IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2017, No. 1, pp. 175–202. DOI:10.13154/tosc.v2017.i1.175-202

Conditional Cube Attack on Round-Reduced ASCON

Zheng Li[1], Xiaoyang Dong[1,2*] and Xiaoyun Wang[1,2*]

[1] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Shandong, China
{lizhengcn,dongxiaoyang}@mail.sdu.edu.cn
[2] Institute for Advanced Study, Tsinghua University, Beijing, China
xiaoyunwang@tsinghua.edu.cn.

Abstract. This paper evaluates the secure level of authenticated encryption Ascon against cube-like method. Ascon submitted by Dobraunig *et al.* is one of 16 survivors of the 3rd round CAESAR competition. The cube-like method is first used by Dinur *et al.* to analyze Keccak keyed modes. At CT-RSA 2015, Dobraunig *et al.* applied this method to 5/6-round reduced Ascon, whose structure is similar to Keccak keyed modes. However, for Ascon the non-linear layer is more complex and state is much smaller, which make it hard for the attackers to select enough cube variables that do not multiply with each other after the first round. This seems to be the reason why the best previous key-recovery attack is on 6-round Ascon, while for Keccak keyed modes (Keccak-MAC and Keyak) the attacked round is no less than 7-round.
In this paper, we generalize the conditional cube attack proposed by Huang *et al.*, and find new cubes depending on some key bit conditions for 5/6-round reduced Ascon, and translate the previous theoretic 6-round attack with 2^{66} time complexity to a practical one with 2^{40} time complexity. Moreover, we propose the first 7-round key-recovery attack on Ascon. By introducing *the cube-like key-subset technique*, we divide the full key space into many subsets according to different key conditions. For each key subset, we launch the cube tester to determine if the key falls into it. Finally, we recover the full key space by testing all the key subsets. The total time complexity is about $2^{103.9}$. In addition, for a weak-key subset, whose size is 2^{117}, the attack is more efficient and costs only 2^{77} time complexity. Those attacks do not threaten the full round (12 rounds) Ascon.

Keywords: Ascon · CAESAR · Cube-like · Key Recovery · Authenticated Encryption

Introduction

Nowadays, when confidential messages are transmitted using an insecure channel, both their privacy and integrity are usually needed. Authenticated encryption (AE) schemes are proposed to meet both goals simultaneously. In 2014, the CAESAR competition [com14] was launched to identify good authenticated encryption (AE) candidates as better alternatives to current options such as AES-GCM [NIS]. Totally, 57 candidates have been submitted to the first round of the CAESAR competition. After two rounds of assessments from world-wide cryptographers and engineers, only 16 survivors were announced to be included in the third round of the CAESAR competition in 15 Aug 2016. In order to get a secure scheme, many more cryptographic analyses on these candidates are needed urgently.

Ascon [DEMS15a] is one of the 16 candidates, which is submitted by Dobraunig *et al.* Ascon uses a lightweight sponge construction, the internal state is only 320-bit. The

*Corresponding authors

Received: 2016-09-01, Revised: 2016-11-23, Accepted: 2017-01-24, Published: 2017-03-0

Table 1: Summary of Key-recovery Attacks on Ascon

Type	Attacked Rounds	Time	Source
Differential-Linear	4/12	2^{18}	[DEMS15b]
	5/12	2^{36}	[DEMS15b]
Cube-like Method	5/12	2^{35}	[DEMS15b]
	6/12	2^{66}	[DEMS15b]
	5/12	2^{24}	Section 4
	6/12	2^{40}	Section 5
	7/12	$2^{103.9}$	Section 6
	7/12	2^{77} for 2^{117} keys	Section 6

previous best attack [DEMS15b] is proposed by Ascon's authors themselves, where the key-recovery attack is on 6 out of 12-round using a cube-like method with time complexity of 2^{66}. In this paper, we will focus on the analysis of Ascon against cube-like method.

Cube attack [DS09] is a chosen plaintext key-recovery attack, which was introduced by Dinur and Shamir. Since then, cube attack was applied to many different cryptographic primitives [ADMS09, DS11, FV13]. At Eurocrypt 2015, Dinur *et al.* [DMP+15] published a key-recovery attack on Keccak keyed modes, where the cube variables are selected not to multiply with each other after the first round, then the output degree of the polynomials is reduced. Later it was applied to Ketje [BDP+16] by Dong *et al.* [DLWQ17]. Huang *et al.* [HWX+] proposed a new *conditional cube attack* on Keccak keyed modes and presented an 8-round attack on Keyak. By restraining some bit conditions of the key, they obtain a new set of cube variables which not only do not multiply with each other after the first round, but also contains one cube variable that does not multiply with other cube variables after the second round, and then the output degree over cube variables is further reduced.

Our Contributions

In this paper, we continue to explore the secure level of Ascon against cube-like method. Ascon has a more complex non-linear layer and smaller state than Keccak keyed modes, which make it hard for the attackers to select enough cube variables for the 7-round attack that do not multiply with each other after the first round. This seems to be the reason why the best key-recovery attack is on 6-round Ascon, while for Keccak keyed modes (Keccak-MAC and Keyak) the attacked round is no less than 7-round. In this paper, we firstly generalize the conditional cube attack, which is first proposed by Huang *et al.* [HWX+]. By exploring the details of non-linear layer and the conditional cube attack method, we improve the complexity of the 6-round attack from 2^{66} to a practical one 2^{40}. And then inspired by a so-called key-dependent strategy [DLJW15, LJWD15], we develop a new *cube-like key-subset technique*. Based on this technique, we construct a series of 65-dimension cubes for different key subsets, for each key subset we give a 7-round key-recovery attack, finally the key-recovery attacks cover the full key space. Our attacks work on the latest version Ascon v1.2. The results are summarized in Table 1. Our contribution is four fold:

1. We generalize the conditional cube attack proposed by Huang *et al.* [HWX+], where a cube variable of certain cube (all the cube variables do not multiply with each other after the first round) does not multiply with other cube variables under some key bit conditions after the second round. In our generalized model, these key bits actually produce some common divisors of all the cube sums on the output bits. That means, if these divisors are zero by restraining conditions on these key bits, all the cube sums will be zero. Hence, using cube testers, one can test whether these divisors are zero or not, and then deduce the key bit conditions.

2. When applying the generalized conditional cube attack to Ascon, we find some sets of cube variables for Ascon. In each set, the cube variables do not multiply with each other after the first round. Moreover, by assigning some bit conditions of the key, one cube variable does not multiply with the other cube variables after the second round. This reduces the cube dimension and leads to 5/6-round key-recovery attacks with time complexity of 2^{24} and 2^{40}, respectively. This is the first practical key-recovery attack on 6-round Ascon.

3. In our 7-round attack, different from previous cube-like attacks [DMP+15, HWX+, DEMS15b] which make the cube variables not multiply with each other in the first round, we let two of the cube variables multiply with each other to generate quadratic terms after the first round. In the second round, by restraining some bit conditions of the key and adding some auxiliary variables, the produced quadratic terms do not multiply with other monomials of cube variables. This solves the problem that there are no enough cube variables to launch a 7-round cube attack on Ascon.

4. By introducing *the cube-like key-subset technique*, we construct many new 65-dimension cubes on Ascon, whose cube sum after 7th round is zero when some key bit conditions are met. In other words, if we divide the full key space according to the key bit conditions into many key subsets, we can test the cube sum of different 65-dimension cubes to determine which subset the secret key falls into and then determine which key bit conditions the secret key meets. Finally, the full key space is divided into 63 key subsets and one small remaining set[1]. The computations of cube sums of different 65-dimension cubes are repeated on these subsets until the right key is recovered. If the right key is not recovered, we assume that it is in the remaining subset and search it for the right one. The time complexity of the 7-round key-recovery attack on Ascon is $2^{103.9}$. Moreover, if the key falls into a weak-key set, whose size is 2^{117}, the total complexity is reduced to 2^{77}.

Organization of the Paper

Section 1 gives some notations, a brief description of Ascon cipher, some related properties of S-box and our attack assumptions. In Section 2, we briefly describe the related works. Section 3 introduces models. Then, the new 5/6-round conditional cube attacks on Ascon are introduced in Section 4 and 5. In Section 6, some new 65-dimension cubes corresponding to some *partial divisors* are given, then 7-round conditional cube attack is launched on Ascon. Section 7 gives a discussion on Ascon-128a and a previous version Ascon v1.1. At last, we conclude this paper in Section 8.

1 Preliminaries

In this section, we will give some notations, a brief description of Ascon, some related properties of S-box, together with our attack assumptions.

1.1 Notations

S_i	the intermediate state after i-round, for example $S_{0.5}$ means the intermediate state after S-box in 1st round, esp. S_0 means the initial state of Ascon
$S_i[j]$	the jth word of S_i, $0 \leqslant j \leqslant 4$

[1]The remaining set means the secret key does not fall into any of the 63 key subsets.

Table 2: Parameters Set for Ascon

name	bit size				rounds		IV values
	key	nonce	tag	data block	p^a	p^b	
Ascon-128	128	128	128	64	12	6	$0x80400c0600000000$
Ascon-128a	128	128	128	128	12	8	$0x80800c0800000000$

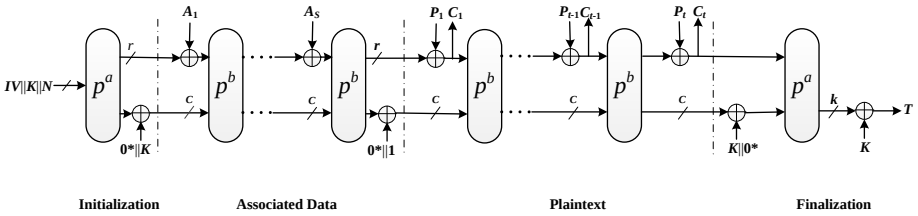

Figure 1: The Encryption of Ascon

$S_i[j][k]$	the kth bit of $S_i[j]$, $0 \leqslant j \leqslant 4$, $0 \leqslant k \leqslant 63$
v_i	the ith cube variable
$IV(i)$	the ith bit of IV, $0 \leqslant i \leqslant 63$
K	128-bit key, $K = k_0 \| k_1$
$k_0(i)$	the ith bit of k_0, k_0 is placed in $S_0[1]$, $0 \leqslant i \leqslant 63$
$k_1(i)$	the ith bit of k_1, k_1 is placed in $S_0[2]$, $0 \leqslant i \leqslant 63$
$n(i)$	the ith bit of $S_0[4]$, $0 \leqslant i \leqslant 63$

1.2 Brief Description of Ascon

Authenticated encryption cipher Ascon is one of the 16 candidates in 3rd round CAESAR competition, whose mode of operation is based on MonkeyDuplex [DPAB12]. We give a brief description of the latest version Ascon v1.2 [DEMS15a] proposed for the 3rd round CAESAR competition. It operates on a 320-bit state (five 64-bit words $x_0, ..., x_4$) in a sponge-like construction. Parameters of two flavors, Ascon-128 and Ascon-128a, are summarized in Table 2, while Ascon-128 is the primary recommendation by the designers. Readers can refer to [DEMS15a] for more details.

Mode. Based on MonkeyDuplex, the encryption of Ascon is organized in four phases as illustrated in Figure 1: initialization, processing associated data, processing the plaintext and finalization. In the initialization, concatenation of IV(64-bit), the secret key K(128-bit) and nonce N(128-bit) initializes the state of Ascon, where IV is constant in both flavors with values listed in Table 2. It is processed by p^a followed by XORing K. The associated 64-bit data block A_i is XORed and then p^b is applied to the intermediate state in sequence for $i = 1, ..., s$. A bit 1 is XORed to the least significant output bit of the last p^b in the process of associated data. Each plaintext block P_i with $i = 1, ..., t$ is processed similarly to A_i, while the corresponding C_i is outputted. In the finalization, K is XORed, then apply p^a. Finalization outputs T after the K is XORed to the least significant 128-bit.

Permutation. Permutations p^a and p^b only differ in the number of iterations of round function p, which is shown in Table 2. p is composed of a constant addition to x_2, the substitution layer and the linear diffusion layer. Some constants are added in different rounds. The substitution layer applies a 5-bit S-box as shown in Table 3 in parallel to each bit-slice of the five words $x_0, x_1, ..., x_4$, where x_0 acts as the most significant bit of the S-box. It will be explored further in Section 1.3. The linear diffusion layer, shown in Eq. (1), provides diffusion in each 64-bit state-word x_i with $\Sigma_i(x_i)$.

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S(x)$	4	11	31	20	26	21	9	2	27	5	8	18	29	3	6	28
x	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$S(x)$	30	19	7	14	0	13	17	24	16	12	1	25	22	10	15	23

Table 3: The 5-bit S-box in the Substitution Layer of p

$$\begin{aligned}
\Sigma_0(x_0) &= x_0 \oplus (x_0 \ggg 19) \oplus (x_0 \ggg 28)\\
\Sigma_1(x_1) &= x_1 \oplus (x_1 \ggg 61) \oplus (x_1 \ggg 39)\\
\Sigma_2(x_2) &= x_2 \oplus (x_2 \ggg 1) \oplus (x_2 \ggg 6)\\
\Sigma_3(x_3) &= x_3 \oplus (x_3 \ggg 10) \oplus (x_3 \ggg 17)\\
\Sigma_4(x_4) &= x_4 \oplus (x_4 \ggg 7) \oplus (x_4 \ggg 41)
\end{aligned} \tag{1}$$

1.3 Properties of S-box

Denote the 5-bit input and output of the S-box as x_0, x_1, x_2, x_3, x_4 and y_0, y_1, y_2, y_3, y_4 respectively, and we use x_0 to mark the most significant bit or the first register word of the S-box. The algebraic normal form (ANF) of the S-box is as follow:

$$\begin{aligned}
y_0 &= x_4x_1 + x_3 + x_2x_1 + x_2 + x_1x_0 + x_1 + x_0,\\
y_1 &= x_4 + x_3x_2 + x_3x_1 + x_3 + x_2x_1 + x_2 + x_1 + x_0,\\
y_2 &= x_4x_3 + x_4 + x_2 + x_1 + 1,\\
y_3 &= x_4x_0 + x_4 + x_3x_0 + x_3 + x_2 + x_1 + x_0,\\
y_4 &= x_4x_1 + x_4 + x_3 + x_1x_0 + x_1.
\end{aligned}$$

By studying the ANF of the S-box, the following two properties are given.

Property 1. *Among the 5-bit output of the S-box, x_4x_3 only exists in y_2.*

If we choose x_3 and x_4 as cube variables, then the quadratic term x_4x_3 only exists in the ANF of y_2.

Property 2. *x_2 will only multiply with x_1 and x_3. Especially, quadratic terms containing x_2 exist only in y_0 with x_2x_1 and y_1 with $x_3x_2 + x_2x_1$.*

If x_2 is a cube variable and x_1 is a nonce bit, then we can select $x_1 = 0$ to delete x_2x_1.

1.4 Our Attack Assumptions

The Ascon's design document [DEMS15a] shows, if there is no associated data, the associated data processing phase will be removed. In our attack, some input and output bits of p^a are needed, we omit the associated data processing phase too, i.e. our attack target is the initialization of Ascon as shown in Figure 2. This assumption is the same to [DEMS15b]. Our attacks work on both flavors Ascon-128 and Ascon-128a of the latest version Ascon v1.2, whose parameters are summarized in Table 2. We describe the analysis of 5/6/7-round initialization of Ascon-128 in details, then give a discussion on Ascon-128a and Ascon v1.1 in Section 7.

2 Related Work

2.1 Cube Attack

The cube attack [DS09] was introduced by Dinur and Shamir at EUROCRYPT 2009. It assumes that the output bit of a symmetric cryptographic scheme can be regarded as

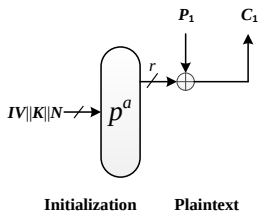

Figure 2: Objective Procedure of ASCON

a polynomial $f(k_0, ..., k_{n-1}, v_0, ..., v_{m-1})$ over $GF(2)$, $k_0, ..., k_{n-1}$ are the secret variables (the key bits), $v_0, ..., v_{m-1}$ are the public variables (e.g. IV or nonce bits).

Theorem 1. *([DS09])*

$$f(k_0, ..., k_{n-1}, v_0, ..., v_{m-1}) = T \cdot P + Q(k_0, ..., k_{n-1}, v_0, ..., v_{m-1}) \tag{2}$$

T is a monomial which is actually the product of certain public variables, for example $(v_0, ..., v_{s-1})$, $1 \leq s \leq m$, denoted as cube C_T. None of the monomials in Q is divisible by T. P is called superpoly, which does not contain any variables of C_T. Then the sum of f over all values of the cube C_T (cube sum) is

$$\sum_{v'=(v_0,...,v_{s-1})\in C_T} f(k_0, ..., k_{n-1}, v', v_s, ..., v_{m-1}) = P \tag{3}$$

where C_T contains all binary vectors of the length s, $v_s, ..., v_{m-1}$ are fixed to constant.

The basic idea is to find enough T whose P is linear and not a constant. This enables the key recovery through solving a system of linear equations.

2.2 Dynamic Cube Attack

Dynamic cube attack [DS11] was first introduced to analyse Grain-128 by Dinur and Shamir at FSE 2011. The basic idea is to simplify a complex polynomial P: $P = P_1 \cdot P_2 + P_3$ where P_3's degree is relatively lower than P, and P_1 contains a linear public term called a dynamic variable. A dynamic variable is a variable assigned with a function in some secret variables (i.e. key bits) and cube variables to zero P_1. Thus, P is simplified to P_3. One must firstly guess these key bits to compute dynamic variable. The right guess of key bits will lead to zero cube sums with high probability, otherwise the cube sums will be random.

2.3 Conditional Differential Cryptanalysis

Knellwolf, Meier and Naya-Plasenciaa [KMN10] applied conditional differential characteristic to NFSR-based constructions and extended to higher order differential attacks at ASIACRYPT 2010. The input of a synchronous stream cipher is an IV and a key. Suppose that the keystream for many chosen IVs under the same secret key can be observed. By imposing specific conditions on certain bits of the IV, the attacker can control the propagation of a difference through the first few rounds of the initialization process. Taking IV pairs conformed to these conditions as input, the resulting keystream differences will present a bias. Additionally, conditions upon key define classes of weak keys.

3 Cube-like Attack Models

In this section, we firstly generalize the *conditional cube attack*, which was first introduced by Huang *et al.* to attack Keccak keyed mode. Then a new *cube-like key-subset technique* is introduced. At last, the rationality tests about these attack models are presented.

3.1 Generalizing Conditional Cube Attack

Conditional cube attack [HWX+] was proposed by Huang *et al.* to attack Keccak keyed mode. Inspired by dynamic cube attack [DS09], which reduces the degree of output polynomials of cube variables by adding some bit conditions on IV, they reduce the degree by appending key bit conditions. The techniques are similar to message modification technique [WY05, WYY05] and conditional differential cryptanalysis [KMN10] which used bit conditions to control differential propagation. They also construct a cube tester based on cube variables and corresponding conditions, called *conditional cube tester*.

In this section, we will generalize the conditional cube attack. In Theorem 1, the cube sum P is calculated in Eq. (3). In Keccak-MAC and other similar ciphers, after certain rounds, the cipher produces l-bit output. Each of the output bits is written as a polynomial $f_i(k_0, ..., k_{n-1}, v_0, ..., v_{m-1})$, $i = 0, 1, ..., l-1$. Choose a common cube C_T, e.g $(v_0, ..., v_{s-1})$, $1 \le s \le m$, then $f_i = T \cdot P_i + Q_i$, $i = 0, 1, ..., l-1$. In conditional cube attack, a common divisor of P_i is found, which is a polynomial $g(k_0, ..., k_{n-1}, v_s, ..., v_{m-1})$. $v_s, ..., v_{m-1}$ are constant when computing cube sums $P_i = g(k_0, ..., k_{n-1}, v_s, ..., v_{m-1}) \cdot P'_i$. Then the Corollary 1 is given.

Corollary 1. *Given a series of polynomials f_i $(i \in \{0, 1, ..., l-1\})$:{0,1}$^n \to$ {0,1}.*

$$\begin{cases} f_0(k_0, ..., k_{n-1}, v_0, ..., v_{m-1}) = T \cdot g(k_0, ..., k_{n-1}, v_s, ..., v_{m-1}) \cdot P'_0 + Q_0 \\ f_1(k_0, ..., k_{n-1}, v_0, ..., v_{m-1}) = T \cdot g(k_0, ..., k_{n-1}, v_s, ..., v_{m-1}) \cdot P'_1 + Q_1 \\ ... \\ f_{l-1}(k_0, ..., k_{n-1}, v_0, ..., v_{m-1}) = T \cdot g(k_0, ..., k_{n-1}, v_s, ..., v_{m-1}) \cdot P'_{l-1} + Q_{l-1} \end{cases} \tag{4}$$

where none of the monomials in $Q_i(x)$ is divisible by T. Then the sums of f_i $(i \in \{0, 1, ..., l-1\})$ over all values of the cube (cube sum) are

$$\begin{cases} \sum\limits_{v' \in C_T} f_0(k_0, ..., k_{n-1}, v', v_s, ..., v_{m-1}) = g(k_0, ..., k_{n-1}, v_s, ..., v_{m-1}) \cdot P'_0 \\ \sum\limits_{v' \in C_T} f_1(k_0, ..., k_{n-1}, v', v_s, ..., v_{m-1}) = g(k_0, ..., k_{n-1}, v_s, ..., v_{m-1}) \cdot P'_1 \\ ... \\ \sum\limits_{v' \in C_T} f_{l-1}(k_0, ..., k_{n-1}, v', v_s, ..., v_{m-1}) = g(k_0, ..., k_{n-1}, v_s, ..., v_{m-1}) \cdot P'_{l-1} \end{cases} \tag{5}$$

where the C_T contains all binary vectors of the length s.

Among the output polynomials, a common factor which is related to key instead of any cube bits exists. g is introduced to represent the common factor for clearness.

As shown in Eq. (5), we get the following Property 3 and Assumption 1.

Property 3. *If $g = 0$, cube sums of f_i $(i \in \{0, 1, ..., l-1\})$ will be all 0 with probability 1.*

Assumption 1. *If $g = 1$, cube sums of f_i $(i \in \{0, 1, ..., l-1\})$ will be determined by P'_i $(i \in \{0, 1, ..., l-1\})$, the cube sums of f_i $(i \in \{0, 1, ..., l-1\})$ all equal to 0 with probability about 2^{-l} if f_i $(i \in \{0, 1, ..., l-1\})$ is a random oracle.*

According to Property 3 and Assumption 1, we introduce the cube tester, which has the Property 4 and Assumption 2.

Property 4. *If at least one nonzero cube sum occurs among the cube sums of f_i $(i \in \{0, 1, ..., l-1\})$, we will determine that $g = 1$. It is guaranteed to be right.*

Assumption 2. *If the cube sums of f_i ($i \in \{0, 1, ..., l-1\}$) all equal to 0, we will determine that $g = 0$. Note that, in a random oracle, $g = 0$ is wrong with probability of 2^{-l}, because P'_i is zero with probability of about $\frac{1}{2}$.*

If the common divisor $g(k_0, ..., k_{n-1}, v_s, ..., v_{m-1})$ is simple enough, e.g. $g = k_0$, we will use the cube tester of Property 4 and Assumption 2 to detect k_0.

3.2 The Cube-like Key-subset Technique

In Corollary 1, all cube sums $P_i = g(k_0, ..., k_{n-1}, v_s, ..., v_{m-1}) \cdot P'_i$ have a common divisor g, hence $g = 0$ will produce all zero cube sums illustrated in Property 3. When the l cube sums P_i do not have a common divisor, but the cube sums can be divided into several sets, there is a common divisor for cube sums in the same set, we denote these divisors as *partial divisor*. For example, there are three divisors g_1, g_2 and g_3 which are the common divisor of cube sums $\{P_0, P_1, ..., P_{19}\}$, $\{P_{20}, P_{21}, ..., P_{39}\}$ and $\{P_{40}, P_{41}, ..., P_{l-1}\}$, respectively. Then $g_1 = 0$, $g_2 = 0$ and $g_3 = 0$ will produce all zero cube sums which is similar to Property 3.

Specially, the *partial divisor*s g_1, g_2 and g_3 are some linear key-dependent polynomials, for example $g_1 = k_1 + a$, $g_2 = k_2 + b$ and $g_3 = k_3 + c$, where k_1, k_2 and k_3 are three key bit variables and (a, b, c) are in $\mathbb{F}_2^3$. If for each (a, b, c) in $\mathbb{F}_2^3$, a cube (denoted as $C_{T(abc)}$) is found, that maintain the $g_1 = k_1 + a$, $g_2 = k_2 + b$ and $g_3 = k_3 + c$ to be *partial divisor*s (which means $g_1 = 0$, $g_2 = 0$ and $g_3 = 0$ will produce all zero cube sums). Then we can divide the full 128-bit key space (we assume that it is 128-bit) by (a, b, c) to 2^3 different key subsets. At last, we process the cube testers of all the 2^3 cubes $C_{T(abc)}$ to determine which key subset the key candidate falls into. For example, according to Assumption 2, if the cube tester of $C_{T(000)}$ produce all zero cube sums, we get $k_1 = 0$, $k_2 = 0$ and $k_3 = 0$. Hence the key candidates are reduced from the 128-bit full key space to a subset of size 2^{125}, which meets the condition $k_1 = 0$, $k_2 = 0$ and $k_3 = 0$. We summarize those above techniques as *cube-like key-subset technique*.

3.3 Applications and Rationality test

We apply the cube tester of Property 4 in our attacks on 5/6-round reduced initialization of Ascon. And we apply Assumption 2 and *the cube-like key-subset technique* to 7-round reduced initialization of Ascon. For Ascon-128, the number of output bits are 64-bit (128-bit for Ascon-128a). The success rate of the corresponding cube tester is dependent on the density of monomial T in the output bits.

1. If the monomial T does not appear in anyone of f_i ($i = 0, 1, ..., 63$), the cube tester will fail to detect keys.

2. If the monomial T appears in some (a small fraction) of f_i ($i = 0, 1, ..., 63$), we can use Property 4 to detect the keys. When using Property 4, every detection returned is guaranteed to be right.

3. If the monomial T does appear in all (or most, e.g. l_1) of the f_i ($i = 0, 1, ..., 63$), we can use Property 4 or Assumption 2 to detect the keys. When using Assumption 2, wrong key detection is returned with probability of about 2^{-64} or 2^{-l_1}, besides, if there exist three partial divisors g_1, g_2 and g_3 as shown in Section 3.2, we get $g_1 = 0$, $g_2 = 0$ and $g_3 = 0$ with the same negligible probability of wrong detection.

Thus, for a given cube, we should firstly test if the monomial T appears in output f_i ($i = 0, 1, ..., 63$) of the round-reduced Ascon-128. For 5/6-round attack on Ascon in Section 4 and 5, the monomial of the chosen cubes appears in the output f_i, we use Property 4 to detect the right key. The 5/6-round attacks are practical and the key-recovery

programs are listed in https://github.com/lizhengcn/Ascon_test. But the 7-round attack is impractical, so in the following we present strong evidence for the correctness of our analysis. For 7-round attack, 65 (64+1) dimension cubes are chosen, so we have to test if the monomial T of 65 degree appears in outputs f_i, however, the test should compute the cube sums of 65-dimension cube, which is out of reach by the computing resource. We test similar cubes that reduced to 17 (16+1) dimension for 5-round and 33 (32+1) dimension for 6-round, to deduce the appearance of monomial of 65 (64+1) dimension cubes. Both **Test 1** and **Test 2** provide evidences for the correctness of our assumptions.

Test 1. We test 5-round cube sums of 1000 17-dimension random cubes: For each of the 1000 simulations, we first place 16 cube variables in $S_0[3][i_j]$ ($j \in \{0, 1, ..., 15\}$) randomly and place a cube variable in $S_0[4][i_0]$. The monomial (product of 17 variables) appears (that means the 64-bit cube sums of the 5-round are not all zero) in more than 850 simulations.

We test 6-round cube sum of 1000 33-dimension random cubes: For each of the 1000 simulations, we firstly place 32 cube variables in $S_0[3][i_j]$ ($j \in \{0, 1, ..., 31\}$) randomly, and place a cube variable in $S_0[4][i_0]$. The monomial (product of 33 variables) appears (that means the 64-bit cube sums of the 6-round are not all zero) in all the 1000 random tests.

So we deduce, in our attack for the 65-dimension cube (64 cube variables in $S_0[3][i_j]$ ($j \in \{0, 1, ..., 63\}$), one cube variable in $S_0[4][i_0]$), the monomial (product of 65 variables) will also appear with significantly high probability.

Test 2. We test 5-round cube sums over a 17-dimension random cube (cube variables are selected in the same way as Test 1.) with 1000 keys. The monomial T does really exist in some of f_i ($i = 0, 1, ..., 63$), as shown in Table 9.

We test 6-round cube sums over a 33-dimension random cube with 982 keys. In each one of f_i ($i = 0, 1, ..., 63$), the cube sum is nonzero with a probability of about $\frac{1}{2}$, as shown in Table 10, which implies the monomial T exists in all f_is. Note that if the monomial T does not exist in one of f_is, e.g. f_0, the cube sum of f_0 is definitely zero. Experiments verify Assumption 2 for our 6-round attack on Ascon, so we conjecture it will also hold for the 7-round attack.

The test programs are listed in https://github.com/lizhengcn/Ascon_test.

4 Attack on 5-round initialization of Ascon

Basic Ideas. In the 5-round attack, we select a set of 16 cube variables $\{v_0, v_1...v_{15}\}$. $\{v_i\}_{i=0,1,...,15}$ are located in distinct S-boxes, thus they do not multiply with each other in the first round. With some bit conditions on key and nonce, v_0 does not multiply with any of $\{v_1, v_2...v_{15}\}$ in the second round. Therefore, no v_0v_i exists in S_2. As the algebraic degree of the round function is 2, $v_0v_1...v_{15}$ will not appear in S_5, which means the cube sums of the output of 5-round Ascon over $v_0, v_1...v_{15}$ are all zero. If some of the cube sums are nonzero, bit conditions in the above are not met. According to Corollary 1, the bit conditions are actually determined by the common divisor of the cube sums.

In details, we construct a 16-dimension cube whose 64-bit cube sums have a common divisor $g = k_0(0)$. Similarly, we get $64 \times 4 = 256$ 16-dimension cubes corresponding to different common divisors: $k_0(t), k_0(t)+1, k_0(t)+k_1(t), k_0(t)+k_1(t)+1$ with $t \in \{0, 1, ..., 63\}$. Thus, we apply Property 4 to recover the key bits one by one. The cubes are listed in Table 7 in Appendix A.

An Example to Determine $\mathbf{k_0(0) = 1}$. For example, we select cube variables when $t = 0$ and set nonce according to (1) of Table 7. Figure 3 and 4 shows the diffusion of v_i in the first round with $k_0(0) = 0$ and $k_0(0) = 1$ respectively, in which yellow bits represent the diffusion of v_0, and grey ones represent the diffusion of $\{v_1, v_2 \dots v_{15}\}$, while a red

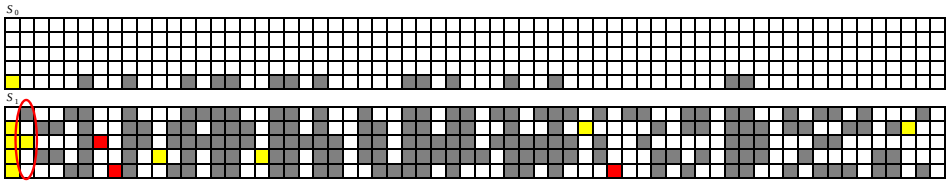

Figure 3: Diffusion of v_i with $k_0(0) = 0$

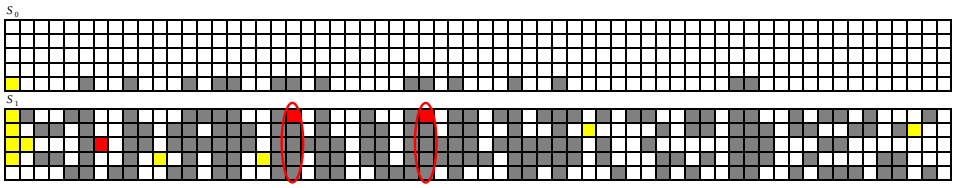

Figure 4: Diffusion of v_i with $k_0(0) = 1$

one's polynomial is in form $h_1v_0 + h_2v_i + h_3$ ($i \in \{1, \ldots, 15\}, h_1, h_2, h_3$ only depend on nonce and key).

If $k_0(0) = 0$, state S_1 of Figure 3 shows the distribution of v_i at the input of the S-box layer in the second round: yellow and red bits are located in different S-boxes from grey ones except for $S_1[0][1]$ and $S_1[2][1]$ highlighted in a red circle. And Property 2 tells us that $S_1[0][1]$ does not multiply with $S_1[2][1]$. So v_0 will not multiply with any of $\{v_i\}_{i=1,2,...,15}$ in the S-box operation of the second round and no v_0v_i exists in S_2. As the algebraic degree of the round function is 2, a term in S_5 with the highest degree is product of 8 terms in S_2. If this term contains v_0 as a factor, its degree over cube variables $\{v_i\}_{i=0,1,...,15}$ is at most 15 with all the other 7 terms being quadratic. If this term does not contain v_0, its degree over cube variables $\{v_i\}_{i=0,1,...,15}$ will not exceed 15 obviously. Thus, $v_0v_1...v_{15}$ does not appear in S_5.

However, if $k_0(0) = 1$, v_0 is certain to multiply with some of $\{v_i\}_{i=1,2,...,15}$ as highlighted in red circles in Figure 4. v_0v_i exists in S_2. $v_0v_1...v_{15}$ will appear in S_5 with a high probability. It means that $g = k_0(0)$ in Corollary 1. Therefore, if we obtain a nonzero cube sum, we can determine that $k_0(0) = 1$, which is identical to Property 4.

The Whole Procedure to Recover the Full Key. With parameters in Table 7, 1-values for $k_0(t), k_0(t) + 1, k_0(t) + k_1(t), k_0(t) + k_1(t) + 1$ with $t \in \{0, 1, ..., 63\}$ are detected by Property 4. Therefore, 0/1-values for $k_0(t), k_0(t) + k_1(t)$ with $t \in \{0, 1, ..., 63\}$ are detected. We describe the procedure in Algorithm 1.

We illustrate some special bit positions here. As a constant $0x00000000000000000f0$ is added to $S_0[2]$ before S-box in the first round, in which only 4 bits are 1 indexed by 56,57,58,59, $k_0(t) + k_1(t)$ should reverse when $t \in \{56, 57, 58, 59\}$. In Ascon-128 only six bits of IV value are 1, whose indexes are $\{0, 9, 20, 21, 29, 30\}$, and other IV bits are 0, it leads to a special cube selection (not rotated from others) to detect 1-value of $k_0(t) + k_1(t) + 1$ for $t \in \{0, 9, 20, 21, 29, 30\}$.

The whole procedure is performed as follow: with parameters in Table 7, for $t \in \{0, 1, \ldots, 63\}$, if we obtain a nonzero cube sum in (1), we can determine that $k_0(t) = 1$, if we obtain a nonzero cube sum in (2), we can determine that $k_0(t) = 0$. For $t \in \{0, 1, \ldots, 63\}$ while $t \neq 56, 57, 58, 59$, if we obtain a nonzero cube sum in (3), we can determine that $k_0(t) + k_1(t) = 1$; for $t \in \{56, 57, 58, 59\}$, if we obtain a nonzero cube sum in (3), we can determine that $k_0(t) + k_1(t) = 0$; for $t \in \{0, 9, 20, 21, 29, 30\}$, if we obtain a nonzero cube sum in (4), we can determine that $k_0(t) + k_1(t) = 0$; for $t \in \{0, 1, \ldots, 63\}$ while $t \neq 0, 9, 20, 21, 29, 30, 56, 57, 58, 59$, if we obtain a nonzero cube sum in (5), we can determine

that $k_0(t) + k_1(t) = 0$; for $t \in \{56, 57, 58, 59\}$, if we obtain a nonzero cube sum in (5), we can determine that $k_0(t) + k_1(t) = 1$.

Computation of a cube sum need 2^{16} (nonce, $P_1 \oplus C_1$) (as shown in Figure 2) pairs, and each pair corresponds to an operation of initialization of Ascon. Thus, time and data share the same complexity in cube computation. As each cube sum consumes 2^{16}, tester described in Algorithm 1 with Table 7 needs at most $64 \times 2 \times 2 \times 2^{16} = 2^{24}$. In our experiments, *remain* returned by Algorithm 1 with Table 7 is always less than 14. For 2^{14} is far less than 2^{24}, the whole time complexity is 2^{24}. The data complexity is 2^{24} as well. Source code is in https://github.com/lizhengcn/Ascon_test.

5 Attack on 6-round initialization of Ascon

Similar to the attack on 5-round, we choose cube $\{v_0, v_1...v_{31}\}$ obeying the following rules:

1. Any two of $\{v_0, v_1...v_{31}\}$ do not multiply in the S-box operation of the first round.
2. If $k_0(t)$=0, v_0 doesn't multiply with any of $\{v_1, v_2...v_{31}\}$ in the S-box operation of the second round.
3. If $k_0(t)$=1, v_0 multiplies with some of $\{v_1, v_2...v_{31}\}$ in the S-box operation of the second round.

Thus, if $k_0(t)$=0, no $v_0v_i, i \in \{1, 2, ..., 31\}$, exists in S_2, and $v_0v_1...v_{31}$ does not appear in S_6. However, if $k_0(t) = 1$, $v_0v_1...v_{31}$ will appear in S_6 with a high probability. It means that $g = k_0(t)$ in Corollary 1. Therefore, if we obtain a nonzero cube sum of the output of 6-round Ascon over $v_0, v_1...v_{31}$ with parameters in (1) of Table 8, we can determine that $k_0(t) = 1$, which is identical to Property 4. The whole procedure to recover the whole key is similar to the attack on 5-round initialization of Ascon in Section 4 applying Algorithm 1 with parameters in Table 8.

As each cube sum consumes 2^{32} (time and data as shown in Section 4), tester described in Algorithm 1 with Table 8 needs at most $64 \times 2 \times 2 \times 2^{32} = 2^{40}$. In our experiments, *remain* returned by Algorithm 1 with Table 8 is always 0. The whole time complexity is 2^{40}. The data complexity is 2^{40} as well. Source code is in https://github.com/lizhengcn/Ascon_test.

6 Attack on 7-round initialization of Ascon

In this section, we apply the so-called *the cube-like key-subset technique* to the 7-round key-recovery attack on Ascon. We find a series sets of *partial divisors* related to the key bits for many 65-dimension cubes. That means, if the *partial divisors* in a set equal to zero, the cube sums of the corresponding 65-dimension cube will be zero. The full key space is divided into many subsets indexed by the key bits in each set of *partial divisors*. Then *the cube-like key-subset technique* is applied to recover the key.

In detail, we set $S_0[3][j] = v_j$ for $j = 0, 1 \ldots 63$ and $S_0[4][i] = v_{64}$ where i could take a value from $\{0, 1 \ldots 63\}$. We denote the 65-dimension cube set as *original cube set*. As $v_0, v_1, \ldots, v_{63}$ belong to distinct S-boxes, they do not multiply with each other in the first round. While v_{64} and v_i lie in the same ith S-box, v_iv_{64} will be the only quadratic term after the first substitution layer. Based on Property 1, v_iv_{64} just appear in $S_{0.5}[2][i]$ after the S-box in the first round. The first linear diffusion layer makes the ANF of $S_1[2][i]$, $S_1[2][i+1]$[2] and $S_1[2][i+6]$ contain v_iv_{64}. According to our **Test 1** and **Test 2** in Section 3.3, the monomial $v_0v_1 \cdots v_{64}$ will appear in the output bits of 7-round. However, by assigning some key bit conditions, the quadratic terms v_iv_{64} will not multiply with other cube variables in the second round. Thus, after the second round, only quadratic and linear terms exist (no terms exist with degree more than 2). Then in the output bits of

[2] $i + 1$ is short for $(i + 1) mod\ 64$, similarly $i + x$ is short for $(i + x) mod\ 64$ in this section.

Algorithm 1 Tester in Attack on 5/6-round Initialization of Ascon.
Note: cube sum $C_i[t]$ means sum of the output of 5/6-round Ascon *over cube variables in* (i) *of Table 7/ 8; k_0, $k_0 + k_1$ record values of key bits recovered; $flag[0][t] = 1$ represents $k_0(t)$ has been recovered, $flag[0][t] = 0$ represents $k_0(t)$ hasn't been recovered, and $flag[1][t]$ for $k_0(t)+k_1(t)$ similarly; remain represents the number of key bits that need to be searched exhaustively.*

Require: k_0, k_1, $flag[2][64]$ all set to 0
Ensure: $remain$, $k_0, k_0 + k_1$ and $flag[2][64]$
all elements of $k_0, k_0 + k_1, remain, flag[2][64]$ are set to 0
for $t \in \{0, 1, ..., 63\}$ **do**
compute cube sum $C_1[t]$;
if $C_1[t] \neq 0$ **then**
$k_0(t) = 1$; $flag[0][t] = 1$;
else
compute cube sum $C_2[t]$;
if $C_2[t] \neq 0$ **then**
$k_0(t) = 0$; $flag[0][t] = 1$;
end if
end if
compute cube sum $C_3[t]$;
if $C_3[t] \neq 0$ **then**
if $t \in \{56, 57, 58, 59\}$ **then**
$k_0(t) + k_1(t) = 0$; $flag[1][t] = 1$;
else
$k_0(t) + k_1(t) = 1$; $flag[1][t] = 1$;
end if
else
if $t \in \{0, 9, 20, 21, 29, 30\}$ **then**
compute cube sum $C_4[t]$;
if $C_4[t] \neq 0$ **then**
$k_0(t) + k_1(t) = 0$; $flag[1][t] = 1$;
end if
else
compute cube sum $C_5[t]$;
if $C_5[t] \neq 0$ **then**
if $t \in \{56, 57, 58, 59\}$ **then**
$k_0(t) + k_1(t) = 1$; $flag[1][t] = 1$;
else
$k_0(t) + k_1(t) = 0$; $flag[1][t] = 1$;
end if
end if
end if
end if
end for
for $i \in \{0, 1\}$ **do**
for $t \in \{0, 1 \ldots 63\}$ **do**
if $flag[i][t] == 0$ **then**
$remain$++;
end if
end for
end for
return $remain, k_0, k_0 + k_1$ and $flag[2][64]$;

Table 4: Coefficients of Cubic Terms with No Additional Cube Set

index of S-box	cubic terms	corresponding coefficients (*partial divisors*)
$i+1$	$v_i v_{64} v_{i+1}$	$k_0(i+1)+k_1(i+1)+1$
		$k_0(i+1)+k_1(i+1)+IV(i+1)$
	$v_i v_{64} v_{i+4}$	$k_0(i+4)+k_1(i+4)+1$
	$v_i v_{64} v_{i+26}$	$k_0(i+26)+k_1(i+26)+1$
	$v_i v_{64} v_{i+48}$	$IV(i+48)+1$
	$v_i v_{64} v_{i+55}$	$IV(i+55)+1$
i	$v_i v_{64} v_{i+3}$	$k_0(i+3)+k_1(i+3)+1$
	$v_i v_{64} v_{i+25}$	$k_0(i+25)+k_1(i+25)+1$
	$v_i v_{64} v_{i+47}$	$IV(i+47)+1$
	$v_i v_{64} v_{i+54}$	$IV(i+54)+1$
$i+6$	$v_i v_{64} v_{i+6}$	$k_0(i+6)+k_1(i+6)+1$
		$k_0(i+6)+k_1(i+6)+IV(i+6)$
	$v_i v_{64} v_{i+9}$	$k_0(i+9)+k_1(i+9)+1$
	$v_i v_{64} v_{i+31}$	$k_0(i+31)+k_1(i+31)+1$
	$v_i v_{64} v_{i+53}$	$IV(i+53)+1$
	$v_i v_{64} v_{i+61}$	$IV(i+60)+1$

7-round, the degree of the product of cube variables is at most 64. In fact, the assigned key bit conditions lead all the coefficients of the cubic terms $v_i v_{64} v_{j'}$ ($j' = 0, 1 \ldots 63$ and $j' \neq i$) after the second round to be zero, which makes the disappearance of the monomial of degree 65 in the output bits of 7-round. **Those coefficients are actually the so-called *partial divisors* of the cube sums of the 65-dimension cube.** In the following, we will extract the *partial divisors* along with key bit conditions.

Based on Property 2, only $S_1[1][i]$, $S_1[3][i]$ multiply with $S_1[2][i]$, and similarly $S_1[1][i+1]$, $S_1[3][i+1]$ multiply with $S_1[2][i+1]$, and $S_1[1][i+6]$, $S_1[3][i+6]$ multiply with $S_1[2][i+6]$. In detail, for the input of the $(i+1)$th S-box of second round, we list the ANF of $S_1[1][i+1]$, $S_1[3][i+1]$ and $S_1[2][i+1]$ in Eq. (6) as an example, similar equations can be obtained for the input of the ith and $(i+6)$th S-box. Note that $S_0[4][t] = n(t)$ for $t \in \{0, 1 \ldots 63\}$ while $t \neq i$. According to Property 1 and 2, in $(i+1)$th S-box of second round, only cubic terms $v_i v_{64} v_{i+1}$, $v_i v_{64} v_{i+4}$, $v_i v_{64} v_{i+26}$, $v_i v_{64} v_{i+48}$, $v_i v_{64} v_{i+55}$ are possibly produced by terms underlined in Eq. (6). We display all the possible cubic terms in $S_{1.5}$ and their corresponding coefficients in Table 4.

$$
\begin{aligned}
S_1[1][i+1] &= \underline{(k_0(i+1)+k_1(i+1)+1)*v_{i+1} + (k_0(i+4)+k_1(i+4)+1)*v_{i+4}} \\
&\underline{+(k_0(i+26)+k_1(i+26)+1)*v_{i+26}} + \boldsymbol{n(i+1) + n(i+4) + n(i+26)} + k_0(i+1)* \\
&k_1(i+1) + k_0(i+1) + k_0(i+4)*k_1(i+4) + k_0(i+4) + k_0(i+26)*k_1(i+26) + \\
&k_0(i+26) + k_1(i+1) + k_1(i+4) + k_1(i+26) + IV(i+1) + IV(i+4) + IV(i+26) \\
S_1[2][i+1] &= \underline{v_i * v_{64}} + k_0(i) + k_0(i+1) + k_0(i+59) + k_1(i) + k_1(i+1) + k_1(i+ \\
&59) + n(i+1)*v_{i+1} + n(i+1) + n(i+59)*v_{i+59} + n(i+59) + v_{i+64} \\
S_1[3][i+1] &= \underline{(IV(i+1)+1)*v_{i+1} + (IV(i+48)+1)*v_{i+48} + (IV(i+55)+1)*} \\
&\underline{v_{i+55}} + \boldsymbol{(IV(i+1)+1)*n(i+1) + (IV(i+48)+1)*n(i+48) + (IV(i+55)+1)*} \\
&\boldsymbol{n(i+55)} + k_0(i+1) + k_0(i+48) + k_0(i+55) + k_1(i+1) + k_1(i+48) + k_1(i+55) + \\
&IV(i+1) + IV(i+48) + IV(i+55)
\end{aligned}
\tag{6}
$$

CASE 1: all the IV bits in Table 4 are 0.

As $IV(j) = 0$ for $j \in \{0, 1 \ldots 63\}$ while $j \neq 0, 9, 20, 21, 29, 30$, we firstly consider the case that all the relative bits in IV are 0. To achieve a zero cube sum, all the coefficients in Table 4 should be 0. However, in Table 4, if all the relative IV bits are 0, then the coefficients $k_0(i+1) + k_1(i+1) + 1$, $k_0(i+1) + k_1(i+1) + IV(i+1)$ can not be zero simultaneously, and $IV(i+48) + 1$, $IV(i+55) + 1$ etc are certain to equal 1. To solve this problem, we introduce some auxiliary cube variables to zero the coefficients. These auxiliary cube variables are introduced as follows in Definition 1.

Definition 1. *For any value of key, some additional nonce bits in $S_0[4]$ should be set as cube variables to eliminate the cases that no key can zero all the coefficients in Table 4, the additional cube variable is called auxiliary cube variable. Auxiliary cube variable is necessary to nullify cubic terms in $S_{1.5}$. It should be noted that $S_0[4][j]$ is set as v_j which is equal to $S_0[3][j]$ for some given $j \in \{0, 1, ..., 63\}$.*

If the added auxiliary cube variables are $S_0[4][i+1] = n(i+1) = v_{i+1}, S_0[4][i+48] = n(i+48) = v_{i+48}, S_0[4][i+55] = n(i+55) = v_{i+55}$, in the ANF of $S_1[3][i+1]$ of Eq. (6), parts of the underline formula and the bold formula will be equal and cancelled out. Then the Eq. (6) is simplified as follow in Eq. (7).

$$\begin{aligned}
S_1[1][i+1] &= \underline{(k_0(i+1) + k_1(i+1)) * v_{i+1} + (k_0(i+4) + k_1(i+4) + 1) * v_{i+4}} \\
&\underline{+(k_0(i+26)+k_1(i+26) + 1) * v_{i+26}} + \boldsymbol{n(i+4) + n(i+26)} + k_0(i+1) * k_1(i+1) \\
&+ k_0(i+1) + k_0(i+4) * k_1(i+4) + k_0(i+4) + k_0(i+26) * k_1(i+26) + k_0(i+26) \\
&+ k_1(i+1) + k_1(i+4) + k_1(i+26) + IV(i+1) + IV(i+4) + IV(i+26) \\
S_1[2][i+1] &= \underline{v_i * v_{64}} + k_0(i) + k_0(i+1) + k_0(i+59) + k_1(i) + k_1(i+1) + \\
&k_1(i+59) + n(i+1) * v_{i+1} + n(i+1) + n(i+59) * v_{i+59} + n(i+59) + v_{i+64} \\
S_1[3][i+1] &= k_0(i+1) + k_0(i+48) + k_0(i+55) + k_1(i+1) + k_1(i+48) + k_1(i+55) \\
&IV(i+1) + IV(i+48) + IV(i+55)
\end{aligned} \tag{7}$$

Then, compared to Table 4, the coefficients of $v_i v_{64} v_{i+48}$ and $v_i v_{64} v_{i+55}$ turn to 0, and $k_0(i+1)+k_1(i+1)$ becomes the only coefficient of $v_i v_{64} v_{i+1}$ instead of $k_0(i+1)+k_1(i+1)+1$ and $k_0(i+1)+k_1(i+1)+IV(i+1)$. Additionally, we can add more auxiliary cube variables to zero more coefficients as shown in the new coefficients Table 5. Hence, the nonzero coefficients of the cubic terms are all determined by some key bits. We set these coefficients to be zero and get a equation set Eq. (8). Hence, if a secret key conforms to Eq. (8), then all the coefficients in Table 5 are 0 and there will be no cubic terms after second round.

$$\begin{cases}
k_0(i+1) + k_1(i+1) = 0 \\
k_0(i+4) + k_1(i+4) = 1 \\
k_0(i+26) + k_1(i+26) = 1 \\
k_0(i+3) + k_1(i+3) = 1 \\
k_0(i+25) + k_1(i+25) = 1 \\
k_0(i+6) + k_1(i+6) = 0 \\
k_0(i+9) + k_1(i+9) = 1 \\
k_0(i+31) + k_1(i+31) = 1
\end{cases} \tag{8}$$

In order to get more different key conditions that delete the cubic terms, we introduce the control cube variable which is defined as follows.

Table 5: Coefficients of Cubic Terms with Auxiliary Cube Variables

index of S-box	cubic terms	auxiliary cube variables	corresponding coefficients (*partial divisors*)
$i+1$	$v_iv_{64}v_{i+1}$	$S_0[4][i+1]=v_{i+1}$	$k_0(i+1)+k_1(i+1)$
	$v_iv_{64}v_{i+4}$		$k_0(i+4)+k_1(i+4)+1$
	$v_iv_{64}v_{i+26}$		$k_0(i+26)+k_1(i+26)+1$
	$v_iv_{64}v_{i+48}$	$S_0[4][i+48]=v_{i+48}$	0
	$v_iv_{64}v_{i+55}$	$S_0[4][i+55]=v_{i+55}$	0
i	$v_iv_{64}v_{i+3}$		$k_0(i+3)+k_1(i+3)+1$
	$v_iv_{64}v_{i+25}$		$k_0(i+25)+k_1(i+25)+1$
	$v_iv_{64}v_{i+47}$	$S_0[4][i+47]=v_{i+47}$	0
	$v_iv_{64}v_{i+54}$	$S_0[4][i+54]=v_{i+54}$	0
$i+6$	$v_iv_{64}v_{i+6}$	$S_0[4][i+6]=v_{i+6}$	$k_0(i+6)+k_1(i+6)$
	$v_iv_{64}v_{i+9}$		$k_0(i+9)+k_1(i+9)+1$
	$v_iv_{64}v_{i+31}$		$k_0(i+31)+k_1(i+31)+1$
	$v_iv_{64}v_{i+53}$	$S_0[4][i+53]=v_{i+53}$	0
	$v_iv_{64}v_{i+61}$	$S_0[4][i+60]=v_{i+60}$	0

Definition 2. *When some additional nonce bit in $S_0[4]$ set as cube variable, the value of coefficients in Table 5 will be XORed by 1. The additional cube variable is called* control cube variable. *Control cube variable is tweakable to provide different coefficients of cubic terms in $S_{1.5}$ according to the recovery of different keys. $S_0[4][j]$ is set as v_j which is equal to $S_0[3][j]$ for some given $j \in \{0,1,...,63\}$.*

For example, we add $S_0[4][i+4]=n(i+4)=v_{i+4}$ as a control cube variable, the Eq. (7) is changed to be Eq. (9). In the ANF of $S_1[1][i+1]$ of Eq. (7), the bold formula ***n(i+4)*** and $(k_0(i+4)+k_1(i+4)+1)*v_{i+4}$ are added to be $(k_0(i+4)+k_1(i+4))*v_{i+4}$. Obviously, the coefficient is changed.

$$
\begin{aligned}
S_1[1][i+1] &= \underline{(k_0(i+1)+k_1(i+1))*v_{i+1}+(k_0(i+4)+k_1(i+4))*v_{i+4}} \\
&\underline{+(k_0(i+26)+k_1(i+26)+1)*v_{i+26}} + \boldsymbol{n(i+26)} + k_0(i+1)*k_1(i+1) \\
&+k_0(i+1)+k_0(i+4)*k_1(i+4)+k_0(i+4)+k_0(i+26)*k_1(i+26)+k_0(i+26) \\
&+k_1(i+1)+k_1(i+4)+k_1(i+26)+IV(i+1)+IV(i+4)+IV(i+26) \\
S_1[2][i+1] &= \underline{v_i*v_{64}}+k_0(i)+k_0(i+1)+k_0(i+59)+k_1(i)+k_1(i+1)+ \\
&k_1(i+59)+n(i+1)*v_{i+1}+n(i+1)+n(i+59)*v_{i+59}+n(i+59)+v_{i+64} \\
S_1[3][i+1] &= k_0(i+1)+k_0(i+48)+k_0(i+55)+k_1(i+1)+k_1(i+48)+k_1(i+55)
\end{aligned}
\tag{9}
$$

Differences among *original cube variables*, *auxiliary cube variables* and *control cube variables*: We firstly select a 65-dimension *original cube* which occupies $S_0[3]$ and 1-bit of $S_0[4]$. After the second round, cubic terms appear with coefficients determined by key bits and IV bits. We would like to zero those coefficients, however, some coefficients can not be zero, e.g. the coefficient of $v_iv_{64}v_{i+48}$ is $IV(i+55)+1$ while $IV(i+55)=0$, shown in Table 4. Hence, *auxiliary cube variables* are introduced to zero these shown in Table 5. Then we get key condition equation set – Eq. (8) to zero those coefficients. When the secret key meets Eq. (8), the cube sums will be zero. However, it is just a weak key attack that covers a key subset of size 2^{120}. We want get many more such key subsets to extend the weak key attack to a full key attack. Hence, the tweakable *control cube variables* are introduced to provide different coefficients of cubic terms and get different key condition equation sets to cover different key subsets. So when an *original cube* is

Table 6: Coefficients of Cubic Terms with Auxiliary and Control Cube Variable

	cubic terms	auxiliary cube variables	control cube variable	corresponding coefficients
$i+1$	$v_i v_{64} v_{i+1}$	$S_0[4][i+1] = v_{i+1}$		$k_0(i+1) + k_1(i+1)$
	$v_i v_{64} v_{i+4}$		$S_0[4][i+4] = v_{i+4}$	$k_0(i+4) + k_1(i+4)$
	$v_i v_{64} v_{i+26}$			$k_0(i+26) + k_1(i+26) + 1$
	$v_i v_{64} v_{i+48}$	$S_0[4][i+48] = v_{i+48}$		0
	$v_i v_{64} v_{i+55}$	$S_0[4][i+55] = v_{i+55}$		0
i	$v_i v_{64} v_{i+3}$			$k_0(i+3) + k_1(i+3) + 1$
	$v_i v_{64} v_{i+25}$			$k_0(i+25) + k_1(i+25) + 1$
	$v_i v_{64} v_{i+47}$	$S_0[4][i+47] = v_{i+47}$		0
	$v_i v_{64} v_{i+54}$	$S_0[4][i+54] = v_{i+54}$		0
$i+6$	$v_i v_{64} v_{i+6}$	$S_0[4][i+6] = v_{i+6}$		$k_0(i+6) + k_1(i+6)$
	$v_i v_{64} v_{i+9}$			$k_0(i+9) + k_1(i+9) + 1$
	$v_i v_{64} v_{i+31}$			$k_0(i+31) + k_1(i+31) + 1$
	$v_i v_{64} v_{i+53}$	$S_0[4][i+53] = v_{i+53}$		0
	$v_i v_{64} v_{i+61}$	$S_0[4][i+60] = v_{i+60}$		0

fixed, the *auxiliary cube variables* are also fixed, but *control cube variables* are tweakable to cover different key subsets. Those are the differences between the three cube variables.

We list coefficients of cubic terms in Table 6 after adding control cube variable $S_0[4][i+4] = n(i+4) = v_{i+4}$. Compared to Table 5, only $k_0(i+4) + k_1(i+4) + 1$ is changed to $k_0(i+4) + k_1(i+4)$. Key conditions required for a zero cube sum are adjusted to Eq. (10) from Eq. (8) by a control cube variable $S_0[4][i+4] = v_{i+4}$.

$$\begin{cases} k_0(i+1) + k_1(i+1) = 0 \\ k_0(i+4) + k_1(i+4) = 0 \\ k_0(i+26) + k_1(i+26) = 1 \\ k_0(i+3) + k_1(i+3) = 1 \\ k_0(i+25) + k_1(i+25) = 1 \\ k_0(i+6) + k_1(i+6) = 0 \\ k_0(i+9) + k_1(i+9) = 1 \\ k_0(i+31) + k_1(i+31) = 1 \end{cases} \tag{10}$$

Similarly, for $t \in \{3, 4, 9, 25, 26, 31\}$, control cube variable $S_0[4][i+t] = v_{i+t}$ can change the corresponding kind of coefficients from $k_0(i+t) + k_1(i+t) + 1$ to $k_0(i+t) + k_1(i+t)$. Therefore, there are $2^6 = 64$ kinds of control cube variable combinations corresponding to 64 groups of coefficients respectively. Hence, there are 64 different equation sets correspondingly shown in Eq. (11), where $(a, b, c, d, e, f) \in F_2^6$ varies according to different control cube variable combination.

$$\begin{cases} k_0(i+1) + k_1(i+1) = 0 \\ k_0(i+4) + k_1(i+4) = a \\ k_0(i+26) + k_1(i+26) = b \\ k_0(i+3) + k_1(i+3) = c \\ k_0(i+25) + k_1(i+25) = d \\ k_0(i+6) + k_1(i+6) = 0 \\ k_0(i+9) + k_1(i+9) = e \\ k_0(i+31) + k_1(i+31) = f \end{cases} \tag{11}$$

When key meets the corresponding conditions, all the coefficients of cubic terms will be zero. It means that there are no cubic terms in $S_{1.5}$. Due to linearity of the linear diffusion layer, no cubic terms exist in S_2. The highest degree of monomials in S_2 is 2, which has also been verified using SAGE [WD05] by computer. As the algebraic degree of S-box is 2, degree of a term in the output of 7-round Ascon is at most $2^5 = 32$ times of the highest degree in S_2 (i.e. 2). Thus, the algebraic degree of the 7-round Ascon's output is less than or equal to 64, which means that $v_0 v_1 \dots v_{64}$ will not appear in the output, then the cube sum of cube $\{v_0, v_1, \dots, v_{64}\}$ is zero. When key doesn't meet the corresponding conditions, some cubic terms will appear in S_2. Therefore, $v_0 v_1 \dots v_{64}$ will appear in the output of 7-round.

In details, for a key subset where the key meets the two equations of $k_0(i+1)+k_1(i+1) = 0$ and $k_0(i+6)+k_1(i+6) = 0$ in Eq. (11), we determine the values of (a, b, c, d, e, f) by testing the cube sum of each of 64 65-dimension cubes indexed by control cube variable combination. The time complexity of the test is $2^{65} \times 64 = 2^{71}$. We denote this key subset as U_i which is determined by the two key conditions $k_0(i+1)+k_1(i+1) = 0$ and $k_0(i+6)+k_1(i+6) = 0$, whose size is $2^{128} \times \frac{1}{4}$. Else, we can know that $k_0(i+1)+k_1(i+1)$ and $k_0(i+6)+k_1(i+6)$ do not equal to 0 at the the same time. Note that the cube variable v_{64} is placed in $S_0[4][i]$, where $i \in \{0, 1, ..., 63\}$, however, in **CASE 1**, we suppose the related bits in IV are zero, so there are 52 such U_is in total that are determined by 52 sets of two bits key conditions.

CASE 2: some *IV* bits in Table 4 are 1.

As shown in Table 4, if $IV(i+1) = 1$, then the coefficients $k_0(i+1)+k_1(i+1)+1$ and $k_0(i+1)+k_1(i+1)+IV(i+1)$ are translated to the same one $k_0(i+1)+k_1(i+1)+1$. Then, there is no need to add the auxiliary cube variables $S_0[4][i+1] = v_{i+1}$. The freed auxiliary cube variable can be used as control cube variable. In fact, if we use $S_0[4][i+1] = v_{i+1}$ as a control cube variable, $k_0(i+1)+k_1(i+1)+1$ becomes another coefficient that can be changed to $k_0(i+1)+k_1(i+1)$. Then Eq. (11) is translated to Eq. (12), where $(a, b, c, d, e, f, g) \in F_2^7$ varies according to different control cube variable combinations. Therefore, similar to the previous analysis, if a key meets the one bit key condition of $k_0(i+6)+k_1(i+6) = 0$, we can obtain the value of (a, b, c, d, e, f, g) by testing the cube sum of the 2^7 65-dimension cubes. We denote those keys as a key subset U_i', whose size is 2^{127}. Else, we claim $k_0(i+6)+k_1(i+6) = 1$. The time complexity of the test is $2^{65} \times 2^7 = 2^{72}$.

$$\begin{cases} k_0(i+1)+k_1(i+1) = a \\ k_0(i+4)+k_1(i+4) = b \\ k_0(i+26)+k_1(i+26) = c \\ k_0(i+3)+k_1(i+3) = d \\ k_0(i+25)+k_1(i+25) = e \\ k_0(i+6)+k_1(i+6) = 0 \\ k_0(i+9)+k_1(i+9) = f \\ k_0(i+31)+k_1(i+31) = g \end{cases} \tag{12}$$

Since there are 6 bits of IV are 1, i.e. $IV(i+1) = 1$ where $(i+1) \in \{0, 9, 20, 21, 29, 30\}$, there are 6 similar equation sets as Eq. (12). For each equation set, there is a bit key

condition correspondingly, which are listed as follows in Eq. (13) to (18):

$$k_0(5) + k_1(5) = 0 \tag{13}$$
$$k_0(14) + k_1(14) = 0 \tag{14}$$
$$k_0(25) + k_1(25) = 0 \tag{15}$$
$$k_0(26) + k_1(26) = 0 \tag{16}$$
$$k_0(34) + k_1(34) = 0 \tag{17}$$
$$k_0(35) + k_1(35) = 0 \tag{18}$$

Hence, there are 6 such U_i' key subsets corresponding to the above 6 one-bit key conditions, respectively.

Similarly, as shown in Table 4, if $IV(i+6) = 1$, then the auxiliary cube variable $S_0[4][i+6] = v_{i+6}$ is translated as control cube variable. Then, Eq. (11) is translated as Eq. (19), where $(a, b, c, d, e, f, g) \in F_2^7$ varies according to different control cube variable combination. We can not translate $S_0[4][i+1] = v_{i+1}$ and $S_0[4][i+6] = v_{i+6}$ to control cube variable simultaneously, because $IV(i+1) = 1$ and $IV(i+6) = 1$ can not hold simultaneously according to the given IV.

$$\begin{cases} k_0(i+1) + k_1(i+1) = 0 \\ k_0(i+4) + k_1(i+4) = a \\ k_0(i+26) + k_1(i+26) = b \\ k_0(i+3) + k_1(i+3) = c \\ k_0(i+25) + k_1(i+25) = d \\ k_0(i+6) + k_1(i+6) = e \\ k_0(i+9) + k_1(i+9) = f \\ k_0(i+31) + k_1(i+31) = g \end{cases} \tag{19}$$

Similarly, for different $IV(i+6) = 1$ where $(i+6) \in \{0, 9, 20, 21, 29, 30\}$, there are totally 6 similar equation sets as Eq. (19). For each equation set, there is a bit key condition correspondingly, which are also listed as follows in Eq. (20) to (25):

$$k_0(59) + k_1(59) = 0 \tag{20}$$
$$k_0(4) + k_1(4) = 0 \tag{21}$$
$$k_0(15) + k_1(15) = 0 \tag{22}$$
$$k_0(16) + k_1(16) = 0 \tag{23}$$
$$k_0(24) + k_1(24) = 0 \tag{24}$$
$$k_0(25) + k_1(25) = 0 \tag{25}$$

Note that Eq. (15) and (25) are equal. So there are totally 12-1=11 such U_i' key subsets, where $i \in \{63, 8, 19, 20, 28, 29, 58, 3, 14, 15, 23\}$[3], corresponding to the above 11 different one-bit key conditions. Each U_i' key subset's size is 2^{127}. In **CASE 1**, the indexes i of the 52 U_i key subsets are in $\{0, 1, 2, ..., 63\} - \{63, 8, 19, 20, 28, 29, 58, 3, 14, 15, 23, 30\}$.

The total attack procedures are as follows:

(i) Test the cube sum of 2^7 65-dimension cubes obtained in CASE 2 and determine 8-bit key information. If all the cube sums are not zero, the key is not in the corresponding key subset U_i'.

[3]The indexes i represent the positions of cube variable v_{64}, i.e. $S_0[4][i] = v_{64}$

(ii) Repeat step (i) for 11 times according to different one bit key condition shown in Eq. (13) to Eq. (24), which cover all U_i's .

(iii) Test the cube sum of 2^6 64-dimension cubes obtained in CASE 1 and determine 8-bit key information. If all the cube sums are not zero, then the key is not in the corresponding key subset U_i.

(iv) Repeat step (iii) for 52 times to cover all U_is.

The time complexity of the above 4 procedures is $2^7 \times 2^{65} \times 11 + 2^6 \times 2^{65} \times 52 = 2^{77.2}$.

In the worst case, if the secret key does not fall into any U_i and U_i', then the key falls into a *remaining key subset* whose size is about $2^{103.92}$. We explain the range of the *remaining key subset* in details.

We propose a fast key filter phase to get the *remaining key subset* in the following. We introduce a array of index $KI_0 = \{5, 14, 25, 26, 34, 35, 59, 4, 15, 16, 24\}$. In the *remaining key subset*, any key does not fall into any U_i', which means any key does not obey Eq. (13) to (18) and Eq. (20) to (25). Hence, the keys meet the following conditions in Eq. (26):

$$k_0(i) + k_1(i) = 1, i \in KI_0. \tag{26}$$

For each of $(k_0(i), k_1(i))(i \in KI_0)$, only 2 guesses out of 2^2 values of $(k_0(i), k_1(i))$ obey the corresponding one of Eq. (26). Each filtration ratio is $\frac{1}{2}$, so 11 $(k_0(i), k_1(i))$ lead a ratio 2^{-11}. Additionally, any key in the *remaining key subset* does not fall into any U_i, which means $k_0(i+1) + k_1(i+1) = 0$ and $k_0(i+6) + k_1(i+6) = 0$ can not be obeyed at the same time in Eq. (11). It can be converted to the following tester:

$$(k_0(i+1) + k_1(i+1) + 1)(k_0(i+6) + k_1(i+6) + 1) = 0 \tag{27}$$

If we consider the four bits independently, 2^4 key guesses over $k_0(i+1), k_1(i+1), k_0(i+6), k_1(i+6)$ will be filtered by Eq. (27) with filtration ratio of $\frac{3}{4}$.

As the index varies, it seems that 52 testers for key bits can be performed independently. However, the relevant key bits intersect for 52 testers. Firstly, we classify the indexes $i+1$ and $i+6$ for k_0 and k_1 in Eq. (27) into the following arrays KI_1, KI_2 and KI_3.

$$\begin{cases} KI_1 = \{40, 45, 50, 55, 60, 1, 6, 11\} \\ KI_2 = \{31, 36, 41, 46, 51, 56, 61, 2, 7, 12, 17, 22, 27, 32, 37, 42, \\ \qquad 47, 52, 57, 62, 3, 8, 13, 18, 23, 28, 33, 38, 43, 48, 53, 58, 63\} \\ KI_3 = \{39, 44, 49, 54\} \end{cases} \tag{28}$$

KI_j doesn't intersect with each other for $j \in \{0, 1, 2, 3\}$. In array KI_1, suppose that i_0 is an arbitrary element and i_1 is next to it, then Eq. (29) should be hold.

$$(k_0(i_0) + k_1(i_0) + 1)(k_0(i_1) + k_1(i_1) + 1) = 0 \tag{29}$$

For 2^8 guesses of $k_0(i) + k_1(i), i \in KI_1$, 55 values meet above conditions Eq. (29). So KI_1 provides a filtration ratio of $\frac{55}{2^8}$. Similarly, for $KI_j(j \in \{2, 3\})$, the filtration ratios are $\frac{9227465}{2^{33}}$, $\frac{8}{2^4}$, respectively. Due to the disappearances of key in any U_i of **CASE 1**, it provides a filtration ratio equals $\frac{55}{2^8} \times \frac{9227465}{2^{33}} \times \frac{8}{2^4} = 2^{-13.08}$. The disappearance of key in any U_i and U_i' gives a total filtration ratio of $2^{-11} \times 2^{-13.08}$. Since, the filter phase can be performed for each parts of the key independently, the time complexity is about 2^{11} checks of Eq. (26) and $2^8 + 2^{33} + 2^4 = 2^{33.1}$ checks of Eq. (29).

After the filter phase, all the remaining keys should be checked by a $(nonce, plaintext, ciphertext, tag)$ pair to determine the right one. The size *remaining key subset* size is

about $2^{128-11} \times 2^{-13.08} = 2^{103.92}$. So in the worst case, the total complexity is $2^{11} + 2^{33.1} + 2^{103.92} + 2^{77.2} = 2^{103.92}$.

In the best case, if the secret key falls into the intersection of all 11 U_i's, denoted as *weak-key set*, whose size is $2^{128-11} = 2^{117}$. 52 bits[4] key information are recovered and other $128 - 52 = 76$ bits key are recovered by exhaustive search. The total time complexity to recover the right key for the 2^{117} *weak-key set* is about $2^{76} + 2^7 \times 2^{65} \times 11 = 2^{77}$.

7 Discussion on Ascon-128a and Ascon v1.1

For attacks on 5/6-round initialization of Ascon-128a, with the same cubes and conditions to Ascon-128, key recovery can be performed in the same time and data complexity.

In the attack on 7-round initialization of Ascon, as **CASE 2** described in Section 6, it is considered specially when IV bits are 1. Since the IVs used by Ascon-128 and Ascon-128a are slightly different, the attack on 7-round Ascon-128a is a little different from that on Ascon-128. The details are described in Appendix C. The total time complexity of the 7-round attack on Ascon-128a is about $2^{103.45}$.

Ascon v1.1 is a previous version. As the tweak only affects the ordering of constants in p^b, our attacks can also be applied to it.

8 Conclusion

This paper improves the previous cube-like attack on Ascon by the *generalized conditional cube attack* and *the cube-like key-subset technique*. Different from previous cube-like method or conditional cube method [DMP+15, HWX+, DEMS15b], we release the restriction that all cube variables must not multiply with each other in the first round. This makes it possible for the 7-round key-recovery attack on Ascon which needs more cube variables to construct some new cubes with bigger dimension. In each of the new cubes, two cube variables multiply with each other to generate quadratic terms after the first round. In the second round, by restraining some bit conditions of the key and adding some auxiliary variables, the quadratic terms of cube variables do not multiply with other monomials of cube variables. Interestingly, we finally divide the full key space into 63 key subsets and one remaining set according to different key bit conditions. In each subset, some 65-dimension cubes are constructed. Then, by testing the cube sum of these cubes, we determine which subsets the secret key falls into. If all the cube tests fail, we claim that the key falls into the remaining set and search it for the right one. This leads to a 7-round key-recovery attack on Ascon with time complexity $2^{103.92}$. Moreover, if the key falls into a weak-key set, whose size is 2^{117}, the total complexity is reduced to 2^{77}. We also give the first practical key-recovery attack on 6-round Ascon. Those are the best attacks on the round-reduced Ascon.

Acknowledgments

We would like to thank Itai Dinur, Willi Meier, Senyang Huang and the anonymous reviewers who helped improve this paper. This work is supported by China's 973 Program (No. 2013CB834205), the Strategic Priority Research Program of the Chinese Academy of Sciences (No. XDB01010600), the National Natural Science Foundation of China (No. 61672019 and 61402256), the Fundamental Research Funds of Shandong University (No. 2016JC029), and the Foundation of Science and Technology on Information Assurance Laboratory (No. KJ-15-002).

[4]Note that for 11 U_i's, we get 11 sets of equations, such as Eq. (19). totally $11 \times 8 = 88$ linear equations on key bits are retrieved. However, the linear equations may repeat, and only 52 linear equations are left.

References

[ADMS09] Jean-Philippe Aumasson, Itai Dinur, Willi Meier, and Adi Shamir. Cube testers and key recovery attacks on reduced-round MD6 and trivium. In Orr Dunkelman, editor, *FSE 2009*, volume 5665 of *Lecture Notes in Computer Science*, pages 1–22. Springer, 2009.

[BDP+16] Guido Berton, Joan Daemen, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. CAESAR submission: KETJE v2, 2016. http://competitions.cr.yp.to/round3/ketjev2.pdf.

[com14] The CAESAR committee. CAESAR: Competition for authenticated encryption: Security, applicability, and robustness, 2014. http://competitions.cr.yp.to/caesar.html.

[DEMS15a] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1.1. submission to the CAESAR competition, 2015. http://competitions.cr.yp.to/round2/asconv11.pdf.

[DEMS15b] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Cryptanalysis of ascon. In Kaisa Nyberg, editor, *CT-RSA 2015*, volume 9048 of *Lecture Notes in Computer Science*, pages 371–387. Springer, 2015.

[DLJW15] Xiaoyang Dong, Leibo Li, Keting Jia, and Xiaoyun Wang. Improved attacks on reduced-round camellia-128/192/256. In Kaisa Nyberg, editor, *CT-RSA 2015*, volume 9048 of *Lecture Notes in Computer Science*, pages 59–83. Springer, 2015.

[DLWQ17] Xiaoyang Dong, Zheng Li, Xiaoyun Wang, and Ling Qin. Cube-like Attack on Round-Reduced Initialization of Ketje Sr. *IACR Trans. Symmetric Cryptol.*, 2017(1), 2017. http://eprint.iacr.org/2017/159.

[DMP+15] Itai Dinur, Pawel Morawiecki, Josef Pieprzyk, Marian Srebrny, and Michal Straus. Cube attacks and cube-attack-like cryptanalysis on the round-reduced keccak sponge function. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015*, volume 9056 of *Lecture Notes in Computer Science*, pages 733–761. Springer, 2015.

[DPAB12] Joan Daemen, Michaël Peeters, Gilles Van Assche, and Guido Bertoni. Permutation-based encryption, authentication and authenticated encryption. In *Workshop Records of DIAC 2012*, pages 159–170, 2012.

[DS09] Itai Dinur and Adi Shamir. Cube attacks on tweakable black box polynomials. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 278–299. Springer, 2009.

[DS11] Itai Dinur and Adi Shamir. Breaking grain-128 with dynamic cube attacks. In Antoine Joux, editor, *FSE 2011*, volume 6733, pages 167–187. Springer, 2011.

[FV13] Pierre-Alain Fouque and Thomas Vannet. Improving key recovery to 784 and 799 rounds of trivium using optimized cube attacks. In Shiho Moriai, editor, *FSE 2013*, volume 8424 of *Lecture Notes in Computer Science*, pages 502–517. Springer, 2013.

[HWX+] Senyang Huang, Xiaoyun Wang, Guangwu Xu, Meiqin Wang, and Jingyuan Zhao. Conditional cube attack on reduced-round KECCAK sponge function. *EUROCRYPT 2017* (*to appear*). http://eprint.iacr.org/2016/790.

[KMN10] Simon Knellwolf, Willi Meier, and María Naya-Plasencia. Conditional differential cryptanalysis of nlfsr-based cryptosystems. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 130–145. Springer, 2010.

[LJWD15] Leibo Li, Keting Jia, Xiaoyun Wang, and Xiaoyang Dong. Meet-in-the-middle technique for truncated differential and its applications to CLEFIA and camellia. In Gregor Leander, editor, *FSE 2015*, volume 9054, pages 48–70. Springer, 2015.

[NIS] NIST. Advanced encryption standard (AES) (november 2001), federal information processing standards publication fips 197.

[WD05] Stein William and Joyner David. SAGE: system for algebra and geometry experimentation. *ACM SIGSAM Bulletin*, 39(2):61–64, 2005.

[WY05] Xiaoyun Wang and Hongbo Yu. How to break MD5 and other hash functions. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 19–35. Springer, 2005.

[WYY05] Xiaoyun Wang, Hongbo Yu, and Yiqun Lisa Yin. Efficient collision search attacks on SHA-0. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2005.

A Parameters sets

Parameters sets for attacks on 5/6-round initialization of Ascon are listed in Table 7/8 respectively.

B Test 2

B.1 Test 2 for 5-round initialization of Ascon-128

We test 5-round cube sums over a 17-dimension random cube with 1000 keys. The random cube set in two-word nonce is as Figure 5, where each grey bit represents one cube variable in the two-word nonce. For each bit of the 64-bit output of 5-round initialization of Ascon-128, the accumulation and the ratio of nonzero cube sums among 1000 keys are listed in Table 9.

Figure 5: The Random Cube Set with Dimension 17

B.2 Test 2 for 6-round initialization of Ascon-128

We test 6-round cube sums over a 33-dimension random cube with 982 keys. The random cube set in two-word nonce is as Figure 6, where each grey bit represents one cube variable in the two-word nonce. For each bit of the 64-bit output of 6-round initialization of Ascon-128, the accumulation and the ratio of nonzero cube sums among 982 keys are listed in Table 10.

Figure 6: The Random Cube Set with Dimension 33

C Attacks on 7-round initialization of Ascon-128a

The complexity of the attack on 7-round initialization of Ascon-128a is analysed carefully here. With 0 refers to the most significant bit, in Ascon-128 six bits of IV value 1, whose indexes are $\{0, 9, 20, 21, 29, 30\}$, while it turns to 5 bits for Ascon-128a, whose indexes are $\{0, 8, 20, 21, 28\}$. We compute the complexity of the worst case similar to Ascon-128. The range of the *remaining key subset* is analysed as follow. We denote the index set of key bits certainly determined by **CASE 2** as

$$KI_0 = \{4, 12, 24, 25, 32, 53, 61, 13, 10, 17\} \tag{30}$$

For each of $(k_0(i), k_1(i))(i \in KI_0)$, only 2 guesses out of 2^2 values of $(k_0(i), k_1(i))$ obey the corresponding one of Eq. (26) with KI_0 in Eq. (30). Each filtration ratio is $\frac{1}{2}$, so 10 $(k_0(i), k_1(i))$ lead a ratio 2^{-10}. Additionally, the index sets of key bits certainly influenced

Table 7: Parameters Set for Attack on the 5-round Initialization of Ascon

$k_0(t)$		
(1)	cube variables	$S_0[4][t]=v_0$,$S_0[4][5+t]^{\dagger}=v_1$,$S_0[4][8+t]=v_2$,$S_0[4][12+t]=v_3$, $S_0[4][14+t]=v_4$,$S_0[4][15+t]=v_5$,$S_0[4][18+t]=v_6$, $S_0[4][19+t]=v_7$,$S_0[4][21+t]=v_8$,$S_0[4][27+t]=v_9$, $S_0[4][28+t]=v_{10}$,$S_0[4][30+t]=v_{11}$,$S_0[4][34+t]=v_{12}$, $S_0[4][37+t]=v_{13}$,$S_0[4][49+t]=v_{14}$,$S_0[4][50+t]=v_{15}$.
	nonce	$S_0[3][t]=0$,$S_0[3][5+t]=0$,$S_0[3][8+t]=0$,$S_0[3][12+t]=0$, $S_0[3][14+t]=0$,$S_0[3][15+t]=0$,$S_0[3][18+t]=0$,$S_0[3][19+t]=0$, $S_0[3][21+t]=0$,$S_0[3][27+t]=0$,$S_0[3][28+t]=0$,$S_0[3][30+t]=0$, $S_0[3][34+t]=0$,$S_0[3][37+t]=0$,$S_0[3][49+t]=0$,$S_0[3][50+t]=0$.
	key information	If a cube sum is nonzero, $k_0(t)=1$.
(2)	cube variables	$S_0[4][t]=v_0$,$S_0[4][5+t]=v_1$,$S_0[4][7+t]=v_2$,$S_0[4][8+t]=v_3$, $S_0[4][14+t]=v_4$,$S_0[4][15+t]=v_5$,$S_0[4][24+t]=v_6$, $S_0[4][27+t]=v_7$,$S_0[4][30+t]=v_8$,$S_0[4][34+t]=v_9$, $S_0[4][37+t]=v_{10}$,$S_0[4][41+t]=v_{11}$,$S_0[4][43+t]=v_{12}$, $S_0[4][49+t]=v_{13}$,$S_0[4][50+t]=v_{14}$,$S_0[4][52+t]=v_{15}$.
	nonce	$S_0[3][t]=0$,$S_0[3][5+t]=0$,$S_0[3][7+t]=0$,$S_0[3][8+t]=0$, $S_0[3][14+t]=0$,$S_0[3][15+t]=0$,$S_0[3][24+t]=0$,$S_0[3][27+t]=0$, $S_0[3][30+t]=0$,$S_0[3][34+t]=0$,$S_0[3][37+t]=0$,$S_0[3][41+t]=0$, $S_0[3][43+t]=0$,$S_0[3][49+t]=0$,$S_0[3][50+t]=0$,$S_0[3][52+t]=0$.
	key information	If a cube sum is nonzero, $k_0(t)=0$.
$k_0(t)+k_1(t)^{\ddagger}$		
(3)	cube variables	$S_0[3][t]=S_0[4][t]=v_0$,$S_0[3][1+t]=v_1$,$S_0[3][4+t]=v_2$, $S_0[3][5+t]=v_3$,$S_0[3][6+t]=v_4$,$S_0[3][8+t]=v_5$,$S_0[3][14+t]=v_6$, $S_0[3][15+t]=v_7$,$S_0[3][16+t]=v_8$,$S_0[3][17+t]=v_9$, $S_0[3][20+t]=v_{10}$,$S_0[3][26+t]=v_{11}$,$S_0[3][27+t]=v_{12}$, $S_0[3][29+t]=v_{13}$,$S_0[3][30+t]=v_{14}$,$S_0[3][33+t]=v_{15}$.
	nonce	$S_0[4][1+t]=0$,$S_0[4][4+t]=0$,$S_0[4][5+t]=0$,$S_0[4][6+t]=0$, $S_0[4][8+t]=0$,$S_0[4][14+t]=0$,$S_0[4][15+t]=0$,$S_0[4][16+t]=0$, $S_0[4][17+t]=0$,$S_0[4][20+t]=0$,$S_0[4][26+t]=0$,$S_0[4][27+t]=0$, $S_0[4][29+t]=0$,$S_0[4][30+t]=0$,$S_0[4][33+t]=0$.
	key information	If a cube sum is nonzero, $k_0(t)+k_1(t)=1$.
(4)	cube variables	$S_0[3][t]=v_0$,$S_0[3][5+t]=v_1$,$S_0[3][8+t]=v_2$,$S_0[3][14+t]=v_3$, $S_0[3][15+t]=v_4$,$S_0[3][16+t]=v_5$,$S_0[3][17+t]=v_6$, $S_0[3][20+t]=v_7$,$S_0[3][27+t]=v_8$,$S_0[3][29+t]=v_9$, $S_0[3][30+t]=v_{10}$,$S_0[3][33+t]=v_{11}$,$S_0[3][34+t]=v_{12}$, $S_0[3][35+t]=v_{13}$,$S_0[3][37+t]=v_{14}$,$S_0[3][38+t]=v_{15}$.
	key information	If a cube sum is nonzero, $k_0(t)+k_1(t)=0$.
(5)	cube variables	$S_0[3][t]=v_0$,$S_0[3][5+t]=v_1$,$S_0[3][8+t]=v_2$,$S_0[3][14+t]=v_3$, $S_0[3][15+t]=v_4$,$S_0[3][27+t]=v_5$,$S_0[3][29+t]=v_6$, $S_0[3][30+t]=v_7$,$S_0[3][34+t]=v_8$,$S_0[3][36+t]=v_9$, $S_0[3][37+t]=v_{10}$,$S_0[3][38+t]=v_{11}$,$S_0[3][39+t]=v_{12}$, $S_0[3][45+t]=v_{13}$,$S_0[3][49+t]=v_{14}$,$S_0[3][50+t]=v_{15}$.
	key information	If a cube sum is nonzero, $k_0(t)+k_1(t)=0$.

†: (5+t) means (5+t) mod 64, similarly (x+t) means (x+t) mod 64 in Table 7 and 8.
‡: $k_0(t)+k_1(t)$ should reverse when $t \in \{56,57,58,59\}$ in Table 7 and 8.

Table 8: Parameters Set for Attack on the 6-round Initialization of Ascon

$k_0(t)$		
(1)	cube	$S_0[3][t]=S_0[4][t]=v_0$,$S_0[3][1+t]=v_1$,$S_0[3][4+t]=v_2$,$S_0[3][5+t]=v_3$, $S_0[3][6+t]=v_4$,$S_0[3][7+t]=v_5$,$S_0[3][8+t]=v_6$,$S_0[3][10+t]=v_7$, $S_0[3][13+t]=v_8$,$S_0[3][14+t]=v_9$,$S_0[3][15+t]=v_{10}$,$S_0[3][16+t]=v_{11}$, $S_0[3][17+t]=v_{12}$,$S_0[3][24+t]=v_{13}$,$S_0[3][26+t]=v_{14}$,$S_0[3][27+t]=v_{15}$, $S_0[3][30+t]=v_{16}$,$S_0[3][34+t]=v_{17}$,$S_0[3][35+t]=v_{18}$,$S_0[3][37+t]=v_{19}$, $S_0[3][40+t]=v_{20}$,$S_0[3][41+t]=v_{21}$,$S_0[3][43+t]=v_{22}$,$S_0[3][46+t]=v_{23}$, $S_0[3][48+t]=v_{24}$,$S_0[3][49+t]=v_{25}$,$S_0[3][50+t]=v_{26}$,$S_0[3][52+t]=v_{27}$, $S_0[3][56+t]=v_{28}$,$S_0[3][59+t]=v_{29}$,$S_0[3][60+t]=S_0[4][60+t]=v_{30}$, $S_0[3][63+t]=S_0[4][63+t]=v_{31}$.
	key	If a cube sum is nonzero, $k_0(t)=1$.
(2)	cube	$S_0[3][t]=S_0[4][t]=v_0$,$S_0[3][1+t]=v_1$,$S_0[3][4+t]=v_2$,$S_0[3][5+t]=v_3$, $S_0[3][6+t]=v_4$,$S_0[3][8+t]=v_5$,$S_0[3][9+t]=v_6$,$S_0[3][12+t]=v_7$, $S_0[3][14+t]=v_8$,$S_0[3][15+t]=v_9$,$S_0[3][16+t]=v_{10}$,$S_0[3][17+t]=v_{11}$, $S_0[3][18+t]=v_{12}$,$S_0[3][19+t]=v_{13}$,$S_0[3][21+t]=v_{14}$,$S_0[3][26+t]=v_{15}$, $S_0[3][27+t]=v_{16}$,$S_0[3][28+t]=v_{17}$,$S_0[3][30+t]=v_{18}$,$S_0[3][34+t]=v_{19}$, $S_0[3][35+t]=v_{20}$,$S_0[3][37+t]=v_{21}$,$S_0[3][40+t]=v_{22}$,$S_0[3][46+t]=v_{23}$, $S_0[3][48+t]=v_{24}$,$S_0[3][49+t]=v_{25}$,$S_0[3][50+t]=v_{26}$,$S_0[3][53+t]=v_{27}$, $S_0[3][56+t]=v_{28}$,$S_0[3][59+t]=v_{29}$,$S_0[3][60+t]=S_0[4][60+t]=v_{30}$, $S_0[3][63+t]=S_0[4][63+t]=v_{31}$.
	key	If a cube sum is nonzero, $k_0(t)=0$.
$k_0(t)+k_1(t)$		
(3)	cube	$S_0[3][t]=S_0[4][t]=v_0$,$S_0[3][1+t]=v_1$,$S_0[3][4+t]=v_2$,$S_0[3][5+t]=v_3$, $S_0[3][6+t]=v_4$,$S_0[3][8+t]=v_5$,$S_0[3][14+t]=v_6$,$S_0[3][15+t]=v_7$, $S_0[3][16+t]=v_8$,$S_0[3][17+t]=v_9$,$S_0[3][20+t]=v_{10}$,$S_0[3][26+t]=v_{11}$, $S_0[3][27+t]=v_{12}$,$S_0[3][29+t]=v_{13}$,$S_0[3][30+t]=v_{14}$,$S_0[3][33+t]=v_{15}$, $S_0[3][34+t]=v_{16}$,$S_0[3][35+t]=v_{17}$,$S_0[3][37+t]=v_{18}$,$S_0[3][38+t]=v_{19}$, $S_0[3][39+t]=v_{20}$,$S_0[3][40+t]=v_{21}$,$S_0[3][46+t]=v_{22}$,$S_0[3][48+t]=v_{23}$, $S_0[3][49+t]=v_{24}$,$S_0[3][50+t]=v_{25}$,$S_0[3][55+t]=v_{26}$,$S_0[3][56+t]=v_{27}$, $S_0[3][58+t]=v_{28}$,$S_0[3][59+t]=v_{29}$,$S_0[3][62+t]=v_{30}$,$S_0[3][63+t]=v_{31}$.
	key	If a cube sum is nonzero, $k_0(t)+k_1(t)=1$.
(4)	cube	$S_0[3][t]=v_0$,$S_0[3][1+t]=v_1$,$S_0[3][3+t]=v_2$,$S_0[3][4+t]=v_3$, $S_0[3][5+t]=v_4$,$S_0[3][6+t]=v_5$,$S_0[3][8+t]=v_6$,$S_0[3][14+t]=v_7$, $S_0[3][15+t]=v_8$,$S_0[3][16+t]=v_9$,$S_0[3][17+t]=v_{10}$,$S_0[3][20+t]=v_{11}$, $S_0[3][26+t]=v_{12}$,$S_0[3][27+t]=v_{13}$,$S_0[3][29+t]=v_{14}$,$S_0[3][30+t]=v_{15}$, $S_0[3][33+t]=v_{16}$,$S_0[3][34+t]=v_{17}$,$S_0[3][35+t]=v_{18}$,$S_0[3][37+t]=v_{19}$, $S_0[3][38+t]=v_{20}$,$S_0[3][39+t]=v_{21}$,$S_0[3][40+t]=v_{22}$,$S_0[3][46+t]=v_{23}$, $S_0[3][49+t]=v_{24}$,$S_0[3][50+t]=v_{25}$,$S_0[3][55+t]=v_{26}$,$S_0[3][58+t]=v_{27}$, $S_0[3][59+t]=v_{28}$,$S_0[3][60+t]=v_{29}$,$S_0[3][62+t]=v_{30}$,$S_0[3][63+t]=v_{31}$.
	nonce	$S_0[4][t]=0$.
	key	If a cube sum is nonzero, $k_0(t)+k_1(t)=0$.
(5)	cube	$S_0[3][t]=v_0$,$S_0[3][1+t]=v_1$,$S_0[3][4+t]=v_2$,$S_0[3][5+t]=v_3$,$S_0[3][6+t]=v_4$, $S_0[3][8+t]=v_5$,$S_0[3][9+t]=S_0[4][9+t]=v_6$,$S_0[3][11+t]=S_0[4][11+t]=v_7$, $S_0[3][14+t]=v_8$,$S_0[3][15+t]=v_9$,$S_0[3][16+t]=v_{10}$, $S_0[3][18+t]=S_0[4][18+t]=v_{11}$,$S_0[3][24+t]=S_0[4][24+t]=v_{12}$, $S_0[3][26+t]=v_{13}$,$S_0[3][27+t]=v_{14}$,$S_0[3][29+t]=v_{15}$,$S_0[3][30+t]=v_{16}$, $S_0[3][34+t]=v_{17}$,$S_0[3][36+t]=v_{18}$,$S_0[3][37+t]=v_{19}$,$S_0[3][38+t]=v_{20}$, $S_0[3][39+t]=v_{21}$,$S_0[3][45+t]=v_{22}$,$S_0[3][47+t]=S_0[4][47+t]=v_{23}$, $S_0[3][48+t]=v_{24}$,$S_0[3][49+t]=v_{25}$,$S_0[3][50+t]=v_{26}$,$S_0[3][56+t]=v_{27}$, $S_0[3][58+t]=v_{28}$,$S_0[3][59+t]=v_{29}$,$S_0[3][60+t]=v_{30}$,$S_0[3][63+t]=v_{31}$.
	nonce	$S_0[4][t]=0$,$S_0[4][4+t]=0$,$S_0[4][16+t]=0$,$S_0[4][58+t]=0$,$S_0[4][63+t]=0$.
	key	If a cube sum is nonzero, $k_0(t)+k_1(t)=0$.

Table 9: Statistics of Nonzero Cube Sums in the 64-bit Output of 5-round Ascon-128

bit	#nonzero	%nonzero	bit	#nonzero	%nonzero
0	0	0	32	125	0.125
1	268	0.268	33	110	0.11
2	0	0	34	0	0
3	0	0	35	6	0.006
4	198	0.198	36	37	0.037
5	91	0.091	37	350	0.35
6	75	0.075	38	0	0
7	14	0.014	39	92	0.092
8	31	0.031	40	282	0.282
9	92	0.092	41	0	0
10	74	0.074	42	0	0
11	16	0.016	43	20	0.02
12	123	0.123	44	0	0
13	0	0	45	6	0.006
14	0	0	46	106	0.106
15	75	0.075	47	0	0
16	0	0	48	92	0.092
17	22	0.022	49	4	0.004
18	86	0.086	50	61	0.061
19	0	0	51	75	0.075
20	92	0.092	52	31	0.031
21	4	0.004	53	0	0
22	0	0	54	0	0
23	125	0.125	55	0	0
24	95	0.095	56	272	0.272
25	0	0	57	0	0
26	26	0.026	58	0	0
27	31	0.031	59	156	0.156
28	92	0.092	60	0	0
29	0	0	61	19	0.019
30	16	0.016	62	28	0.028
31	134	0.134	63	0	0

Table 10: Statistics of Nonzero Cube Sums in the 64-bit Output of 6-round Ascon-128

bit	#nonzero	%nonzero	bit	#nonzero	%nonzero
0	512	0.521385	32	488	0.496945
1	528	0.537678	33	483	0.491853
2	494	0.503055	34	506	0.515275
3	489	0.497963	35	487	0.495927
4	501	0.510183	36	503	0.51222
5	499	0.508147	37	488	0.496945
6	481	0.489817	38	487	0.495927
7	492	0.501018	39	499	0.508147
8	472	0.480652	40	491	0.5
9	501	0.510183	41	479	0.48778
10	472	0.480652	42	483	0.491853
11	479	0.48778	43	465	0.473523
12	504	0.513238	44	494	0.503055
13	491	0.5	45	478	0.486762
14	511	0.520367	46	492	0.501018
15	500	0.509165	47	474	0.482688
16	474	0.482688	48	499	0.508147
17	485	0.49389	49	479	0.48778
18	484	0.492872	50	508	0.517312
19	482	0.490835	51	486	0.494908
20	469	0.477597	52	479	0.48778
21	491	0.5	53	500	0.509165
22	492	0.501018	54	487	0.495927
23	493	0.502037	55	496	0.505092
24	477	0.485743	56	495	0.504073
25	519	0.528513	57	477	0.485743
26	498	0.507128	58	497	0.50611
27	507	0.516293	59	489	0.497963
28	484	0.492872	60	494	0.503055
29	501	0.510183	61	494	0.503055
30	493	0.502037	62	484	0.492872
31	482	0.490835	63	483	0.491853

by **CASE 1** is as follow:

$$\begin{cases} KI_1 = \{30, 35, 40, 45, 50, 55, 60, 1, 6, 11, 16, 21, 26, 31, 36, 41, 46, 51, 56\} \\ KI_2 = \{2, 7\} \\ KI_3 = \{9, 14, 19\} \\ KI_4 = \{15, 20\} \\ KI_5 = \{28, 33, 38, 43, 48\} \\ KI_6 = \{29, 34, 39, 44, 49, 54, 59, 0, 5\} \\ KI_7 = \{37, 42, 47, 52, 57, 62, 3, 8\} \\ KI_8 = \{58, 63\} \end{cases} \tag{31}$$

As shown in Eq. (30) and (31), KI_j doesn't intersect with each other for $j \in \{0, 1, ..., 8\}$. In each of $KI_j (j \in \{1, ..., 8\})$, suppose that i_0 is an arbitrary index and i_1 is next to it, then Eq. (32) should be hold.

$$(k_0(i_0) + k_1(i_0) + 1)(k_0(i_1) + k_1(i_1) + 1) = 0 \tag{32}$$

For $KI_j (j \in \{1, ..., 8\})$, the filtration ratios are $\frac{10946}{2^{19}}, \frac{3}{2^2}, \frac{5}{2^3}, \frac{3}{2^2}, \frac{13}{2^5}, \frac{89}{2^9}, \frac{55}{2^8}, \frac{3}{2^2}$ respectively. Due to the disappearance of key in any U_i of **CASE 1**, it provides a filtration ratio equals $2^{-14.55}$ which is the product of the ratios above. The keys passed all the above testers will be checked by a $(nonce, plaintext, ciphertext, tag)$ pair to determine the right one.

The disappearance of key in any U_i and U_i' give a total filtration ratio of $2^{-10} \times 2^{-14.55}$, and the keys with key bits passed through the testers above are exactly keys in the *remaining key subset* whose size is $2^{128-10} \times 2^{-14.55} = 2^{103.45}$. So in worst case, omitting complexity of cube computation, the total complexity is $2^{103.45}$.