Des. Codes Cryptogr. (2018) 86:1295–1310
https://doi.org/10.1007/s10623-017-0396-7

Conditional cube attack on round-reduced River Keyak

**Wenquan Bi[1] · Zheng Li[1] · Xiaoyang Dong[2] ·
Lu Li[1] · Xiaoyun Wang[1,2]**

Received: 10 January 2017 / Revised: 8 May 2017 / Accepted: 19 July 2017 /
Published online: 28 July 2017

Abstract This paper evaluates the security level of the River Keyak against the cube-like attack. River Keyak is the only lightweight scheme of the Keccak-permutation-based authenticated encryption cipher Keyak, which is one of the 16 survivors of the third round CAESAR competition. Dinur et al. gave the seven-round cube-like attack on Lake Keyak (1600-bit) using the divide-and-conquer method at EUROCRYPT 2015, then Huang et al. improved the result to eight-round using a new conditional cube attack at EUROCRYPT 2017. While for River Keyak, the 800-bit state is so small that the equivalent key (256-bit capacity) occupy double lanes, the attacks can not be applied to the River Keyak trivially. In this paper, we comprehensively explore the conditional cube attack on the small state (800-bit) River Keyak. Firstly, we find a new conditional cube variable which has a much weaker diffusion than Huang et al.'s, this makes the conditional cube attack possible for small state (800-bit) River Keyak. Then we find enough cube variables for six/seven-round River Keyak and successfully launch the key recovery attacks on six/seven-round River Keyak with the time complexity 2^{33} and 2^{49}, respectively. We also verify the six and seven-round attack on a laptop. Finally, by using linear structure technique with our new conditional cube variable, we greatly increase the freedom degree to find more cube variables for conditional cube attacks as it is complex for 800-bit state to find enough cube variables for eight-round attack. And then we use the new variables by this new method to launch eight-round conditional cube attack with the time complexity 2^{81}. These are the first cryptanalysis results on round-reduced River Keyak. Our attacks do not threaten the full-round (12) River Keyak.

Communicated by V. Rijmen.

✉ Xiaoyang Dong
xiaoyang-dong@163.com; dongxiaoyang@mail.sdu.edu.cn

✉ Xiaoyun Wang
xiaoyunwang@tsinghua.edu.cn

[1] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan 250100, China

[2] Institute for Advanced Study, Tsinghua University, Beijing 100084, China

Keywords River Keyak · Conditional cube · Key recovery · Authentication encryption · CAESAR

Mathematics Subject Classification 94A60

1 Introduction

Nowadays, the authenticated encryption (AE) schemes, which provide message confidentiality and integrity simultaneously, attract a lot of attentions of the worldwide cryptanalysts. In order to find the alternatives for AES-GCM [17], the CAESAR [3] competition was launched to find secure AE algorithms in 2014, which processed to the third round and 16 survivors are remained. In order to get the secure finalist, there have been a lot of security evaluations for them such as [4,10,16] and more analyses are needed.

Keyak [12] is one of the 16 candidates of third round CAESAR competition, which is based on the Keccak-p permutation. It has five instances: River Keyak, Lake Keyak, Sea Keyak, Ocean Keyak and Lunar Keyak. The River Keyak is the only lightweight, 800-bit-state instance and the others all have 1600-bit state. Its key size is variable, with a minimum of 128 bits; its tag sizes is 128 bits long if not truncated and the capacity is set to 256 bits long when its security strength is 128 bits according to their security claims.

Cube attack [7] is a chosen IV key-recovery attack, which was introduced by Dinur and Shamir. Since then, cube attack was applied to many different cryptographic primitives such as [1,8,11]. In Eurocrypt 2015, Dinur et al. [9] presented a key-recovery cube-like attack on round-reduced Keccak-MAC and Lake Keyak using a divide-and-conquer method. They use some auxiliary variables and carefully select the cube variables so that the cube sums depend only on a small number of key bits. They finally achieved seven-round key-recovery attack on Lake Keyak (1600 bit). Then Huang et al. [15] proposed a new *conditional cube attack* on Keccak-MAC and Lake Keyak. By inducing some bit conditions, they carefully select a new set of cube variables so that they do not multiply with each other in the first round as well as meet the condition that there is one variable does not multiply with other variables in the second round and then the degree over the cube variables is further reduced.

However, those attacks can not translate to the River Keyak trivially. For River Keyak, the state is 800-bit, with 25 32-bit lanes. The 256-bit capacity occupy eight lanes, double of the situation of Lake Keyak. For the divide-and-conquer method by Dinur et al., it is impossible to achieve eight-round attack and for 800-bit-state River Keyak, the seven-round cube-like attack is also difficult. When applying Huang's conditional cube variable to the 800-bit-state cipher, only 25 cube variables could be found by Huang's algorithms, which just could be used to achieve six-round cube-like attack for River Keyak.

1.1 Our contributions

In this paper, we comprehensively explore the secure level of River Keyak against cube-like attack. Our contributions are in three folds. Firstly, we find a new set of conditional cube variable which has a much weaker diffusion than Huang et al.'s. Then we find some new sets of 16/32 cube variables for River Keyak, which meet the condition that they do not multiply with each other after the first round as well as meet the condition that one cube variable does not multiply with the others after the second round. This makes it possible to achieve six/seven-round key-recovery attacks on River Keyak. Secondly, we launch the six/seven-round conditional cube attack on River Keyak successfully with the time complexity 2^{33} and

Table 1 Summary of key recovery attacks on Keyak

Instances	Rounds	Methods	Time	Data	Memory	Source
Lake Keyak	7	Divide and conquer	2^{76}	2^{75}	2^{43}	[9]
Lake Keyak	7	Conditional cube attack	2^{42}	2^{42}	Negligible	[15]
Lake Keyak	8	Conditional cube attack	2^{74}	2^{74}	Negligible	[15]
River Keyak	6	Conditional cube attack	2^{33}	2^{33}	Negligible	Sect. 4.1
River Keyak	7	Conditional cube attack	2^{49}	2^{49}	Negligible	Sect. 4.2
River Keyak	8	Conditional cube attack	2^{81}	2^{81}	Negligible	Sect. 5

2^{49}, respectively. Our six/seven-round attacks are practical and we give the experimental verification. Finally, by applying linear structure technique [14], we find 64 cube variables (including the conditional cube variable) and extend the key-recovery attack on River Keyak to eight rounds with the time complexity 2^{81}. The attacks are summarized in Table 1. Those are the first attacks on round-reduced River Keyak.

This paper is organized as follows: Sect. 2 introduces some notations, Keccak-p permutations, River Keyak and assumptions for our attack. In Sect. 3, some related works are introduced. We present the attacks on six/seven-round River Keyak in Sect. 4. In Sect. 5, we use the linear structure technique to find enough new dynamic conditional cube variables and give the eight-round conditional attack result on River Keyak. At last, we conclude this paper in Sect. 6.

2 Preliminary

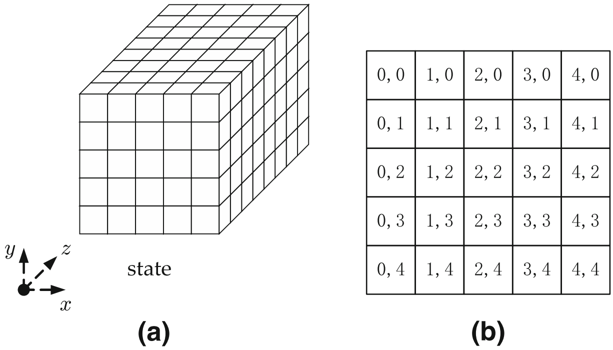
In this section we give some notations and theorems used in this paper, a brief description of Keccak-p and River Keyak, together with our attack assumptions.

2.1 Notations

$\oplus$, $\neg$ *and* &	bitwise xor, negation, and logic AND
S_i	the state after ith round, for example, $S_{0.5}$ means before the χ operation of first round; that is, we treat the θ, ρ and π operation as the first half round,
A	the state after the first inner permutation of River Keyak, where the first message is involved
$A[x][y]$	a lane in xth column and yth row of state A, $0 \le x, y \le 4$,
$A[x][y][z]$	zth MSB of $A[x][y]$, $0 \le x, y \le 4$, $0 \le z \le 31$,
K	the master 128-bit key used in the initialization stage, the master key of River Keyak,
k_i	equivalent 32-bit key after first inner permutation of River Keyak, $1 \le i \le 8$,
$k_i[j]$	the jth MSB bit of k_i, $1 \le i \le 8$, $0 \le j \le 31$.

2.2 The Keccak-p permutations

The Keccak-p permutations are derived from the Keccak-f permutations [2] and have a tunable number of rounds. A Keccak-p permutation is defined by its width $b = 25 \times 2^l$,



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

Fig. 1 **a** State of Keccak, and **b** State in two-dimension

with $b \in \{25, 50, 100, 200, 400, 800, 1600\}$, and its numbers of rounds n_r, denoted as Keccak-$p[b, n_r]$. When $n_r = 12 + 2l$, Keccak-$p[b, n_r]$=Keccak-f. The round function R consists of five operations:

$$R = \iota \circ \chi \circ \pi \circ \rho \circ \theta.$$

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

$$\theta \colon A[x, y] = A[x, y] \oplus \sum_{j=0}^{4} (A[x-1, j]) \oplus (A[x+1, j] \lll 1)).$$

$$\rho \colon A[x, y] = A[x, y] \lll r[x, y].$$

$$\pi \colon A[y, 2x+3y] = A[x, y].$$

$$\chi \colon A[x, y] = A[x, y] \oplus ((\neg A[x+1, y]) \& A[x+2, y]).$$

$$\iota \colon A[0, 0] = A[0, 0] \oplus RC.$$

In River Keyak, $b = 800$, $n_r = 12$, and in other schemes, $b = 1600$, $n_r = 12$. In this paper, we refer to the linear step θ, ρ and π as the first half of a round, and the remaining step χ and ι as the second half of a round.

2.3 A brief description of Keyak

AE cipher Keyak is one of the 16 candidates in the third round CAESAR competition, whose mode is based on Motorist mode, which is sponge-based and supports one or more duplex instances operating in parallel. The Motorist makes duplexing calls with input containing key, nonce, plaintext and metadata (possible associated data) bits and uses its output as tag or as key stream bits. To start a session, Motorist takes input as a secret and unique value, and it has three layers: Motorist, Engine and Piston layers. The Motorist layer implement user interface, Engine layer control $\Pi \geq 1$ Piston objects and the Piston layer keeps a b-bit state and applies permutation f(Keccak-$p[b]$ in Keyak) to it.

In Keyak, five instances are proposed, shown in Table 2. For all instances, the round numbers of Keccak-$p[b]$ is $n_r = 12$, the capacity $c = 256$ and the tag length $\tau = 128$. The

Table 2 Five instances of Keyak

Names	b	Π	Main use cases	Second use case
River Keyak	800	1	Defense-in-depth	Lightweight
Lake Keyak	1600	1	Defense-in-depth	High performance
Sea Keyak	1600	2	Defense-in-depth	High performance
Ocean Keyak	1600	4	Defense-in-depth	High performance
Lunar Keyak	1600	8	Defense-in-depth	High performance

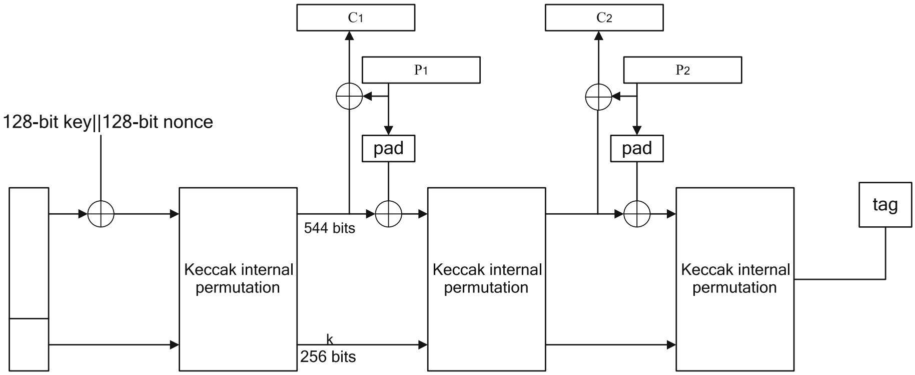

C1
P1
C2
P2
128-bit key||128-bit nonce
pad
pad
tag
544 bits
Keccak internal permutation
Keccak internal permutation
Keccak internal permutation
k
256 bits

Fig. 2 Construction of River Keyak on two blocks

River Keyak is the only instance which has 800-bit state and 1600-bit state for others. The primary recommendation is the Lake Keyak. Readers can refer to [12] for more details.

2.4 Our attack assumptions

According to the specification of Keyak [12], in order to assure confidentiality of data, a nonce cannot be reused; however, when confidentiality of data is not required, a variable nonce is not required. That is, a nonce can be reused when just authenticity and integrity of data is required. Therefore, in our attack, we only aim to break the authenticity and integrity of River Keyak. Like in [9] and [15], in our attack, the River Keyak processes two plaintext blocks as shown in Fig. 2. Although the associated data could provide additional degrees of freedom, it has no effect on our cube attacks. Thus, the input of the first permutation call contains a key and a nonce but no optional associated data.

3 Related works

3.1 Cube attack

The cube attack [7] is a key-recovery attack introduced by Dinur and Shamir at EUROCRYPT 2009. It assumes the output bit of a cipher can be regard as a polynomial $f(k_1, \ldots, k_n, v_1, \ldots, v_m)$ over $GF(2)$, where $k_1, \ldots, k_n$ are secret variables (e.g., the key bits), $v_1, \ldots, v_m$ are public variables (e.g., the nonce or IV bits). The main observation is the following theorem.

Theorem 1 *[7] Given a polynomial* $f : X^n \to 0,\ 1$ *of degree d. It can be written as a sum of two polynomials*:

$$f\ (k_1, \ldots, k_n,\ v_1, \ldots, v_m) = T \cdot P + Q\ (k_1, \ldots, k_n,\ v_1, \ldots, v_m)\,.$$

T is called maxterm and is a product of certain public variables, for example $(v_1, \ldots, v_s)$, $1 \le s \le m$, *which is called a cube* C_T; *P is called superpoly*; $Q(k_1, \ldots, k_n,\ v_1, \ldots, v_m)$ *is the remainder polynomial and none of its terms is divisible by T. Then the sum of f over all values of the cube* C_T (*cube sum*) *is*:

$$\sum_{x'=(v_1,\ldots,v_s)\in C_T} f\left(k_1, \ldots, k_n,\ x', \ldots, v_m\right) = P,$$

whose degree is at most d-s, where the cube C_T *contains all binary vectors of the length s and the other public variables are fixed to constants.*

3.2 Dynamic cube attack

Dinur and Shamir introduce a variant of cube attack called dynamic cube attack [8] in FSE 2011. The basic idea is to find dynamic variable, which depends on some of the public cube variables and some private variables (the key bits), to nullify the complex function $P = P_1 \cdot P_2 + P_3$, where P_3's degree is relatively lower than P and $P_1 \cdot P_2$ is complex. Then guess the key and compute the dynamic cube variables to make P_1 to be zero and the function is simplified greatly. The right guess of key will lead the cube sum to be zero otherwise the cube sums will be random generally.

3.3 Dinur et al.'s divide-and-conquer method

Dinur et al. presented cube-like attacks on Keccak keyed modes using divide-and-conquer method in EUROCRYPT 2015 [9]. They achieved key recovery attack for the Keccak-MAC, Keyak and stream cipher mode, respectively. They explored the property that the cube variables multiply less secret variables in the first round then recover a part of secret variables and recover other secret bits using other cubes. In Dinur et al.'s attack for seven-round Lake Keyak, where the secret bits are the 256 capacity which occupy four lanes. They choose the 32 cube variables among the five lanes with $x = 0$. More precisely, the eight LSBs of first four lanes $A[0][0]$, $A[0][1]$, $A[0][2]$, $A[0][3]$ are set as independent cube variables while the eight LSBs of $A[0][4]$ act as "parity checks". Those 40 bits multiply 80 bits in the χ operation of the first round which are regarded as secret expressions. They enumerated all the possible values of 80 bits and store the 2^{80} cube sums in a hash table with 2^{112} computational complexity and 2^{80} memory complexity. In order to recover all the 256 secret variables, they used eight cubes by rotating the initial cube variables eight bits towards the MSB. In the balanced attack, they calculated 2^{40} cube sums only in the precomputing phase and the computational complexity is 2^{76}, data complexity is 2^{75}.

3.4 Linear structure

Inspired by Dinur et al.'s work [8], Guo et al. [14] developed a new technique named linear structure, that allows linearization of the underlying permutation of Keccak for up to three rounds. After that, a series of distinguishers and preimage attacks were launched by this

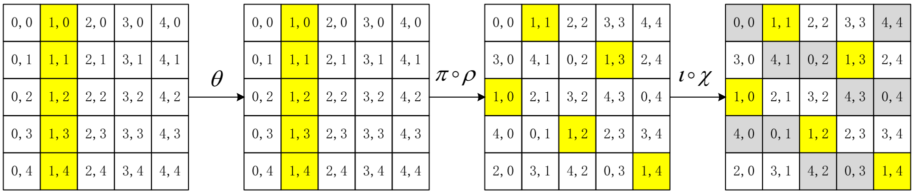


Fig. 3 One-round linear structure

technique. Let $A[1, i]$, $i = \{0, 1, 2, 3\}$ be variables and $A[1, 4] = \oplus_{i=0}^{3} A[1, i]$, then the sum of variables in each column is zero. How these variables affect the internal state under the transformation of Keccak-f round function R is shown in Fig. 3. The algebraic degree of all the yellow lanes is 1, and the light grey's is at most 1, the other lanes are all constants. In fact, only the non-linear operation χ can increase the algebraic degree through two neighbouring bits due to the term $(a_{i+1} \oplus 1) \cdot a_{i+2}$. Hence, the algebraic degree of the state bits remains at most 1 after one round function R. We denote the linear structure as one-round linear structure. The size of free variables can be at most four lanes. If we use the one-round linear structure as a cube, then the cube variables will not multiply with each other after one round.

3.5 Huang et al. conditional cube attack

In [15], Huang et al. developed the Conditional cube attack and applied to Keccak keyed mode including the Lake Keyak and accomplished the eight-round key-recovery attack on Lake Keyak. By introducing some bit conditions in the first round, Huang et al. found 64-dimension cubes which do not multiply in the first round and have one variable does not multiply others in the second round. They achieved eight-round key-recovery attack with the time complexity 2^{74}. We quote some definitions and theorems in [15] here.

Definition 1 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**.

Definition 2 Given a Boolean function $f(x_0, x_1, \ldots, x_{n-1})$, the bitwise derivative of f with respect to the variable x_m is defined as

$$\delta_{x_m} f = f_{x_m=1} + f_{x_m=0}.$$

Property 1 *[15]* (Bit Conditions) *Write the input of* χ *operation to be boolean function* $F = (f_0, f_1, f_2, f_3, f_4)$ *and the corresponding output as* $G = (g_0, g_1, g_2, g_3, g_4)$, *denote the conditional cube variable as* v_0, *if* $\delta_{v_0} F = (1, 0, 0, 0, 0)$, *then* $\delta_{v_0} G = (1, 0, 0, 0, 0)$ *if and only if* $f_1 = 0$ *and* $f_4 = 1$.

From the view of the differential characteristic, if $f_1 = 0$ and $f_4 = 1$, then the differential characteristic $(1, 0, 0, 0, 0) \rightarrow (1, 0, 0, 0, 0)$ holds with probability 1. The all of the five derivation cases shown in Table 3 where each input bitwise derivative has only one non-zero bit.

Theorem 2 (*[15]*) *For* $(n + 2)$*-round Keccak sponge function* $(n > 0)$, *if there are* p $(0 \leq p < 2^n + 1)$ *conditional cube variables* $v_0, \ldots, v_{p-1}$, *and* $q = 2^{n+1} - 2p + 1$

Table 3 Five derivation cases of χ in Keccak-p

Input/output bitwise derivative (difference)	Conditions
$(1, 0, 0, 0, 0) \rightarrow (1, 0, 0, 0, 0)$	$f_1 = 0,\ f_4 = 1$
$(0, 1, 0, 0, 0) \rightarrow (0, 1, 0, 0, 0)$	$f_2 = 0,\ f_0 = 1$
$(0, 0, 1, 0, 0) \rightarrow (0, 0, 1, 0, 0)$	$f_3 = 0,\ f_1 = 1$
$(0, 0, 0, 1, 0) \rightarrow (0, 0, 0, 1, 0)$	$f_4 = 0,\ f_2 = 1$
$(0, 0, 0, 0, 1) \rightarrow (0, 0, 0, 0, 1)$	$f_0 = 0,\ f_3 = 1$

ordinary cube variables, $u_0, \ldots, u_{q-1}$ (*if* $q = 0$, *we set* $p = 2^n + 1$), *the term* $v_0, v_1, \ldots, v_{p-1}u_0, \ldots, u_{q-1}$ *will not appear in the output polynomials of* $(n+2)$*-round Keccak sponge function.*

Actually, we 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*, $u_0, \ldots, u_{q-1}$, *the term* $v_0, \ldots, u_0, \ldots, u_{q-1}$ *will not appear in the output polynomials of* $(n+2)$*-round Keccak sponge function.*

For more details and proofs you can refer to [15].

4 Conditional cube attack on six/seven-round River Keyak

In this section, we present our key recovery attack results on six/seven-round River Keyak, including the attack process, complexity analysis and experimental results. First, we describe a few property of Keccak permutation as follows:

Property 2 *[13] If the sum of the cube variables in one column is zero, these variables will not diffuse to other bits after* θ *operation.*

Property 3 *In* χ *operation, one bit* $A[i][j][k]$ *only multiplies with* $A[i-1][j][k] \oplus 1$ *and* $A[i+1][j][k]$, *when* $A[i][j][k]$ *is cube variable* v_0 *and* $A[i-1][j][k] \oplus 1 = A[i+1][j][k] = 0$, *then* v_0 *does not multiply with other bits after* χ *operation.*

Property 4 *[15] If we set two equal conditional variables* v_0 *in one column which both meet the condition in Property 3 in* χ *operation will affect 22 variables after 1.5-round Keccak internal permutation as shown in Fig. 4a, called 2–2–22 pattern.*

Property 5 *[5,6] Six equal conditional variables* v_0 *shown in Fig. 4b which all meet the condition in Property 3 in* χ *operation will affect six variables only after 1.5-round Keccak internal permutation as shown in Fig. 4b, called 6–6–6 pattern.*

Comparing with Huang et al.'s conditional cube attack. As shown in Fig. 4a, if select two conditional variables v_0 in the same column, after 1.5-round the conditional variables v_0 are diffused to as more as 22 bits. For 1600-bit state used in Lake Keyak, it is relatively sparse to find enough cube variables which do not multiply with v_0 after the second round. However, for River Keyak's 800-bit state, the effective bits are too densely to find 32-dimension

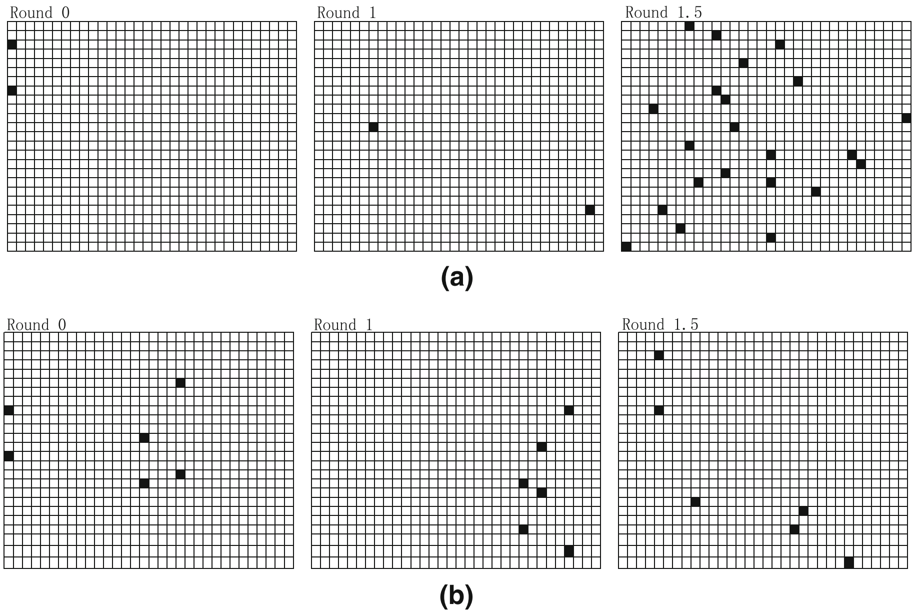


Fig. 4 The 2–2–2 pattern and 6–6–6 pattern of Keccak-p permutation. **a** The 2–2–22 pattern of Keccak-p permutation. **b** The 6–6–6 pattern of Keccak-p permutation

ordinary cube variables even. In fact, one could find at most 25 ordinary variables using the search algorithm by Huang et al.'s. In order to solve this problem, we find a new conditional variable which follows the 6–6–6 pattern as shown in Fig. 4b. We select these six bits A[0][0][0], A[1][0][31], A[0][1][0], A[2][1][30], A[1][2][31] and A[2][2][30] as conditional cube variables v_0, and use the bit conditions as follows:

$$\begin{cases} A_{0.5}[4][0][0] = 1; \ A_{0.5}[1][0][0] = 0 \\ A_{0.5}[1][0][9] = 1; \ A_{0.5}[3][0][9] = 0 \\ A_{0.5}[4][2][0] = 1; \ A_{0.5}[1][2][0] = 0 \\ A_{0.5}[0][2][4] = 1; \ A_{0.5}[2][2][4] = 0 \\ A_{0.5}[0][3][4] = 1; \ A_{0.5}[2][3][4] = 0 \\ A_{0.5}[1][3][9] = 1; \ A_{0.5}[3][3][9] = 0 \end{cases}$$

The $A_{0.5}$ is the state before χ operation in the first round, thus the conditional variables are only six bits which are so sparse that we could find enough ordinary variables to launch the attack for 800-bit River Keyak.

4.1 Conditional cube attack on six-round River Keyak

We use the 16-dimension cube variables denoted as $v_0, v_1, \ldots, v_{15}$ to implement the conditional cube attack on six-round River Keyak. As discussed above, we take six bits as the conditional cube variables v_0 which follow the 6–6–6 pattern in Property 5. These six bits impact only six bits after 1.5-round Keyak permutation. Then we take the corresponding bit conditions and the cube variables following these requirements:

(1) $v_0, v_1, \ldots, v_{15}$ do not multiply with each other in the first round;

Table 4 Parameters set for attack on six-round River Keyak

Ordinary cube variables

$A[1][0][0]=A[1][2][0]=v_1$, $A[1][0][1]=A[1][2][1]=v_2$, $A[1][0][2]=A[1][2][2]=v_3$,
$A[1][0][5]=A[1][2][5]=v_4$, $A[1][0][8]=A[1][2][8]=v_5$, $A[1][0][15]=A[1][2][15]=v_6$,
$A[1][0][20]=A[1][2][20]=v_7$, $A[1][0][23]=A[1][2][23]=v_8$, $A[1][0][24]=A[1][2][24]=v_9$,
$A[1][0][25]=A[1][2][25]=v_{10}$, $A[1][0][28]=A[1][2][28]=v_{11}$, $A[2][0][4]=A[2][2][4]=v_{12}$,
$A[2][0][13]=A[2][2][13]=v_{13}$, $A[2][0][14]=A[2][2][14]=v_{14}$, $A[2][0][15]=A[2][2][15]=v_{15}$

Conditional cube variables

$A[0][0][0]=A[1][0][31]=A[0][1][0]=A[2][1][30]=A[1][2][31]=A[2][2][30]=v_0$

Bit condition

$A[1][1][29]=A[2][0][28]+A[0][1][29]+A[0][0][29]+A[2][1][28]+A[0][2][29]+A[2][2][28]+A[0][3][29]$
$+k_1[28]+k_4[29]+k_6[28]+1$,
$A[4][1][19]=A[4][0][19]+A[2][1][20]+A[2][0][20]+A[2][2][20]+A[4][2][19]$
$+k_1[20]+k_2[20]+k_3[19]+k_6[20]+k_8[19]$,
$A[3][1][18]=A[3][0][18]+A[0][1][17]+A[0][0][17]+A[0][2][17]+A[3][2][18]+A[0][3][17]$
$+k_2[18]+k_4[17]+k_7[18]+k_8[18]+1$,
$A[1][1][20]=A[2][0][19]+A[0][1][20]+A[0][0][20]+A[2][1][19]+A[0][2][20]+A[2][2][19]+A[0][3][20]$
$+k_1[19]+k_4[20]+k_6[19]$,
$A[2][1][11]=A[4][0][10]+A[2][0][11]+A[4][1][10]+A[2][2][11]+A[3][2][11]+A[4][2][10]$
$+k_1[11]+k_3[10]+k_6[11]+k_8[10]$,
$A[1][1][13]=A[4][0][14]+A[1][0][13]+A[4][1][14]+A[1][2][13]+A[4][2][14]+A[1][3][13]$
$+k_3[14]+k_4[14]+k_5[13]+k_8[14]+1$,
$A[2][1][26]=k_1[26]$
$A[0][1][3]=A[1][0][3]+A[2][0][2]+A[0][0][3]+A[2][1][2]+A[0][2][3]+A[2][2][2]+A[0][3][3]$
$+k_1[2]+k_4[3]+k_6[2]+1$,
$A[0][1][26]=A[2][0][25]+A[0][0][26]+A[2][1][25]+A[0][2][26]+A[1][2][26]+A[2][2][25]+A[0][3][26]$
$+k_1[25]+k_4[26]+k_6[25]$,
$A[1][1][4]=A[4][0][5]+A[0][1][5]+A[1][0][4]+A[4][1][5]+A[1][2][4]+A[4][2][5]+A[1][3][4]$
$+k_3[5]+k_5[4]+k_8[5]+1$,
$A[1][1][26]=A[3][0][25]+A[1][0][26]+A[3][1][25]+A[1][2][26]+A[3][2][25]+A[1][3][26]+k_1[26]$
$+k_2[25]+k_5[26]+k_7[25]$,
$A[0][1][8]=A[3][0][9]+A[4][0][9]+A[0][0][8]+A[3][1][9]+A[0][2][8]+A[3][2][9]+A[0][3][8]$
$+k_2[9]+k_4[8]+k_7[9]+1$

Guessed key bits

$k_1[28]+k_4[29]+k_6[28]+1$, $k_1[20]+k_2[20]+k_3[19]+k_6[20]+k_8[19]$, $k_1[19]+k_4[20]+k_6[19]$,
$k_2[18]+k_4[17]+k_7[18]+k_8[18]+1$, $k_1[11]+k_3[10]+k_6[11]+k_8[10]$, $k_3[14]+k_4[14]+k_5[13]+k_8[14]+1$,
$k_1[26]$, $k_1[2]+k_4[3]+k_6[2]+1$, $k_1[25]+k_4[26]+k_6[25]$, $k_3[5]+k_5[4]+k_8[5]+1$,
$k_1[26]+k_2[25]+k_5[26]+k_7[25]$, $k_2[9]+k_4[8]+k_7[9]+1$

(2) v_0 does not multiply with any of $v_1, \ldots, v_{15}$ in the second round.

The item (1) guarantees the algebraic degree of the first round is only 1 and of the first two rounds is 2 at most. Under item (2), the term $v_0, v_1, \ldots, v_{15}$ would not appear in A_5 which means after five rounds the algebraic degree over $v_0, v_1, \ldots, v_{15}$ is at most 15 not 16. Thus the cube sums of the output of five-round River Keyak over $v_0, v_1, \ldots, v_{15}$ are all zero. On

Table 5 Examples of test result of six-round attack

Guessed values	Cube sums
000000 000000	0xcb84f226 16c557f8 c34b83bf 13ca478d
000000 000001	0xade38cde 03efa194 f3626098 2b9d71de
...	
011001 111100	0x00000000 00000000 00000000 00000000
...	
111111 111110	0xb82ea8ca ab719819 cbc44f46 d1230742
111111 111111	0xb7143bec ee410854 db23fad0 2f176599

the other hand, we have as more as $800 - 256 = 544$ known output bits, as the θ, ρ and π are linear operations, the ANF of $A_{5.5}$ and A_5 are of the same algebraic degree over cube variables, we could compute backward for one round (especially over the χ operation) of first 15 lanes. That is, we could extend the attack to six rounds with these 16-dimension cube variables. The conditional and ordinary cube variables are listed in the Table 4 as well as the bit conditions. By controlling the bit conditions and guess the 12-bit equivalent key listed in Table 4, we compute the cube sums for 2^{16} different messages which take all possible values in the cube bits. When the equivalent key guesses are right, the propagation of the conditional cube variable v_0 follows the 6–6–6 pattern and both the above two requirements are met, which makes cube sums to be all zeros.

The conditional cube attack procedure, complexity analysis and experimental result are presented as follows:
Attack procedure

- *Step 1* For all possible 2^{12} equivalent keys, after the first Keccak internal permutation for initialization, request 2^{16} messages over the 16-dimension cube variables listed in the Table 4, choose the message to make sure the first $800 - 256 - 16 = 528$ bits to be zero (or any other arbitrary constants). Then get the outputs of the six-round second Keccak internal permutation, compute backward for ι and χ operation and compute the cube sums for the first five-lane bits with these values, if the cube sums are all zeros, we treat the 12 bits right key relations.
- *Step 2* Move the cube variables to the other 31 different depth in z axis and repeat step 1 for 31 times with the new cube variables and bit conditions, then get another $12 \times 31 = 372$ key bits relations, however, there are four guessed key relations would be repeated with each other pairwise and only 10 effective equations remained in each attack. In total, there are $10 \times 32 = 320$ key bits relations which are enough to get all the 256-bit equivalent key and just compute backward the Keccak permutation to get the initial real 128-bit key K.

Complexity analysis

For each of the 2^{12} possible equivalent key, we compute 2^{16} encryption for six-round River Keyak, and repeat for 32 times in total. So the complexity for our attack is $2^{12} \times 2^{16} \times 32 = 2^{33}$. Our attack can be implemented in several minutes on a laptop for one cube attack, so we just need several hours to recover the right key. We present a experiment result of random key and a right key in Table 5. We should note that this complexity is so low enough so reducing it is no more significance. In fact, we use the 2–2–22 patten used by Huang et al. could reduce a few complexity.

Table 6 Parameters set for attack on seven-round River Keyak

Ordinary cube variables

A[1][0][0] =A[1][2][0] =v_1,A[1][0][1] =A[1][2][1] =v_2,A[1][0][2] =A[1][2][2] =v_3,

A[1][0][5] =A[1][2][5] =v_4,A[1][0][8] =A[1][2][8] =v_5,A[1][0][15]=A[1][2][15]=v_6,

A[1][0][20]=A[1][2][20] =v_7,A[1][0][23]=A[1][2][23] =v_8,A[1][0][24]=A[1][2][24]=v_9,

A[1][0][25]=A[1][2][25]=v_{10},A[1][0][28]=A[1][2][28]=v_{11},A[2][0][4] =A[2][2][4] =v_{12},

A[2][0][13]=A[2][2][13]=v_{13},A[2][0][14]=A[2][2][14]=v_{14},A[2][0][15]=A[2][2][15]=v_{15},

A[2][0][18]=A[2][2][18]=v_{16},A[2][0][23]=A[2][2][23]=v_{17},A[2][0][27]=A[2][2][27]=v_{18},

A[2][0][28]=A[2][2][28]=v_{19},A[2][0][29]=A[2][2][29]=v_{20},A[3][0][0] =A[3][2][0] =v_{21},

A[3][0][1] =A[3][2][1] =v_{22},A[3][0][4] =A[3][2][4] =v_{23},A[3][0][5] =A[3][2][5] =v_{24},

A[3][0][8] =A[3][2][8] =v_{25},A[3][0][10]=A[3][2][10]=v_{26},A[3][0][13]=A[3][2][13]=v_{27},

A[3][0][15]=A[3][2][15]=v_{28},A[3][0][17]=A[3][2][17]=v_{29},A[3][0][25]=A[3][2][25]=v_{30},

A[3][0][30]=A[3][2][30]=v_{31}

Conditional cube variables

A[0][0][0]=A[1][0][31]=A[0][1][0]=A[2][1][30]=A[1][2][31]=A[2][2][30]=v_0

Bit condition

A[1][1][29]=A[0][1][29]+A[0][0][29]+A[2][1][28]+A[0][2][29]+A[0][3][29]
+k_1[28]+k_4[29]+k_6[28]+1,

A[4][1][19]=A[4][0][19]+A[2][1][20]+A[2][0][20]+A[2][2][20]+A[4][2][19]
+k_1[20]+k_2[20]+k_3[19]+k_6[20]+k_8[19],

A[3][1][18]=A[3][0][18]+A[0][1][17]+A[0][0][17]+A[0][2][17]+A[3][2][18]
+A[0][3][17]+k_2[18]+k_4[17]+k_7[18]+k_8[18]+1,

A[1][1][20]=A[2][0][19]+A[0][1][20]+A[0][0][20]+A[2][1][19]+A[0][2][20]
+A[2][2][19]+A[0][3][20]+k_1[19]+k_4[20]+k_6[19],

A[2][1][11]=A[4][0][10]+A[2][0][11]+A[4][1][10]+A[2][2][11]+A[3][2][11]
+A[4][2][10]+k_1[11]+k_3[10]+k_6[11]+k_8[10],

A[1][1][13]=A[4][0][14]+A[1][0][13]+A[4][1][14]+A[1][2][13]+A[4][2][14]
+A[1][3][13]+k_3[14]+k_4[14]+k_5[13]+k_8[14]+1,

A[2][1][26]=k_1[26],

A[0][1][3]=A[1][0][3]+A[2][0][2]+A[0][0][3]+A[2][1][2]+A[0][2][3]+A[2][2][2]
+A[0][3][3]+k_1[2]+k_4[3]+k_6[2]+1,

A[0][1][26]=A[2][0][25]+A[0][0][26]+A[2][1][25]+A[0][2][26]+A[1][2][26]
+A[2][2][25]+A[0][3][26]+k_1[25]+k_4[26]+k_6[25],

A[1][1][4]=A[4][0][5]+A[0][1][5]+A[1][0][4]+A[4][1][5]+A[1][2][4]
+A[4][2][5]+A[1][3][4]+k_3[5]+k_5[4]+k_8[5]+1,

A[1][1][26]=A[1][0][26]+A[3][1][25]+A[1][2][26]+A[1][3][26]
+k_1[26]+k_2[25]+k_5[26]+k_7[25],

A[0][1][8]=A[3][0][9]+A[4][0][9]+A[0][0][8]+A[3][1][9]+A[0][2][8]
+A[3][2][9]+A[0][3][8]+k_2[9]+k_4[8]+k_7[9]+1

Guessed key bits

k_1[28]+k_4[29]+k_6[28]+1, k_1[20]+k_2[20]+k_3[19]+k_6[20]+k_8[19],

k_2[18]+k_4[17]+k_7[18]+k_8[18]+1, k_1[19]+k_4[20]+k_6[19],

k_1[11]+k_3[10]+k_6[11]+k_8[10], k_3[14]+k_4[14]+k_5[13]+k_8[14]+1, k_1[26],

k_1[2]+k_4[3]+k_6[2]+1, k_1[25]+k_4[26]+k_6[25], k_3[5]+k_5[4]+k_8[5]+1,

k_1[26]+k_2[25]+k_5[26]+k_7[25], k_2[9]+k_4[8]+k_7[9]+1

Table 7 Examples of test result of seven-round attack

Guessed values	Cube sums
000000 000000	0xf763c579 c45e671a 09c44a23 f59c4097
000000 000001	0x1be9aa12 c3c3c3e7 30914305 100832df
. . .	
001111 110101	0x00000000 00000000 00000000 00000000
. . .	
111111 111110	0x5dc795e5 8823058b 60f7367a acffc5ea
111111 111111	0x95563ecf 1e272a2b 722cc7db 86ae94cc

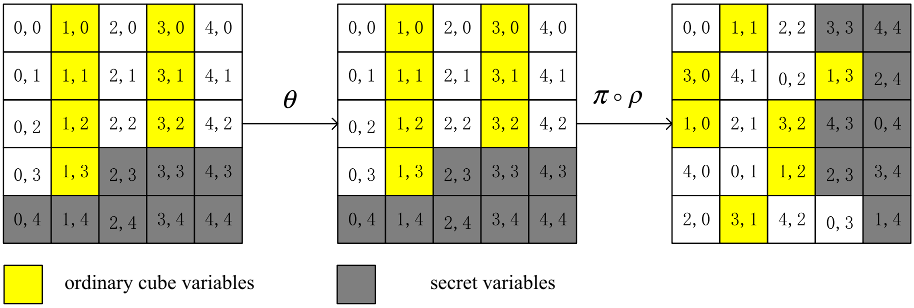


Fig. 5 Linear structure of cube variables for eight-round attack

Experimental result

We present a simple experimental result in Table 5. In our experiment, we choose messages to make the first 544 bits of the second Keccak internal permutation's input to be zeros (which could be any other arbitrary constants). The equivalent 256-bit key (capacity) is $0x724aa646\ 07a2f7ae\ 29063398\ b972526e\ 5e0c0988\ 7c634775\ 0711c592\ b39481dc$, and the right guessed key bits listed in the Table 4 are 100001011011, for all possible 2^{12} guessed keys we compute the cube sums and present in the Table 5, only when the guessed key matches the right 12-bit equivalent key, the cube sums are all zeros.

4.2 Conditional cube attack on seven-round River Keyak

We use the 32-dimension cube variables to implement the conditional cube attack on seven-round River Keyak. As shown in Table 6, we use the same condition cube variables v_0 with the six-round attack's. The 12 bit conditions are also similar to conditions presented in Sect. 4.1 (they are same when the $800 - 256 - 32 = 512$ bits are constrained to be zeros). For all possible 2^{12} equivalent keys, we compute the 2^{32} cube sums of the first 5 lanes of the output after 6.5-round Keyak round functions, if the sums are all zeros, then we treat the 12 bits are the right key guesses. By shifting the cube variables along the z axis rotational for 31 times, we could use as more as 320 bit relations get all the 256-bit equivalent key (capacity) and compute backward the first Keccak internal permutation to get the real key K. The time complexity of seven-round attack is $2^{12} \times 2^{32} \times 2^5 = 2^{49}$. Respect to experiment, we use the right 12-bit equivalent key and some random equivalent keys to compute the cube sums in several hours on a laptop, the experiment results are listed in Table 7. Like the experiment in Sect. 4.1, in this experiment we use the equivalent 256-bit key (capacity)

Table 8 Parameters set for attack on eight-round River Keyak

Ordinary cube variables

A[1][0][0]=v_1, A[1][2][0]=v_2, A[1][1][0]=$v_1 + v_2$, A[1][0][1]=v_3, A[1][2][1]=v_4, A[1][3][1]=$v_3 + v_4$,
A[1][0][2]=A[1][2][2]=v_5, A[1][1][4]=A[1][3][4]=v_6, A[1][0][5]=A[1][2][5]=v_7,
A[3][1][9]=A[3][2][9]=v_8, A[1][0][7]=A[1][1][7]=v_9, A[1][0][8]=A[1][2][8]=v_{10},
A[1][1][9]=A[1][3][9]=v_{11}, A[1][0][10]=A[1][3][10]=v_{12}, A[1][0][11]=v_{13}, A[1][3][11]=v_{14},
A[1][1][11]=$v_{13} + v_{14}$, A[1][0][12]=v_{15}, A[1][3][12]=v_{16}, A[1][1][12]=$v_{15} + v_{16}$, A[1][2][13]=v_{17},
A[1][3][13]=v_{18}, A[1][1][13]=$v_{17} + v_{18}$, A[1][2][14]=v_{19}, A[1][3][14]=v_{20}, A[1][1][14]=$v_{19} + v_{20}$,
A[1][0][15]=v_{21}, A[1][2][15]=v_{22}, A[1][3][15]=v_{23}, A[1][1][15]=$v_{21} + v_{22} + v_{23}$, A[1][2][16]=v_{24},
A[1][3][16]=v_{25}, A[1][1][16]=$v_{24} + v_{25}$,A[1][1][17]=A[1][2][17]=v_{26}, A[1][1][18]=A[1][2][18]=v_{27},
A[1][0][20]=v_{28}, A[1][2][20]=v_{29}, A[1][3][20]=$v_{28} + v_{29}$, A[3][0][6]=A[3][1][6]=v_{30},
A[3][1][9]=A[3][2][9]=v_{31}, A[1][0][22]=v_{32}, A[1][3][22]=v_{33}, A[1][1][22]=$v_{32} + v_{33}$,
A[1][2][23]=v_{35}, A[1][1][23]=$v_{34} + v_{35}$, A[1][0][24]=v_{36}, A[1][2][24]=v_{37}, A[1][3][24]=v_{38},
A[1][1][24]=$v_{36} + v_{37} + v_{38}$, A[1][0][25]=v_{39}, A[1][2][25]=v_{40}, A[1][3][25]=$v_{39} + v_{40}$,
A[1][0][26]=v_{41}, A[1][3][26]=v_{42}, A[1][1][26]=$v_{41} + v_{42}$, A[3][0][7]=A[3][1][7]=v_{43},
A[1][0][27]=v_{44}, A[1][3][27]=v_{45}, A[1][1][27]=$v_{44} + v_{45}$, A[1][0][28]=v_{46}, A[1][2][28]=v_{47},
A[1][1][28]=$v_{46} + v_{47}$, A[1][2][29]=A[1][3][29]=v_{48}, A[3][0][8]=A[3][2][8]=v_{49},
A[1][2][30]=A[1][3][30]=v_{50}, A[3][0][15]=v_{51}, A[3][1][15]=v_{52}, A[3][2][15]=$v_{51} + v_{52}$,
A[1][1][31]=A[1][3][31]=v_{53}, A[3][0][0]=v_{54}, A[3][1][0]=v_{55}, A[3][2][0]=$v_{54} + v_{55}$,
A[3][0][1]=v_{56}, A[3][1][1]=v_{57}, A[3][2][1]=$v_{56} + v_{57}$, A[3][1][2]=A[3][2][2]=v_{58},
A[3][1][3]=A[3][2][3]=v_{59}, A[3][0][4]=v_{60}, A[3][1][4]=v_{61}, A[3][2][4]=$v_{60} + v_{61}$,
A[3][0][5]=v_{62}, A[3][1][5]=v_{63}, A[3][2][5]=$v_{62} + v_{63}$

Conditional cube variables

A[0][0][0]=A[1][0][31]=A[0][1][0]=A[2][1][30]=A[1][2][31]=A[2][2][30]=v_0

Bit condition

A[0][0][29]=A[2][0][28]+A[0][1][29]+A[1][1][29]+A[2][1][28]+A[0][2][29]
+ A[2][2][28]+A[0][3][29]+k_1[28]+k_4[29]+k_6[28]+1,
A[2][0][20]=A[4][0][19]+A[2][1][20]+A[4][1][19]+A[2][2][20]+A[4][2][19]
+ k_1[20]+k_2[20]+k_3[19]+k_6[20]+k_8[19],
A[0][0][17]=A[3][0][18]+A[0][1][17]+A[3][1][18]+A[0][2][17]+A[3][2][18]
+ A[0][3][17]+k_2[18]+k_4[17]+k_7[18]+k_8[18]+1,
A[0][0][20]=A[2][0][19]+A[0][1][20]+A[1][1][20]+A[2][1][19]+A[0][2][20]
+ A[2][2][19]+A[0][3][20]+k_1[19]+k_4[20]+k_6[19],
A[2][0][11]=A[4][0][10]+A[2][1][11]+A[4][1][10]+A[2][2][11]+A[3][2][11]
+ A[4][2][10]+k_1[11]+k_3[10]+k_6[11]+k_8[10],
A[4][0][14]=A[1][0][13]+A[4][1][14]+A[4][2][14]+ k_3[14]+k_4[14]+k_5[13]+k_8[14]+1,
A[2][1][26]=k_1[26],
A[0][0][3]=A[1][0][3]+A[2][0][2]+A[0][1][3]+A[2][1][2]+A[0][2][3]
+ A[2][2][2]+A[0][3][3]+k_1[2]+k_4[3]+k_6[2]+1,
A[0][0][26]=A[2][0][25]+A[0][1][26]+A[2][1][25]+A[0][2][26]+A[1][2][26]
+ A[2][2][25]+A[0][3][26]+k_1[25]+k_4[26]+k_6[25],
A[4][0][5]=A[1][0][4]+A[0][1][5]+A[4][1][5]+A[1][2][4]+A[4][2][5]+ k_3[5]+k_5[4]+k_8[5]+1,
A[3][0][25]=A[3][1][25]+A[1][2][26]+A[3][2][25]+ k_1[26]+k_2[25]+k_5[26]+k_7[25],
A[0][0][8]=A[3][0][9]+A[4][0][9]+A[0][1][8]+A[0][2][8]+ A[0][3][8]+k_2[9]+k_4[8]+k_7[9]+1

Table 8 continued

Guessed key bits
$k_1[28]+k_4[29]+k_6[28]+1$, $k_1[20]+k_2[20]+k_3[19]+k_6[20]+k_8[19]$,
$k_2[18]+k_4[17]+k_7[18]+k_8[18]+1$, $k_1[19]+k_4[20]+k_6[19]$,
$k_1[11]+k_3[10]+k_6[11]+k_8[10]$, $k_3[14]+k_4[14]+k_5[13]+k_8[14]+1$, $k_1[26]$,
$k_1[2]+k_4[3]+k_6[2]+1$, $k_1[25]+k_4[26]+k_6[25]$, $k_3[5]+k_5[4]+k_8[5]+1$,
$k_1[26]+k_2[25]+k_5[26]+k_7[25]$, $k_2[9]+k_4[8]+k_7[9]+1$

$0xef4bdff3$ $6ebf27ce$ $49b01de8$ $ac1e40c6$ $a47af7c9$ $35dc18cc$ $cf42d588$ $5df5c110$, and the right 12-bit guessed key bits are 001111110101.

5 Conditional cube attack on eight-round River Keyak using linear structure technique

When we attack the eight-round River Keyak using the conditional cube attack we need to find 64-dimension cube variables. However, it is so complex to search for 64 variables which meet the condition that these cube variables do not multiply with each other in the χ operation of the first round and the condition that there is one variable v_0 does not multiply with other cube variables in the second round. The probability is a little high for ordinary variables which do not multiply with each other in the first round, but in the second round, most of them would multiply with v_0. In order to solve this problem, we use the linear structure technique to find enough conditional cube variables for our eight-round attack.

We use the same conditional cube variables with that in the Sect. 4.1, as shown in Fig. 5, we choose the second column and the fourth column in yellow color to search the ordinary variables. Firstly, we let $A[1][3] = \oplus_{i=0}^{2} A[1][i]$ and $A[3][2] = \oplus_{i=0}^{1} A[1][i]$, and we could note after θ, ρ and π operations, these variables do not multiply with each other in the χ operation in the first round. Then, what we need to do is filtering the variables which do not multiply with v_0 in the second round. For each column, the bits which meet the second condition would be remained when the bits (do not multiply with v_0) equal or more than 2 in the same column. Following this method, the freedom degree for cube variables increase greatly, we could find more than 64 ordinary cube variables easily. We use the 64-dimension conditional cube variables listed in Table 8 to launch the eight-round conditional cube attack for River Keyak. The attack procedure is very similar to the six/seven-round attack. The complexity of our key-recovery attack is $2^{12} \times 2^{64} \times 32 = 2^{81}$.

6 Conclusion

In this paper, we comprehensively explore the security of River Keyak against the conditional cube attack. Firstly, we find a new conditional cube variable which diffuses much weaker than that used by Huang et al. Based on this conditional cube variable, we find 16/32 conditional cube variables for River Keyak. Then we launch the six/seven-round conditional cube attacks for River Keyak for the first time and give the experimental results for six/seven-round attacks. At last, by using linear structure technique, we propose a new method to find conditional cube variables with plenty of freedom degree, and we find enough 64-dimension cube variables to launch 8-round attack successfully. Those attacks are the first results on River Keyak and they do not threaten the full-round River Keyak.

Acknowledgements This work is supported by China's 973 Program (No. 2013CB834205), the National Key Research and Development Program of China (No. 2017YFA0303903), the National Natural Science Foundation of China (No. 61672019), the Fundamental Research Funds of Shandong University (No. 2016JC029).

References

1. Aumasson, J.P., Dinur I., Meier W., Shamir A. : Cube testers and key recovery attacks on reduced-round MD6 and trivium. In: FSE 2009, LNCS, vol. 5665, pp. 1–22. Springer, Heidelberg (2009).
2. Bertoni G., Daemen J., Peeters M., Van Assche G.: Keccak sponge function family. http://keccak.noekeon.org/.
3. CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness. http://competitions.cr.yp.to/caesar.html.
4. Chaigneau C., Fuhr T., Gilbert H., Jean J., Reinhard J.R.: Cryptanalysis of NORX v2. 0. IACR Trans. Symmetric Cryptol. doi:10.13154/tosc.v2017.i1.156-174 (2017).
5. Das S., Meier W.: Differential biases in reduced-round Keccak. In: AFRICACRYPT 2014, LNCS, vol. 8469, pp. 69–87. Springer, Heidelberg (2014).
6. Daemen J., Van Assche G. :Differential propagation analysis of Keccak. In: FSE 2012, pp. 422–441. Springer, Heidelberg (2012).
7. Dinur I., Shamir A.: Cube attacks on tweakable black box polynomials. In: EUROCRYPT 2009, LNCS, vol. 5479, pp. 278–299. Springer, Heidelberg (2009).
8. Dinur I., Shamir A. : Breaking Grain-128 with dynamic cube attacks. In: FSE 2011, LNCS, vol. 6733, pp. 167–187. Springer, Heidelberg (2011).
9. Dinur I., Morawiecki P., Pieprzyk J., Srebrny M., Straus M.: Cube attacks and cube-attack-like cryptanalysis on the round-reduced Keccak sponge function. In: EUROCRYPT 2015, Part I, LNCS, vol. 9056, pp. 733–761. Springer, Heidelberg (2015)
10. Dong X., Li Z., Wang X., Qin L.: Cube-like attack on round-reduced initialization of Ketje Sr. IACR Trans. Symmetric Cryptol. doi:10.13154/tosc.v2017.i1 (2017).
11. Fouque P.A., Vannet T.: Improving key recovery to 784 and 799 rounds of trivium using optimized cube attacks. In: FSE 2013, LNCS, vol. 8424, pp. 502–517. Springer, Heidelberg (2014).
12. Guido B., Joan D., Michaël P., Assche G.V.: Keyak. http://keyak.noekeon.org.
13. Guido B., Joan D., Michaël P., Assche G.V.: The Keccak SHA-3 Submission. Submission to NIST (Round 3) (2011).
14. Guo J., Liu M., Song L.: Linear structures: applications to cryptanalysis of round-reduced Keccak. In: ASIACRYPT 2016, Part I, LNCS, vol. 10031, pp. 249–274. Springer, Heidelberg (2016).
15. Huang S., Wang X., Xu G., Wang M., Zhao J.: Conditional cube attack on reduced-round Keccak sponge function. In: EUROCRYPT 2017, Part II, LNCS, vol. 10211, pp. 259–288. Springer, Heidelberg (2017).
16. Li Z., Dong X., Wang X.: Conditional cube attack on round-reduced ASCON. IACR Trans. Symmetric Cryptol. doi:10.13154/tosc.v2017.i1 (2017).
17. NIST FIPS.197: Advanced Encryption Standard (AES). http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf (2001).