On the Need of Randomness in Fault Attack Countermeasures – Application to AES

Victor LOMNE[1], Thomas ROCHE[1], Adrian THILLARD[1]

[1] ANSSI (French Network and Information Security Agency)

FDTC 2012, Sunday, September 9th, 2012

Leuven, Belgium

Context of this work (1/2)

- Embedded Systems integrating Cryptography are susceptible to Physical Attacks, namely:
  - Side-Channel Attacks (SCA)
  - Fault Attacks (FA)
  - Combined Attacks (CA)

Context of this work (2/2)

- In this work we consider the security of Block Ciphers vs:
  - Side-Channel Attacks (SCA)
  - Fault Attacks (FA)
  - Combined Attacks (CA)
- As example we will use the AES cipher

Outline

1. Physical Attacks
   - Side-Channel Attacks
   - Fault Attacks
   - Combined Attacks
2. New Attacks on Classical Countermeasures
   - Combined Attack on Detection CM
   - Fault Attacks on Infection CM
   - On the Need of Randomness
3. Extended Countermeasures
   - Secure Detection
   - Secure Infection
   - Summary

Outline

1. Physical Attacks
   - Side-Channel Attacks
   - Fault Attacks
   - Combined Attacks
2. New Attacks on Classical Countermeasures
   - Combined Attack on Detection CM
   - Fault Attacks on Infection CM
   - On the Need of Randomness
3. Extended Countermeasures
   - Secure Detection
   - Secure Infection
   - Summary

Side-Channel Attacks

- A CMOS device leaks information about its state during a computation through side-channels *(power, electromagnetic radiations, time ...)*

- SCA: exploits these physical leakages correlated with computed data to guess a secret

    - Simple SCA (SSCA): exploits 1 crypto. operation
    - Differential SCA (DSCA): exploits several crypto. operations
      ⇒ *very powerful due to its resistance to noise*
    - Template Attacks (TA): profiling phase / matching phase
      ⇒ *allow to capture the maximum of information*

SCA Countermeasures

- Masking: only family of countermeasures with formal proofs
  - Principle: randomize input of the crypto. operation
  - Based on secret sharing
  - Input is shared in d shares $\Rightarrow$ masking scheme of order d
- Attack on Masking: High-Order DSCA
  - A d^{th} order masking scheme can be defeated by a $(d+1)^{th}$ order DSCA
  - It consists in combining the handling of the d shares before applying a 1^{st} order DSCA
  - HO-DSCA complexity is exponential in the masking order

Fault Attacks (1/2)

- Induce a logical error during a crypto. operation
- Different physical means to induce such an error
  power glitch, clock glitch, light beam, EM field ...
- Exploit few pairs of valid/faulty ciphertexts to retrieve the key
- A FA requires a Fault Model based on an Invariant

Fault Attacks (2/2)

Definition

A Fault Model is a function f such that:

$$f : x \rightarrow x \star e \tag{1}$$

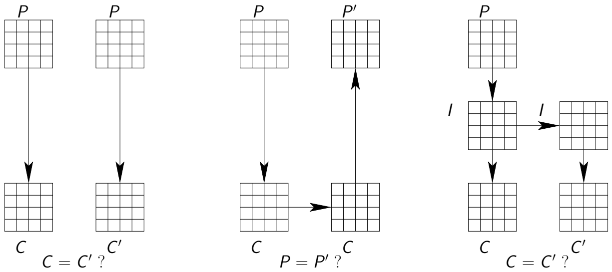
x target variable, e fault logical effect and $\star$ a logical operation

New classification of FA based on the Invariant

- FA based on a Fixed Fault Diffusion Pattern
  [Piret+ 2003], [Mukhopadhyay+ 2009] ...
- FA based on a Fixed Fault Logical Effect
  Safe Error Attack, [Roche+ 2011]...

Classical FA Countermeasures (1/2)

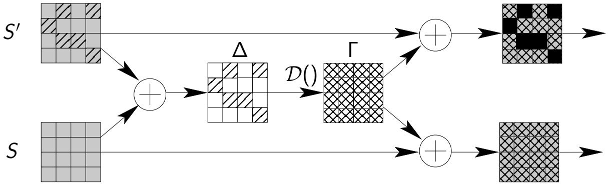
- First classical FA countermeasure: Detection scheme
- 3 classical Detection schemes:



Full Duplication — Encrypt/Decrypt — Partial Duplication

Classical FA Countermeasures (2/2)

- Second classical FA countermeasure: Infection scheme
- Generic sketch exhibiting the Infection CM:
  - S, S' the two `States`
  - $\mathcal{D}$ the *diffusion function* (such as $\mathcal{D}(0) = 0$)



Combined Attacks (1/2)

- Consider a secure AES implementation using:
  - A masking scheme such that SCA are unpracticable
  - A duplication countermeasure to avoid FA
- Is such an implementation really secure ?
  - If one takes each attack path alone yes ...
  - But if one mixes both attack paths ...
- Combined Attacks exploit the *side-channel leakage* of a faulty encryption to bypass both SCA and FA CM
  - Combined Attack of [Clavier+ 2010]
  - Combined Attack of [Roche+ 2011]

Combined Attacks (2/2)

- Example: Combined Attack of [Roche+ 2011]
    - Encrypt N plaintexts $P_1 \dots P_N$ and keep the N ciphertexts $C_1 \dots C_N$
    - Encrypt the N plaintexts once again by injecting a fault during the penultimate round of the Key-Schedule and record the leakage traces $\Omega_1 \dots \Omega_N$
    - Exploit the side-channel leakage of the faulty ciphertext:
      $k = argmax(\rho(HW(SB(SB^{-1}(C_j^i \oplus \hat{k}) \oplus \hat{e}_9) \oplus \hat{k} \oplus \hat{e}_{10}), \Omega_i))$
    - The attack will work if the fault has the effect of a XOR with a non negligible rate
- Interestingly enough, up to now only FA based on a Fixed Fault Logical Effect have been extended to CA

Combined Attack Countermeasure

- In [Roche+ 2011], authors propose to perform a secure comparison to avoid the leakage of the faulty ciphertext:

Algorithm 1 Secure Comparison

INPUT: two masked ciphertexts $C \oplus M$ and $C' \oplus M'$ and their respective masks M and M'

OUTPUT: C if $C = C'$, 0 otherwise

1. **do** $a = M \oplus (C' \oplus M')$
2. **do** $b = M' \oplus (C \oplus M)$
3. **if** $a = b$ **then return** C
4. **else return** 0

Outline

1. Physical Attacks
   - Side-Channel Attacks
   - Fault Attacks
   - Combined Attacks
2. New Attacks on Classical Countermeasures
   - Combined Attack on Detection CM
   - Fault Attacks on Infection CM
   - On the Need of Randomness
3. Extended Countermeasures
   - Secure Detection
   - Secure Infection
   - Summary

Combined Attack on Detection CM

- New Combined Attack on [Roche+ 2011] countermeasure:
  - At step 3 of algorithm 1, one check if $a = b$
  - In a lot of architectures, a comparison involves:
    $\Rightarrow$ *exclusive-or* or *substraction*
    $\Rightarrow \Pr(HW(a-b) = HW(a \oplus b) | (a,b) \in GF(2^8)^2) > 36\%$
  - Thus $\Delta = (M' \oplus (C \oplus M)) \oplus (M \oplus (C' \oplus M'))$ leaks $(C \oplus C')$
  - Possibility to adapt the CA of Roche *et al.* to exploit Δ:
    $k = argmax(\rho(HW(SB(SB^{-1}(C_j^i \oplus \hat{k}) \oplus \hat{e}_9) \oplus \hat{k} \oplus \hat{e}_{10} \oplus C_j^i), \Omega_i))$

Fault Attack on Infection CM (1/2)

- We show that any Deterministic Infection CM is inefficient:
  - If Infection placed before last `MixColumns`

    ⇒ inject a fault between Infection and last `MixColumns`
    ⇒ case of a classical *Piret Attack*
  - If Infection placed between last `MixColumns` & last `SubBytes`

    ⇒ inject a fault before the Infection
    ⇒ leads to a modified *Piret Attack*
    exploit the Infection instead of the MixColumns
  - If Infection placed after the last `SubBytes`

    ⇒ inject a fault before the `MixColumns`
    ⇒ leads to a modified *Piret Attack*
    make an hypothesis on 5 *bytes instead of* 4

Fault Attack on Infection CM (2/2)

- [Roche+ 2011] DFA breaks any Deterministic Infection CM:
- As the fault model:
  - has to affect the Key-Schedule during its penultimate round *(thus round keys* 9 *and* 10 *will be affected)*
  - could be of any kind, and affect all the bytes at the same time
  - must have a good repeatability *(two faults have a good chance to induce the same error)*
- Any Deterministic Infection CM will have no effect against this attack

On the Need of Randomness

- Any Deterministic Detection or Infection scheme can be defeated via FA or CA
- About Detection CM:
  - CM of [Roche+ 2011]
- About Infection CM:
  - CM of [Joye+ 2007]
  - CM of [Fournier+ 2011]
- The flaw comes from the deterministic property of the CM $\Rightarrow$ need of Randomness

Outline

1. Physical Attacks
   - Side-Channel Attacks
   - Fault Attacks
   - Combined Attacks
2. New Attacks on Classical Countermeasures
   - Combined Attack on Detection CM
   - Fault Attacks on Infection CM
   - On the Need of Randomness
3. Extended Countermeasures
   - Secure Detection
   - Secure Infection
   - Summary

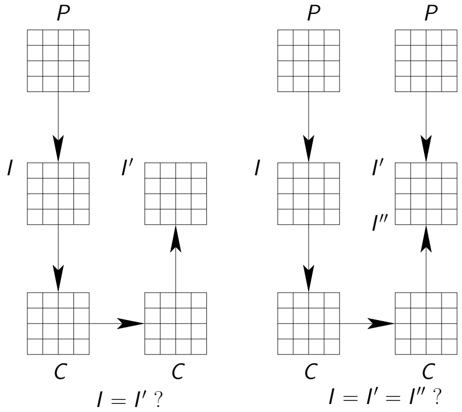
Secure Detection (1/2)

Algorithm 2 Secure Comparison

INPUT: two masked States $S \oplus M_1$ and $S' \oplus M_2$, their respective masks M_1 and M_2 and a fresh random mask $M_3 \neq 0$.
OUTPUT: S if $S = S'$, 0 otherwise

1. **do** $a = M_3 \cdot (S \oplus M_1)$
2. **do** $b = M_3 \cdot (S' \oplus M_2)$
3. **do** $c = a \oplus b$ $\quad [= M_3 \cdot (S \oplus M_1 \oplus S' \oplus M_2)]$
4. **do** $d = M_1 \oplus M_2$
5. **do** $e = M_3 \cdot d$ $\quad [= M_3 \cdot (M_1 \oplus M_2)]$
6. **if** $e = c$ **then return** $(S \oplus M_1) \oplus M_1$
7. **else return** 0

Secure Detection (2/2)



Encrypt/Partial Decrypt

Encrypt/Partial Encrypt/Partial Decrypt

Secure Infection

Algorithm 3 Secure Infection

INPUT: two masked `States` $S \oplus M_1$ and $S' \oplus M_2$, their respective masks M_1 and M_2 and a fresh random mask $M_3 \neq 0$ and $\neq 1$.
OUTPUT: the infected `States` $S \oplus M_1 \oplus \Gamma$ and $S' \oplus M_2 \oplus \Gamma$

1. **do** $a = M_3 \cdot (S \oplus M_1)$
2. **do** $b = M_3 \cdot (S' \oplus M_2)$
3. **do** $c = a \oplus b$
4. **do** $d = M_1 \oplus M_2$
5. **do** $e = M_3 \cdot d$
6. **do** $f = (S \oplus M_1) \oplus c$
7. **do** $g = f \oplus e$
8. **do** $h = (S' \oplus M_2) \oplus c$
9. **do** $i = h \oplus e$
10. **return** (g, i)

Summary

Countermeasures	Threats
Full Duplication	- Combined Attacks - Double Faults (bypass comparison)
Encrypt/Decrypt	- Combined Attacks - Double Faults (bypass comparison)
Partial Duplication	- Single Fault + Ability to Decrypt - Combined Attacks - Double Faults (bypass comparison)
Full Duplication + Mult. Mask based Secure Comp.	- Double Faults (bypass comparison)
Encrypt/Partial Decrypt	- Single Fault + Ability to Decrypt - Double Faults (bypass comparison)
Encrypt/Partial Encrypt/Partial Decrypt	- Double Faults (bypass comparison)
Infection with Fixed Diffusion	- Fixed fault diffusion DFA - Fixed fault effect DFA
Mult. Mask based Secure Infection	–
Encrypt/Partial Decrypt Infection	–

⌣̈ Thank you for your attention !