DFA Mechanism on the AES Key Schedule

Junko Takahashi, Toshinori Fukunaga and Kimihiro Yamakoshi

NTT Information Sharing Platform Laboratories, JAPAN
Outline

- Motivation
- Our results
- Analysis of DFA mechanism
- Our attack
- Conclusions
Motivation

Previous studies have not addressed general attack approach for DFA against AES key schedule

- What is the general approach?
- Is there a more efficient attack than existing ones?
Our results

■ Previous studies
  - No general expression of attack
  - Complicated simultaneous equations must be solved to obtain keys

■ Our study
  - We found that DFA can be clearly represented, if seen from two sides
  - Only simple expressions and attack rules needed
Our results

- Our attack
- Peacham’s attack
- Chen’s attack

Graph showing retrieved key information (bits) vs. # of fault injection point.
Our results

![Graph showing retrieval of key information vs. number of pairs for different attacks.]

- **Our attack**: Shows a linear increase in retrieved key information with the number of pairs.
- **Peacham’s attack**: Demonstrates a more gradual increase.
- **Chen’s attack**: Exhibits a sharp increase at the beginning, followed by a plateau.

The graph illustrates the effectiveness of different attacks in retrieving key information, with our attack outperforming the others in terms of efficiency.
Motivation

Our results

Analysis of DFA mechanism

Our attack

Conclusions
DFA against AES key schedule

- States calculated by correct and faulty outputs must be equal, \( m = m' \)
- Solve simultaneous equations to obtain keys

AES-128

States calculated by correct and faulty outputs must be equal, \( m = m' \). Solve simultaneous equations to obtain keys.
Attack assumptions

- Attacker can corrupt any byte(s) of the round key, but he can not choose the corrupted value of the byte(s) as he likes.

- Faults are not injected into byte(s) of the same row of the 9th round.

\[ \varepsilon_{i,j} = K_{i,j} \oplus \tilde{K}_{i,j} \] : error values (difference between correct and faulty keys)
Relation between $m'$ and output

- Each byte of \( \begin{array}{c} \text{equation} \\ m = m' \end{array} \) represents a one-to-one correspondence with keys and outputs.
Fault propagation in AES-128

9th round

10th round

\( R_{con_9} \)

\( R_{con_{10}} \)
Fault propagation in AES-128

9th round
- RotWord
- SubWord
- $R_{con_9}$

10th round
- RotWord
- SubWord
- $R_{con_{10}}$
**Classification : 8 patterns**

Each byte of 8 patterns \( \{ \) can be classified into

- type F
- type E
- type D
- type C
- type B
- type A

\( \text{equation } m = m' \) \( \text{related to } \tilde{K}^{10} \) *Not used in analysis

<table>
<thead>
<tr>
<th>( \tilde{K}^9 )</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>*</td>
<td>type A</td>
<td>*</td>
</tr>
<tr>
<td>type C</td>
<td>type D</td>
<td>type E</td>
<td>type F</td>
</tr>
</tbody>
</table>
m=m' assigned to one of 8 patterns

Matrix consisting of 16 equations:

\[ m_{i,j} = m'_{i,j} \]

Related to \( \tilde{K}^{10} \):

\[
\begin{array}{|c|c|c|c|}
\hline
& \text{type A} & \text{type B} & \\
\hline
\text{-} & \text{-} & \text{-} & \\
\hline
\text{type C} & \text{type D} & \text{type E} & \text{type F} \\
\hline
\end{array}
\]
Our idea

- 16 equations of $m_{i,j} = m'_{i,j}$ are classified into 8 patterns
- Some types are related
- Attack utilizes position of types and known values during the attack

Matrix consisting of 16 equations: $m_{i,j} = m'_{i,j}$

```
A A F D
A A D F
B A D D
A B D D
```
Proposed 7 attack rules

- General expression of equation: \( m_{i,j} = m'_{i,j} \)

\[
K_{i,j} \oplus S^{-1}[Q_{i,j} \oplus S[K_{i+1(\text{mod}4),3}] \oplus y_{i,j}] = \tilde{K}_{i,j} \oplus S^{-1}[\tilde{Q}_{i,j} \oplus S[\tilde{K}_{i+1(\text{mod}4),3}] \oplus \tilde{y}_{i,j}]
\]

- In the case of type A byte on \((i, j)\):

\[
K_{i,j} \oplus S^{-1}[Q_{i,j} \oplus S[K_{i+1(\text{mod}4),3}] \oplus y_{i,j}] = K_{i,j} \oplus S^{-1}[Q_{i,j} \oplus S[K_{i+1(\text{mod}4),3} \oplus \varepsilon_{i+1(\text{mod}4),3}] \oplus \tilde{y}_{i,j}]
\]

\[
S[K_{i+1(\text{mod}4),3}] \oplus y_{i,j} = S[K_{i+1(\text{mod}4),3} \oplus \varepsilon_{i+1(\text{mod}4),3}] \oplus \tilde{y}_{i,j}
\]

**attack rule.2**

If we know \( \varepsilon_{i+1(\text{mod}4),j} \) below type A, we can obtain \( K_{i+1(\text{mod}4),3} \) in the most right byte of the row below type A. We have to use 2 pairs of correct and faulty ciphertexts to determine \( K_{i+1(\text{mod}4),3} \).
Motivation

Our results

Analysis on DFA mechanism

Our attack

Conclusions
Our attack with one fault injection

Matrix consisting of 16 equations: \( m_{i,j} = m'_{i,j} \)

<table>
<thead>
<tr>
<th>A</th>
<th>A</th>
<th>F</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
<td>D</td>
<td>F</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>D</td>
<td>D</td>
</tr>
</tbody>
</table>
Attack procedure

Matrix consisting of 16 equations: $m_{i,j} = m'_{i,j}$

Error values \( \mathcal{E} \)

9th round key \( K \)
Attack procedure

apply to rule 2

matrix consisting of 16 equations: $m_{i,j} = m_{i,j}'$

\[
\begin{array}{cccc}
A & A & F & D \\
A & A & D & F \\
B & A & D & D \\
A & B & D & D \\
\end{array}
\]

error values $\mathcal{E}$

9th round key $K$

NTT Information Sharing Platform Laboratories
Attack procedure

apply rule.3 and rule.5

matrix consisting of 16 equations: $m_{i,j} = m'_{i,j}$

error values $\mathcal{E}$

9th round key $K$
**Attack procedure**

Apply rule 3 and rule 5.

Matrix consisting of 16 equations: \( m_{i,j} = m'_{i,j} \)

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>A</th>
<th>F</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
<td>A</td>
<td>D</td>
<td>F</td>
</tr>
<tr>
<td>A</td>
<td>A</td>
<td>A</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
</tbody>
</table>

Error values \( \mathcal{E} \)

9th round key \( K \)

XOR operations
Attack procedure

apply rule 1

matrix consisting of 16 equations: $m_{i,j} = m'_{i,j}$

error values $\mathcal{E}$

9th round key $K$

NTT Information Sharing Platform Laboratories
Apply rule 2 and rule 3.

Matrix consisting of 16 equations: $m_{i,j} = m'_{i,j}$

Error values $\mathcal{E}$

9th round key $K$

XOR
Attack procedure

apply rule.2

error values $\mathcal{E}$

matrix consisting of 16 equations: $m_{i,j} = m'_{i,j}$

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>A</th>
<th>F</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
<td>A</td>
<td>D</td>
<td>F</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
</tbody>
</table>

9th round key $K$
We can obtain information equivalent to 80 bits of key.

matrix consisting of 16 equations: \( m_{i,j} = m'_{i,j} \)

We can obtain information equivalent to 80 bits of key.

apply rule.3 and rule.5

error values \( \mathcal{E} \)

9th round key \( K \)
How to retrieve a complete key

- 80-bit
  - 48-bit brute-force (1 year/3.0GHz PC)
  - 16-bit brute-force (<1 sec/3.0GHz PC)

- 112-bit
  - 2 pairs

- 128-bit
  - 3 pairs

9th round

Image: NTT Information Sharing Platform Laboratories
Comparison to existing attacks

retrieved key information (bits)

# of fault injection point

0
1
2
3
4

0
32
64
96
112
128

our attack

Peacham’s attack

Chen’s attack

Peacham’s attack
Chen’s attack

our attack
Comparison to existing attacks

Chen’s attack

Peacham’s attack
Motivation

Our results

Analysis on DFA mechanism

Our attack

Conclusions
Conclusions

- Analysis of DFA mechanism
  - We found that DFA against the AES key schedule can be clearly represented, when seen from two sides,
    - how each key byte is affected by fault injection
    - position of each type affected by fault injection
  - We proposed how to get the complete key with the position of types read from simple expressions and attack rules.

- Efficient attack
  - It is much more efficient.
    - 2-pairs needed with 48-bit brute-force search
    - 7-pairs needed without brute-force search
Thank you very much for your attention!!