September 13, 2015

To exploit fault injection on non-injective Sboxes

Guillaume BETHOUART
Nicolas DEBANDE

Agenda

1. Introduction
   - Overview of fault attacks
   - Principle of our attack
2. Application to the Data Encryption Standard
   - Data Encryption Standard
   - Attack Simulation
   - Countermeasures
3. Conclusion

Outline

1. Introduction
   - Overview of fault attacks
   - Principle of our attack
2. Application to the Data Encryption Standard
   - Data Encryption Standard
   - Attack Simulation
   - Countermeasures
3. Conclusion

Overview of fault attacks

- **Safe Error Attacks**
  + Just need to know if the calculus has been disturbed or not
- **Differential Fault Attacks**
  + Work with masked implementations
- **Collision Fault Attacks**
  + Do not need to encrypt the same plaintext

Take the best of each

Overview of fault attacks

- **Safe Error Attacks**
  - + Just need to know if the calculus has been disturbed or not
- **Differential Fault Attacks**
  - + Work with masked implementations
- **Collision Fault Attacks**
  - + Do not need to encrypt the same plaintext

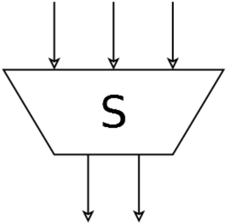
Take the best of each

Outline

1. Introduction
   - Overview of fault attacks
   - Principle of our attack
2. Application to the Data Encryption Standard
   - Data Encryption Standard
   - Attack Simulation
   - Countermeasures
3. Conclusion

Principle of our attack

- A non-injective Sbox from $\mathbb{F}_2^3$ to $\mathbb{F}_2^2$:



Non injectivity

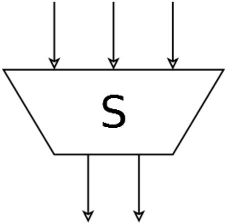
- there exist two different inputs a_1, a_2 such as $S(a_1) = S(a_2)$
- there are an input a and a differential δ such as $S(a \oplus \delta) = S(a)$

N-Differential

For a given δ, if there exists a such as $S(a \oplus \delta) = S(a)$, δ is called a **N-differential**

Principle of our attack

- A non-injective Sbox from $\mathbb{F}_2^3$ to $\mathbb{F}_2^2$:



Non injectivity

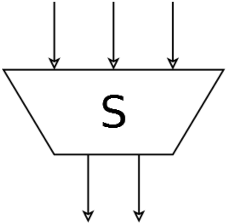
- there exist two different inputs a_1, a_2 such as $S(a_1) = S(a_2)$
- there are an input a and a differential δ such as $S(a \oplus \delta) = S(a)$

N-Differential

For a given δ, if there exists a such as $S(a \oplus \delta) = S(a)$, δ is called a **N-differential**

Principle of our attack

- A non-injective Sbox from $\mathbb{F}_2^3$ to $\mathbb{F}_2^2$:



Non injectivity

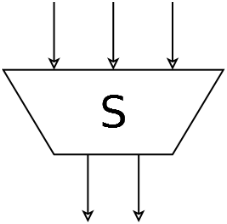
- there exist two different inputs a_1, a_2 such as $S(a_1) = S(a_2)$
- there are an input a and a differential δ such as $S(a \oplus \delta) = S(a)$

N-Differential

For a given δ, if there exists a such as $S(a \oplus \delta) = S(a)$, δ is called a **N-differential**

Principle of our attack

- A non-injective Sbox from $\mathbb{F}_2^3$ to $\mathbb{F}_2^2$:



Non injectivity

- there exist two different inputs a_1, a_2 such as $S(a_1) = S(a_2)$
- there are an input a and a differential δ such as $S(a \oplus \delta) = S(a)$

N-Differential

For a given δ, if there exists a such as $S(a \oplus \delta) = S(a)$, δ is called a **N-differential**

Principle of our attack

Truth table

a	S(a)
0	1
1	0
2	2
3	3
4	3
5	1
6	2
7	0

Example

If the calculus is not disturbed by the fault δ, we know :

$$S(a \oplus \delta) = S(a)$$

For a **known** fault $\delta = 4$

$$S(0 \oplus \delta) = S(4) \neq S(0)$$

$$S(1 \oplus \delta) = S(5) \neq S(1)$$

$$S(2 \oplus \delta) = S(6) = S(2)$$

$$S(3 \oplus \delta) = S(7) \neq S(3)$$

Principle of our attack

Truth table

a	S(a)
0	1
1	0
2	2
3	3
4	3
5	1
6	2
7	0

Example

If the calculus is not disturbed by the fault δ, we know :

$$S(a \oplus \delta) = S(a)$$

For a **known** fault $\delta = 4$

$$S(0 \oplus \delta) = S(4) \neq S(0)$$
$$S(1 \oplus \delta) = S(5) \neq S(1)$$
$$S(2 \oplus \delta) = S(6) = S(2)$$
$$S(3 \oplus \delta) = S(7) \neq S(3)$$

Principle of our attack

Truth table

a	S(a)
0	1
1	0
2	2
3	3
4	3
5	1
6	2
7	0

Example

If the calculus is not disturbed by the fault δ, we know :

$$S(a \oplus \delta) = S(a)$$

For a **known** fault $\delta = 4$

$$S(0 \oplus \delta) = S(4) \neq S(0)$$

$$S(1 \oplus \delta) = S(5) \neq S(1)$$

$$S(2 \oplus \delta) = S(6) = S(2)$$

$$S(3 \oplus \delta) = S(7) \neq S(3)$$

Principle of our attack

Truth table

a	S(a)
0	1
1	0
2	2
3	3
4	3
5	1
6	2
7	0

Example

If the calculus is not disturbed by the fault δ, we know :

$$S(a \oplus \delta) = S(a)$$

For a **known** fault $\delta = 4$

$$S(0 \oplus \delta) = S(4) \neq S(0)$$

$$S(1 \oplus \delta) = S(5) \neq S(1)$$

$$S(2 \oplus \delta) = S(6) = S(2)$$

$$S(3 \oplus \delta) = S(7) \neq S(3)$$

Principle of our attack

Truth table

a	S(a)
0	1
1	0
2	2
3	3
4	3
5	1
6	2
7	0

Example

If the calculus is not disturbed by the fault δ, we know :

$$S(a \oplus \delta) = S(a)$$

For a **known** fault $\delta = 4$

$$S(0 \oplus \delta) = S(4) \neq S(0)$$

$$S(1 \oplus \delta) = S(5) \neq S(1)$$

$$S(2 \oplus \delta) = S(6) = S(2)$$

$$S(3 \oplus \delta) = S(7) \neq S(3)$$

Principle of our attack

Truth table

a	S(a)
0	1
1	0
2	2
3	3
4	3
5	1
6	2
7	0

Example

If the calculus is not disturbed by the fault δ, we know :

$$S(a \oplus \delta) = S(a)$$

For a **known** fault $\delta = 4$

$$S(0 \oplus \delta) = S(4) \neq S(0)$$

$$S(1 \oplus \delta) = S(5) \neq S(1)$$

$$S(2 \oplus \delta) = S(6) = S(2)$$

$$S(3 \oplus \delta) = S(7) \neq S(3)$$

Principle of our attack

Result

For a **known** fault $\delta = 4$

If

$$S(a \oplus \delta) = S(a)$$

We deduce :

$$a = 2 \quad \text{or} \quad a = 6$$

To deduce information about the input we only need to know :

- The fault value δ
- **If the calculus is disturbed or not**

Outline

1 Introduction

- Overview of fault attacks
- Principle of our attack

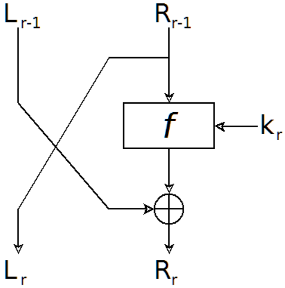
2 Application to the Data Encryption Standard

- Data Encryption Standard
- Attack Simulation
- Countermeasures

3 Conclusion

Data Encryption Standard

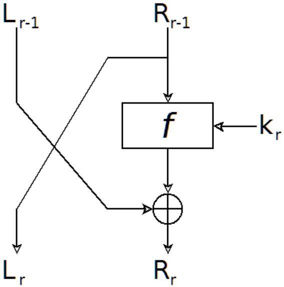
- DES follows a Feistel scheme :



- 64-bit block cipher using a 56-bit key **k**
- 16 times the same round transformation **f**

Data Encryption Standard

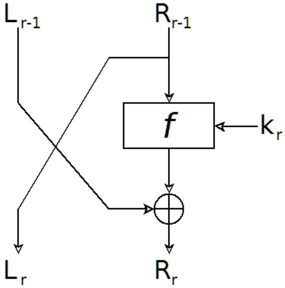
- DES follows a Feistel scheme :



- 64-bit block cipher using a 56-bit key **k**
- 16 times the same round transformation **f**

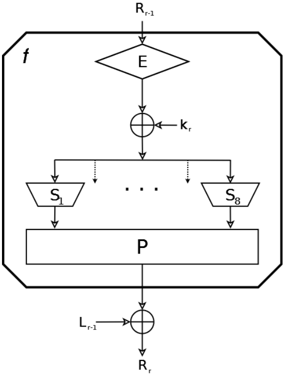
Data Encryption Standard

- DES follows a Feistel scheme :



- 64-bit block cipher using a 56-bit key **k**
- 16 times the same round transformation **f**

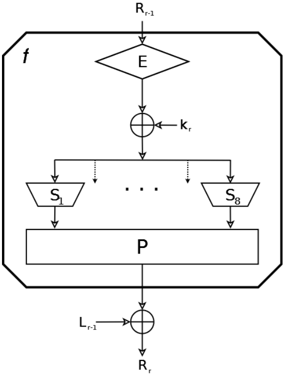
- Round function **f**



- Expansion function
- 48-bit round key k_r
- 8 different non-injective Sboxes
- Permutation

Data Encryption Standard

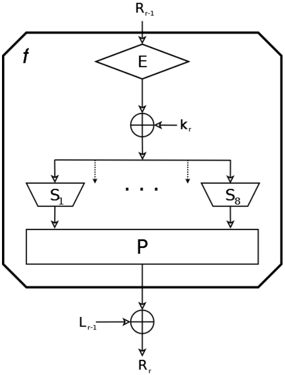
- Round function **f**



- Expansion function
- 48-bit round key k_r
- 8 different non-injective Sboxes
- Permutation

Data Encryption Standard

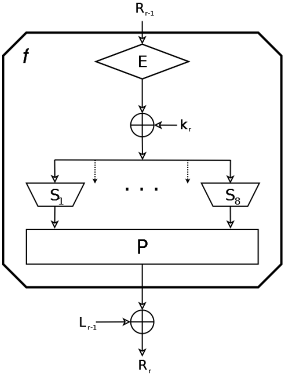
- Round function **f**



- Expansion function
- 48-bit round key k_r
- 8 different non-injective Sboxes
- Permutation

Data Encryption Standard

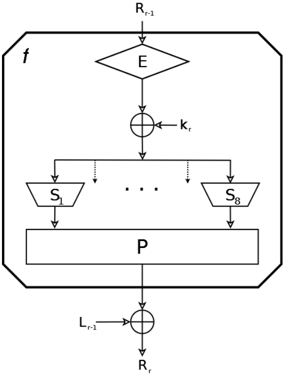
- Round function **f**



- Expansion function
- 48-bit round key k_r
- 8 different non-injective Sboxes
- Permutation

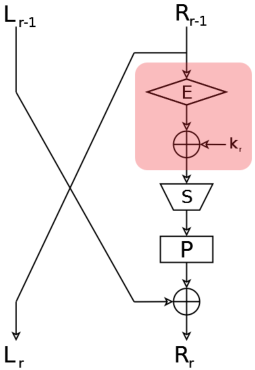
Data Encryption Standard

- Round function **f**



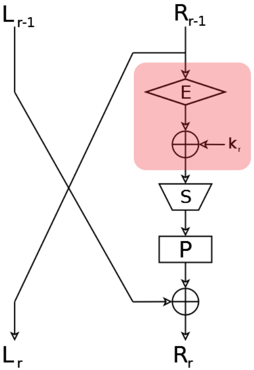
- Expansion function
- 48-bit round key k_r
- 8 different non-injective Sboxes
- Permutation

Attack timing



- First or last round
- After the data propagation
- Before Sboxes
- Fault affects only one Sbox

Attack timing



- First or last round
- After the data propagation
- Before Sboxes
- Fault affects only one Sbox

Attack with known fault

If the fault value is known

If we know $S(a \oplus \delta) = S(a)$ we deduce information on a

During the DES : $a = x \oplus k$, x the Expansion output and k the key
If we know :

- The fault δ
- The Expansion output x
- If $S(x \oplus k \oplus \delta) = S(x \oplus k)$ or not

We deduce information on k

But its a too restrictive model

- Fault injection does not have a 100% success rate (missed faults)
- The fault value is rarely constant

Attack with known fault

If the fault value is known

If we know $S(a \oplus \delta) = S(a)$ we deduce information on a

During the DES : $a = x \oplus k$, x the Expansion output and k the key

If we know :

- The fault δ
- The Expansion output x
- If $S(x \oplus k \oplus \delta) = S(x \oplus k)$ or not

We deduce information on k

But its a too restrictive model

- Fault injection does not have a 100% success rate (missed faults)
- The fault value is rarely constant

Attack with known fault

If the fault value is known

If we know $S(a \oplus \delta) = S(a)$ we deduce information on a

During the DES : $a = x \oplus k$, x the Expansion output and k the key
If we know :

- The fault δ
- The Expansion output x
- If $S(x \oplus k \oplus \delta) = S(x \oplus k)$ or not

We deduce information on k

But its a too restrictive model

- Fault injection does not have a 100% success rate (missed faults)
- The fault value is rarely constant

Attack with known fault

If the fault value is known

If we know $S(a \oplus \delta) = S(a)$ we deduce information on a

During the DES : $a = x \oplus k$, x the Expansion output and k the key
If we know :

- The fault δ
- The Expansion output x
- If $S(x \oplus k \oplus \delta) = S(x \oplus k)$ or not

We deduce information on k

But its a too restrictive model

- Fault injection does not have a 100% success rate (missed faults)
- The fault value is rarely constant

Attack with known fault

If the fault value is known

If we know $S(a \oplus \delta) = S(a)$ we deduce information on a

During the DES : $a = x \oplus k$, x the Expansion output and k the key
If we know :

- The fault δ
- The Expansion output x
- If $S(x \oplus k \oplus \delta) = S(x \oplus k)$ or not

We deduce information on k

But its a too restrictive model

- Fault injection does not have a 100% success rate (missed faults)
- The fault value is rarely constant

Improvement

Characterization stage

Characterization :

- Fault injection with known key
- We estimate a fault occurrence probability p for each fault value

Attack stage

Attack :

```
If the fault has no effect
.     For each $(\delta\ ,\ p)$
.     .     For each $k \in [\![0, 63]\!]$
.     .     .     If $S(x \oplus k \oplus \delta) = S(x \oplus k)$
.     .     .     .     $counter[k] \mathrel{+}= \ p$
```

Improvement

Characterization stage

Characterization :

- Fault injection with known key
- We estimate a fault occurrence probability p for each fault value

Attack stage

Attack :

If the fault has no effect

```
.   For each $(\delta\ ,\ p)$
.   .   For each $k \in [\![0,63]\!]$
.   .   .   If $S(x \oplus k \oplus \delta) = S(x \oplus k)$
.   .   .   .   $counter[k] += \ p$
```

Improvement

Characterization stage

Characterization :

- Fault injection with known key
- We estimate a fault occurrence probability p for each fault value

Attack stage

Attack :

If the fault has no effect

```
.    For each $(\delta , p)$
.    .    For each $k \in [\![0, 63]\!]$
.    .    .    If $S(x \oplus k \oplus \delta) = S(x \oplus k)$
.    .    .    .    $counter[k] += p$
```

Improvement

Characterization stage

Characterization :

- Fault injection with known key
- We estimate a fault occurrence probability p for each fault value

Attack stage

Attack :

```
If the fault has no effect
.    For each $(\delta\ ,\ p)$
.    .    For each $k \in [\![0, 63]\!]$
.    .    .    If $S(x \oplus k \oplus \delta) = S(x \oplus k)$
.    .    .    .    $counter[k] += p$
```

Improvement

Get information when fault has an effect

If the fault has an effect

```
.    For each ($\delta$ , $p$)
.    .    For each $k \in [\![0, 63]\!]$
.    .    .    If $S(x \oplus k \oplus \delta) = S(x \oplus k)$
.    .    .    .    $counter[k] \; -= \; p$
```

Improvement

Get information when fault has an effect

```
If the fault has an effect
.    For each ($\delta$ , $p$)
.    .    For each $k \in [\![0, 63]\!]$
.    .    .    If $S(x \oplus k \oplus \delta) = S(x \oplus k)$
.    .    .    .    $counter[k] -= p$
```

Improvement

Get information when fault has an effect

```
If the fault has an effect
.    For each $(\delta\ ,\ p)$
.    .    For each $k \in [\![0, 63]\!]$
.    .    .    If $S(x \oplus k \oplus \delta) = S(x \oplus k)$
.    .    .    .    $counter[k] \ -= \ p$
```

Improvement

Combined algorithm

```
For each ($\delta$ , $p$)
.   For each $k \in [\![0, 63]\!]$
.   .   If $S(x \oplus k \oplus \delta) = S(x \oplus k)$
.   .   .   If the fault **has an effect**
.   .   .   .   $counter[k] -= p$
.   .   .   else
.   .   .   .   $counter[k] += p$
```

Masked implementation

How it works with masked implementation

- To build a masked Sbox S' : $\forall a$

$$S'(a \oplus z_1) = S(a) \oplus z_2$$

- Then

$$\begin{array}{lccc} \text{if} & S'(a \oplus z_1 \oplus \delta) & = & S'(a \oplus z_1) \\ & \shortparallel & & \shortparallel \\ \text{we have} & S(a \oplus \delta) \oplus z_2 & = & S(a) \oplus z_2 \end{array}$$

$$S(a \oplus \delta) = S(a)$$

How it works with masked implementation

- To build a masked Sbox S' : $\forall a$

$$S'(a \oplus z_1) = S(a) \oplus z_2$$

- Then

$$\begin{array}{ll} \text{if} & S'(a \oplus z_1 \oplus \delta) = S'(a \oplus z_1) \\ & \qquad \shortparallel \qquad\qquad\qquad \shortparallel \\ \text{we have} & S(a \oplus \delta) \oplus z_2 = S(a) \oplus z_2 \end{array}$$

$$S(a \oplus \delta) = S(a)$$

Masked implementation

How it works with masked implementation

- To build a masked Sbox S' : $\forall a$

$$S'(a \oplus z_1) = S(a) \oplus z_2$$

- Then

$$\begin{array}{lccc} \text{if} & S'(a \oplus z_1 \oplus \delta) & = & S'(a \oplus z_1) \\ & \shortparallel & & \shortparallel \\ \text{we have} & S(a \oplus \delta) \oplus z_2 & = & S(a) \oplus z_2 \end{array}$$

$$S(a \oplus \delta) = S(a)$$

Outline

1. Introduction
   - Overview of fault attacks
   - Principle of our attack
2. Application to the Data Encryption Standard
   - Data Encryption Standard
   - Attack Simulation
   - Countermeasures
3. Conclusion

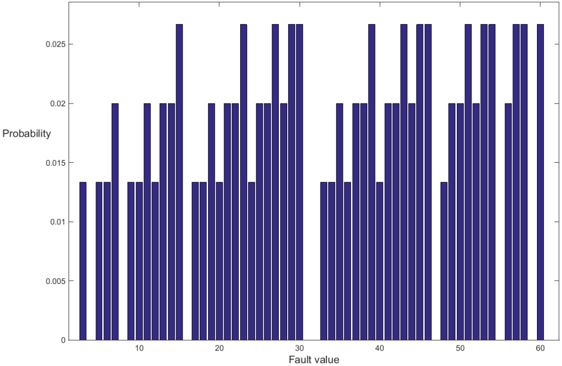
Simulation scenario

- Random plaintexts and random keys
- Theoretical fault distribution
- Mean of 1000 simulations

Simulation results

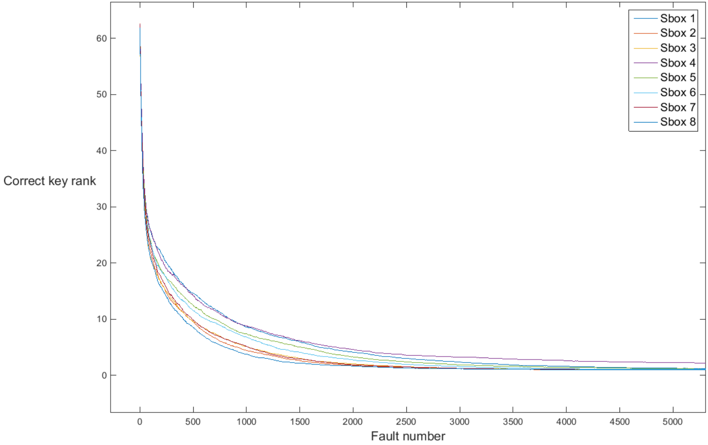
Fault Distribution

$$\begin{aligned}
HW(\delta) &= 0 && \rightarrow p = 0\\
HW(\delta) &= 1 && \rightarrow p = 0\\
HW(\delta) &= 2 && \rightarrow p = 0.013\\
HW(\delta) &= 3 && \rightarrow p = 0.02\\
HW(\delta) &= 4 && \rightarrow p = 0.027\\
HW(\delta) &= 5 && \rightarrow p = 0\\
HW(\delta) &= 6 && \rightarrow p = 0
\end{aligned}$$



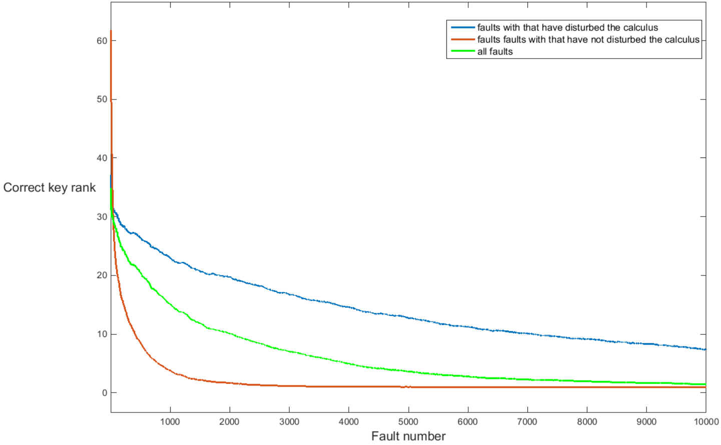
Simulation Results

Rank of the key when fault number increases



Simulation Results

Comparison between the 3 possible models



Outline

1. Introduction
   - Overview of fault attacks
   - Principle of our attack
2. Application to the Data Encryption Standard
   - Data Encryption Standard
   - Attack Simulation
   - Countermeasures
3. Conclusion

Countermeasures

Fault counter

- Do the calculus twice, compare and increase the counter in case of different results
- When the counter limit is reached : Block the device

- Our attack is theoretically possible
- The success depends on the counter limit

An error correction countermeasure

- Do the calculus three times
- Return the result obtained twice

- The attacker cannot know if a fault has an effect or not
- Our attack is no longer possible

Countermeasures

Fault counter

- Do the calculus twice, compare and increase the counter in case of different results
- When the counter limit is reached : Block the device

- Our attack is theoretically possible
- The success depends on the counter limit

An error correction countermeasure

- Do the calculus three times
- Return the result obtained twice

- The attacker cannot know if a fault has an effect or not
- Our attack is no longer possible

Countermeasures

Fault counter

- Do the calculus twice, compare and increase the counter in case of different results
- When the counter limit is reached : Block the device

- Our attack is theoretically possible
- The success depends on the counter limit

An error correction countermeasure

- Do the calculus three times
- Return the result obtained twice

- The attacker cannot know if a fault has an effect or not
- Our attack is no longer possible

Countermeasures

Fault counter

- Do the calculus twice, compare and increase the counter in case of different results
- When the counter limit is reached : Block the device

- Our attack is theoretically possible
- The success depends on the counter limit

An error correction countermeasure

- Do the calculus three times
- Return the result obtained twice

- The attacker cannot know if a fault has an effect or not
- Our attack is no longer possible

Outline

1. Introduction
   - Overview of fault attacks
   - Principle of our attack
2. Application to the Data Encryption Standard
   - Data Encryption Standard
   - Attack Simulation
   - Countermeasures
3. Conclusion

Comparison

	Safe Error	DFA	CFA	Our Attack
Works with masked implementation	✗	✓	✓	✓
Does not need to encrypt the same plaintext	✓	✗	✓	✓
Does not need to know the calculus output	✓	✗	✗	✓
Fault number ≃	100	10	100	10000

Comparison

	Safe Error	DFA	CFA	Our Attack
Works with masked implementation	✗	✓	✓	✓
Does not need to encrypt the same plaintext	✓	✗	✓	✓
Does not need to know the calculus output	✓	✗	✗	✓
Fault number ≃	100	10	100	10000

Comparison

	Safe Error	DFA	CFA	Our Attack
Works with masked implementation	✗	✓	✓	✓
Does not need to encrypt the same plaintext	✓	✗	✓	✓
Does not need to know the calculus output	✓	✗	✗	✓
Fault number $\simeq$	100	10	100	10000

Comparison

	Safe Error	DFA	CFA	Our Attack
Works with masked implementation	✗	✓	✓	✓
Does not need to encrypt the same plaintext	✓	✗	✓	✓
Does not need to know the calculus output	✓	✗	✗	✓
Fault number $\simeq$	100	10	100	10000

The End

Any Questions ?

The End

Any Questions ?