Passive and Active Combined Attacks

Combining Fault Attacks and Side Channel Analysis

Frédéric AMIEL[1]	Benoît FEIX[1]	Louis MARCEL	Karine VILLEGAS
Texas Instruments	Inside Contactless	K2Crypt	Gemalto

Communicated by:
Michael TUNSTALL
University College Cork, Ireland

[1] *: part of this work done when with GEMALTO*

Outline

- **Introduction**
  - Passive Attack: Power Analysis.
  - Active Attack: Fault Attack.
  - State of the art.
- **PACA – Combined Attacks**
  - Principle.
  - Application to RSA.
- **Countermeasure**
  - Detect and Derive.
- **Conclusion / Future Works**

Introduction

- *Two types of countermeasures*:
  - ***SCA***: Side Channel Attacks (DPA, SPA, Template-Analysis, Timing Attacks, …).
  - ***FA***: Fault Attacks (Invasive, Transient, …).

- *Problem*: Each protection is usually focused to protect against ***SCA*** **or** ***FA.***

- *Idea*: Combine both kind of attacks to defeat a classical set of countermeasures.
  - ***PACA***: Passive and Active Combined Attacks.

Introduction: *Previous Work*

- **“Optically Enhanced Position-Locked Power Analysis” by Sergei Skorobogatov (CHES’06).**

  Use a focused laser to enhance the power consumption of a sensitive part in a chip.

  - Active Attack : Optical Enhancement of Power Consumption.
  - Passive Attack : DPA, …

- **PACA : A ‘Fashionable’ Attack ?**

  Presented this morning :

  “How can we overcome **both Side Channel Analysis and Fault Attacks** on RSA-CRT ?” by Chong Hee Kim and Jean-Jacques Quisquater.

Passive and Combine Attacks

In order to make the attack realistic…

…let’s take the take the simplest of both worlds.

Let's choose our Passive Attack…

- Simplest SCA… Simple Power Analysis !!!

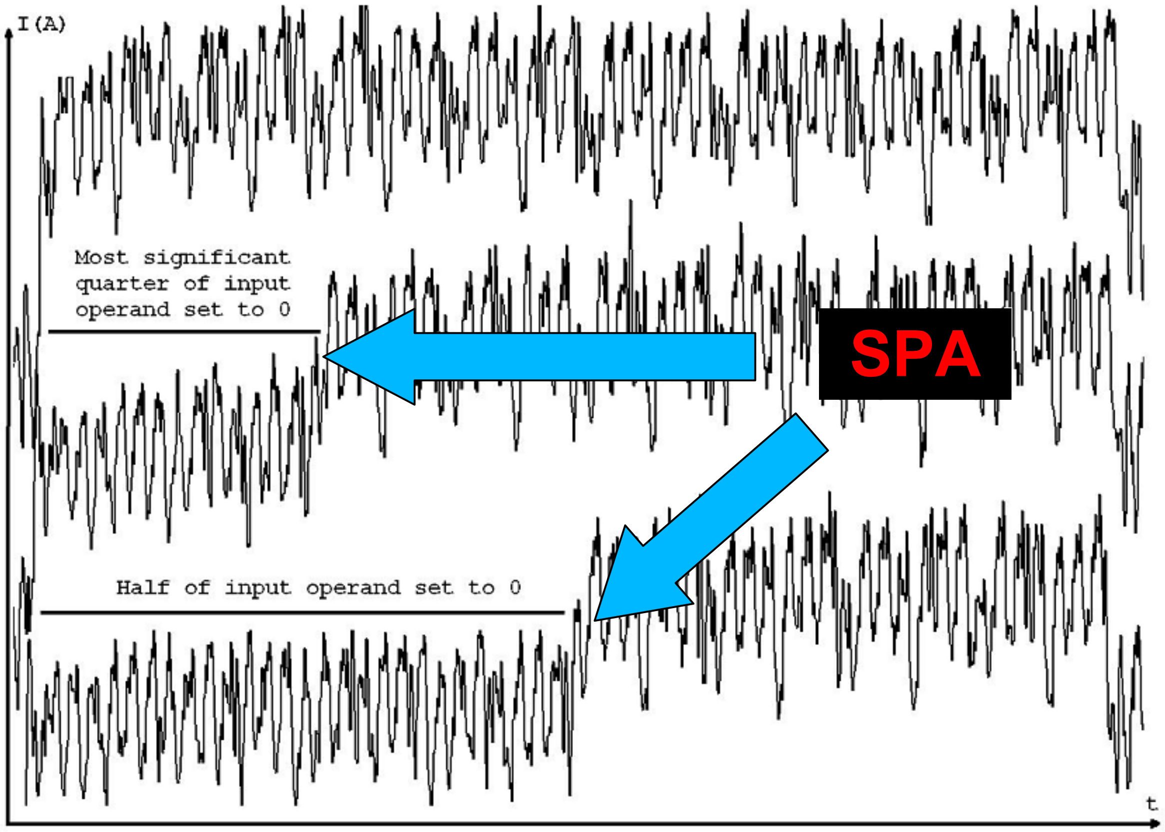
- Particularly adapted to RSA: Need only one curve on naïve implementation to retrieve the private key value.

- Without randomization techniques, well-chosen message can reveal the nature of operation:

Square or Multiply

… which give the value of the private key.

SPA on Multiplication Operation



Square or Mult

Random values for Operands

Mult A x B

Quarter of A set to 0

Mult A x B

Half of A set to 0

Fault Model

- Modify a part of an input operand.
  - **Corrupt of a word during a memory transfer (typically set to zero).**

- Corrupt address pointer of a multiplicand.
  - **Modified pointer value could point to an uninitialized RAM aera (also typically set to zero).**

- etc.

Is this RSA implementation protected ?

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)**

It seems…

but what would happen if …

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

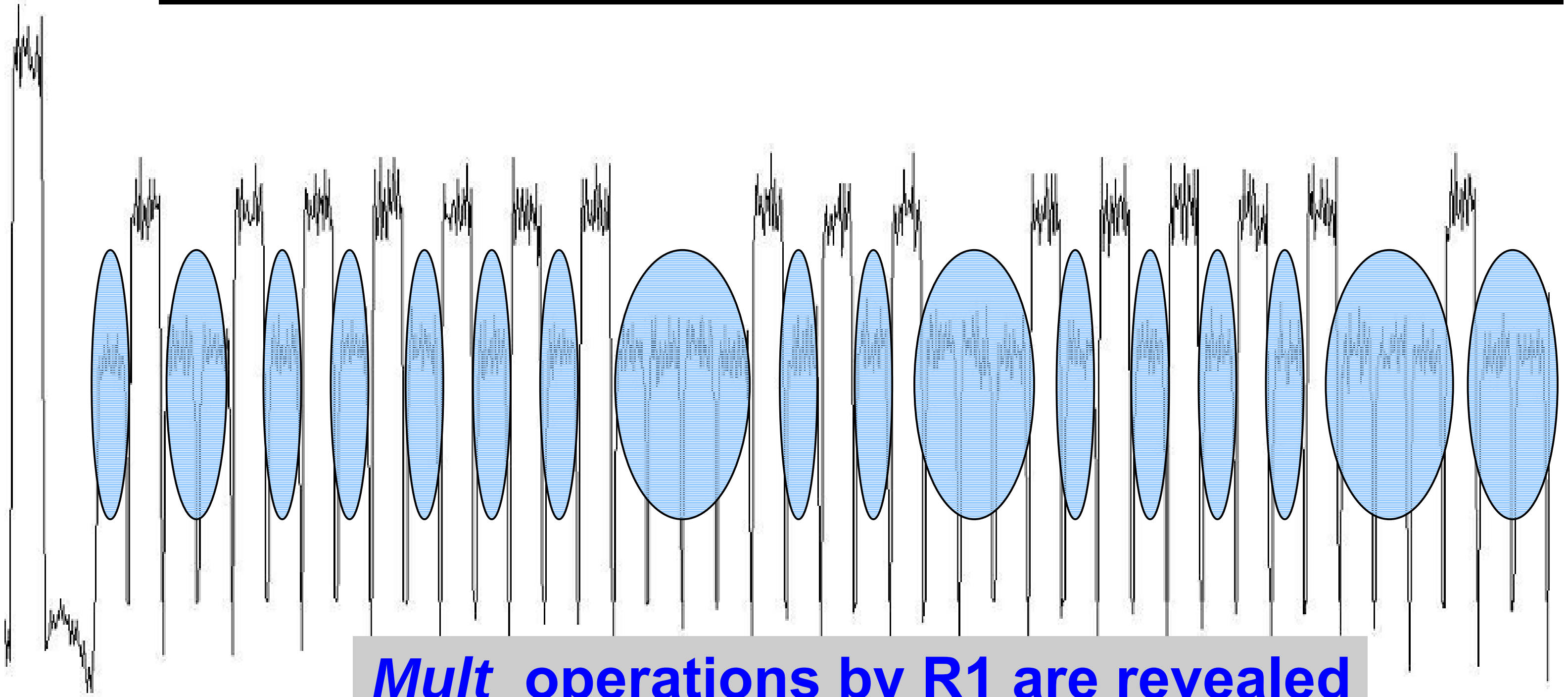
Countermeasures : Randomization Scheme / Side-Channel Atomicity / Fault Protection

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 leakage with only 1 successful fault !



Mult **operations by R1 are revealed**

What about the FA countermeasures ?

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

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

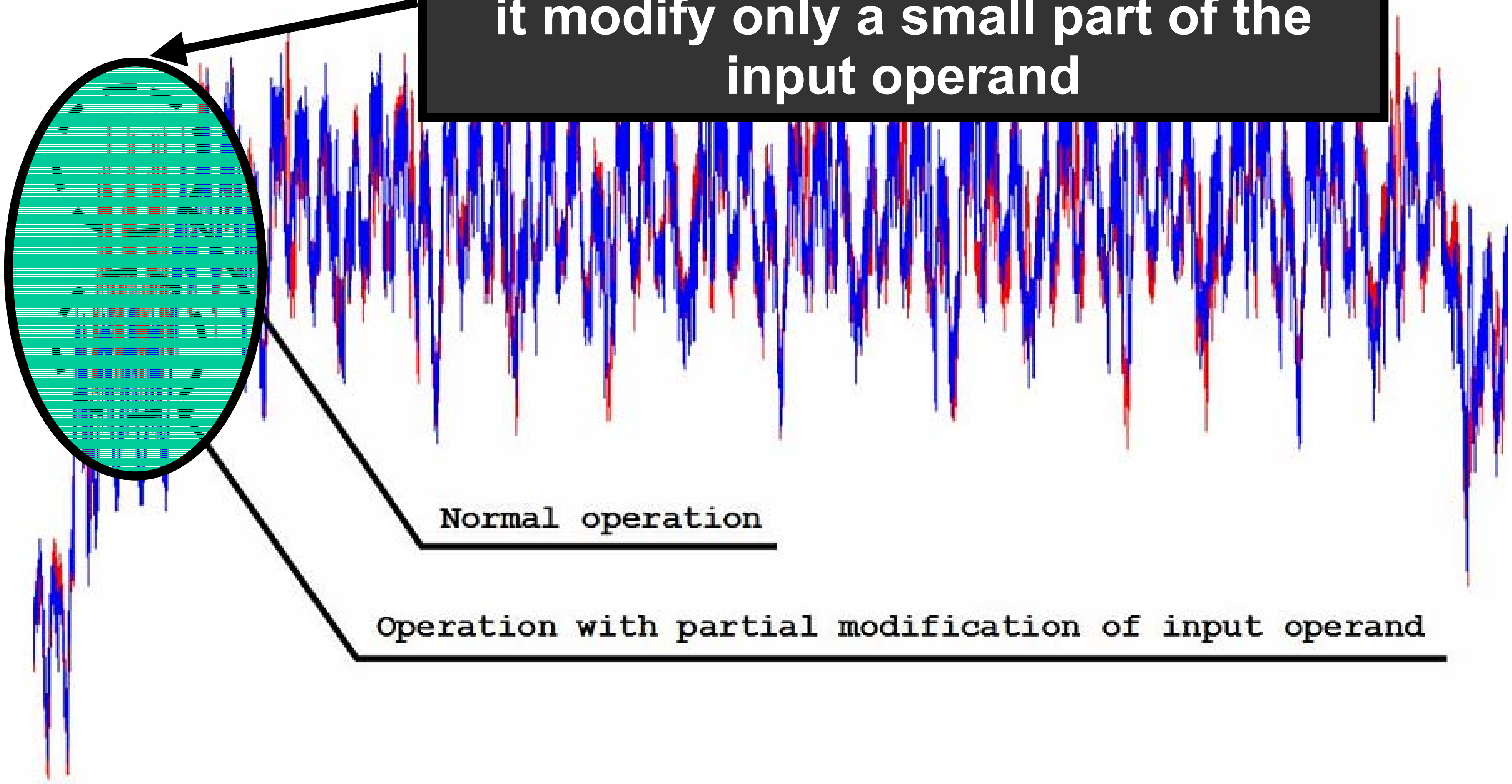
Implementation :

- 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)

Problem : The fault will be detected but too late, the side channel leakage can be exploited !

Side Channel Leakage

The fault could be exploited even if it modify only a small part of the input operand

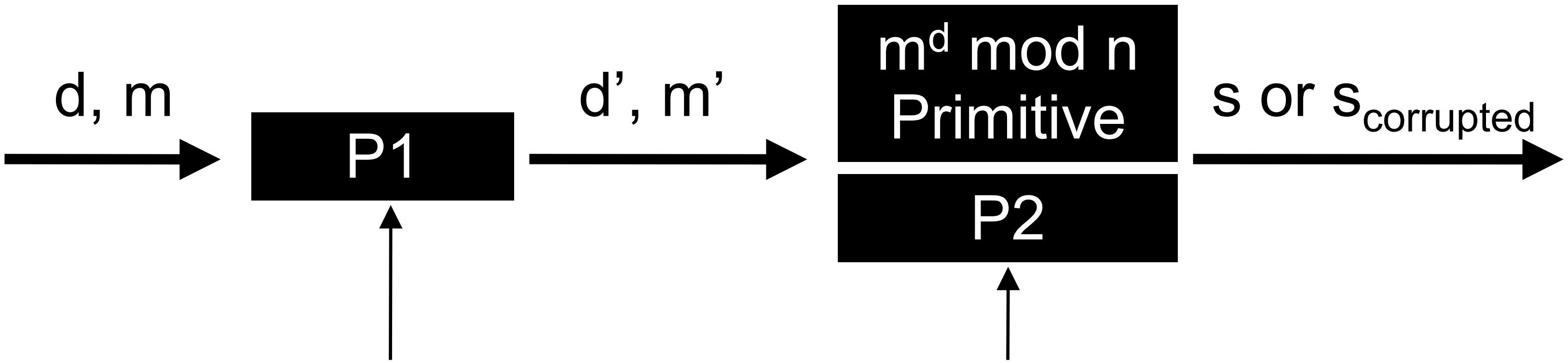


Possible Countermeasures

- **Detect And Derive !**
- **Corrupt secret data in an non-invertible way before the sensitive process if a fault is detected.**
- **During the sensitive process execution, data is untransformed if no fault is detected.**

Detect And Derive !

- **Example on RSA Signature - s = m^d mod n**



<u>Infection process</u>

Transform elements

<u>Disinfection process</u>

- ***Corrupt elements in case of FA***
- ***Disinfect elements if no FA***

Detect And Derive ! -- RSA Implementation 1

- let $d_{check} = \Psi(n,d,m)$
  - typically d_{check} would be checksum..
- let $U = d_{check} \mid d_{check} \mid \ldots \mid d_{check}$
- $d^* = d \text{ XOR } U$
- during the exponentation loop:
  - for each word d_i^* of d^* :
    - compute $d_{check} = \Psi(n,d,m)$
    - $d_i = d_i^* \text{ XOR } d_{check}$
- So the corruption of *d*, *n* or *m* will imply the computation of a signature with a wrong *private key* value !

Detect And Derive ! -- RSA Implementation 2

- **$d = (d_{k-1}, ..., d_1, d_0)$ the RSA private key.**
- **let ID be a secret number.**
- **let $A = \Psi(n,d,m)$**
- **let *f* be a bitwise function.**
- **compute d^* such as :**
  - **$d^*_{k-1} = d_{k-1}$ XOR $f(ID, A)$**
  - **For i from k-2 to 0**
    - **$d_i^* = d_i$ XOR $f(d^*_{i+1}, d_{i+1})$**
- **before the exponentiation loop :**
  - **$B = \Psi(n,d,m)$**
- **during the exponentiation :**
  - **Compute for each loop : $d_i = d^*_i$ XOR $f(d^*_{i+1}, d_{i+1})$**
- **So the corruption of *d*, *n* or *m element will sequentially corrupt the whole d_i sequence.***

Conclusion

- **New class of combined attacks.**
  - Experiments were conducted on “protected” implementation.
  - Only 1 successful fault is necessary to recover the entire private exponent value on certain implementations.

- **Need careful design of cryptographic modules.**
  - Important literature on fault or side channel protections but may not be enough to protect against PACA.
  - *Detect And Derive* strategy presented.

- **Not limited to RSA ...**

Questions ?