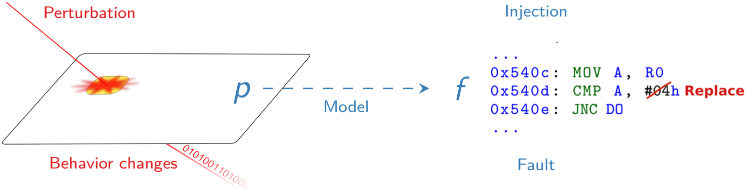
Fault Model Inference in Practice

(1) Laboratoire VERIMAG, Université de Grenoble-Alpes
(2) CEA-LETI
(3) SAFRAN IDENTITY AND SECURITY

SERTIF Workshop, 2016-10-11

Two spaces of parameters



Perturbation

Behavior changes

p Model f

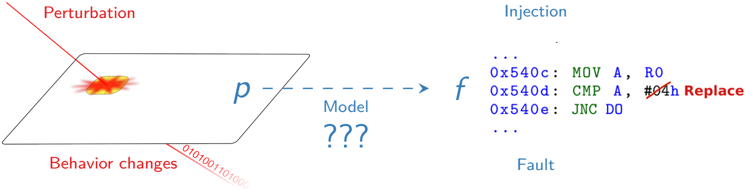
Injection

```
...
0x540c: MOV A, R0
0x540d: CMP A, #04h Replace
0x540e: JNC D0
...
```

Fault

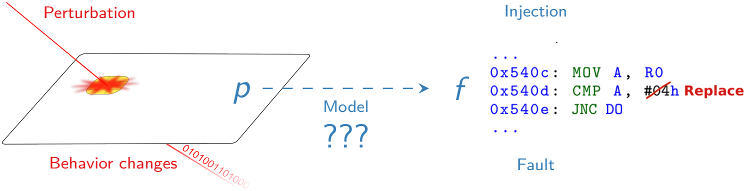
- Two spaces of parameters:
  - **parameter of the equipment** $p \in \mathcal{P}$: $p \mathrel{\hat{=}} (x = 12\,\mu\text{m},\ y = 24\,\mu\text{m},\ d = 3800\,\text{ns},\ w = 850\,\text{ns})$
  - **effect on the code** $f \in \mathcal{F}$: $f \mathrel{\hat{=}} (i = 124,\ \text{store}([\texttt{0x540d}],\ 0))$

Two spaces of parameters



- Two spaces of parameters:
  - **parameter of the equipment** $p \in \mathcal{P}$: $p \hat{=} (x = 12\,\mu\text{m},\ y = 24\,\mu\text{m},\ d = 3800\,\text{ns},\ w = 850\,\text{ns})$
  - **effect on the code** $f \in \mathcal{F}$: $f \hat{=} (i = 124,\ \text{store}([\texttt{0x540d}],\ 0))$
- **How to model the effects of perturbation attack on code?**

Two spaces of parameters



```
...
0x540c: MOV A, R0
0x540d: CMP A, #04h Replace
0x540e: JNC D0
...
```

- Two spaces of parameters:
  - **parameter of the equipment** $p \in \mathcal{P}$:
    $p \mathrel{\hat{=}} (x = 12\,\mu\text{m},\ y = 24\,\mu\text{m},\ d = 3800\,\text{ns},\ w = 850\,\text{ns})$
  - **effect on the code** $f \in \mathcal{F}$: $f \mathrel{\hat{=}} (i = 124,\ \text{store}([\texttt{0x540d}], 0))$
- **How to model the effects of perturbation attack on code?**
- The model will depend on the equipment of attack, and the attacked device

Fault as a relationship

- Fault: $p \underset{f}{\rightsquigarrow} c$

$$(x = 12\,\mu\text{m},\ y = 24\,\mu\text{m},\ d = 3800\,\text{ns}) \underset{f_{\text{A}}}{\rightsquigarrow} (i = 124,\ \text{store}(\text{A},\ 0))$$

- Fault model: set of faults

$$\{(x = 12,\ y = 24,\ d = 3000 + 200k) \underset{f_{\text{A}}(k)}{\rightsquigarrow} (i = 120 + k,\ \text{store}(\text{A},\ 0)),\ k \in \mathbb{N}\}$$

- Probabilistic fault model to compute:

$$\Pr(F = f \mid p)$$

Challenges

- The size of the space of parameters is too large
  Hundreds of years of attacks to cover the whole space!
- Several faults can have the same effect:
  - Register corruption
  - Store instruction corruption
  - Memory corruption

$\Longrightarrow$ **Black-box effect**

Defeating the black-box effect

Lionel Rivière's PhD thesis: Fault model extraction

Fault detection program

- Programs to disambiguate between possible faults.
- Get knowledge about the content of the black-box
- An example: EEPROM-RAM buffer copy
  - Executed from RAM
  - Sentinel RAM-RAM buffer copy

```
; main_loop:
  58: ldrb   r5, [r0, #0] ; r5 <- @EEPROM
  5a: strb   r5, [r2, #0] ; r5 -> @IO_EEPROM
  5c: ldrb   r5, [r1, #0] ; r5 <- @RAM
  5e: strb   r5, [r3, #0] ; r5 -> @IO_RAM
  60: add.w r0, r0, #1 ; @EEPROM += 1
  64: add.w r1, r1, #1 ; @RAM += 1
  68: add.w r2, r2, #1 ; @IO_EEPROM += 1
  6c: add.w r3, r3, #1 ; @IO_RAM += 1
```

However, obtained knowledge is partial

Fault Model Inference Method

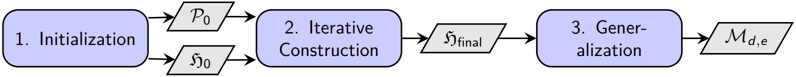
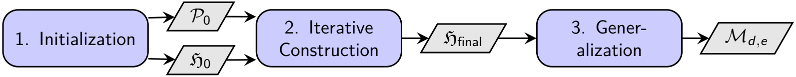


Figure: Fault model inference

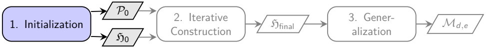
1. Initialization phase: parameter discovery to reduce the space of parameters
2. Iterative phase: physically attack several *ad-hoc* fault detection programs on the reduced space
3. Generalization phase: extend results to bigger set of parameters

A Case Study



- “Card C”: “Unsecure” Cortex M-4 8 MHz
- Attacked with EM injector (100 µm copper loop with a 500 A current during 10 ns)
- The method in practice:
  1. Initialization phase: effect of the parameters of equipment
  2. Iterative phase: 3 successive programs
  3. Generalization phase

Initialization phase: Effect of position and angle



1. Initialization

$\mathcal{P}_0$

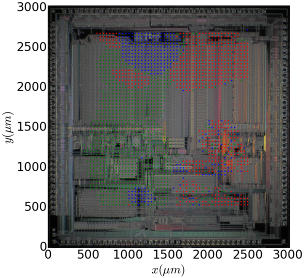
$\mathfrak{H}_0$

2. Iterative Construction

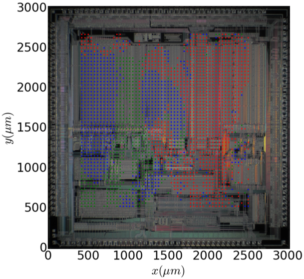
$\mathfrak{H}_{\text{final}}$

3. Generalization

$\mathcal{M}_{d,e}$



: $\theta = -90^\circ$

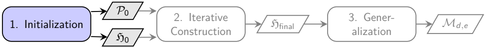


: $\theta = 0^\circ$

- EEPROM faults
- RAM/registers faults
- Mute

Choose one angle

Initialization phase: Effect of position and angle



1. Initialization

$\mathcal{P}_0$

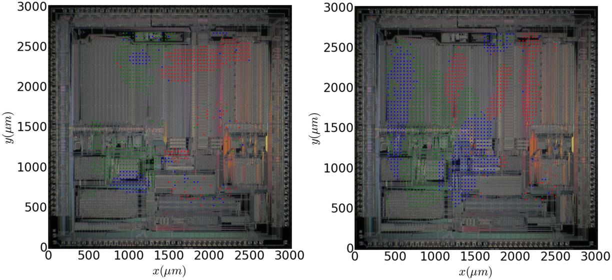
$\mathfrak{H}_0$

2. Iterative Construction

$\mathfrak{H}_{\text{final}}$

3. Generalization

$\mathcal{M}_{d,e}$



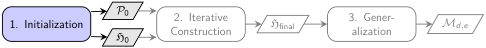
: $\theta = 90°$

: $\theta = 180°$

- EEPROM faults
- RAM/registers faults
- Mute

Choose one angle

Initialization phase: Effect of altitude



1. Initialization

$\mathcal{P}_0$

$\mathfrak{H}_0$

2. Iterative Construction

$\mathfrak{H}_{\text{final}}$

3. Generalization

$\mathcal{M}_{d,e}$

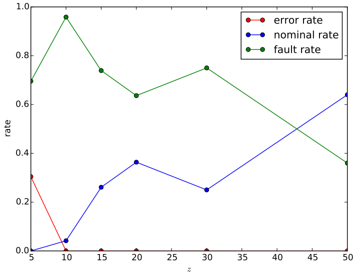
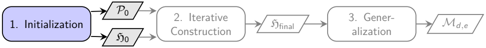


Figure: Influence of z

Choose one z

Initialization phase: Effect of delay


1. Initialization

$\mathcal{P}_0$

$\mathfrak{H}_0$

2. Iterative Construction

$\mathfrak{H}_{\text{final}}$

3. Generalization

$\mathcal{M}_{d,e}$

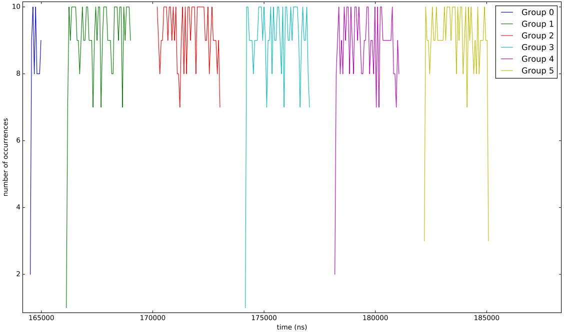
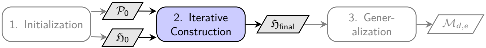


Figure: Fault count as a function of delay

Choose one delay

Iterative Phase: Fault in EEPROM



1. Initialization

$\mathcal{P}_0$

$\mathfrak{H}_0$

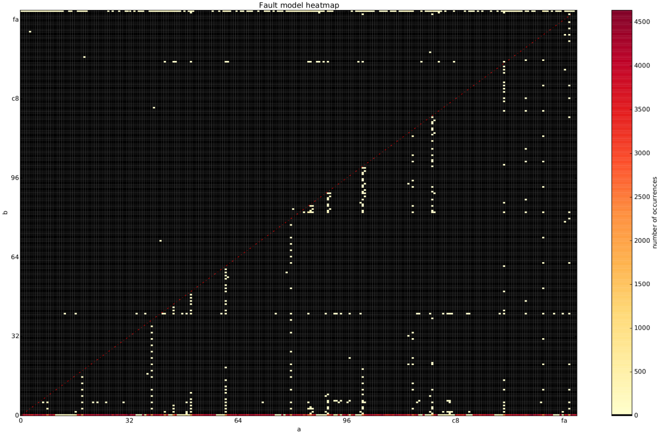
2. Iterative Construction

$\mathfrak{H}_{\text{final}}$

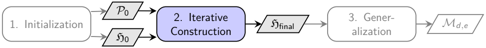
3. Generalization

$\mathcal{M}_{d,e}$

Previous knowledge: **None**



Iterative Phase: Fault in EEPROM


1. Initialization

$\mathcal{P}_0$

$\mathfrak{H}_0$

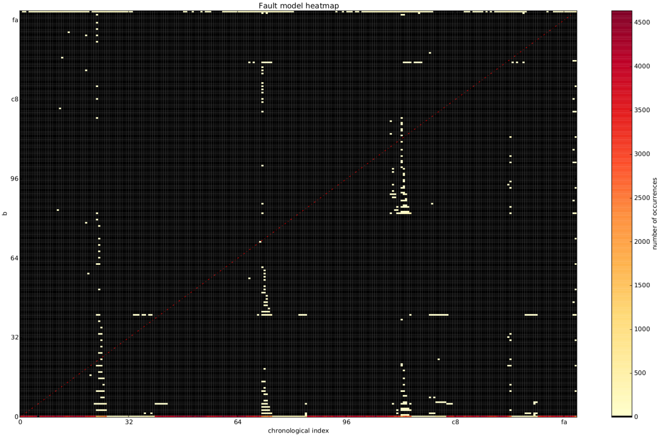
2. Iterative Construction

$\mathfrak{H}_{\text{final}}$

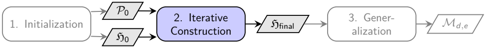
3. Generalization

$\mathcal{M}_{d,e}$

Previous knowledge: **None**



Iterative Phase: Results of Faults on EEPROM

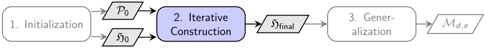


p where only EEPROM reads are perturbed. Perturbations are:

- 16 consecutive bytes are faulted to `0x00` or `0xFF`
- The first perturbed address has always a 16-bytes alignment

Goal: True for data EEPROM read. Check that on code!

Iterative Phase: Effect on Code


1. Initialization
$\mathcal{P}_0$
$\mathfrak{H}_0$
2. Iterative Construction
$\mathfrak{H}_{\text{final}}$
3. Generalization
$\mathcal{M}_{d,e}$

Previous knowledge:

- p where only EEPROM is faulted (no RAM or register faults)

Test:

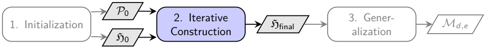
- Instruction `0x00: movs r0, r0` is unchanged.

Program:

```
test_nop:
; initialization
04: mov r0, IO ; r0 <- @IO
08: mov r4, IO_sentinel ; r4 <- @IO_sentinel
0c: mov r1, #10 ; r1 <- 10
10: mov r2, #20 ; r2 <- 20
18: str r1, [r4]; r1 -> @IO_sentinel
1c: str r2, [r4]; r2 -> @IO_sentinel
20: movs r0, r0 ; NOP
24: movs r0, r0 ; NOP
; [...]
a0: movs r0, r0 ; NOP
; check in memory
a4: str r1, [r0] ; r1 -> @IO
a8: str r2, [r0+4] ; r2 -> @IO
```

Diagnostic: Success

Iterative Phase: Offset Confirmation



Previous knowledge:

- p where only EEPROM is faulted (no RAM or register faults)
- Instruction `0x00: movs r0, r0` is unchanged.

Test:

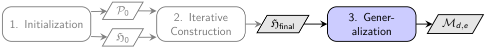
- Only aligned blocks of 16 consecutive addresses are affected.

Program:

```
test_align:
; initialization
; [...]
20: movs r0, r0 ; NOP
24: movs r0, r0 ; NOP
; [...]
78: movs r0, r0 ; NOP
7c: adds r1, #1 ; r1 <- r1 + 1
80: adds r1, #1 , r1 <- r1 + 1
84: movs r0, r0 ; NOP
; [...]
a0: movs r0, r0 ; NOP
; check in memory
; [...]
```

Diagnostic: Success

Generalization Phase: Final Extracted Model



Parameter	Effect	Probability
$(d = d_0 + k\delta)$	16 bytes: $(a_d \rightarrow \texttt{0x00})$	16%
$(d = d_0 + k\delta)$	16 bytes: $(a_d \rightarrow \texttt{0xFF})$	0.3%

Laser Fault Model

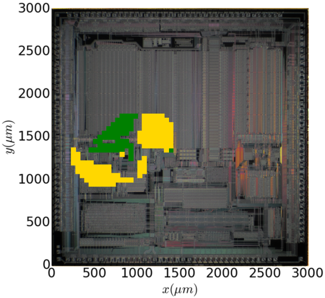


Figure: Cartography: `0xFF`, `0x00`

Parameter	Effect	Probability
$(x = x_0,\ y = y_0,\ d = d_0 + k\delta)$	16 bytes: $(a_d \rightarrow \texttt{0x00})$	21%
$(x = x_1,\ y = y_1,\ d = d_0 + k\delta)$	16 bytes: $(a_d \rightarrow \texttt{0xFF})$	69%

Conclusion on Fault Model Inference

- 4 inferred models
  - On 3 cards (2 Cortex-M, 1 proprietary CISC)
  - 2 with laser, 2 with EM
- High sensibility to equipment parameters
- New probabilistic aspect
- Fault Detection Programs in sequence to defeat the black-box effect
- Find model at the device level to reuse with various applications
- *ad-hoc* method... fault detection program database?

Fault	Pr
$a \rightarrow 0 \mid a \neq 0$	4.8%
$a \rightarrow b \mid a \neq 0 \wedge \boldsymbol{d(a,b) \leq 1\%}$	1.8%
$a \rightarrow b \mid a \neq 0 \wedge \boldsymbol{1\% < d(a,b) \leq 20\%}$	1.6%
$a \rightarrow b \mid a \neq 0 \wedge b \neq 0 \wedge \boldsymbol{d(a,b) > 20\%}$	1.3%
$(a \rightarrow 0, a' \rightarrow 0) > \mid (a, a') \neq 0$	0.5%

Table: Card A, EM

Fault	Pr
Bitreset of 1 byte: $a \rightarrow 0 \mid a \neq 0$	4.32%
Bitreset of 2 bytes	2.93%
Bitreset of 3 bytes	3.13%
Bitreset of 4 bytes	2.98%
Bitreset of 5 bytes	6.56%
Bitreset of 6 bytes	2.48%

Table: Card B, laser