BLIND FAULT ATTACK AGAINST SPN CIPHERS
FDTC 2014

Roman Korkikian, Sylvain Pelissier, David Naccache
September 23, 2014
IN BRIEF

- Substitution Permutation Networks (SPN)
- Fault attacks
- Blind fault attack against SPN ciphers
- Results
- Questions
BLOCK CIPHER

\[ M \xrightarrow{E} C \quad \text{with key} \quad K \]
Block cipher construction which uses a sequence of invertible transformations:

- **Substitution stage or S-box** $S$ (Usually 4-bit or 8-bit S-boxes)
- **Permutation stage** $P$
- **Key mixing operation** $A$

Structure used by AES, LED, SAFER++, ...
SUBSTITUTION PERMUTATION NETWORKS

Figure : A typical SPN-based block cipher
DIFFERENTIAL FAULT ANALYSIS

1. Inject faults in the last rounds of a block cipher
2. Collect pairs \((C_1, \tilde{C}_1), (C_2, \tilde{C}_2), \ldots\)
3. Apply a statistical method on these pairs and retrieve the key \(K\).

The input messages do not need to be known:

\[ M_1 \rightarrow (C_1, \tilde{C}_1), \]
\[ M_2 \rightarrow (C_2, \tilde{C}_2), \]
\[ \ldots \]
DIFFERENTIAL FAULT ANALYSIS

Plaintext

A[\(K_0\)]: \(K_0\) mixing

\(S_{1,1}\) \(S_{1,j}\) \(S_{1,m}\)

P: diffusion

A[\(K_1\)]: \(K_1\) mixing

\(S_{r,1}\) \(S_{r,j}\) \(S_{r,m}\)

P: diffusion

A[\(K_r\)]: \(K_r\) mixing

\(S_{R,1}\) \(S_{R,j}\) \(S_{R,m}\)

A[\(K_R\)]: \(K_R\) mixing

Ciphertext

Round 1

Round \(r\)

Round \(R\)
The same idea can be applied for inputs:

1. Inject faults in the first rounds
2. Find colliding input pairs \((M_1, \tilde{M}_1), (M_2, \tilde{M}_2), \ldots\)
3. Apply a statistical method on these pairs and retrieve the key \(K\).

The output messages do not have to be known:

\[(M_1, \tilde{M}_1) \rightarrow C_1\]
\[(M_2, \tilde{M}_2) \rightarrow C_2\]
\[
\ldots
\]

But they have to be somehow compared (equality check \(O(C_1 = C_2)\)).
# CURRENT ATTACKS FOR AES

<table>
<thead>
<tr>
<th>Attack</th>
<th>Rounds</th>
<th>Plaintexts</th>
<th>Ciphertexts</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFA</td>
<td>6-10</td>
<td>Unknown</td>
<td>Known</td>
</tr>
<tr>
<td>CFA</td>
<td>1-2</td>
<td>Known</td>
<td>Unknown*</td>
</tr>
</tbody>
</table>

* equality test check \( \mathcal{O}(C_1 = C_2) \)
BLIND FAULT ATTACK

What if input and output values are not directly accessible?
EXAMPLES

- Input and output whitening
- Cascade encryption
- Hardware security module
## OUR CONTRIBUTION

<table>
<thead>
<tr>
<th>Attack</th>
<th>Rounds</th>
<th>Plaintexts</th>
<th>Ciphertexts</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFA</td>
<td>6-10</td>
<td>Unknown</td>
<td>Known</td>
</tr>
<tr>
<td>CFA</td>
<td>1-2</td>
<td>Known</td>
<td>Unknown*</td>
</tr>
<tr>
<td>BFA</td>
<td>Any</td>
<td>Unknown</td>
<td>Unknown*</td>
</tr>
</tbody>
</table>

* equality test check ($\mathcal{O}(C_1 = C_2)$)
ASSUMPTIONS

1. A multi-bit set or reset fault can be injected to an internal byte/nibble $X$ of a SPN block cipher.
2. Unknown plaintexts can be encrypted several times.
3. The different faulted or correct outputs can be compared pairwise without revealing their values (pairwise equality check $O(C_1 = C_2)$).
BLIND FAULT ATTACK OVERVIEW

1. For each plaintext:
   1.1 Introduce faults during a round execution and compare the different outputs.
   1.2 From the number of different faulted outputs determine the Hamming weights of an algorithm’s internal state.

2. For each possible key candidate:
   2.1 Perform a key search to recover a key byte/nibble.
FAULT MODEL

- Multi-bit reset fault

- Multi-bit set fault
These fault models have been observed in practice:

- Laser fault injection in SRAM: [Roscian, Sarafianos, Dutertre, Tria] FDTC 2013
- Electromagnetic glitch fault injections: [Moro, Dehbaoui, Heydemann, Robisson, Encrenaz] FDTC 2013
HAMMING WEIGHT GUESS

\[ M \xrightarrow{00001001} \]

\[ \mathcal{HW}(X) = 2 \]

\[ \begin{align*}
00001000 & \rightarrow \tilde{C}_1 \\
00000000 & \rightarrow \tilde{C}_2 \\
00000001 & \rightarrow \tilde{C}_3 \\
00001001 & \rightarrow C
\end{align*} \]

4 different outputs

\[ 2^{\mathcal{HW}(X)} \]
The number of faults injections can be minimized when considered as an "occupancy problem".

- The probability that after $\ell$ fault injections, $Y_\ell$ different possible ciphertexts (among $\lambda = 2^{\mathcal{HW}(X)}$) are received can be considered as the probability that $Y_\ell$ out of $\lambda$ bins are occupied after throwing randomly $\ell$ balls.
OCCUPANCY PROBLEM

\[
\Pr(Y_\ell = \kappa) = \begin{cases} 
\frac{\lambda!\alpha_{\kappa,\ell}}{(\lambda-\kappa)!\lambda^{\ell}} & \kappa \in \{1, \ldots, \min(\lambda, \ell)\} \\
0 & \text{else}
\end{cases}
\]

\(\hat{\lambda}\) with maximum likelihood is assumed as correct:

\[
\hat{\lambda} = \arg \max_{\lambda_i} \Pr(Y_\ell = \kappa|\lambda_i)
\]
15 faults give a 99% success probability for a 4-bit variable.
62 faults give a 99% success probability for a 8-bit variable.
TARGETED STATES

Round r

\[ X_{r,1}^{SP} \rightarrow A[K_{r,1}] \rightarrow S_{r+1,1} \rightarrow X_{r+1,1}^{S} \]

\[ X_{r,j}^{SP} \rightarrow A[K_{r,j}] \rightarrow S_{r+1,j} \rightarrow X_{r+1,j}^{S} \]

\[ X_{r,M}^{SP} \rightarrow A[K_{r,M}] \rightarrow S_{r+1,M} \rightarrow X_{r+1,M}^{S} \]

Round r+1

\[ \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \]
TARGETED STATES

\[ h_r = \mathcal{HW}(X^{SP}_{r,1}) \]

\[ h_{r+1} = \mathcal{HW}(S_{r+1,1} \circ A | K(X^{SP}_{r,1})) \]
KEY SIFTING

- For each key byte/nibble candidate $K_i$:
  - if $\exists X \ H\!W(X) = h_r$ and $H\!W(S_{r+1,j} \circ A |_{K_i}(X)) = h_{r+1}$:
    - $K_i$ is discarded from the candidate list.

$\Rightarrow$ A lot of Hamming weight pairs needed to reduce the candidate list

$\Rightarrow$ Can be improved with key likelihood estimation.
It was determined that Hamming weight distribution of key mixing and S-box is unique for the tested ciphers:
1. Before the attack, the Hamming weight distributions are precomputed for each key candidate.
2. The Euclidean distance between the distribution of the recovered Hamming weight pairs and the precomputed distributions is computed for all remaining key candidates.
3. The key candidate with the minimal distance is assumed to be correct.
**Table: Specification of operation for different ciphers**

<table>
<thead>
<tr>
<th>Cipher</th>
<th>Exact operation</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>LED</td>
<td>$X_{r+1,j}^S = S$</td>
<td>$K_{r,j} \oplus X_{r,j}^{SP}$</td>
</tr>
<tr>
<td>AES</td>
<td>$X_{r+1,j}^S = S$</td>
<td>$K_{r,j} \oplus X_{r,j}^{SP}$</td>
</tr>
<tr>
<td>SAFER++</td>
<td>$X_{r+1,j}^S = S$</td>
<td>$K_{r,j} + X_{r,j}^{SP}$</td>
</tr>
</tbody>
</table>
LED SIMULATION

- Number of Hamming weight pairs
- Key recovery success rate

Success rate after key sifting
Success rate after key sifting and key likelihood estimation

Key recovery success rate vs. Number of Hamming weight pairs:
- Dashed blue line: Success rate after key sifting
- Solid red line: Success rate after key sifting and key likelihood estimation
RESULTS

Number of faults used to recover a key byte/nibble:

<table>
<thead>
<tr>
<th>Cipher</th>
<th># plaintexts</th>
<th># faults per plaintext</th>
<th>Total # faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>LED</td>
<td>50</td>
<td>40</td>
<td>2,000</td>
</tr>
<tr>
<td>AES</td>
<td>250</td>
<td>120</td>
<td>30,000</td>
</tr>
<tr>
<td>SAFER++</td>
<td>200</td>
<td>120</td>
<td>24,000</td>
</tr>
</tbody>
</table>
Fault attacks are feasible even when input and output messages are not known but ciphertext equality check is available.

Fault attacks can be applied against any SPN round.

Fault model is generic and has been observed in practice.

The total number of faults to recover a key is the price to pay for blindness (480,000 for a complete AES key).
FUTURE DEVELOPMENTS

- New methods to reduce the number of required fault injections.
- Hamming weight distribution theory.
- Results and problems when applied in practice.
HAMMING WEIGHT PROBABILITY DISTRIBUTION

\[ \Pr_k \left[ \mathcal{HW}(x), \mathcal{HW} (S_{r+1,j} \circ A \mid_k (x)) \right] \]