Superpoly Recovery of Grain-128AEAD Using Division Property

Debasmita Chakraborty(✉) and Santu Pal

Indian Statistical Institute, Kolkata, Kolkata, India
debasmitachakraborty1@gmail.com, santu.pal@niser.ac.in

Abstract. The cube attack is a powerful cryptanalytic technique against stream ciphers. Cube attacks exploit the algebraic properties of symmetric ciphers by recovering a particular polynomial, the superpoly, and subsequently, the secret key. Nowadays, the division property-based approach has become very popular, allowing us to recover the exact superpoly cleverly. However, the computational cost to recover the superpoly becomes prohibitive as the number of rounds of the cipher increases. In this paper, we study NIST lightweight 3rd round candidate Grain-128AEAD in the light of division property-based cube attacks. We first introduce some good cubes of dimensions 91, 92, 93, 94, and then we construct an algorithm to find conditional key bits for the cubes of Grain-128AEAD mentioned above. Next, we apply three-subset division property without unknown subset-based cube attacks to recover exact superpolies for 192, 193, 194, 195-round Grain-128AEAD in the weak-key setting, which are the longest till now. Moreover, we are able to find good cubes that are used to build distinguishers of Grain-128AEAD in the weak-key setting. In particular, we show that Grain-128AEAD can be distinguished from a random source up to 193-rounds in the weak-key setting, which is the best zero-sum distinguisher of Grain-128AEAD till now using division property-based cube attacks.

Keywords: Cube attack · Division property · Three-subset division property · MILP · Grain-128AEAD

1 Introduction

Cube attack, proposed by Dinur and Shamir [6] at EUROCRYPT 2009, is one of the most powerful cryptanalytic techniques against symmetric cryptosystems. The target of cube attack is to recover secret variables from the simplified polynomial called superpoly. To mount a cube attack, one first recovers the superpoly in an offline phase. Then, the value of the superpoly is obtained by querying the encryption oracle and computing the summation. From the equation between the superpoly and its value, information about the secret key can be revealed. Therefore, the superpoly recovery is a central step in the cube attack.

G. Bella et al. (Eds.): SecITC 2022, LNCS 13809, pp. 65–80, 2023.
https://doi.org/10.1007/978-3-031-32636-3_4

Traditional cube attacks [6,8,25,31] regard ciphers as black boxes so the superpolies are recovered experimentally. Only linear or quadratic superpolies are applicable. At CRYPTO 2017, [27] Todo *et al* treated the polynomial as non-blackbox and applied Conventional Bit-based Division Property (CBDP) to cube attacks on stream ciphers for the first time. Then, at CRYPTO 2018, Wang *et al* [29] improved it by introducing flag and term enumeration techniques. For CBDP based cube attacks, the superpolies of large cubes can be recovered by the theoretical method. But the theory of CBDP cannot ensure that the superpoly of a cube is non-constant. Hence the key recovery attack may be just a distinguishing attack. To solve this problem, at ASIACRYPT 2019, Wang *et al* [30] proposed the cube attack based on Bit-based Division Property using Three Subsets (BDPT) and proved that BDPT without an unknown subset can recover the accurate superpoly of cube attack. Then, at EUROCRYPT 2020, Hao *et al* [11] proposed a new modeling method for the BDPT without an unknown subset. Their algorithm is more efficient, and it can improve existing key-recovery attacks on many ciphers. Moreover, in [13,15] the authors embedded the monomial prediction technique into a nested framework, which allows them to recover superpolies and in [31], the authors also developed a pure algebraic method to recover the exact superpoly. However, as the number of rounds of the cipher increases, such useful cubes are hard to find.

One of the most significant security criteria for a keyed cryptographic primitive is its unpredictable behaviour concerning any randomly chosen key from the whole key space. When a key is used with a given cipher, it is considered to be *weak* if it causes the cipher to behave in an undesirable way (like it reduces the algebraic degree significantly). Many attacks in the weak-key setting for block cipher [12,16], as well as stream ciphers [23,26] have been presented. However, finding a weak-key set is a computationally hard problem. For example, the invariant subspace attack [18,19], is a general weak-key attack, that is known in the literature. Recently, cube attacks that investigate key conditions which may lead to weak-key attacks, have been proposed in [21,22].

Table 1. Previous Works of Superpoly Recovery for Grain-128AEAD using Division Property

Round	Number of Cubes	Cube Size	Time	References
190	–	95	–	[11]
191	2	95–96	–	[15]
192	1	94	45 days	[13]

Related Works. NIST has launched a process for soliciting, evaluating, and standardising lightweight cryptographic algorithms suited for use in limited contexts. In August 2018, NIST issued a call for algorithms to be considered for lightweight cryptography standards. There were initially 57 submissions and

NIST released ten candidates following the third round of pruning. Grain-128AEAD is one of these candidates. Grain-128AEAD is designed by modifying the authentication module of Grain-128a. Grain-128a has been adopted as an ISO standard for radio frequency identification (RFID) devices. Further, the encryption module of Grain-128AEAD and Grain-128a are the same. The cryptanalysis of the encryption module of Grain-128a can be applied to the cryptanalysis of the encryption module of Grain-128AEAD.

As Grain-128AEAD is one of the candidates in the competition by NIST, the cryptanalysis of Grain-128AEAD is an important research area. In 2012, Lehmann *et al* [20] proposed an attack using the conditional cube tester on Grain-128a of 177 KSA (Key Scheduling Algorithm) round in the single key setup and 189 KSA round in the weak-key setup. Recently, Ma *et al* [24] and Karlsson *et al* [17] proposed a differential attack and nonrandomness detectors on Grain-128a up to 195 and 203 KSA rounds, respectively in the weak-key setup. Readers may refer to [2–5,7,28] for detailed cryptanalytic results on the Grain family. Moreover, using the concept of division property-based cube attacks, exact superpolies for 190, 191, 192-round Grain-128AEAD have been recovered efficiently using which key-recovery attacks are also mounted [11,13,15] (The results we have listed in Table 1). But, for these cube attacks, the cube dimensions are on the higher side. Now, the following question arises in our mind:

Can we reduce the cube dimension and recover exact superpoly of the cube for higher round Grain-128AEAD?

1.1 Our Contributions

To address this question, we begin by studying the most popular cipher Grain-128AEAD in the light of division property-based cube attacks. Our primary focus is to reduce the cube dimension of Grain-128AEAD and recover exact superpoly using those cubes for higher round Grain-128AEAD. The details of our technical contributions are listed as follows:

Finding Cubes and Searching Conditional Key Bits. First, we search for good cubes with less dimensions than the previous division property-based cube attacks for which we can recover superpoly efficiently (which is illustrated in Sect. 3.1). Here, we use the cube dimensions of Grain-128AEAD as 91, 92, 93, and 94. Therefore, our other important contribution is to search for conditional key bits for which we can efficiently recover superpolies of above-mentioned cubes of Grain-128AEAD (which is described in Sect. 3.2). To do this, we provide an algorithm (Algorithm 1) using which we can set conditions on key bits which depend on cube variables.

Application on Grain-128AEAD. As for the application of our concept, we apply three-subset division property without unknown subset in order to recover exact superpoly of Grain-128AEAD of our cubes in the weak-key setting. As a

result of this, we find exact superpolies of 192-195 round Grain-128AEAD in the weak-key setting which are the best results on Grain-128AEAD till now. Moreover, we also present a zero-sum distinguisher of 193-round Grain-128AEAD which is the longest distinguisher of Grain-128AEAD using division property-based cube attacks. The detailed results are shown in Table 2.

Table 2. Summarization of our Superpoly Recovery Results for Grain-128AEAD in the Weak-Key Setup using Division Property

Round	Number of Cubes	Cube Size	Time	References
192	2	91, 92	2 min	Sect. 4
193	2	92, 94	7 min	Sect. 4
194	1	93	1 h	Sect. 4
195	1	94	7 days	Sect. 4

1.2 Organization of the Paper

This paper is organized as follows: In Sect. 2, we briefly recall some background knowledge and the relationship between the division property and cube attack. In Sect. 3, we construct good cubes and propose an algorithm to construct appropriate weak-key conditions to perform cube attack on Grain-128AEAD. Therefore, we show some results (superpoly recovery, zero-sum distinguisher) on Grain-128AEAD in Sect. 4. At last we conclude the paper in Sect. 5.

2 Preliminaries

2.1 Notations

Let $\mathbb{F}_2$ denote the finite field $\{0,1\}$ and $\boldsymbol{a} = (a_0, a_1, \ldots, a_{n-1}) \in \mathbb{F}_2^n$ be an n-bit vector, where a_i denotes the i-th bit of a. For n-bit vectors $\boldsymbol{x}$ and $\boldsymbol{u}$, define $\boldsymbol{x}^{\boldsymbol{u}} = \prod_{i=0}^{n-1} x_i^{u_i}$. Then, for any $\boldsymbol{k} \in \mathbb{F}_2^n$ and $\boldsymbol{k'} \in \mathbb{F}_2^n$, define $\boldsymbol{k} \succeq \boldsymbol{k'}$ if $k_i \geq k'_i$ holds for all $i = 0, 1, \ldots, n-1$, and define $\boldsymbol{k} \succ \boldsymbol{k'}$ if $k_i > k'_i$ holds for all $i = 0, 1, \ldots, n-1$. For a subset $\mathcal{I} \subseteq \{0, 1, ..., n-1\}$, $\boldsymbol{u}_{\mathcal{I}}$ denotes an n-dimensional bit vector $(u_0, u_1, \ldots, u_{n-1})$ satisfying $u_i = 1$ if $i \in \mathcal{I}$ and $u_i = 0$ otherwise. We simply write $\mathbb{K} \leftarrow \boldsymbol{k}$ when $\mathbb{K} = \mathbb{K} \cup \{\boldsymbol{k}\}$ and $\mathbb{K} \rightarrow \boldsymbol{k}$ when $\mathbb{K} = \mathbb{K} \setminus \{\boldsymbol{k}\}$. And $|\mathbb{K}|$ denotes the number of elements in the set $\mathbb{K}$. We denote $[n] = \{1, 2, \ldots, n\}$, $\mathbf{1} = 1^n$, and $\mathbf{0} = 0^n$.

2.2 Specification of Grain128AEAD

Grain-128AEAD [14] is a member of the Grain family and also one of the winner of the NIST LWC standardization process. Grain-128AEAD inherits many specifications from Grain-128a, which was proposed in 2011 [1]. There are four

differences between Grain-128AEAD and Grain-128a: (i) larger Macs, (ii) no encryption-only mode, (iii) initialization hardening, and (iv) keystream limitation. These differences do not come only from the requirement for the NIST LWC standardization process but also from recent cryptanalysis results against Grain-128a [10].

The internal state is represented by two 128-bit states, $(b_0, b_1, \ldots, b_{127})$ and $(s_0, s_1, \ldots, s_{127})$. The 128-bit key $\boldsymbol{K}$ is loaded to the first register $\boldsymbol{b}$, and the 96-bit initialization vector is loaded to the second register $\boldsymbol{s}$. The other state bits are set to 1 except for the last one bit in the second register. Namely, the initial states are represented as

$$\begin{cases} (b_0, b_1, \ldots, b_{127}) = (K_1, K_2, \ldots, K_{128}) \\ (s_0, s_1, \ldots, s_{127}) = (IV_1, IV_2, \ldots, IV_{96}, 1, 1, \ldots, 1, 0) \end{cases}$$

We denote IV is a set consisting of IV_1, IV_2, $\ldots$, IV_{96}. The pseudo-code of the update function in the initialization is given as follows.

$$\begin{cases} g \leftarrow b_0 + b_{26} + b_{56} + b_{91} + b_{96} + b_3 b_{67} + b_{11} b_{13} + b_{17} b_{18} + b_{27} b_{59} \\ \qquad + b_{40} b_{48} + b_{61} b_{65} + b_{68} b_{84} + b_{88} b_{92} b_{93} b_{95} + b_{22} b_{24} b_{25} + b_{70} b_{78} b_{82}, \\ f \leftarrow s_0 + s_7 + s_{38} + s_{70} + s_{81} + s_{96}, \\ h \leftarrow b_{12} s_8 + s_{13} s_{20} + b_{95} s_{42} + s_{60} s_{79} + b_{12} b_{95} s_{94}, \\ z \leftarrow h + s_{93} + b_2 + b_{15} + b_{36} + b_{45} + b_{64} + b_{73} + b_{89}, \\ (b_0, b_1, \ldots, b_{127}) \leftarrow (b_1, \ldots, b_{127}, g + s_0 + z), \\ (s_0, s_1, \ldots, s_{127}) \leftarrow (s_1, \ldots, s_{127}, f + z). \end{cases}$$

In the initialization, the state is updated 256 times without producing an output. After the initialization, the update function is tweaked such that z is not fed to the state, and z is used as a pre-output key stream. Hereinafter, we assume that the first bit of the pre-output key stream can be observed. Note that there is no difference between Grain128a and Grain-128AEAD under this assumption.

2.3 Cube Attack and Division Property

Cube Attack. The cube attack was proposed by Dinur and Shamir in [6]. A cipher is regarded as a public Boolean function whose input is divided into two parts: secret variables $\boldsymbol{x}$ and public ones $\boldsymbol{v}$. Then, the ANF of the Boolean function is represented as

$$f(\boldsymbol{x}, \boldsymbol{v}) = \bigoplus_{\boldsymbol{u} \in \mathbb{F}_2^{n+m}} a_{\boldsymbol{u}}^f (\boldsymbol{x} \,||\, \boldsymbol{v})^{\boldsymbol{u}}.$$

For a set of indices $\mathcal{I} = \{i_1, i_2, \ldots, i_{|\mathcal{I}|}\} \subset \{1, 2, \ldots, m\}$, which is referred as cube indices, $t_{\mathcal{I}}$ denotes a monomial as $t_{\mathcal{I}} = v_{i_1} \cdot v_{i_2} \cdots v_{i_{|\mathcal{I}|}}$. The Boolean function $f(\boldsymbol{x}, \boldsymbol{v})$ can also be decomposed as

$$f(\boldsymbol{x}, \boldsymbol{v}) = t_{\mathcal{I}} \cdot p(\boldsymbol{x}, \boldsymbol{v}) + q(\boldsymbol{x}, \boldsymbol{v}).$$

Let $C_{\mathcal{I}}$, which is referred as a cube (defined by $\mathcal{I}$), be a set of $2^{|\mathcal{I}|}$ values where variables in $\{v_{i_1}, v_{i_2}, \ldots, v_{i_{\mathcal{I}}}\}$ are taking all possible combinations of values, and all remaining variables are fixed to any value. The sum of f over all values of the cube $C_{\mathcal{I}}$ is

$$\bigoplus_{C_{\mathcal{I}}} f(\boldsymbol{x}, \boldsymbol{v}) = \bigoplus_{C_{\mathcal{I}}} t_{\mathcal{I}} \cdot p(\boldsymbol{x}, \boldsymbol{v}) \; + \; \bigoplus_{C_{\mathcal{I}}} q(\boldsymbol{x}, \boldsymbol{v}) = p(\boldsymbol{x}, \boldsymbol{v})$$

because $t_{\mathcal{I}} = 1$ for only one case in $C_{\mathcal{I}}$ and each term in $q(\boldsymbol{x}, \boldsymbol{v})$ misses at least one variable from $\{v_{i_1}, v_{i_2}, \ldots, v_{i_{\mathcal{I}}}\}$. Then, $p(\boldsymbol{x}, \boldsymbol{v})$ is called the superpoly of the cube $C_{\mathcal{I}}$, and the goal of the cube attack is to recover the superpoly.

Division Property. The division property is formally developed as the generalization of the integral property, and it has been initially used to evaluate the integral distinguisher. Now, the relationship between the division property and the ANF of public functions is discussed below:

Definition 1. (Three-Subset Division Property without Unknown Subset *[11]*). $\mathbb{X}$ *be a multi set whose elements take a value of* $\mathbb{F}_2^n$. *Let* $\tilde{\mathbb{L}}$ *be also a multi set whose elements also take a value of* $\mathbb{F}_2^n$. *When the multi-set* $\mathbb{X}$ *has three-subset division property without unknown subset* $(\mathcal{T}_{\tilde{\mathbb{L}}}^{1^n})$, *it fulfills the following conditions:*

$$\bigoplus_{\boldsymbol{x} \in \mathbb{X}} \boldsymbol{x}^{\boldsymbol{u}} = \begin{cases} 1, & \textit{if there are odd number of } \boldsymbol{u}'\textit{s in } \tilde{\mathbb{L}} \\ 0, & \textit{otherwise} \end{cases}$$

Using this definition, the authors also defined three-subset division trail and explained the propagation rules of COPY, XOR and AND in [11].

Mixed Integer Linear Programming (MILP). MILP is a kind of optimization or feasibility program whose objective function and constraints are linear, and the variables can be continuous or integers. Generally, an MILP model M consists of variables $M.var$, constraints $M.con$, and the objective function $M.obj$. MILP models can be solved by solver like Gurobi [9]. If there is no feasible solution, the solver will returns infeasible. And if there are feasible solutions, the solver will returns the optimal value of the objective function. When there is no objective function in M, the MILP solver will only return whether M is feasible or not.

Algorithm to Recover ANF Coefficients of Public Function [11]. Let f be a Boolean function whose input denotes an n-bit string $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, and let it consist of the iteration of simple public functions. Then, the algebraic normal form of f is represented as

$$f(\boldsymbol{x}) = \bigoplus_{\boldsymbol{u} \in \mathbb{F}_2^n} a_{\boldsymbol{u}}^f \boldsymbol{x}^{\boldsymbol{u}}$$

Our goal is to recover the value of $a_{\boldsymbol{u}}^{f}$ for some $\boldsymbol{u}$. To do this, we have to first construct MILP model that represents the three-subset division property without unknown subset of the function f. The authors in [11] proposed an algorithm (Algorithm 1 in [11]) which recovers an ANF coefficient $a_{\boldsymbol{u}}^{f}$. The initial three-subset division property without unknown subset is defined by $\boldsymbol{u}$, and the number of feasible solutions is enumerated by using the MILP solver. Note that the efficiency of Algorithm 1 in [11] depends on the number of feasible solutions.

3 Superpoly Recovery for Grain-128AEAD Using Weak Keys

The most important and challenging part of the cube attack is to recover the ANF of the superpoly of the cube. As Grain-128AEAD is a finalist in a recent NIST competition, it will be challenging to recover the superpoly of such cipher. Before recovering the superpoly, one needs to search for a good cube of the cipher. If one works on a weak-key setting, another important task is finding conditional key variables, which leads to the recovery of the superpoly of the cube.

3.1 Cube Searching Algorithm for Grain-128AEAD

Constructing a cube-searching algorithm nowadays is a crucial task for a cube attack. Many such algorithms exist for such purposes as maximum last zero, and maximum last α $(0 \leq \alpha \leq 1)$. The last method gives a better cube searching for Grain-128a. So we have used this algorithm to find a better result. In the paper [4], the authors have found a cube of $\{63, 64, 66, 68, 69\}$ of size five to mount a distinguishing attack for 191 round in a single key scenario. They construct the cube of size five from a cube of size one.

Following a similar method, we also start to find the cube of size one. The best cube of size one is $\{s_{69}\}$ because it attains the maximum last alpha round at 123. By similar process, we get the cube variables s_{68}, s_{67}, s_{66}, s_{65}, s_{64} simultaneously. As those cube variables expose some weakness of the Boolean functions at particular rounds, so we again work with those cube variables. The variables mentioned above are crucial in getting a distinguisher for Grain-128a. But our challenge is tougher and more exciting. We want to recover the superpoly of the Boolean function at some particular round using the division property. As we know, a small dimensional cube will not be useful for superpoly recovery in the division property-based cube attacks. Due to the success of our cube variables in the previous attack on Grain-128a, we decide to work with the complement of the set of cube variables. As superpoly searching is lengthy and time-consuming process, so we start to find the superpoly using the cube of size $96 - 2 = 94$. As previously cube of sizes $96, 95$ was used, so we used a cube of less size to reduce the complexity of superpoly recovery. Then we decrease the cube size one by one, following less complexity for superpoly recovery. Also, we vary the initialization

round to reduce the complexity. Finally, we find the best trade off between the initialization round and cube size to get better complexity.

Algorithm 1: Searching for Conditional Bits Corresponding to Chosen Cubes

Input: Set of strong variables[a] $\mathcal{S}$
Output: Set of Conditional key variables $\mathcal{W}$
begin
  Start with a single element from $\mathcal{S}$ and store it in $\mathcal{C}$
  while $|\mathcal{C}| \leq |\mathcal{S}|$ **do**
    Choose the cube variables as $IV \setminus \mathcal{C}$
    Store the conditional key variables from SAGE corresponding to variables in $IV \setminus \mathcal{C}$ in $\mathcal{W}$.
    Also, store conditional key variables from structure observation of the cipher in $\mathcal{W}$
    Run division property-based cube attacks using the cube $IV \setminus \mathcal{C}$
    if *Superpoly corresponding to* $IV \setminus \mathcal{C}$ *is recovered* **then**
      Take $\mathcal{W}$ as a set of conditional key bits
    end
    else
      Add some additional conditional key variables in $\mathcal{W}$
      Run division property-based cube attacks
      Repeat Else part until superpoly is recovered
    end
    Take another subset $\mathcal{C}$ of $\mathcal{S}$ and repeat the while part.
  end
  return $\mathcal{W}$
end

[a] The set of those variables in IV using which we can construct good cubes for Grain-128AEAD.

3.2 Searching Weak-Key Domain for Grain-128AEAD

Putting conditions on key and IV variables plays an important role in upgrading the attacks on any cipher. Conditions on the variables help us to find weaknesses in the corresponding Boolean function at a particular round. In the previous paper [4], the authors found the conditions on key bits corresponding to cube variables using SAGE software. Also, some conditions are found using the structure observation with theoretical analysis. We have also followed their approaches. But the conditions retrieved for corresponding cubes do not help us to recover superpoly using division property-based approaches. So again, we try to find the additional conditions to recover the superpoly. We try to find the subset of key bits which contributes to superpoly recovery. As the division property-based attack takes all IV bits as zero, we do not worry about the conditions on IV bits. The selection of key bits is made in the following way:

In our case, we have implemented the algorithm on the different rounds of Grain-128AEAD in the following way.

- We collect the strong variables of Grain-128AEAD as $\mathcal{S} = \{s_{42}, \cdots, s_{69}\}$.
- The conditions on key variables for each strong variable are given in Table 3.
- Select an element say r from $\mathcal{S}$ and take $IV \setminus \{r\}$ as cube. Therefore, collect all corresponding conditional key bits from Table 3 in the set $\mathcal{W}$ corresponding to the chosen cube.
- Also, we collect the conditional key variables getting through the structure observation of Grain-128AEAD in the set $\mathcal{W}$ (Given in the last of this section).
- For example, we take $IV \setminus \{s_{69}, s_{68}\}$ as a cube for 195-round Grain-128AEAD and run the division property-based cube attacks to recover the superpoly.
- As we can not recover the superpoly, we add some additional conditional key variables $b_{42}, b_{43}, b_{44}, b_{45}, b_{72}, b_{73}, b_{76}, b_{77}, b_{121}, b_{122}, b_{123}, b_{124}, b_{126}, b_{127}$ in the set $\mathcal{W}$.
- Again, we run the program. This time, we recover the superpoly for 195-round Grain-128AEAD. Similar way, we find $\mathcal{W}$ for different cubes and recover superpolies.

Note 1. As the running of division property-based cube attack is a time-consuming process, we optimize the $\mathcal{W}$ set as much as possible.

Table 3. Conditions on key variables for 1-dimensional cubes

Cube	Conditions on key variables	Cube	Conditions on key variables
$\{s_{42}\}$	$b_{46} = b_{50} = b_{95} = 0$	$\{s_{56}\}$	$b_{60} = b_{64} = b_{109} = 0$
$\{s_{43}\}$	$b_{47} = b_{51} = b_{96} = 0$	$\{s_{57}\}$	$b_{61} = b_{65} = b_{110} = 0$
$\{s_{44}\}$	$b_{48} = b_{52} = b_{97} = 0$	$\{s_{58}\}$	$b_{62} = b_{66} = b_{111} = 0$
$\{s_{45}\}$	$b_{49} = b_{53} = b_{98} = 0$	$\{s_{59}\}$	$b_{63} = b_{67} = b_{112} = 0$
$\{s_{46}\}$	$b_{50} = b_{54} = b_{99} = 0$	$\{s_{60}\}$	$b_{64} = b_{68} = b_{113} = 0$
$\{s_{47}\}$	$b_{51} = b_{55} = b_{100} = 0$	$\{s_{61}\}$	$b_{65} = b_{69} = b_{114} = 0$
$\{s_{48}\}$	$b_{52} = b_{56} = b_{101} = 0$	$\{s_{62}\}$	$b_{66} = b_{70} = b_{115} = 0$
$\{s_{49}\}$	$b_{53} = b_{57} = b_{102} = 0$	$\{s_{63}\}$	$b_{67} = b_{71} = b_{80} = b_{116} = 0$
$\{s_{50}\}$	$b_{54} = b_{58} = b_{103} = 0$	$\{s_{64}\}$	$b_{68} = b_{72} = b_{117} = 0$
$\{s_{51}\}$	$b_{55} = b_{59} = b_{104} = 0$	$\{s_{65}\}$	$b_{69} = b_{73} = b_{118} = 0$
$\{s_{52}\}$	$b_{56} = b_{60} = b_{105} = 0$	$\{s_{66}\}$	$b_{70} = b_{74} = b_{119} = 0$
$\{s_{53}\}$	$b_{57} = b_{61} = b_{106} = 0$	$\{s_{67}\}$	$b_{71} = b_{75} = b_{120} = 0$
$\{s_{54}\}$	$b_{58} = b_{62} = b_{107} = 0$	$\{s_{68}\}$	$b_{72} = b_{76} = b_{121} = 0$
$\{s_{55}\}$	$b_{59} = b_{63} = b_{108} = 0$	$\{s_{69}\}$	$b_{73} = b_{77} = b_{122} = 0$

From the structure observation, the additional conditional key bits for the above
cubes are $b_{64}, b_{67}, b_{70} - b_{74}, b_{76} - b_{87}, b_{91}, b_{94}, b_{95}, b_{102}, b_{104}, b_{105}, b_{108}, b_{110}, b_{112} - b_{114}, b_{116}, b_{118}, b_{119}, b_{121}, b_{122}, b_{125}$.

3.3 Division Property-Based Cube Attack for Grain-128AEAD

The most important part of a cube attack is to recover the superpoly, and we simply call it the *superpoly recovery* in this paper. In [11], the authors explained how three-subset division property without unknown subset can be used as a tool to analyze ANF coefficients of the superpoly for a public Boolean function.

Superpoly Recovery. The encryption module of Grain-128AEAD is regarded as a public boolean function $f(x, v)$ whose input is divided into two parts: secret variable x and public variable v. Now, we construct MILP model $\mathcal{M}$ where the encryption module of Grain-128AEAD is represented by the context of division property as described in Algorithm 5 in [11]. Here, we denote $\boldsymbol{x}$ and $\boldsymbol{v}$ as the MILP variables corresponding to secret and public variables and in our case, $\boldsymbol{x} = (b_0^0, \ldots, b_{127}^0)$, and $\boldsymbol{v} = (s_0^0, \ldots, s_{127}^0)$. Therefore, to represent the initial division property, elements of $\boldsymbol{v}$ indexed by $\mathcal{I}$ (cube indices) are constrained by 1 and the elements of $\boldsymbol{v}$ indexed by the other IV indices are constrained by 0. Moreover, we add the constraints corresponding to weak-key conditions in MILP model $\mathcal{M}$.

After constructing MILP model $\mathcal{M}$ with initial division property corresponding to cube and non-cube indices and weak-key conditions, we solve MILP model $\mathcal{M}$ as all monomials that could be involved in the superpoly can be found as feasible solutions (Algorithm 2 in [11]). Finally, we enumerate feasible solutions and finally get the superpoly of Grain-128AEAD corresponding to the cube $C_{\mathcal{I}}$ where $\mathcal{I}$ be the cube indices. Although using this method, we can accurately find superpoly of the cube $C_{\mathcal{I}}$, it is practically impossible to enumerate all feasible solutions when there are too many solutions.

After recovering the superpoly, an attacker can retrieve the information regarding the Boolean function of the cipher. Also, the attacker can use the drawbacks in the superpoly to find loopholes in the output function of the cipher, which leads to a distinguishing attack. Further, one can extend it to a key recovery attack using a sufficient number of superpolies.

4 Experimental Results

We apply the three-subset division property without unknown subset based cube attacks on the encryption module of Grain-128AEAD in the weak-key setting. First, we search appropriate cubes and weak-key using Algorithm 1, and therefore using division property-based cube attack technique we accurately recover the superpolies for 192-195 rounds using cube sizes 91, 92, 93, 94 respectively in the weak-key setting where the size of the corresponding weak-key class is 2^{43}. The details of our results are given in Table 4. These are the best-known attacks on Grain-128AEAD in the weak-key setting till now. Moreover, we construct zero-sum distinguishers on 192-193 round Grain-128AEAD in the weak-key setting which are the longest distinguisher in this direction. The detailed parameters of *superpoly recovery* of 192-round and 193-round Grain-128AEAD and zero-sum

distinguishers are in the following subsections. The recovered superpoly for 194-round Grain-128AEAD is in the Appendix A.

Superpoly Recovery for 192-Round Grain-128AEAD. The cube indices of size 91 to recover superpoly of 192-Round Grain-128AEAD are

$$\mathcal{I} = \{1, 2, \ldots, 65, 71, \ldots, 96\}$$

and $IV_{66} = IV_{67} = IV_{68} = IV_{69} = IV_{70} = 0$. Therefore, we get the superpoly corresponding to $C_{\mathcal{I}}$ which is represented as the sum of 2 monomials, and the following

$$p(\boldsymbol{x}) = x_{40}x_{42} + x_{29}$$

is the recovered superpoly, where $\boldsymbol{x} = (x_1, x_2, \ldots, x_{128})$ denotes the secret key, i.e., $x_i = K_i$. This superpoly is a balanced Boolean function because there is a monomial x_{29} that is independent of other monomials.

Superpoly Recovery for 193-Round Grain-128AEAD. The cube indices of size 92 to recover superpoly of 193-Round Grain-128AEAD are

$$\mathcal{I} = \{1, 2, \ldots, 66, 71, \ldots, 96\}$$

and $IV_{67} = IV_{68} = IV_{69} = IV_{70} = 0$. Therefore, we get the superpoly corresponding to $C_{\mathcal{I}}$ which is represented as the sum of 38 monomials, and the following

$$\begin{aligned} p(\boldsymbol{x}) \;&= 1 + x_{43} + x_{42}x_{43} + x_{41} + x_{40}x_{42}x_{43} + x_{39}x_{41} + x_{39}x_{40} \\ &\quad + x_{38} + x_{36}x_{38} + x_{35}x_{36} + x_{33} + x_{33}x_{35} + x_{32} + x_{32}x_{36} \\ &\quad + x_{31}x_{41}x_{42} + x_{31}x_{40}x_{41} + x_{31}x_{35}x_{37} + x_{30} + x_{29}x_{38} \\ &\quad + x_{29}x_{36}x_{37} + x_{29}x_{34}x_{37} + x_{29}x_{31} + x_{28} + x_{28}x_{42}x_{43} \\ &\quad + x_{28}x_{36}x_{38} + x_{28}x_{29} + x_{28}x_{29}x_{37} + x_{26} + x_{26}x_{29} + x_{25} \\ &\quad + x_{25}x_{28} + x_{24} + x_{24}x_{43} + x_{24}x_{32} + x_{24}x_{31} + x_{22} + x_{21} + x_{18} \end{aligned}$$

is the recovered superpoly, where $\boldsymbol{x} = (x_1, x_2, \ldots, x_{128})$ denotes the secret key, i.e., $x_i = K_i$. This superpoly is a balanced Boolean function because there are monomials x_{22}, x_{21}, and x_{18} that are independent of other monomials.

Zero-Sum Distinguishers for 192-193 Round Grain-128AEAD. To construct the cube attack against 192-round Grain-128AEAD, we choose the cube indices of size 92 as follows:

$$\mathcal{I} = \{1, 2, \ldots, 66, 71, \ldots, 96\}$$

where $IV_{67} = IV_{68} = IV_{69} = IV_{70} = 0$. Therefore, in the weak-key setting, we find that the superpoly does not involve secret key (where $\boldsymbol{x} = (x_1, x_2, \ldots, x_{128})$

denotes the secret key). Hence, the cube attack against 192-round Grain-128AEAD is a zero-sum distinguisher.

Moreover, the cube attack against 193-round Grain-128AEAD is also a zero-sum distinguisher where we choose the cube indices of size 94 as follows:

$$\mathcal{I} = \{1, 2, \ldots, 68, 71, \ldots, 96\}$$

where $IV_{69} = IV_{70} = 0$. This is the longest zero-sum distinguisher on Grain-128AEAD using division property-based cube attack best known to us.

Table 4. Results of Superpoly Recovery for Different Cubes on Grain-128AEAD

Cube size	Cube variables	Round	Additional Conditional Key Variables
91	$IV \setminus \{s_{65}, s_{66}, s_{67}, s_{68}, s_{69}\}$	192	$b_{42}, b_{43}, b_{44}, b_{45}, b_{69}, b_{70}, b_{71}, b_{72}, b_{73}$ $b_{74}, b_{75}, b_{76}, b_{77}, b_{118}, b_{119}, b_{120}$ $b_{121}, b_{122}, b_{123}, b_{124}, b_{126}, b_{127}$
92	$IV \setminus \{s_{66}, s_{67}, s_{68}, s_{69}\}$	193	$b_{42}, b_{43}, b_{44}, b_{45}, b_{70}, b_{71}, b_{72}, b_{73}$ $b_{74}, b_{75}, b_{76}, b_{77}, b_{119}, b_{120}$ $b_{121}, b_{122}, b_{123}, b_{124}, b_{126}, b_{127}$
93	$IV \setminus \{s_{67}, s_{68}, s_{69}\}$	194	$b_{42}, b_{43}, b_{44}, b_{45}, b_{71}, b_{72}, b_{73}, b_{75}, b_{76}, b_{77}$ $b_{120}, b_{121}, b_{122}, b_{123}, b_{124}, b_{126}, b_{127}$
94	$IV \setminus \{s_{68}, s_{69}\}$	195	$b_{42}, b_{43}, b_{44}, b_{45}, b_{72}, b_{73}, b_{76}, b_{77}$ $b_{121}, b_{122}, b_{123}, b_{124}, b_{126}, b_{127}$

5 Conclusion and Future Work

In this paper, we revisit division property-based cube attacks and study NIST lightweight 3rd round candidate Grain-128AEAD in the light of cube attacks based on division property. First, we find some good cubes and propose an algorithm to find conditional key bits for our cubes of Grain-128AEAD. Therefore, we efficiently apply three-subset division property without unknown subset based cube attacks on Grain-128AEAD and recover superpolies up to 195 rounds in the weak-key setting which are best-known results on Grain-128AEAD till now. Moreover, we find zero-sum distinguishers on 193-round Grain-128AEAD which is the longest distinguisher in this direction.

As, it is hard to find good cubes with less dimension in order to construct division property-based cube attacks, how to construct an efficient cube searching algorithm so that we can recover exact superpolies of higher rounds Grain-128AEAD is an open problem. Moreover, in the single-key setup, how to mount distinguishing as well as key recovery attacks on stream ciphers efficiently using division property will be nice future work.

Acknowledgement. The authors would like to thank the anonymous reviewers for their valuable comments and suggestions.

A Detailed Result for Cube Attacks Against Grain-128AEAD

The cube indices of size 93 to recover superpoly of 194-Round Grain-128AEAD are

$$\mathcal{I} = \{1, 2, \ldots, 67, 71, \ldots, 96\}$$

and $IV_{68} = IV_{69} = IV_{70} = 0$. Therefore, we get the superpoly corresponding to $C_{\mathcal{I}}$ which is represented as the sum of 38 monomials, and the following

$$\begin{aligned}
p(\boldsymbol{x}) \;=\; & 1 + x_{43} + x_{42}x_{43} + x_{41} + x_{41}x_{42} + x_{40}x_{43} + x_{40}x_{42} + x_{40}x_{41} \\
& + x_{40}x_{41}x_{43} + x_{40}x_{41}x_{42} + x_{39} + x_{39}x_{41} + x_{39}x_{40} + x_{39}x_{40}x_{41}x_{42} \\
& + x_{38}x_{43} + x_{38}x_{41}x_{43} + x_{38}x_{41}x_{42} + x_{38}x_{40} + x_{38}x_{39} + x_{38}x_{39}x_{41}x_{42} \\
& + x_{37}x_{39}x_{40} + x_{37}x_{38} + x_{36}x_{40} + x_{35} + x_{35}x_{37}x_{41} + x_{35}x_{37}x_{40} \\
& + x_{34}x_{36} + x_{34}x_{35}x_{40} + x_{34}x_{35}x_{38} + x_{33} + x_{33}x_{42} + x_{33}x_{41}x_{43} \\
& + x_{32}x_{33} + x_{31} + x_{31}x_{42} + x_{31}x_{40} + x_{31}x_{39}x_{40} + x_{31}x_{36} \\
& + x_{31}x_{33} + x_{30}x_{43} + x_{30}x_{41}x_{42} + x_{30}x_{40}x_{42} + x_{30}x_{40}x_{41} \\
& + x_{30}x_{38} + x_{30}x_{37} + x_{30}x_{35}x_{37} + x_{30}x_{33} + x_{30}x_{33}x_{37} + x_{30}x_{32} \\
& + x_{30}x_{32}x_{41} + x_{30}x_{32}x_{40} + x_{30}x_{31} + x_{30}x_{31}x_{41} + x_{29} + x_{29}x_{41}x_{42} \\
& + x_{29}x_{40} + x_{29}x_{38} + x_{29}x_{35}x_{37}x_{41} + x_{29}x_{33} + x_{29}x_{32}x_{41} + x_{29}x_{30}x_{39} \\
& + x_{29}x_{30}x_{33} + x_{28} + x_{28}x_{41} + x_{28}x_{41}x_{43} + x_{28}x_{40} + x_{28}x_{35}x_{40} \\
& + x_{28}x_{35}x_{40}x_{41} + x_{28}x_{33} + x_{28}x_{31} + x_{28}x_{31}x_{40} + x_{28}x_{30} \\
& + x_{28}x_{30}x_{40} + x_{28}x_{30}x_{40} + x_{28}x_{30}x_{38} + x_{28}x_{30}x_{35}x_{40} + x_{28}x_{30}x_{31} \\
& + x_{28}x_{29}x_{39} + x_{27} + x_{27}x_{40} + x_{26}x_{40} + x_{26}x_{38} + x_{25}x_{40} + x_{25}x_{30}x_{40} \\
& + x_{24}x_{41}x_{43} + x_{24}x_{40} + x_{24}x_{40}x_{41} + x_{24}x_{38} + x_{24}x_{30}x_{41} + x_{24}x_{30}x_{40} \\
& + x_{24}x_{30}x_{39} + x_{24}x_{29}x_{41} + x_{24}x_{26} + x_{23} + x_{23}x_{40} + x_{23}x_{30} \\
& + x_{23}x_{29} + x_{22} + x_{22}x_{40} + x_{21} + x_{21}x_{38}x_{40} + x_{21}x_{33} \\
& + x_{21}x_{31}x_{40}x_{41} + x_{21}x_{31}x_{34}x_{36} + x_{21}x_{27} + x_{21}x_{26}x_{31}x_{40}x_{42} + x_{21}x_{26}x_{30} \\
& + x_{21}x_{26}x_{29}x_{31} + x_{21}x_{26}x_{28}x_{31} + x_{21}x_{23}x_{31} + x_{20}x_{42}x_{43} + x_{20}x_{40} \\
& + x_{20}x_{40}x_{42}x_{43} + x_{20}x_{38} + x_{20}x_{38}x_{40} + x_{20}x_{37}x_{39} + x_{20}x_{36} + x_{20}x_{36}x_{38} \\
& + x_{20}x_{35} + x_{20}x_{35}x_{36} + x_{20}x_{33} + x_{20}x_{33}x_{35}x_{36} + x_{20}x_{32} + x_{20}x_{32}x_{41}x_{42} \\
& + x_{20}x_{32} + x_{20}x_{32}x_{41}x_{42} + x_{20}x_{32}x_{35}x_{36} + x_{20}x_{29}x_{37}x_{41}x_{42} \\
& + x_{20}x_{29}x_{36}x_{38} + x_{20}x_{29}x_{36}x_{37} + x_{20}x_{29}x_{35}x_{37} + x_{20}x_{29}x_{31} \\
& + x_{20}x_{29}x_{30}x_{37} + x_{20}x_{29}x_{30}x_{32} + x_{20}x_{28} + x_{20}x_{28}x_{39}x_{40} + x_{20}x_{28}x_{37} \\
& + x_{20}x_{28}x_{33}x_{35} + x_{20}x_{28}x_{32} + x_{20}x_{28}x_{32}x_{35} + x_{20}x_{28}x_{29} + x_{20}x_{27} \\
& + x_{20}x_{26} + x_{20}x_{25} + x_{20}x_{25}x_{29} + x_{20}x_{24}x_{32} + x_{20}x_{24}x_{29}x_{37} \\
& + x_{20}x_{24}x_{28} + x_{20}x_{23} + x_{20}x_{23}x_{29}x_{32} + x_{20}x_{22}x_{28} + x_{20}x_{22}x_{28} \\
& + x_{20}x_{22}x_{24} + x_{20}x_{21} + x_{20}x_{21}x_{40}x_{42} + x_{20}x_{21}x_{23}x_{29} + x_{20}x_{21}x_{22} \\
& + x_{19}x_{20}x_{29} + x_{18}x_{20} + x_{17}x_{40} + x_{17}x_{38} + x_{13} + x_{11}x_{20}
\end{aligned}$$

is the recovered superpoly, where $\boldsymbol{x} = (x_1, x_2, \ldots, x_{128})$ denotes the secret key, i.e., $x_i = K_i$.
As the superpoly for 195-round Grain-128AEAD contains a huge number of terms, therefore we can not present it here.

References

1. Ågren, M., Hell, M., Johansson, T., Meier, W.: Grain-128a: a new version of grain-128 with optional authentication. Int. J. Wirel. Mob. Comput. **5**(1), 48–59 (2011)
2. Aumasson, J.P., Dinur, I., Henzen, L., Meier, W., Shamir, A.: Efficient FPGA implementations of high-dimensional cube testers on the stream cipher grain-128. IACR Cryptol. ePrint Arch., p. 218 (2009)
3. Banik, S., Maitra, S., Sarkar, S., Meltem Sönmez, T.: A chosen IV related key attack on grain-128a. In: Boyd, C., Simpson, L. (eds.) ACISP 2013. LNCS, vol. 7959, pp. 13–26. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39059-3_2
4. Dalai, D.K., Pal, S., Sarkar, S.: Some conditional cube testers for grain-128a of reduced rounds. IEEE Trans. Comput. **71**(6), 1374–1385 (2022)
5. Dinur, I., Güneysu, T., Paar, C., Shamir, A., Zimmermann, R.: An experimentally verified attack on full grain-128 using dedicated reconfigurable hardware. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 327–343. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_18
6. Dinur, I., Shamir, A.: Cube attacks on tweakable black box polynomials. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 278–299. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01001-9_16
7. Dinur, I., Shamir, A.: Breaking grain-128 with dynamic cube attacks. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 167–187. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21702-9_10
8. Fouque, P.-A., Vannet, T.: Improving key recovery to 784 and 799 rounds of trivium using optimized cube attacks. In: Moriai, S. (ed.) FSE 2013. LNCS, vol. 8424, pp. 502–517. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43933-3_26
9. Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual (2021)
10. Hamann, M., Krause, M.: On stream ciphers with provable beyond-the-birthday-bound security against time-memory-data tradeoff attacks. Cryptogr. Commun. **10**(5), 959–1012 (2018)
11. Hao, Y., Leander, G., Meier, W., Todo, Y., Wang, Q.: Modeling for three-subset division property without unknown subset. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12105, pp. 466–495. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45721-1_17
12. Hawkes, P.: Differential-linear weak key classes of IDEA. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 112–126. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054121
13. He, J., Hu, K., Preneel, B., Wang, M.: Stretching cube attacks: improved methods to recover massive superpolies. IACR Cryptol. ePrint Arch., p. 1218 (2022)
14. Hell, M., Johansson, T., Meier, W., Sönnerup, J., Yoshida, H.: An AEAD variant of the grain stream cipher. In: Carlet, C., Guilley, S., Nitaj, A., Souidi, E.M. (eds.) C2SI 2019. LNCS, vol. 11445, pp. 55–71. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-16458-4_5

15. Hu, K., Sun, S., Todo, Y., Wang, M., Wang, Q.: Massive superpoly recovery with nested monomial predictions. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021. LNCS, vol. 13090, pp. 392–421. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92062-3_14
16. Kara, O., Manap, C.: A new class of weak keys for blowfish. In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 167–180. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74619-5_11
17. Karlsson, L., Hell, M., Stankovski, P.: Not so greedy: enhanced subset exploration for nonrandomness detectors. In: Mori, P., Furnell, S., Camp, O. (eds.) ICISSP 2017. CCIS, vol. 867, pp. 273–294. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93354-2_13
18. Leander, G., Abdelraheem, M.A., AlKhzaimi, H., Zenner, E.: A cryptanalysis of PRINTcipher: the invariant subspace attack. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 206–221. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_12
19. Leander, G., Minaud, B., Rønjom, S.: A generic approach to invariant subspace attacks: cryptanalysis of robin, iSCREAM and zorro. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 254–283. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46800-5_11
20. Lehmann, M., Meier, W.: Conditional differential cryptanalysis of grain-128a. In: Pieprzyk, J., Sadeghi, A.-R., Manulis, M. (eds.) CANS 2012. LNCS, vol. 7712, pp. 1–11. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35404-5_1
21. Li, Z., Bi, W., Dong, X., Wang, X.: Improved conditional cube attacks on keccak keyed modes with MILP method. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017. LNCS, vol. 10624, pp. 99–127. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70694-8_4
22. Li, Z., Dong, X., Wang, X.: Conditional cube attack on round-reduced ASCON. IACR Trans. Symm. Cryptol. **2017**(1), 175–202 (2017)
23. Liu, F., Isobe, T., Meier, W., Sakamoto, K.: Weak keys in reduced AEGIS and tiaoxin. IACR Trans. Symm. Cryptol. **2021**(2), 104–139 (2021)
24. Ma, Z., Tian, T., Qi, W.-F.: Improved conditional differential attacks on grain v1. IET Inf. Secur. **11**(1), 46–53 (2017)
25. Mroczkowski, P., Szmidt, J.: The cube attack on stream cipher trivium and quadraticity tests. Fundam. Informaticae **114**(3–4), 309–318 (2012)
26. Rohit, R., Sarkar, S.: Diving deep into the weak keys of round reduced ascon. IACR Trans. Symm. Cryptol. **2021**(4), 74–99 (2021)
27. Todo, Y., Isobe, T., Hao, Y., Meier, W.: Cube attacks on non-blackbox polynomials based on division property. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10403, pp. 250–279. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63697-9_9
28. Todo, Y., Isobe, T., Meier, W., Aoki, K., Zhang, B.: Fast correlation attack revisited. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10992, pp. 129–159. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96881-0_5
29. Wang, Q., Hao, Y., Todo, Y., Li, C., Isobe, T., Meier, W.: Improved division property based cube attacks exploiting algebraic properties of superpoly. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10991, pp. 275–305. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_10
30. Wang, S., Hu, B., Guan, J., Zhang, K., Shi, T.: MILP-aided method of searching division property using three subsets and applications. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019. LNCS, vol. 11923, pp. 398–427. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34618-8_14

31. Ye, C., Tian, T.: A new framework for finding nonlinear superpolies in cube attacks against trivium-like ciphers. In: Susilo, W., Yang, G. (eds.) ACISP 2018. LNCS, vol. 10946, pp. 172–187. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93638-3_11