Improved Division Property Based Cube Attacks Exploiting Algebraic Properties of Superpoly

Qingju Wang[1] Yonglin Hao[2] Yosuke Todo[3] Chaoyun Li[4] Takanori Isobe[5] Willi Meier[6]

[1]SnT, University of Luxembourg, LU
[2]State Key Laboratory of Cryptology, Beijing, CN
[3]NTT Secure Platform Laboratories, JP
[4]imec-COSIC, KU Leuven, BE
[5]University of Hyogo, JP
[6]FHNW, CH

August 20, 2018

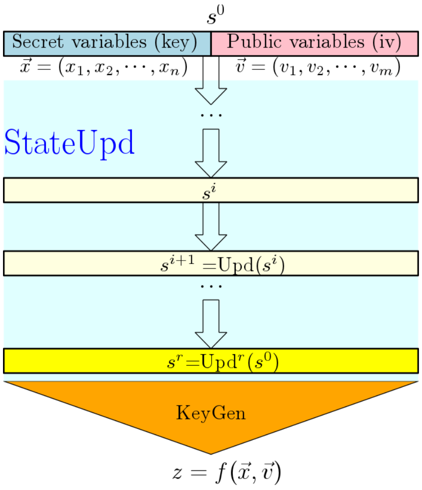
Outline

Outline

Why Stream Ciphers?

- Fast in software
  - RC4, Chacha
- Efficient in hardware
  - Grain, Trivium
- Low multiplications
  - Trivium, Kreyvium, FLIP, Rasta
- Used as authenticated encryptions
  - Acorn

Stream Ciphers

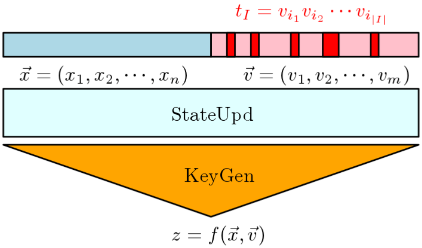


- n-bit secret variables (key) $\vec{x} = (x_1, x_2, \cdots, x_n)$
- m-bit public variables (iv) $\vec{v} = (v_1, v_2, \cdots, v_m)$
- $s^{i+1} = Upd(s^i)$, $0 \leq i \leq r-1$, where $s^0 = (\vec{x}, \vec{v})$.
- z is the first bit of the key stream.

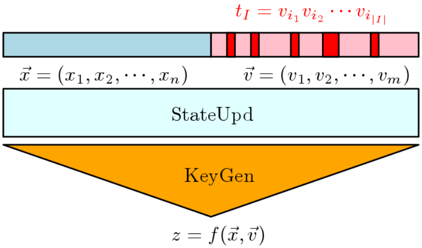
$$z = f(\vec{x}, \vec{v}) = \sum_{\vec{u} \in F_2^m} \alpha_{\vec{u}}^f \vec{v}^{\vec{u}},$$

where $\vec{v}^{\vec{u}} = \prod_{i=1}^m v_i^{u_i}$

The Idea of the Classical Cube Attacks

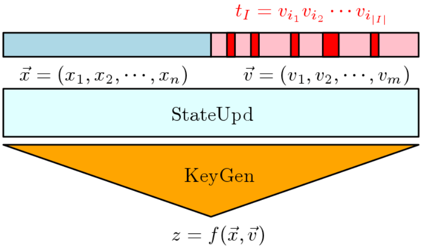


The Idea of the Classical Cube Attacks



- $I = \{i_1, i_2, \cdots i_{|I|}\}$ is the indices set of active bits of iv.
- C_I is the set of all $2^{|I|}$ values of v_i where $i \in I$.
- $z = f(\vec{x}, \vec{v}) = t_I \cdot p_I(\vec{x}, \vec{v}) + q_I(\vec{x}, \vec{v})$,
  q_I has at least one term in t_I missing.
- $\bigoplus_{v \in C_I} z = p_I(\vec{x}, \vec{v})$ is called **superpoly of C_I.**

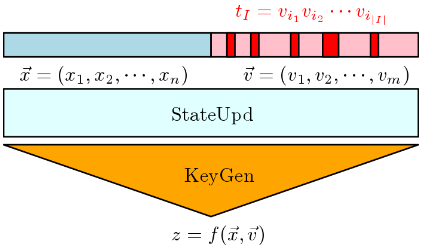
The Idea of the Classical Cube Attacks



- $I = \{i_1, i_2, \cdots i_{|I|}\}$ is the indices set of active bits of iv.
- C_I is the set of all $2^{|I|}$ values of v_i where $i \in I$.
- $z = f(\vec{x}, \vec{v}) = t_I \cdot p_I(\vec{x}, \vec{v}) + q_I(\vec{x}, \vec{v})$,
  q_I has at least one term in t_I missing.
- $\bigoplus_{v \in C_I} z = p_I(\vec{x}, \vec{v})$ is called **superpoly of C_I.**

- Attackers can recover secret information of $\vec{x}$ by analyzing p_I.

The Idea of the Classical Cube Attacks



- $I = \{i_1, i_2, \cdots i_{|I|}\}$ is the indices set of active bits of iv.
- C_I is the set of all $2^{|I|}$ values of v_i where $i \in I$.
- $z = f(\vec{x}, \vec{v}) = t_I \cdot p_I(\vec{x}, \vec{v}) + q_I(\vec{x}, \vec{v})$,
  q_I has at least one term in t_I missing.
- $\bigoplus_{v \in C_I} z = p_I(\vec{x}, \vec{v})$ is called **superpoly of C_I**.

- Attackers can recover secret information of $\vec{x}$ by analyzing p_I.
- We cannot decompose f in real since stream ciphers are complicated.

Experimental Approach for Classical Cube Attacks

- Stream cipher is regarded as a black box.
- How to recover the ANF of $p_I(\vec{x}, \vec{v})$:
  1. Compute $\bigoplus_{\vec{v} \in C_I} f(\vec{x}, \vec{v}) = p_I(\vec{x}, \vec{v})$ for a randomly chosen $\vec{x}$.
  2. Linearity tests are executed many times to see whether

$$p_I(\vec{x}, \vec{v}) \oplus p_I(\vec{x'}, \vec{v}) = p_I(\vec{x} \oplus \vec{x'}, \vec{v}).$$

  3. If the test is passed, the ANF of the superpoly can be recovered.
- Drawbacks of this approach:
  The size of cube is limited to experimental range: ≤ 40.

Contributions of TodoIHM17

- Introduce division property to cube attacks for the first time: analyze the ANF of the superpoly.
- The first theoretical attack: exploit very large cubes: e.g. 72 for 832-round Trivium.
- Provide upper bounds to recover the ANF of the superpoly.

Outline

(Bit-Based) Division Property, Todo Eurocrypt'15

Let $\mathbb{X} \in \mathbb{F}_2^n$ be a multiset, and $\mathbb{K} = \{\vec{k} | \vec{k} \in \mathbb{F}_2^n\}$. When $\mathbb{X}$ has the division property $\mathcal{D}_{\mathbb{K}}^n$, it fulfills

$$\bigoplus_{\vec{x} \in \mathbb{X}} \vec{x}^{\vec{u}} = \begin{cases} \text{unknown} & \text{if there exist } \vec{k} \in \mathbb{K} \text{ s.t. } \vec{u} \succeq \vec{k}, \\ 0 & \text{otherwise,} \end{cases}$$

where $\vec{u} \succeq \vec{k}$ if $u_i \geq k_i$ for all i.

Division Trail, Xiang et al. Asiacrypt'16

Assume the initial division property of a cipher be $\mathbb{K}_0 \triangleq \mathcal{D}_{\mathbb{K}_0}$, and the division property after the i-th round function R is $\mathbb{K}_i \triangleq \mathcal{D}_{\mathbb{K}_i}$. We have a trail of r rounds division property propagations

$$\mathbb{K}_0 \xrightarrow{R} \mathbb{K}_1 \xrightarrow{R} \cdots \xrightarrow{R} \mathbb{K}_r.$$

For $(\vec{k}_0, \vec{k}_1, \cdots, \vec{k}_r) \in (\mathbb{K}_0, \mathbb{K}_1, \cdots, \mathbb{K}_r)$, if $\vec{k}_i \to \vec{k}_{i+1}$, for all $0 \leq i \leq r-1$, then $(\vec{k}_0, \vec{k}_1, \cdots \vec{k}_r)$ is called an r-round division trail.

Evaluation of Division Trials

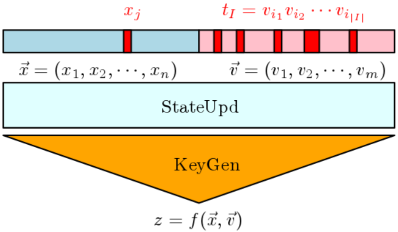
Ask for CP-based solver's help (Xiang et al., Asiacrypt'16)

- Create a MILP model $\mathcal{M}$ for the propagation of division property.
  - MILP, SAT/SMT, constraint programming etc.

  $\vec{k_0} \xrightarrow{Upd} \cdots \vec{k_i} \xrightarrow{Upd} \vec{k_{i+1}} \xrightarrow{Upd} \cdots \xrightarrow{Upd} \vec{k_r}.$
  - Entries of $\vec{k_0}, \cdots, \vec{k_r}$ are binary variables of $\mathcal{M}.var$.
  - $Upd(\cdot)$ is described by some constraints $\mathcal{M}.con$.
- Solvers can efficiently evaluate the feasibility of division trails.

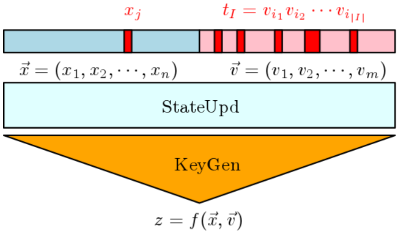
If $\vec{k_0} \to \vec{e_j}$ is infeasible, the jth bit is balanced (the sum is always 0).

Evaluate ANF Coefficients of Superpoly by Division Property



- Check division trail $(\vec{e_j}, \vec{k}) \xrightarrow{?} 1$, where $(\vec{e_j}, \vec{k}) \in F_2^n \times F_2^m$ and $\vec{v}^{\vec{k}} = t_I$.
- If no division trail $(\vec{e_j}, \vec{k}) \rightarrow 1 \Rightarrow x_j$ is not involved in superpoly.

Evaluate ANF Coefficients of Superpoly by Division Property



- Check division trail $(\vec{e}_j, \vec{k}) \xrightarrow{?} 1$, where $(\vec{e}_j, \vec{k}) \in F_2^n \times F_2^m$ and $\vec{v}^{\vec{k}} = t_I$.
- If no division trail $(\vec{e}_j, \vec{k}) \to 1 \Rightarrow x_j$ is not involved in superpoly.
- By repeating this procedure, all the secret variables of $\vec{x}$ involved in the superpoly can be determined and denoted as $J = \{x_{j_1}, x_{j_2}, \cdots, x_{j_{|J|}}\}$.

Overview of Attack Strategy in TodoIHM17

1. Evaluation phase.
   - Construct a random set I.
   - Determine the key bits J involved in the corresponding superpoly p_I.

   This phase is feasible: several hours by using Gurobi.
2. Off-line phase.
   - Sum the output over the given cube (C_I) and construct the whole truth table of the superpoly p_I.

   This phase is not practical, but time & memory complexity is bounded by $2^{|I|+|J|}$ and $2^{|J|}$.
3. On-line phase.
   - Query encryption oracle to attain the exact value of the superpoly.
   - Check the precomputed truth table and recover secret variables.

   Time & data complexity is $2^{|I|}$.

Limitation 1: Finding Proper $\vec{IV}$s May Require Multiple Trials In The 2nd Phase.

- Assumptions on the existence of IVs that can guarantee $p_I(\vec{x}, \vec{IV}) \not\equiv 0$ are proposed.
- When $|I| + |J|$ is small, practical experiments can be executed to find a specific IV.
- The rationality of assumptions is hard to be proved, especially when $|I| + |J|$ is close to n.

Limitation 1: Finding Proper $\vec{IV}$s May Require Multiple Trials In The 2nd Phase.

- Assumptions on the existence of IVs that can guarantee $p_I(\vec{x}, \vec{IV}) \not\equiv 0$ are proposed.
- When $|I| + |J|$ is small, practical experiments can be executed to find a specific IV.
- The rationality of assumptions is hard to be proved, especially when $|I| + |J|$ is close to n.

- We will provide a solution "flag technique" to determine a proper IV in the MILP model before implementing the attack.

Limitation 2: $|I| + |J| < n$

- After obtaining J, the attackers construct the whole truth table for the superpoly in the off-line phase, then the complexity of the off-line phase is about $2^{|I|+|J|}$.
- The restriction of $|I| + |J| < n$ barricades the adversary from exploiting larger cubes or mounting more rounds (where $|J|$ may expand).

Limitation 2: $|I| + |J| < n$

- After obtaining J, the attackers construct the whole truth table for the superpoly in the off-line phase, then the complexity of the off-line phase is about $2^{|I|+|J|}$.
- The restriction of $|I| + |J| < n$ barricades the adversary from exploiting larger cubes or mounting more rounds (where $|J|$ may expand).

- The restriction can be removed if the whole truth table construction can be avoided in the off-line phase.
- We will provide solutions to lower the bound of complexity:
  - Degree evaluation for the superpoly.
  - Terms enumeration for the superpoly.

Outline

Features cannot be Captured by the Previous MILP Models

- $COPY + AND$ operation:

$$(s_1, s_2) \rightarrow (s_1, s_2, s_1 \wedge s_2).$$

- Division property propagation (previous):

$$(x_1, x_2) \xrightarrow{COPY+AND} (y_1, y_2, a)$$

$$(1, 0) \xrightarrow{COPY+AND} \{(0, 0, 1), (1, 0, 0)\}$$

Features cannot be Captured by the Previous MILP Models

- *COPY* + *AND* operation:

$$(s_1, s_2) \rightarrow (s_1, s_2, s_1 \wedge s_2).$$

- Division property propagation (previous):

$$(x_1, x_2) \xrightarrow{COPY+AND} (y_1, y_2, a)$$

$$(1, 0) \xrightarrow{COPY+AND} \{(0, 0, 1), (1, 0, 0)\}$$

If $s_2 = 0$, then $s_1 \wedge s_2 = 0$ should have division property value $a = 0$.
The following division trail should be disabled

$$(1, 0) \xrightarrow{COPY+AND} (0, 0, 1).$$

Flag Technique

- Each division property value x is not only a binary variable of the MILP model

$$\mathcal{M}.var \leftarrow x$$

- It has an additional flag value

$$x.F \in \{0_c, 1_c, \delta\},$$

where

0_c: constant 0 bit
1_c: constant 1 bit
δ: variable bit

Rules for Flag Value operation: $=$, $\oplus$, $\times$.

- Naturally, $1_c = 1_c, 0_c = 0_c, \delta = \delta$.
- The $\oplus$ operation follows the rules:

$$\begin{cases} 1_c \oplus 1_c = 0_c \\ 0_c \oplus x = x \oplus 0_c = x \quad \text{for arbitrary } x \in \{1_c, 0_c, \delta\} \\ \delta \oplus x = x \oplus \delta = \delta \end{cases}$$

- The $\times$ operation follows the rules:

$$\begin{cases} 1_c \times x = x \times 1_c = x \\ 0_c \times x = x \times 0_c = 0_c \quad \text{for arbitrary } x \in \{1_c, 0_c, \delta\} \\ \delta \times \delta = \delta \end{cases}$$

MILP Model for Operations with Flag: The Example of AND

Let $(a_1, a_2, \dots, a_m) \xrightarrow{AND} b$ be a division trail of AND. The following inequalities are sufficient to describe the propagation of the division property for `andf`.

$$\begin{cases} \mathcal{M}.var \leftarrow a_1, a_2, \dots, a_m, b \text{ as binary.} \\ \mathcal{M}.con \leftarrow b \geq a_i \text{ for all } i \in \{1, 2, \dots, m\} \\ b.F = a_1.F \times a_2.F \times \cdots a_m.F \\ \mathcal{M}.con \leftarrow b = 0 \quad \text{if } b.F = 0_c \end{cases}$$

We denote this process as $(\mathcal{M}, b) \leftarrow \mathtt{andf}(\mathcal{M}, a_1, \dots, a_m)$.

Find Proper IVs to Guarantee Non-constant Superpoly and Determine J

Evaluate J by MILP with Flags for I and $\vec{IV} = \texttt{NULL}$

1. $\mathcal{M}.con \leftarrow \sum_{i=1}^{n} x_i = 1$
   and assign $x_i.F = \delta$ for all $i \in \{1, \ldots, n\}$
2. $\mathcal{M}.con \leftarrow v_i = 1$
   and assign $v_i.F = \delta$ for all $i \in I$
3. $\mathcal{M}.con \leftarrow v_i = 0$ for all $i \in \{1, 2, \ldots, n\} \setminus I$
4. $\boxed{v_i.F = \delta, \text{ for all } i \in \{1, 2, \ldots, m\} \backslash I}$
5. Update $\mathcal{M}$ with $Upd()$ and f
6. Solve $\mathcal{M}$ and return J.

Find Proper IVs to Guarantee Non-constant Superpoly and Determine J

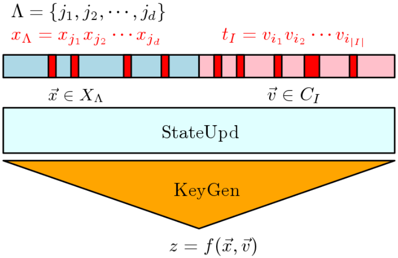
Evaluate J by MILP with Flags for I and $\vec{IV} = \texttt{NULL}$

1. $\mathcal{M}.con \leftarrow \sum_{i=1}^{n} x_i = 1$
   and assign $x_i.F = \delta$ for all $i \in \{1, \ldots, n\}$
2. $\mathcal{M}.con \leftarrow v_i = 1$
   and assign $v_i.F = \delta$ for all $i \in I$
3. $\mathcal{M}.con \leftarrow v_i = 0$ for all $i \in \{1, 2, \ldots, n\} \setminus I$
4. $\boxed{v_i.F = \delta, \text{ for all } i \in \{1, 2, \ldots, m\} \setminus I}$
5. Update $\mathcal{M}$ with $Upd()$ and f
6. Solve $\mathcal{M}$ and return J.

Evaluate J with I and some random specific assignments to the non-cube IVs until the same J is found.

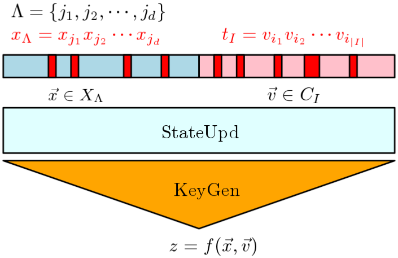
$$\boxed{v_i.F = \begin{cases} 1_c & \text{if } \vec{IV}[i] = 1 \\ 0_c & \text{if } \vec{IV}[i] = 0 \end{cases}}$$

Degree Evaluation for Superpoly



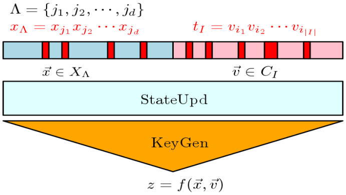
- Check division trail $(\vec{k}_\Lambda, \vec{k}) \xrightarrow{?} 1$, where $\vec{x}^{\vec{k}_\Lambda} = x_\Lambda$ and $\vec{v}^{\vec{k}} = t_I$.
- No division trail $\Rightarrow \vec{x}_\Lambda = x_{j_1} x_{j_2} \cdots x_{j_d}$ is not involved in superpoly.

Degree Evaluation for Superpoly



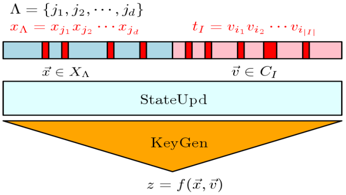
- Check division trail $(\vec{k}_\Lambda, \vec{k}) \xrightarrow{?} 1$, where $\vec{x}^{\vec{k}_\Lambda} = x_\Lambda$ and $\vec{v}^{\vec{k}} = t_I$.
- No division trail $\Rightarrow \vec{x}_\Lambda = x_{j_1} x_{j_2} \cdots x_{j_d}$ is not involved in superpoly.

For all $\Lambda \subseteq \{1, 2, \cdots, n\}$ of size $d + 1$, evaluate division trail $(\vec{k}_\Lambda, \vec{k}) \xrightarrow{?} 1$.
If not, the degree of the superpoly is bounded by d.



- Check division trail $(\vec{k}_\Lambda, \vec{k}) \xrightarrow{?} 1$, where $\vec{x}^{\vec{k}_\Lambda} = x_\Lambda$ and $\vec{v}^{\vec{k}} = t_I$.
- No division trail $\Rightarrow \vec{x}_\Lambda = x_{j_1} x_{j_2} \cdots x_{j_d}$ is not involved in superpoly.

For all $\Lambda \subseteq \{1, 2, \cdots, n\}$ of size $d+1$, evaluate division trail $(\vec{k}_\Lambda, \vec{k}) \xrightarrow{?} 1$.
If not, degree of the superpoly is bounded by d.



- Check division trail $(\vec{k}_\Lambda, \vec{k}) \xrightarrow{?} 1$, where $\vec{x}^{\vec{k}_\Lambda} = x_\Lambda$ and $\vec{v}^{\vec{k}} = t_I$.
- No division trail $\Rightarrow \vec{x}_\Lambda = x_{j_1} x_{j_2} \cdots x_{j_d}$ is not involved in superpoly.

For all $\Lambda \subseteq \{1, 2, \cdots, n\}$ of size $d+1$, evaluate division trail $(\vec{k}_\Lambda, \vec{k}) \xrightarrow{?} 1$.
If not, degree of the superpoly is bounded by d.

MILP description

$$\mathcal{M}.var \leftarrow x_1, \cdots, x_n$$
$$\mathcal{M}.var \leftarrow v_1, \cdots, v_m$$
$$\mathcal{M}.var \leftarrow z$$
$$\mathcal{M}.con \leftarrow v_i = \begin{cases} 1, & \text{if } i \in I \\ 0, & \text{otherwise} \end{cases}$$
$$\mathcal{M}.con \leftarrow x_i = \begin{cases} 1, & \text{if } i \in \Lambda \\ 0, & \text{otherwise} \end{cases}$$
$$\mathcal{M}.con \leftarrow Upd()$$
$$\mathcal{M}.con \leftarrow z = 1$$
$$\mathcal{M}.obj \leftarrow max \sum_{i=1}^{n} x_i$$

$\mathcal{M}$ is feasible and $\mathcal{M}.obj = d$

Our Attack Strategy: 1st Phase – Evaluation phase.

- Construct a random set I.
- Determine the key bits J involved in the corresponding superpoly.
- Use Flag Technique to find a proper IV.
- Use Degree Evaluation to determine d.

Our Attack Strategy: 2nd Phase – Off-line Phase.

- There are at most $\binom{|J|}{\leq d} = \sum_{i=0}^{d} \binom{|J|}{i}$ monomials have non-zero coefficients s.t.

$$p_I(x, IV) = \bigoplus_{\vec{u} \in F_2^{|J|}, hw(\vec{u}) \leq d} \alpha_{\vec{u}} \vec{x}^{\vec{u}}$$

- Pick $\binom{|J|}{\leq d}$ different $\vec{x}$'s and sum over the cube C_I to generate a linear system of the coefficients $\alpha_{\vec{u}}$ and store the solution.

The time complexity of this phase is $2^{|I|} \times \binom{|J|}{\leq d}$ ($\leftarrow 2^{|I|} \times 2^{|J|}$ TodoIHM17).
The memory complexity is $\binom{|J|}{\leq d}$ ($\leftarrow 2^{|J|}$ TodoIHM17).

Our Attack Strategy: 3rd Phase – Online Phase

1. Access encryption oracle under chosen iv setting and compute the exact value of the superpoly with a cube summation:

$$\lambda = p_I(\vec{x}, \vec{v}) = \bigoplus_{\vec{v} \in C_I} f(\vec{x}, \vec{v}).$$

2. With the knowledge of coefficient $\alpha_{\vec{u}}$'s, reconstruct the truth table T:
If $T[i] = \lambda$, then i is a candidate value of $(x_{j_1}, x_{j_2}, \cdots, x_{j_{|J|}})$. Otherwise, i is a wrong guess.

The data complexity is $2^{|I|}$ (same as TodoIHM).
The time complexity is $2^{|I|} + 2^{|J|} \times \binom{|J|}{\le d}$ ($2^{|I|}$ in TodoIHM17).

The total time complexity of the attack

$$\max\left\{2^{|I|} \times \binom{|J|}{\le d}, 2^{|I|} + 2^{|J|} \times \binom{|J|}{\le d}\right\}.$$

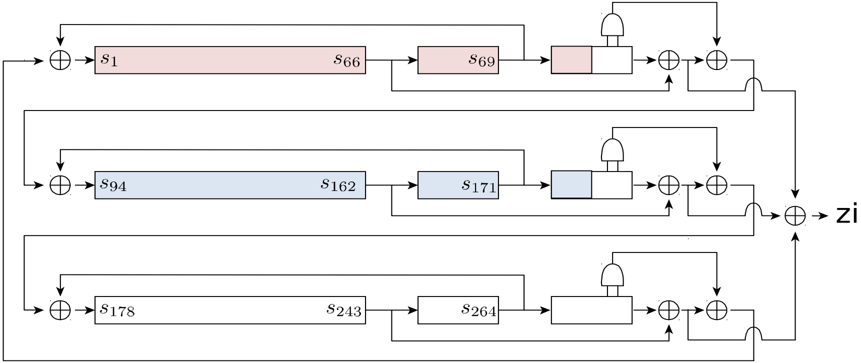
Terms Enumeration for Superpoly

- Based on the MILP model for Degree Evaluation.
- Update the model by adding the constraint:
  $\mathcal{M}.con \leftarrow \sum_{i\in\Lambda} x_i = t$ for $1 \le t \le d-1$.
- Obtain set J_t $(1 \le t \le d)$: all possible terms of degree t involved in the superpoly

$$2^{|I|} \times (\sum_{t=0}^{d} |J_t|) \le 2^{|I|} \times \binom{|J|}{\le d}.$$

Outline

Trivium



80-bit secret key

80-bit initialization vector

state size = 288 bits
initialization = 1152 rounds

Application to Trivium: Experimental Verification

Active IVs	Involved keys	Round	Complexity
$I = \{1, 11, 21, 31, 41, 51, 61, 71\}$ $\lvert I \rvert = 8$	$J = \{23, 24, 25, 66, 67\}$ $\lvert J \rvert = 5$	591	2^{13}

- $d = 3$, IV=0xcc2e487b,0x78f99a93,0x$beae$
  $p_I(\vec{x}, \vec{v}) = x_{66}(x_{23}x_{24} \oplus x_{25} \oplus x_{67} \oplus 1)$
- $d = 2$, $IV =$ 0x61fbe5da, 0x19f5972c, 0x65c1
  $p_I(\vec{x}, \vec{v}) = x_{23}x_{24} \oplus x_{25} \oplus x_{67} \oplus 1$
- $d = 0$, $IV =$ 0x5b942db1,0x83ce1016,0x6ce
  $p_I(\vec{x}, \vec{v}) = 0$

Application to Trivium: Theoretical Key Recoveries

Active IVs	d	Involved keys	Round	Complexity
$I = \{1, 2, ..., 65, 67, 69, ..., 79\}$ $\|I\| = 72$	3	$J = \{34, 58, 59, 60, 61\}$ $\|J\| = 5, \|J_2\| = 5, \|J_3\| = 1$	832	$2^{76.7}$ (Degree) $2^{75.58}$(Term)
$I = \{1, 2, ..., 67, 69, 71, ..., 79\}$ $\|I\| = 73$	3	$J = \{49, 58, 60, 74, 75, 76\}$ $\|J\| = 7, \|J_2\| = 5, \|J_3\| = 1$	833	2^{79}(Degree) $2^{76.9}$(Term)
$I = \{1, ...33, 35, ..., 46, 48, ..., 80\}$ $\|I\| = 78$ $IV[47] = 1$	1	$J = \|\{61\}$ $\|J\| = 1$	839	2^{79}

Summary of Our Improved Results

Ciphers	Round	Complexity	Source
Trivium	832	2^{77}	TodoIHM17
	839	2^{79}	Ours
Kreyvium	872	2^{124}	TodoIHM17
	891	$2^{120.73}$	Ours
Grain-128a	183	2^{108}	TodoIHM17
	184	$2^{109.61}$	Ours
Acorn	704	2^{122}	TodoIHM17
	750	$2^{120.92}$	Ours

Outline

- Conclusions
  - Division property based cube attack is an effective tool for conducting partial key recoveries on stream ciphers.
  - We exploit algebraic structures of the superpoly: upper bound degree, non-zero coefficients of ANF.

- Conclusions
  - Division property based cube attack is an effective tool for conducting partial key recoveries on stream ciphers.
  - We exploit algebraic structures of the superpoly: upper bound degree, non-zero coefficients of ANF.
- Future works
  - Other targets for launching division property based cube attacks (block ciphers?).
  - Further modifying the MILP modeling is also meaningful.
  - Links among division property based cube attack with other cube attack variants (dynamic, correlation etc.)

Thanks