# New Conditional Cube Attack on Keccak Keyed Modes

Zheng Li[1], Xiaoyang Dong[2], Wenquan Bi[1], Keting Jia[3], Xiaoyun Wang[1,2] and Willi Meier[4]

[1] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, P. R. China,
{lizhengcn,biwenquan}@mail.sdu.edu.cn

[2] Institute for Advanced Study, Tsinghua University, P. R. China

[3] Department of Computer Science and Technology, Tsinghua University, P. R. China
{xiaoyangdong,ktjia,xiaoyunwang}@tsinghua.edu.cn

[4] FHNW, Switzerland
willi.meier@fhnw.ch

**Abstract.** Conditional cube attack on round-reduced Keccak keyed modes was proposed by Huang *et al.* at EUROCRYPT 2017. In their attack, a conditional cube variable was introduced, whose diffusion was significantly reduced by certain key bit conditions. Then, a set of cube variables were found, that were not multiplied after the first round, meanwhile, the conditional cube variable was not multiplied with other cube variables (called ordinary cube variables) after the second round. This has an impact on the degree of the output of Keccak and hence gives a distinguisher. Later, MILP method was applied to find ordinary cube variables. However, for some Keccak based versions with low degrees of freedom, one could not find enough ordinary cube variables, which weakens or even invalidates the conditional cube attack.

In this paper, a new conditional cube attack on Keccak is proposed. We remove the limitation that no cube variables multiply with each other in the first round. As a result, some quadratic terms may appear after the first round. We make use of some new bit conditions to prevent the quadratic terms from multiplying with other cube variables in the second round, so that there will be no cubic terms after the second round. Furthermore, we introduce the *kernel quadratic term* and construct a *6-2-2 pattern* to reduce the diffusion of quadratic terms significantly, where the $\theta$ operation even in the second round becomes an identity transformation (CP-kernel property) for the *kernel quadratic term.* Previous conditional cube attacks on Keccak only explored the CP-kernel property of $\theta$ operation in the first round. Therefore, more degrees of freedom are available for ordinary cube variables and fewer bit conditions are used to remove the cubic terms after the second round, which plays a key role in the conditional cube attack on versions with very low degrees of freedom. We also use MILP method in the search of cube variables and give key-recovery attacks on round-reduced Keccak keyed modes.

As a result, we reduce the time complexity of key-recovery attacks on 7-round Keccak-MAC-512, and 7-round Ketje Sr v2 from $2^{111}$, $2^{99}$ to $2^{72}$, $2^{77}$, respectively. Additionally, we have reduced the time complexity of attacks on 9-round KMAC256 and 7-round Ketje Sr v1. Besides, practical attacks on 6-round Ketje Sr v1 and v2 are also given in this paper for the first time.

**Keywords:** Conditional Cube Attack, Keccak, KMAC, Ketje, MILP

# 1 Introduction

Keccak [BDPVA09], designed by Bertoni *et al.*, has been selected as the new cryptographic hash function standard SHA-3. As one of the most important cryptographic standards, Keccak attracts a lot of attention from world wide researchers and engineers. Till now, many cryptanalysis results [BCC11, DDS12, DDS13, DGPW12, GLS16, JN15, MPS13] and evaluation tools [DA12, DEM15, MDA17] have been proposed, including the recent impressive collision attacks [QSLG17, SLG17].

Besides, some important keyed constructions based on Keccak-$p$ permutations are proposed, including Keccak-MAC [BDPA11], KMAC [KCP16], Keyak [BDP+16b], Ketje [BDP+16a] and Kravatte [BDH+17], *et al.* At EUROCRYPT 2015, Dinur *et al.* [DMP+15] for the first time considered the security of the Keccak keyed modes utilizing cube-attack-like cryptanalysis and give some key-recovery attacks on reduced-round Keccak-MAC and Keyak. At CT-RSA 2015, Dobraunig *et al.* [DEMS15] evaluated the security of Ascon [DEMS16] against the cube-attack-like cryptanalysis. Later, this method was improved by Dong *et al.* [DLWQ17], Bi *et al.* [BDL+18] and Song *et al.* [SG18].

Conditional cube attack on round-reduced Keccak keyed modes was proposed by Huang *et al.* [HWX+17] at EUROCRYPT 2017. Firstly, they select a conditional cube variable. Bit conditions are added to reduce the diffusion of the conditional cube variable significantly. Thus, they could find a set of cube variables, that are not multiplied after the first round, meanwhile, the conditional cube variable is not multiplied with other cube variables (called ordinary cube variables) after the second round. The key bit conditions lead to a key-recovery attack. Later, Li *et al.* [LDW17] applied conditional cube attack to Ascon. At ASIACRYPT 2017, Li *et al.* first invented a MILP model based method to improve the conditional cube attack, [LBDW17]. Later, the MILP model was improved by Song *et al.* [SGSL18] at ASIACRYPT 2018. However, as shown in previous works [LBDW17, SGSL18, DLWQ17], for some Keccak based versions with very small degrees of freedom, one could not find enough ordinary cube variables, which weakens or even invalidates the conditional cube attack.

In this paper, we put forward a new conditional cube attack on Keccak keyed modes. We use the 7-round attack on Keccak to explain our new ideas:

1. We remove the limitation that no cube variables multiply with each other in the first round. In Huang *et al.*'s attack [HWX+17], the selected 64 cube variables must not be multiplied together in the first round. Therefore, the expected degree of the output of the 7-th round Keccak-$p$ permutation is $2^6 = 64$ (note that the degree of Keccak-$p$'s round function is 2). However, they find a conditional cube variable that does not multiply with other cube variables under suitable bit conditions. So, the expected degree under correct conditions will be 63, rather than 64. This gives the distinguisher.

   In our new attack, different from previous conditional cube attacks, we select 65 cube variables, two of which will be multiplied together to generate quadratic terms after the first round. Therefore, the degree of the output of the 7-th round Keccak-$p$ permutation will be $2^6 + 1 = 65$ (this will be proved in Section 4). However, by restraining some bit conditions, the quadratic terms of cube variables will not be multiplied with other cube variables in the 2nd round. Therefore, the degree of the output of 2nd round Keccak-$p$ is still 2. So, the expected output degree of the 7-th round under correct conditions will be 64, rather than 65. This gives the new distinguisher.

2. In Huang *et al.*'s conditional cube attack [HWX+17], the so-called 2-2-22 pattern is used to select a conditional cube variable. In their work, the conditional cube variable

$v_0$ is in the CP-kernel in the first round, that means, it occupies 2 bit positions in the same column of the state before $\theta$. Therefore $\theta$ in the first round becomes an identity transformation for $v_0$. With correct bit conditions, $v_0$ still occupies 2 bit positions in the output state of $\chi$ in the first round. After $\theta$ in the second round, the positions of $v_0$ will spread to 22 bits.

Then they select ordinary cube variables that do not multiply together after the first round, and do not multiply with $v_0$ after the second round. As shown in previous works [LBDW17, SGSL18, DLWQ17], it is hard to find enough ordinary cube variables that do not multiply with $v_0$ after the second round for Keccak versions with low degrees of freedom. A natural idea to improve the previous works is to reduce the $v_0$'s bit positions. In [LBDW17, BLD+18], the 6-6-6 pattern is used to select $v_0$. However, for Keccak-MAC-512 and Ketje Sr, such 6-6-6 pattern does not exist (the 6-6-6 pattern's bit positions are occupied by the key bits or padding bits and could not be selected as variables).

Things become different in our attack. In the second round, we only care that the quadratic terms (e.g. $v_0v_1$) of cube variables should not be multiplied with other cube variables. So what we have to do is to reduce the number of bit positions of $v_0v_1$.

3. In order to make $v_0v_1$ as sparse as possible, we introduce the so-called *kernel quadratic term*, which follows a new *6-2-2 pattern*. In our attack, $v_0v_1$ appears in only 2 bit positions after the first round. Furthermore, the 2 bit positions are in the same column and follow the CP-kernel property even in the $\theta$ operation of the second round. Therefore, the number of bit positions of $v_0v_1$ are still 2 before $\chi$ operation in the second round. Thus, we could find enough other cube variables that do not multiply with $v_0v_1$ in the second round with ease and could perform new key-recovery attacks on versions with low degrees of freedom, like Keccak-MAC-512 and Ketje Sr v2.

The results are summarized in Table 1. Based on our new conditional cube attack, the time complexity of key-recovery attacks on 7-round Keccak-MAC-512, 7-round Ketje Sr v2, and 9-round KMAC256 is reduced from $2^{111}$, $2^{99}$ and $2^{147}$ to $2^{72}$, $2^{77}$ and $2^{139}$, respectively. We also give the first practical attacks on 6-round Ketje Sr v1 and v2.

**Organization.** Section 2 gives some notations, a brief description of Keccak-MAC, Ketje and KMAC. In Section 3, we briefly describe the related works. The model of new conditional cube attack is introduced in Section 4, and its applications to Keccak-MAC, Ketje and KMAC are provided in Section 5. At last, we conclude this paper in Section 6.

## 2 Preliminaries

### 2.1 Notations

| | |
|---|---|
| $S_0$ | the initial state of Keccak-$p$ permutation, |
| $S_{i-1,\theta}$ | the internal state after $\theta$ in $i$-th round of Keccak-$p$, $i \geq 1$, |
| $S_{i-1,\pi}$ | the internal state after $\pi$ in $i$-th round, $i \geq 1$, |
| $S_i$ | the output state of the $i$-th round, $i \geq 1$, |
| | thus the internal states of $i$-th round are as follows: |

$$S_{i-1} \xrightarrow{\theta} S_{i-1,\theta} \xrightarrow{\rho} S_{i-1,\rho} \xrightarrow{\pi} S_{i-1,\pi} \xrightarrow{\chi} S_{i-1,\chi} \rightarrow S_i. \tag{1}$$

Table 1: Summary of Key-recovery Attacks on Keccak Keyed Modes

| Variant | Key Size | Capacity | Method | Rounds | Time | Source |
|---|---|---|---|---|---|---|
| KMAC256 | 256 | 512 | conditional† | 9 | $2^{147}$ | [SGSL18] |
| | | | conditional | 9 | $2^{139}$ | Subsection 5.3 |
| Keccak-MAC | 128 | 1024 | conditional | 6 | $2^{58}$ | [LBDW17] |
| | | | conditional | 6 | $2^{40}$ | [SGSL18] |
| | | | cube-like | 7 | $2^{112.6}$ | [BDL+18] |
| | | | cube-like | 7 | $2^{111}$ | [SG18] |
| | | | conditional | 7 | $2^{72}$ | Subsection 5.1 |
| Ketje Sr v1 | 128 | – | cube-like | 6 | $2^{73}$ | [DLWQ17] |
| | | | conditional | 6 | $2^{40.6}$ | Section 5.2.1 |
| | | | cube-like | 7 | $2^{115.32}$ | [DLWQ17] |
| | | | conditional | 7 | $2^{91}$ | [SGSL18] |
| | | | conditional | 7 | $2^{75}$ | Section 5.2.1 |
| Ketje Sr v2 | 128 | – | cube-like | 6 | $2^{65.6}$ | [DLWQ17] |
| | | | conditional | 6 | $2^{40.6}$ | Section 5.2.2 |
| | | | cube-like | 7 | $2^{113.58}$ | [DLWQ17] |
| | | | cube-like | 7 | $2^{99}$ | [SG18] |
| | | | conditional | 7 | $2^{77}$ | Section 5.2.2 |

†: in this table, “conditional“ is short for conditional cube attack, and “cube-like“ means cube-attack-like method.

| | |
|---|---|
| *lanesize* | the word size of one lane, |
| $(*, j, k)$ | the index of row, $0 \leq j \leq 4$, $0 \leq k \leq lanesize - 1$, |
| $(i, *, k)$ | the index of column, $0 \leq i \leq 4$, $0 \leq k \leq lanesize - 1$, |
| $(i, j, *)$ | the index of lane, $0 \leq i, j \leq 4$, |
| $(i, j, k)$ | the index of bit, $0 \leq i, j \leq 4$, $0 \leq k \leq lanesize - 1$, |
| $A[i][j]$ | the lane indexed by $(i, j, *)$ of state $A$, $0 \leq i, j \leq 4$, |
| $A[i][j][k]$ | the bit indexed by $(i, j, k)$ of state $A$, $0 \leq i, j \leq 4$, $0 \leq k \leq lanesize - 1$. |

## 2.2 The Keccak-$p$ permutations

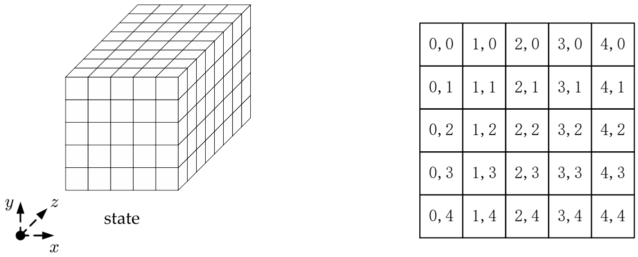
The Keccak-$p$ permutations are derived from the Keccak-$f$ permutations [BDPVA09] and have a tunable number of rounds. A Keccak-$p$ permutation is defined by its width $b = 25 \times 2^l$, with $b \in \{25, 50, 100, 200, 400, 800, 1600\}$, and its number of rounds $n_r$, denoted as Keccak-$p[b, n_r]$. The round function $\mathtt{R}$ consists of five operations, denoted as $\mathtt{R} = \iota \circ \chi \circ \pi \circ \rho \circ \theta$, and the details are as follows:

$$\begin{aligned}
&\theta : A[x][y] = A[x][y] \oplus \textstyle\sum_{j=0}^{4} (A[x-1][j] \oplus (A[x+1][j] \lll 1)).\\
&\rho : A[x][y] = A[x][y] \lll \rho[x, y].\\
&\pi : A[y][2x + 3y] = A[x][y].\\
&\chi : A[x][y] = A[x][y] \oplus ((\neg A[x+1][y]) \wedge A[x+2][y].\\
&\iota : A[0][0] = A[0][0] \oplus RC.
\end{aligned}$$

Keccak-$p[b, n_r]$ works on a state $A$ of size $b$, which can be represented as $5 \times 5$ $\frac{b}{25}$-bit lanes, as depicted in Figure 1, $A[i][j]$ with $i$ for the index of column and $j$ for the index of row. In what follows, indexes of $i$ and $j$ are in set $\{0, 1, 2, 3, 4\}$ and they are working in modulo 5 without other specification.

## 2.3 Keccak-MAC

Based on the sponge construction [BDPVA07], Keccak[$c$] uses Keccak-$p[1600, 24]$ as the internal permutation. The sponge construction has two parameters, the capacity $c$



| | | | | |
|---|---|---|---|---|
| 0, 0 | 1, 0 | 2, 0 | 3, 0 | 4, 0 |
| 0, 1 | 1, 1 | 2, 1 | 3, 1 | 4, 1 |
| 0, 2 | 1, 2 | 2, 2 | 3, 2 | 4, 2 |
| 0, 3 | 1, 3 | 2, 3 | 3, 3 | 4, 3 |
| 0, 4 | 1, 4 | 2, 4 | 3, 4 | 4, 4 |

Figure 1: (a) The Keccak State [BDPVA09], (b) A Slice of State $A$

and bit rate $r$. The state of Keccak is initialized to 0. Then a message $M$ is input and a digest is output. The message $M$ is split into $r$-bit blocks. Keccak processes the blocks iteratively by absorbing them into the first $r$ bits of the state and the permutation of Keccak-$p[1600, 24]$. Keccak-MAC [BDPA11] is a MAC form of Keccak where the input is the concatenation of key and message. The key size is assumed to be 128 bits. Keccak-MAC-512 applies Keccak[1024] as the internal permutation.

## 2.4 Ketje

Ketje [BDP+16a] is a submission by the Keccak team. It is a sponge-like construction. For Ketje v1, two instances are proposed, Ketje Sr and Jr with 400-bit and 200-bit state sizes, respectively. In the latest Ketje v2, another two instances Ketje Minor and Major are added to the family, with 800-bit and 1600-bit state sizes, respectively. Ketje Sr is the primary recommendation. The four concrete instances of Ketje v2 are shown in Table 2. In the following, we give a brief overview on Ketje v2. For a complete description, we refer to the design document [BDP+16a].

The structure of Ketje is an authenticated encryption mode MonkeyWrap, which is based on MonkeyDuplex [BDPA11]. It consists of four parts: initialization, processing associated data, processing the plaintext, finalization. Figure 2 illustrates the scheme of Ketje v2, where the finalization is omitted. In Ketje v2, *the twisted permutations*, Keccak-$p^*[b]$=$\pi\circ$ Keccak-$p[b] \circ \pi^{-1}$, are introduced to effectively re-order the bits in the state. $\pi^{-1} : A[x + 3y][x] = A[x][y]$ is the inverse of $\pi$, shown in Figure 3. Specially, $f_0$=Keccak-$p^*[b, 12]$ and $f_1$=Keccak-$p^*[b, 1]$.

Table 2: Four Instances of Ketje v2

| Name | $f$ | $\rho$ | Main use case |
|---|---|---|---|
| Ketje Jr | Keccak-$p^*[200]$ | 16 | lightweight |
| Ketje Sr | Keccak-$p^*[400]$ | 32 | lightweight |
| Ketje Minor | Keccak-$p^*[800]$ | 128 | lightweight |
| Ketje Major | Keccak-$p^*[1600]$ | 256 | high performance |

## 2.5 KMAC

KMAC (Keccak Message Authentication Code) [KCP16] is a keyed hash function, whose output length is variable. It has two variants: KMAC128 and KMAC256 with capacities set as 256 and 512 bits, respectively. They are actually based on Keccak$[c = 256](M, L)$ and Keccak$[c = 512](M, L)$. KMAC is initialized by the key $K$, the main message $M$, the output length $L$, the name string $N$ = "KMAC" and the optional customization bit string $S$ of any length (including 0).

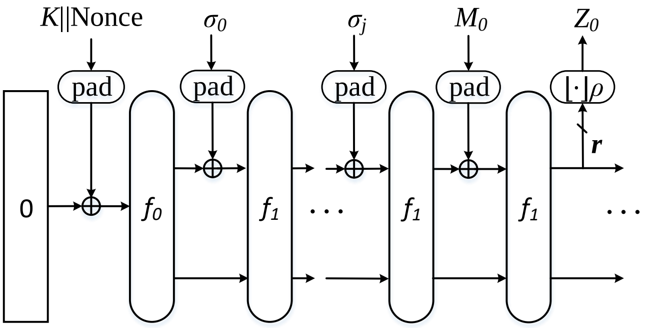
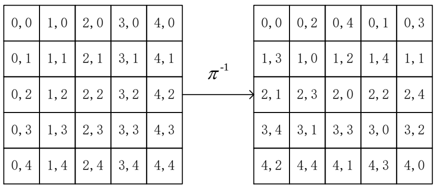


Figure 2: Ketje, where the finalization is omitted [SGSL18].



| | | | | |
|---|---|---|---|---|
| 0, 0 | 1, 0 | 2, 0 | 3, 0 | 4, 0 |
| 0, 1 | 1, 1 | 2, 1 | 3, 1 | 4, 1 |
| 0, 2 | 1, 2 | 2, 2 | 3, 2 | 4, 2 |
| 0, 3 | 1, 3 | 2, 3 | 3, 3 | 4, 3 |
| 0, 4 | 1, 4 | 2, 4 | 3, 4 | 4, 4 |

$\xrightarrow{\pi^{-1}}$

| | | | | |
|---|---|---|---|---|
| 0, 0 | 0, 2 | 0, 4 | 0, 1 | 0, 3 |
| 1, 3 | 1, 0 | 1, 2 | 1, 4 | 1, 1 |
| 2, 1 | 2, 3 | 2, 0 | 2, 2 | 2, 4 |
| 3, 4 | 3, 1 | 3, 3 | 3, 0 | 3, 2 |
| 4, 2 | 4, 4 | 4, 1 | 4, 3 | 4, 0 |

Figure 3: $\pi^{-1}$

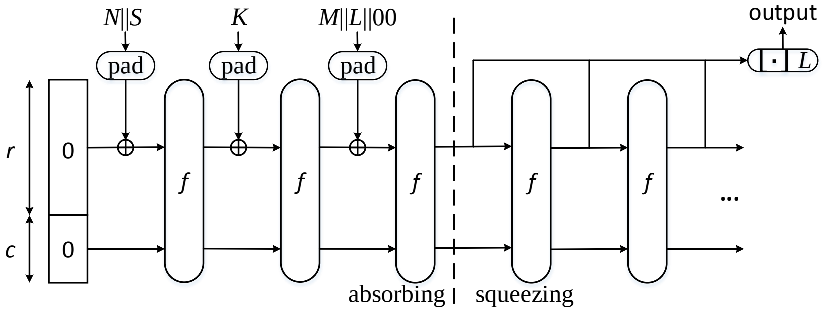


Figure 4: Construction of `KMAC`

Figure 4 illustrates the process of one message block. Actually, we treat the state before padding of the message as the initial state in our attack. Thus the structure is similar to the above two ciphers, and the whole secret 1600-bit state is treated as an equivalent of the key to be recovered. The output length of `KMAC` is variable, while that of `KMAC128` is no less than 256 bits and `KMAC256` is no less than 512 bits.

# 3 Related Work

## 3.1 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 a polynomial over $GF(2)$.

**Theorem 1.** *([DS09]) Let $f(k_0, ..., k_{n-1}, v_0, ..., v_{m-1})$ be a polynomial over $GF(2)$, where $k_0, ..., k_{n-1}$ are secret variables, and $v_0, ..., v_{m-1}$ are public variables.*

*For a set $I = \{i_1, i_2, ..., i_{|I|}\} \subset \{0, ..., m-1\}$, $f(k_0, ..., k_{n-1}, v_0, ..., v_{m-1})$ can be represented uniquely as*

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

*where $T_I = v_{i_1} \cdots v_{i_{|I|}}$. The polynomial $P$ only relates to $v_s's$ $(s \notin I)$ and the secret variables, and $Q(k_0, ..., k_{n-1}, v_0, ..., v_{m-1})$ misses at least one variable in $T_I$. $T_I$ is called maxterm and $P$ is called superpoly.*

*Denote by $C_I$ the structure, called cube, consisting of all $2^{|I|}$ different vectors with $v_i, i \in I$ being active (traversing all 0-1 combinations) and non-cube indices $v_s, s \notin I$ being static constants.*

*Then the sum of $f$ over all values of the cube $C_I$ (cube sum) is*

$$\sum_{v_{i_1}, ..., v_{i_{|I|}} \in C_I} f(k_0, ..., k_{n-1}, v_0, ..., v_{m-1}) = P. \tag{3}$$

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

### 3.2 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 $P$: $P = P_1 \times P_2 + P_3$ where the degree of $P_3$ is relatively lower than that of $P$, and $P_1$ contains a linear public term called a dynamic variable. A dynamic variable is a variable assigned with a function in some key bits and cube variables, and is used to zeroize $P_1$. Thus, $P$ is simplified to $P_3$. One must firstly guess these key bits to compute a dynamic variable. The right guess of key bits will lead to zero cube sums with high probability, otherwise the cube sums will be random.

### 3.3 Conditional Differential Cryptanalysis

Knellwolf, Meier and Naya-Plasencia [KMN10] applied conditional differential characteristic to NFSR-based constructions and extended it to higher order differential attacks at ASIACRYPT 2010. The input of a synchronous stream cipher is an initial value (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 in the IV through the first few rounds of the initialization process. Taking IV pairs conforming to these conditions as input, the resulting keystream differences will present a bias. Additionally, conditions upon key define classes of weak keys.

### 3.4 Conditional Cube Attack

Conditional cube attack [HWX+17] was proposed by Huang *et al.* at EUROCRYPT 2017 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 the initial value (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.

**Definition 1.** ([HWX+17]) Cube variables that have propagation controlled in the first round and are not multiplied with each other after the second round of Keccak are called **conditional cube variables**. Cube variables that are not multiplied with each other after the first round and are not multiplied with any conditional cube variable after the second round are called **ordinary cube variables**.

**Theorem 2.** *([HWX+17]) For $(n+2)$-round* Keccak *sponge function $(n > 0)$, if there are $p$ $(0 \leq p < 2^n+1)$ conditional cube variables $v_0, ..., v_{p-1}$, and $q = 2^{n+1} - 2p + 1$ ordinary cube variables, $u_0, ..., u_{q-1}$ (If $q = 0$, we set $p = 2^n + 1$), the term $v_0v_1...v_{p-1}u_0...u_{q-1}$ will not appear in the output polynomials of $(n+2)$-round* Keccak *sponge function.*

Actually, in the previous conditional cube attacks [HWX+17, LBDW17, SGSL18], they only use the special case of the above theorem when $p = 1$. We describe it as a corollary for clearness.

**Corollary 1.** *For $(n+2)$-round* Keccak *sponge function $(n > 0)$, if there is one conditional cube variable $v_0$, and $q = 2^{n+1} - 1$ ordinary cube variables, $u_0, ..., u_{q-1}$, the term $v_0u_0...u_{q-1}$ will not appear in the output polynomials of $(n+2)$-round* Keccak *sponge function.*

### 3.5 MILP Model of Conditional Cube Attack

Recently, the cryptographic community found that many classical cryptanalysis methods could be converted to mathematical optimization problems which aim to achieve the minimal or maximal value of an objective function under certain constraints. Mixed-integer Linear Programming (MILP) is the most widely studied technique to solve these optimization problems. One of the most successful applications of MILP is to search differential and linear trails. Mouha *et al.* [MWGP11] and Wu *et al.* [WW11] first applied MILP method to count active Sboxes of word-based block ciphers. Then, at ASIACRYPT 2014, by deriving some linear inequalities through the H-Representation of the convex hull of all differential patterns of Sbox, Sun *et al.* [SHW+14] extended this technique to search differential and linear trails. Another two important applications are to search integral distinguisher [XZBL16] and impossible differentials [ST17, CJF+16].

At ASIACRYPT 2017, Li *et al.* [LBDW17] for the first time applied MILP method to cube attacks on keyed Keccak. Later, the MILP model was improved by [SGSL18], and also applied to cube-attack-like method by Bi *et al.* [BDL+18] and Song *et al.* [SG18]. In the previous MILP model of conditional cube attack, no cube variables multiply with each other in the first round, and the conditional cube variable of degree one is considered not to multiply with any ordinary cube variables in the second round. To limit the diffusion of conditional cube variables, conditions are added in the first round. Actually, as described in Corollary 1, $(2^{n+1} - 1)$ ordinary cube variables are needed to perform a $(n+2)$-round attack. To obtain enough ordinary cube variables for the attack, the objective of Li *et al.*'s MILP model [LBDW17] at ASIACRYPT 2017 was to maximize the number of ordinary cube variables. To reduce the attack complexity further, Song *et al.* [SGSL18] proposed a new MILP model to minimize the number of conditions at ASIACRYPT 2018. We list the core relations among the MILP intermediate variables in Table 4, Table 5 of Appendix B.

## 4 New Conditional Cube Attack

In Huang *et al.*'s work [HWX+17] according to Corollary 1, conditional cube attack is performed when all the cube variables do not multiply in 1st round and one conditional cube variable does not multiply with others in 2nd round under certain bit conditions. For $(n+2)$-round attacks, the $2^{n+1}$-dimension cube sum is zero when the bit conditions are satisfied.

Different from Huang *et al.*'s work [HWX+17], we remove the limitation that all the cube variables do not multiply in 1st round. Actually, only two cube variables $v_0$, $v_1$

multiply in 1st round, i.e. $v_0v_1$ is the unique quadratic term after 1st round. Under certain bit conditions, the unique quadratic term $v_0v_1$ does not multiply with any other cube variable in 2nd round. In the output of 2nd round, the bit conditions make any cubic term containing $v_0v_1$ disappear. For $(n+2)$-round attacks, the $(2^{n+1}+1)$-dimension cube sum is zero when the bit conditions are satisfied. To give a detailed description, we give the following definition first.

**Definition 2.** Suppose all the $(q+2)$ cube variables are $v_0, v_1, u_0, ...u_{q-1}$, and constraints are as follows:

- After the first round, $v_0v_1$ is the only quadratic term;
- In the second round, if the bit conditions are satisfied, $v_0v_1$ does not multiply with any of $u_0, ...u_{q-1}$, i.e. no cubic term occurs.
- In the second round, if the bit conditions are not satisfied, $v_0v_1$ multiplies with some of $u_0, ...u_{q-1}$, i.e. some cubic terms like $v_0v_1u_i$ $(i = 0, ..., q-1)$ occur.

Then $v_0v_1$ is called **kernel quadratic term**. The remaining cube variables except $v_0$ and $v_1$, i.e. $u_0, ...u_{q-1}$, are called **ordinary cube variables**.

Then we describe the new conditional cube attack using **kernel quadratic term** in the following corollary.

**Corollary 2.** *For $(n+2)$-round* Keccak *sponge function $(n > 0)$, if there is one kernel quadratic term $v_0v_1$, and $q = 2^{n+1} - 1$ ordinary cube variables, $u_0, u_1, ..., u_{q-1}$, the term $v_0v_1u_0u_1...u_{q-1}$ will not appear in the output polynomials of $(n+2)$-round* Keccak *sponge function under certain bit conditions.*

*Proof.* Note that the round function of Keccak is two, so the upper bound output degree of $(n+2)$-round Keccak is $2^{n+2}$. However, we only selected $q+2$ cube variables, the highest possible output degree of $(n+2)$-round is $q+2 = 2^{n+1}+1 \ll 2^{n+2}$.

According to Definition 2, if the bit conditions are satisfied, the degree of the output polynomial after 2nd round of Keccak is two. So the output degree of $n$+2 rounds is no more than $2^{n+1}$. So the term $v_0v_1u_0u_1...u_{q-1}$ with degree $q+2 = 2^{n+1}+1$ will not appear in the output polynomials of $(n+2)$-round Keccak sponge function under such bit conditions. □

So the distinguisher is that:

- under right bit conditions, the degree of output polynomials of $n$+2 rounds is no more than $2^{n+1}$;
- under the wrong bit conditions, the degree of output polynomials of $n$+2 rounds is $q+2 = 2^{n+1}+1$.

A key question is whether the output degree of $n$+2 rounds could reach $2^{n+1}+1$ under wrong bit conditions. Our experiments in **supplementary material A** on 6-round Ketje Sr v2 show that the output degree could always reach the highest possible degree, i.e. $q+2 = 2^{4+1}+1 = 33$. This thanks to the very good diffusion of Keccak round function. We assume that the 7-round Keccak has a similar property to 6-round Keccak and the distinguisher also holds in the attack on 7-round Keccak. In fact, all previous conditional cube attacks [HWX$^+$17, LDW17, LBDW17, SGSL18] share similar assumptions.

As for the search of cube variables, we list the code of searching the 33-dimension cube satisfying the constraints in Definition 2 for Ketje Sr v2 in **supplementary material B**, as well as the verification code and the code of key-recovery attack on 6-round Ketje Sr v1 and v2 is also given, while the details of experiments are in Subsection 5.2.

## 4.1 New 6-2-2 Pattern

In the Keccak submission document [BDPVA09], the authors introduce the CP-kernel as follows: if all columns in a state have even parity, $\theta$ is the identity. Suppose that $A[i][j][k]$ is variable. If $\bigoplus_{j=0}^{4} A[i][j][k] = 0$, $\theta$ is also a identity for variables in the column $(i, *, k)$. For example, if $A[i][0][k] = v_1, A[i][1][k] = v_2, A[i][2][k] = v_3, A[i][3][k] = v_4, A[i][4][k] = v_1 \oplus v_2 \oplus v_3 \oplus v_4$, $\theta$ acts as a identity for variables in the column $(i, *, k)$ owing to $\bigoplus_{j=0}^{4} A[i][j][k] = 0$.

It is vital for the attack to find enough ordinary cube variables do not multiply with the conditional cube variable in 2nd round. A conditional cube variable with more sparse distribution will provide more degrees of freedom to find ordinary cube variables.

In Huang *et al.*'s work [HWX$^+$17], as the left part of Figure 5 shows, one *conditional cube variable* $v_0$ is placed in two black bits of $S_0$, i.e. $S_0[2][0][0] = S_0[2][1][0] = v_0$, which is exactly set in CP-kernel. After adding some conditions, the conditional cube variable $v_0$ maintains only 2 active bits after the 1st round (i.e. $S_1$) and diffuses to 22 bits after the linear part $\theta, \rho, \pi$ of the 2nd round as shown in state $S_{1,\pi}$. The diffusion pattern is denoted as 2-2-22. This pattern provides a nice restriction on the diffusion of the conditional cube variable $v_0$. Actually, 2-2-22 patten is chosen by all the previous conditional cube attacks for Keccak keyed modes [HWX$^+$17, LBDW17, SGSL18].

In our new conditional cube attack, we only care that the *kernel quadratic term* (i.e. $v_0v_1$) should not be multiplied with ordinary cube variables in the second round. Therefore, in order to get more degrees of freedom to find ordinary cube variables, we have to reduce the diffusion of $v_0v_1$. Hence, we introduce the so-called *6-2-2 pattern* shown in the right part of Figure 5. In the initial state $S_0$, $v_0$ occupies 4 black bits of $S_0$, i.e. $S_0[2][0][0] = S_0[2][1][0] = S_0[3][0][34] = S_0[3][1][34] = v_0$, which are also set in CP-kernel, and $v_1$ occupies the other 2 grey bits, i.e. $S_0[0][1][60] = S_0[1][1][1] = v_1$.

In $\chi$ operation of the 1st round, the *kernel quadratic term* $v_0v_1$ is generated, as shown in Figure 6. Before $\chi$ operation, $v_0$ (initialized in CP-kernel) only appears in the 4 black bits, while $v_1$ appears in 22 grey bits. Only 2 S-boxes highlighted by red rectangles are related to $v_0$ and $v_1$ simultaneously. According to the expression of $\chi$ operation: $A[x][y] = A[x][y] \oplus ((\neg A[x+1][y]) \wedge A[x+2][y]$, the quadratic term $v_0v_1$ is just generated in two bits in black slashes after $\chi$ operation.

The bit positions of $v_0v_1$ are also shown in $S_1$ (the output state of the 1st round) in the right part of Figure 5. Considering the quadratic term $v_0v_1$, the two bits in $S_1$ are in CP-kernel, thus the $\theta$ operation in the 2nd round is the identity regarding $v_0v_1$. Then the second $\rho$ and $\pi$ just permute the positions of these two bits as shown in $S_{1,\pi}$ of Figure 5.

In Corollary 1 of Huang *et al.*'s attacks, to perform an attack on a $(n+2)$-round Keccak sponge function $(n > 0)$, finding $q = 2^{n+1} - 1$ other cube variables that do not multiply with $v_0$ even in the second round is the basis of conditional cube attack. While the basis of new conditional cube attack shown in Corollary 2 is quite similar, it is to find $q = 2^{n+1} - 1$ ordinary cube variables that are not multiplied with $v_0v_1$ even in the second round. Thus, through $S_{1,\pi}$ shown in both parts of Figure 5, it is obvious that the search for ordinary cube variables that do not multiply with $v_0v_1$ in 6-2-2 pattern is much easier than that with $v_0$ in 2-2-22 pattern. As a result, we achieve attacks on 7-round Keccak-MAC-512, i.e., one more round than the previous conditional cube attacks.

**The method to find 6-2-2 pattern.** As shown in Figure 5, $v_0v_1$ satisfies CP-kernel in $S_1$. Denote the bit positions of $v_0v_1$ as $(x, y_0, z)$, $(x, y_1, z)$. According to the expression of $\chi$, $v_0v_1$ in $S_1[x][y_0][z]$ is generated by multiplying $v_0$ in $S_{0,\pi}[x+1][y_0][z]$ and $v_1$ in $S_{0,\pi}[x+2][y_0][z]$, or $v_0$ in $S_{0,\pi}[x+2][y_0][z]$ and $v_1$ in $S_{0,\pi}[x+1][y_0][z]$. The same happens to $v_0v_1$ in $S_1[x][y_1][z]$. So there will be 4 cases to determine the bit positions for $v_0$ and $v_1$ in reverse. For example, in Figure 7, $v_0v_1$ appears in $S_1[1][2][z]$ by multiplication of $v_0$ in $S_{0,\pi}[2][2][z]$ and $v_1$ in $S_{0,\pi}[3][2][z]$, and similarly $v_0v_1$ appears in $S_1[1][4][z]$ by multiplication of $v_1$ in $S_{0,\pi}[2][4][z]$ and $v_0$ in $S_{0,\pi}[3][4][z]$.

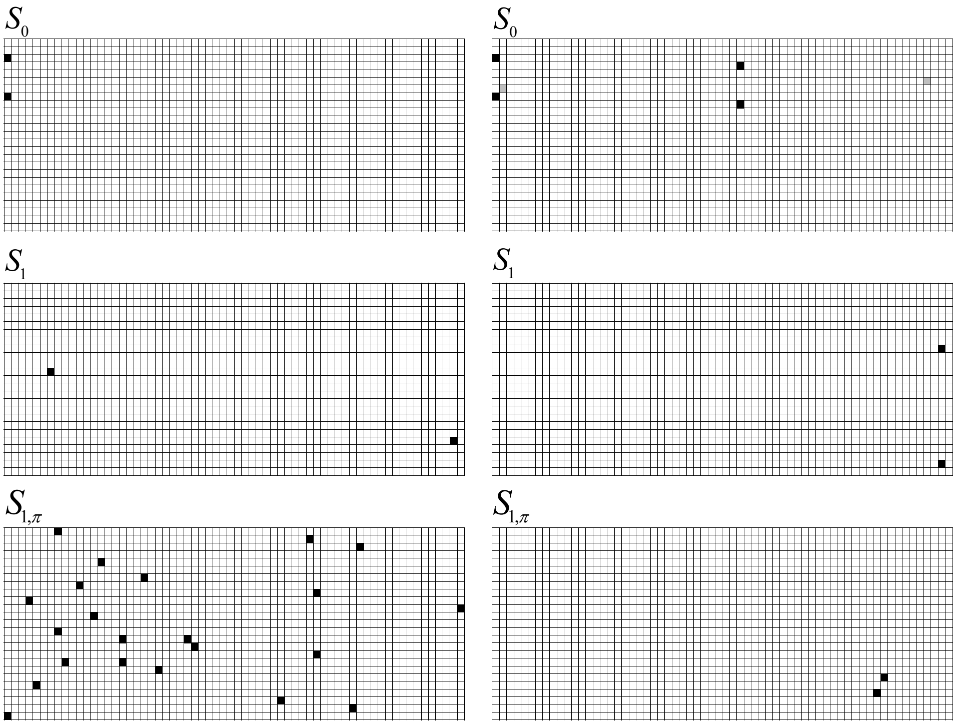


Figure 5: Diffusions of the Conditional Cube Variable in 2-2-22 and 6-2-2 Pattern in Keccak-MAC-512

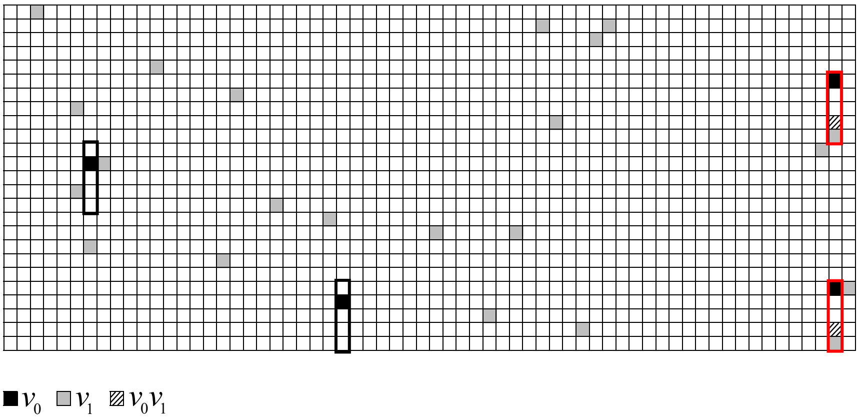


Figure 6: 6-2-2 Pattern: Generation of Kernel Quadratic Terms in the First $\chi$

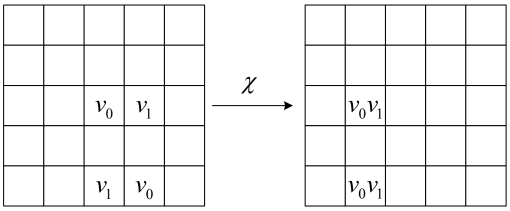


Figure 7: Slice $(*,*,z)$: from $S_{0,\pi}$ to $S_1$

Table 3: Related Indexes of Bits Containing $v_0$, $v_1$ and $v_0v_1$

| Index | $v_0v_1$ | |
|---|---|---|
| $S_1$ | $(x,y_0,z)$<br>$(x,y_1,z)$ | |
| Index | $v_0$ | $v_1$ |
| $S_{0,\pi}$ | $(x+1,y_0,z)$<br>$(x+1,y_1,z)$ | $(x+2,y_0,z)$<br>$(x+2,y_1,z)$ |
| $S_{0,\rho}$ | $(x+3y_0+1,x+1,z)$<br>$(x+3y_1+1,x+1,z)$ | $(x+3y_0+2,x+2,z)$<br>$(x+3y_1+2,x+2,z)$ |
| $S_{0,\theta}$ | $(x+3y_0+1,x+1,z-\rho[x+3y_0+1,x+1])$<br>$(x+3y_1+1,x+1,z-\rho[x+3y_1+1,x+1])$ | $(x+3y_0+2,x+2,z-\rho[x+3y_0+2,x+2])$<br>$(x+3y_1+2,x+2,z-\rho[x+3y_1+2,x+2])$ |
| $S_0$ | $(x+3y_0+1,x+1,z-\rho[x+3y_0+1,x+1])$<br>$\boldsymbol{(x+3y_0+1,x_1,z-\rho[x+3y_0+1,x+1])}$<br>$(x+3y_1+1,x+1,z-\rho[x+3y_1+1,x+1])$<br>$\boldsymbol{(x+3y_1+1,x_2,z-\rho[x+3y_1+1,x+1])}$ | $(x+3y_0+2,x+2,z-\rho[x+3y_0+2,x+2])$<br><br>$(x+3y_1+2,x+2,z-\rho[x+3y_1+2,x+2])$ |

Under one of the 4 cases, Table 3 describes the bit positions of $v_0v_1$, $v_0$ and $v_1$ inversely from $S_1$ to $S_{0,\theta}$, while the other cases are similar. In Table 3, in order to reduce the diffusion of $v_0$, the 4 bit positions containing $v_0$ in $S_0$ are set in CP-kernel, where $x_1, x_2 \neq x+1$. At last, all the 6 bits in $S_0$ should be selected in free space for ordinary cube variables. Accordingly, we can determine 6-2-2 patterns.

## 4.2 MILP Model of New Conditional Cube Attack

The 6-2-2 pattern has already determined the bit positions of $v_0$ and $v_1$ in the initial state, as well as the bit positions of *kernel quadratic term* $v_0v_1$ in the output of the first round. To perform an attack on a $(n+2)$-round Keccak sponge function $(n>0)$, the remaining $2^{n+1}-1$ ordinary cube variables and related conditions should be found by the following MILP model.

The notations of the corresponding state in the model are just similar to the previous ones used in [LBDW17, SGSL18]. We list the involved symbols here: Boolean variable $a[x][y][z]=1$ means initial state bit $S_0[x][y][z]=A[x][y][z]$ contains at least one ordinary cube variable; Boolean variable $b[x][y][z]=1$ means that of state bit $S_{0,\pi}[x][y][z]$, and Boolean variable $c[x][y][z]$ means that of state bit $S_1[x][y][z]$. Another two Boolean variables $v[x][y][z]$ and $h[x][y][z]$ are used to determine bit conditions:

- If $v[x][y][z]=1$, bit condition "$S_{0,\pi}[x][y][z]=h[x][y][z]$" is added to the attack;
- If $v[x][y][z]=0$, there are no bit conditions about $S_{0,\pi}[x][y][z]$.

Note that the sum of $v[x][y][z]$ means the number of bit conditions needed in our attack. If the bit condition "$S_{0,\pi}[x][y][z]=h[x][y][z]$" is related to a key bit, we need to guess it to assign a correct bit condition. In order to reduce the time complexity, we need to minimize the number of bit conditions that are related to key bits. Thus, the aim of the MILP model is to minimize the objective function:

$$\sum_{(S_{0,\pi}[x][y][z]/k)\neq 0} v[x][y][z],$$

where $k$ represents any key bit in $K$, and $(S_{0,\pi}[x][y][z]/k) \neq 0$ means at least one key bit appears in $S_{0,\pi}[x][y][z]$. Thus the sum is exactly the number of key bits to be guessed when assigning correct bit conditions.

### Modeling the First Round

When constraining the properties of the first $\theta$ operation, we use similar constraints as in [SGSL18]. $F[x][z]$ and $G[x][z]$ represent properties of column $(x,*,z)$. $G[x][z]=1$ means that the sum of 5 bits in column $(x,*,z)$ of the initial state $A$ is nonzero, while $G[x][z]=0$ means that the sum is zero. $F[x][z]=1$ represents the case that some variables exist in column $(x,*,z)$ but the sum of 5 bits in the column is zero, while $F[x][z]=0$ represents all the other cases. The corresponding system of inequalities are listed in Equation 4 in Appendix B.

However, we illustrate the following constraints different from the previous works. In our model, $v_0$ and $v_1$ are supposed to multiply with each other in 1st round. In addition, constraints should be added to prevent $v_0$ or $v_1$ from multiplying by ordinary cube variables. Take the 6-2-2 pattern in Table 3 as an example: in $S_0$, 4 bits contain $v_0$, 2 bits contain $v_1$. After $\theta$, $\rho$, $\pi$ operations, $v_0$ appears in 4 bits of $S_{0,\pi}$, and $v_1$ occupies at most 22 bits of $S_{0,\pi}$. Let set $\mathbb{T}$ contain the above bit positions in $S_{0,\pi}$, $|\mathbb{T}| \leq 26$. According to the $\chi$ operation, only neighbouring bits in the same S-box will be multiplied together. Thus, for any $(x,y,z) \in \mathbb{T}$, add constraints $b[x-1][y][z] = b[x+1][y][z] = 0$.

For the other bits in $S_{0,\pi}$ instead of the above cases, the constraints should be added as in the previous MILP model. We list the corresponding values and constraints in Table 4 and Equation 5 in Appendix B.

### Modeling the Second Round

In the input state of $\chi$ operation in the second round (i.e. $S_{1,\pi}$), 2 bits contain the **kernel quadratic term** $v_0v_1$. According to the expression of $\chi$ operation, only two adjacent bits multiply with each other. Thus some ordinary cube variables in the 4 bits next to the 2 bits containing $v_0v_1$ in $S_{1,\pi}$ should satisfy the following rules:

1. If the bit conditions are satisfied, no ordinary cube variables out of $u_0, ...u_{q-1}$ multiply with $v_0v_1$ in the second round.

2. If the bit conditions are not satisfied, some ordinary cube variables out of $u_0, ...u_{q-1}$ multiply with $v_0v_1$ in the second round.

Even though $v_0$ or $v_1$ multiply with the terms containing $v_0v_1$, the degree of corresponding terms containing $v_0v_1$ remains 2. So we omit the distribution of single $v_0$ and $v_1$ in the second round.

If a bit $S_0[x][y][z]$ contains any ordinary cube variables, $A[x][y][z] = 1$, otherwise $A[x][y][z] = 0$. However, for a set of more than one bit, dummy variable is put forward. If any bit in the set contains any ordinary cube variables, the corresponding dummy variable equals 1, otherwise it equals 0. The 4 bit positions in $S_{1,\pi}$ next to the 2 bits containing $v_0v_1$ are derived from $S_{1,\theta}$ through bit shift. The corresponding 4 bit positions in $S_{1,\theta}$ are denoted as $S_{1,\theta}[x_i][y_i][z_i], i = 0, 1, 2, 3$. We bring in dummy variables $e[i],\ i = 0, 1, 2, 3$, to

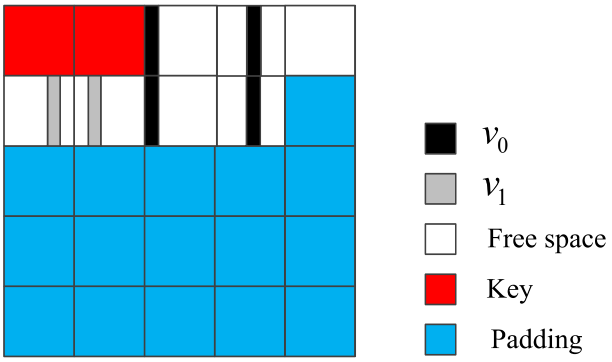


Figure 8: The Initial State of Keccak-MAC-512

avoid unexpected propagations for the 4 bits, which is similar to [SGSL18]. According to the expression of $\theta$ operation,

$$S_{1,\theta}[x_i][y_i][z_i] = S_1[x_i][y_i][z_i] + \sum_{y=0,1,2,3} S_1[x_i+4][y][z_i] + \sum_{y=0,1,2,3} S_1[x_i+1][y][z_i-1].$$

Corresponding to 11 bits above in $S_1$, if at least one of 11-bit $c$ equals 1 (note that Boolean variable $c[x][y][z] = 1$ indicates state bit $S_1[x][y][z]$ contains at least one ordinary cube variable), $e[i] = 1$. In this case, one linear equation (the XOR of the 11 bits equals a constant, i.e. 0 or 1) should be satisfied, thus it consumes one bit degree of freedom. Therefore, when computing the number of cube variables, subtraction of $\sum_{i=0}^{3} e[i]$ is needed. If all of the 11-bit $c$ equal 0, $e[i] = 0$.

The details of restrictions and values are listed in Equation 6 and Table 5 of Appendix B. Similar to [SGSL18], to obtain enough ordinary cube variables, the following constraint is also added in our model:

$$\sum_{x,y,z} A[x][y][z] - \sum_{x,z} F[x][z] - \sum_{i=0,1,2,3} e[i] = 2^{n+1} - 1.$$

# 5 Applications on Round-Reduced Keccak Keyed Modes

## 5.1 Attack on 7-round Keccak-MAC-512

For Keccak-MAC-512, a 7-round attack can be performed with 65 cube variables. In the whole 1600-bit state, rate occupies 576 bits, and capacity 1024 bits. Suppose $v_0v_1$ is the *kernel quadratic term*. As Figure 8 shows, the 128-bit key $K$ is located at the first 2 red lanes, and $v_0$ is set in CP-kernel as $S_0[2][0][0] = S_0[2][1][0] = S_0[3][0][34] = S_0[3][1][34] = v_0$ in 4 black bits, while $v_1$ is located at 2 grey bits, i.e. $S_0[0][1][60] = S_0[1][1][1] = v_1$. The white part represents free space for ordinary cube variables, i.e. nonce bits which can be selected as ordinary cube variables, while the padding is in blue. According to the new conditional cube attack illustrated in Section 4, we search for the minimal number of key bit conditions and ordinary cube variables satisfying the corresponding rules.

Suppose the state $S_0$ is initialized by symbols, where key bits are $k_0, k_1 ... k_{127}$, the following 448 bits are $n_0, n_1 ... n_{447}$, and each padding bit is 0 or 1. Through symbolic operation, each bit of $S_{0,\pi}$ is a polynomial. Take $S_{0,\pi}[x][y][z]$ as an example, considering its coefficient of $k_i$ for $i = 0, 1 ... 127$, if a certain coefficient is nonzero, $S_{0,\pi}[x][y][z]$ is related to some key bits. We list the lanes in $S_{0,\pi}$ whose bits are related to some key bits in a set

$\mathbb{S}$:

$$\begin{aligned}\mathbb{S} = \{&(0,0,*),(1,0,*),(2,0,*),(4,0,*),(1,1,*),\\ &(2,1,*),(3,1,*),(4,1,*),(0,2,*),(1,2,*),\\ &(3,2,*),(4,2,*),(0,3,*),(1,3,*),(2,3,*),\\ &(3,3,*),(0,4,*),(2,4,*),(3,4,*),(4,4,*)\}.\end{aligned}$$

The objective function to be minimized is

$$\sum_{(x,y)\in\mathbb{S}} v[x][y][z],$$

As at least one key bit condition is needed, the following constraint should be added:

$$\sum_{(x,y)\in\mathbb{S}} v[x][y][z] \geq 1.$$

The ordinary cube variables are denoted as $u_0, u_1...u_{62}$, and the number of ordinary cube variables is 63. The following equation is the constraint for enough ordinary cube variables:

$$\sum_{x,y,z} A[x][y][z] - \sum_{x,z} F[x][z] - \sum_{i=0,1,2,3} e[i] = 63.$$

The other constraints illustrated in Section 4 should be added. With the help of Gurobi [Gur], the objective function is optimized under all the above constraints, i.e. with 63 ordinary cube variables, the minimal number of key bit conditions is 1. Actually, $v_0$ and $v_1$ determined by 6-2-2 pattern, and $u_0, u_1...u_{62}$, total 65 cube variables are enough to perform the 7-round attack on Keccak-MAC-512. Both the cube variables and conditions are listed in Table 6 in Appendix C.

In 7-round attack on Keccak-MAC-512, $2^6 + 1 = 65$ cube variables are denoted by $v_0, v_1, u_0, u_1...u_{62}$. Based on Corollary 2, $v_0$ and $v_1$ are elements of the kernel quadratic term fixed in the beginning and $u_0, u_1...u_{62}$ are ordinary cube variables found by MILP search strategy. We summarize the requirements as follows:

(1) Only $v_0$ and $v_1$ multiply out of $v_0, v_1, u_0, u_1...u_{62}$ in the first round;

(2) Under some conditions on key and nonce, $v_0v_1$ does not multiply with any of $u_0, u_1...u_{62}$ in the second round;

(3) Under the conditions of nonce in (2), if the key conditions are not satisfied, $v_0v_1$ multiplies with some $u_i$ $(i = 0, 1, ..., 62)$ in the second round.

According to the requirements above, the *kernel quadratic term* $v_0v_1$ is the unique quadratic term in the output of the first round (i.e. $S_1$). So in the output of the second round, the available degree is at most 3 and the corresponding term should contain $v_0v_1$ as a factor. As illustrated in the above requirements (2) and (3), under the conditions of nonce, all cubic terms disappear with the key conditions satisfied, while some cubic terms like $v_0v_1u_i$ $(i = 0, 1, ..., 62)$ appear without the key conditions. As the algebraic degree of the round function in Keccak-$p$ permutation is 2, the highest degree of terms in the output of the 7 round Keccak-MAC-512 (i.e. $S_7$) is $2^6$ with the bit conditions satisfied, while the highest degree of $S_7$ is $2^6 + 1 = 65$ in the case that the key bit conditions are not satisfied. Therefore, if the 65-dimension cube sums of 7-round output bits are 0, we conjecture that the key guess is correct with very high probability; if the term $v_0v_1u_0...u_{62}$ appears, the key guess is wrong.

While all the nonce bits are constant, all the bit conditions are satisfied if and only if all the key bits are guessed correctly. Thus, the key guess is conjectured to be correct when the cube sum over the 128-bit tag is zero, while parameters are set as Table 6 in Appendix C.

We analyze the time and data complexity of the 7-round attack on Keccak-MAC-512: with the parameter sets in Table 6, one guessed key bits $k_{33}$ can be recovered. The time complexity of one recovery is $2^1 \times 2^{65}$. Due to properties of the permutation, there is a complete symmetry in direction of the $z$-axis. Thus we can obtain corresponding parameters set with any rotation index $i$ $(0 \leq i < 64)$ in $z$-axis. Therefore, the guessed key bits rotated by $i$ bits, i.e., $k_{i+33}$ can be recovered. Through simple count, for $0 \leq i < 64$, 64 independent key bits (i.e. the whole first lane) out of 128 key bits can be recovered, 64 iterations consumes $64 \times 2^1 \times 2^{65}$ and the remaining 64 key bits are left to exhaustive search consuming $2^{64}$. Combining the two parts, the procedure consumes $64 \times 2^1 \times 2^{65} + 2^{64} = 2^{72}$ computations of 7-round Keccak-MAC-512, correspondingly $2^{72}$ $(message, tag)$ pairs are needed. After the procedure above, all the 128 bits in $k$ can be recovered. Therefore, both time and data complexity of the attack are $2^{72}$.

## 5.2 Attack on Round-Reduced Ketje Sr

For Ketje Sr with 400-bit state, the length of the key $K$ is 128, while the shortest padding occupies 18 bits. For the state size of only 400 rather than 1600 bits, we are able control the diffusion of $v_1$ by slightly tweaking the 6-2-2 pattern. So 4 bits containing $v_0$ are shown in black and 3 bits containing $v_1$ are shown in grey in Figure 9, Figure 10.

According to the new conditional cube attack illustrated in Section 4, similar to attacks in Subsection 5.1, we search for the minimal number of key bit conditions and ordinary cube variables satisfying the corresponding rules. As each bit in $S_{0,\pi}[x][y][z]$ is related to some key bits, the objective function to be minimized is

$$\sum_{x,y \in \{0,1\ldots4\}, z \in \{0,1\ldots63\}} v[x][y][z],$$

As at least one key bit condition is needed, the following constraint should be added:

$$\sum_{x,y \in \{0,1\ldots4\}, z \in \{0,1\ldots63\}} v[x][y][z] \geq 1.$$

In the 6-round attack, the number of ordinary cube variables is 31, thus we add the following constraint:

$$\sum_{x,y,z} A[x][y][z] - \sum_{x,z} F[x][z] - \sum_{i=0,1,2,3} e[i] = 31.$$

In the 7-round attack, the number of ordinary cube variables is 63, thus we add the following constraint:

$$\sum_{x,y,z} A[x][y][z] - \sum_{x,z} F[x][z] - \sum_{i=0,1,2,3} e[i] = 63.$$

The other constraints illustrated in Section 4 should be added. With the help of Gurobi [Gur], the objective function is optimized under all the above constraints.

### 5.2.1 Attack on Round-Reduced Ketje Sr v1

Suppose $v_0v_1$ is the *kernel quadratic term.* As Figure 9 shows us, the 128-bit key is located at the red parts, while the padding part is shown in blue. and $v_0$ is set in CP-kernel as $S_0[0][2][0] = S_0[0][4][0] = S_0[1][2][9] = S_0[1][4][9] = v_0$ in black, and $v_1$ is located at 3 grey bits, i.e. $S_0[1][3][6] = S_0[1][4][6] = S_0[2][3][4] = v_1$. The white bits represent free space to be selected as ordinary cube variables.

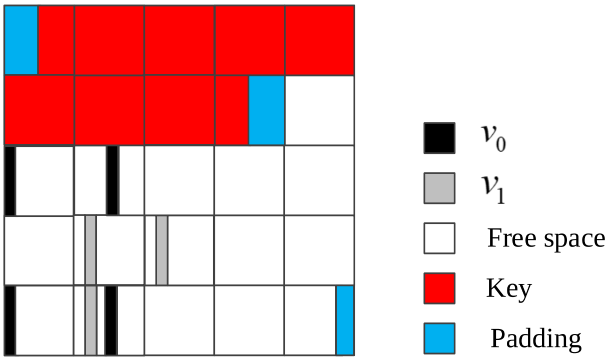


Figure 9: The Initial State of Ketje Sr v1

### Attack on 6-round Ketje Sr v1

With 31 ordinary cube variables, the minimal number of key bit conditions is 1, then a 6-round attack on Ketje Sr v1 can be performed. Both the cube variables and conditions are listed in Table 7 in Appendix C.

We analyze the time and data complexity of the attack: with the parameter sets in Table 7 in Appendix C, the guessed key bit $k_{39} + k_{70} + k_{119}$ can be recovered. The time complexity of one recovery is $2^1 \times 2^{33}$. According to the property of permutation, it is totally symmetric in $z$-axis. Thus we can obtain corresponding parameters set with any $i$-bit rotation $(0 \leq i < 64)$ in $z$-axis. Therefore, the guessed key bits rotated by $i$ bits, i.e. $k_{i+39} + k_{i+70} + k_{i+119}$ can be recovered. Actually, as different choices of 7-2-2 pattern with different $v_0$ and $v_1$ are possible, we have different choices for the predetermined parameters in the MILP model. Meanwhile, there are plenty of solutions of the MILP model with one set of predetermined parameters. Therefore, other sets of parameters related to different key bits are easily obtained, and we can recover 96 key bits. 96 iterations consume $96 \times 2^1 \times 2^{33}$ and the remaining 32 key bits are left to exhaustive search consuming $2^{32}$. Combining the two parts, the procedure consumes $2^{40.6}$ computations of 6-round Ketje Sr v1, accordingly $2^{40.6}$ $(message, tag)$ pairs are needed. After the procedure above, all the 128 bits in $K$ can be recovered. Therefore, both time and data complexity of the attack are $2^{40.6}$.

We give an example here for intuition, in which the key is generated randomly and all the controllable nonce bits are set to zero. Actually, all the 1000 experiments obtained correct key recovery of 6-round Ketje Sr v1. The program is run in Visual Studio 2010 with x64 platform Release. The time is about 6 hours for recovery of one key bit using one CPU core (Intel i7 3.6GHz), and parallelism can reduce time. The test code is provided in **supplementary material C**.

128-bit key $K$:

```
101000000110101100111010011011100011100100000111011101110010110110
1111110010011101001011000101010100010111101000111100101100000101
```

The correct value for the guessed key bit in Table 7 in Appendix C is `1`.

guessed value: `0`,

cube sums: `0xaa1b, 0x46d`

guessed value: `1`,

cube sums: `0x0, 0x0`

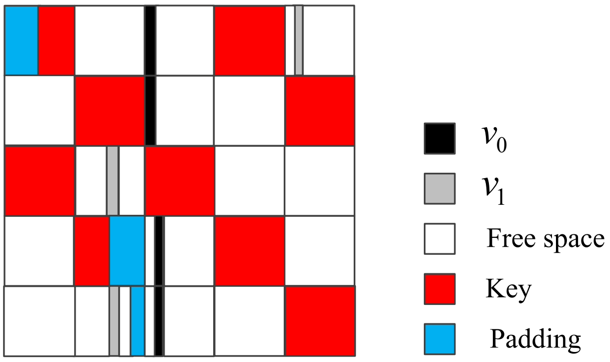


Figure 10: The State after $\pi^{-1}$ of Ketje Sr v2

### Attack on 7-round Ketje Sr v1

With 63 ordinary cube variables, the minimal number of key bit conditions is 7, then a 7-round attack on Ketje Sr v1 can be performed. Both the cube variables and conditions are listed in Table 8 in Appendix C. We analyze the time and data complexity of the attack: with the parameter sets in Table 8, the 7 guessed key bits $k_8 + k_{57} + k_{88}$, $k_{13} + k_{44} + k_{93} + k_{124}$, $k_{39} + k_{70} + k_{119}$, $k_{26} + k_{75} + k_{91} + k_{106}$, $k_{21} + k_{52} + k_{101}$, $k_{15} + k_{46} + k_{95} + k_{111} + k_{126}$, $k_9 + k_{40} + k_{89} + k_{120}$ can be recovered. The time complexity of one recovery is $2^7 \times 2^{65}$. According to the property of permutation, it is totally symmetric in $z$-axis. Thus we can obtain corresponding parameters set with any $i$-bit rotation $(0 \leq i < 64)$ in $z$-axis. Therefore, the guessed key bits rotated by $i$ bits can be recovered. Combining procedures of key recovery by related parameters and exhaustive search, the procedure consumes $2^{75}$ computations of 7-round Ketje Sr v1, correspondingly $2^{75}$ $(message, tag)$ pairs are needed. After the procedure above, all the 128 bits in $K$ can be recovered. Therefore, both time and data complexity of the attack are $2^{75}$.

### 5.2.2 Attack on Round-Reduced Ketje Sr v2

Suppose $v_0v_1$ is the *kernel quadratic term*. As Figure 10 shows us, the 128-bit key is located at the red parts, while the padding part is shown in blue. and $v_0$ is set in CP-kernel as $S_0[2][0][0] = S_0[2][1][0] = S_0[2][3][1] = S_0[2][4][1] = v_0$ in black, and $v_1$ is located at 3 grey bits, i.e. $S_0[1][2][12] = S_0[1][4][12] = S_0[4][0][1] = v_1$. The white bits represent nonce bits which can be selected as ordinary cube variables.

### Attack on 6-round Ketje Sr v2

With 31 ordinary cube variables, the minimal number of key bit conditions is 1, then a 6-round attack on Ketje Sr v2 can be performed. Both the cube variables and conditions are listed in Table 9 in Appendix C. We analyze the time and data complexity of the attack: with the parameters set in Table 9 in Appendix C, the guessed key bit $k_{34} + k_{65} + k_{82} + k_{97}$ can be recovered. The time complexity of one recovery is $2^1 \times 2^{33}$. According to the property of permutation, it is totally symmetric in $z$-axis. Thus we can obtain corresponding parameter sets with any $i$-bit rotation $(0 \leq i < 64)$ in $z$-axis. Therefore, the guessed key bits rotated by $i$ bits i.e. $k_{i+34} + k_{i+65} + k_{i+82} + k_{i+97}$ can be recovered. By a computation similar to Ketje Sr v1, both time and data complexity of the attack are $2^{40.6}$.

We give an example here for intuition, in which the key is generated randomly and all the controllable nonce bits are set to zero. Actually, all the 1000 experiments obtained

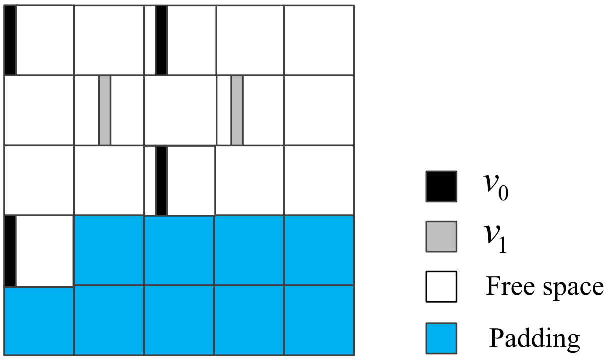


Figure 11: The State of `KMAC256`

correct key recovery of 6-round Ketje Sr v2. The program is run in Visual Studio 2010 with x64 platform Release. The time is about 6 hours for recovery of one key bit using one CPU core (Intel i7 3.6GHz), and parallelism can reduce time. The test code is given in **supplementary material C**.

128-bit key $K$:
`0111111100001001011011111010010011001100011011110111111111001000`
`0101010010011101110010110000110010110101110111011000111101000011`
The correct value for the guessed key bit in Table 9 in Appendix C is `1`.
guessed value: `0`,
cube sums: `0x3182, 0xbd7`
guessed value: `1`,
cube sums: `0x0, 0x0`

### Attack on 7-round Ketje Sr v2

With 63 ordinary cube variables, the minimal number of key bit conditions is 10, then a 7-round attack on Ketje Sr v2 can be performed. Both the cube variables and conditions are listed in Table 10 in Appendix C. We analyze the time and data complexity of the attack: with the parameters set in Table 10, the 10 guessed key bits $k_{28}+k_{109}$, $k_{27}+k_{108}$, $k_{39}+k_{104}$, $k_2+k_{33}+k_{114}$, $k_{39}+k_{70}+k_{87}+k_{102}$, $k_{25}+k_{56}+k_{88}$, $k_1+k_{32}+k_{113}$, $k_{26}+k_{107}$, $k_{28}+k_{59}+k_{91}$, $k_5+k_{36}+k_{117}$ can be recovered. The time complexity of one recovery is $2^{10}\times 2^{65}$. Therefore, the guessed key bits rotated by $i$ bits can be recovered. Through similar computation, both time and data complexity of the attack are $2^{77}$.

## 5.3 Attack on 9-round `KMAC256`

For `KMAC256`, a 9-round attack can be performed with 129 cube variables. The state of `KMAC256` is 1600-bit, the key size is 256-bit. The output length can be more than 320-bit, the first 5 lanes of $S_9$ can be inverted through the $\chi$ operation of the 9th round, i.e. the first 5 lanes of $S_{8,\pi}$ can be obtained. Due to the linearity of $\theta,\rho,\pi$, $S_{8,\pi}$ and $S_8$ share the same algebraic degree. Therefore, 129 cube variables can achieve 9-round attacks on `KMAC256`. The state just before the message is involved is the state to place cube variables. The whole state is secret while the length of free nonce involved is 1048. Suppose $v_0v_1$ is the *kernel quadratic term*. As Figure 11 shows us, $v_0$ is set in CP-kernel as $S_0[0][0][0]=S_0[0][3][0]=S_0[2][0][2]=S_0[2][2][2]=v_0$ in black, and $v_1$ is located at 2 grey bits, i.e. $S_0[1][1][20]=S_0[3][1][9]=v_1$, while the padding is in blue. The white bits provide free space for ordinary cube variables. According to the new conditional cube attack illustrated in Section 4, we search for the minimal number of key bit conditions and ordinary cube variables satisfying the corresponding rules. As each bit in $S_{0,\pi}[x][y][z]$ is related to some key bit, the objective function to be minimized is

$$\sum_{x,y\in\{0,1...4\},z\in\{0,1...63\}} v[x][y][z].$$

As at least one key bit condition is needed, the following constraint should be added:

$$\sum_{x,y\in\{0,1...4\},z\in\{0,1...63\}} v[x][y][z] \geq 1.$$

The ordinary cube variables are denoted as $u_0, u_1, ..., u_{126}$, and the number of ordinary cube variables is 127. The following equation is the constraint for enough ordinary cube variables:

$$\sum_{x,y,z} A[x][y][z] - \sum_{x,z} F[x][z] - \sum_{i=0,1,2,3} e[i] = 127.$$

The other constraints illustrated in Section 4 should be added. With the help of Gurobi [Gur], the objective function is optimized under all the above constraints, and the minimal number of key bit conditions is 1. Actually, $v_0$ and $v_1$ are determined by the 6-2-2 pattern, and $u_0, u_1, ..., u_{126}$, 129 cube variables are sufficient to perform the 9-round attack on `KMAC256`. Both the cube variables and conditions are listed in Table 11 in Appendix C. According to Corollary 2, $v_0 v_1 u_0 ... u_{126}$ appears without the key conditions; If the key bit condition is satisfied, term $v_0 v_1 u_0 ... u_{126}$ does not appear. Thus, with parameters set as in Table 11 in Appendix C, if the cube sum over the known bits of $S_{8,\pi}$ is zero, we conjecture that the key bit condition is satisfied, i.e. the key guess is conjectured to be correct.

We analyze the time and data complexity of the 9-round attack on `KMAC256`: with the parameter sets in Table 11, if the key bit equals 0 (i.e. $k_{14} + k_{207} + k_{334} + k_{527} + k_{654} + k_{847} + k_{911} + k_{974} + k_{1167} + k_{1294} + k_{1487} + 1 = 0$), the cube sums are zero; if the key bit equals 1 (i.e. $k_{14} + k_{207} + k_{334} + k_{527} + k_{654} + k_{847} + k_{911} + k_{974} + k_{1167} + k_{1294} + k_{1487} + 1 = 1$), the cube sums are nonzero. So the key bit $k_{14} + k_{207} + k_{334} + k_{527} + k_{654} + k_{847} + k_{911} + k_{974} + k_{1167} + k_{1294} + k_{1487}$ can be recovered. The time complexity of one recovery is $2^{129}$. According to the property of permutation, it is totally symmetric in $z$-axis. Thus we can obtain corresponding parameters set with any $i$-bit rotation ($0 \leq i < 64$) in $z$-axis. Therefore, the guessed key bits rotated $i$-bit can be recovered. Through simple count, for $0 \leq i < 64$, 64 independent key bits can be recovered, and 64 iterations consume $64 \times 2^{129}$. Different sets of parameters can be got similarly, it is owing to the many choices of 6-2-2 pattern with different $v_0$, $v_1$ and numerous solutions by the MILP model. We omit them here, and totally 1470 iterations consume $1470 \times 2^{129}$ to recover 1470 bits of the intermediate state. Then the remaining 130 state bits are left to exhaustive search consuming $2^{130}$. Combining the two parts, the procedure consumes $1470 \times 2^{129} + 2^{130} = 2^{139}$ computations of 9 rounds of `KMAC256`, correspondingly $2^{139}$ $(message, tag)$ pairs are needed. After the procedure above, all 1600 bits of the internal state can be recovered, so we can recover the initial state containing 256 key bits. Therefore, both time and data complexity of the attack are $2^{139}$.

# 6 Conclusion

In this paper, we introduce a new conditional cube attack on Keccak keyed modes. In our new attack, we let quadratic terms occur after the first round, which is different from previous conditional cube attacks. While in the second round, we add bit conditions so that the quadratic terms could not multiply with other cube variables. Thus, after the second round, there are no cubic terms. Furthermore, we introduce the *kernel quadratic term* and construct a *6-2-2 pattern*, which reduces the diffusion of quadratic terms significantly. The ideas above make make the operation $\theta$ in the 2nd round an identity transformation for the distribution of the kernel quadratic term, while the $\theta$ operation is the significant part providing diffusion. So more degrees of freedom are available to find enough ordinary cube variables, which is the key part to even validate the conditional cube attack on versions with low degrees of freedom. Furthermore, fewer bit conditions are used to make the cubic terms disappear. So we can perform key-recovery attacks. MILP method is also used in the search of ordinary cube variables.

Based on the above, we could attack 7-round Keccak-MAC-512, 7-round Ketje Sr v2 and 9-round KMAC256 with time complexity $2^{72}$, $2^{77}$, and $2^{139}$, respectively, which are faster than the best previous attack by a factor of $2^{39}$, $2^{22}$ and $2^{8}$. Focusing on conditional cube attack, we have improved the attack on both round-reduced Keccak-MAC-512 and Ketje Sr v2 by one round.

## References

[BCC11] Christina Boura, Anne Canteaut, and Christophe De Cannière. Higher-Order Differential Properties of Keccak and *Luffa*. In *Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers*, volume 6733 of *Lecture Notes in Computer Science*, pages 252–269, 2011.

[BDH+17] Guido Bertoni, Joan Daemen, Seth Hoffert, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. Farfalle: Parallel Permutation-Based Cryptography. *IACR Trans. Symmetric Cryptol.*, 2017(4):1–38, 2017.

[BDL+18] Wenquan Bi, Xiaoyang Dong, Zheng Li, Rui Zong, and Xiaoyun Wang. MILP-aided Cube-Attack-Like Cryptanalysis on Keccak Keyed Modes. *Designs, Codes and Cryptography*, pages 1–26, 2018.

[BDP+16a] Guido Bertoni, Joan Daemen, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. Ketje v2. *Submission to the CAESAR Competition*, 2016.

[BDP+16b] Guido Bertoni, Joan Daemen, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. Keyak v2. *Submission to the CAESAR Competition*, 2016.

[BDPA11] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications. In *Selected Areas in Cryptography - 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers*, volume 7118 of *Lecture Notes in Computer Science*, pages 320–337, 2011.

[BDPVA07] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Sponge Functions. In *ECRYPT hash workshop*, volume 2007. Citeseer, 2007.

[BDPVA09] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Keccak Sponge Function Family Main Document. *Submission to NIST (Round 2)*, 3(30), 2009.

[BLD+18] Wenquan Bi, Zheng Li, Xiaoyang Dong, Lu Li, and Xiaoyun Wang. Conditional Cube Attack on Round-Reduced River Keyak. *Des. Codes Cryptography*, 86(6):1295–1310, 2018.

[CJF+16] Tingting Cui, Keting Jia, Kai Fu, Shiyao Chen, and Meiqin Wang. New Automatic Search Tool for Impossible Differentials and Zero-Correlation Linear Approximations. *IACR Cryptology ePrint Archive*, 2016:689, 2016.

[DA12] Joan Daemen and Gilles Van Assche. Differential Propagation Analysis of Keccak. In *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, volume 7549 of *Lecture Notes in Computer Science*, pages 422–441, 2012.

[DDS12] Itai Dinur, Orr Dunkelman, and Adi Shamir. New Attacks on Keccak-224 and Keccak-256. In *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, volume 7549 of *Lecture Notes in Computer Science*, pages 442–461, 2012.

[DDS13] Itai Dinur, Orr Dunkelman, and Adi Shamir. Collision Attacks on Up to 5 Rounds of SHA-3 Using Generalized Internal Differentials. In *Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 11-13, 2013. Revised Selected Papers*, volume 8424 of *Lecture Notes in Computer Science*, pages 219–240, 2013.

[DEM15] Christoph Dobraunig, Maria Eichlseder, and Florian Mendel. Heuristic Tool for Linear Cryptanalysis with Applications to CAESAR Candidates. In *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II*, volume 9453 of *Lecture Notes in Computer Science*, pages 490–509, 2015.

[DEMS15] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Cryptanalysis of Ascon. In *Topics in Cryptology - CT-RSA 2015, The Cryptographer's Track at the RSA Conference 2015, San Francisco, CA, USA, April 20-24, 2015. Proceedings*, volume 9048 of *Lecture Notes in Computer Science*, pages 371–387, 2015.

[DEMS16] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1. 2. *Submission to the CAESAR Competition*, 2016.

[DGPW12] Alexandre Duc, Jian Guo, Thomas Peyrin, and Lei Wei. Unaligned Rebound Attack: Application to Keccak. In *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, volume 7549 of *Lecture Notes in Computer Science*, pages 402–421, 2012.

[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):259–280, 2017.

[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 *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 733–761, 2015.

[DS09] Itai Dinur and Adi Shamir. Cube Attacks on Tweakable Black Box Polynomials. In *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, 2009.

[DS11] Itai Dinur and Adi Shamir. Breaking Grain-128 with Dynamic Cube Attacks. In *Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers*, volume 6733 of *Lecture Notes in Computer Science*, pages 167–187, 2011.

[GLS16] Jian Guo, Meicheng Liu, and Ling Song. Linear Structures: Applications to Cryptanalysis of Round-Reduced KECCAK. In *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, volume 7549 of *Lecture Notes in Computer Science*, pages 249–274, 2016.

[Gur] http://www.gurobi.com/.

[HWX$^+$17] Senyang Huang, Xiaoyun Wang, Guangwu Xu, Meiqin Wang, and Jingyuan Zhao. Conditional Cube Attack on Reduced-Round KECCAK Sponge Function. In *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.

[JN15] Jérémy Jean and Ivica Nikolic. Internal Differential Boomerangs: Practical Analysis of the Round-Reduced KECCAK-$f$ Permutation. In *Fast Software Encryption - 22nd International Workshop, FSE 2015, Istanbul, Turkey, March 8-11, 2015, Revised Selected Papers*, volume 9054 of *Lecture Notes in Computer Science*, pages 537–556, 2015.

[KCP16] John Kelsey, Shu-jen Chang, and Ray Perlner. SHA-3 Derived Functions: cSHAKE, KMAC, TupleHash, and ParallelHash. *NIST special publication*, 800:185, 2016.

[KMN10] Simon Knellwolf, Willi Meier, and María Naya-Plasencia. Conditional Differential Cryptanalysis of NLFSR-Based Cryptosystems. In *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, volume 6477 of *Lecture Notes in Computer Science*, pages 130–145, 2010.

[LBDW17] Zheng Li, Wenquan Bi, Xiaoyang Dong, and Xiaoyun Wang. Improved Conditional Cube Attacks on KECCAK Keyed Modes with MILP Method. In *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I*, volume 10624 of *Lecture Notes in Computer Science*, pages 99–127, 2017.

[LDW17] Zheng Li, Xiaoyang Dong, and Xiaoyun Wang. Conditional Cube Attack on Round-Reduced ASCON. *IACR Trans. Symmetric Cryptol.*, 2017(1):175–202, 2017.

[MDA17] Silvia Mella, Joan Daemen, and Gilles Van Assche. New Techniques for Trail Bounds and Application to Differential Trails in KECCAK. *IACR Trans. Symmetric Cryptol.*, 2017(1):329–357, 2017.

[MPS13] Pawel Morawiecki, Josef Pieprzyk, and Marian Srebrny. Rotational Cryptanalysis of Round-Reduced KECCAK. In *Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 11-13, 2013. Revised*

*Selected Papers*, volume 8424 of *Lecture Notes in Computer Science*, pages 241–262, 2013.

[MWGP11] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and Linear Cryptanalysis Using Mixed-Integer Linear Programming. In *Information Security and Cryptology - 7th International Conference, Inscrypt 2011, Beijing, China, November 30 - December 3, 2011. Revised Selected Papers*, volume 7537 of *Lecture Notes in Computer Science*, pages 57–76, 2011.

[QSLG17] Kexin Qiao, Ling Song, Meicheng Liu, and Jian Guo. New Collision Attacks on Round-Reduced Keccak. In *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 III*, volume 10212 of *Lecture Notes in Computer Science*, pages 216–243, 2017.

[SG18] Ling Song and Jian Guo. Cube-Attack-Like Cryptanalysis of Round-Reduced Keccak Using MILP. *IACR Trans. Symmetric Cryptol.*, 2018(3):182–214, 2018.

[SGSL18] Ling Song, Jian Guo, Danping Shi, and San Ling. New MILP Modeling: Improved Conditional Cube Attacks on Keccak-Based Constructions. In *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part II*, volume 11273 of *Lecture Notes in Computer Science*, pages 65–95, 2018.

[SHW+14] Siwei Sun, Lei Hu, Peng Wang, Kexin Qiao, Xiaoshuang Ma, and Ling Song. Automatic Security Evaluation and (Related-Key) Differential Characteristic Search: Application to SIMON, PRESENT, LBlock, DES(L) and Other Bit-Oriented Block Ciphers. In *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, volume 8873 of *Lecture Notes in Computer Science*, pages 158–178, 2014.

[SLG17] Ling Song, Guohong Liao, and Jian Guo. Non-Full Sbox Linearization: Applications to Collision Attacks on Round-Reduced Keccak. In *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 428–451, 2017.

[ST17] Yu Sasaki and Yosuke Todo. New Impossible Differential Search Tool from Design and Cryptanalysis Aspects - Revealing Structural Properties of Several Ciphers. In *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 III*, volume 10212 of *Lecture Notes in Computer Science*, pages 185–215, 2017.

[WW11] Shengbao Wu and Mingsheng Wang. Security Evaluation against Differential Cryptanalysis for Block Cipher Structures. *IACR Cryptology ePrint Archive*, 2011:551, 2011.

[WY05] Xiaoyun Wang and Hongbo Yu. How to Break MD5 and Other Hash Functions. In Ronald Cramer, editor, *Advances in Cryptology - 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 *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, volume 3621 of *Lecture Notes in Computer Science*, pages 1–16, 2005.

[XZBL16] Zejun Xiang, Wentao Zhang, Zhenzhen Bao, and Dongdai Lin. Applying MILP Method to Searching Integral Distinguishers Based on Division Property for 6 Lightweight Block Ciphers. In *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, volume 10031 of *Lecture Notes in Computer Science*, pages 648–678, 2016.

# A Revised Parts According to the Comments

We thank the reviewers for their valuable comments and list the major parts revised accordingly.

For Reviewer 1,
Review: I also think it would be good to include a proof of concept of their technique by using it in a round-reduced version or so.
The theoretical proof of concept is described in lines 270 to 292 of Page 8 and Page 9; the outline of proof experiments are described in lines 293 to 308 of Page 9, the details of experiments on 6-round Ketje SR v1 and v2 are described in lines 542 to 555 of Page 17 and lines 587 to 600 of Page 18.

For Reviewer 2,
1.Review: - l253: "some of ordinary cube variables may undertake bit conditions"?
This part is rewritten to compare our work with Huang *et al*'s work in lines 257 to 269 in Page 8.

2.Review: - l465: "If the term does not appear, the key guess is correct". Why is this guaranteed?
This sentence is replaced by "if the 65-dimension cube sums of 7-round output bits are 0, we conjecture that the key guess is correct with very high probability" in lines 479 to 481 in Page 15. And the corresponding distinguisher and assumption are described in lines 293 to 308 of Page 9.

3.Review: "Please describe your practical results and experiments in full sentences and/or tables..."
The details of practical results experiments on 6-round Ketje SR v1 and v2 are described in lines 542 to 555 of Page 17 and lines 587 to 600 of Page 18.

# B Supplementary of Variables in MILP model: Values and Constraints for Some Cases

In this section, we list some values and corresponding constraints used in the MILP model here for a supplementary, as similar constraints are also used in [SGSL18], while they may be used for different scenes there. Equation 4 illustrates the constraints for ordinary cube variables (i.e. cube variables except $v_0$ and $v_1$) in $\theta$ operation of the first round.

$$\begin{cases} -F[x][z] - G[x][z] \geq \text{-1} \\ -A[x][0][z] + F[x][z] + G[x][z] \geq 0 \\ -A[x][1][z] + F[x][z] + G[x][z] \geq 0 \\ -A[x][2][z] + F[x][z] + G[x][z] \geq 0 \\ -A[x][3][z] + F[x][z] + G[x][z] \geq 0 \\ -A[x][4][z] + F[x][z] + G[x][z] \geq 0 \\ A[x][0][z] + A[x][1][z] + A[x][2][z] + A[x][3][z] + A[x][4][z] - 2F[x][z] - G[x][z] \geq 0 \end{cases} \tag{4}$$

We list the values of model variables for ordinary $\chi$ operation of the first round in Table 4, and corresponding constraints are listed in Equation 5.

Table 4: Summary of Variables for Ordinary $\chi$ Operation in the 1st round

| $B[x]$ | $B[x+1]$ | $B[x+2]$ | $V[x+1]$ | $V[x+2]$ | $H[x+1]$ | $H[x+2]$ | $C[x]$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | * | * | * | * | 0 |
| 1 | 0 | 0 | * | * | * | * | 1 |
| 0 | 0 | 1 | 0 | 0 | * | * | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | * | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | * | 1 |
| 0 | 1 | 0 | 0 | 0 | * | * | 1 |
| 0 | 1 | 0 | 0 | 1 | * | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | * | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | * | * | 1 |
| 1 | 0 | 1 | 1 | 0 | * | * | 1 |

$$\begin{cases} -B[x] - B[x+1] \geq \text{-1} \\ -B[x] + C[x] \geq 0 \\ -B[x+2] - V[x+2] \geq \text{-1} \\ -B[x+1] - V[x+1] \geq \text{-1} \\ -B[x] - B[x+1] - H[x+2] + C[x] \geq \text{-1} \\ B[x] - V[x+1] - H[x+1] - C[x] \geq \text{-2} \\ B[x] - V[x+2] + H[x+2] - C[x] \geq \text{-1} \\ B[x] + B[x+1] + B[x+2] - C[x] \geq 0 \\ -B[x+1] - B[x+2] + V[x+1] + V[x+2] + C[x] \geq 0 \\ -B[x+1] - B[x+2] + V[x+2] + H[x+1] + C[x] \geq 0 \end{cases} \tag{5}$$

As we bring in dummy variables $e[i], i = 0, 1, 2, 3$ to record and eliminate some uncertain propagation of cube variables, the possible values of corresponding model variables are listed in Table 5, and corresponding constraints are listed are in Equation 6.

$$\begin{cases} -e[i] - B[x+1][y][z] - B[x+2][y][z] \geq \text{-2} \\ -e[i] - B[x+1][y][z] + V[x+2][y][z] \geq \text{-1} \\ -e[i] - B[x+2][y][z] + V[x+1][y][z] \geq \text{-1} \\ -e[i] - B[x+1][y][z] - V[x+1][y][z] \geq \text{-2} \\ -e[i] - B[x+2][y][z] - V[x+2][y][z] \geq \text{-2} \\ -e[i] - B[x][y][z] - B[x+1][y][z] \geq \text{-2} \end{cases} \tag{6}$$

Table 5: Summary of Dummy Variables in the 2nd round

| $e[i]$ | $B[x][y][z]$ | $B[x+1][y][z]$ | $B[x+2][y][z]$ | $V[x+1][y][z]$ | $V[x+2][y][z]$ |
|---|---|---|---|---|---|
| 0 | * | * | * | * | * |
| 1 | 0 | 0 | 0 | * | * |
| 1 | 1 | 0 | 0 | * | * |
| 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 |

# C Parameters Sets for Attacks on Round-Reduced Keccak-MAC-512, Ketje Sr v1&v2 and KMAC256

In this section, we list the parameters sets for attacks on round-reduced Keccak-MAC-512, Ketje sr v1&v2 and KMAC256.

Table 6: Parameters set for attack on 7-round Keccak-MAC-512

| **kernel quadratic term** |
|---|
| $A[2][0][0]=A[2][1][0]=A[3][0][34]=A[3][1][34]=v_0$,$A[0][1][60]=A[1][1][1]=v_1$ |
| **Bit Condition** |
| $A[0][1][33]=k_{33}+A[1][1][33]$,$A[2][0][52]=A[4][0][51]+A[2][1][52]$,$A[4][0][0]=0$ |
| **Ordinary Cube Variables** |
| $A[0][1][44]=u_0$,$A[1][1][5]=u_1$,$A[1][1][43]=u_2$,$A[2][0][1]=A[2][1][1]=u_3$,<br>$A[2][0][2]=A[2][1][2]=u_4$,$A[2][0][3]=A[2][1][3]=u_5$,$A[2][0][4]=A[2][1][4]=u_6$,<br>$A[2][0][5]=A[2][1][5]=u_7$,$A[2][0][6]=A[2][1][6]=u_8$,$A[2][0][7]=A[2][1][7]=u_9$,<br>$A[2][0][9]=A[2][1][9]=u_{10}$,$A[2][0][10]=A[2][1][10]=u_{11}$,$A[2][0][11]=A[2][1][11]=u_{12}$,<br>$A[2][0][12]=A[2][1][12]=u_{13}$,$A[2][0][13]=A[2][1][13]=u_{14}$,$A[2][0][14]=A[2][1][14]=u_{15}$,<br>$A[2][0][15]=A[2][1][15]=A[3][0][33]=u_{16}$,$A[2][0][17]=A[2][1][17]=u_{17}$,<br>$A[2][0][18]=A[2][1][18]=u_{18}$,$A[2][0][19]=A[2][1][19]=u_{19}$,$A[2][0][20]=A[2][1][20]=u_{20}$,<br>$A[2][0][21]=A[2][1][21]=u_{21}$,$A[2][0][22]=A[2][1][22]=u_{22}$,$A[2][0][23]=A[2][1][23]=u_{23}$,<br>$A[2][0][24]=A[2][1][24]=u_{24}$,$A[2][0][25]=A[2][1][25]=u_{25}$,$A[2][0][26]=A[2][1][26]=u_{26}$,<br>$A[2][0][28]=A[2][1][28]=A[3][0][62]=A[3][1][62]=u_{27}$,$A[2][0][30]=A[2][1][30]=u_{28}$,<br>$A[2][0][31]=A[2][1][31]=u_{29}$, $A[2][0][32]=A[2][1][32]=u_{30}$,$A[2][0][34]=A[2][1][34]=u_{31}$,<br>$A[2][0][36]=A[2][1][36]=u_{32}$,$A[2][0][37]=A[2][1][37]=u_{33}$,$A[2][0][38]=A[2][1][38]=u_{34}$,<br>$A[2][0][41]=A[2][1][41]=u_{35}$,$A[2][0][42]=A[2][1][42]=u_{36}$,$A[2][0][43]=A[2][1][43]=u_{37}$,<br>$A[2][0][44]=A[2][1][44]=u_{38}$,$A[2][0][45]=A[2][1][45]=u_{39}$,$A[2][0][46]=A[2][1][46]=u_{40}$,<br>$A[2][0][50]=A[2][1][50]=u_{41}$,$A[2][0][51]=A[2][1][51]=u_{42}$,$A[2][0][54]=A[2][1][54]=u_{43}$,<br>$A[2][0][57]=A[2][1][57]=u_{44}$,$A[2][0][59]=A[2][1][59]=u_{45}$,$A[2][0][60]=A[2][1][60]=u_{46}$,<br>$A[2][0][61]=A[2][1][61]=u_{47}$,$A[2][0][62]=A[2][1][62]=u_{48}$,$A[2][0][63]=A[2][1][63]=u_{49}$,<br>$A[3][0][1]=A[3][1][1]=u_{50}$,$A[3][0][15]=A[3][1][15]=u_{51}$,$A[3][0][23]=A[3][1][23]=u_{52}$,<br>$A[3][0][60]=A[3][1][60]=u_{53}$,$A[3][0][36]=A[3][1][36]=u_{54}$,$A[3][0][40]=A[3][1][40]=u_{55}$,<br>$A[3][0][42]=A[3][1][42]=u_{56}$,$A[3][0][46]=A[3][1][46]=u_{57}$,$A[3][0][47]=A[3][1][47]=u_{58}$,<br>$A[3][0][54]=A[3][1][54]=u_{59}$,$A[3][0][55]=A[3][1][55]=u_{60}$,$A[3][0][56]=A[3][1][56]=u_{61}$,<br>$A[3][0][59]=A[3][1][59]=u_{62}$ |

Table 7: Parameters set for attack on 6-round Ketje SR v1

| **kernel quadratic term** |
|---|
| $A[0][2][0]=A[0][4][0]=A[1][2][9]=A[1][4][9]=v_0$,$A[1][3][6]=A[1][4][6]=A[2][3][4]=v_1$, |
| **Bit Condition** |
| $A[4][1][14]=k_{39}+k_{70}+k_{119}+n_{63}+n_{94}+n_{143}+n_{159}+n_{174}+n_{223}+1$ |
| **Ordinary Cube Variables** |
| $A[1][2][0]=A[1][3][0]=u_0$,$A[1][3][2]=A[1][4][2]=u_1$,$A[1][3][4]=A[1][4][4]=u_2$,<br>$A[1][2][7]=A[1][4][7]=u_3$,$A[2][2][7]=u_4$,$A[2][3][7]=u_5$,$A[2][4][7]=u_4+u_5$,<br>$A[2][2][9]=u_6$,$A[2][3][9]=u_7$,$A[2][4][9]=u_6+u_7$,$A[2][2][10]=u_8$,$A[2][3][10]=u_9$,<br>$A[2][4][10]=u_8+u_9$,$A[2][2][12]=A[2][3][12]=u_{10}$,$A[2][2][13]=A[2][4][13]=u_{11}$,<br>$A[4][2][0]=u_{12}$,$A[4][3][0]=u_{13}$,$A[4][4][0]=u_{12}+u_{13}$,$A[4][1][1]=u_{14}$,$A[4][2][1]=u_{15}$,<br>$A[4][3][1]=u_{16}$,$A[4][4][1]=u_{14}+u_{15}+u_{16}$,$A[4][1][2]=u_{17}$,$A[4][2][2]=u_{18}$,$A[4][3][2]=u_{19}$,<br>$A[4][4][2]=u_{17}+u_{18}+u_{19}$,$A[4][1][3]=u_{20}$,$A[4][2][3]=u_{21}$,$A[4][3][3]=u_{22}$,<br>$A[4][4][3]=u_{20}+u_{21}+u_{22}$,$A[4][1][6]=u_{23}$,$A[4][2][6]=u_{24}$,$A[4][4][6]=u_{23}+u_{24}$,<br>$A[4][1][13]=A[4][3][13]=u_{25}$,$A[4][2][8]=u_{26}$,$A[4][3][8]=u_{27}$,$A[4][4][8]=u_{26}+u_{27}$,<br>$A[4][2][9]=u_{28}$,$A[4][3][9]=u_{29}$,$A[4][4][9]=u_{28}+u_{29}$,$A[4][1][10]=u_{14}+u_{15}+u_{16}$,<br>$A[4][2][10]=u_{30}$,$A[4][4][10]=u_{14}+u_{15}+u_{16}+u_{30}$ |

Table 8: Parameters set for attack on 7-round KETJE SR v1

| **kernel quadratic term** |
|---|
| $A[0][2][0]{=}A[0][4][0]{=}A[1][2][9]{=}A[1][4][9]{=}v_0$,$A[1][3][6]{=}A[1][4][6]{=}A[2][3][4]{=}v_1$, |
| **Bit Condition** |
| $A[0][2][1]{=}k_8 + k_{57} + k_{88} + n_{32} + n_{112} + n_{192} + 1$<br>$A[2][3][5]{=}k_{13} + k_{44} + k_{93} + k_{124}$<br>$A[4][1][14]{=}k_{39} + k_{70} + k_{119} + n_{94} + n_{143} + n_{159} + n_{174} + 1$<br>$A[0][3[3]{=}k_{26} + k_{75} + k_{91} + k_{106} + n_{130} + 1$<br>$A[1][3[13]{=}k_{21} + k_{52} + k_{101} + n_{141}$<br>$A[1][4][7]{=}k_{15} + k_{46} + k_{95} + k_{111} + k_{126} + 1$<br>$A[1][2][1]{=}k_9 + k_{40} + k_{89} + k_{120}+ n_{49} + n_{64} + n_{113} + n_{193} + 1$ |
| **Ordinary Cube Variables** |
| $A[0][2][3]{=}A[0][4][3]{=}u_0$,$A[0][2][5]{=}A[0][4][5]{=}u_1$,$A[0][2][6]{=}A[0][4][6]{=}u_2$,<br>$A[0][2][8]{=}A[0][4][8]{=}u_3$,$A[0][3][9]{=}A[0][4][9]{=}u_4$,$A[0][2][10]{=}A[0][4][10]{=}u_5$,<br>$A[0][2][12]{=}u_6$,$A[0][3][12]{=}u_7$,$A[0][4][12]{=}u_6{+}u_7$,$A[0][2][13]{=}u_8$,$A[0][3][13]{=}u_9$,<br>$A[0][4][13]{=}u_8{+}u_9$,$A[0][2][14]{=}A[0][4][14]{=}u_{10}$,$A[1][2][2]{=}A[1][4][2]{=}u_{11}$,<br>$A[1][2][5]{=}u_{12}$,$A[1][3][5]{=}u_{13}$,$A[1][4][5]{=}u_{12}{+}u_{13}$,$A[1][2][7]{=}A[1][3][7]{=}u_{14}$,<br>$A[1][2][8]{=}u_{15}$,$A[1][3][8]{=}u_{16}$,$A[1][4][8]{=}u_{15}{+}u_{16}$,$A[1][2][10]{=}u_{17}$,<br>$A[1][3][10]{=}u_{18}$,$A[1][4][10]{=}u_{17}{+}u_{18}$,$A[1][2][11]{=}u_{19}$,<br>$A[1][4][11]{=}u_{20}$,$A[1][2][12]{=}A[1][4][12]{=}u_{21}$,$A[1][2][13]{=}A[1][4][13]{=}u_{22}$,<br>$A[1][2][14]{=}A[1][4][14]{=}u_{23}$,$A[1][2][15]{=}A[1][4][15]{=}u_{24}$,$A[2][3][1]{=}A[2][4][1]{=}u_{25}$,<br>$A[2][2][2]{=}A[2][4][2]{=}u_{26}$,$A[2][2][4]{=}u_{27}$,$A[2][4][4]{=}u_{28}$,$A[2][2][9]{=}A[2][4][9]{=}u_{29}$,<br>$A[2][2][11]{=}u_{30}$,$A[2][3][11]{=}u_{31}$,$A[2][4][11]{=}u_{30}{+}u_{31}$,$A[2][2][12]{=}u_{32}$,$A[2][3][12]{=}u_{33}$,<br>$A[2][4][12]{=}u_{32}{+}u_{33}$,$A[2][2][13]{=}A[2][4][13]{=}u_{34}$,$A[2][3][14]{=}A[2][4][14]{=}u_{35}$,<br>$A[2][2][15]{=}A[2][4][15]{=}u_{36}$,$A[3][3][0]{=}A[3][4][0]{=}u_{37}$,$A[3][2][1]{=}A[3][4][1]{=}u_{38}$,<br>$A[3][2][4]{=}u_{39}$,$A[3][3][4]{=}u_{40}$,$A[3][4][4]{=}u_{39}{+}u_{40}$,$A[3][2][6]{=}u_{41}$,$A[3][3][6]{=}u_{42}$,<br>$A[3][4][6]{=}u_{41}{+}u_{42}$,$A[3][2][7]{=}u_{43}$,$A[3][3][7]{=}u_{44}$,$A[3][4][7]{=}u_{43}{+}u_{44}$,<br>$A[3][2][9]{=}A[3][4][9]{=}u_{45}$,$A[3][2][12]{=}u_{46}$,$A[3][3][12]{=}u_{47}$,$A[3][4][12]{=}u_{46}{+}u_{47}$,<br>$A[3][2][13]{=}u_{48}$,$A[3][3][13]{=}u_4{+}u_{13}{+}u_{27}{+}u_{28}$,$A[3][4][13]{=}u_4{+}u_{13}{+}u_{27}{+}u_{28}{+}u_{48}$,<br>$A[3][2][14]{=}A[3][4][14]{=}u_{50}$,$A[3][2][15]{=}A[3][4][15]{=}u_{51}$,$A[4][2][0]{=}A[4][4][0]{=}u_{52}$,<br>$A[4][1][1]{=}u_{53}$,$A[4][2][1]{=}u_{54}$,$A[4][3][1]{=}u_{55}$,$A[4][4][1]{=}u_{53}{+}u_{54}{+}u_{55}$,<br>$A[4][2][2]{=}A[4][4][2]{=}u_{56}$,$A[4][1][3]{=}A[4][2][3]{=}u_{57}$,$A[4][1][6]{=}u_{58}$,$A[4][2][6]{=}u_{59}$,<br>$A[4][4][6]{=}u_{58}{+}u_{59}$,$A[4][1][10]{=}u_8{+}u_9{+}u_{22}{+}u_{26}{+}u_{43}{+}u_{44}{+}u_{53}{+}u_{54}{+}u_{55}$,<br>$A[4][2][10]{=}u_{60}$,$A[4][4][10]{=}u_8{+}u_9{+}u_{22}{+}u_{26}{+}u_{43}{+}u_{44}{+}u_{53}{+}u_{54}{+}u_{55}{+}u_{60}$,<br>$A[4][1][8]{=}u_{20}$,$A[4][2][8]{=}u_{61}$,$A[4][4][8]{=}u_{20}{+}u_{61}$,$A[4][2][9]{=}u_{62}$,$A[4][3][9]{=}u_{49}$,<br>$A[4][4][9]{=}u_{62}{+}u_{49}$ |

Table 9: Parameters set for attack on 6-round Ketje SR v2

| **kernel quadratic term** |
|---|
| $A[2][0][0]=A[2][1][0]=A[2][3][1]=A[2][4][1]=v_0$,$A[1][2][12]=A[1][4][12]=A[4][0][1]=v_1$ |
| **Bit Condition** |
| $A[2][4][10]=k_{34} + k_{65} + k_{82} + k_{97} + n_{42} + n_{73} + n_{105} + n_{154} + n_{186} + n_{217}$ |
| **Ordinary Cube Variables** |
| $A[0][1][0]=u_0$,$A[0][4][0]=u_0$,$A[0][1][1]=u_1$,$A[0][3][1]=u_2$,$A[0][4][1]=u_1+u_2$,<br>$A[0][1][7]=u_3$,$A[0][3][7]=u_4$,$A[0][4][7]=u_3+u_4$,$A[0][3][8]=u_5$,$A[0][4][8]=u_5$,<br>$A[0][1][10]=u_6$,$A[0][3][10]=u_7$,$A[0][4][10]=u_6+u_7$,$A[0][1][11]=u_8$,$A[0][3][11]=u_9$,<br>$A[0][4][11]=u_8+u_9$,$A[0][1][15]=u_{10}$,$A[0][3][15]=u_{11}$,$A[0][4][15]=u_{10}+u_{11}$,<br>$A[1][0][7]=u_{12}$,$A[1][4][7]=u_{12}$,$A[1][0][14]=u_{13}$,$A[1][2][14]=u_{13}$,$A[2][0][2]=u_{14}$,<br>$A[2][3][2]=u_{15}$,$A[2][4][2]=u_{14}+u_{15}$,$A[2][0][3]=u_{16}$,$A[2][1][3]=u_{17}$,$A[2][3][3]=u_{18}$,<br>$A[2][4][3]=u_{16}+u_{17}+u_{18}$,$A[2][0][4]=u_{19}$,$A[2][3][4]=u_{20}$,$A[2][4][4]=u_{19}+u_{20}$,<br>$A[2][0][6]=u_{21}$,$A[2][1][6]=u_{22}$,$A[2][3][6]=u_{23}$,$A[2][4][6]=u_{21}+u_{22}+u_{23}$,<br>$A[2][0][9]=u_{24}$,$A[2][4][9]=u_{24}$,$A[2][1][11]=u_{25}$,$A[2][4][11]=u_{25}$,$A[3][1][12]=u_{26}$,<br>$A[3][2][12]=u_{27}$,$A[3][4][12]=u_{26}+u_{27}$,$A[3][1][15]=u_{28}$,$A[3][2][15]=u_{28}$,$A[4][0][7]=u_{29}$,<br>$A[4][2][7]=u_{30}$,$A[4][3][7]=u_{29}+u_{30}$ |

Table 10: Parameters set for attack on 7-round Ketje SR v2

| **kernel quadratic term** |
|---|
| $A[2][0][0]$=$A[2][1][0]$=$A[2][3][1]$=$A[2][4][1]$=$v_0$,$A[1][2][12]$=$A[1][4][12]$=$A[4][0][1]$=$v_1$ |
| **Bit Condition** |
| $A[0][3][5]$=$k_{28} + k_{109} + n_{21}$,$A[1][4][4]$=$k_{27} + k_{108} + n_{116}$,$A[1][2][0]$=$k_{39} + k_{104}$, |
| $A[2][0][9]$=$k_2 + k_{33} + k_{114} + n_9 + n_{26} + n_{41} + n_{90} + n_{153}$ |
| $A[4][2][14]$=$k_{39} + k_{70} + k_{87} + k_{102} + n_{78} + n_{110}$ |
| $A[4][2][0]$=$k_{25} + k_{56} + k_{88} + n_{33} + n_{49} + n_{64} + n_{96} + n_{177}$ |
| $A[1][4][9]$=$k_1 + k_{32} + k_{113} + n_{152} + 1$ |
| $A[0][3][3]$=$k_{26} + k_{107} + n_{83} + n_{115} + n_{131}$ |
| $A[4][2][3]$=$k_{28} + k_{59} + k_{91} + n_{52} + n_{67} + n_{99}$ |
| $A[0][3][13]$=$k_5 + k_{36} + k_{117} + n_{29} + n_{93} + n_{125} + 1$ |
| **Ordinary Cube Variables** |
| $A[0][1][0]$=$u_0$,$A[0][3][0]$=$u_1$,$A[0][4][0]$=$u_0$+$u_1$,$A[0][1][1]$=$u_2$,$A[0][3][1]$=$u_3$, |
| $A[0][4][1]$=$u_2$+$u_3$,$A[0][3][4]$=$u_4$,$A[0][4][4]$=$u_4$,$A[0][1][5]$=$u_5$,$A[0][4][5]$=$u_5$, |
| $A[0][1][6]$=$u_6$,$A[0][4][6]$=$u_6$,$A[0][1][7]$=$u_7$,$A[0][3][7]$=$u_8$,$A[0][4][7]$=$u_7$+$u_8$, |
| $A[0][3][8]$=$u_9$,$A[0][4][8]$=$u_9$,$A[0][1][9]$=$u_{10}$,$A[0][3][9]$=$u_{11}$,$A[0][4][9]$=$u_{10}$+$u_{11}$, |
| $A[0][1][10]$=$u_{12}$,$A[0][3][10]$=$u_{12}$,$A[0][1][15]$=$u_{13}$,$A[0][3][15]$=$u_{14}$,$A[0][4][15]$=$u_{13}$+$u_{14}$, |
| $A[1][2][5]$=$u_{15}$,$A[1][4][5]$=$u_{15}$,$A[1][0][14]$=$u_{16}$,$A[1][2][14]$=$u_{17}$,$A[2][0][2]$=$u_{18}$, |
| $A[2][1][2]$=$u_{19}$,$A[2][3][2]$=$u_{20}$,$A[2][4][2]$=$u_{18}$+$u_{19}$+$u_{20}$,$A[2][0][3]$=$u_{21}$, |
| $A[2][1][3]$=$u_{22}$,$A[2][3][3]$=$u_{23}$,$A[2][4][3]$=$u_{21}$+$u_{22}$+$u_{23}$,$A[2][0][4]$=$u_{24}$, |
| $A[2][1][4]$=$u_{25}$,$A[2][3][4]$=$u_{26}$,$A[2][4][4]$=$u_{24}$+$u_{25}$+$u_{26}$,$A[2][0][5]$=$u_{27}$, |
| $A[2][3][5]$=$u_{28}$,$A[2][4][5]$=$u_{27}$+$u_{28}$,$A[2][0][6]$=$u_{29}$,$A[2][1][6]$=$u_{30}$,$A[2][3][6]$=$u_{31}$, |
| $A[2][4][6]$=$u_{29}$+$u_{30}$+$u_{31}$,$A[2][0][7]$=$u_{32}$,$A[2][1][7]$=$u_{33}$,$A[2][4][7]$=$u_{32}$+$u_{33}$, |
| $A[2][0][8]$=$u_{34}$,$A[2][1][8]$=$u_{35}$,$A[2][4][8]$=$u_{34}$+$u_{35}$,$A[2][0][10]$=$u_{36}$,$A[2][1][10]$=$u_{37}$, |
| $A[2][3][10]$=$u_{38}$,$A[2][4][10]$=$u_{36}$+$u_{37}$+$u_{38}$,$A[2][1][11]$=$u_{39}$,$A[2][4][11]$=$u_{39}$, |
| $A[2][0][12]$=$u_{40}$,$A[2][1][12]$=$u_{41}$,$A[2][3][12]$=$u_{40}$,$A[2][4][12]$=$u_{41}$,$A[4][0][12]$=$u_{42}$, |
| $A[4][3][12]$=$u_{42}$,$A[2][0][13]$=$u_{43}$,$A[2][1][13]$=$u_{44}$,$A[2][3][13]$=$u_{45}$, |
| $A[2][4][13]$=$u_{43}$+$u_{44}$+$u_{45}$,$A[2][0][14]$=$u_{46}$,$A[2][1][14]$=$u_{47}$,$A[2][3][14]$=$u_{48}$, |
| $A[2][4][14]$=$u_{46}$+$u_{47}$+$u_{48}$,$A[2][0][15]$=$u_{49}$,$A[2][1][15]$=$u_{50}$,$A[2][3][15]$=$u_{51}$, |
| $A[2][4][15]$=$u_{49}$+$u_{50}$+$u_{51}$,$A[3][1][2]$=$u_{52}$,$A[3][2][2]$=$u_{53}$,$A[3][4][2]$=$u_{52}$+$u_{53}$, |
| $A[4][0][4]$=$u_{54}$,$A[4][2][4]$=$u_{55}$,$A[4][3][4]$=$u_{54}$+$u_{55}$,$A[4][0][5]$=$u_{56}$,$A[4][2][5]$=$u_{57}$, |
| $A[4][3][5]$=$u_{56}$+$u_{57}$,$A[4][0][6]$=$u_{49}$+$u_{51}$+$u$,$A[4][3][6]$=$u_{49}$+$u_{51}$+$u$, |
| $A[4][2][8]$=$u_{58}$,$A[4][3][8]$=$u_{58}$,$A[4][0][7]$=$u_{59}$,$A[4][2][7]$=$u_{60}$,$A[4][3][7]$=$u_{59}$+$u_{60}$, |
| $A[4][0][13]$=$u_{61}$,$A[4][2][13]$=$u_{62}$,$A[4][3][13]$=$u_{61}$+$u_{62}$ |

Table 11: Parameters set for attack on 9-round `KMAC256`

| **kernel quadratic term** |
|---|
| $A[0][0][0]=A[0][3][0]=A[2][0][2]=A[2][2][2]=v_0,A[1][1][20]=A[3][1][9]=v_1$ |
| **Bit Condition** |
| $k_{14}+k_{207}+k_{334}+k_{527}+k_{654}+k_{847}+k_{911}+k_{974}+k_{1167}+k_{1294}+k_{1487}+1=0$ |
| **Ordinary Cube Variables** |
| $A[0][0][1]=u_0,A[0][1][1]=u_1,A[0][2][1]=u_2,A[0][3][1]=u_0+u_1+u_2,A[0][0][2]=u_3,A[0][1][2]=u_4,A[0][2][2]=u_5,A[0][3][2]=u_3+u_4+u_5,A[0][0][3]=u_6,A[0][1][3]=u_7,A[0][2][3]=u_8,A[0][3][3]=u_6+u_7+u_8,A[0][0][4]=u_9,A[0][1][4]=u_{10},A[0][2][4]=u_{11},A[0][3][4]=u_9+u_{10}+u_{11},A[0][0][5]=u_{12},A[0][1][5]=u_{13},A[0][2][5]=u_{14},A[0][3][5]=u_{12}+u_{13}+u_{14},A[0][0][6]=u_{15},A[0][1][6]=u_{16},A[0][2][6]=u_{17},A[0][3][6]=u_{15}+u_{16}+u_{17},A[0][0][7]=u_{18},A[0][1][7]=u_{19},A[0][2][7]=u_{18}+u_{19},A[0][0][9]=u_{20},A[0][1][9]=u_{21},A[0][2][9]=u_{22},A[0][3][9]=u_{20}+u_{21}+u_{22},A[0][0][10]=u_{23},A[0][1][10]=u_{24},A[0][3][10]=u_{23}+u_{24},A[0][0][11]=u_{25},A[0][1][11]=u_{26},A[0][2][11]=u_{27},A[0][3][11]=u_{25}+u_{26}+u_{27},A[0][0][12]=u_{28},A[0][1][12]=u_{29},A[0][2][12]=u_{30},A[0][3][12]=u_{28}+u_{29}+u_{30},A[0][0][13]=u_{31},A[0][1][13]=u_{32},A[0][2][13]=u_{33},A[0][3][13]=u_{31}+u_{32}+u_{33},A[0][0][21]=u_{34},A[0][1][21]=u_{35},A[0][2][21]=u_{36},A[0][3][21]=u_{34}+u_{35}+u_{36},A[0][1][23]=u_{37},A[0][2][23]=u_{38},A[0][3][23]=u_{37}+u_{38},A[0][0][26]=u_{39},A[0][3][26]=u_{39},A[0][0][27]=u_{40},A[0][1][27]=u_{41},A[0][2][27]=u_{42},A[0][3][27]=u_{40}+u_{41}+u_{42},A[0][0][33]=u_{43},A[0][1][33]=u_{44},A[0][2][33]=u_{45},A[0][3][33]=u_{43}+u_{44}+u_{45},A[0][0][35]=u_{46},A[0][1][35]=u_{47},A[0][2][35]=u_{48},A[0][3][35]=u_{46}+u_{47}+u_{48},A[0][0][43]=u_{49},A[0][1][43]=u_{50},A[0][2][43]=u_{51},A[0][3][43]=u_{49}+u_{50}+u_{51},A[0][0][45]=u_{52},A[0][1][45]=u_{53},A[0][2][45]=u_{54},A[0][3][45]=u_{52}+u_{53}+u_{54},A[0][0][46]=u_{55},A[0][1][46]=u_{56},A[0][2][46]=u_{57},A[0][3][46]=u_{55}+u_{56}+u_{57},A[0][0][47]=u_{58},A[0][1][47]=u_{59},A[0][2][47]=u_{60},A[0][3][47]=u_{58}+u_{59}+u_{60},A[0][0][49]=u_{61},A[0][1][49]=u_{62},A[0][2][49]=u_{63},A[0][3][49]=u_{61}+u_{62}+u_{63},A[0][0][53]=u_{64},A[0][1][53]=u_{65},A[0][2][53]=u_{66},A[0][3][53]=u_{64}+u_{65}+u_{66},A[0][0][57]=u_{67},A[0][1][57]=u_{68},A[0][2][57]=u_{69},A[0][3][57]=u_{67}+u_{68}+u_{69},A[0][1][60]=u_{70},A[0][2][60]=u_{71},A[0][3][60]=u_{70}+u_{71},A[0][0][61]=u_{72},A[0][1][61]=u_{73},A[0][2][61]=u_{74},A[0][3][61]=u_{72}+u_{73}+u_{74},A[0][0][63]=u_{75},A[0][1][63]=u_{76},A[0][2][63]=u_{77},A[0][3][63]=u_{75}+u_{76}+u_{77},A[1][0][6]=u_{78},A[1][1][6]=u_{79},A[1][2][6]=u_{78}+u_{79},A[1][0][16]=u_{80},A[1][1][16]=u_{81},A[1][2][16]=u_{80}+u_{81},A[1][0][26]=u_{82},A[1][2][26]=u_{82},A[1][0][43]=u_{83},A[1][1][43]=u_{84},A[1][2][43]=u_{83}+u_{84},A[1][0][52]=u_{85},A[1][1][52]=u_{86},A[1][2][52]=u_{85}+u_{86},A[1][0][57]=u_{87},A[1][1][57]=u_{88},A[1][2][57]=u_{87}+u_{88},A[1][0][62]=u_{89},A[1][1][62]=u_{90},A[1][2][62]=u_{89}+u_{90},A[2][0][10]=u_{91},A[2][2][10]=u_{91},A[2][0][18]=u_{92},A[2][1][18]=u_{93},A[2][2][18]=u_{92}+u_{93},A[2][0][22]=u_{94},A[2][1][22]=u_{95},A[2][2][22]=u_{94}+u_{95},A[2][0][23]=u_{96},A[2][1][23]=u_{97},A[2][2][23]=u_{96}+u_{97},A[2][0][25]=u_{98},A[2][1][25]=u_{99},A[2][2][25]=u_{98}+u_{99},A[2][0][32]=u_{100},A[2][1][32]=u_{101},A[2][2][32]=u_{100}+u_{101},A[2][0][42]=u_{102},A[2][1][42]=u_{103},A[2][2][42]=u_{102}+u_{103},A[2][0][55]=u_{104},A[2][1][55]=u_{105},A[2][2][55]=u_{104}+u_{105},A[3][0][2]=u_{106},A[3][1][2]=u_{107},A[3][2][2]=u_{106}+u_{107},A[3][0][8]=u_{108},A[3][1][8]=u_{109},A[3][2][8]=u_{108}+u_{109},A[3][0][24]=u_{110},A[3][1][24]=u_{111},A[3][2][24]=u_{110}+u_{111},A[3][0][33]=u_{112},A[3][1][33]=u_{113},A[3][2][33]=u_{112}+u_{113},A[3][0][42]=u_{114},A[3][1][42]=u_{115},A[3][2][42]=u_{114}+u_{115},A[3][0][44]=u_{116},A[3][1][44]=u_{117},A[3][2][44]=u_{116}+u_{117},A[3][0][46]=u_{118},A[3][1][46]=u_{119},A[3][2][46]=u_{118}+u_{119},A[3][0][48]=u_{120},A[3][1][48]=u_{121},A[3][2][48]=u_{120}+u_{121},A[3][0][51]=u_{122},A[3][1][51]=u_{123},A[3][2][51]=u_{122}+u_{123},A[3][0][57]=u_{124},A[3][2][57]=u_{124},A[4][0][31]=u_{125},A[4][1][31]=u_{126},A[4][2][31]=u_{125}+u_{126}$ |