Resistance of Ascon Family against Conditional Cube Attacks in Nonce-Misuse Setting

Donghoon Chang[1,2,4], Deujo Hong[1,3], Jinkeon Kang[1], and Meltem Sönmez Turan[1]

[1]National Institute of Standards and Technology, Gaithersburg, Maryland, USA (e-mail: {donghoon.chang, deukjo.hong, jinkeon.kang, meltem.turan}@nist.gov)
[2]Strativia, Largo, Maryland, USA
[3]Jeonbuk National University, Jeonju-si, Korea (e-mail: deukjo.hong@jbnu.ac.kr)
[4]Department of Computer Science, Indraprastha Institute of Information Technology Delhi(IIIT-Delhi), Delhi, India (e-mail: donghoon@iiitd.ac.in)

Abstract. Ascon family is one of the finalists of the National Institute of Standards and Technology (NIST) lightweight cryptography standardization process. The family includes three Authenticated Encryption with Associated Data (AEAD) schemes: Ascon-128 (primary), Ascon-128a, and Ascon-80pq. In this paper, we study the resistance of the Ascon family against conditional cube attacks in nonce-misuse setting, and present new state– and key–recovery attacks. Our attack recovers the full state information and the secret key of Ascon-128a using 7-round Ascon-permutation for the encryption phase, with 2^{117} data and $2^{116.2}$ time. This is the best known attack result for Ascon-128a as far as we know. We also show that the partial state information of Ascon-128 can be recovered with $2^{44.8}$ data. Finally, by assuming that the full state information of Ascon-80pq was recovered by Baudrin et al.'s attack, we show that the 160-bit secret key of Ascon-80pq can be recovered with 2^{128} time. Although our attacks do not invalidate designers' claim, those allow us to understand the security of Ascon in nonce-misuse setting.

Keywords: Ascon, Conditional cube attack, Lightweight cryptography, State recovery, Key recovery

1 Introduction

Ascon, designed by Dobraunig et al. [5], is one of the finalists of the National Institute of Standards and Technology (NIST) lightweight cryptography standardization process. Ascon is also selected as the primary choice for lightweight authenticated encryption in the final portfolio of the CAESAR competition [3]. Ascon family includes three Authenticated Encryption with Associated Data (AEAD) schemes: Ascon-128 (primary submission for the NIST process), Ascon-128a, and Ascon-80pq. The mode of operation for these algorithms is based on duplex modes like MonkeyDuplex [2], and use key initialization and finalization functions. They use 12-round Ascon permutation for their initialization and finalization with key. For data processing such as associated data (AD) absorption and plaintext encryption, Ascon-128 and Ascon-80pq use 6-round Ascon permutation, while Ascon-128a use 8-round one. The state size of Ascon permutation is 320 bits, the key size of Ascon-128 and Ascon-128a is 128 bits, and the key size of Ascon-80pq is 160 bits.

Throughout this paper, for reduced-round versions of Ascon family, we denote Ascon-X for $X \in \{128, 128a, 80pq\}$ with r_1-round initialization, r_2-round data processing, and r_3-round finalization, by (r_1, r_2, r_3)-round Ascon-X. (r_1, r_2, r_3) is $(12, 6, 12)$ for Ascon-128 and Ascon-80pq, and $(12, 8, 12)$ for Ascon-128a. We also use the notation $r_i = \star$ for $i \in \{1, 2, 3\}$ when r_i can be any positive number.

Ascon designers claim that AEAD variants provide 128-bit security of privacy and authenticity when unique nonce values are used for the encryption under the same key [5]. The maximum available data to the attacker is limited to 2^{64} 64-bit blocks per key. In nonce-misuse scenario, the designers claimed that Ascon-128a provides 128-bit security of privacy and authenticity if nonces are reused a few times by accident as long as the combination of nonce and AD stays unique. The designers said that they do not expect that any key recovery attack on ASCON-128a can be found with complexity significantly below 2^{96} even after a secret state is recovered by an implementation attack, due to the extra key additions during initialization and finalization.

The security of ASCON family have been widely analyzed. In particular, most successful key recovery attacks for ASCON are based on cube attacks [4]. Li et al. [9] presented two nonce-respecting key recovery attacks. One of them is a cube attack on $(7,\star,\star)$–round ASCON, where the data complexity is $2^{77.2}$ and the time complexity is $2^{103.92}$. The other one is a conditional cube attack on $(6,\star,\star)$–round ASCON, where both data and time complexities are 2^{40}. Rohit et al. [10] presented the nonce-respect key recovery attack on $(7,\star,\star)$–round ASCON which is a cube attack with data complexity of 2^{64} and time complexity of 2^{123}. The security of ASCON in nonce-misuse setting was studied in [8].

Huang et al. [6] introduced the concept of *conditional* cube attacks. The authors proposed some of conditions on the key to obtain the set $\mathcal{V}$ of cube variables such that the variables in $\mathcal{V}$ are not multiplied with each other after the first round and $\mathcal{V}$ contains one cube variable that are not multiplied with other cube variables after the second round. With this technique, Huang et al. achieved the optimal cube propagation for KECCAK permutation. We observe that the security of ASCON AEAD algorithms against cube attacks has been analyzed a lot in nonce-respecting setting, but not so much in nonce-misuse setting. Baudrin et al. [1] suggested a conditional cube attack on full $(\star,6,\star)$-round ASCON-128 recovering the full state information with the data complexity less than 2^{40} in nonce-misuse setting. Considering these existing work results, we have been motivated to study how conditional cube attack techniques can be most effectively applied to ASCON AEAD algorithms in nonce-misuse setting.

1.1 Contributions

Our main idea is to recover the secret state information using cubes where certain conditions make the cube-sums zero, and then recover the secret key for the finalization permutation. We use five cube patterns to make new nonce-misuse state-recovery and key recovery attacks on $(\star,7,\star)$-round ASCON-128a. Those attacks require 2^{117} data and $2^{116.2}$ time, and are the best known attack results for ASCON-128a as far as we know. We also use a family of patterns to make a nonce-misuse partial-state-recovery conditional-cube attack on $(\star,6,\star)$-round ASCON-128 and ASCON-80pq, where 192 bits out of 320-bit state are recovered, with $2^{44.8}$ time and data complexity and negligible memory complexity. This attack was researched independently of those by Baudrin et al. [1]. Additionally, we show that the 160-bit secret key of ASCON-80pq can be recovered based on the recovered state information with 2^{128} time, much faster than an exhaustive key search.

Table 1 summarizes the existing cube attacks and our attacks on ASCON AEAD algorithms. In the table, an entry for 'Target' field can be 128, 128a and 80pq, which mean ASCON-128, ASCON-128a and ASCON-80pq as a target of the attack, respectively. An entry for 'Type' field can be KR, SR and F, which mean Key Recovery, State Recovery and Forgery as a type of the attack, respectively. An entry for 'Complexity' field is a 3-tuple of data, time, and memory. An entry of 'Set.' field can be NR and NM, which mean Nonce-Respecting and Nonce-Misuse as a setting of the attack, respectively.

Table 1: Summary of cube attacks on ASCON AEAD algorithms

Target	Type	Rounds	Complexity (D,T,M)	Set.	Ref.
128, 128a	KR	$(6,\star,\star)$	$(2^{40},2^{40},\text{-})$	NR	[9]
128, 128a	KR	$(7,\star,\star)$	$(2^{77.2},2^{103.92},\text{-})$	NR	[9]
128, 128a	KR	$(7,\star,\star)$	$(2^{64},2^{123},\text{-})$	NR	[10]
128, 128a	KR	$(7,5,\star)$	$(2^{33},2^{97},\text{-})$	NM	[8]
128a	KR	$(\star,7,\star)$	$(2^{117},2^{116.2},2^{32})$	NM	Sect. 4.2
80pq	KR	$(\star,6,\star)$	$(2^{39.6},2^{128},2^{32})$	NM	Sect. 5.2
All	SR	$(\star,5,\star)$	$(2^{18},2^{66},\text{-})$	NM	[8]
128a	SR	$(\star,7,\star)$	$(2^{117},2^{116.2},\text{-})$	NM	Sect. 4.1
128, 80pq	SR[a]	$(\star,6,\star)$	$(2^{44.8},\text{-},\text{-})$	NM	Sect. 5.1
128, 80pq	SR	$(\star,6,\star)$	$(2^{39.6},2^{39.6},\text{-})$	NM	[1]
All	F	$(\star,\star,5)$	$(2^{17},2^{17},\text{-})$	NM	[8]
All	F	$(\star,\star,6)$	$(2^{33},2^{33},\text{-})$	NM	[8]

[a]Partial 192-bit state-recovery

Our attacks do not invalidate the security claims of the ASCON designers. Nevertheless, they are meaningful in analyzing how secure ASCON is in the nonce-misuse setting.

1.2 Organization

In Section 2, we introduce notations, ASCON AEAD algorithms, and cube attacks. In Section 3, we describe cube patterns used in our attacks. In Section 4, we explain how the attacks on $(\star,7,\star)$-round ASCON-128a recover the internal state and the secret key. In Section 5, we explain the partial-state-recovery attack on $(\star,6,\star)$-round ASCON-128 and ASCON80pq and the key recovery attack on $(\star,6,\star)$-round ASCON80pq.

2 Preliminaries

2.1 Definitions and Notations

Let x and y be bitstrings of same length. We denote bitwise XOR, and bitwise AND of x and y by $x \oplus y$ and xy. We denote the concatenation of x and y by $x \| y$ or (x, y). $MSB_m(x)$ and $LSB_m(x)$ mean the most and the least significant m bits of x, respectively. $Len(x)$ means the length of x in bits. We denote a bitstring of n consecutive 0-bits by 0^n. 0^* means an arbitrary-length bitstring of consecutive 0-bits. Likewise, we denote a bitstring of n consecutive 1-bits by 1^n.

2.2 Ascon AEAD Algorithms

The design of Ascon AEAD algorithms is based on monkeyDuplex construction [2] with extra key additions during initialization and finalization. The 320-bit state $State$ is initialized as

$$State = IV \| K \| N,$$

where IV is a constant as an initial value, K is a k-bit secret key, and N is a 128-bit nonce. Let p^i be the i-round Ascon permutation. The algorithm works in the order initialization phase, data processing phase, and finalization phase. In the initial phase, the state $State$ is updated as

$$State \leftarrow p^a(State) \oplus (0^{320-k} \| K).$$

The data processing phase consists of AD absorption phase and plaintext encryption phase, but we assume AD is empty because it is not necessary for our attacks. The plaintext P is padded to $P \| 1 \| 0^*$ such that the length of the padded string is the least multiple of r, where r is the block size of plaintext and the rate of sponge-like construction. $c = 320 - r$ is the capacity. Then, the padded plaintext string is divided into t blocks P_1, ..., P_t. We consider $State$ as $State = State_r \| State_c$, where $State_r$ is r bits and $State_c$ is c bits. When the AD is empty, in the encryption phase, the update of $State$ and the encryption of the plaintext P proceed as follows:

$$\begin{aligned}
&State \leftarrow State \oplus (0^{319} \| 1)\\
&\textbf{for } i = 1, ..., t \textbf{ do}:\\
&\quad State_r \leftarrow State_r \oplus P_i\\
&\quad C_i \leftarrow State_r\\
&\quad State \leftarrow p^b(State)\\
&State_r \leftarrow State_r \oplus P_t\\
&C_t \leftarrow MSB_\ell(State_r)
\end{aligned}$$

, where C_i's are ciphertext blocks and $\ell = Len(P) \bmod r$. Finally, in the finalization phase, $State$ is updated and the tag T is produced as follows:

$$\begin{aligned}
&State \leftarrow p^a(State \oplus (0^r \| K \| 0^{320-r-k})) \text{ and}\\
&T \leftarrow LSB_{128}(State) \oplus LSB_{128}(K).
\end{aligned}$$

Therefore, an Ascon AEAD algorithm outputs the ciphertext $C_1 \| \cdots \| C_t$ and the tag T.

The nonces and tags are 128 bits for every Ascon AEAD algorithm. Table 2 summarizes the parameters a, b, r, c, and k. See [5] for a more detailed description of Ascon.

Table 2: Parameters of Ascon AEAD Algorithm

Algorithm	a	b	r	c	k
Ascon-128	12	6	64	256	128
Ascon-128a	12	8	128	192	128
Ascon-80pq	12	6	64	256	160

The j-th round function, Round$j$ of Ascon permutation is defined as $p_L \circ p_S \circ p_C$, where p_C adds the 64-bit constant c_j to the internal state, p_S is the substitution layer, and p_L is the linear diffusion layer. Note that the round number j starts from zero (i.e., $j = 0, 1, ...$). We denote the input state of Round$j$ by $S_j = S_j[0] \| \cdots \| S_j[4]$, where each $S_j[i]$ is 64 bits. We also regard S_j as a 5×64 array and $S_j[i]$ as the i-th row of S_j. The m-th bit of $S_j[i]$ is denoted by $S_j[i][m]$ for $0 \le m \le 63$. $S_j[i][0]$ and $S_j[i][63]$ are the least and the most significant bits of $S_j[i]$, respectively. The i-th column of S_j is defined by $(S_j[0][i], ..., S_j[4][i])^T$ and is called the column $\#i$ of S_j. We denote the internal state after the $p_S \circ p_C$ operation at Round$j$ by $S_{j+0.5} = S_{j+0.5}[0] || \cdots || S_{j+0.5}[4]$. Then, the internal state after the p_L layer is denoted by S_{j+1}, which is the output of Round$j$.

2.2.1 Substitution Layer p_S

The nonlinearity of Ascon permutation is provided by the p_S layer. Let x_i for $0 \le i \le 4$ be the i-th row in a 5×64 array of 320-bit state. $p_S(x_0, ..., x_4)$ is computed as follows:

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

We can also regard (1) as a system of equations for a 5-bit input $x = (x_4, ..., x_0) \in \{0, 1\}^5$, and p_S

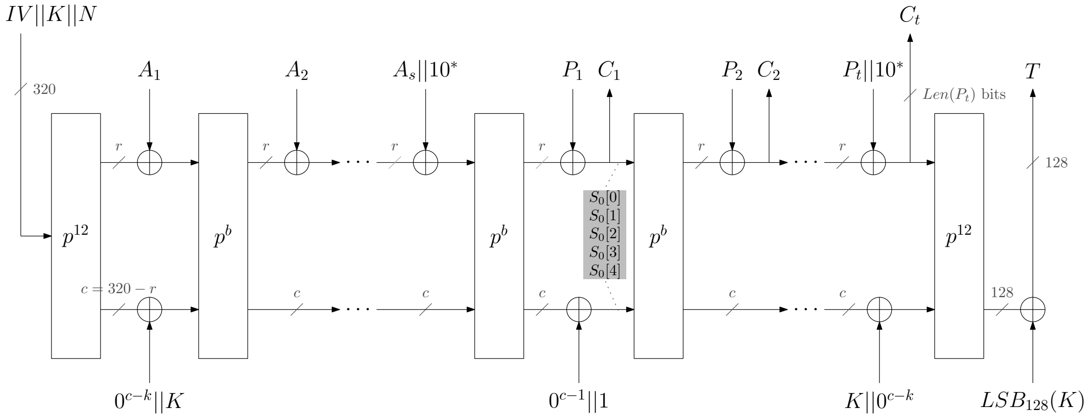

Figure 1: ASCON AEAD mode. (b, r, k) is $(6, 64, 128)$ for ASCON-128, $(6, 64, 160)$ for ASCON-80pq, and $(8, 128, 128)$ for ASCON-128a.

as the application of 64 5-bit nonlinear S-boxes to the columns of 5×64 array. We have the following properties derived from (1): for $x \in \{0,1\}^5$,

$$y_1 = \begin{cases} x_4 \oplus x_3x_2 \oplus x_1 \oplus x_0 & \text{if } x_2 = x_3 \\ x_4 \oplus x_3x_2 \oplus x_0 \oplus 1 & \text{otherwise,} \end{cases} \quad (2)$$

$$y_3 = \begin{cases} x_2 \oplus x_1 \oplus x_0 & \text{if } x_3 = x_4 \\ x_2 \oplus x_1 \oplus 1 & \text{otherwise.} \end{cases} \quad (3)$$

2.2.2 Linear Diffusion Layer p_L with 64-bit Diffusion Functions $\Sigma_i(x_i)$

The p_L layer consists of 64-bit linear functions $\Sigma_i(x_i)$ for $i = 0, 1, ..., 4$, where each x_i is the i-th row in a 5×64 array of 320-bit state.

$$\begin{aligned} \Sigma_0(x_0) &= x_0 \oplus (x_0 \ggg 19) \oplus (x_0 \ggg 28), \\ \Sigma_1(x_1) &= x_1 \oplus (x_1 \ggg 61) \oplus (x_1 \ggg 39), \\ \Sigma_2(x_2) &= x_2 \oplus (x_2 \ggg 1) \oplus (x_2 \ggg 6), \\ \Sigma_3(x_3) &= x_3 \oplus (x_3 \ggg 10) \oplus (x_3 \ggg 17), \\ \Sigma_4(x_4) &= x_4 \oplus (x_4 \ggg 7) \oplus (x_4 \ggg 41). \end{aligned} \quad (4)$$

2.3 Time complexity of exhaustive key search

We should compare the time complexity of our attack to that of an exhaustive key search. If we assume that AD A is empty and the length of the plaintext P is less than r, the number of permutation calls per one encryption are minimized to two calls of p^{12}, which approximates p^{24}. So, we can compare the time complexity of our attack to 2^{128} operations of p^{24} for ASCON-128 and ASCON-128a, and 2^{160} operations of p^{24} for ASCON-80pq.

2.4 Cube Attacks

2.4.1 Cube and Cube-Sum

Let $v = (v_{l-1}, ..., v_1, v_0)$ be a l-bit string. We consider a l-bit variable $v = (v_{l-1}, ..., v_1, v_0)$. The set $C_v = \{0,1\}^l$ of all possible l-bit vectors for v is called the *cube* for l cube variables $v_0, v_1, ..., v_{l-1}$. Let $f(v, x)$ be a nonlinear Boolean function from $\{0,1\}^{l+m}$ to $\{0,1\}$ where x is a m-bit variable. We consider the division expression where f is the dividend, the monomial $v_0v_1 \cdots v_{l-1}$ is the divisor, Q is the quotient and R is the remainder. When Q only depends on x, the expression is as follows:

$$f(v, x) = v_0v_1 \cdots v_{l-1} \cdot Q(x) \oplus R(v, x). \quad (5)$$

For (5), Xuejia Lai [7] proved the following relation between the cube C_v and the quotient $Q(x)$.

$$\bigoplus_{v \in C_v} f = Q(x). \quad (6)$$

The left side of (6) is called the *cube-sum* of f for the cube $C_{\mathcal{V}}$. The relation shows that the cube-sum is equal to the quotient $Q(x)$. When x is fixed, $Q(x)$ is constant, so the cube-sum is constant.

Let $g(v, x)$ be another nonlinear Boolean function from $\{0,1\}^{l+m}$ to $\{0,1\}$. Suppose that we obtain the division expression of g with the divisor $v_0v_1 \cdots v_{l-2}$:

$$g(v, x) = v_0 \cdots v_{l-2} \cdot Q'(x) \oplus R'(v, x), \quad (7)$$

where $Q'(x)$ is the quotient and $R'(v, x)$ is the remainder. Let $C_v^{(0)} = \{(v_{l-1}, ..., v_0) \in \{0,1\}^l \mid v_{l-1} = 0\}$ and $C_v^{(1)} = \{(v_{l-1}, ..., v_0) \in \{0,1\}^l \mid v_{l-1} = 1\}$. When x is fixed, the cube-sum of g for C_v is zero because

$$\bigoplus_{(v_{l-1},...,v_0)\in C_v} g = \bigoplus_{(v_{l-1},...,v_0)\in C_v^{(0)}} g \oplus \bigoplus_{(v_{l-1},...,v_0)\in C_v^{(1)}} g$$
$$= Q(x) \oplus Q(x)$$
$$= 0.$$

Let $F = (f_{n-1}, ..., f_1, f_0)$ be a vectorial Boolean function from $\{0,1\}^{l+m}$ to $\{0,1\}^m$ where each $f_i(v,x)$ is a Boolean function from $\{0,1\}^{l+m}$ to $\{0,1\}$. When we say that the cube-sum of F for C_v is zero, we mean that for every $i = 0, ..., n-1$, the cube-sum of f_i for C_v is zero. In other words, we can say that the cube-sum on $y = F(v,x)$ is zero.

2.4.2 Conditional Cube Attacks

We briefly introduce the concept of conditional cube attacks which was proposed by Huang et al. [6]. Let F be a nonlinear permutation which iterates a round function with algebraic degree of 2. Note that the degree of the round function of Ascon permutation is also 2. For the set $\mathcal{V} = \{v_0, v_1, ..., v_{l-1}\}$ of cube variables, they considered a partition $\{\mathcal{V}_0, \mathcal{V}_1\}$ of $\mathcal{V}$ such that $\mathcal{V}_0 \cap \mathcal{V}_1 = \varnothing$ and $\mathcal{V} = \mathcal{V}_0 \cup \mathcal{V}_1$. We assume that that the following requirements hold if and only if certain conditions are true.

- *Requirement 1*: After the first round, there is no multiplication of two different cube variables from $\mathcal{V}$. For the case of Ascon permutation, the output state S_1 of Round0 has no v_iv_j such that $i \neq j$ and $v_i, v_j \in \mathcal{V}$.
- *Requirement 2*: After the second round, there is no multiplication of two different cube variables from $\mathcal{V}_0$. For the case of Ascon permutation, the output state S_2 of Round1 has no v_iv_j such that $i \neq j$ and $v_i, v_j \in \mathcal{V}_0$.
- *Requirement 3*: After the second round, there is no multiplication between a cube variable from $\mathcal{V}_0$ and a cube variable from $\mathcal{V}_1$. For the case of Ascon permutation, the output state S_2 of Round1 has no v_iv_j such that $v_i \in \mathcal{V}_0$ and $v_j \in \mathcal{V}_1$.

Theorem 1. [6] Let F be a nonlinear permutation which iterates a round function with algebraic degree of 2. Let $\mathcal{V} = \{v_0, v_1, ..., v_{\lambda+\mu-1}\}$ be the set of cube variables on the input state of F, and let $\{\mathcal{V}_0, \mathcal{V}_1\}$ be a partition of $\mathcal{V}$ where $|\mathcal{V}_0| = \lambda$ and $|\mathcal{V}_1| = \mu$. Assume that Requirements 1, 2 and 3 hold, and that for a positive number n,

$$\lambda, \mu \geq 1 \text{ and } \mu = 2^{n+1} - 2\lambda + 1, \text{ or} \quad (8)$$
$$\mu = 0 \text{ and } \lambda = 2^n + 1. \quad (9)$$

Then, the term $v_0v_1 \cdots v_{\lambda+\mu-1}$ does not appear on the output state of $(n+2)$-th round function.

Section 2.4.1 and Theorem 1 implies, if the nonlinear permutation F satisfies Requirements 1, 2, and 3 and satisfies (8) or (9), then F does not have the term $v_0v_1 \cdots v_{\lambda+\mu-1}$ on its output state of $(n+2)$-th round function, and so the cube-sum for $(n+2)$ rounds of F is zero. In the next section, for Ascon permutation, we make a set of cube variables satisfying (8), and construct the conditions under which Requirements 1, 2, and 3 hold.

3 Variables and Conditions of Cube Patterns

Let S_0 be the input state to the Ascon permutation of the first block in the encryption phase, as depicted in Figure 1. We describe five cube patterns used in our conditional cube attack on Ascon-128a in Sections 3.1 to 3.5. We use the set $\mathcal{V}$ of 64 cube variables and consider the partition $\{\mathcal{V}_0, \mathcal{V}_1\}$ of $\mathcal{V}$ such that $\lambda = |\mathcal{V}_0| = 1$ and $\mu = |\mathcal{V}_1| = 63$. Following the notations in Theorem 1, we have $n = 5$. We can choose the first block P_1 of plaintext to control the first two rows of S_0 for Ascon-128a. In each pattern, cube variables are assigned to the first two rows of S_0 so that each column of S_0 has at most one cube variable. We consider that $(S_0[2], S_0[3], S_0[4])$ is secret in Ascon-128a, and use a guessed value to construct cube patterns in Sections 3.2 to 3.5.

We also show a family of cube patterns used in our conditional cube attacks on Ascon-128 and Ascon-80pq in Section 3.6. We can choose the first block P_1 of plaintext to control the first row of S_0 for Ascon-128 and Ascon-80pq. We use 40 cube variables. In each pattern, 32 among them are assigned to the first row of S_0 such that $\lambda = |\mathcal{V}_0| = 1$ and $\mu = |\mathcal{V}_1| = 31$. Following the notations in Theorem 1, we have $n = 4$.

With this configuration, each pattern satisfies Requirements 1 and 2 in Section 2.4.2. Requirement 1 holds because in the beginning of Round0, each column uses at most one cube variable and the nonlinear S-box operation is applied column-wise in parallel. Requirement 2 holds because $|\mathcal{V}_0| = 1$. In the following subsections, we describe the structure of each pattern, and show what conditions satisfy Requirement 3. For simplicity, the cube variables in $\mathcal{V}_0$ is called $\mathcal{V}_0$-*variables* and the cube variables in $\mathcal{V}_1$ is called $\mathcal{V}_1$-*variables*.

3.1 Pattern-A

Pattern-A has 64 cube variables v_i for $0 \leq i \leq 63$. v_{63} is the only $\mathcal{V}_0$-variable and assigned as $S_0[0][63] = S_0[1][63] = v_{63}$. $\mathcal{V}_1$-variables are assigned to S_0 as listed in Table 3. The other bits of S_0 are constants. Table 4 lists 38 conditions of Pattern-A for satisfying Requirement 3. The 'Conditions' field of Table 4 shows condition expressions and the 'Count' field provides the number of conditions for each expression. We get Lemma 1.

Table 3: Assignment of the $\mathcal{V}_1$-variables for Pattern-A

Setting	Setting
$S_0[0][62] = S_0[1][62] = v_{62}$	$S_0[0][30] = v_{30}$
$S_0[0][61] = S_0[1][61] = v_{61}$	$S_0[0][29] = v_{29}$
$S_0[0][60] = S_0[1][60] = v_{60}$	$S_0[0][28] = v_{28}$
$S_0[0][59] = S_0[1][59] = v_{59}$	$S_0[0][27] = v_{27}$
$S_0[0][58] = v_{58}$	$S_0[0][26] = v_{26}$
$S_0[0][57] = S_0[1][57] = v_{57}$	$S_0[0][25] = v_{25}$
$S_0[0][56] = v_{56}$	$S_0[0][24] = v_{24}$
$S_0[0][55] = v_{55}$	$S_0[0][23] = v_{23}$
$S_0[0][54] = S_0[1][54] = v_{54}$	$S_0[0][22] = v_{22}$
$S_0[0][53] = v_{53}$	$S_0[0][21] = S_0[1][21] = v_{21}$
$S_0[0][52] = S_0[1][52] = v_{52}$	$S_0[0][20] = v_{20}$
$S_0[0][51] = S_0[1][51] = v_{51}$	$S_0[0][19] = S_0[1][19] = v_{19}$
$S_0[0][50] = v_{50}$	$S_0[0][18] = v_{18}$
$S_0[0][49] = v_{49}$	$S_0[0][17] = v_{17}$
$S_0[0][48] = v_{48}$	$S_0[0][16] = S_0[1][16] = v_{16}$
$S_0[0][47] = v_{47}$	$S_0[0][15] = S_0[1][15] = v_{15}$
$S_0[0][46] = v_{46}$	$S_0[0][14] = v_{14}$
$S_0[0][45] = S_0[1][45] = v_{45}$	$S_0[0][13] = v_{13}$
$S_0[0][44] = S_0[1][44] = v_{44}$	$S_0[0][12] = S_0[1][12] = v_{12}$
$S_0[0][43] = v_{43}$	$S_0[0][11] = v_{11}$
$S_0[0][42] = S_0[1][42] = v_{42}$	$S_0[0][10] = S_0[1][10] = v_{10}$
$S_0[0][41] = S_0[1][41] = v_{41}$	$S_0[0][9] = S_0[1][9] = v_9$
$S_0[0][40] = S_0[1][40] = v_{40}$	$S_0[0][8] = S_0[1][8] = v_8$
$S_0[0][39] = v_{39}$	$S_0[0][7] = v_7$
$S_0[0][38] = S_0[1][38] = v_{38}$	$S_0[0][6] = S_0[1][6] = v_6$
$S_0[0][37] = S_0[1][37] = v_{37}$	$S_0[0][5] = v_5$
$S_0[0][36] = v_{36}$	$S_0[0][4] = v_4$
$S_0[0][35] = S_0[1][35] = v_{35}$	$S_0[0][3] = S_0[1][3] = v_3$
$S_0[0][34] = v_{34}$	$S_0[0][2] = v_2$
$S_0[0][33] = v_{33}$	$S_0[0][1] = v_1$
$S_0[0][32] = S_0[1][32] = v_{32}$	$S_0[0][0] = v_0$
$S_0[0][31] = v_{31}$	

Lemma 1. Under the setting of Pattern-A, the state S_2 does not have any $v_{63}v_i$ for $v_i \in \mathcal{V}_1$ if and only if all conditions in Table 4 are true.

Proof. We assume that all conditions in Table 4 are true. Then, by the setting of $S_0[0][63] = S_0[1][63] = v_{63}$ and (1), $S_{0.5}[0][63]$ and $S_{0.5}[2][63]$ contain v_{63} only as a linear term and the other bits in the column #63 of $S_{0.5}$ are constants. After the p_L layer, the only state bits $S_1[0][63]$, $S_1[0][44]$, $S_1[0][35]$,

Table 4: 38 conditions of Pattern-A to satisfy Requirement 3

Conditions	Count
$S_0[2][63] = S_0[3][63] = S_0[4][63] = 0$	3
$S_0[2][62] = S_0[3][62] = S_0[4][62]$	2
$S_0[3][61] = S_0[4][61]$	1
$S_0[2][60] = S_0[3][60]$	1
$S_0[2][59] = S_0[3][59]$	1
$S_0[2][57] = S_0[3][57] = S_0[4][57]$	2
$S_0[2][54] = S_0[3][54] = S_0[4][54]$	2
$S_0[3][52] = S_0[4][52]$	1
$S_0[4][51] = 0$	1
$S_0[3][45] = S_0[4][45]$	1
$S_0[2][44] = S_0[3][44] = S_0[4][44] = 0$	3
$S_0[4][42] = 0$	1
$S_0[2][41] = S_0[3][41]$	1
$S_0[4][40] = 0$	1
$S_0[2][38] = S_0[3][38]$	1
$S_0[2][37] = S_0[3][37]$	1
$S_0[2][35] = S_0[3][35] = S_0[4][35] = 0$	3
$S_0[2][32] = S_0[3][32]$	1
$S_0[4][21] = 0$	1
$S_0[2][19] = S_0[3][19]$	1
$S_0[3][16] = S_0[4][16]$	1
$S_0[3][15] = S_0[4][15]$	1
$S_0[4][12] = 0$	1
$S_0[2][10] = S_0[3][10] = S_0[4][10]$	2
$S_0[3][9] = S_0[4][9]$	1
$S_0[3][8] = S_0[4][8]$	1
$S_0[4][6] = 0$	1
$S_0[3][3] = S_0[4][3]$	1

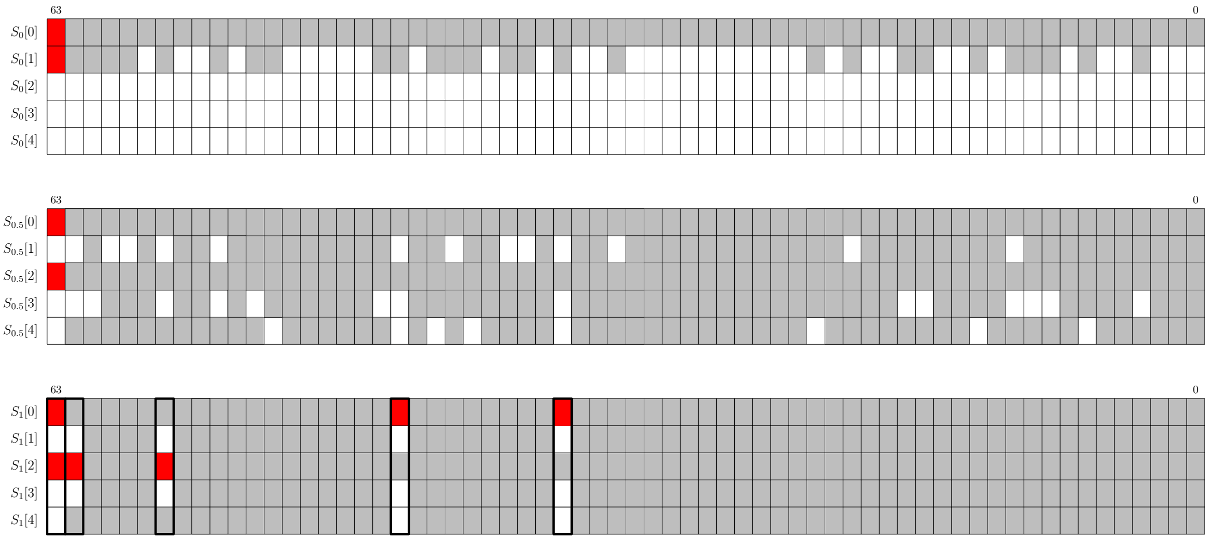

Figure 2: Propagation of the $\mathcal{V}_0$-variable v_{63} an $\mathcal{V}_1$-variables in Pattern-A. The bits influenced by v_{63} are marked in red; the bits influenced by $\mathcal{V}_1$-variables and not by v_{63} are marked in gray; The bits marked in white are constants.

$S_1[2][63]$, $S_1[2][62]$, and $S_1[2][57]$ contain v_{63} as a linear term, too. The state bits $S_1[0][62]$, $S_1[0][57]$, $S_1[2][44]$, $S_1[2][35]$, $S_1[4][62]$, $S_1[4][57]$, $S_1[4][44]$, and $S_1[4][35]$ are not influenced by v_{63}. The other bits in columns #63, #62, #57, #44, and #35 of S_1 are constants. Therefore, by (1), no multiplication between v_{63} and v_i for any $v_i \in \mathcal{V}_1$ appears on $S_{1.5}$ and S_2. See Figure 2.

Next, we consider the case that not all conditions in Table 4 are true. It can be split into two subcases as follows:

- **Subcase 1:** If any of the conditions on column #63 of S_0 is false, then $v_{63}v_i$ for some $v_i \in \mathcal{V}_1$ appears on S_2. The proof is provided in Appendix A.1.

- **Subcase 2:** If all conditions on column #63 of S_0 are true, but any of the other conditions is false, then $v_{63}v_i$ for some $v_i \in \mathcal{V}_1$ appears on S_2. The proof is provided in Appendix A.2.

Therefore, the proof is completed. □

3.2 Pattern-B

We assume that $\alpha = (\alpha_6, ..., \alpha_0) \in \{0,1\}^7$ is a guessed value for $(S_0[2][58] \oplus S_0[3][58], S_0[2][53] \oplus S_0[3][53], S_0[2][36] \oplus S_0[3][36], S_0[2][31] \oplus S_0[3][31], S_0[3][14] \oplus S_0[4][14], S_0[3][7] \oplus S_0[4][7], S_0[3][2] \oplus S_0[4][2])$. Pattern-B has 64 cube variables v_i for $0 \le i \le 63$. v_{62} is the only $\mathcal{V}_0$-variable and is assigned as $S_0[0][62] = v_{62}$ and $S_0[1][62] = v_{62} \oplus 1$. $\mathcal{V}_1$-variables are assigned to S_0 as listed in Table 5. Additionally, we assign more $\mathcal{V}_1$-variables depending on α as follows.

$$\begin{aligned}
S_0[0][58] &= v_{58} && \text{if } \alpha_6 = 0;\\
S_0[0][53] &= v_{53} && \text{if } \alpha_5 = 0;\\
S_0[0][36] &= v_{36} && \text{if } \alpha_4 = 0;\\
S_0[0][31] &= v_{31} && \text{if } \alpha_3 = 0;\\
S_0[1][14] &= v_{14} && \text{if } \alpha_2 = 0;\\
S_0[1][7] &= v_{7} && \text{if } \alpha_1 = 0;\\
S_0[1][2] &= v_{2} && \text{if } \alpha_0 = 0.
\end{aligned} \tag{10}$$

The other bits of S_0 are constants. 10 implies that Pattern-B contains 128 different assignments of cube variables depending on $\alpha \in \{0,1\}^7$. Table 6 lists 12 conditions of Pattern-B with α for satisfying Requirement 3. Then, we get Lemma 2.

Lemma 2. Under the setting of Pattern-B with a guessed value $\alpha \in \{0,1\}^7$, the state S_2 does not have any quadratic term $v_{62}v_i$ for $v_i \in \mathcal{V}_1$ if and only if α is correct and all conditions in Table 6 are true.

Proof. We concentrate on showing that a quadratic term $v_{62}v_i$ for $v_i \in \mathcal{V}_1$ appears on $S_{1.5}$ if all the conditions on column #62 of S_0 are true and α is wrong. When all the conditions on column #62 of S_0 are true, by (1), $S_{0.5}[2][62]$ contains v_{62} only as a linear term and the other bits of the column #62 of $S_{0.5}$ are constant. After the p_L layer, $S_1[2][62]$, $S_1[2][61]$ and $S_1[2][56]$ contain v_{62} as a linear term.

Firstly, we consider the case of $\alpha_0 \neq S_0[3][2] \oplus S_0[4][2]$. If $\alpha_0 = 0$ and $S_0[3][2] \neq S_0[4][2]$, then the equation for $S_{0.5}[3][2]$ is $y_3 = x_2 \oplus x_1 \oplus 1$ by

Table 5: Assignment of $\mathcal{V}_1$-variables for Pattern-B

Setting	Setting
$S_0[1][63] = v_{63}$	$S_0[0][30] = v_{30}$
$S_0[0][61] = S_0[1][61] = v_{61}$	$S_0[1][29] = v_{29}$
$S_0[0][60] = v_{60}$	$S_0[0][28] = v_{28}$
$S_0[0][59] = S_0[1][59] = v_{59}$	$S_0[0][27] = v_{27}$
$S_0[1][58] = v_{58}$	$S_0[0][26] = v_{26}$
$S_0[0][57] = v_{57}$	$S_0[0][25] = v_{25}$
$S_0[0][56] = S_0[1][56] = v_{56}$	$S_0[1][24] = v_{24}$
$S_0[0][55] = v_{55}$	$S_0[1][23] = v_{23}$
$S_0[0][54] = v_{54}$	$S_0[0][22] = v_{22}$
$S_0[1][53] = v_{53}$	$S_0[0][21] = v_{21}$
$S_0[1][52] = v_{52}$	$S_0[0][20] = v_{20}$
$S_0[1][51] = v_{51}$	$S_0[0][19] = v_{19}$
$S_0[0][50] = v_{50}$	$S_0[0][18] = v_{18}$
$S_0[0][49] = v_{49}$	$S_0[0][17] = v_{17}$
$S_0[0][48] = v_{48}$	$S_0[0][16] = v_{16}$
$S_0[0][47] = v_{47}$	$S_0[0][15] = S_0[1][15] = v_{15}$
$S_0[1][46] = v_{46}$	$S_0[0][14] = v_{14}$
$S_0[1][45] = v_{45}$	$S_0[0][13] = v_{13}$
$S_0[0][44] = v_{44}$	$S_0[0][12] = v_{12}$
$S_0[0][43] = v_{43}$	$S_0[0][11] = v_{11}$
$S_0[0][42] = v_{42}$	$S_0[0][10] = v_{10}$
$S_0[0][41] = v_{41}$	$S_0[0][9] = S_0[1][9] = v_9$
$S_0[0][40] = v_{40}$	$S_0[0][8] = S_0[1][8] = v_8$
$S_0[0][39] = v_{39}$	$S_0[0][7] = v_7$
$S_0[0][38] = v_{38}$	$S_0[0][6] = v_6$
$S_0[0][37] = S_0[1][37] = v_{37}$	$S_0[0][5] = v_5$
$S_0[1][36] = v_{36}$	$S_0[1][4] = v_4$
$S_0[0][35] = v_{35}$	$S_0[0][3] = v_3$
$S_0[0][34] = v_{34}$	$S_0[0][2] = v_2$
$S_0[0][33] = v_{33}$	$S_0[1][1] = v_1$
$S_0[0][32] = v_{32}$	$S_0[0][0] = v_0$
$S_0[1][31] = v_{31}$	

Table 6: 12 Conditions of Pattern-B to satisfy Requirement 3

Conditions	Count
$S_0[2][62] = S_0[3][62] = S_0[4][62] = 1$	3
$S_0[2][61] = S_0[3][61] = S_0[4][61]$	2
$S_0[2][59] = S_0[3][59]$	1
$S_0[2][56] = S_0[3][56] = S_0[4][56]$	2
$S_0[2][37] = S_0[3][37]$	1
$S_0[3][15] = S_0[4][15]$	1
$S_0[3][9] = S_0[4][9]$	1
$S_0[3][8] = S_0[4][8]$	1

(3) but v_2 is assigned to $S_0[1][2]$. If $\alpha_0 = 1$ and $S_0[3][2] = S_0[4][2]$, then the equation for $S_{0.5}[3][2]$ is $y_3 = x_2 \oplus x_1 \oplus x_0$ by (3) but v_2 is not assigned to $S_0[1][2]$. It implies that $S_{0.5}[3][2]$ contains v_2 as a linear term. After p_L layer, $S_1[3][56]$ contains v_2 as a linear term. Thus, the quadratic term $v_{62}v_2$ appears on the column #56 of $S_{1.5}$. We can similarly show that $v_{62}v_7$ appears on the column #61 of $S_{1.5}$ when $\alpha_1 \neq S_0[3][7] \oplus S_0[4][7]$, and that $v_{62}v_{14}$ appears on the column #61 of $S_{1.5}$ when $\alpha_2 \neq S_0[3][14] \oplus S_0[4][14]$.

Secondly, we consider the case of $\alpha_3 \neq S_0[2][31] \oplus S_0[3][31]$. If $\alpha_3 = 0$ and $S_0[2][31] \neq S_0[3][31]$, then the equation for $S_{0.5}[1][31]$ is $x_4 \oplus x_3x_2 \oplus x_0 \oplus 1$ by (2) but v_{31} is assigned to $S_0[0][31]$. If $\alpha_3 = 1$ and $S_0[2][31] = S_0[3][31]$, then the equation for $S_{0.5}[1][31]$ is $x_4 \oplus x_3x_2 \oplus x_1 \oplus x_0$ by (2) but v_{31} is not assigned to $S_0[0][31]$. It implies that $S_{0.5}[1][31]$ contains v_{31} as a linear term. After p_L layer, $S_1[1][56]$ contains v_{31} as a linear term. Thus, the quadratic term $v_{62}v_{31}$ appears on the column #56 of $S_{1.5}$. We can similarly show that $v_{62}v_{36}$ appears on the column #61 of $S_{1.5}$ when $\alpha_4 \neq S_0[2][36] \oplus S_0[3][36]$, that $v_{62}v_{53}$ appears on the column #56 of $S_{1.5}$ when $\alpha_5 \neq S_0[2][53] \oplus S_0[3][53]$, and that $v_{62}v_{58}$ appears on the column #61 of $S_{1.5}$ when $\alpha_6 \neq S_0[2][58] \oplus S_0[3][58]$.

The remaining of the proof is similar to that of Lemma 1. □

3.3 Pattern-C

We assume that $\beta = (\beta_5, ..., \beta_0) \in \{0,1\}^6$ is a guessed value for $(S_0[2][53] \oplus S_0[3][53], S_0[2][48] \oplus S_0[3][48], S_0[2][36] \oplus S_0[3][36], S_0[2][31] \oplus S_0[3][31], S_0[2][26] \oplus S_0[3][26], S_0[3][4] \oplus S_0[4][4], S_0[3][2] \oplus S_0[4][2])$. Pattern-C has 64 cube variables v_i for $0 \leq i \leq 63$. v_{57} is the only $\mathcal{V}_0$-variable and assigned as $S_0[0][57] = v_{57}$ and $S_0[1][57] = v_{57} \oplus 1$. v_i's for $i \neq 57$ are assigned to S_0 as listed Table 7. Additionally, we assign more $\mathcal{V}_1$-variables depending on β as follows.

$$\begin{array}{ll} S_0[0][53] = v_{53} & \text{if } \beta_5 = 0; \\ S_0[0][48] = v_{48} & \text{if } \beta_4 = 0; \\ S_0[0][31] = v_{31} & \text{if } \beta_3 = 0; \\ S_0[0][26] = v_{26} & \text{if } \beta_2 = 0; \\ S_0[1][4] = v_4 & \text{if } \beta_1 = 0; \\ S_0[1][2] = v_2 & \text{if } \beta_0 = 0. \end{array} \tag{11}$$

The other bits of S_0 are constants. (11) implies that Pattern-C contains 64 different assignments of cube variables depending on $\beta \in \{0,1\}^6$. Table 8 lists 13 conditions of Pattern-C with β for satisfying Requirement 3. Then, we get Lemma 3.

Table 7: Assignment of $\mathcal{V}_1$-variables for Pattern-C

Setting	Setting
$S_0[1][63] = v_{63}$	$S_0[0][30] = v_{30}$
$S_0[0][62] = v_{62}$	$S_0[0][29] = v_{29}$
$S_0[0][61] = S_0[1][61] = v_{61}$	$S_0[0][28] = v_{28}$
$S_0[1][60] = v_{60}$	$S_0[0][27] = v_{27}$
$S_0[0][59] = v_{59}$	$S_0[1][26] = v_{26}$
$S_0[1][58] = v_{58}$	$S_0[0][25] = v_{25}$
$S_0[0][56] = S_0[1][56] = v_{56}$	$S_0[1][24] = v_{24}$
$S_0[0][55] = v_{55}$	$S_0[0][23] = v_{23}$
$S_0[0][54] = S_0[1][54] = v_{54}$	$S_0[0][22] = v_{22}$
$S_0[1][53] = v_{53}$	$S_0[0][21] = v_{21}$
$S_0[0][52] = v_{52}$	$S_0[0][20] = v_{20}$
$S_0[0][51] = S_0[1][51] = v_{51}$	$S_0[1][19] = v_{19}$
$S_0[0][50] = v_{50}$	$S_0[1][18] = v_{18}$
$S_0[0][49] = v_{49}$	$S_0[0][17] = v_{17}$
$S_0[1][48] = v_{48}$	$S_0[0][16] = v_{16}$
$S_0[1][47] = v_{47}$	$S_0[0][15] = v_{15}$
$S_0[1][46] = v_{46}$	$S_0[0][14] = v_{14}$
$S_0[0][45] = v_{45}$	$S_0[0][13] = v_{13}$
$S_0[0][44] = v_{44}$	$S_0[0][12] = v_{12}$
$S_0[0][43] = v_{43}$	$S_0[0][11] = v_{11}$
$S_0[0][42] = v_{42}$	$S_0[0][10] = S_0[1][10] = v_{10}$
$S_0[1][41] = v_{41}$	$S_0[0][9] = S_0[1][9] = v_9$
$S_0[1][40] = v_{40}$	$S_0[0][8] = v_8$
$S_0[0][39] = v_{39}$	$S_0[0][7] = v_7$
$S_0[0][38] = v_{38}$	$S_0[0][6] = v_6$
$S_0[0][37] = v_{37}$	$S_0[0][5] = v_5$
$S_0[0][36] = v_{36}$	$S_0[0][4] = v_4$
$S_0[0][35] = v_{35}$	$S_0[0][3] = S_0[1][3] = v_3$
$S_0[0][34] = v_{34}$	$S_0[0][2] = v_2$
$S_0[0][33] = v_{33}$	$S_0[0][1] = v_1$
$S_0[0][32] = S_0[1][32] = v_{32}$	$S_0[0][0] = v_0$
$S_0[1][31] = v_{31}$	

Table 8: 13 Conditions of Pattern-C to satisfy Requirement 3

Conditions	Count
$S_0[3][61] = S_0[4][61]$	1
$S_0[2][57] = S_0[3][57] = S_0[4][57] = 1$	3
$S_0[2][56] = S_0[3][56] = S_0[4][56]$	2
$S_0[2][54] = S_0[3][54]$	1
$S_0[2][51] = S_0[3][51] = S_0[4][51]$	2
$S_0[2][32] = S_0[3][32]$	1
$S_0[3][10] = S_0[4][10]$	1
$S_0[3][9] = S_0[4][9]$	1
$S_0[3][3] = S_0[4][3]$	1

Lemma 3. Under the setting of Pattern-C with a guessed value $\beta \in \{0,1\}^6$, the state S_2 does not have any quadratic term $v_{57}v_i$ for $v_i \in \mathcal{V}_1$ if and only if β is correct and all conditions in Table 8 are true.

Proof. We concentrate on showing that a quadratic term $v_{57}v_i$ for $v_i \in \mathcal{V}_1$ appears on $S_{1.5}$ if all the conditions on column #57 of S_0 are true and β is wrong. When all the conditions on column #57 of S_0 are true, by (1), $S_{0.5}[2][57]$ contains v_{57} only as a linear term and the other bits of the column #57 of $S_{0.5}$ are constant. After the p_L layer, $S_1[2][57]$, $S_1[2][56]$ and $S_1[2][51]$ contain v_{57} as a linear term.

Similarly to the proof of Lemma 2, we can show that $v_{57}v_2$ appears on the column #56 of $S_{1.5}$ when $\beta_0 \neq S_0[3][2] \oplus S_0[4][2]$, that $v_{57}v_4$ appears on the column #51 of $S_{1.5}$ when $\beta_1 \neq S_0[3][4] \oplus S_0[4][4]$, that $v_{57}v_{26}$ appears on the column #51 of $S_{1.5}$ when $\beta_2 \neq S_0[2][26] \oplus S_0[3][26]$, that $v_{57}v_{31}$ appears on the column #56 of $S_{1.5}$ when $\beta_3 \neq S_0[2][31] \oplus S_0[3][31]$, that $v_{57}v_{48}$ appears on the column #51 of $S_{1.5}$ when $\beta_4 \neq S_0[2][48] \oplus S_0[3][48]$, and that $v_{57}v_{53}$ appears on the column #56 of $S_{1.5}$ when $\beta_5 \neq S_0[2][53] \oplus S_0[3][53]$.

The remaining of the proof is similar to that of Lemma 1. □

3.4 Pattern-D

We assume that $\gamma = (\gamma_4, ..., \gamma_0) \in \{0,1\}^5$ is a guessed value for $(S_0[2][47] \oplus S_0[3][47], S_0[2][46] \oplus S_0[3][46], S_0[2][25] \oplus S_0[3][25], S_0[2][24] \oplus S_0[3][24], S_0[3][2] \oplus S_0[4][2])$. Pattern-D has 64 cube variables v_i for $0 \leq i \leq 63$. v_{50} is the only $\mathcal{V}_0$-variable and assigned as $S_0[0][50] = v_{50}$ and $S_0[1][50] = v_{50} \oplus 1$. $\mathcal{V}_1$-variables are assigned to S_0 as listed in Table 9. Additionally, we assign more $\mathcal{V}_1$-variables depending on γ as follows.

$$\begin{array}{ll} S_0[0][47] = v_{47} & \text{if } \gamma_4 = 0; \\ S_0[0][46] = v_{46} & \text{if } \gamma_3 = 0; \\ S_0[0][25] = v_{25} & \text{if } \gamma_2 = 0; \\ S_0[0][24] = v_{24} & \text{if } \gamma_1 = 0; \\ S_0[1][2] = v_2 & \text{if } \gamma_0 = 0. \end{array} \quad (12)$$

The other bits of S_0 are constants. (12) implies that Pattern-D contains 32 different assignments of cube variables depending on $\gamma \in \{0,1\}^5$. Table 10 lists 14 conditions of Pattern-D with γ for satisfying Requirement 3. Then, we get Lemma 4.

Lemma 4. Under the setting of Pattern-D with a guessed value $\gamma \in \{0,1\}^5$, the state of S_2 does not have any $v_{50}v_i$ for $v_i \in \mathcal{V}_1$ if and only if γ is correct and all conditions in Table 10 are true.

Table 9: Assignment of $\mathcal{V}_1$-variables for Pattern-D

Setting	Setting
$S_0[1][63] = v_{63}$	$S_0[0][30] = v_{30}$
$S_0[0][62] = v_{62}$	$S_0[0][29] = v_{29}$
$S_0[0][61] = S_0[1][61] = v_{61}$	$S_0[0][28] = v_{28}$
$S_0[0][60] = S_0[1][60] = v_{60}$	$S_0[0][27] = v_{27}$
$S_0[0][59] = S_0[1][59] = v_{59}$	$S_0[0][26] = v_{26}$
$S_0[0][58] = v_{58}$	$S_0[1][25] = v_{25}$
$S_0[0][57] = v_{57}$	$S_0[1][24] = v_{24}$
$S_0[1][56] = v_{56}$	$S_0[0][23] = v_{23}$
$S_0[0][55] = v_{55}$	$S_0[0][22] = v_{22}$
$S_0[0][54] = S_0[1][54] = v_{54}$	$S_0[0][21] = v_{21}$
$S_0[1][53] = v_{53}$	$S_0[0][20] = v_{20}$
$S_0[0][52] = v_{52}$	$S_0[0][19] = S_0[1][19] = v_{19}$
$S_0[1][51] = v_{51}$	$S_0[0][18] = v_{18}$
$S_0[0][49] = S_0[1][49] = v_{49}$	$S_0[1][17] = v_{17}$
$S_0[0][48] = v_{48}$	$S_0[0][16] = v_{16}$
$S_0[1][47] = v_{47}$	$S_0[0][15] = v_{15}$
$S_0[1][46] = v_{46}$	$S_0[0][14] = v_{14}$
$S_0[0][45] = v_{45}$	$S_0[0][13] = v_{13}$
$S_0[0][44] = S_0[1][44] = v_{44}$	$S_0[1][12] = v_{12}$
$S_0[0][43] = v_{43}$	$S_0[1][11] = v_{11}$
$S_0[0][42] = v_{42}$	$S_0[0][10] = v_{10}$
$S_0[0][41] = S_0[1][41] = v_{41}$	$S_0[0][9] = v_9$
$S_0[1][40] = v_{40}$	$S_0[0][8] = v_8$
$S_0[1][39] = v_{39}$	$S_0[0][7] = v_7$
$S_0[0][38] = v_{38}$	$S_0[0][6] = v_6$
$S_0[0][37] = v_{37}$	$S_0[0][5] = v_5$
$S_0[0][36] = v_{36}$	$S_0[0][4] = v_4$
$S_0[0][35] = v_{35}$	$S_0[0][3] = S_0[1][3] = v_3$
$S_0[1][34] = v_{34}$	$S_0[0][2] = S_0[1][2] = v_2$
$S_0[1][33] = v_{33}$	$S_0[0][1] = v_1$
$S_0[0][32] = v_{32}$	$S_0[0][0] = v_0$
$S_0[0][31] = v_{31}$	

Table 10: 14 Conditions of Pattern-D to satisfy Requirement 3

Conditions	Count
$S_0[3][61] = S_0[4][61]$	1
$S_0[3][60] = S_0[4][60]$	1
$S_0[3][59] = S_0[4][59]$	1
$S_0[3][54] = S_0[4][54]$	1
$S_0[2][50] = S_0[3][50] = S_0[4][50] = 1$	3
$S_0[2][49] = S_0[3][49] = S_0[4][49]$	2
$S_0[2][44] = S_0[3][44] = S_0[4][44]$	2
$S_0[2][41] = S_0[3][41]$	1
$S_0[2][19] = S_0[3][19]$	1
$S_0[3][3] = S_0[4][3]$	1

Proof. We concentrate on showing that a quadratic term $v_{50}v_i$ for $v_i \in \mathcal{V}_1$ appears on $S_{1.5}$ if all the conditions on column #50 of S_0 are true and γ is wrong. When all the conditions on column #50 of S_0 are true, by (1), $S_{0.5}[2][50]$ contains v_{50} only as a linear term and the other bits of the column #50 of $S_{0.5}$ are constant. After the p_L layer, $S_1[2][50]$, $S_1[2][49]$ and $S_1[2][44]$ contain v_{50} as a linear term.

Similarly to the proof of Lemma 2, we can show that $v_{50}v_2$ appears on the column #49 of $S_{1.5}$ when $\gamma_0 \neq S_0[3][2] \oplus S_0[4][2]$, that $v_{50}v_{24}$ appears on the column #49 of $S_{1.5}$ when $\gamma_1 \neq S_0[2][24] \oplus S_0[3][24]$, that $v_{50}v_{25}$ appears on the column #50 of $S_{1.5}$ when $\gamma_2 \neq S_0[2][25] \oplus S_0[3][25]$, that $v_{50}v_{46}$ appears on the column #49 of $S_{1.5}$ when $\gamma_3 \neq S_0[2][46] \oplus S_0[3][46]$, and that $v_{50}v_{47}$ appears on the column #50 of $S_{1.5}$ when $\gamma_4 \neq S_0[2][47] \oplus S_0[3][47]$.

The remaining of the proof is similar to that of Lemma 1. □

3.5 Pattern-E

We assume that $\delta = (\delta_7, ..., \delta_0) \in \{0,1\}^8$ is a guessed value for $(S_0[2][34] \oplus S_0[3][34], S_0[2][29] \oplus S_0[3][29], S_0[3][13] \oplus S_0[4][13], S_0[3][12] \oplus S_0[4][12], S_0[3][7] \oplus S_0[4][7], S_0[3][6] \oplus S_0[4][6], S_0[3][5] \oplus S_0[4][5], S_0[3][0] \oplus S_0[4][0])$. Pattern-E has 64 cube variables v_i for $0 \leq i \leq 63$. v_{60} is the only $\mathcal{V}_0$-variable and assigned as $S_0[0][60] = v_{60}$ and $S_0[1][60] = v_{60} \oplus 1$. $\mathcal{V}_1$-variables are assigned to S_0 as listed in Table 11. Additionally, we assign more $\mathcal{V}_1$-variables depending on δ as follows.

$$\begin{array}{ll} S_0[0][34] = v_{34} & \text{if } \delta_7 = 0; \\ S_0[0][29] = v_{29} & \text{if } \delta_6 = 0; \\ S_0[1][13] = v_{13} & \text{if } \delta_5 = 0; \\ S_0[1][12] = v_{12} & \text{if } \delta_4 = 0; \\ S_0[1][7] = v_7 & \text{if } \delta_3 = 0; \\ S_0[1][6] = v_6 & \text{if } \delta_2 = 0; \\ S_0[1][5] = v_5 & \text{if } \delta_1 = 0; \\ S_0[1][0] = v_0 & \text{if } \delta_0 = 0. \end{array} \tag{13}$$

The other bits of S_0 are constants. (13) implies that Pattern-E contains 256 different assignments of cube variables depending on $\delta \in \{0,1\}^8$. Table 12 lists 11 conditions of Pattern-E with δ for satisfying Requirement 3. Then, we get Lemma 5.

Lemma 5. Under the setting of Pattern-E with a guessed value $\delta \in \{0,1\}^8$, the state of S_2 does not have any $v_{60}v_i$ for $v_i \in \mathcal{V}_1$ if and only if δ is correct and all conditions in Table 12 are true.

Proof. We concentrate on showing that a quadratic term $v_{60}v_i$ for $v_i \in \mathcal{V}_1$ appears on $S_{1.5}$ if all the conditions on column #60 of S_0 are true and δ is wrong. When all the conditions on column #60 of

Table 11: Assignment of $\mathcal{V}_1$-variables for Pattern-E

Setting	Setting
$S_0[1][63] = v_{63}$	$S_0[0][30] = v_{30}$
$S_0[0][62] = v_{62}$	$S_0[1][29] = v_{29}$
$S_0[1][61] = v_{61}$	$S_0[0][28] = v_{28}$
$S_0[0][59] = S_0[1][59] = v_{59}$	$S_0[1][27] = v_{27}$
$S_0[0][58] = v_{58}$	$S_0[0][26] = v_{26}$
$S_0[0][57] = S_0[1][57] = v_{57}$	$S_0[0][25] = v_{25}$
$S_0[0][56] = S_0[1][56] = v_{56}$	$S_0[0][24] = v_{24}$
$S_0[0][55] = v_{55}$	$S_0[0][23] = v_{23}$
$S_0[0][54] = S_0[1][54] = v_{54}$	$S_0[1][22] = v_{22}$
$S_0[0][53] = v_{53}$	$S_0[1][21] = v_{21}$
$S_0[0][52] = v_{52}$	$S_0[0][20] = v_{20}$
$S_0[0][51] = S_0[1][51] = v_{51}$	$S_0[0][19] = v_{19}$
$S_0[1][50] = v_{50}$	$S_0[0][18] = v_{18}$
$S_0[1][49] = v_{49}$	$S_0[0][17] = v_{17}$
$S_0[0][48] = v_{48}$	$S_0[0][16] = v_{16}$
$S_0[0][47] = v_{47}$	$S_0[0][15] = v_{15}$
$S_0[0][46] = v_{46}$	$S_0[0][14] = v_{14}$
$S_0[0][45] = v_{45}$	$S_0[0][13] = v_{13}$
$S_0[1][44] = v_{44}$	$S_0[0][12] = v_{12}$
$S_0[1][43] = v_{43}$	$S_0[0][11] = v_{11}$
$S_0[0][42] = v_{42}$	$S_0[0][10] = v_{10}$
$S_0[0][41] = v_{41}$	$S_0[0][9] = v_9$
$S_0[0][40] = v_{40}$	$S_0[0][8] = v_8$
$S_0[0][39] = v_{39}$	$S_0[0][7] = S_0[1][7] = v_7$
$S_0[0][38] = v_{38}$	$S_0[0][6] = v_6$
$S_0[0][37] = v_{37}$	$S_0[0][5] = v_5$
$S_0[0][36] = v_{36}$	$S_0[0][4] = v_4$
$S_0[0][35] = S_0[1][35] = v_{35}$	$S_0[0][3] = v_3$
$S_0[1][34] = v_{34}$	$S_0[1][2] = v_2$
$S_0[0][33] = v_{33}$	$S_0[0][1] = v_1$
$S_0[0][32] = v_{32}$	$S_0[0][0] = v_0$
$S_0[0][31] = v_{31}$	

Table 12: 11 Conditions of Pattern-E to satisfy Requirement 3

Conditions	Count
$S_0[2][60] = S_0[3][60] = S_0[4][60] = 1$	3
$S_0[2][59] = S_0[3][59] = S_0[4][59]$	2
$S_0[2][57] = S_0[3][57]$	1
$S_0[2][56] = S_0[3][56]$	1
$S_0[2][54] = S_0[3][54] = S_0[4][54]$	2
$S_0[2][51] = S_0[3][51]$	1
$S_0[2][35] = S_0[3][35]$	1

S_0 are true, by (1), $S_{0.5}[2][60]$ contains v_{60} only as a linear term and the other bits of the column #60 of $S_{0.5}$ are constant. After the p_L layer, $S_1[2][60]$, $S_1[2][59]$ and $S_1[2][54]$ contain v_{60} as a linear term.

Similarly to the proof of Lemma 2, we can show that $v_{60}v_0$ appears on the column #54 of $S_{1.5}$ when $\delta_0 \neq S_0[3][0] \oplus S_0[4][0]$, that $v_{60}v_5$ appears on the column #59 of $S_{1.5}$ when $\delta_1 \neq S_0[3][5] \oplus S_0[4][5]$, that $v_{60}v_6$ appears on the column #60 of $S_{1.5}$ when $\delta_2 \neq S_0[3][6] \oplus S_0[4][6]$, that $v_{60}v_7$ appears on the column #54 of $S_{1.5}$ when $\delta_3 \neq S_0[3][7] \oplus S_0[4][7]$, that $v_{60}v_{12}$ appears on the column #59 of $S_{1.5}$ when $\delta_4 \neq S_0[3][12] \oplus S_0[4][12]$, that $v_{60}v_{13}$ appears on the column #60 of $S_{1.5}$ when $\delta_5 \neq S_0[3][13] \oplus S_0[4][13]$, that $v_{60}v_{29}$ appears on the column #54 of $S_{1.5}$ when $\delta_6 \neq S_0[2][29] \oplus S_0[3][29]$, and that $v_{60}v_{34}$ appears on the column #59 of $S_{1.5}$ when $\delta_7 \neq S_0[2][34] \oplus S_0[3][34]$.

The remaining of the proof is similar to that of Lemma 1. □

3.6 A Family of Patterns Pattern-F

We denote a cube variable assigned to bits of the column $\#i$ for $0 \leq i \leq 63$ of the initial state S_0 by v_i. We define a family of patterns, {Pattern-F(t)} for $t \in \{0, 1, ..., 63\}$. Pattern-F(t) has 40 cube variables. v_t is the only $\mathcal{V}_0$-variable of Pattern-F(t) and assigned as $S_0[0][t] = v_t$. We define the set $\mathcal{V}_1(t)$ of the $\mathcal{V}_1$-variables of Pattern-F(t) as

$$\mathcal{V}_1(t) = \{v_i \mid i \in \mathcal{I}_t \cup \mathcal{J}_t\},$$

where

$$\begin{aligned} \mathcal{I}_0 &= \{1,4,5,6,8,12,14,15,16,21,23,26,27,\\ &\quad 30,34,37,38,40,48,49,50,56,57,58,\\ &\quad 59,60,63\},\\ \mathcal{I}_t &= \{a + t \bmod 64 \mid a \in \mathcal{I}_0\},\\ \mathcal{J}_0 &= \{7,17,19,28,32,35,41,43,46,52,55,\\ &\quad 62\}, \text{ and}\\ \mathcal{J}_t &= \{a + t \bmod 64 \mid a \in \mathcal{J}_0\}. \end{aligned}$$

Then, 39 $\mathcal{V}_1$-variables are assigned as $S_0[0][i] = v_i$ for $v_i \in \mathcal{V}_1(t)$. The other bits of S_0 are constants. Table 13 lists 13 conditions of Pattern-F(t) for satisfying Requirement 3. Then, we get Lemma 6.

Lemma 6. For a given $t \in \{0, 1, ..., 63\}$, under the setting of Pattern-F(t), the state S_2 does not have any $v_t v_i$ for $i \in \mathcal{V}_1(t)$ if and only if all conditions in Table 13 are true.

Proof. We assume that all conditions in Table 13 are true. Then, by the setting of $S_0[0][t] = v_t$ and the equations (1) of the p_S layer, $S_{0.5}[0][t]$,

Table 13: 13 conditions of Pattern-F(t) to satisfy Requirement 3

Conditions	Count
$S_0[1][t] = 0$	1
$S_0[3][t+62] \oplus S_0[4][t+62] = 1$	1
$S_0[3][t+55] \oplus S_0[4][t+55] = 1$	1
$S_0[1][t+52] = 0$	1
$S_0[3][t+46] \oplus S_0[4][t+46] = 1$	1
$S_0[1][t+43] = 0$	1
$S_0[1][t+41] = 0$	1
$S_0[3][t+35] \oplus S_0[4][t+35] = 1$	1
$S_0[1][t+32] = 0$	1
$S_0[1][t+28] = 1$	1
$S_0[1][t+19] = 1$	1
$S_0[3][t+17] \oplus S_0[4][t+17] = 1$	1
$S_0[1][t+7] = 0$	1

$S_{0.5}[1][t]$, and $S_{0.5}[3][t]$ contain v_t only as a linear term. After the p_L layer, the only state bits $S_1[0][t]$, $S_1[1][t]$, $S_1[3][t]$, $S_1[1][t+3]$, $S_1[1][t+25]$, $S_1[0][t+36]$, $S_1[0][t+45]$, $S_1[3][t+47]$, and $S_1[3][t+54]$ contain v_t as a linear term, too. Likewise, it is easy to see that the other bits in columns $\#t$, $\#(t+3)$, $\#(t+25)$, $\#(t+36)$, $\#(t+45)$, $\#(t+47)$, and $\#(t+54)$ are constants. Therefore, by (1), no multiplication between v_t and v_i for any $i \in \mathcal{I}_t \cup \mathcal{J}_t$ appears on $S_{1.5}$ and S_2.

Next, we consider the case that not all conditions in Table 13 are true. It can be split into two subcases as follows:

- **Subcase 1:** If any of the conditions on the column $\#t$ of S_0 is false, then $v_t v_i$ for some $i \in \mathcal{I}_t$ appears on S_2. This is proved similarly to that given in Appendix A.1.

- **Subcase 2:** If all conditions on the column $\#t$ of S_0 are true, but any of the other conditions is false, then $v_t v_j$ for some $j \in \mathcal{J}_t$ appears on S_2. This is proved similarly to that given in Appendix A.2.

Therefore, the proof is completed. □

Additionally, we define Pattern-F($t, \mathcal{G}$) as the cube pattern having v_t as the only $\mathcal{V}_0$-variable and $\{v_i\}_{i \in \mathcal{I}_t \cup \mathcal{G}}$ as $\mathcal{V}_1$-variables, where $\mathcal{G} \subset \mathcal{J}_t$. Lemma 6 also implies that Pattern-F($t, \mathcal{G}$) satisfies *Requirement* 3 if and only if all conditions of the columns $\#j$ ($j \in \mathcal{G}$) of the state S_0 in Table 13 are true.

4 Conditional Cube Attack on Ascon-128a

We assume that the AD A is empty. Let $S_0 = S_0[0] \| \cdots \| S_0[4]$ be the input state to the Ascon permutation for the first block of plaintext in the encryption phase. Because of the rate $r = 128$ for Ascon-128a, we can control or know values of $S_0[0] \| S_0[1]$ by choosing the first plaintext blocks and obtaining the corresponding ciphertext blocks, while $S_0[2] \| S_0[3] \| S_0[4]$ is secret and uncontrollable, and depends on the nonce N. We show how the state-recovery attack in Section 4.1 recovers $S_0[2] \| S_0[3] \| S_0[4]$ for $(\star, 7, \star)$-round Ascon-128a, and how the key recovery attack in Section 4.2 recovers the secret key K for $(\star, 7, \star)$-round Ascon-128a based on the knowledge of the recovered state S_0.

4.1 State-recovery Attack

Our state-recovery attack on Ascon-128a recovers 192 bits $S_0[2] \| S_0[3] \| S_0[4]$ of the state S_0 by using Pattern-A, Pattern-B, Pattern-C, Pattern-D, and Pattern-E in this order. We construct cubes based on them. Since we have $\lambda = 1$ and $\mu = 63$ for each pattern, we have $n = 5$ and so, by Theorem 1, the cube-sums corresponding to the patterns after Round6 of Ascon permutation would be zero if all conditions for them are satisfied. Figure 5 shows how we use these patterns. The conditions which are highlighted with gray color were already defined from the previous pattern. We expect each of the conditions which are highlighted with a black-line box be satisfied with probability $\frac{1}{2}$. In total, 74 different conditions should be satisfied for success of the attack.

In order to make a cube, for the same nonce N, we choose the first plaintext blocks such that the cube variables on $S_0[0] \| S_0[1]$ are activated and the other bits on $S_0[0] \| S_0[1]$ are constants. The second plaintext blocks can be any values. Then, we check whether the cube-sum is zero by using the knowledge of plaintexts and ciphertexts in the second block. Namely, we use online cubes and online cube-sum computations.

The procedure of our state-recovery attack on $(\star, 7, \star)$-round Ascon-128a consists of five steps as depicted in Figure 6, and is described as follows.

Step 1. We make a cube for Pattern-A and check whether its cube-sum on $S_7[0] \| S_7[1]$ is zero, where S_7 is the state right after Round6 and the first two rows of the output of p^7 in the first block of the encryption phase. We repeat this process until

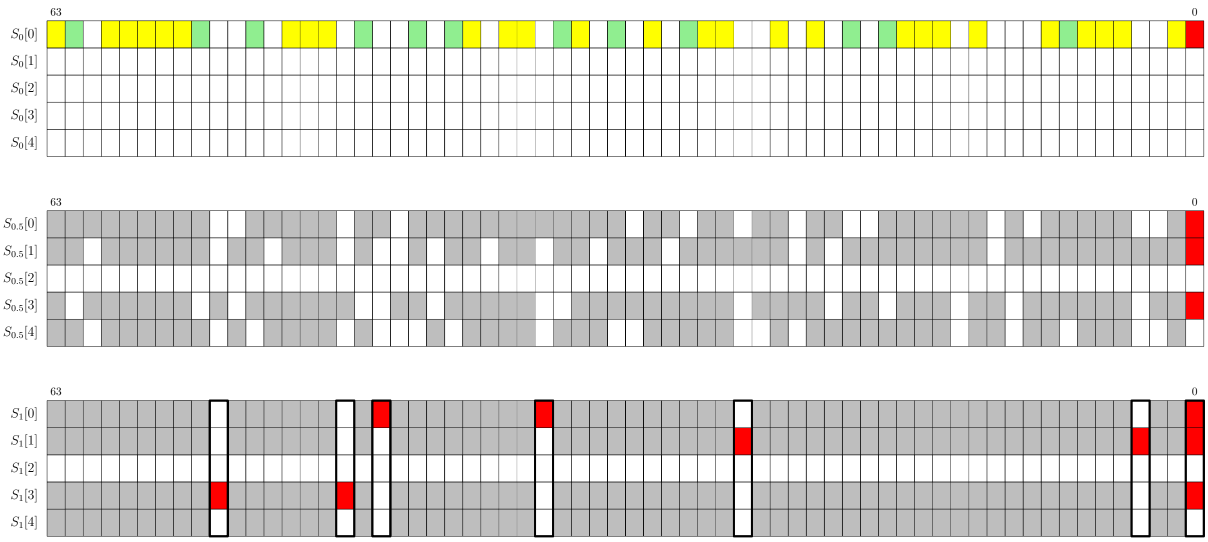

Figure 3: Propagation of the $\mathcal{V}_0$-variable v_0 and $\mathcal{V}_1$-variables in Pattern-F(0). The bits influenced by v_0 are marked in red; v_i's for $i \in \mathcal{I}_0$ are marked in yellow; v_i's for $i \in \mathcal{J}_0$ are marked in green; the bits influenced by $\mathcal{V}_1$-variables and not by v_0 are marked in gray; The bits marked in white are constants.

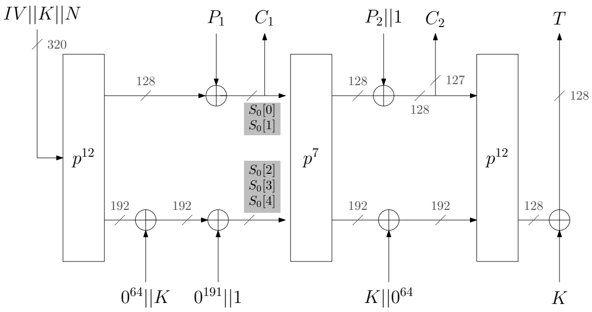

Figure 4: State recovery attack on $(\star, 7, \star)$-round Ascon-128a

a zero cube-sum is found, by choosing the nonce N randomly. Since the all the 38 conditions in Table 4 hold with the probability 2^{-38}, on average, we expect 2^{38} iterations for it. With each N, we choose 2^{64} two-block plaintexts of the form $P = P_1 \| P_2$ where P_1's are used for constructing a cube and (P_2, C_2)'s are used for evaluating the cube-sum. In particular, to minimize the data complexity, we choose the last plaintext block P_2's with $\text{Len}(P_2) = 127$. It implies that we can check the cube-sum for 127 bits at the output of p^7. We expect that Step 1 require 2^{102} $(=2^{38} \times 2^{64})$ plaintexts, while the chance that false conditions make such a cube-sum to be zero is negligible. If we find a zero cube-sum, we go to Step 2 together with the corresponding nonce N, which we denote by N_{zero}.

Step 2. We make cubes for Pattern-B with N_{zero}. Pattern-A and Pattern-B share 8 conditions, highlighted with gray color in Pattern-B column in Figure 5. Guessing $\alpha \in \{0,1\}^7$, we try at most 2^7 cubes. If we find a zero cube-sum on $S_7[0] \| S_7[1]$, we go to Step 3. Otherwise, we go to Step 1. So, Step 2 requires at most 2^{71} $(=2^7 \times 2^{64})$ two-block plaintexts. Since Step 1 ensures those common conditions are true, we only need to consider the remaining 4 conditions for Pattern-B. Therefore, we go to Step 3 with the probability of 2^{-4}.

Step 3. We make cubes for Pattern-C with N_{zero}. Pattern-C shares 10 conditions with previous patterns, highlighted with gray color in Pattern-C column in Figure 5. Letting $(\beta_5, \beta_3, \beta_0) = (\alpha_5, \alpha_3, \alpha_0)$ and guessing $(\beta_4, \beta_2, \beta_1) \in \{0,1\}^3$, we try at most 2^3 cubes. If we find a zero cube-sum on $S_7[0] \| S_7[1]$, we go to Step 4. Otherwise, we go to Step 1. So, Step 3 requires at most 2^{67} $(=2^3 \times 2^{64})$ two-block plaintexts. Since Step 2 ensures those common conditions are true, we only need to consider the remaining 3 conditions for Pattern-C. Therefore, we go to Step 4 with the probability of 2^{-3}.

Step 4. We make cubes for Pattern-D with zero. Pattern-D shares 7 conditions with previous patterns, highlighted with gray color in Pattern-D column in Figure 5. Letting $\gamma_0 = \beta_0$ and guessing $(\gamma_4, ..., \gamma_1) \in \{0,1\}^4$, we try at most 2^4 cubes. If we find a zero cube-sum on $S_7[0] \| S_7[1]$, we go to Step 5. Otherwise, we go to Step 1. So, Step 4 requires at most 2^{68} $(=2^4 \times 2^{64})$ two-block plaintexts. Since Step 3 ensures those common conditions are

	Pattern-A (38 conditions)	Pattern-B (19 conditions)	Pattern-C (19 conditions)	Pattern-D (19 conditions)	Pattern-E (19 conditions)
63:	$S_0[2][63] = S_0[3][63] = S_0[4][63] = 0$				
62:	$S_0[2][62] = S_0[3][62] = S_0[4][62]$	$S_0[2][62] = S_0[3][62] = S_0[4][62] = 1$			
61:	$S_0[3][61] = S_0[4][61]$	$S_0[2][61] = S_0[3][61] = S_0[4][61]$	$S_0[3][61] = S_0[4][61]$	$S_0[3][61] = S_0[4][61]$	
60:	$S_0[2][60] = S_0[3][60]$			$S_0[3][60] = S_0[4][60]$	$S_0[2][60] = S_0[3][60] = S_0[4][60] = 1$
59:	$S_0[2][59] = S_0[3][59]$	$S_0[2][59] = S_0[3][59]$		$S_0[3][59] = S_0[4][59]$	$S_0[2][59] = S_0[3][59] = S_0[4][59]$
58:					
57:	$S_0[2][57] = S_0[3][57] = S_0[4][57]$		$S_0[2][57] = S_0[3][57] = S_0[4][57] = 1$		$S_0[2][57] = S_0[3][57]$
56:		$S_0[2][56] = S_0[3][56] = S_0[4][56]$	$S_0[2][56] = S_0[3][56] = S_0[4][56]$		$S_0[2][56] = S_0[3][56]$
55:					
54:	$S_0[2][54] = S_0[3][54] = S_0[4][54]$		$S_0[2][54] = S_0[3][54]$	$S_0[3][54] = S_0[4][54]$	$S_0[2][54] = S_0[3][54] = S_0[4][54]$
53:					
52:	$S_0[3][52] = S_0[4][52]$				
51:	$S_0[4][51] = 0$		$S_0[2][51] = S_0[3][51] = S_0[4][51]$		$S_0[2][51] = S_0[3][51]$
50:				$S_0[2][50] = S_0[3][50] = S_0[4][50] = 1$	
49:				$S_0[2][49] = S_0[3][49] = S_0[4][49]$	
48:					
47:					
46:					
45:	$S_0[3][45] = S_0[4][45]$				
44:	$S_0[2][44] = S_0[3][44] = S_0[4][44] = 0$			$S_0[2][44] = S_0[3][44] = S_0[4][44]$	
43:					
42:	$S_0[4][42] = 0$				
41:	$S_0[2][41] = S_0[3][41]$			$S_0[2][41] = S_0[3][41]$	
40:	$S_0[4][40] = 0$				
39:					
38:	$S_0[2][38] = S_0[3][38]$				
37:	$S_0[2][37] = S_0[3][37]$	$S_0[2][37] = S_0[3][37]$			
36:					
35:	$S_0[2][35] = S_0[3][35] = S_0[4][35] = 0$				$S_0[2][35] = S_0[3][35]$
34:					
33:					
32:	$S_0[2][32] = S_0[3][32]$		$S_0[2][32] = S_0[3][32]$		
31:					
30:					
29:					
28:					
27:					
26:					
25:					
24:					
23:					
22:					
21:	$S_0[4][21] = 0$				
20:					
19:	$S_0[2][19] = S_0[3][19]$			$S_0[2][19] = S_0[3][19]$	
18:					
17:					
16:	$S_0[3][16] = S_0[4][16]$				
15:	$S_0[3][15] = S_0[4][15]$	$S_0[3][15] = S_0[4][15]$			
14:					
13:					
12:	$S_0[4][12] = 0$				
11:					
10:	$S_0[2][10] = S_0[3][10] = S_0[4][10]$		$S_0[3][10] = S_0[4][10]$		
9:	$S_0[3][9] = S_0[4][9]$	$S_0[3][9] = S_0[4][9]$	$S_0[3][9] = S_0[4][9]$		
8:	$S_0[3][8] = S_0[4][8]$	$S_0[3][8] = S_0[4][8]$			
7:					
6:	$S_0[4][6] = 0$				
5:					
4:					
3:	$S_0[3][3] = S_0[4][3]$		$S_0[3][3] = S_0[4][3]$	$S_0[3][3] = S_0[4][3]$	
2:					
1:					
0:					

Figure 5: Five patterns for the state recovery attack (74 different conditions in total)

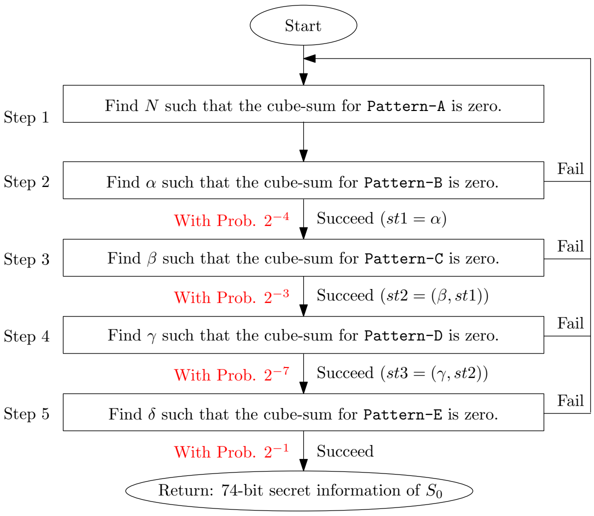

Figure 6: Procedure of the state recovery attack on Ascon-128a

true, we only need to consider the remaining 7 conditions for Pattern-D. Therefore, we go to Step 5 with the probability of 2^{-7}.

Step 5. We make cubes for Pattern-E with zero. Pattern-E shares 10 conditions with previous patterns, highlighted with gray color in Pattern-E column in Figure 5. Letting $\delta_3 = \alpha_1$ and guessing $(\delta_7, ..., \delta_4, \delta_2, ..., \delta_0) \in \{0,1\}^7$, we try at most 2^7 cubes. If we find a zero cube-sum on $S_7[0]\|S_7[1]$, we return 74 bits of secret information of $S_0[2]\|S_0[3]\|S_0[4]$ because the number of essentially considered conditions from Pattern-A to Pattern-E is 53 and the number of essentially guessed bits Step 2 to Step 5 is 21. Otherwise, we go to Step 1. So, Step 5 requires at most 2^{71} ($=2^7 \times 2^{64}$) two-block plaintexts. Since Step 4 ensures those common conditions are true, we only need to consider the one remaining condition for Pattern-E. Therefore, we terminate this procedure and get the 74-bit information of $S_0[2]\|S_0[3]\|S_0[4]$ with the probability of 2^{-1}.

Note that the nonce N is randomly chosen for different cubes. Therefore, this attack requires the data complexity of 2^{117} two-block plaintexts because Step 1 requires the data complexity of 2^{102} and is repeated $2^{15}(= 2^4 \times 2^3 \times 2^7 \times 2)$. After recovering 74 bits of $S_0[2]\|S_0[3]\|S_0[4]$, we recover the remaining 118 bits of $S_0[2]\|S_0[3]\|S_0[4]$ through an exhaustive search by using a plaintext $P = P_1\|P_2$ and the corresponding ciphertext $C = C_1\|C_2$. Considering the discussion about time complexity in Section 2.3, we estimate its time complexity as $2^{116.2}(\approx 2^{118} \times 7/24)$.

4.2 Key Recovery Attack

We assume that the full information of the state right after the initialization phase is recovered by the state-recovery attack in Section 4.1 and that we reuse the nonce value N_{zero} such that the recovered state information is fixed during the key recovery attack. The attack consists of online and offline phases, and requires the memory complexity of 2^{32} 256-bit values. Since the state-recovery attack must be preceded for the key recovery attack and the complexity of the former largely dominates that of the latter both in data and time, we do not specifically explain any cost except memory in Sections 4.2.1 and 4.2.2.

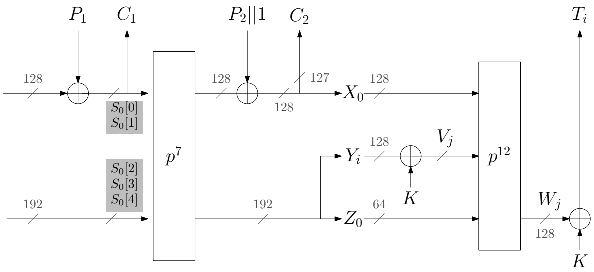

Figure 7: Key recovery attack on $(\star, 7, \star)$-round Ascon-128a

4.2.1 Online Phase

Let $X\|Y\|Z$ be the state right after the encryption phase, where X and Y are 128 bits and Z is 64 bits, as you see Figure 7. We construct a two-block plaintext as follows. The first 128-bit block P_1 is randomly chosen. Since we know the state value after the initialization phase, we can use P_1 to get the output state of the permutation p^7 by offline computation. P_2 is chosen as the 127-bit value for which the padded last block $P_2\|1$ is XORed with the first 128 bits of the output state of the permutation p^7 to fix the most significant 127 bits of X to a certain value. After this computation, (X, Z) is fixed to a certain 192-bit value (X_0, Z_0) with the probability of 2^{-65}. At cost of 2^{97} operations of p^7, we can collect 2^{32} $X_0\|Y\|Z_0$'s. We denote them by $\{X_0\|Y_i\|Z_0\}_{1\leq i\leq 2^{32}}$. Then, by using the plaintexts $P_1\|P_2$'s corresponding to $\{X_0\|Y_i\|Z_0\}_{1\leq i\leq 2^{32}}$ as an online query where the nonce is fixed, we get $\{(X_0\|Y_i\|Z_0, T_i)\}_{1\leq i\leq 2^{32}}$, where T_i is the tag corresponding to $X_0\|Y_i\|Z_0$. Essentially, we only need to store $\{(Y_i, T_i)\}_{1\leq i\leq 2^{32}}$ and (X_0, Z_0) in a table $\mathcal{Q}$.

4.2.2 Offline Phase

Let V be the 128-bit value such that $V = Y \oplus K$ for the 128-bit secret key K. Let W be the 64-bit value such that $T = W \oplus K$ for the tag T by Ascon-128a. As you see Figure 7, V and W are contained in the input and output states of the Ascon permutation p^{12} in the finalization phase.

Given X_0 and Z_0 which were computed from the online phase, we randomly choose 2^{96} 128-bit values of V_j to get the corresponding W_j's by the offline computation of $p^{12}(X_0\|V_j\|Z_0)$. Since we have 2^{32} tuples of (Y_i, T_i)'s and tuples of 2^{96} (V_j, W_j) and (14) holds with the probability of 2^{-128}, we expect to get one match.

$$Y_i \oplus T_i = V_j \oplus W_j. \tag{14}$$

Then, we expect to obtain the right key value for K by computing the form of $Y_i \oplus V_j$. We can run this process without any additional memory.

4.2.3 Complexity

We should consider that the full-state-recovery attack must precede the key recovery attack. Therefore, we estimate the total cost for the key recovery attack on the $(\star, 7, \star)$-round Ascon-128a as the data complexity of 2^{117}, the time complexity of $2^{116.2}$, and the memory complexity of 2^{32}.

5 Conditional Cube Attack on Ascon-128 and Ascon-80pq

We assume that the AD A is empty. Let $S_0 = S_0[0]\|\cdots\|S_0[4]$ be the input state to the Ascon permutation for the first block of plaintext in the encryption phase. Because of the rate $r = 64$ for Ascon-128 and Ascon-80pq, we can control or know values of $S_0[0]$ by choosing the first plaintext blocks and obtaining the corresponding ciphertext blocks, while $S_0[1]\|S_0[2]\|S_0[3]\|S_0[4]$ is secret and uncontrollable, and only depends on the nonce N to change. We show how the state-recovery attack in Section 5.1 recovers $S_0[1]\|S_0[2]\|S_0[3]\|S_0[4]$ for $(\star, 6, \star)$-round Ascon-128 and Ascon-80pq, and how the key recovery attack in Section 4.2 recovers the secret keys for $(\star, 6, \star)$-round Ascon-128 and Ascon-80pq based on the knowledge of the recovered state S_0.

5.1 State-recovery attack

Our state-recovery attack on Ascon-128 and Ascon-80pq recovers 256 bits $S_0[1]\|S_0[2]\|S_0[3]\|S_0[4]$ of S_0 by using {Pattern-F(t)} described in Section 3.6. We construct cubes based on them. Each of them uses $\lambda = 1$ and $\mu = 31$. So, $n = 4$ and by Theorem 1, the cube-sums corresponding to the patterns after Round5 of Ascon permutation would be zero if all conditions for them are satisfied.

In order to make a cube, for the same nonce N, we choose the first plaintext blocks such that the cube variables on $S_0[0]$ are activated and the other bits on $S_0[0]$ are constants. The second plaintext blocks can be any values. Then, we check wether the cube-sum is zero by using the knowledge of plaintexts and ciphertexts in the second block.

The procedure of the state-recovery attack on $(\star, 6, \star)$-round Ascon-128 or Ascon-80pq consists of four steps and is described as follows.

Step 1. We use Pattern-F($0, \mathcal{G}$) by defining $\mathcal{G} = \{7, 17, 19, 28\} \subset \mathcal{J}_0$. We choose a nonce N randomly, make a cube for Pattern-F($0, \mathcal{G}$), and check whether its cube-sum on $S_6[0]$ is zero, where S_6 is the state right after Round5 and the first row of the output of p^6 in the first block of the encryption phase. We repeat this process until a zero cube-sum is found, by choosing the nonce N randomly. Since the five conditions related to v_0, v_7, v_{17}, v_{19} and v_{28} in Table 13 hold with the probability of 2^{-5}, on average, we expect 2^5 iterations for it. With each N, we choose 2^{32} two-block plaintexts of the form $P = P_1\|P_2$ where P_1's are used for constructing a cube and (P_2, C_2)'s are used for evaluating the cube-sum. So, we expect Step 1 require $2^{37}(= 2^5 \times 2^{32})$ plaintexts, while the chance that false conditions mask such a cube-sum to be zero is negligible. If we find a zero cube-sum, we go to Step 2 together with the corresponding nonce N_{zero}.

Step 2. We consider a set $\mathcal{A} = \{7, 17, 19, 28\}$ and an index $i_0 = 28$, and do the followings.

We make a cube for Pattern-F($0, \mathcal{G}$) with N_{zero} by redefining $\mathcal{G}$ such that the element i_0 is replaced with any other element in $\mathcal{J}_0 \setminus \mathcal{A}$ and by choosing 2^{32} two-block plaintexts similarly to Step 1. If the cube-sum on $S_6[0]$ is zero, we guess the condition corresponding to i_0 is true. Otherwise, we guess the condition corresponding to i_0 is wrong.

We repeat the above process by updating $\mathcal{A} \leftarrow \mathcal{A} \cup \{i_0\}$ until we get information of all conditions corresponding to $\mathcal{J}_0$. Since we need 8 iterations, we expect Step 2 require $2^{35}(= 8 \times 2^{32})$ plaintexts. At the end of Step 2, we have 13-bit information of $S_0[1]\|S_0[2]\|S_0[3]\|S_0[4]$, and go to Step 3.

Step 3. In Step 3, we use various Pattern-F($t, \mathcal{G}$) with $t \neq 0$. We should take an index t such that $S_0[1][t] = 0$ which is necessary for the application of Pattern-F($t, \mathcal{G}$). For example, we can take $t = 7$, 32, 41, 43 or 52 because we have $S_0[1][i] = 0$ for

$i = 7, 32, 41, 43, 52$ from Step 2.

Then, with an index t such that $S_0[1][t] = 0$, we define $\mathcal{G}$ as the set of any four elements randomly selected from $\mathcal{J}_t$, make a cube for Pattern-F$(t, \mathcal{G})$, and check whether its cube-sum on $S_6[0]$ is zero.

If the cube-sum is nonzero, we try the above process by selecting any other four elements from $\mathcal{J}_t$ to redefine $\mathcal{G}$. Since the number of ways to choose 4 out of 12 indices is $\binom{12}{4} = 495$, we should repeat the above process at most 495 times for finding a zero cube-sum. However, since a cube has a zero cube-sum with the high probability of 92.7%, we expect the number of iterations to average 16. It requires $2^{36} = 2^4 \times 2^{32}$ plaintexts.

If we do not find any zero cube-sum, we apply a new cube for Pattern-F$(t, \mathcal{G})$ with a different t to the above process. If a zero-sum is found, we go to Step 4 together with t.

Step 4. Step 4 is similar to Step 2. Let $\mathcal{A} = \mathcal{G}$. We select an index i_0 from $\mathcal{A}$, and do the followings.

We make a cube for Pattern-F$(t, \mathcal{G})$ with N_{zero} by redefining $\mathcal{G}$ such that the element i_0 is replaced with any other element in $\mathcal{J}_t \backslash \mathcal{A}$. If the cube-sum on $S_6[0]$ is zero, we guess the condition corresponding to i_0 is true. Otherwise, we guess the condition corresponding to i_0 is wrong.

We repeat the above process by updating $\mathcal{A} \leftarrow \mathcal{A} \cup \{i_0\}$ until we get information of all conditions corresponding to $\mathcal{J}_t$. Since we need 8 iterations, we expect it require $2^{35} (= 8 \times 2^{32})$ plaintexts.

We repeat these Steps 3 and 4 31 times to collect secret information of $(S_0[1], S_0[3] \oplus S_0[4])$.

We expect around 32 patterns among 64 patterns in {Pattern-F(t)} be available because the condition $S_0[1][t] = 0$ corresponding to the cube variable v_t is true with the probability of 2^{-1}. Our experiment with 10,000 trials shows that 31 iterations of Steps 3 and 4 lead to about 12% chance to recover the full 128-bit information of $(S_0[1], S_0[3] \oplus S_0[4])$. Therefore, the full information of $(S_0[1], S_0[3] \oplus S_0[4])$ is recovered with high probability, during 9 $(> (12\%)^{-1})$ iterations of the entire process from Step 1 to Step 4. We expect recovering 128-bit $(S_0[1], S_0[3] \oplus S_0[4])$ require $2^{44.78}$ $(=9 \times (2^{37} + 2^{35} + 31 \times (2^{36} + 2^{35})))$ two-block plaintexts.

Based on this attack result, the full state information can be recovered. Independently, Baudrin et al. [1] presented a better full-state-recovery attack on full-round Ascon-128 with both data complexity and time complexity of $2^{39.6}$. Ours and Baudrin et al.'s attacks can be also applied to Ascon-80pq.

5.2 Key Recovery Attack on Ascon-80pq

We assume that the full state information is already recovered with the empty AD A by Baudrin et al's attack [1] and that we know the nonce N_{zero} related to the recovered information. We describe the key recovery attack on Ascon-80pq based on the state information. The attack recovers the 160-bit secret key through online and offline phases. We reuse the nonce value N_{zero} such that the recovered state information is fixed during the key recovery attack.

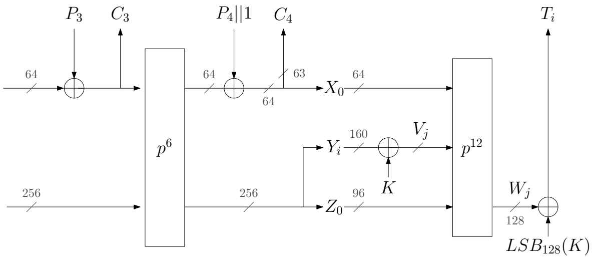

Figure 8: Key recovery attack on Ascon-80pq

5.2.1 Online Phase

Let $X\|Y\|Z$ be the state right after the encryption phase, where X is 64 bits, Y is 160 bits, and Z is 96 bits, as you see Fig. 8. We need 2^{32} $X\|Y\|Z$'s for the next attack phase, where X and Z are fixed. Note that we know the state value after the initialization phase. We can compute them with four-block plaintext $P = (P_1, P_2, P_3, P_4)$ at cost of 2^{129} offline computations of p^6, as follows. Firstly, we perform two operations for the first p^6 permutation with two randomly chosen 64-bit values for P_1. Secondly, for two output states of the first p^6 permutation, we perform 2^{64} operations for the second p^6 permutation with all possible 2^{64} 64-bit values for P_2. Thirdly, for the 2^{65} output states of the second p^6 permutation, we perform 2^{64} operations for the third p^6 permutation with all possible 2^{64} 64-bit values for P_3. Finally, for the 2^{129} output states of the third p^6 permutation, we choose the 63-bit values for P_4 such that the most significant 63 bits of X is fixed to a certain 63-bit value. After these computations, for a fixed 160-bit value (X_0, Z_0), we obtain 2^{32} $X_0\|Y\|Z_0$'s because each P leads to (X_0, Z_0) with the probability of 2^{-97}.

We denote the computed $X_0\|Y\|Z_0$'s by $\{X_0\|Y_i\|Z_0\}_{1 \le i \le 2^{32}}$. Then, by using the plaintexts $P_1\|P_2\|P_3$'s corresponding to $\{X_0\|Y_i\|Z_0\}_{1 \le i \le 2^{32}}$ as online queries where the nonce is fixed, we get $\{(X_0\|Y_i\|Z_0, T_i)\}_{1 \le i \le 2^{32}}$, where T_i is the tag corresponding to $X_0\|Y_i\|Z_0$. Essentially, we only need to store $\{(Y_i, T_i)\}_{1 \le i \le 2^{32}}$ and (X_0, Z_0) in a table $\mathcal{Q}$.

5.2.2 Offline Phase

Let V be the 160-bit value such that $V = Y \oplus K$ for the 160-bit secret key K. Let W be the 128-bit value such that $T = W \oplus \mathrm{LSB}_{128}(K)$ for the tag T by Ascon-80pq. As you see Figure 8, V and W are contained in the input and output states of the Ascon permutation p^{12} in the finalization phase.

Given X_0 and Z_0 which were computed from the online phase, we randomly choose 2^{128} 160-bit values of V_j to get the corresponding W_j's by the offline computation of $p^{12}(X_0 \| V_j \| Z_0)$. Since we have 2^{32} tuples of (Y_i, T_i)'s and tuples of 2^{128} (V_j, W_j) and (15) holds with the probability of 2^{-128}, we expect to get 2^{32} matches.

$$\mathrm{LSB}_{128}(Y_i) \oplus T_i = \mathrm{LSB}_{128}(V_j) \oplus W_j. \tag{15}$$

Then, we expect to obtain the right key value for K by testing 2^{32} candidates with the form of $Y_i \oplus V_j$. We can run this process without any additional memory.

5.2.3 Complexity

We should consider that the full-state-recovery attack must precede the key recovery attack. Therefore, considering the discussion about time complexity in Section 2.3, we estimate the total cost for the key recovery attack on the full-round Ascon-80pq as the data complexity of $2^{39.6}$, the time complexity of $2^{128}(\approx 2^{129} \times 6/24 + 2^{128} \times 12/24)$, and the memory complexity of 2^{32}.

6 Conclusion

In this paper, we study the resistance of the Ascon family against conditional cube attacks in nonce-misuse setting, and present new state- and key-recovery attacks. In particular, our attack results on Ascon-128a are the best known ones as far as we know. Although our attacks do not invalidate designers' claim, those allow us to understand the security of Ascon in nonce-misuse setting.

References

[1] Jules Baudrin, Anne Canteaut, and Léo Perrin. Practical Cube-attack against Nonce-misused Ascon. Fifth NIST Lightweight Cryptography Workshop 2022, May 2022.

[2] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Permutation-based Encryption, Authentication and Authenticated Encryption. DIAC – Directions in Authenticated Ciphers, 2012. `https://keccak.team/files/KeccakDIAC2012.pdf`.

[3] CAESAR committee. CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness, 2014-2019. `https://competitions.cr.yp.to/caesar.html`.

[4] Itai Dinur and Adi Shamir. Cube attacks on tweakable black box polynomials. In Antoine Joux, editor, *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, volume 5479 of *Lecture Notes in Computer Science*, pages 278–299. Springer, 2009.

[5] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer. Ascon. Submission to the NIST Lightweight Cryptography Standardization Process, 2021. `https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/ascon-spec-final.pdf`.

[6] Senyang Huang, Xiaoyun Wang, Guangwu Xu, Meiqin Wang, and Jingyuan Zhao. Conditional cube attack on reduced-round keccak sponge function. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part II*, volume 10211 of *Lecture Notes in Computer Science*, pages 259–288, 2017.

[7] Xuejia Lai. Higher order derivatives and differential cryptanalysis. In Richard E. Blahut-Daniel J. CostelloJr.Ueli MaurerThomas Mittelholzer, editor, *Communications and Cryptography. The Springer International Series in Engineering and Computer Science (Communications and Information Theory)*, volume 276, pages 227–233. Springer, 1994.

[8] Yanbin Li, Guoyan Zhang, Wei Wang, and Meiqin Wang. Cryptanalysis of round-reduced ASCON. *Sci. China Inf. Sci.*, 60(3):38102, 2017.

[9] Zheng Li, Xiaoyang Dong, and Xiaoyun Wang. Conditional Cube Attack on Round-Reduced ASCON. *IACR Transactions on Symmetric Cryptology*, 2017(1):175–202, Mar. 2017.

[10] Raghvendra Rohit, Kai Hu, Sumanta Sarkar, and Siwei Sun. Misuse-free key-recovery and distinguishing attacks on 7-round ascon. *IACR Trans. Symmetric Cryptol.*, 2021(1):130–155, 2021.

A Proofs of Subcases in the Proof of Lemma 1

A.1 Proof of Subcase 1

We assume that all conditions on the column #0 of S_0, $S_0[2][63] = S_0[3][63] = S_0[4][63] = 0$, in Table 4 are not true. Namely, we consider that $S_0[2][63] \neq S_0[3][63]$, $S_0[3][63] \neq S_0[4][63]$, or $S_0[4][63] \neq 0$.

Firstly, we suppose that $S_0[2][63] \neq S_0[3][63]$. Then, by (2), $S_{0.5}[1][63]$ contains the $\mathcal{V}_0$-variable v_{63} as a linear term. By Σ_1 operation, $S_1[1][63]$, $S_1[1][24]$, and $S_1[1][2]$ also contain v_{63} as a linear term. On the other hand, $S_{0.5}[2][8]$ and $S_{0.5}[2][3]$ contain v_8 and v_3 as linear terms, respectively. By Σ_2 operation, $S_1[2][2]$ contains v_8 and v_3 as a linear term. After p_S operation on the state S_1, the product between $S_1[1][2]$ and $S_1[2][2]$ appears on the column #2 of $S_{1.5}$, which contains $v_{63}v_8$ and $v_{63}v_3$.

Secondly, we suppose that $S_0[3][63] \neq S_0[4][63]$. Then, by (3), $S_{0.5}[3][63]$ contains the $\mathcal{V}_0$-variable v_{63} as a linear term. By Σ_3 operation, $S_1[3][63]$, $S_1[3][53]$, and $S_1[3][46]$ also contain v_{63} as a linear term. On the other hand, $S_{0.5}[2][59]$, $S_{0.5}[2][54]$, and $S_{0.5}[2][52]$ contain v_{59}, v_{54}, and v_{52} as linear terms, respectively. By Σ_2, $S_1[2][53]$ contains v_{59} and v_{54} as linear terms, and $S_1[2][46]$ contains v_{52} as a linear term. After p_S operation on the state S_1, the product between $S_1[2][53]$ and $S_1[3][53]$ appears on the column #53 of $S_{1.5}$, which contains $v_{63}v_{59}$ and $v_{63}v_{54}$, and the product between $S_1[2][46]$ and $S_1[3][46]$ appears on the column #46 of $S_{1.5}$, which contains $v_{63}v_{52}$.

Finally, we suppose that $S_0[4][63] = 1$. Then, by (1), $S_{0.5}[4][63]$ contains the conditional cube variable v_{63} as a linear term. By Σ_4 operation, $S_1[4][63]$, $S_1[4][56]$, and $S_1[4][23]$ also contain v_{63} as a linear term. On the other hand, $S_{0.5}[1][53]$ and $S_{0.5}[1][31]$ contain v_{53} and v_{31} as linear terms, respectively. By Σ_1, $S_1[1][56]$ contains v_{53} and v_{31} as linear terms. After p_S operation on the state S_1, the product between $S_1[1][56]$ and $S_1[4][56]$ appears on the column #56 of $S_{1.5}$, which contains $v_{63}v_{53}$ and $v_{63}v_{31}$.

A.2 Proof of Subcase 2

Since the conditions on the column #63 of S_0 are true, by (1), $S_{0.5}[0][63]$ and $S_{0.5}[2][63]$ contain the $\mathcal{V}_0$-variable v_{63} only as a linear term. After the p_L layer, $S_1[0][63]$, $S_1[0][44]$, $S_1[0][35]$, $S_1[2][63]$, $S_1[2][62]$, and $S_1[2][57]$ contain v_{63} as a linear term. Then, for any $i \neq 63$, if any condition on the column #i of S_0 in Table 4 is false, then the $\mathcal{V}_1$-variable v_i appears as a linear term on the columns #63, #62, #57, #44, or #35 of the state S_1, It is followed by the appearance of the quadratic term $v_{63}v_i$ appear on the states $S_{1.5}$ and S_2.

Table 14 summarizes (i, j)'s such that $v_{63}v_i$ appears on the column #j of $S_{1.5}$ if any condition on the column #i for nonzero i of S_1 in Table 4 is false. Note that $j \in \{35, 44, 57, 62, 63\}$. With Table 14 we can conclude the proof because it implies a quadratic term $v_{63}v_i$ for some nonzero i also appears on the state S_2 under the same assumption.

Table 14: (i, j)'s implying that $v_{63}v_i$ appears on the column #j of $S_{1.5}$ if all conditions on the column #i for nonzero i of S_1 in Table 4 are not true, while conditions on the column #63 of S_1 in Table 4 are true.

i	j	i	j	i	j
62	62	44	44	19	44
61	44	42	35	16	63
60	63	41	44	15	62
59	62	40	63	12	35
57	57	38	63	10	57
54	57	37	62	9	63
52	35	35	35	8	62
51	44	32	57, 35	6	63
45	35	21	44	3	57