Fault-based Cryptanalysis on Block Ciphers

Victor LOMNE

LIRMM / university of Montpellier

COSADE 2017, Thursday April 13 2017, *Paris, France*

Outline

Agenda

Fault Zoology (1/2)

- Different ways to generate a fault:
  - Under / over-powering the IC
  - Tamper with the IC clock
  - Light injection
  - ElectroMagnetic (EM) field injection
  - Physical modification of the IC
    e.g. laser cutter, FIB
  - Software induced fault
    e.g. overclocking, register / memory modification

Fault Zoology (2/2)

- The duration of the fault can be:
  - Transient
  - Permanent
- Different effects:
  - Modification of operation flow
  - Modification of operands
- Different goals:
  - Bypassing a security mechanism
    PIN verification, file access right control, secure bootchain, ...
  - Generating faulty encryptions/signatures
    $\Rightarrow$ *fault-based cryptanalysis*
  - Combined Attacks
    JavaCard based, FA + SCA

Agenda

Under / Over-powering the IC (1/3)

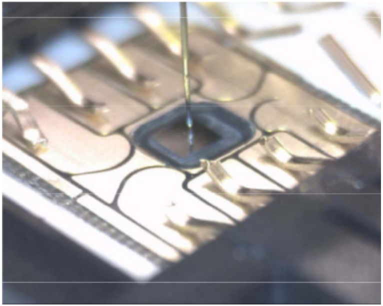
- Under/over-power an IC during a very short time
- Over-powering cause unexpected electrical phenomenoms inside the IC
  e.g. local shortcuts
- Under-powering slows down the processing of the IC
  e.g. bad memory read/write, bad coprocessor execution
- Low / medium-cost attack
  e.g. power supply, pulse generator, custom electronic board

Under / Over-powering the IC (2/3)

- Adversary can control:
  - Amplitude of the glitch
  - Duration of the glitch
  - Shape of the glitch
- Generally no control of the fault precision:
  - On a microcontroller running code, modification of the current executed opcode and/or operand(s)
  - On a hardware coprocessor, modification of (some of) the current processed word(s) (e.g. registers)

Under / Over-powering the IC (3/3)

- Recent variant [Tobich+ 2012]:
  BBI: Body Bias Injection
- Consist in putting a needle in contact with the IC silicon through its backside



Tamper with the clock (1/2)

- Reduce one or several clock period(s) feeding the IC
- Accelerates the processing of the IC
  e.g. DFF sampling before correct computation of current instruction / combinational logic
- Low / medium-cost attack
  e.g. signal generator, custom electronic board

Tamper with the clock (2/2)

- Adversary can control:
  - Duration of the reduced clock period
  - Number of reduced clock period(s)
- Generally no control of the fault precision:
  - On a microcontroller running code, modification of the current executed opcode and/or operand(s)
  - On a hardware coprocessor, modification of (some of) the current processed word(s) (e.g. registers)

Agenda

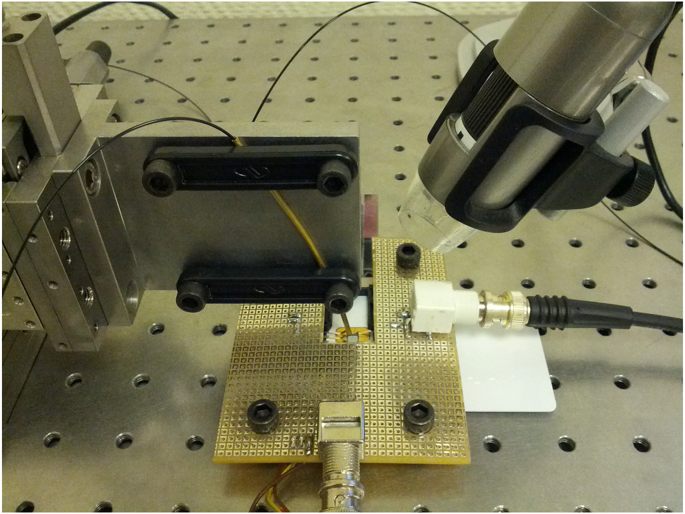
Light based Fault Injection (1/2)

- Inject a light beam into the IC
- A photoelectric phenomenom transforms light energy into electrical energy, provoking unexpected behaviour of transistors
- Old school setups were using flash lamp
- Modern setups are based on laser modules
- Medium / high-cost attack
  e.g. pulse generator, laser diode module, motorized X-Y-Z stage, optical microscope

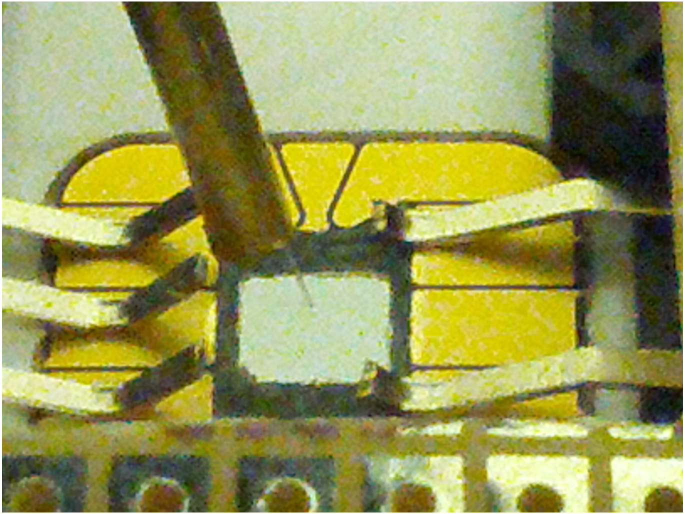
Light based Fault Injection (2/2)

- Requires to open the package of the IC in order the light beam can be injected into the frontside or the backside of the die
- On complex ICs with many metal layers, or on *secure* ICs with anti-probing shield, it can be difficult to inject light on the frontside of the IC
- As silicon is transparent to infrared light, backside light injection uses infrared light

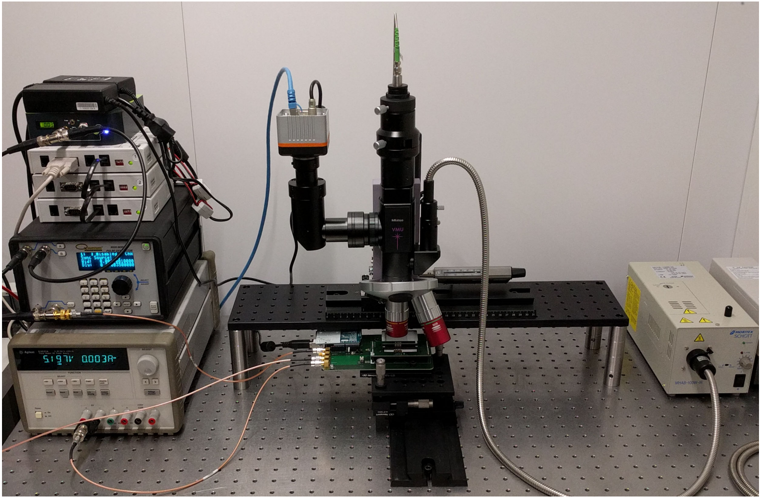
Laser Setup example 1 (1/2)



Laser Setup example 1 (2/2)



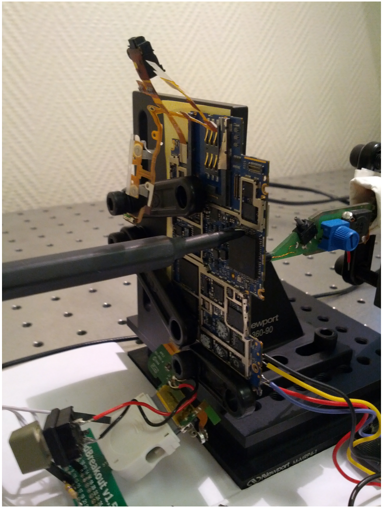
Laser Setup example 2



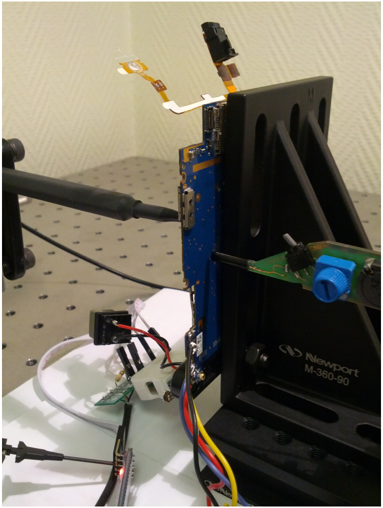
ElectroMagnetic Fault Injection (EMFI)

- Inject an electromagnetic field inside the IC
- Can be done without removing the package of the IC
- In practice, a glitch of high power is injected into an EM probe positionned above the IC
- Medium / high-cost attack
  e.g. high power pulse generator, EMFI probe, motorized X-Y-Z stage

ElectroMagnetic Injection Setup example (1/2)



ElectroMagnetic Injection Setup example (2/2)



Software Induced Faults

- White-Box cryptography is a concept where the key is hidden in the cryptographic implementation
- WBC usually used when several programs can run on the same device
- By running the binary program containing the WBC implem. over an emulator, it is possible to modify register values or memory access during its execution
  $\Rightarrow$ *software induced fault cryptanalysis (Sanfelix+ 2015, Alibert+ 2015)*
- Low-cost attack
  e.g. computer, RE software

Agenda

Synchronization Mean

- In many cases, need of a synchronization mean to trig the fault at the right instant
- Classical method: monitoring power consumption / EM activity of the IC to find the side-channel signature of the event one wants disturb
- Several solutions:
  - Triggering capabilities of oscilloscopes
  - Real-time waveform-matching based triggering system *Beckers+ 2016*

Agenda

Classification of Fault Models

- One can define a Fault Model as a function f such that:

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

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

- Any Fault-based Cryptanalysis requires an Invariant
  $\Rightarrow$ new classification of FA based on the Invariant:
  - FA based on a Fixed Fault Diffusion Pattern
    Differential Fault Analysis [Biham+ 1997], [Piret+ 2003]
  - FA based on a Fixed Fault Logical Effect
    Safe Error Attacks [Biham+ 1997]
    Statistical Fault Attacks [Fuhr+ 2013]

Agenda

Safe Error Attack (SEA) [Biham+ 1997]

- Requires two copies of the target device:
  - a first copy that the adversary can fully control
  - a second copy set at an unknown secret
- Requires the ability to encrypt several times the same plaintext
- Does not require any faulty ciphertext
- SEA requires two phases:
  - a profiling phase
  - an attack phase

Safe Error Attack (SEA) - Sketch

1. Profiling phase
   - Use the device the adversary can fully control
   - For every bit of the master key, find the fault parameters allowing to reset this bit
2. Attack phase
   - Use the device set at an unknown secret
   - Encrypt a plaintext and keep the ciphertext
   - For every bit of the key, encrypt once again the same plaintext, while injecting a fault with parameters of profiling phase for the current bit
   - If both ciphertexts are equal, the current bit is equal to 0, otherwise equal to 1

Agenda

Differential Fault Analysis (DFA) [Biham+ 1997]

- Requires the ability to encrypt two times the same plaintext
- Requires to have one or several pairs of correct and wrong ciphertexts corresponding to the same plaintext
  $P_1 \rightarrow (C_1, \widetilde{C_1})$
  $P_2 \rightarrow (C_2, \widetilde{C_2})$
  $\ldots$
  $P_N \rightarrow (C_N, \widetilde{C_N})$
- Requires to be able to fault only a part of the **State** at a particular position in the encryption
  e.g. one byte of the AES State before the last MixColumns

Differential Fault Analysis (DFA) - Sketch (1/2)

1. Let's assume the fault modify one State byte before the last MixColumns, compute the list D of all possible differences after last MixColumns
2. Consider two pairs of correct and faulty ciphertexts $(C_1, \widetilde{C_1})$ and $(C_2, \widetilde{C_2})$
3. Make an hypothesis on the 2 left most bytes of K, Kh^1, Kh^2. For each of the 2^{16} candidates, compute:
$\delta_{C_1} = S^{-1}(C_1^1 \oplus Kh^1, C_1^2 \oplus Kh^2) \oplus S^{-1}(\widetilde{C_1^1} \oplus Kh^1, \widetilde{C_1^2} \oplus Kh^2)$
$\delta_{C_2} = S^{-1}(C_2^1 \oplus Kh^1, C_2^2 \oplus Kh^2) \oplus S^{-1}(\widetilde{C_2^1} \oplus Kh^1, \widetilde{C_2^2} \oplus Kh^2)$

Differential Fault Analysis (DFA) - Sketch (2/2)

1. Compare the results with the 2 left-most bytes of the differences in D. The (Kh^1, Kh^2) for which a match is found for both ciphertext pairs are stored in a list L
2. For each candidate of L, try to extend it by one byte (computing both differences to check)
3. Keep extending candidates in L until they are 16-bytes long. At this stage, only the right key is remaining

Agenda

1. Fault Injection Means
   - Fault Zoology
   - Global Effect Faults
   - Local Effect Faults
   - Other Tools
2. Cryptanalysis methods
   - Fault Model
   - Safe Error Attack
   - DFA
   - **Statistical Fault Attack**
3. Countermeasures
   - Analog Level
   - Digital Level
   - Application to Crypto
4. Conclusion

Statistical Fault Attack (SFA) [Fuhr+ 2013]

- SFA has the property to work even with a set of faulty ciphertexts corresponding to different unknown plaintexts
  $P_1 \rightarrow \widetilde{C_1}$
  $P_2 \rightarrow \widetilde{C_2}$
  $\ldots$
  $P_N \rightarrow \widetilde{C_N}$
- Nevertheless it requires a Fixed Fault Logical Effect
  e.g. stuck-at a fixed value a **State** *byte with a good probability*
- SFA cannot be thwarted at the protocol level !!!

Statistical Fault Attack (SFA) - Sketch (1/2)

1. Collect a set of faulty AES ciphertexts $\widetilde{C_1}, \widetilde{C_2}, \ldots, \widetilde{C_N}$, by injecting a fault on one byte of the **State** after the penultimate **AddRoundKey**. We assume that the fault has a stuck-at effect to an unknown value e:
$\widetilde{S^1_{ak}} \;=\; S^1_{ak}\ AND\ e, \quad e \in [0, 255]$

2. A collection of correct ciphertext bytes $C_1, C_2, \ldots, C_N$ would have an uniform distribution
Here, due to the stuck-at fault, the collection of faulted ciphertext bytes $\widetilde{C_1}, \widetilde{C_2}, \ldots, \widetilde{C_N}$ has a biaised distribution

Statistical Fault Attack (SFA) - Sketch (2/2)

1. We can express $\tilde{S}ak_9^i$ as a function of $\tilde{C}^i$ and an hypothesis on one byte of K_{10}:
$\tilde{S}ak_9^i = SB^{-1} \circ SR^{-1}(\tilde{C}^i \oplus K_{10})$
2. Use a distinguisher to discriminate the correct key hypothesis. For instance, use the Minimal mean Hamming weight:
$h(\hat{K}) = \frac{1}{n}\sum_{i=1}^{n} HW(\hat{S}ak_r^i)$.

Agenda

(De)synchronization

- A fault injection requires a precise timing to be effective
- Adding temporal randomness makes the timing of the fault harder to set
- Classical ways to add temporal randomness:
  - jittered clock
  - dummy instructions
  - randomize operation flow
  - ...

Glitch Detectors

- The historical way to inject a fault in an IC is to under/over-power it during a short time
- Some IC manufacturers add glitch detectors after IC pads, checking that the current signal voltage stays in a defined range
- If a signal voltage goes outside from the defined range, a mechanism triggers an alarm
  e.g. flag set, interruption, reset, . . .

Laser Detectors (1/2)

- Laser injection often requires to only disturb a small IC area
- It requires to perform a spatial cartography to find hot spots
  CPU/co-processor registers, memory cells or decoders, ...
- Laser detectors that are small dedicated blocks are placed among the other IC cells

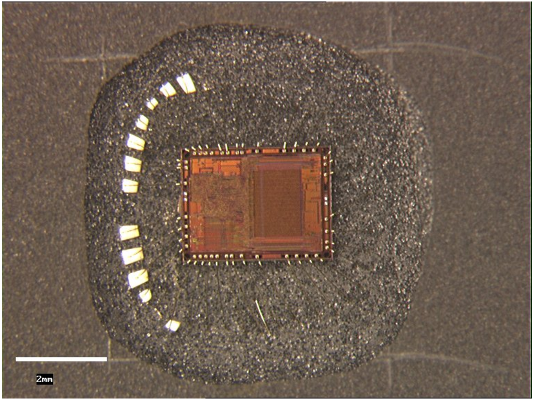
Laser Detectors (2/2)

- Different kind of Laser detectors:
  - analog based laser detectors
    e.g. based on photodiodes
  - digital based laser detectors
    e.g. based on custom logic cells
- Laser detectors do not cover the whole suface of the IC, but make the job of the adversary harder

IC Package as Countermeasure

- Several kind of fault injection techniques require to expose the die of the IC to perform the attack
  BBI, laser, ...
- Depending on the type of package, it can be more or less easy to expose the die:
  - smartcard packages are easy to open
  - metallic packages can be mechanically opened
  - epoxy packages require a chemical attack
  - Package-on-Package or 3D IC technology make the chip opening a nightmare

IC Package as Countermeasure: example 1



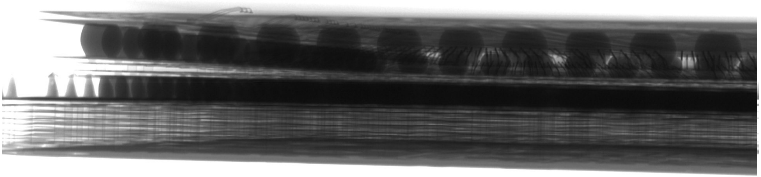
Epoxy package opened with fuming nitric acid

(courtesy of C. Toulemont, SERMA)

IC Package as Countermeasure: example 2

Application processor with memory stacked above
(courtesy of C. Toulemont, SERMA)

IC Package as Countermeasure: example 2



Application processor with memory stacked above - X-ray view
(courtesy of C. Toulemont, SERMA)

Agenda

Redundancy

- Redundancy consists in:
  - performing two times an operation
  - comparing results of both operation executions
    $\Rightarrow$ *require a conditionnal test*
- From a code theory point-of-view, it corresponds to the most obvious code one can construct
  duplication code
- A variant consists in performing the operation and the inverse operation, then checking that the obtained result is equal to the initial data

Examples of Redundancy

- Redundancy can be used in different ways:
  - Sequential redundancy for a software function
  - Sequential or Parallel redundancy for a hardware function
  - Use of redundant logics
    WDDL, STTL, ...
  - Securization of special registers by duplication or by storing a value and its inverse
    2 flip-flops are necessary to store one bit

Error Detection Codes

- Error Detection Codes are efficient tools to check the integrity of data
- ECC well suited to protect linear operations
  they are based on linear applications
- ECC bad suited to protect non-linear operations
  in particular they are not well suited to protect cryptographic primitives

Examples of Error Detection Codes

- Error Correcting Codes can be used in different ways:
  - Ensure the integrity of a secret data stored in NVM
  - Protect a memory decoder
    $\rightarrow$ ensure the integrity of opcodes
  - Protect linear parts of cryptographic algorithms
  - ...

Infection

- Infection consists in mixing a diffusion scheme with the operation to protect such that:

  1. if the processed data are not modified by a fault, the diffusion scheme has no effect on the final result
  2. if the processed data are modified by a fault, the diffusion scheme expands the erroenous data such that the final result is no more exploitable by the adversary

Agenda

Classical Detection Schemes For Block Ciphers

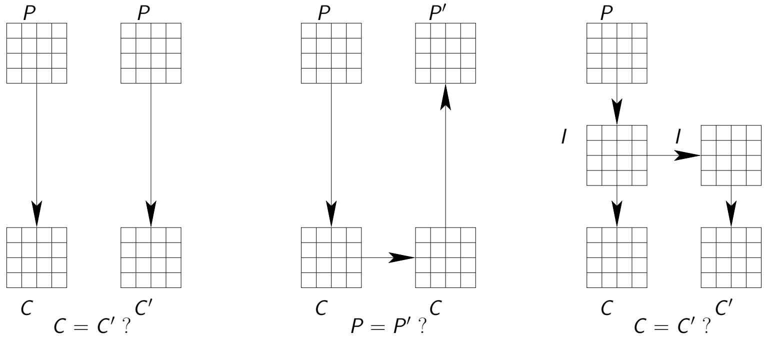
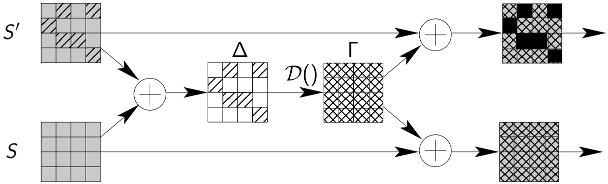


Figure: Three classical detection countermeasures. From left to right : Full Duplication, Encrypt/Decrypt, and Partial Duplication

Classical Infection Schemes For Block Ciphers

- Generic sketch exhibiting the Infection CM:
  - S, S' the two States
  - $\mathcal{D}$ the diffusion function (such as $\mathcal{D}(0) = 0$)



Agenda

Conclusion (1/2)

- Fault Attacks are a very powerful attack path:
  - Allow to modify the normal behaviour of a HW or SW function
  - Allow to extract cryptographic secrets
- Recent trend: high order fault attack
  - Temporal multi-fault attack
  - Spatio-temporal multi-fault attack

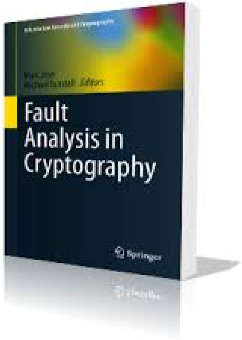
Conclusion (2/2)

- A lot of Fault Attack Countermeasures have been proposed in the litterature
- Generally mixed to increase the security level of the IC
  principle of defense in depth
- No countermeasure is perfect !
- A developper has firstly to define the level of the adversary he wants to thwart, and then choose the adequate tradeoff between efficiency and security

Certification Schemes

- Procedure to evaluate the security level of a product
- Three actors:
  the developper / the security lab / the scheme
- Some certification schemes:
  - Common Critera
  - EMVCo
  - ...

To go further



- book Fault Analysis in Cryptography
  Marc Joye and Michael Tunstall - SPRINGER

Questions ?

- contact: victor.lomne@lirmm.fr

Bonus 1: Bug Attack

- Pentium FDIV bug was a bug in the Intel *P*5 Pentium floating point unit (FPU)
- Because of the bug, the processor would return incorrect results for many calculations
- Nevertheless, bug is hard to detect
  1 *in* 9 *billion floating point divides with random parameters would produce inaccurate results*
- Shamir proposed a modified version of the Bellcore attack which exploits this bug to retrieve a RSA private key
- More dangerous than a classical fault attack because can be perfomed remotely

Bonus 2: PS3 Hack

- George Hotz (a.k.a. Geohot) published in 2009 a hack of the Sony PS3
- The otherOS functionnality of the PS3 allowed to boot a Linux OS
- A bus glitch allowed him to gain control of the hypervisor
  $\Rightarrow$ ring 0 access
  $\Rightarrow$ full memory access
  $\Rightarrow$ control gain of the OS bootchain
- In consequence Sony took George Hotz to court
- Sony and Hotz had settled the lawsuit out of court, on the condition that Hotz would never again resume any hacking work on Sony products