Algebraic Fault Attacks on Block Ciphers

Towards Automatic Fault Analysis

Philipp Jovanovic
(@Daeinar)

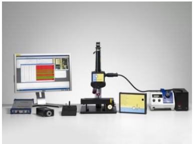
Martin Kreuzer
Ilia Polian

Department of Informatics and Mathematics
University of Passau

TRUDEVICE, Freiburg, Germany

Fault attacks

- Active side-channel attack.
- Very powerful cryptanalytic technique.
- Faults have to be very precise and the exact fault location has to be known (usually) to an attacker.
- Differential fault analysis has to be done from scratch for every cipher.

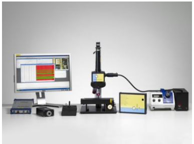


Pictures by riscure.

Q: How can algebraic cryptanalysis help?

Fault attacks

- Active side-channel attack.
- Very powerful cryptanalytic technique.
- Faults have to be very precise and the exact fault location has to be known (usually) to an attacker.
- Differential fault analysis has to be done from scratch for every cipher.



Pictures by riscure.

Q: How can algebraic cryptanalysis help?

UNIVERSITÄT PASSAU

The setting

- Attacker has access to a *black box*, implementing the to-be-analysed (block) cipher (with a fixed, unknown key k).
- Attacker can query the black box with a *plaintext* p and obtain the corresponding *ciphertext* c. He can re-query to encrypt the same p.
- Attacker is able to inject faults during a query to generate *faulty ciphertexts* c'.

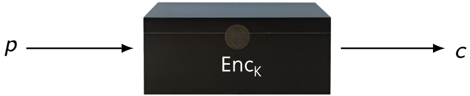
UNIVERSITÄT PASSAU

The setting

- Attacker has access to a *black box*, implementing the to-be-analysed (block) cipher (with a fixed, unknown key k).
- Attacker can query the black box with a *plaintext* p and obtain the corresponding *ciphertext* c. He can re-query to encrypt the same p.
- Attacker is able to inject faults during a query to generate *faulty ciphertexts* c'.

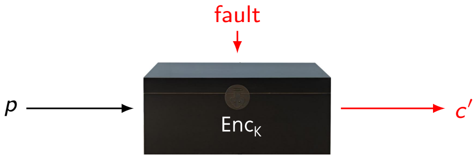
The setting

- Attacker has access to a *black box*, implementing the to-be-analysed (block) cipher (with a fixed, unknown key k).
- Attacker can query the black box with a *plaintext* p and obtain the corresponding *ciphertext* c. He can re-query to encrypt the same p.
- Attacker is able to inject faults during a query to generate *faulty ciphertexts* c'.



The setting

- Attacker has access to a *black box*, implementing the to-be-analysed (block) cipher (with a fixed, unknown key k).
- Attacker can query the black box with a *plaintext* p and obtain the corresponding *ciphertext* c. He can re-query to encrypt the same p.
- Attacker is able to inject faults during a query to generate *faulty ciphertexts* c'.



Attack phases

1. Online: Generate correct and faulty ciphertexts pairs (c_i, c_i') using plaintexts p_i, with $1 \leq i \leq n$.
2. Offline: Analyse (p_i, c_i, c_i') obtained in the online phase using algebraic cryptanalysis, in order to reconstruct the secret key k.

In this talk

- Focus on offline phase.
- Assumption: (p_i, c_i, c_i') already given.

UNIVERSITÄT PASSAU

Attack phases

1. Online: Generate correct and faulty ciphertexts pairs (c_i, c_i') using plaintexts p_i, with $1 \leq i \leq n$.
2. Offline: Analyse (p_i, c_i, c_i') obtained in the online phase using algebraic cryptanalysis, in order to reconstruct the secret key k.

In this talk

- Focus on offline phase.
- Assumption: (p_i, c_i, c_i') already given.

Attack phases

1. Online: Generate correct and faulty ciphertexts pairs (c_i, c_i') using plaintexts p_i, with $1 \leq i \leq n$.
2. Offline: Analyse (p_i, c_i, c_i') obtained in the online phase using algebraic cryptanalysis, in order to reconstruct the secret key k.

In this talk

- Focus on offline phase.
- Assumption: (p_i, c_i, c_i') already given.

Attack phases

1. Online: Generate correct and faulty ciphertexts pairs (c_i, c_i') using plaintexts p_i, with $1 \leq i \leq n$.
2. Offline: Analyse (p_i, c_i, c_i') obtained in the online phase using algebraic cryptanalysis, in order to reconstruct the secret key k.

In this talk

- Focus on offline phase.
- Assumption: (p_i, c_i, c_i') already given.

UNIVERSITÄT PASSAU

General approach

1. Let E be an *encryption function* with
   - *input bits* $x_0, \dots, x_{n-1}$,
   - *output bits* $y_0, \dots, y_{n-1}$,
   - *key bits* $k_0, \dots, k_{m-1}$

   and let $(p_0 \parallel \cdots \parallel p_{n-1}, c_0 \parallel \cdots \parallel c_{n-1})$ be a plaintext-ciphertext pair.
2. Model E using (Boolean) polynomials f_i with $0 \leq i \leq n-1$:

$$\begin{aligned} y_0 &= \quad f_0(k_0, \dots, k_{m-1}, x_0, \dots, x_{n-1}) \\ &\dots \\ y_{n-1} &= f_{n-1}(k_0, \dots, k_{m-1}, x_0, \dots, x_{n-1}) \end{aligned}$$

3. Substitute p_i for x_i and c_i for y_i.
4. Solve for k_j.

UNIVERSITÄT PASSAU

General approach

1. Let E be an *encryption function* with
   - *input bits* $x_0, \dots, x_{n-1}$,
   - *output bits* $y_0, \dots, y_{n-1}$,
   - *key bits* $k_0, \dots, k_{m-1}$

   and let $(p_0 \parallel \cdots \parallel p_{n-1}, c_0 \parallel \cdots \parallel c_{n-1})$ be a plaintext-ciphertext pair.
2. Model E using (Boolean) polynomials f_i with $0 \leq i \leq n-1$:

$$\begin{aligned} y_0 &= f_0(k_0, \dots, k_{m-1}, x_0, \dots, x_{n-1}) \\ &\dots \\ y_{n-1} &= f_{n-1}(k_0, \dots, k_{m-1}, x_0, \dots, x_{n-1}) \end{aligned}$$

3. Substitute p_i for x_i and c_i for y_i.
4. Solve for k_j.

UNIVERSITÄT PASSAU

General approach

1. Let E be an *encryption function* with
   - *input bits* $x_0, \dots, x_{n-1}$,
   - *output bits* $y_0, \dots, y_{n-1}$,
   - *key bits* $k_0, \dots, k_{m-1}$

   and let $(p_0 \parallel \cdots \parallel p_{n-1}, c_0 \parallel \cdots \parallel c_{n-1})$ be a plaintext-ciphertext pair.
2. Model E using (Boolean) polynomials f_i with $0 \leq i \leq n-1$:

$$y_0 = f_0(k_0, \dots, k_{m-1}, x_0, \dots, x_{n-1})$$
$$\dots$$
$$y_{n-1} = f_{n-1}(k_0, \dots, k_{m-1}, x_0, \dots, x_{n-1})$$

3. Substitute p_i for x_i and c_i for y_i.
4. Solve for k_j.

General approach

1. Let E be an *encryption function* with
   - *input bits* $x_0, \dots, x_{n-1}$,
   - *output bits* $y_0, \dots, y_{n-1}$,
   - *key bits* $k_0, \dots, k_{m-1}$

   and let $(p_0 \parallel \cdots \parallel p_{n-1}, c_0 \parallel \cdots \parallel c_{n-1})$ be a plaintext-ciphertext pair.
2. Model E using (Boolean) polynomials f_i with $0 \leq i \leq n-1$:

$$\begin{aligned} y_0 &= \quad f_0(k_0, \dots, k_{m-1}, x_0, \dots, x_{n-1}) \\ &\dots \\ y_{n-1} &= f_{n-1}(k_0, \dots, k_{m-1}, x_0, \dots, x_{n-1}) \end{aligned}$$

3. Substitute p_i for x_i and c_i for y_i.
4. Solve for k_j.

General approach

1. Let E be an *encryption function* with
   - *input bits* $x_0, \ldots, x_{n-1}$,
   - *output bits* $y_0, \ldots, y_{n-1}$,
   - *key bits* $k_0, \ldots, k_{m-1}$

   and let $(p_0 \parallel \cdots \parallel p_{n-1}, c_0 \parallel \cdots \parallel c_{n-1})$ be a plaintext-ciphertext pair.
2. Model E using (Boolean) polynomials f_i with $0 \leq i \leq n-1$:

$$\begin{aligned} y_0 &= \quad f_0(k_0, \ldots, k_{m-1}, x_0, \ldots, x_{n-1}) \\ &\ldots \\ y_{n-1} &= f_{n-1}(k_0, \ldots, k_{m-1}, x_0, \ldots, x_{n-1}) \end{aligned}$$

3. Substitute p_i for x_i and c_i for y_i.
4. Solve for k_j.

Advantages

- Very generic.
- Easy to set up.
- Require (theoretically) only one plaintext-ciphertext pair.
- Can be combined easily with other cryptanalytic techniques.
- Offer a trade-off: Researcher time vs. CPU time.

Disadvantages

- Often too generic.
- Difficult to include problem specific information.
- In general slower than specialised cryptanalytic methods.

Advantages

- Very generic.
- Easy to set up.
- Require (theoretically) only one plaintext-ciphertext pair.
- Can be combined easily with other cryptanalytic techniques.
- Offer a trade-off: Researcher time vs. CPU time.

Disadvantages

- Often too generic.
- Difficult to include problem specific information.
- In general slower than specialised cryptanalytic methods.

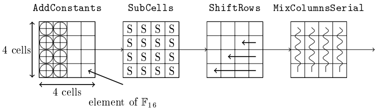
UNIVERSITÄT PASSAU

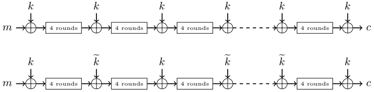
How to solve (Boolean) polynomial systems

- Brute-Force (libFES, ...)
- Gröbner Bases (PolyBoRi, ...)
- SAT Solver (MiniSat, Cryptominisat, ...)
- ...

Overview

- Substitution Permutation Network (SPN)
- 64-bit state
- 64- or 128-bit keys ("no" key schedule)
- 32 or 48 encryption rounds
- Layout similar to AES: `AddRoundKey`, `AddConstants`, `SubCells`, `ShiftRow`, `MixColumnsSerial`





- `AddRoundKey`: The key addition can be written as

$$y_i = x_i + k_i$$

with x_i input bits, y_i output bits and k_i key bits, for $i \in \{0, \ldots, 63\}$.

UNIVERSITÄT PASSAU

- `AddConstants`: Addition of the matrix

$$\begin{pmatrix} 0 & \mathtt{u} & 0 & 0 \\ 1 & \mathtt{v} & 0 & 0 \\ 2 & \mathtt{u} & 0 & 0 \\ 3 & \mathtt{v} & 0 & 0 \end{pmatrix}$$

  with $\mathtt{u} = b_5 \parallel b_4 \parallel b_3$ and $\mathtt{v} = b_2 \parallel b_1 \parallel b_0$ can be represented by the equations

$$\begin{aligned} y_i &= x_i + 1 && \text{for } i \in \{20, 35, 51, 52\} \\ y_i &= x_i + b_5 && \text{for } i \in \{6, 38\} \\ y_i &= x_i + b_4 && \text{for } i \in \{7, 39\} \\ y_i &= x_i + b_3 && \text{for } i \in \{8, 40\} \\ y_i &= x_i + b_2 && \text{for } i \in \{22, 54\} \\ y_i &= x_i + b_1 && \text{for } i \in \{23, 55\} \\ y_i &= x_i + b_0 && \text{for } i \in \{24, 56\} \\ y_i &= x_i && \text{otherwise} \end{aligned}$$

- `SubCells` and `ShiftRows`: The application of the SBox

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

and the `ShiftRows` permutation

$$\begin{aligned}\sigma = {} & (17\ 29\ 25\ 21)(18\ 30\ 26\ 22)(19\ 31\ 27\ 23)(20\ 32\ 28\ 24)\\ & (33\ 41)(34\ 42)(35\ 43)(36\ 44)(37\ 45)(38\ 46)(39\ 47)(40\ 48)\\ & (49\ 53\ 57\ 61)(50\ 54\ 58\ 62)(51\ 55\ 59\ 63)(52\ 56\ 60\ 64)\end{aligned}$$

can be combined into one set of equations

$$\begin{aligned}y_{\sigma(i_1)} &= x_{i_1}x_{i_2}x_{i_4} + x_{i_1}x_{i_3}x_{i_4} + x_{i_2}x_{i_3}x_{i_4} + x_{i_2}x_{i_3} + x_{i_1} + x_{i_3} + x_{i_4} + 1\\ y_{\sigma(i_2)} &= x_{i_1}x_{i_2}x_{i_4} + x_{i_1}x_{i_3}x_{i_4} + x_{i_1}x_{i_3} + x_{i_1}x_{i_4} + x_{i_3}x_{i_4} + x_{i_1} + x_{i_2} + 1\\ y_{\sigma(i_3)} &= x_{i_1}x_{i_2}x_{i_4} + x_{i_1}x_{i_3}x_{i_4} + x_{i_2}x_{i_3}x_{i_4} + x_{i_1}x_{i_2} + x_{i_1}x_{i_3} + x_{i_1} + x_{i_3}\\ y_{\sigma(i_4)} &= x_{i_2}x_{i_3} + x_{i_1} + x_{i_2} + x_{i_4}\end{aligned}$$

with $i_1 = 4i - 3$, $i_2 = 4i - 2$, $i_3 = 4i - 1$ and $i_4 = 4i$ for $i = 1, \ldots, 16$.

UNIVERSITÄT PASSAU

- `SubCells` and `ShiftRows`: The application of the SBox

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

and the `ShiftRows` permutation

$$\begin{aligned}\sigma = &(17\ 29\ 25\ 21)(18\ 30\ 26\ 22)(19\ 31\ 27\ 23)(20\ 32\ 28\ 24)\\ &(33\ 41)(34\ 42)(35\ 43)(36\ 44)(37\ 45)(38\ 46)(39\ 47)(40\ 48)\\ &(49\ 53\ 57\ 61)(50\ 54\ 58\ 62)(51\ 55\ 59\ 63)(52\ 56\ 60\ 64)\end{aligned}$$

can be combined into one set of equations

$$\begin{aligned}y_{\sigma(i_1)} &= x_{i_1}x_{i_2}x_{i_4} + x_{i_1}x_{i_3}x_{i_4} + x_{i_2}x_{i_3}x_{i_4} + x_{i_2}x_{i_3} + x_{i_1} + x_{i_3} + x_{i_4} + 1\\ y_{\sigma(i_2)} &= x_{i_1}x_{i_2}x_{i_4} + x_{i_1}x_{i_3}x_{i_4} + x_{i_1}x_{i_3} + x_{i_1}x_{i_4} + x_{i_3}x_{i_4} + x_{i_1} + x_{i_2} + 1\\ y_{\sigma(i_3)} &= x_{i_1}x_{i_2}x_{i_4} + x_{i_1}x_{i_3}x_{i_4} + x_{i_2}x_{i_3}x_{i_4} + x_{i_1}x_{i_2} + x_{i_1}x_{i_3} + x_{i_1} + x_{i_3}\\ y_{\sigma(i_4)} &= x_{i_2}x_{i_3} + x_{i_1} + x_{i_2} + x_{i_4}\end{aligned}$$

with $i_1 = 4i - 3$, $i_2 = 4i - 2$, $i_3 = 4i - 1$ and $i_4 = 4i$ for $i = 1, \ldots, 16$.

UNIVERSITÄT PASSAU

- `MixColumnsSerial`: The multiplication of the state with the matrix

$$M = \begin{pmatrix} 4 & 1 & 2 & 2 \\ 8 & 6 & 5 & 6 \\ B & E & A & 9 \\ 2 & 2 & F & B \end{pmatrix}$$

can be transformed to 64 equations, with an excerpt shown below:

$$\begin{aligned}
y_1 &= x_3 + x_{17} + x_{34} + x_{50} \\
y_2 &= x_1 + x_4 + x_{18} + x_{35} + x_{51} \\
y_3 &= x_1 + x_2 + x_{19} + x_{33} + x_{36} + x_{49} + x_{52} \\
y_4 &= x_2 + x_{20} + x_{33} + x_{49} \\
&\dots \\
y_{17} &= x_1 + x_4 + x_{18} + x_{19} + x_{33} + x_{35} + x_{50} + x_{51} \\
y_{18} &= x_1 + x_2 + x_{17} + x_{19} + x_{20} + x_{33} + x_{34} + x_{36} + x_{49} + x_{51} + x_{52} \\
y_{19} &= x_2 + x_3 + x_{18} + x_{20} + x_{33} + x_{34} + x_{35} + x_{50} + x_{52} \\
y_{20} &= x_3 + x_{17} + x_{18} + x_{34} + x_{36} + x_{49} + x_{50} \\
&\dots
\end{aligned}$$

Complete algebraic model of LED has 6208 equations in 6336 indeterminates.

- `MixColumnsSerial`: The multiplication of the state with the matrix

$$M = \begin{pmatrix} 4 & 1 & 2 & 2 \\ 8 & 6 & 5 & 6 \\ B & E & A & 9 \\ 2 & 2 & F & B \end{pmatrix}$$

can be transformed to 64 equations, with an excerpt shown below:

$$\begin{aligned}
y_1 &= x_3 + x_{17} + x_{34} + x_{50} \\
y_2 &= x_1 + x_4 + x_{18} + x_{35} + x_{51} \\
y_3 &= x_1 + x_2 + x_{19} + x_{33} + x_{36} + x_{49} + x_{52} \\
y_4 &= x_2 + x_{20} + x_{33} + x_{49} \\
&\dots \\
y_{17} &= x_1 + x_4 + x_{18} + x_{19} + x_{33} + x_{35} + x_{50} + x_{51} \\
y_{18} &= x_1 + x_2 + x_{17} + x_{19} + x_{20} + x_{33} + x_{34} + x_{36} + x_{49} + x_{51} + x_{52} \\
y_{19} &= x_2 + x_3 + x_{18} + x_{20} + x_{33} + x_{34} + x_{35} + x_{50} + x_{52} \\
y_{20} &= x_3 + x_{17} + x_{18} + x_{34} + x_{36} + x_{49} + x_{50} \\
&\dots
\end{aligned}$$

Complete algebraic model of LED has 6208 equations in 6336 indeterminates.

UNIVERSITÄT PASSAU

General approach: Algebraic Attacks

1. Let E be an *encryption function* with
   - *input bits* $x_0, \ldots, x_{n-1}$,
   - *output bits* $y_0, \ldots, y_{n-1}$,
   - *key bits* $k_0, \ldots, k_{m-1}$

   and let $(p_0 \parallel \cdots \parallel p_{n-1}, c_0 \parallel \cdots \parallel c_{n-1})$ be a plaintext-ciphertext pair.
2. Model E using (Boolean) polynomials f_i with $0 \leq i \leq n-1$:

$$\begin{aligned} y_0 &= \quad f_0(k_0, \ldots, k_{m-1}, x_0, \ldots, x_{n-1}) \\ &\ldots \\ y_{n-1} &= f_{n-1}(k_0, \ldots, k_{m-1}, x_0, \ldots, x_{n-1}) \end{aligned}$$

3. Substitute p_i for x_i and c_i for y_i.
4. Solve for k_j.

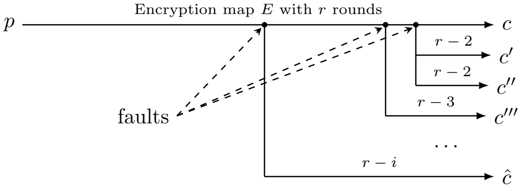
General approach: Algebraic Fault Attacks

1. Let E be an *encryption function* with
   - *input bits* $x_0, \ldots, x_{n-1}$,
   - *output bits* $y_0, \ldots, y_{n-1}$,
   - *key bits* $k_0, \ldots, k_{m-1}$

   and let $(p_0 \parallel \cdots \parallel p_{n-1}, c_0 \parallel \cdots \parallel c_{n-1})$ be a plaintext-ciphertext pair.
2. Model E using (Boolean) polynomials f_i with $0 \leq i \leq n-1$:

$$\begin{aligned} y_0 &= \quad f_0(k_0, \ldots, k_{m-1}, x_0, \ldots, x_{n-1}) \\ &\ldots \\ y_{n-1} &= f_{n-1}(k_0, \ldots, k_{m-1}, x_0, \ldots, x_{n-1}) \end{aligned}$$

3. Substitute p_i for x_i and c_i for y_i.
4. **Model fault injections algebraically.**
5. Solve for k_j.

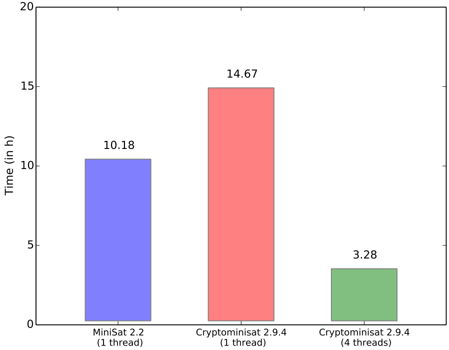


Consider a fault injection in round $r-2$:

- It can be modelled as
$$x_i' = x_i + e_i'$$
with x_i correct intermediate state, e_i' faulty variables and x_i' faulty state.
- Model $y_i' = f_i'(k_j, x_i')$ where f_i' are the polynomials of the last 2 rounds using new variables (only key variables k_j are the same).
- Substitute c_i' for y_i' and append all new (fault) equations to the system of equations of the encryption map E.

Results

Injecting a single fault in round 30 is sufficient to break LED64.

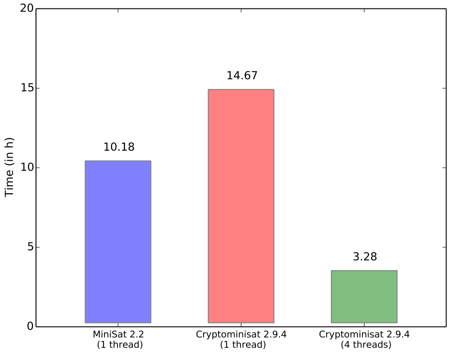


Note: The direct approach (using DFA) is much faster (but less generic). The reconstruction of the key only takes a couple of minutes.

UNIVERSITÄT PASSAU

Results

Injecting a single fault in round 30 is sufficient to break LED64.



Note: The direct approach (using DFA) is much faster (but less generic). The reconstruction of the key only takes a couple of minutes.

Small Scale AES

Overview

- Framework to construct smaller (i.e. less complex) variants of AES.
- Suitable for step-by-step (algebraic) cryptanalysis.
- Integrated into `Sage`: `http://sagemath.org/`
- Notation: `AES(n,r,c,e)` with n #rounds, r #rows, c #columns and e word size.

0

0
1

0	2
1	3

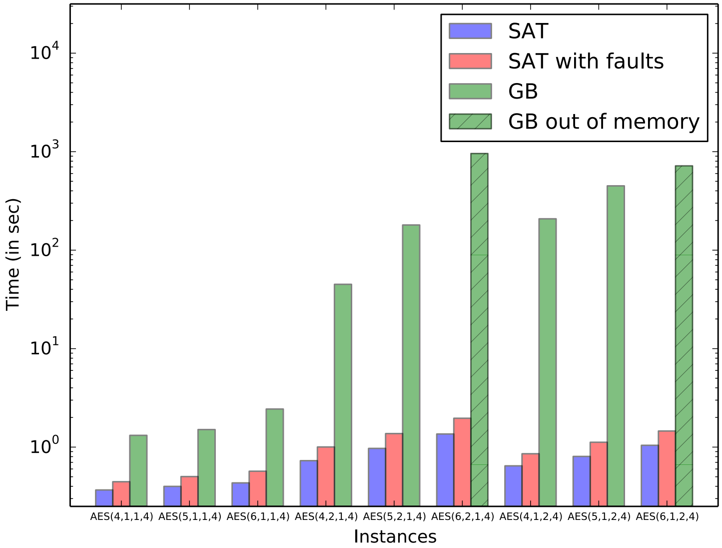
0	4
1	5
2	6
3	7

0	2	4	6
1	3	5	7

0	4	8	12
1	5	9	13
2	6	10	14
3	7	11	15

Various state sizes ($r \cdot c$) of Small Scale AES.

Algebraic (Fault) Attack on Small Scale AES



Conclusion

1. Presented an algebraic framework to execute fault analysis.
2. Inherits properties of algebraic attacks:
   - Generic.
   - Easy to adapt for attacking new cipher designs.
   - Offers trade-off: Researcher time vs. CPU time.
   - Less performant than specialised attacks.
3. Showed applications to LED and Small Scale AES.

Thank you for your attention!

Questions?

Philipp Jovanovic
jovanovic@fim.uni-passau.de