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
      - 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)

AES masked example

```
for (j=0; j<256; j++) {
        ByteSubMasked [j^X] = S_box[j]^X1;
}
```





CPA resistant AES Example
Demo on AES

- See implementation code

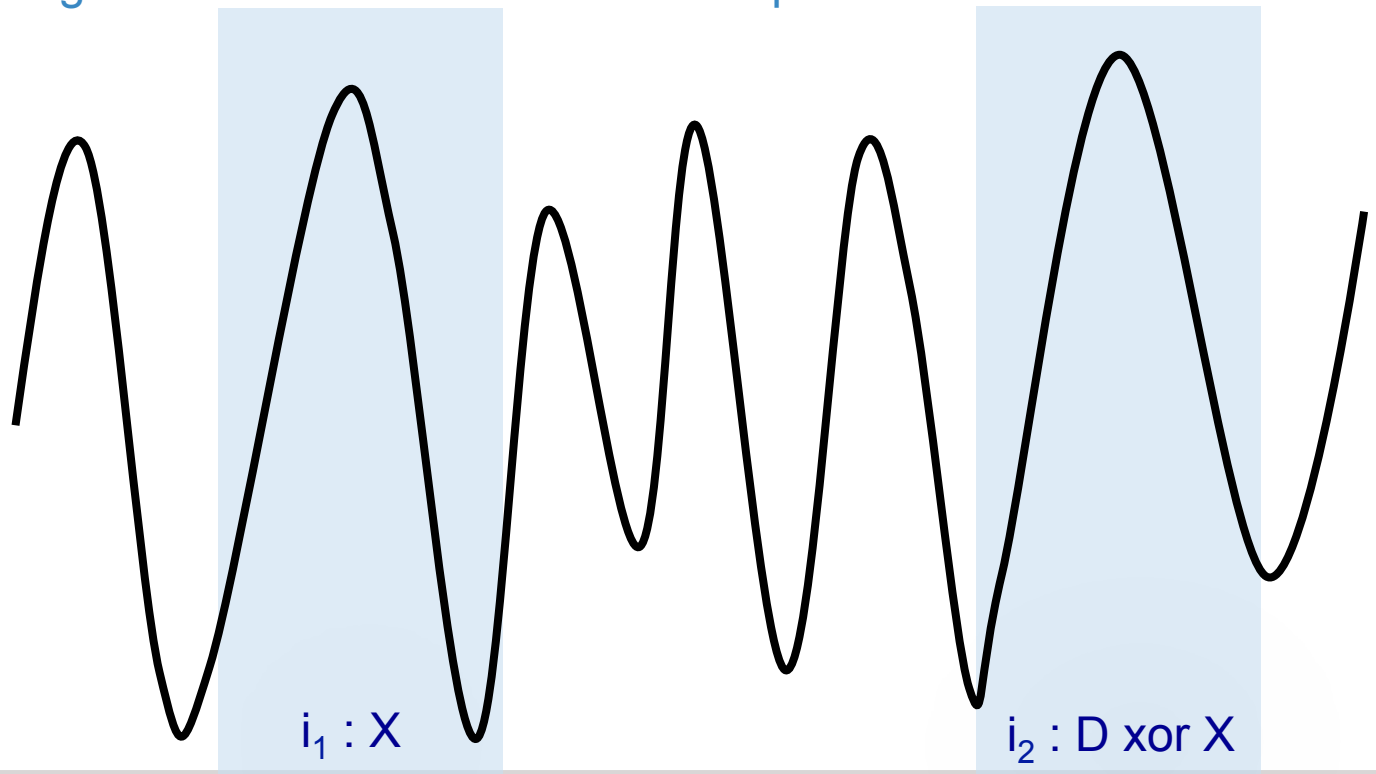
- Notebook presented performs the following tasks:
  - Collect traces from 1st order protected with Boolean masking AES executions
  - Perform first observation on traces
  - Run 1st order CPA attack : no recovery

- Let's move to esDynamic notebook

Second (Higher) Order Attacks

Find two instants on the trace where

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



“ X xor D xor X ≈ D ”

Reminder on Second order attacks

- 2nd Order Attack Principle:
  - Find two instants i_1 and i_2 on the trace(s) where the mask X 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 :
  - $T^* = |i_1 - i_2|$
  - 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*, guesses)

2nd Order CPA Attack Example
Demo on AES

- Notebook presented performs the following tasks:
  - Collect traces from 1st order protected with Boolean masking AES executions
  - Perform first observation on traces
  - Run 1st order CPA attack : no recovery
  - Localize the two instants
  - Run the second order attack: recover the key

- Let's move to esDynamic notebook (again…)

Multiplicative Masking Technique

- Not sensitive to high order attacks
- Sensitive to ZERO VALUE attacks

Power Analysis on Exponentiation

Simple Power Analysis

Code Leakage: sq. vs. mult

Data Leakage: zero values

Big Mac Attack

Distance between template and each mult

Secret is recovered in one curve!

Differential Power Analysis

Make a guess g on secret bits

Sort curves through guess to validate it

Correlation Power Analysis

Define a power model for IC

Make a guess g on secret bits used in model

Correlate curves with model to validate the right g

Template Power Analysis

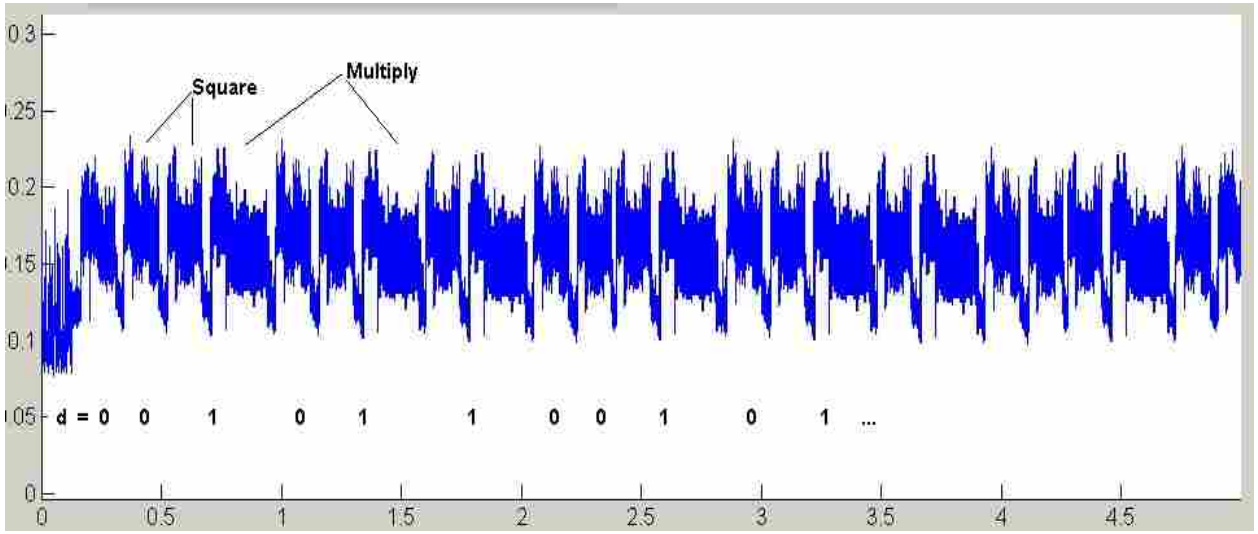
Doubling and Collision Power Analysis

Cross Correlation Power Analysis

Secret recovering needs many curves!

SPA on RSA
Original Attack from Kocher

- SPA initiale de Kocher, Jaffe et Jun
  - Carré distinguable d'une multiplication dans une exponentiation



- Just have to read the secret exponent bits on the trace

Countermeasure : Atomic Exponentiation

Algorithm 6 Atomic multiply-always 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: $\mathrm{Exp}(m,d,n) = m^d \bmod n$

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

RSA Attack Example
Demo SCA and CPA on RSA SFM 1024

- We are targeting the key recovery for a STD RSA 1024-bit

- Notebook presented performs the following tasks:
  - Collect traces from RSA executions
  - Perform first observation on traces
  - Run basic correlation analysis
  - Run improved chosen side-channel single trace attack
  - Run improved chosen side-channel multi traces correlation attack

- Let’s move to esDynamic notebook

Countermeasures: Regular Exponentiations

Algorithm 4 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: $\mathrm{Exp}(m,d,n) = m^d \bmod n$

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

Algorithm 5 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: $\quad R_{1-d_i} \leftarrow \mathrm{ModMul}(R_0, R_1, n)$
4: $\quad R_{d_i} \leftarrow \mathrm{ModSquare}(R_{d_i}, n)$
5: **return** R_0

RSA Attack on Regular Exponentiations Demo ZEMD

- We are targeting the key recovery for a STD RSA 1024-bit

- Notebook presented performs the following tasks:
  - Collect traces from RSA executions
  - Perform first observation on traces
  - Run ZEMD attack on intermediate value M^{guess} *mod N*

- Let's move to esDynamic notebook

Blinding

Message blinding $m \rightarrow m^\star$

- multiplicative: $m^\star = r^e \cdot m$ with r a λ-bit random value.
- additive: $m^\star = m + r_1 \cdot n \bmod r_2 n$ with r_1, r_2 two λ-bit random integer values

Exponent blinding $d \rightarrow d^\star$

- $d^\star = d + r \cdot \varphi(n)$ with r a λ-bit random value and $\varphi(.)$ the Euler Phi function.
- $d_1 = d - r$ and $d_2 = r$ with r a λ-bit random value
  $(m^{d_1} \bmod n) \cdot (m^{d_2} \bmod n) = m^d \bmod n$

Blinded Exponentiation

Algorithm 7 Blinded Atomic multiply-always 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: $\mathrm{Exp}(m,d,n) = m^d \bmod n$

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: $n^* \leftarrow r_2 \cdot n$
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: $d^* \leftarrow d + r_3 \cdot \varphi(n)$
8: $i \leftarrow \ell \cdot t + \lambda - 1$; $\alpha \leftarrow 0$
9: **while** $i \geq 0$ **do**
10: $\quad R_0 \leftarrow \mathrm{ModMul}(R_0, R_\alpha, n^*)$
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

More countermeasures

- Take advantages of paddings (PKCS, ISO) that render choice or knowledge on input message more difficult
- Exponent blinding can be also:
  - $d = q \times r + v$ (divide exponent by a random value r)

  Exponentiation becomes
  - $S = (m^q)^r \times m^v \bmod n$

RSA: so many attack

- SSCA Basic: code
- SSCA zero (tag) value
- DSCA (ZEMD) on Data
- DSCA (DoM) on Data – AND -- Exponent bits manipulation (Sq vs. Mult)
- DSCA on Reductions
- DSCA on CRT recombination
- Dichotomy Attack on recombination
- Profiled Attacks on secrets
- Collision (cross) correlation on Regular Algorithms (CRT and STD)
- Horizontal Attacks
- Distinguishing Squaring from Multiply operations

Squaring vs Multiplication

- The probability of the least significant bit being equal to one is:

	0	1
0	0	-
1	-	1

Square

$$\Pr(\text{bit} = 1) = \frac{1}{2}$$

	0	1
0	0	0
1	0	1

Multiply

$$\Pr(\text{bit} = 1) = \frac{1}{4}$$

- The probability of 2ndleast significant bit being equal to one is:

	00	01	10	11
00	0	-	-	-
01	-	0	-	-
10	-	-	0	-
11	-	-	-	0

Square

$$\Pr(\text{bit} = 1) = 0$$

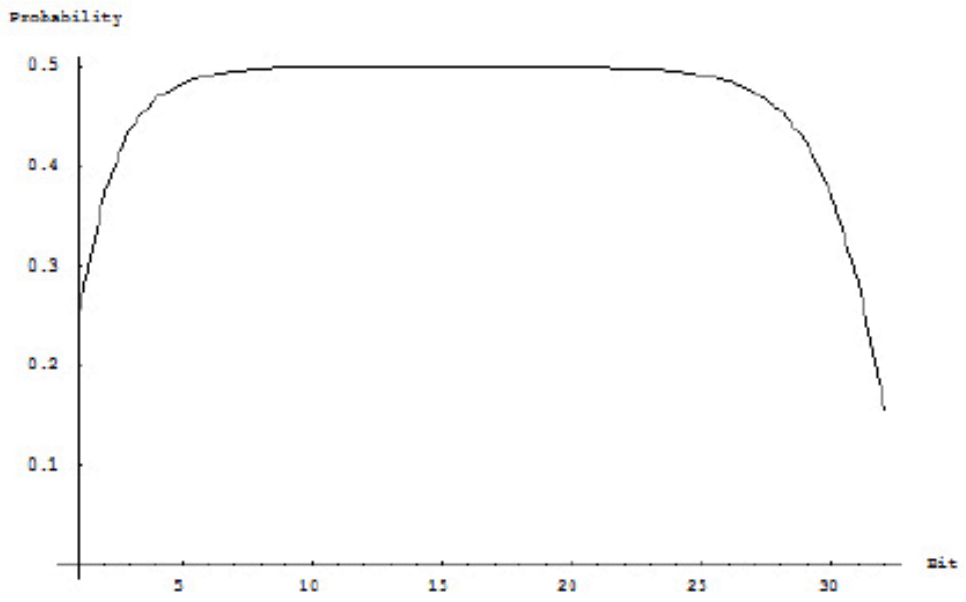
	00	01	10	11
00	0	0	0	0
01	0	0	1	1
10	0	1	0	1
11	0	1	1	0

Multiply

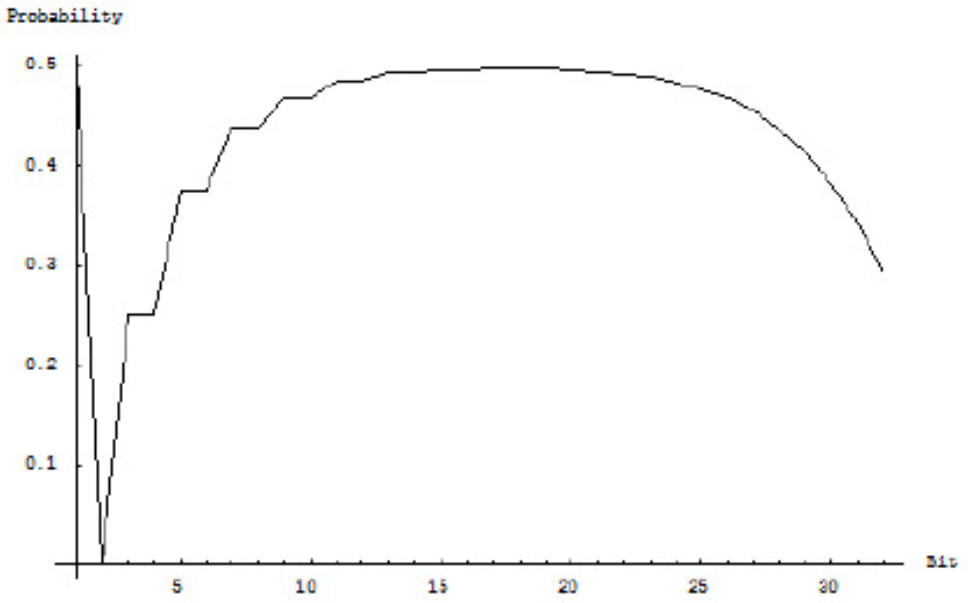
$$\Pr(\text{bit} = 1) = \frac{3}{8}$$

Probability of individual bits

- The probability each bit equals one in a 32-bit word produced by a multiplication of two random uniformly distributed 16-bit words.



- The probability each bit in a 32-bit word produced by a squaring of random uniformly distributed 16-bit word.



Fault Attacks
and
Countermeasures

Fault Attacks

- On Cryptographic Implementations
  - Differential Fault Analysis (DFA)
  - Collision Fault Analysis (CFA)
  - Ineffective Fault Analysis (IFA) or Safe Errors
  - Signature verification

- On Operating System
  - PIN / MAC verification
  - Key loadings
  - Data transfers
  - …

Differential Fault Analysis

- **RSA- CRT :** *Boneh, DeMillo & Lipton*

  one faulty ciphertext allows to retrieve the private secrete key !!!

- **DES :** *Biham and Shamir*

  two faulty ciphertexts allow to retrieve the whole key.

- **AES :** *Piret and Quisquater*

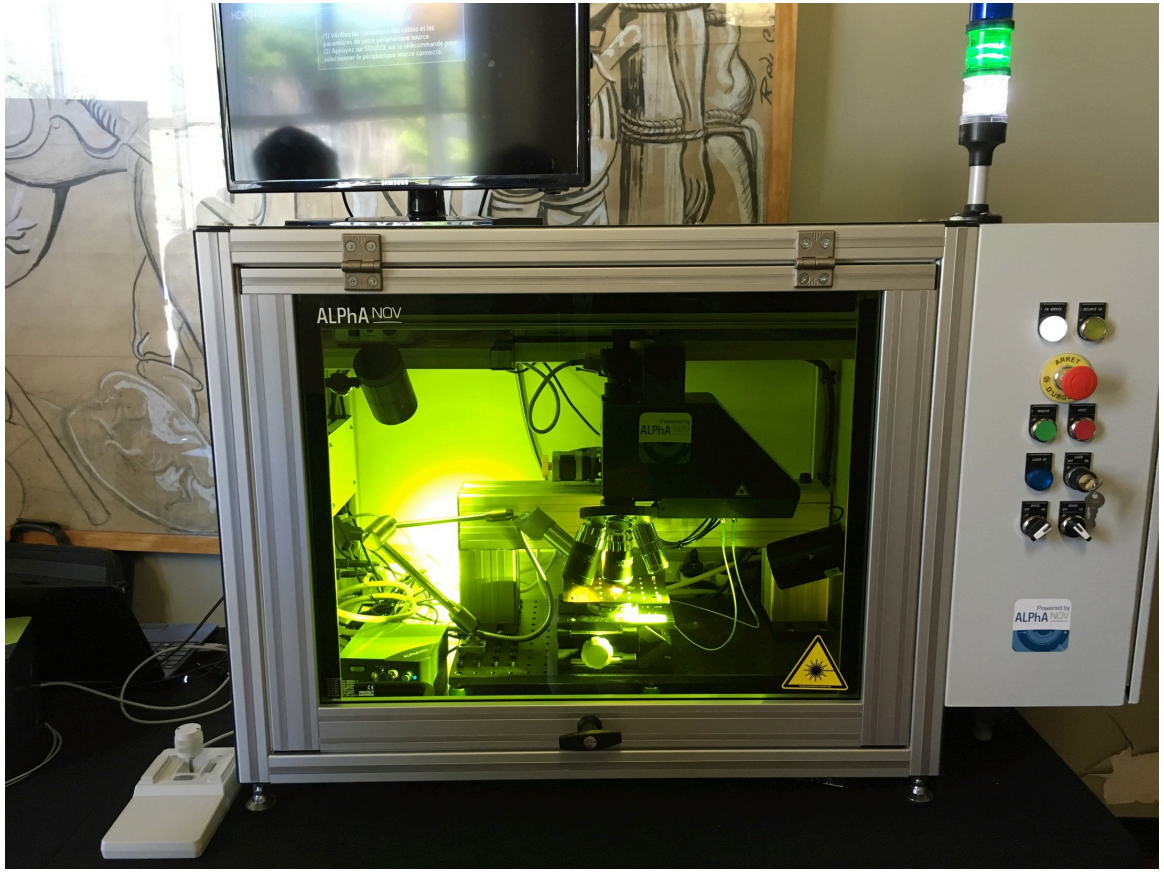
  two faulty ciphertexts allow to retrieve the whole key.

DFA also applies to other algorithms

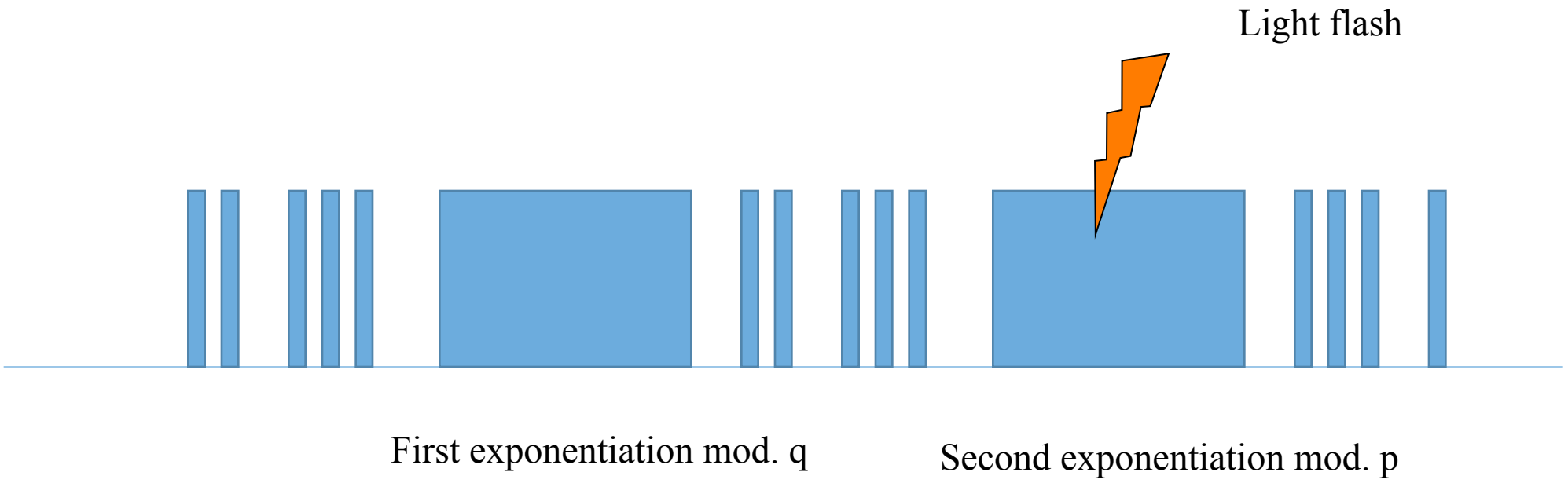
Equipment

- Not expensive
  - Générateur de Glitches
  - Flash (appareil photo)

- Couteux
  - Laser (focalisé à travers un système d'optiques)



DFA on CRT-RSA



DFA on CRT-RSA

Secret key : (d, p, q) Public key : (e, N = p x q)

M : message

S : signature, S' : faulty signature

➔ If error occurred during computation of Sq = S mod q :

$gcd\ (S\text{-}S', N) = q$

➔ Improvement published by Lenstra :

$gcd\ (M\text{-}S'^{e}, N) = q$

Countermeasures:

- Verify the signature with the public key

- Perform twice and compare

Safe Errors
Square and Multiply Always

Algorithm 4.1.4 Exponentiation Square&Multiply Always $ModExpAlways(M, d, N)$

INPUT: N le modulus public, M et d de n bits
OUTPUT: $m^d \mod N$

Step 1. $K_1 \longleftarrow 1$

Step 2. $K_0 \longleftarrow 1$

Step 3. Pour i de $n-1$ à 0

Step 4. $K_1 \leftarrow K^2 \mod N$

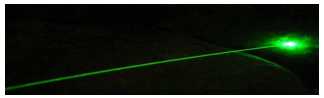
Step 5. $K_{d_i} \leftarrow K_1 \times M \mod N$

Step 6. Fin pour

Step 7. return (K_1)

Generic Fault Attacks

- Perturb OS functions executing an applet during
  - A PIN verification
  - Access read to memory
  - Example:



Authentication

OK good password

KO when bad password

Authentication

Always say OK

- It is then mandatory to protect OS, applications from fault attacks
- Nowadays we can perform multiple very precise fault injections in a same execution

Combined Attacks
and
Countermeasures

Passive and Active Combined Attacks (PACA)

PACA on RSA

Compute an RSA signature : $s = m^d \bmod n$

Countermeasures : Randomization Scheme / Side-Channel Atomicity / Fault Protection

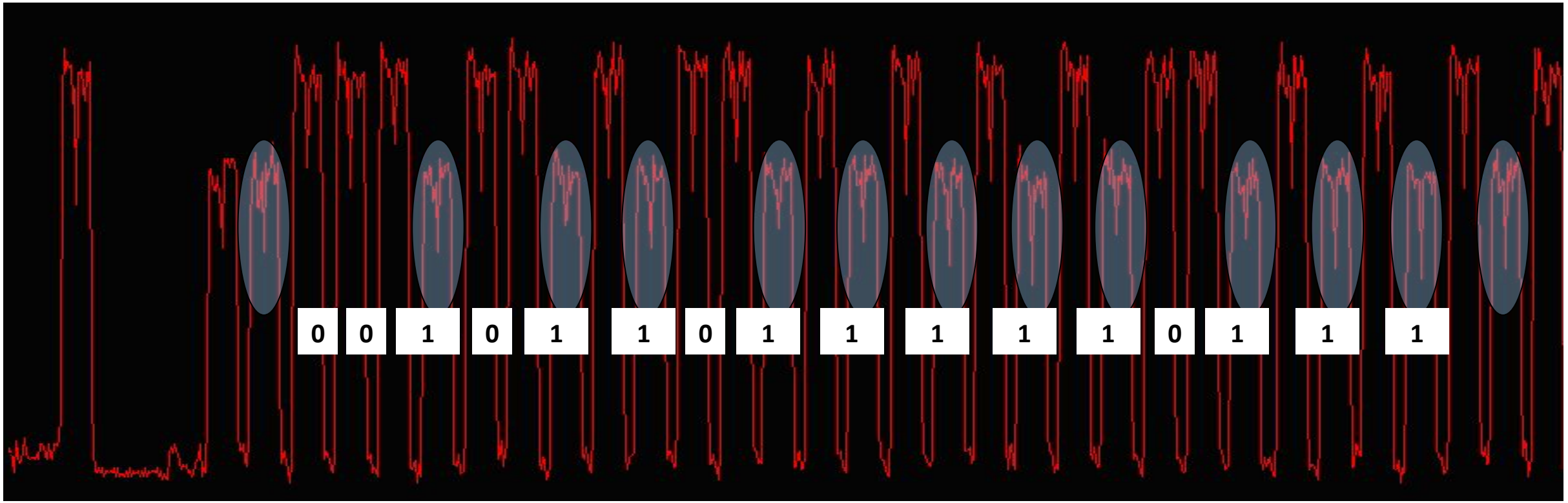
Implementation :

- **Get *r1*, *r2* two non zero small random values**
- ***R0 = 1 + r1.n***
- ***R1 = m + r1.n* mod *r2.n***
- ***k = 0***
- **for *i* from *k-1* to *0* do**
  - ***R0 = R0.Rk* mod *r2.n***
  - ***k = k* xor *di***
  - ***i = i* – not(*k*)**
- ***s = R0* mod *n***
- $m_{redundancy} = s^e \bmod n$
- **if $m \neq m_{redundancy}$ then fault detected !**
- **else return (*s*)**

… this operation is perturbed and gives to R1 a low Hamming Weight ?

SPA Leaking Exponentiation for Tagged Data

SPA leakage with only 1 successful fault !



Mult **operations by R1 are now revealed**

Verification happens too late!

Basic PACA on CM "One DES amongst n "

- Product contains a DES engine non DPA resistant

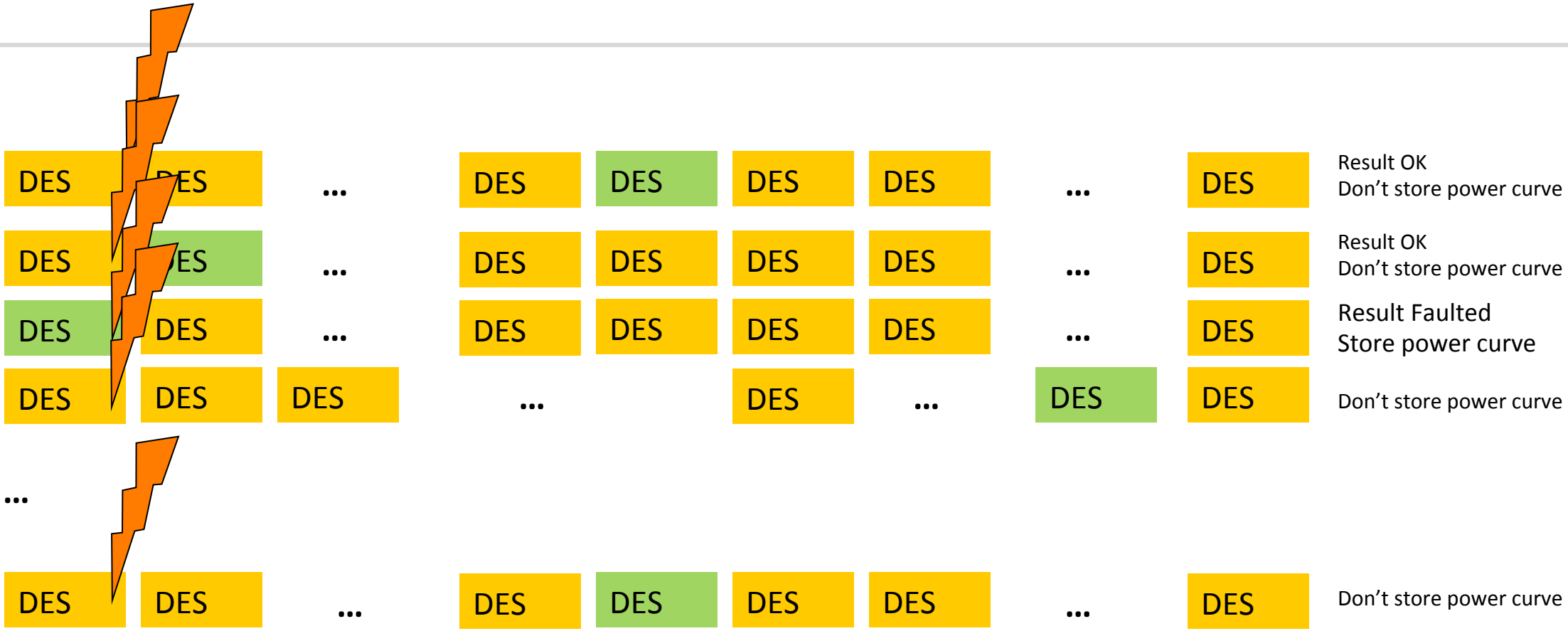
- Solution:

  Processing each DES computation amongst
  n fake DES computations in random order

  → DES becomes resistant.

- PACA attack threatens this countermeasure …

Basic PACA on CM ‘’One DES among N



→ Process DPA/CPA on the curves stored to recover key K

Necessites n more times execution with fault injection than non protected implementation

PACA on CM ‘’One DES amongst n ”

- Countermeasure:

  Processing each DES/DES-1 computation amongst
  n fake DES/DES-1 computations in random order

- Seems ok …

IDEA for Countermeasures

- Side Channel and Fault attacks cryptographic calculations must be protected but countermeasures must be built together.

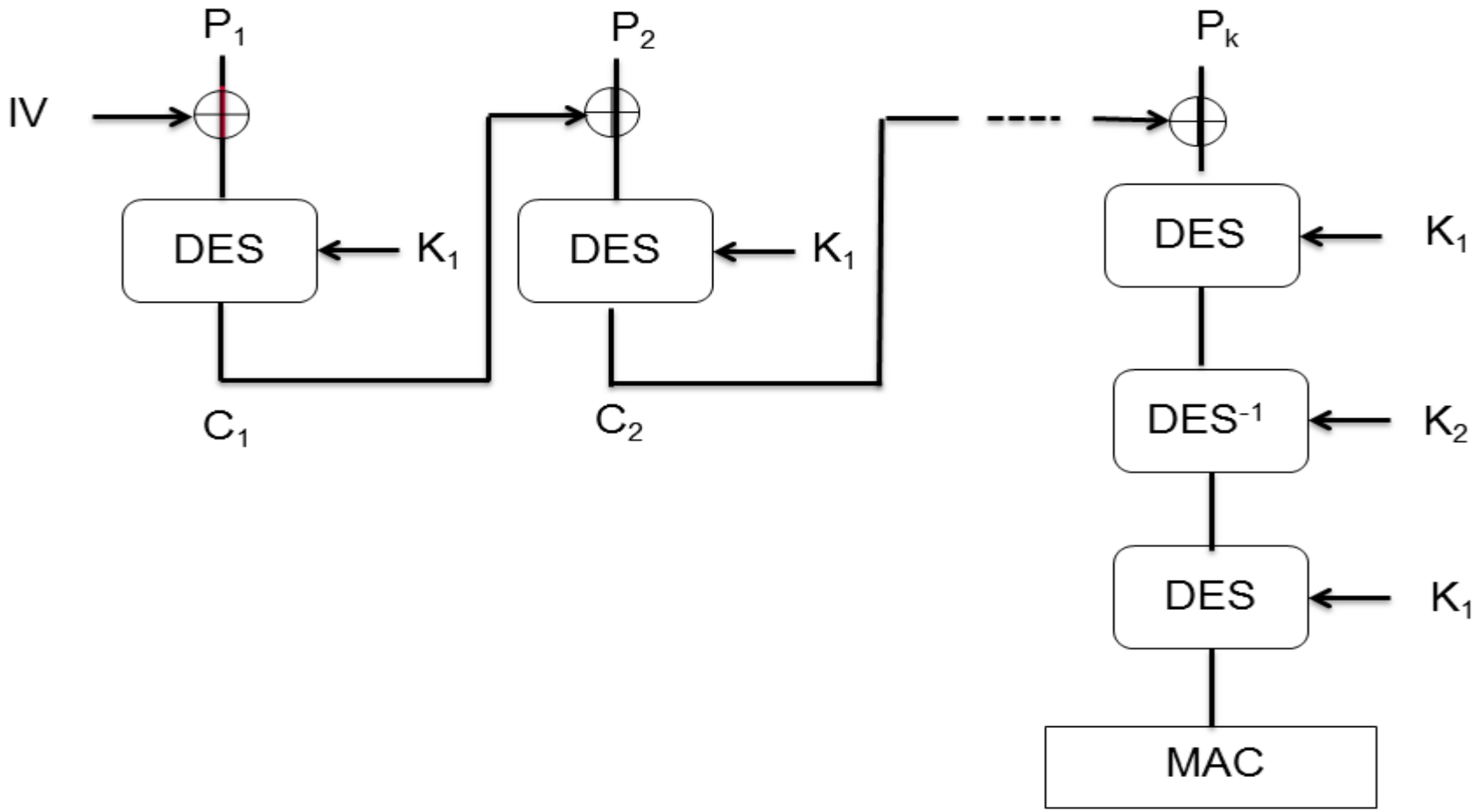
- PACA: … so many attack possibilities …

Sometimes there is still space somewhere else for …

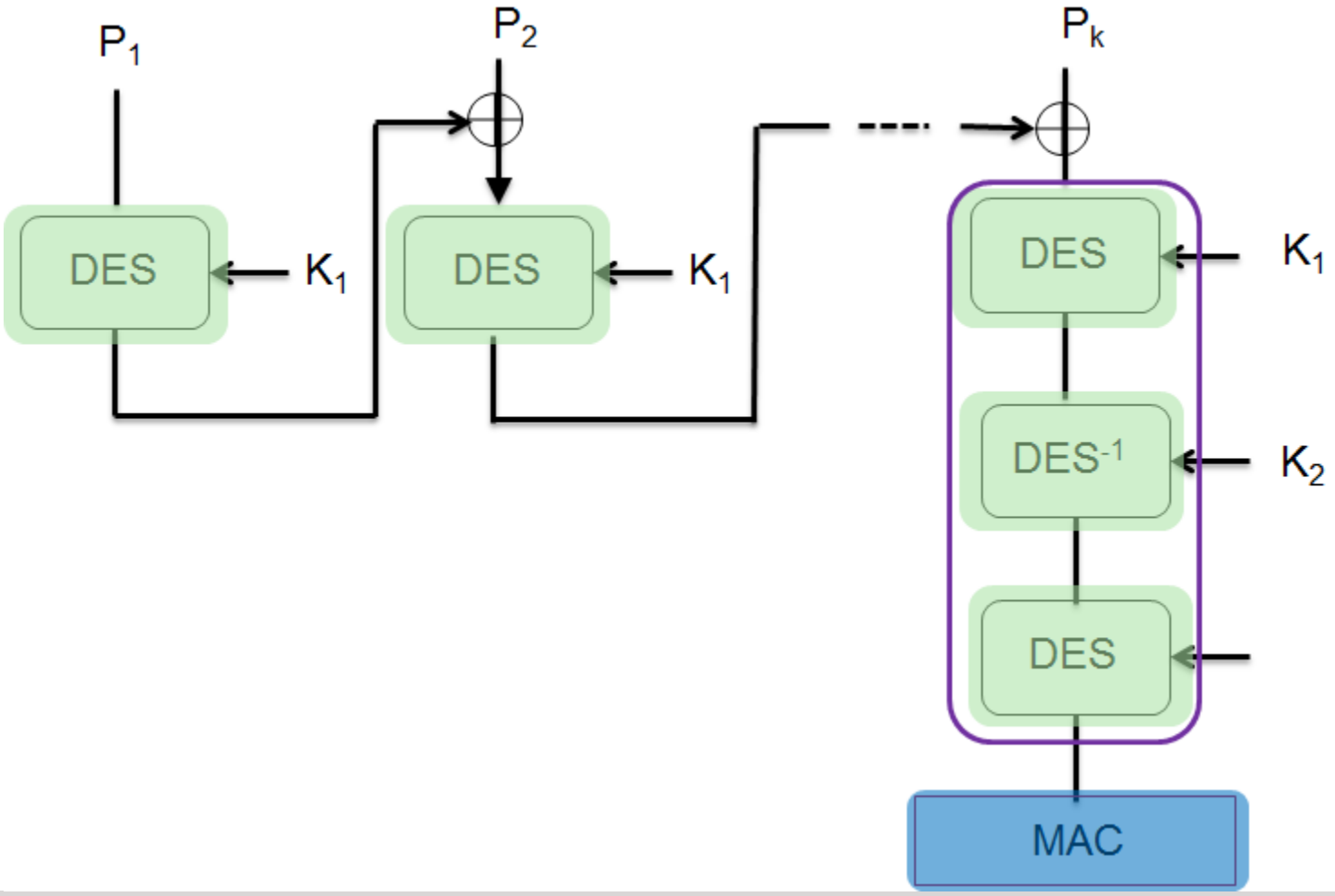
Side-Channel Attack on the MAC Algo3

- 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

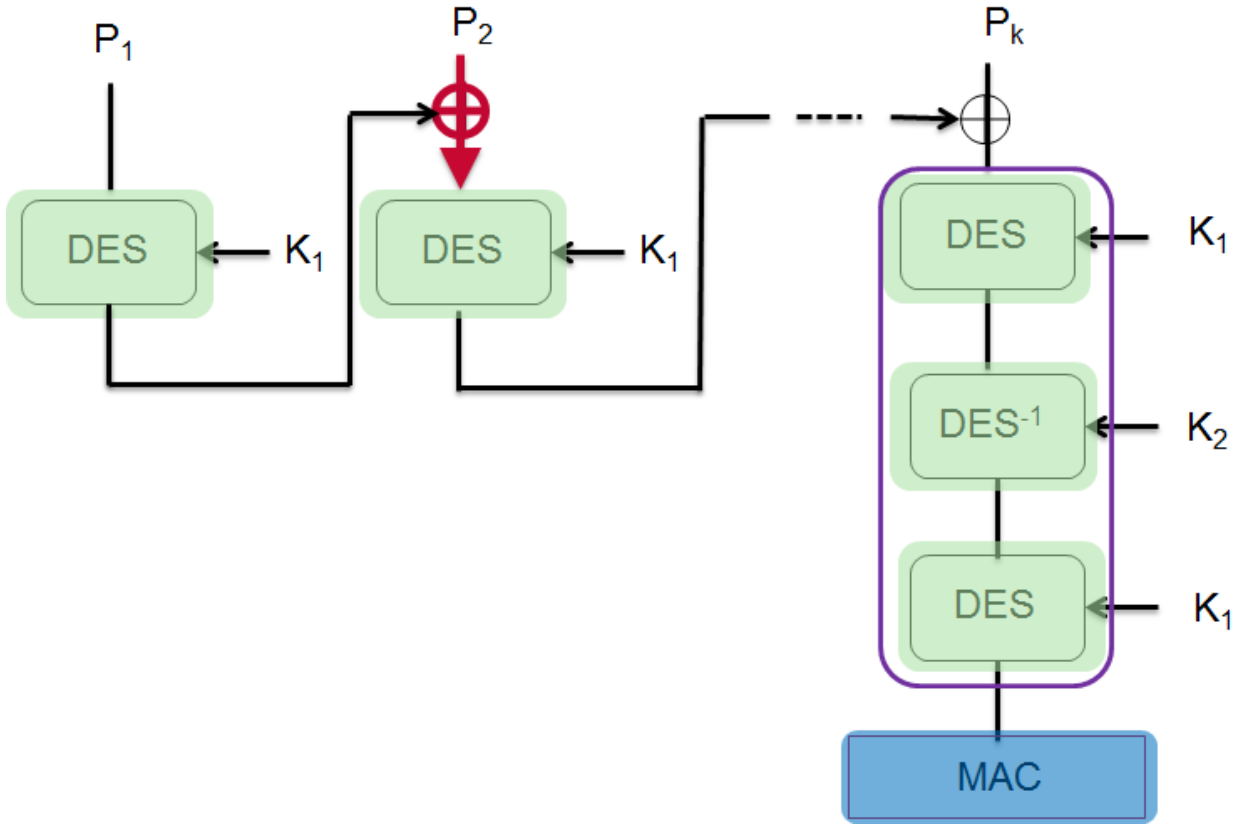


MAC Algo3

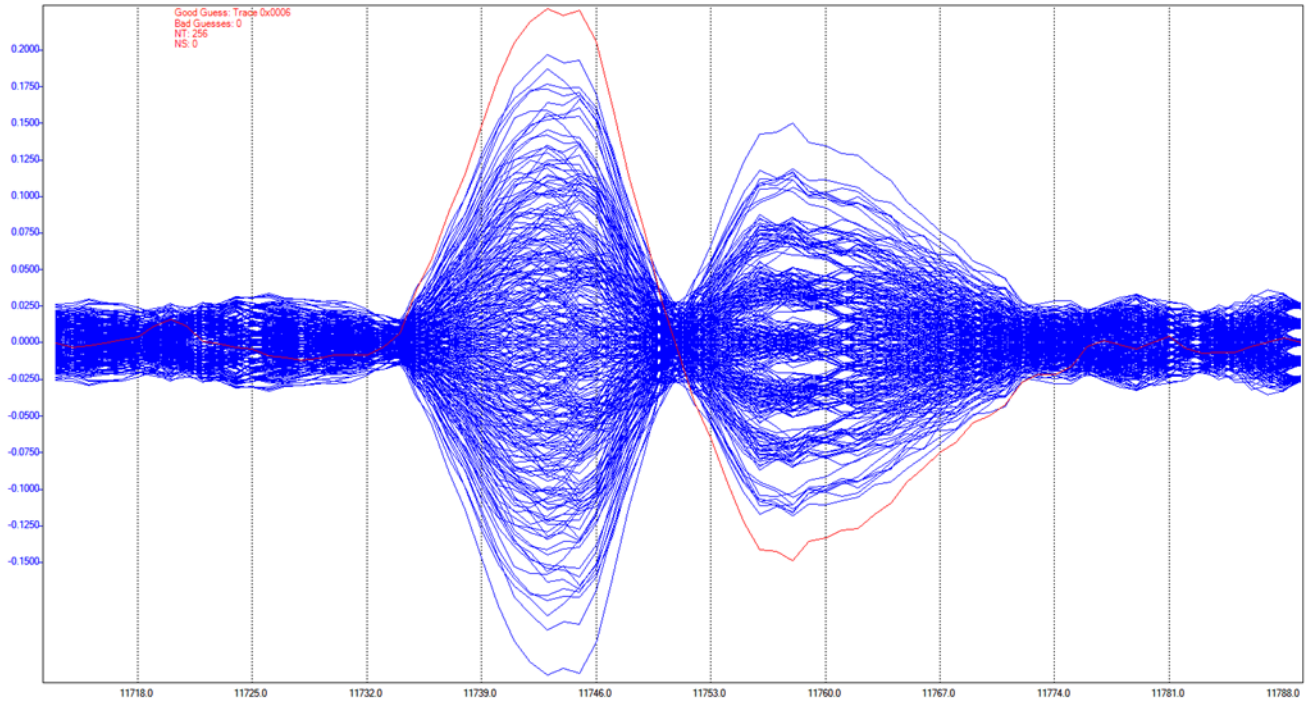


Side-Channel Attack on MAC Algo3

- 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 $C_0 \dots C_n$
- 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.



Attack Result



White-Box Cryptography

New area for side-channel gamers

Security Evaluation and Certification

- *Expertise is a key* -

Pourquoi Evaluer et Certifier ?

- Le certificat est obligatoire pour vendre un produit de sécurité aux clients
- Permet de garantir au client (banques, opérateurs, état) que ses biens et celui des utilisateurs (ses clients) sont protégés
- Volonté des organismes et des états de lutter contre la fraude, le piratage
- Le schéma contribue à améliorer de manière continue la sécurité des produits

Gestion du risque

Selon vous:

- Comment distinguer
  - Cette attaque est grave ?
  - Cette attaque n’est pas gênante ?

- Comment trancher ?

Risk Management

- **Tout produit peut être vulnérabilisé voire cassé**
  - Si on y met les moyens … En investissant des millions on cassera toujours un produit
  - Exemple: des mois (années) de reverse, de FIB …
  - Un produit a une durée de vie limitée

- Comment évaluer de manière basique le risque ?

Comparer le coût de l'attaque

AVEC

Le gain que l'attaquant peut en retirer

- Example:

1.000.000 € pour retrouver ma propre clef de carte bancaire

Pas intéressant pour l'attaquant → Pas dangereux !

1.000.000 € pour retrouver la clef maître d'un système de PayTV

Intéressant pour l'attaquant → Dangereux !

Comment donner confiance ?

Votre avis ?

Vous feriez comment ?

Quelques pistes pour donner confiance

- Grace à votre réputation et reconnaissance industrielle
- Grace à votre savoir faire
- Grace aux fonctions sécuritaires offertes par votre produit
- Grace à la qualité de leur implémentation
- Grace à la preuve de leur efficacité
- En fournissant à l'utilisateur tous les renseignements pour une bonne utilisation sécuritaire (environnement d'utilisation / utilisation d'usage)
- Grace à une analyse de vulnérabilité approfondie prouvant que tous les chemins d'attaque ont été couverts
- Grace à l'analyse experte d'une tierce partie de confiance

Different Schemes

- **Standards propriétaires**
  - Bancaires
    - EMVCo: American Express, Mastercard, VISA, JCB
    - MasterCard CAST
    - VISA VCSP
  - **Obligatoires pour vendre le produit**
  - **Donne confiance au client**
  - Coûteux pour le développeur
  - Requiert aujourd'hui un haut niveau d'expertise pour les développeurs
  - A chaque produit on rejoue … ''*New product … same player play again …*''
- **Evaluation = référence mondiale:**
  - ITSEC
  - **ISO 15408 = Critères communs**
    - Différents niveaux d'assurance possible pour une évaluation (EAL1 à EAL7+)
    - Guides et recommandation standardisés: Protection Profiles, JIL,…
- Des tierces parties de confiance: accrédités par les états et les banquiers
  - **Laboratoires d'évaluation**s
  - **Organismes d'états**: ANSSI, BSI …

Certification Process

Sponsor

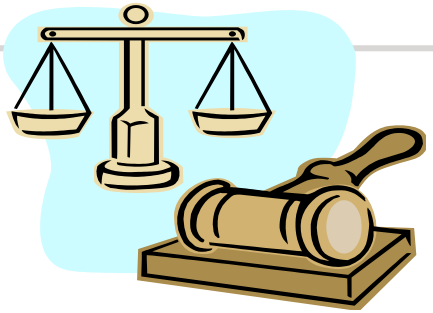
Developer

Certificate

Reports

Deliveries, evidences

Reports



Certification body

Reports

Independent Lab

Definition

- La sécurité c'est:

  Protéger des biens de menaces

- **Biens** = les objets/données à protéger
  - Biens peuvent avoir plusieurs formes : Matériel (argent, objets de valeurs …), Produits, Disponibilité d'un service, Informations confidentielles(Propriété Intellectuelle, clefs, données personnelles, identifiants/ authentifiants …)

- **Menace**: un attaquant présumé
  - Individu (pirate, vilain pas bien), organisation (mafia, terroristes) …

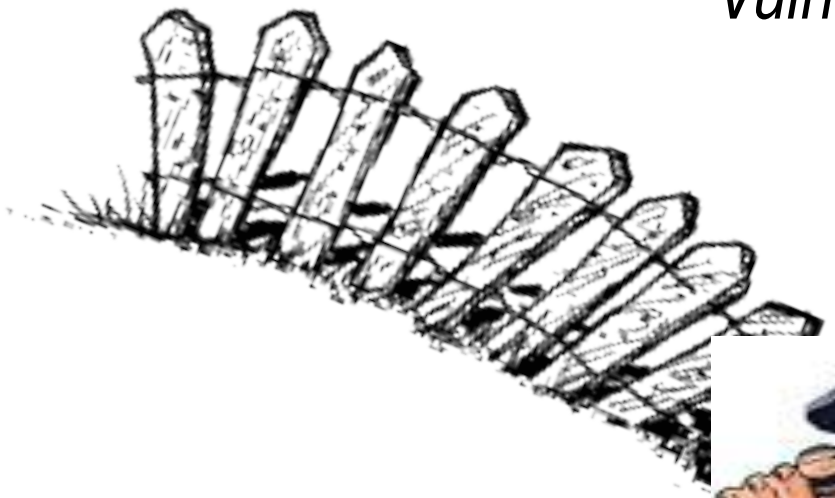
- **Vulnérabilité**: une faiblesse dans le système ou le produit qui peut être utilisé pour casser les protections, dans un environnement et des conditions précises

- **Objectifs**: comment essayer de protéger un système contre des menaces précisément listées et identifiées.

Attack Profiling

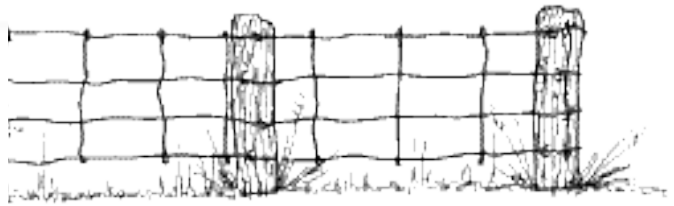
Menace

Mesures de Protection …



Vulnerabilité

Biens



Attaque

Contremesure

Histoire des Standards

- Début 80's : TCSEC (US "Orange Book")
- 1991 : ITSEC (Europe)
- 1993 : CTCSEC (Canada) & FC (USA)

- Critères Communs
  - Janvier 1996 : v1.0
  - Mai 1998 : v2.0 applicable
  - Juin 1999 : ISO-15408
  - Août 1999 : v2.1
  - Septembre 2006 v3.1 (obligatoire à partir de Mars 2008)

- Outils pour l'évaluation
  - Joint Interpretation Library (JIL)
  - Common Evaluation Methodology (CEM) v3.1

Common Criteria Terminology

- **Target of Evaluation (TOE): cible d'évaluation** = décrit le produit qui va être évalué et le niveau d'assurance visé.
- **Protection Profile (PP)** : définit les objectifs et les exigences de sécurité pour un type (une catégorie) de produit (circuit intégré de carte à puce, passeport, carte d'identité …) indépendamment du choix d'implémentation. Utilisé lors de la plupart des évaluations.
- **Security Target (ST):** la **cible de sécurité** décrit en fonction de l'implémentation faite de la TOE les besoins et les exigences de sécurité pour une TOE spécifique en se référant au PP correspondant au type de produit de la TOE.
- **Security Functional Requirement (SFR)**: les comportements de sécurités standard proposés par les CC pour atteindre les objectifs et que la TOE doit garantir.
- **Security Function (SF)**: ensemble des mécanismes de sécurité (fonctions) fournie par la TOE pour sa protection et atteindre les objectifs de sécurité. Une SF implémente une partie d'une SFR.

Security Target

- La cible de sécurité est le document de base pour l'évaluation

- Elle décrit le produit et le contexte sécuritaire:
  - Les biens à protéger et donc à considérer
  - Le cycle de vie du produit
  - Les menaces
  - Les objectifs de sécurité
  - Les exigences de sécurité
  - Liste des fonctions de sécurité
  - Mesures mises en oeuvres pour garantir la sécurité des biens
  - Niveaux d'assurances visées

- La cible de sécurité doit suivre les besoins définies par la CEM et est évaluée

Les Différentes Classes

- APE – Protection Profile Evaluation
- ASE - Security Target evaluation

- ADV - Development
- AGD - Guidance documents
- ALC - Life Cycle support
- ATE - Tests
- AVA - Vulnerability Assessment

- ACO - Composition

Evaluation Assurance Levels

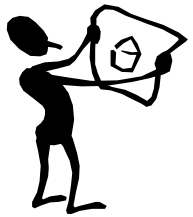
Assurance class	Assurance Family	Assurance Components by Evaluation Assurance Level						
		EAL1	EAL2	EAL3	EAL4	EAL5	EAL6	EAL7
Development	ADV_ARC		1	1	1	1	1	1
	ADV_FSP	1	2	3	4	5	5	6
	ADV_IMP				1	1	2	2
	ADV_INT					2	3	3
	ADV_SPM						1	1
	ADV_TDS		1	2	3	4	5	6
Guidance documents	AGD_OPE	1	1	1	1	1	1	1
	AGD_PRE	1	1	1	1	1	1	1
Life-cycle support	ALC_CMC	1	2	3	4	4	5	5
	ALC_CMS	1	2	3	4	5	5	5
	ALC_DEL		1	1	1	1	1	1
	ALC_DVS			1	1	1	2	2
	ALC_FLR							
	ALC_LCD			1	1	1	1	2
	ALC_TAT				1	2	3	3
Security Target evaluation	ASE_CCL	1	1	1	1	1	1	1
	ASE_ECD	1	1	1	1	1	1	1
	ASE_INT	1	1	1	1	1	1	1
	ASE_OBJ	1	2	2	2	2	2	2
	ASE_REQ	1	2	2	2	2	2	2
	ASE_SPD		1	1	1	1	1	1
	ASE_TSS	1	1	1	1	1	1	1
Tests	ATE_COV		1	2	2	2	3	3
	ATE_DPT			1	1	3	3	4
	ATE_FUN		1	1	1	1	2	2
	ATE_IND	1	2	2	2	2	2	3
Vulnerability assessment	AVA_VAN	1	2	2	3	4	5	5

Table 1 - Evaluation assurance level summary

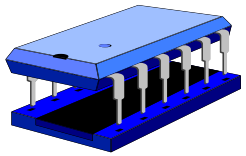
- Classiquement
- EAL5+ pour un circuit intégré
  → EAL6+
- EAL4+ pour une évaluation composite (IC + Software .. Produit utilisateur)
- Tous types de produits concernés par les CC:
  - Cartes
  - Systèmes: Windows
  - Avions: Airbus …
  - Armée: Sous marins, frégates (ITSEC)

Les Etapes

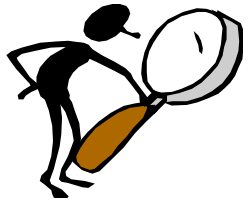
- **Objectif:** imaginez les actions successives nécessaires qu'un attaquant devra réaliser, avant que son attaque soit réussie et pour la réaliser au mieux.



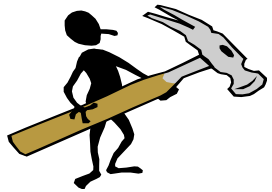
Acquérir du savoir sur la TOE



Acquérir TOE: échantillons



Observer la TOE



Utiliser des outils

Interpréter les résultats

Et on continue si besoin …

Critères pour une attaque

- A chaque étape l'évaluateur considère 5 critères conformément à la CEM:
  - Temps nécessaire à l'attaque
  - Niveau d'expertise requis
  - Niveau de connaissance de la TOE
  - Nombres d'échantillons nécessaires
  - Equipement requis
- Pour chacun des facteurs une valeur (points) est fournie par un tableau
- La note (Attack Potential) est la somme des ces valeurs

Notation d'une attaque

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

- L'évaluateur va noter de cette manière chacune des attaques
- Le niveau global de sécurité du produit sera la note la plus faible parmi la liste des attaques
- Produit VAN.5: signifie toutes les attaques possibles ont été notées VAN.5
  → Produit considéré résistant (does not mean unbreakable ! ☺)

Notation: DPA sur un TDES

- TDES dans une carte bancaire
- CEMA réussie.

Factors	Identification	Exploitation
Elapsed time	2 (less than 1 week)	4 (less than 1 week)
Expertise	5 (Expert)	2 (Proficient)
Knowledge of the TOE	2 (Restricted)	0 (None / Public)
Access to the TOE	0 (less than 10 samples)	0 (less than 10 samples)
Equipment	3 (specialized)	4 (specialized)
Subtotal	**12**	**10**
TOTAL :	**22**	

- Le produit est considéré résistant to an ***Enhanced Basic*** *potential attack* soit VAN.3.