Related-key differential analysis of the AES

Margot Funk

Joint work with Christina Boura and Patrick Derbez

1Paris-Saclay University - Versailles University
2University of Rennes 1

October 2023
Differential cryptanalysis

• Exploits a high probability differential distinguisher

\[
\Delta_{\text{in}} \xrightarrow{F_k} \Delta_{\text{out}} \xrightarrow{F_k} \Delta_{\text{in}}
\]

single-key differential
Differential cryptanalysis

- Exploits a high probability differential distinguisher

\[ x \xrightarrow{\Delta_{\text{in}}} F_k(x) \xrightarrow{\Delta_{\text{out}}} F_k(x) + \Delta_{\text{in}} \]

related-key differential
# AES differential trails

- active S-boxes, max DP of the AES S-box = $2^{-6}$
  - bound on the differential probability

4-round truncated differential trail of AES with 25 active S-boxes: $p \leq 2^{-25 \times 6}$

Single-key model VS Related-key model

- **Single-key:** simple and powerful security proofs.
- **Related-key:** much weaker.

*Related-key attacks on the full AES-192 and AES-256, Biryukov et al., 2009*
Modeling the AES truncated trails

Use of **generic solvers** (Wu and Wang, 2009 and Mouha et al., 2011)

\[
\text{Problem of finding differential trails} \rightarrow \text{Model} \rightarrow \text{Solver}
\]

**Model**

- **Variables**: byte of the truncated trail \( \leftrightarrow \text{var} \in \{0, 1\} \subset \mathbb{Z} \).
- **Objective function**: minimize the sum of variables that pass through an S-box.
- **Set of constraints** (*ex: linear inequalities*)
Modeling the AES truncated trails

Basic propagation rules ...

XOR of two bytes

\[ \oplus \]

\[ \oplus \]

\[ \oplus \]\n
\{\}

\[\rightarrow\]

\[\rightarrow\]

\[\rightarrow\]

\[\rightarrow\]

\[\rightarrow\]

... do not necessarily lead to valid truncated trails.

Ex: \[\rightarrow\] \[\rightarrow\] is not instantiable.
Modeling the AES truncated trails

Gérault et al. (2018, 2020), Rouquette et al. (2022)

• Use a Constraint Programming (CP) solver.

• Few seconds or minutes for most of the instances.

• Outperforms previous works:
  - Branch & bound (Biryukov et al., 2010): several weeks for AES–192,
  - Dynamic programming for AES–128 (Fouque et al., 2013).
Dynamic programming for differential bounds on AES
Dynamic programming for differential bounds

Fouque et al., CRYPTO 2013

• Generic tool based on dynamic programming.
• Complexity easy to understand.
• Application for AES-128: 30 minutes, 60 GB.

Our work

• Extend the work of Fouque et al. (2013) for all versions of AES.
• Running time comparable to that of the CP approach of Gérault et al. (2018, 2020).
Principle of the dynamic programming algorithm of [FJP13]
Principle of the dynamic programming algorithm of [FJP13]
Principle of the dynamic programming algorithm of [FJP13]

# active S-boxes

8 / 17
Principle of the dynamic programming algorithm of [FJP13]
Principle of the dynamic programming algorithm of [FJP13]
Adapting the dynamic programming algorithm of [FJP13]

- Reduce the memory complexity

Truncated difference

<table>
<thead>
<tr>
<th># Truncated differences</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES-128: $2^{32}$</td>
</tr>
<tr>
<td>AES-192: $2^{40}$     X</td>
</tr>
<tr>
<td>AES-256: $2^{48}$     X</td>
</tr>
</tbody>
</table>

Compressed difference

<table>
<thead>
<tr>
<th># Compressed differences</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES-128: $2^{18.58}$</td>
</tr>
<tr>
<td>AES-192: $2^{23.22}$</td>
</tr>
<tr>
<td>AES-256: $2^{27.86}$</td>
</tr>
</tbody>
</table>
Adapting the dynamic programming algorithm of [FJP13]

- Integrate constraints over several rounds in a second step.

```plaintext
\min \rightarrow \text{bound not necessarily tight!}
```
Adapting the dynamic programming algorithm of [FJP13]

• Integrate constraints over several rounds in a second step.

1. Search for a **compressed trail** with \( n \) active S-boxes.
   • depth-first search approach in the backward direction
   • check some linear relations

2. Turn it, if possible, into a **truncated trail**.
Complexity and running time

• For the dynamic programming phase:

<table>
<thead>
<tr>
<th></th>
<th>Time complexity</th>
<th>Memory (Bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES-128</td>
<td>$r \times 2^{22.89}$</td>
<td>$(9r - 9) \times 2^{18.58}$</td>
</tr>
<tr>
<td>AES-192</td>
<td>$r \times 2^{27.53}$</td>
<td>$(3r - 3) \times 2^{23.22}$</td>
</tr>
<tr>
<td>AES-256</td>
<td>$r \times 2^{32.18}$</td>
<td>$(3r - 4) \times 2^{27.86}$</td>
</tr>
</tbody>
</table>
## Complexity and running time

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Rounds</th>
<th>Min nb of active S-boxes</th>
<th># trails</th>
<th>Real time (User time)</th>
<th>Time [RGMS22]</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES-128</td>
<td>4</td>
<td>12</td>
<td>1</td>
<td>1s (1s)</td>
<td>31s</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>17</td>
<td>81</td>
<td>40s (5m6s)</td>
<td>2h24m24s</td>
</tr>
<tr>
<td>AES-192</td>
<td>6</td>
<td>10</td>
<td>3</td>
<td>1s (8s)</td>
<td>17s</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>14</td>
<td>2</td>
<td>1s (9s)</td>
<td>46s</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>18</td>
<td>4</td>
<td>1m35s (12m37s)</td>
<td>1m23s</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>24</td>
<td>6</td>
<td>4d5h (20d4h)</td>
<td>30m</td>
</tr>
<tr>
<td>AES-256</td>
<td>11</td>
<td>20</td>
<td>4</td>
<td>42s (4m30s)</td>
<td>5m30s</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>20</td>
<td>4</td>
<td>42s (4m16s)</td>
<td>4m37s</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>24</td>
<td>4</td>
<td>52s (5m24s)</td>
<td>7m</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>24</td>
<td>4</td>
<td>50s (5m5s)</td>
<td>9m17s</td>
</tr>
</tbody>
</table>

(1) 8-core Ryzen 3700X processor, 3.6 GHz, 32 GB of RAM
(2) 1-core Intel Xeon E5-2630 v4, 3.10 Ghz with 10 cores under a Linux Debian 10 (Buster), 16 GB of RAM (default JVM configuration)
Alternative permutation-based key schedules for AES
Related works

Permutation-based key schedule for AES-128

$K_i$ $\rightarrow$ $K_{i+1}$

• Khoo et al., ToSC 2017
• Derbez et al., SAC 2018

: efficient in software and hardware.
Double MILP model

**Goal:** find a permutation ensuring \( b \) active S-boxes.

Generate \( P \)

Evaluate \( P \)

Ensure that \( P \) is a permutation.
Double MILP model

**Goal**: find a permutation ensuring $b$ active S-boxes.

Ensure that $P$ is a permutation.
Double MILP model

**Goal:** find a permutation ensuring \( b \) active S-boxes.

- **Generate** \( P \)
  - Ensure that \( P \) is a permutation.
- **Evaluate** \( P \)
  - No solution with less than \( b \) active S-boxes
Double MILP model

**Goal:** find a permutation ensuring \( b \) active S-boxes.

Generate \( P \)

Ensure that \( P \) is a permutation.

Remove the *bad subkeys pattern* \((K_1, \ldots, K_n)\).

Evaluate \( P \)

A trail with less than \( b \) active S-boxes.
Removing a bad subkeys pattern

- **1st idea**: forbid the exact trail.

\[
\begin{array}{cccc}
0 & 1 & 2 & 3 \\
4 & 5 & 6 & 7 \\
8 & 9 & 10 & 11 \\
12 & 13 & 14 & 15 \\
\end{array}
\quad P
\quad \begin{array}{cccc}
0 & 1 & 2 & 3 \\
4 & 5 & 6 & 7 \\
8 & 9 & 10 & 11 \\
12 & 13 & 14 & 15 \\
\end{array}
\quad P
\quad \begin{array}{cccc}
0 & 1 & 2 & 3 \\
4 & 5 & 6 & 7 \\
8 & 9 & 10 & 11 \\
12 & 13 & 14 & 15 \\
\end{array}
\]

At most 3 of these equalities should be true.

\[
P(0) = 2 \quad P(1) = 14 \\
P(2) = 3 \quad P(14) = 15
\]
Removing a bad subkeys pattern

- 2\textsuperscript{nd} idea: forbid the subkeys pattern.

\begin{align*}
\begin{array}{cccc}
0 & 1 & 2 & 3 \\
4 & 5 & 6 & 7 \\
8 & 9 & 10 & 11 \\
12 & 13 & 14 & 15 \\
\end{array}
\quad P
\quad \quad 
\begin{array}{cccc}
0 & 1 & 2 & 3 \\
4 & 5 & 6 & 7 \\
8 & 9 & 10 & 11 \\
12 & 13 & 14 & 15 \\
\end{array}
\quad P
\quad \quad 
\begin{array}{cccc}
0 & 1 & 2 & 3 \\
4 & 5 & 6 & 7 \\
8 & 9 & 10 & 11 \\
12 & 13 & 14 & 15 \\
\end{array}
\end{align*}

At most 3 of these equalities should be true.

- $P(0) = 2 \quad P(1) = 14$
- $P(1) = 2 \quad P(0) = 14$
- $P(2) = 3 \quad P(14) = 15$
- $P(14) = 3 \quad P(2) = 15$
## Results

<table>
<thead>
<tr>
<th>Rounds</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES-128</td>
<td>5</td>
<td>12</td>
<td>17</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Khoo et al.</td>
<td>5</td>
<td>10</td>
<td>14</td>
<td>19</td>
<td>23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P_{128}$</td>
<td>5</td>
<td>10</td>
<td>14</td>
<td>20</td>
<td>22</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AES-192</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>10</td>
<td>14</td>
<td>18</td>
<td>24</td>
<td>29</td>
</tr>
<tr>
<td>$P_{192}$</td>
<td>1</td>
<td>5</td>
<td>10</td>
<td>13</td>
<td>17</td>
<td>22</td>
<td>25</td>
<td>28</td>
</tr>
<tr>
<td>AES-256</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>16</td>
</tr>
<tr>
<td>$P_{256}$</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
<td>14</td>
<td>16</td>
<td>22</td>
<td>26</td>
</tr>
</tbody>
</table>
Conclusion and perspectives

The **key schedule** is one of the **less understood components** in block ciphers.

**Perspectives**

- Clarify the security goals.
- Search for key schedules that are not permutations of bytes.