Cube Attacks on Stream Ciphers Based on Division Property

Chaoyun Li

ESAT-COSIC, KU Leuven

12-10-2017, Crete
Plan

1. Cube Attack: An Introduction
2. Cube Attacks with Division Property
3. Our Results
4. Conclusion and Future work
Motivation

Symmetric key ciphers for FHE, MPC, ...
- Trivium [Cannière-Preneel ’07]
- LowMC [Albrecht et al. ’15]
- Kreyvium [Canteaut et al. ’16]

Low Multiplicative Complexity (MC) is crucial
Minimize the number of ANDs and multiplicative depth

Our goal
Cube attacks on low MC ciphers
Low MC stream ciphers

Trivium [Cannière-Preneel ’07]
Low MC stream ciphers

Kreyvium [Canteaut et al. ’16]
Cube attacks [Dinur-Shamir ’09]

- Extension of Higher Order Differential Attack and Algebraic Attacks
- Chosen plaintext key recovery attack
  - Keyed hash functions
  - Stream ciphers
  - Block ciphers
  - MAC algorithms
- Powerful for primitives with low-degree component
  - Stream ciphers based on low-degree NFSR
  - Permutations with only a few XORs and ANDs
Cube attack in a nutshell

Preprocessing:
- Sum over outputs of subspaces over chosen public variables
- Store equations between sums and secret variables

Online:
- Evaluate sums over outputs of chosen plaintexts
- Recover key bits by solving equations

Dinur-Shamir attack only needs **blackbox** access to the cipher
Main observation

Cube sum of Boolean functions

\[ f(x_1, x_2, x_3, x_4) = x_1 + x_1 x_2 + x_3 x_4 + x_1 x_2 x_3 + x_1 x_3 x_4 = x_1 + x_1 x_2 + x_3 x_4(1 + x_1) + x_1 x_2 x_3 \]

Fix \( x_1, x_2 \), sum over all values of \((x_3, x_4)\)

\[
\sum_{(x_3, x_4) \in \mathbb{F}_2^2} f(x_1, x_2, x_3, x_4) = 4x_1 + 4x_1 x_2 + 1 + x_1 + 2x_1 x_2 = 1 + x_1
\]

- The set \( \{(c_1, c_2, x_3, x_4) \in \mathbb{F}_2^4\} \) is a cube with dim 2
- The resulting sum is the superpoly of the cube
The attack

Write a cipher by

\[ f(x, v) \mapsto \text{Output} \]

- Public variables \( v \) controlled by the attacker, e.g., a message or nonce
- Secret variables \( x \)
- Output: Ciphertext, keystream, or a hash bit

Preprocessing

- Find cubes with simple (e.g., linear) superpoly \( p(x) \)
- Reconstruct \( p(x) \)

Online

- Collect a system of linear equations \( p(x) = b \)
- Recover key bits by solving the equations and exhaustive search for remaining key bits if necessary
Preprocessing phase

Given cube $I$ of size $C$

Find cubes with simple (eg. linear) superpoly $p(x)$
- Property test of superpoly
- Complexity $O(N_12^C)$, $N_1$ is number of queries

Reconstruct superpoly $p(x)$

$$\sum_{v \in I} f(v, x) = p(x)$$

- Superpoly $p(x)$ can be recovered by Moebius Transformation
- Complexity $O(N_22^C)$, $N_2$ is number of queries
- More information on $p$, smaller $N_2$
Problems and Progress

How to find the most efficient cube?

Random walk heuristic algorithm [Dinur-Shamir’09]

Cube variables with conditions [Dinur et al. ’15]

Conditional cube attack [Huang et al. ’17]
Problems and Progress

- Attack in blackbox model
  - Cannot leverage the specific structural properties

- Size cube exploitable is limited ($\leq 40$)
  - Due to large complexity of testing superpoly
  - Cannot predict what will happen if bigger cube chosen
Problems and Progress

- Attack in blackbox model
  - Cannot leverage the specific structural properties

- Size cube exploitable is limited ($\leq 40$)
  - Due to large complexity of testing superpoly
  - Cannot predict what will happen if bigger cube chosen

- Kill two birds with one stone: Division property
Division property [Todo ’15]

- A method to construct higher order differential/integral distinguisher
- Successfully used to analyze block ciphers and hash functions
- Efficient evaluation by MILP [Xiang et al. ’16]
Cube attacks with division property

Ideas of the new attack [Todo et al. ’17]

- Analyze **involved variables** in the ANF of superpoly by division property
  - Non-Blackbox attack
  - Applied to nonlinear superpoly

- Model and solve the division propagation by MILP
  - Much more efficient than cube sum
  - Allow to search large cubes since no need to do cube sum to test the property of superpoly
What’s new

- Apply division property to analyze stream ciphers
- Exploit large cubes
- Improve key recovery attacks on stream ciphers, e.g. Trivium

<table>
<thead>
<tr>
<th>Round</th>
<th>Complexity</th>
<th>Cube size</th>
<th>Ref</th>
</tr>
</thead>
<tbody>
<tr>
<td>767</td>
<td>$2^{36}$</td>
<td>30</td>
<td>[Dinur-Shamir ’09]</td>
</tr>
<tr>
<td>799</td>
<td>$2^{62}$</td>
<td>40</td>
<td>[Fouque-Vannet ’13]</td>
</tr>
<tr>
<td>832</td>
<td>$2^{79}$</td>
<td>72</td>
<td>[Todo et al. ’17]</td>
</tr>
</tbody>
</table>
Our idea

- Investigate higher-degree monomials in the ANF of superpoly by division property
- Improve the MILP model by removing redundant division trails

Highlights of improved method

- Detect more information on superpoly
- Reduce complexity of superpoly recovery
- Attack more rounds
Trivium [Cannière-Preneel ’07]

- 80 bit key and 80 bit IV, 288 bit state
- 1152 rounds in initialization phase

\[(s_1, s_2, \ldots, s_{93}) \leftarrow (K_1, K_2, \ldots, K_{80}, 0, \ldots, 0)\]
\[(s_{94}, s_{95}, \ldots, s_{177}) \leftarrow (IV_1, IV_2, \ldots, IV_{80}, 0, \ldots, 0)\]
\[(s_{178}, s_{279}, \ldots, s_{288}) \leftarrow (0, \ldots, 0, 1, 1, 1)\]
\[t_1 \leftarrow s_{66} \oplus s_{93}\]
\[t_2 \leftarrow s_{162} \oplus s_{177}\]
\[t_3 \leftarrow s_{243} \oplus s_{288}\]
\[z \leftarrow t_1 \oplus t_2 \oplus t_3\]
\[t_1 \leftarrow t_1 \oplus s_{91} \cdot s_{92} \oplus s_{171}\]
\[t_2 \leftarrow t_2 \oplus s_{175} \cdot s_{176} \oplus s_{264}\]
\[t_3 \leftarrow t_3 \oplus s_{286} \cdot s_{287} \oplus s_{69}\]
\[(s_1, s_2, \ldots, s_{93}) \leftarrow (t_3, s_1, \ldots, s_{92})\]
\[(s_{94}, s_{95}, \ldots, s_{177}) \leftarrow (t_1, s_{94}, \ldots, s_{176})\]
\[(s_{178}, s_{279}, \ldots, s_{288}) \leftarrow (t_2, s_{178}, \ldots, s_{287})\]
Results on reduced-round Trivium

Improved key recovery attack on Trivium

<table>
<thead>
<tr>
<th>Round</th>
<th>Complexity</th>
<th>Cube size</th>
<th>Ref</th>
</tr>
</thead>
<tbody>
<tr>
<td>799</td>
<td>$2^{62}$</td>
<td>40</td>
<td>[Fouque-Vannet ’13]</td>
</tr>
<tr>
<td>832</td>
<td>$2^{79}$</td>
<td>72</td>
<td>[Todo et al. ’17]</td>
</tr>
<tr>
<td><strong>833</strong></td>
<td>$2^{75}$</td>
<td>74</td>
<td>new</td>
</tr>
</tbody>
</table>

Possible to further improve attack rounds!
Kreyvium [Canteaut et al. ’16]

- 128-bit variant of Trivium, $|K| = |IV| = 128$
- 1152 rounds initialization

$$(K_{127}^*, K_{126}^*, \ldots, K_0^*) \leftarrow (K_1, K_2, \ldots, K_{128})$$
$$(IV_{127}^*, IV_{126}^*, \ldots, IV_0^*) \leftarrow (IV_1, IV_2, \ldots, IV_{128})$$
$$(s_1, s_2, \ldots, s_{93}) \leftarrow (K_1, K_2, \ldots, K_{93})$$
$$(s_{94}, s_{95}, \ldots, s_{177}) \leftarrow (IV_1, IV_2, \ldots, IV_{84})$$
$$(s_{178}, s_{279}, \ldots, s_{288}) \leftarrow (IV_{85}, IV_{86}, \ldots, IV_{128}, 1, \ldots, 1, 0)$$
$$t_1 \leftarrow s_{66} \oplus s_{93}$$
$$t_2 \leftarrow s_{162} \oplus s_{177}$$
$$t_3 \leftarrow s_{243} \oplus s_{288} \oplus K_0^*$$
$$z \leftarrow t_1 \oplus t_2 \oplus t_3$$
$$t_1 \leftarrow t_1 \oplus s_{91} \cdot s_{92} \oplus s_{171} \oplus IV_0^*$$
$$t_2 \leftarrow t_2 \oplus s_{175} \cdot s_{176} \oplus s_{264}$$
$$t_3 \leftarrow t_3 \oplus s_{286} \cdot s_{287} \oplus s_{69}$$
$$(s_1, s_2, \ldots, s_{93}) \leftarrow (t_3, s_1, \ldots, s_{92})$$
$$(s_{94}, s_{95}, \ldots, s_{177}) \leftarrow (t_1, s_{94}, \ldots, s_{176})$$
$$(s_{178}, s_{279}, \ldots, s_{288}) \leftarrow (t_2, s_{178}, \ldots, s_{287})$$
$$(K_{127}^*, K_{126}^*, \ldots, K_0^*) \leftarrow (K_0^*, K_{127}^*, K_{126}^*, \ldots, K_1^*)$$
$$(IV_{127}^*, IV_{126}^*, \ldots, IV_0^*) \leftarrow (IV_0^*, IV_{127}^*, IV_{126}^*, \ldots, IV_1^*)$$
Results on reduced-round Kreyvium

Improved key recovery attack on Kreyvium

<table>
<thead>
<tr>
<th>Round</th>
<th>Complexity</th>
<th>Cube size</th>
<th>Ref</th>
</tr>
</thead>
<tbody>
<tr>
<td>872</td>
<td>$2^{124}$</td>
<td>85</td>
<td>[Todo et al. ’17]</td>
</tr>
<tr>
<td>884</td>
<td>$2^{124}$</td>
<td>95</td>
<td>new</td>
</tr>
</tbody>
</table>

- Still no clue on the security margin
- Lower security margin than Trivium
  - see also Conditional Differential Cryptanalysis [Watanabe et al. ’17]
Conclusion

- Apply division property to analyze stream cipher
- Capable to search large cubes
- Reduce complexity of superpoly recovery
- Improve key recovery attack on stream ciphers Trivium and Kreyvium
Future work

- Find the most efficient cube for stream ciphers
- Optimize the complexity of key recovery phase
- Apply to other designs
  - Cube attack + structural properties
Thank you!

Questions?