

# Related-Tweakey Impossible Differential Attack on Reduced-Round Deoxys-BC-256

Rui ZONG<sup>1</sup>, Xiaoyang DONG<sup>2\*</sup> & Xiaoyun WANG<sup>1,2\*</sup>

<sup>1</sup>*Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan 250100, China;*

<sup>2</sup>*Institute for Advanced Study, Tsinghua University, Beijing 100084, China*

---

**Abstract** Deoxys-BC is the internal tweakable block cipher of Deoxys, a third-round authenticated encryption candidate at the CAESAR competition. In this study, by adequately studying the tweakey schedule, we seek a six-round related-tweakey impossible distinguisher of Deoxys-BC-256, which is transformed from a 3.5-round single-key impossible distinguisher of AES, by application of the mixed integer linear programming (MILP) method. We present a detailed description of this interesting transformation method and the MILP-modeling process.

Based on this distinguisher, we mount a key-recovery attack on 10 (out of 14) rounds of Deoxys-BC-256. Compared to previous results that are valid only when the key size  $> 204$  and the tweak size  $< 52$ , our method can attack 10-round Deoxys-BC-256 as long as the key size  $\geq 174$  and the tweak size  $\leq 82$ . For the popular setting in which the key size is 192 bits, we can attack one round more than previous works.

Note that this work only gives a more accurate security evaluation and does not threaten the security of full-round Deoxys-BC-256.

**Keywords** related-tweakey impossible differential attack, tweakable block cipher, Deoxys-BC-256, tweakey schedule, MILP

---

**Citation** Author A, Author B, Author C, et al. Title for citation. Sci China Inf Sci, for review

---

## 1 Introduction

To satisfy the growing demand for authenticated encryption, the CAESAR competition [1] was launched in 2013 by the international cryptologic research community. The competition has three rounds. In March 2014, the first-round competition received 57 submissions; in July 2015, 30 candidates were chosen during the second round of the competition; and in August 2016, 15 candidate ciphers were selected during the third round. The final winner will be announced at a later date from amongst the third-round competition candidates.

Deoxys [2] is one of the 15 authenticated encryption candidates of the CAESAR third-round competition. The design of Deoxys is based on a tweakable block cipher Deoxys-BC, using the well-studied AES [3] round function as a building block.

The concept of a tweakable block cipher (TBC) was first proposed by Ronald L. Rivest and David Wagner [4] in 2002. In addition to the secret key and a plaintext, a tweakable block cipher employs a third input: the tweak, which can be public, to yield a ciphertext. Its design is mainly motivated to solve the problem that for a traditional block cipher, when encrypted by the same key even in different

---

\* Corresponding author (email: xiaoyangdong@mail.tsinghua.edu.cn, xiaoyunwang@mail.tsinghua.edu.cn)

cases, the plaintext will be transformed into a fixed ciphertext. There have been many TBCs, including Skinny [5], PRINCE [6], and QARMA [7].

In contrast to many tweakable block constructions that take a known permutation or a block cipher as a black box and use the tweak as an independent input, Deoxys-BC adopts the TWEAKEY framework [8], which provides a unified view of the key and the tweak, denoted by tweakey. This means that when given the public round permutation (for instance, the AES round function), the tweakable block cipher can be a primitive with arbitrary tweak and key sizes. For ciphers that adopt this framework, a dedicated tweakey schedule will use the  $(k+t)$ -bit tweakey, composed of a  $k$ -bit key ( $k$  can be almost any value) and a  $t$ -bit tweak, to produce  $n$ -bit round subtweakeys. For Deoxys-BC, there are two versions: Deoxys-BC-256 with a 256-bit tweakey and Deoxys-BC-384 with a 384-bit tweakey. The subtweakey size is 128 bits for both versions. In Deoxys, the size of the key and the tweak can vary within the tweakey length as long as the key size is longer than or equal to the block size, i.e., 128 bits.

**Related Work** At FSE 2018, a work [9] using related-tweakey rectangle attacks to analyze both Deoxys-BC-256 and Deoxys-BC-384 was presented. Compared to the security evaluation given by the designer, the work in [9] improved the number of analyzed rounds by two for Deoxys-BC-256 and five for Deoxys-BC-384. These attacks greatly improved the related-tweakey differential bounds provided by the designers.

**Table 1** Cryptanalysis Results for Deoxys-BC-256. Our attack can be mounted on Deoxys-BC-256 with a wider key size range. Since the cipher adopts the TWEAKEY framework (such as the tweak-updating mode, i.e., the tweak can be changed but the key stays the same), our attack is more efficient as the data complexity can be beyond full-codebook. A beyond-full-codebook attack on SKINNY was published in [11]; we give a more specified description about beyond-full-codebook attacks in subsection 2.2.

Primitive	Number of Rounds	Tweak Size	Key Size	Time	Data	Attack Type	Ref.
Deoxys-BC-256	8/14	128	128	$\leq 2^{128}$	-	MitM	[2]
	$\leq 8/14$	128	128	$\leq 2^{128}$	-	Differential	[2]
	9/14	128	128	$2^{128}$	$2^{117}$	Rectangle	[9]
	10/14	$< 52$	$> 204$	$2^{204}$	$2^{127.58}$	Rectangle	[9]
	10/14	$\leq 82$	$\geq 174$	$2^{173.1}$	$2^{135}$	Impossible Differential	This Paper

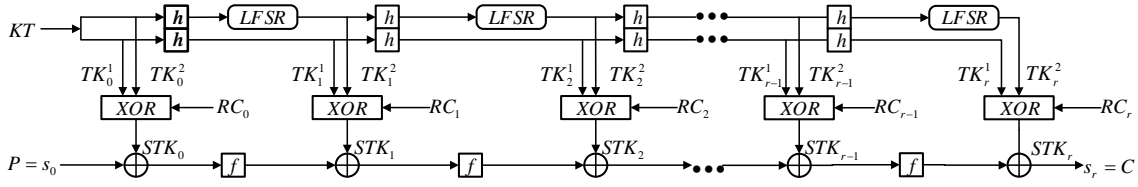
**Our Contribution:** In this study, we analyze Deoxys-BC-256 against impossible differential attacks and give a more accurate security evaluation of 10-round (out of 14-round) Deoxys-BC-256. First, we describe a method that can derive longer related-key impossible distinguishers from single-key impossible distinguishers. Second, using this method, after an adequate study of the tweakey schedule, we build an MILP model of six-round Deoxys-BC-256 and find a six-round related-tweakey impossible distinguisher. Finally, based on this distinguisher, we mount an attack including 10-round Deoxys-BC-256. The attack needs a time complexity of  $2^{173.1}$  10-round encryptions and a data complexity of  $2^{135}$  plaintexts. Compared to previous results, the attack applies to a wider range of key sizes and can attack one more round on Deoxys-BC-256 with a key size of 192 bits, thus providing a more accurate security evaluation. All analysis results are shown in Table 1.

## 2 Preliminaries

First, we give a detailed description of Deoxys-BC-256; second, we present a validation of the beyond-full-codebook impossible attack on Deoxys-BC-256. Next, we discuss some useful propositions and the notations used in this paper.

### 2.1 Description of Deoxys-BC-256

In this section, we recall the details of the Deoxys-BC-256 block cipher. We assume that the reader is familiar with the AES block cipher [3]. Figure 1 shows the structure of Deoxys-BC-256.


**Figure 1** Structure of Deoxys-BC-256

Deoxys-BC is the internal ad-hoc tweakable block cipher of the Deoxys authenticated encryption scheme, conforming to the TWEAKEY framework [8]. Except for the two standard inputs of a block cipher, i.e., a plaintext  $P$  and a key  $K$ , this cipher adopts a third input called a tweak  $T$ , i.e.,  $E_K(T, P) = C$ . According to the TWEAKEY framework, we can use a single input, called the tweakey, to provide a unified view of the tweak and the key. The length of the tweakey is the cumulative size of the key and the tweak. For Deoxys-BC-256, the tweakey size is 256 bits; for Deoxys-BC-384, the tweakey size is 384 bits. In this paper, we focus on Deoxys-BC-256. For more information, we refer to [2].

Deoxys-BC is an AES-like design, i.e., it is an iterative substitution-permutation network (SPN) that transforms the plaintext through a certain number of round functions (that depend on the tweakey) to a ciphertext. As the Deoxys-BC cipher uses the AES round function, we can represent the internal state as a  $4 \times 4$  matrix of bytes. The corresponding index is

$$InternalState = (0, 1, 2, 3, \dots, 14, 15) = \begin{pmatrix} 0 & 4 & 8 & 12 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \end{pmatrix}.$$

**Deoxys-BC-256 round function.** The round function, similar to AES, has four operations applied to the internal state, as follows:

- **AddRoundTweakey(AK)** - XOR the 128-bit round subtweakey (defined further) to the internal state,
- **SubBytes(SB)** - Apply the 8-bit Sbox  $\mathbf{S}$  of AES [3] to each of the 16 bytes of the internal state <sup>1)</sup>,
- **ShiftRows(SR)** - Rotate the 4-byte  $i$ -th row left by  $\rho[i]$  positions, where  $\rho=(0,1,2,3)$ ,
- **MixColumns(MC)** - Multiply the internal state by the  $4 \times 4$  constant MDS matrix  $\mathbf{M}$  defined below whose coefficients lie in  $\mathbb{K}$  <sup>2)</sup>. The matrix  $\mathbf{M}$  and the inverse matrix  $\overline{\mathbf{M}}$  are shown as follows:

$$\mathbf{M} = \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix} \quad \overline{\mathbf{M}} = \begin{pmatrix} 14 & 11 & 13 & 9 \\ 9 & 14 & 11 & 13 \\ 13 & 9 & 14 & 11 \\ 11 & 13 & 9 & 14 \end{pmatrix}$$

After the last round, a final  $AK$  operation is performed to produce the ciphertext.

**Definition of the Subtweakey.** The structure of the tweakey schedule distinguishes Deoxys-BC from the classical construction of an AES-like block cipher.

We denote the concatenation of the key  $K$  and the tweak  $T$  as  $KT$ , i.e.,  $KT = K||T$ . Then, the tweakey state is divided into 128-bit words. For Deoxys-BC-256, the size of  $KT$  is 256 bits, with the first (most significant) 128 bits of  $KT$  denoted as  $KT^2$  and the second as  $KT^1$ .

1) the specified detail of the Sbox is not presented as it does not influence the analysis process in this paper.

2)  $\mathbb{K}$  denotes the base field as  $GF(2^8)$  defined by the irreducible polynomial  $x^8 + x^4 + x^3 + x + 1$ . This is the base field used in AES.

A subtweakey of the  $i$ -th round is defined as

$$STK_i = TK_i^1 \oplus TK_i^2 \oplus RC_i. \quad (1)$$

The 128-bit words  $TK_i^2$  and  $TK_i^1$  are outputs produced by a special tweakey schedule algorithm, initialized with  $TK_0^1 = KT^1$  and  $TK_0^2 = KT^2$ . The tweakey schedule algorithm is defined as

$$TK_{i+1}^1 = h(TK_i^1), TK_{i+1}^2 = h(LFSR(TK_i^2)), \quad (2)$$

The  $h$  operation, shown in Table 2, is a simple byte permutation.

**Table 2**  $h$  - Permutation

$i$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$h(i)$	1	6	11	12	5	10	15	0	9	14	3	4	13	2	7	8

The  $LFSR$  function is simply the application of a linear feedback shifting register to each of the 16 bytes of a tweakey 128-bit word, i.e.,

$$LFSR(x_7||x_6||x_5||x_4||x_3||x_2||x_1||x_0) = (x_6||x_5||x_4||x_3||x_2||x_1||x_0||x_7 \oplus x_5).$$

Finally,  $RC_i$  are the round constants of the tweakey schedule, and are defined as

$$RC_i = \begin{pmatrix} 1 & RCON[i] & 0 & 0 \\ 2 & RCON[i] & 0 & 0 \\ 4 & RCON[i] & 0 & 0 \\ 8 & RCON[i] & 0 & 0 \end{pmatrix}$$

where  $RCON[i]$  denotes the  $i$ -th key schedule constants of the AES.

**Deoxys:** Deoxys is an authenticated encryption design based on Deoxys-BC. It has two modes: Deoxys-I, a nonce-based authenticated encryption scheme to be used in a nonce-respecting setting; and Deoxys-II, a nonce-based authenticated encryption scheme that can provide security even in a nonce-misuse setting.

With the recommended parameters, when instantiated with the Deoxys-BC-256 block cipher, the two modes lead to a 128-bit key version (denoted as Deoxys-I-128-128 and Deoxys-II-128-128). For more information about Deoxys, we refer to the Deoxys document [2].

## 2.2 Beyond Full-Codebook

Recall that a tweakable block cipher takes as its input a key (of fixed length  $n$ ) and a tweak (of fixed length  $t$ ). The TWEAKEY framework [8] offers further flexibility in setting the limit of data resources for an attack. For ciphers adopting the TWEAKEY framework, such as Deoxys-BC-256, one can add a tweak of almost any length and/or extend the key space of the block cipher to (almost) any size as long as the key space and the tweak space are suitable for the tweakey schedule. This provides attackers with a potentially optimal strategy to attack the ciphers: select the key size as large as possible, which results in a higher security claim, as long as the size of the tweak is large enough to supply the required data. Thus, in this scenario, an attack can be valid even if the data complexity is beyond full-codebook, and an attack is more difficult when the key size is smaller (the size range is wider).

In fact, beyond-full-codebook attacks have been shown to be realistic and powerful. In [10], the authors analyze the NIST standard for Format-Preserving Encryption with beyond-full-codebook attacks and exploit the fact that the cipher is a Feistel-based tweakable block cipher. In [11], several beyond-full-codebook attacks on different versions of SKINNY are presented.

In [9], the authors discuss beyond-full-codebook attacks for tweakable block ciphers. They also examine beyond-codebook rectangle attacks on Deoxys-BC. However, the beyond-full-codebook rectangle attack

is complex and may be impossible as the sufficiently large plaintext/tweak space also provides too many wrong pairs that probabilistically satisfy the same input and output differences without following the characteristic. However, when considering the impossible differential attack, this problem does not exist because the differential propagates with probability 1 or 0.

### 2.3 Some Propositions

**Proposition 1** (Differential Property of Sbox, [12]). Given the nonzero input and output differences of an Sbox, there exists only one pair of actual values on average to satisfy these two differences.

**Proposition 2** (The 3.5-Round Single-Key Impossible Distinguisher of AES [13]). Consider 3.5-round AES encryption which omits the last *MC* operation. If a pair of plaintexts differ by only one byte, then the ciphertexts cannot be equal in any of the following combinations of bytes: (0,5,10,15), (3,4,9,14), (2,7,8,13), or (1,6,11,12).

**Proof:** If the plaintexts differ only in one byte, they will be active in all four bytes of one column after the first *MC* operation. Then, after the second *MC*, the difference will be active in all bytes. On the other hand, if the ciphertexts are equal in one of the four prohibited combinations of bytes, after the third *MC*, the data will be equal in one column. Thus, before the third *MC*, the data in this column is also equal. Therefore, after the second *MC*, there are at least 4 bytes in which the data are equal. This is a contradiction since all bytes of the data differ after the *MC* in the forward direction. Therefore, this is impossible.

One possible case is illustrated in Figure 2.

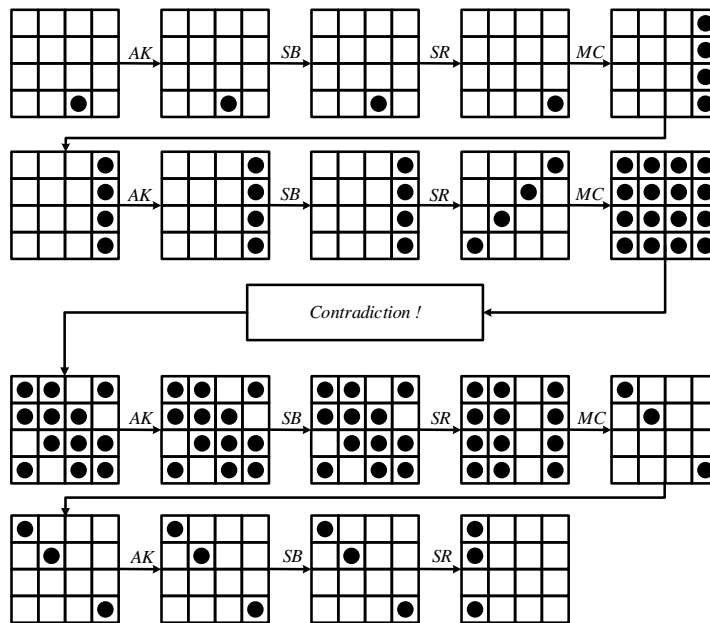


Figure 2 A 3.5-round Single-Key Impossible Distinguisher of AES

**Proposition 3** (Subtweakey Difference Cancellation). As noticed by the designers [2], using the simple LFSR given in subsection 2.1, a single subtweakey difference cancellation can occur every 15 rounds for Deoxys-BC-256. Suppose that a single cell of  $TK^1$  and  $TK^2$  are active. Let  $a_1$  and  $a_2$  be differences of the active cells, respectively. Then, the subtweakey difference of the first round is  $a_2 \oplus a_1$  at this cell, and in the  $i$ -th round, the subtweakey difference is  $a_2 \oplus LFSR^i(a_1)$ , ignoring the position permutation  $h$ . Since  $a_1$  and  $a_2$  are both nonzero differences,  $a_2 \oplus LFSR^i(a_1) = 0$  can occur once every 15 rounds.

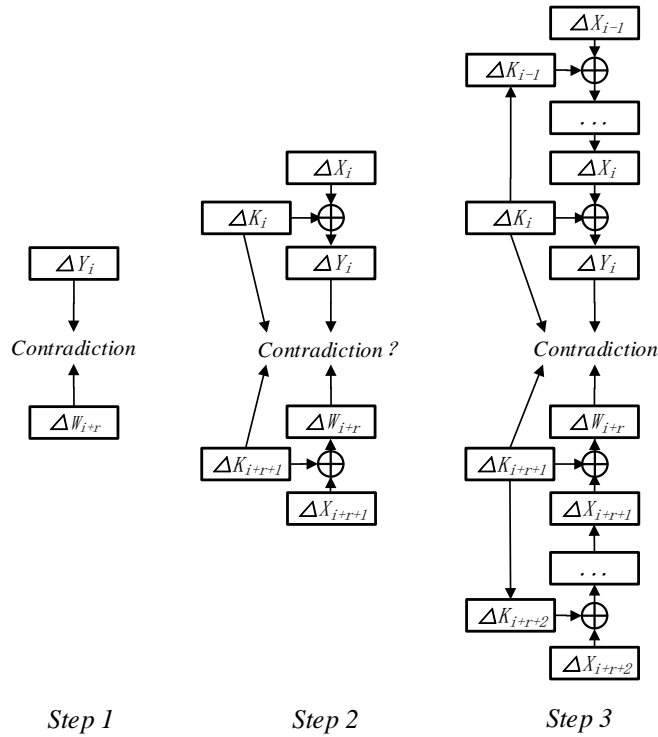
## 2.4 Notation

- $X_i$  represents the internal state after  $AK$  in round  $i$ .
- $Y_i$  represents the internal state after  $SB$  in round  $i$ .
- $Z_i$  represents the internal state after  $SR$  in round  $i$ .
- $W_i$  represents the internal state after  $MC$  in round  $i$ .
- $\Delta S$  represents the difference value of  $S$  and  $S'$ .
- $(S, S')$  represents a pair of internal states where  $S \oplus S' = \Delta S$
- $X[j]$  represents the  $j$ -th byte of  $X$ .
- $\bar{F}$  represents the inverse function of  $F$ .

## 3 Attack on Deoxys-BC-256

### 3.1 Longer Related-Key Impossible Distinguisher

In this section, we explain the process of extending a single-key impossible differential to a longer related-key impossible differential.



**Figure 3** Search for Longer Related-Key Differential

First, suppose we already obtained a  $r$ -round single-key impossible differential of a cipher:

$$\Delta Y_i = \Delta in \rightarrow \Delta out = \Delta W_{i+r},$$

Then, we think about the related-key scenario. When we set  $\Delta X_i = 0$  and  $\Delta K_i = \Delta in$ , after the  $AK$  operation, the internal state difference  $\Delta Y_i$  is also  $\Delta in$  as  $\Delta Y_i = \Delta X_i \oplus \Delta K_i$ . Similarly, when we set  $\Delta X_{i+r+1} = 0$  and  $\Delta K_{i+r+1} = \Delta out$ , then  $\Delta W_{i+r} = \Delta X_{i+r+1} \oplus \Delta K_{i+r+1} = \Delta out$ . Now, we get the input and output difference of the original single-key distinguisher and check whether  $\Delta K_i \rightarrow \Delta K_{i+r+1}$

is possible. Notice that not only  $\Delta Y_i$  and  $\Delta W_{i+r}$  but also the key difference from  $\Delta K_i$  to  $\Delta K_{i+r+1}$  will influence the position of active nibbles from  $\Delta Y_i$  and  $\Delta W_{i+r}$ . We need to check whether the contradiction still holds. If it does, we go to the next step; otherwise, we choose another single-key impossible differential and check in the same way.

If contradictions still exist, we can add one round both on the top and on the bottom of the distinguisher. According to the key schedule, we deduce  $\Delta K_{i-1}$  and  $\Delta K_{i+r+2}$  from  $\Delta K_i$  and  $\Delta K_{i+r+1}$ , respectively. After that, we set  $\Delta X_{i-1} = \Delta K_{i-1}$  and  $\Delta X_{i+r+2} = \Delta K_{i+r+2}$ . This will ensure  $\Delta X_i = 0$  and  $\Delta X_{i+r+1} = 0$ . Thus, we get the input and output difference of the distinguisher in the previous step.

Now, we extend the original differential by two more rounds. If possible, we can continue to extend more rounds in the same way.

### 3.2 Search for Related-Tweakey Impossible Distinguisher with MILP method

MILP problems are mathematical optimization problems in which only some variables are constrained to be integers. The goal is to find the optimal value that minimizes/maximizes the objective function satisfying all of the inequality constraints. This was introduced in [14] and [15] and improved in [16–19].

In [20] and [21], two different automatic tools for searching impossible differentials with the MILP method are presented. However, the tool in [20] is not suitable for the related-key setting. In [21], the authors regard searching for the single-key impossible differential and the related-key impossible differential as two completely independent processes. By contrast, our method looks for related-key impossible differentials and tries to derive the relation between the single-key impossible differentials and the related-key impossible differentials.

By using the method described in subsection 3.1 and the property of the tweakey schedule in Proposition 3, we extend two more rounds in the bottom of the single-key distinguisher in Figure 2 and seek out a six-round related-key distinguisher shown in Figure 4.

Next, we describe the modeling process.

#### Constraints for the Tweakey Schedule

When considered only at the byte level, the subtweakey schedule of two successive rounds is just a byte permutation. We use  $stk_i[j]$  ( $stk_{i+1}[j']$ ) to denote the activeness of the corresponding relevant bytes in  $STK_i$  ( $STK_{i+1}$ ). The constraint is:

$$stk_i[j] - stk_{i+1}[j'] = 0, j' = h(j).$$

When considered at the bit level, the subtweakey differences should satisfy several conditions:

1) As shown in  $R1$  of Figure 4, the  $MC$  operation is a 2-to-3 transformation from  $Z_1$  to  $W_1$ , and  $\Delta W_1[8, 9] = \Delta STK_2[8, 9]$ . To satisfy these two conditions,  $3 \times \Delta STK_2[8]$  should be equal to  $\Delta STK_2[9]$ . As these operations are all in a finite field  $\mathbb{K}$  (the irreducible polynomial is  $x^8 + x^4 + x^3 + x + 1$ ), we can use eight variables to denote the 8-bit variable  $\Delta STK_i[j]$ .

For example,  $(a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0)$  denotes

$$\Delta STK_2[8] = a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0,$$

then

$$\begin{aligned} & 3 \times \Delta STK_2[8] \\ &= (x + 1) \cdot (a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0) \\ &= (a_6 \oplus a_7)x^7 + (a_5 \oplus a_6)x^6 + (a_4 \oplus a_5)x^5 + (a_3 \oplus a_4 \oplus a_7)x^4 + \\ & \quad (a_2 \oplus a_3 \oplus a_7)x^3 + (a_1 \oplus a_2)x^2 + (a_0 \oplus a_1 \oplus a_7)x + (a_0 \oplus a_7). \end{aligned} \quad (3)$$

$(b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0)$  denotes

$$\Delta STK_2[9] = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0,$$

Then,

$$3 \times \Delta STK_2[8] = \Delta STK_2[9]$$

can be represented by

$$\begin{aligned} a_6 \oplus a_7 &= b_7, a_5 \oplus a_6 = b_6, \\ a_4 \oplus a_5 &= b_5, a_3 \oplus a_4 \oplus a_7 = b_4, \\ a_2 \oplus a_3 \oplus a_7 &= b_3, a_1 \oplus a_2 = b_2, \\ a_0 \oplus a_1 \oplus a_7 &= b_1, a_0 \oplus a_7 = b_0. \end{aligned} \quad (4)$$

These equations can be described by MILP constraints. For example,  $a_6 \oplus a_7 = b_7$  can be restrained by

$$a_6 + a_7 + b_7 - 2d_{\oplus} = 0, \quad (5)$$

where  $d_{\oplus}$  is a dummy bit variable.

$a_3 \oplus a_4 \oplus a_7 = b_4$  can be restrained by

$$\begin{aligned} -a_3 + a_4 + a_7 + b_4 &\geq 0, \quad a_3 - a_4 + a_7 + b_4 \geq 0, \\ a_3 + a_4 - a_7 + b_4 &\geq 0, \quad a_3 + a_4 + a_7 - b_4 \geq 0, \\ a_3 - a_4 - a_7 - b_4 &\geq -2, \quad -a_3 + a_4 - a_7 - b_4 \geq -2, \\ -a_3 - a_4 + a_7 - b_4 &\geq -2, \quad -a_3 - a_4 - a_7 + b_4 \geq -2. \end{aligned} \quad (6)$$

The other equations in (3) can be described as MILP constraints in a similar way to (4) and (5).

2) The difference of  $W_5$  is equivalent to the difference of  $STK_6$ ; and in the backward direction,  $W_4$  has at most three active columns. Thus, after a  $MC$  operation, the corresponding column of  $Z_5$  has at most 3 active bytes. What's more, after a  $\overline{SR}$  and  $\overline{SB}$  operation in  $R_5$ , the positions of active bytes in  $STK_5$  and  $X_5$  occupy at most three columns. For example,  $\Delta Z_5[2] = 0$  is equivalent to  $13 \times \Delta W_5[0] = 9 \times \Delta W_5[1]$ . The modeling process is similar to that in Step 1.

3) According to Proposition 3, we can set the 2 bytes in  $R_7$  inactive, so the corresponding two values should be zero. This means  $\Delta TK_7^1[1] = \Delta TK_7^2[1]$  and  $\Delta TK_7^1[6] = \Delta TK_7^2[6]$ . The bit-level modeling process is simple and similar to that in Step 1.

**Constraints for AK.** We use  $(w_{i-1}[j], stk_i[j], x_i[j])$  to denote the activeness of the corresponding relevant bytes in  $(W_{i-1}, STK_i, X_i)$ . Then, for  $(w_{i-1}[j], stk_i[j], x_i[j])$ , the possible values are  $(0, 0, 0), (0, 1, 1), (1, 0, 1), (1, 1, 0)$ , and  $(1, 1, 1)$ . We can use the following inequalities to include all five solutions:

$$\begin{aligned} w_{i-1}[j] + stk_i[j] - x_i[j] &\geq 0 \\ w_{i-1}[j] - stk_i[j] + x_i[j] &\geq 0 \\ -w_{i-1}[j] + stk_i[j] + x_i[j] &\geq 0. \end{aligned}$$

**Constraints for SB.** As the  $SB$  operation does not change the activeness of a byte, it is equivalent to  $y_i[j] - x_i[j] = 0$ .

**Constraints for SR.** This operation is a byte permutation and does not change the activeness of bytes either. We use  $z_i[j'] - y_i[j] = 0$  to denote  $SR$  with  $j = SR(j')$ .

**Constraints for MC.** Modeling the  $MC$  operation is essentially expressing the transformation property of the MDS matrix with branch number 5. For all input and output differences, except the case where all of them are zero, the number of active bytes is at least 5. Then, the number of solutions is  $2^8 - C_8^1 - C_8^2 - C_8^3 - C_8^4 = 94$ . We use  $(z_i[0], z_i[1], z_i[2], z_i[3], w_i[0], w_i[1], w_i[2], w_i[3])$  to denote the activeness of the eight input and output differences. The solutions can be denoted by the following inequalities:

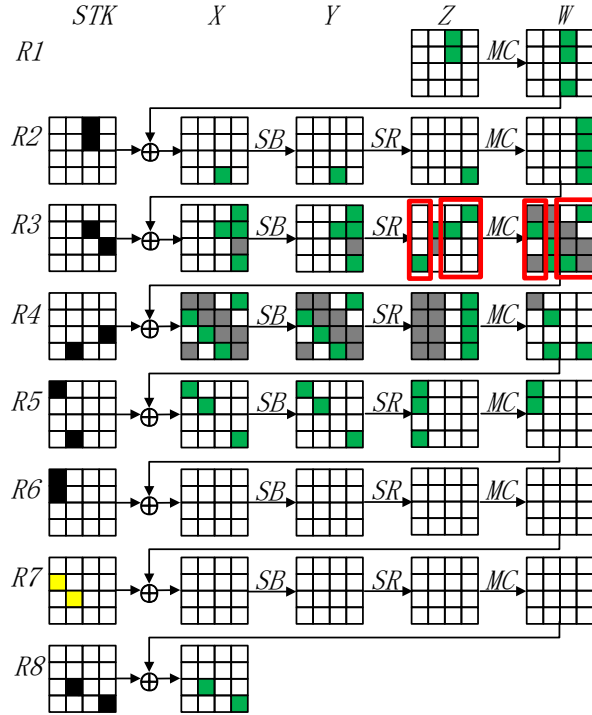
$$-4 \times z_i[0] + z_i[1] + z_i[2] + z_i[3] + w_i[0] + w_i[1] + w_i[2] + w_i[3] \geq 0$$



$$\begin{aligned}
 z_i[0] - 4 \times z_i[1] + z_i[2] + z_i[3] + w_i[0] + w_i[1] + w_i[2] + w_i[3] &\geq 0 \\
 z_i[0] + z_i[1] - 4 \times z_i[2] + z_i[3] + w_i[0] + w_i[1] + w_i[2] + w_i[3] &\geq 0 \\
 z_i[0] + z_i[1] + z_i[2] - 4 \times z_i[3] + w_i[0] + w_i[1] + w_i[2] + w_i[3] &\geq 0 \\
 z_i[0] + z_i[1] + z_i[2] + z_i[3] - 4 \times w_i[0] + w_i[1] + w_i[2] + w_i[3] &\geq 0 \\
 z_i[0] + z_i[1] + z_i[2] + z_i[3] + w_i[0] - 4 \times w_i[1] + w_i[2] + w_i[3] &\geq 0 \\
 z_i[0] + z_i[1] + z_i[2] + z_i[3] + w_i[0] + w_i[1] - 4 \times w_i[2] + w_i[3] &\geq 0 \\
 z_i[0] + z_i[1] + z_i[2] + z_i[3] + w_i[0] + w_i[1] + w_i[2] - 4 \times w_i[3] &\geq 0.
 \end{aligned}$$

In order to satisfy the second constraint of the tweakkey schedule, we need to add one more inequality for the  $MC$  in  $R5$ :

$$z_i[0] + z_i[1] + z_i[2] + z_i[3] + w_i[0] + w_i[1] + w_i[2] + w_i[3] \leq 5.$$



**Figure 4** Related-Key Impossible Distinguisher of Deoxys-BC-256. Green nibbles indicate active bytes of cipher internal states, white nibbles indicate inactive bytes of internal states, gray nibbles indicate uncertain bytes, black nibbles indicate active bytes of the subtweakeys, and yellow nibbles indicate inactive bytes because of Proposition 3. Red boxes indicate contradictions of this distinguisher.

We add all the above constraints into the final model, and set the position of active bytes of the input and the output as fixed. If the returned result we get is '*infeasible*', then the fixed differential is impossible. Finally, we get the impossible differential shown in Figure 4.

In the forward direction from  $Z_1$  to  $Z_3$ , the number of active bytes of the first and last two columns of  $Z_3$  is 1; in the backward direction from  $X_8$  to  $W_3$ , the corresponding number in  $W_3$  is 3. This is inconsistent with the transform property of an MDS matrix with branch number 5 as  $1 + 3 = 4 < 5$ .

As the MILP modeling process of the subtweakey difference values is at the bit level and the constraints are strict, as a proof of work, we provide actual values of the differential characteristics conforming to the distinguisher in Table 3.

**Table 3** Actual Values of Impossible Differential Conforming the Distinguisher

Round	Index	$\Delta TK^1$	$\Delta TK^2$	$\Delta STK$	$\Delta X$	$\Delta Z$	$\Delta W$
1	8					0x3d	0x3d
	9					0x3d	0x47
	10					0x00	0x00
	11					0x00	0x7a
2	8	0x66	0x5b	0x3d			
	9	0x36	0x71	0x47			
	11				0x7a		
3	9	0x66	0xb6	0xd0			
	14	0x36	0xe3	0xd5			
4	14	0x66	0x6c	0x0a			
	7	0x36	0xc6	0xf0			
5	7	0x66	0xd9	0xbf			
	0	0x36	0x8d	0xbb		0x3e	0xd5
	1					0xba	0x2d
	2					0x00	
	3					0x7c	
6	0	0x66	0xb3	0xd5			
	1	0x36	0x1b	0x2d			
7	1	0x66	0x66	0x00			
	6	0x36	0x36	0x00			
8	6	0x66	0xcd	0xab	0xab		
	15	0x36	0x6d	0x5b	0x5b		

### 3.3 Attack Process

We add two rounds both on the top and the bottom of the distinguisher in subsection 3.2, and successfully mount a 10-round key recovery attack on Deoxys-BC-256, as shown in Figure 5.

In order to better attack the bottom two rounds, we bring forward the  $AK$  operation before the  $SR$  operation of the previous round, and use  $ruK_i$  to denote  $\overline{SR}(MC(STK_i))$ . To distinguish the original order, we denote the internal state in  $R9$  and  $R10$  as  $X \xrightarrow{SB} Y \xrightarrow{ruK} ruX \xrightarrow{SR} ruZ \xrightarrow{MC} ruW$ . Notice that like the difference value of  $STK$ , all differences of  $ruK$  are fixed and known.

The attack process is as follows:

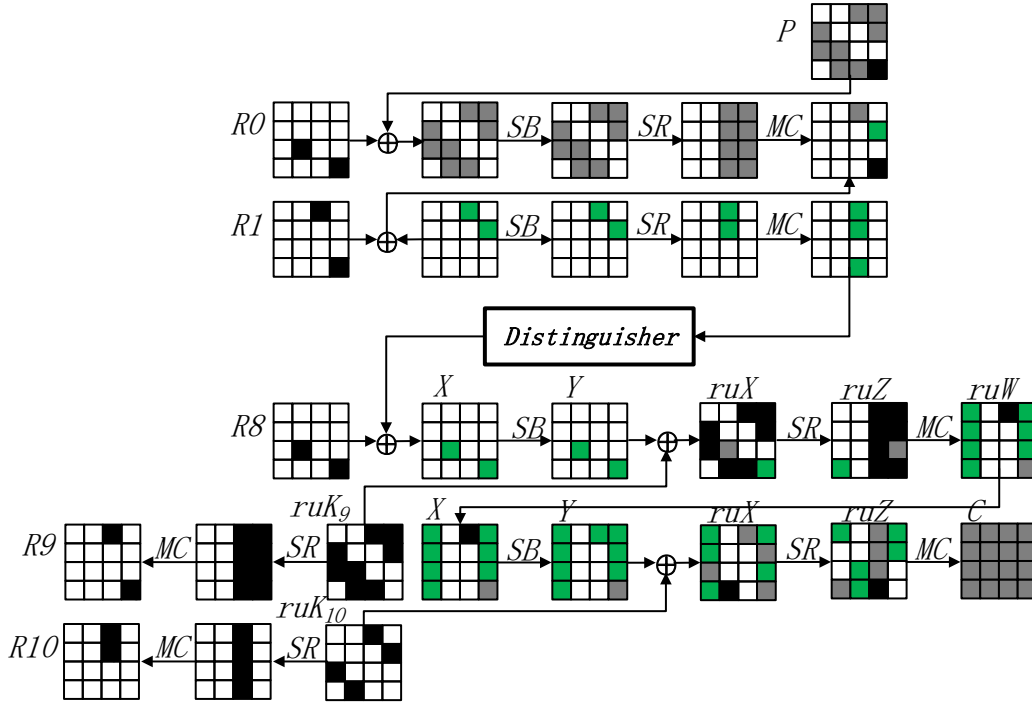
(1) Construct  $2^n$  structures in which each structure is made up of  $2^{64}$  plaintexts. In each structure, we set  $\Delta P[15] = \Delta STK_0[15]$  and  $\Delta P[1, 2, 6, 7, 8, 11, 12, 13]$  for the 8 active bytes. Then, each structure will provide  $2^{128}$  pairs.

(2) Choose  $(KT, KT')$  in that the tweakey difference satisfies the subtweakey difference trail in Figure 5. Encrypt the plaintexts under two tweakeys, and only choose the pairs that satisfy  $\overline{MC}(\Delta C)[1, 2, 4, 5, 14, 15] = 0$  and  $\overline{MC}(\Delta C)[11] = \Delta ruK_{10}[7]$ . In total, we get  $2^{72+n}$  pairs.

(3) For each of the remaining pairs, perform the following steps:

(3.1) Guess the value of  $\Delta W_0[8, 13]$ . Since  $\Delta W_0[15] = \Delta STK_1[15]$  and  $\Delta STK_1[15]$  is known, we can deduce the value of  $\Delta Z_0[8, 9, 10, 11, 12, 13, 14, 15]$  by a  $\overline{MC}$  operation. The value of  $\Delta Y_0[1, 2, 6, 7, 8, 11, 12, 13]$  is also known as  $Y_0[1, 2, 6, 7, 8, 11, 12, 13] = \overline{SR}(Z_0[8, 9, 10, 11, 12, 13, 14, 15])$ . Moreover, we can deduce the value of  $\Delta X_0[1, 2, 6, 7, 8, 11, 12, 13]$  according to the plaintext pair and  $\Delta STK_0$ . Using Proposition 1, we get the value of  $X_0[1, 2, 6, 7, 8, 11, 12, 13]$ . Then, we can get 8 bytes of information of the tweakey as  $STK_0[1, 2, 6, 7, 8, 11, 12, 13] = P[1, 2, 6, 7, 8, 11, 12, 13] \oplus X_0[1, 2, 6, 7, 8, 11, 12, 13]$ .

(3.2) In step (3.1), we also get the value of  $Y_0[1, 2, 6, 7, 8, 11, 12, 13]$ . After a  $SR$  and  $MC$  operation, we can get the value of  $W_0[8, 13]$ . In addition, as  $\Delta X_1[8] = \Delta W_0[8] \oplus \Delta STK_1[8]$  and  $\Delta X_1[13] = \Delta W_0[13]$ ,  $\Delta X_1[8, 13]$  is also known. As shown in Table 3,  $\Delta Z_1[8, 9]$  is also known. We can also know the value of  $\Delta Y_1[8, 13]$  as  $\Delta Y_1[8, 13] = \Delta Z_1[8, 9]$ . Using Proposition 1, we can deduce the value of  $X_1[8, 13]$ . Combining it with the known value of  $W_0[8, 13]$ , we can get the value of  $STK_1[8, 13]$ .



**Figure 5** Attack Process on 10-round Deoxys-BC-256. Black boxes in internal state means these byte differences are brought from subkey differences or are equivalent to difference of some subkey bytes.

(3.3) Guess the value of  $\Delta Y_8[6, 15]$ . As  $\Delta ruX_8[15] = \Delta Y_8[15]$ ,  $\Delta ruX_8[1, 2, 7, 8, 11, 12, 13] = \Delta ruK_9[1, 2, 7, 8, 11, 12, 13]$ , and  $\Delta ruX_8[6] = \Delta Y_8[6] \oplus \Delta ruK_9[6]$ , we can know the value of all active bytes of  $ruX_8$  (the value of  $\Delta ruK$  is fixed and known). After a  $SR$  and  $MC$  operation, we can get the value of  $\Delta X_9[0, 1, 2, 3, 8, 12, 13, 14, 15]$ .

In the backward direction, we can get the difference value of  $ruX_9$  from the ciphertext pairs. As  $\Delta ruK_{10}$  is known, we can get the value  $\Delta Y_9[0, 1, 2, 3, 8, 12, 13, 14, 15]$  from  $\Delta ruX_9[0, 1, 2, 3, 7, 8, 12, 13, 14, 15] \oplus \Delta ruK_{10}[2, 7, 8, 13]$ .

Using Proposition 1, we can deduce the value of  $Y_9[0, 1, 2, 3, 8, 12, 13, 14, 15]$ . From the ciphertext value, we can deduce the value of  $ruX_9[0, 1, 2, 3, 8, 12, 13, 14, 15]$  after a  $\overline{SR}$  and  $\overline{MC}$  operation. Thus, we can get 9 bytes of information of the tweakey:  $ruK_{10}[0, 1, 2, 3, 8, 12, 13, 14, 15] = Y_9[0, 1, 2, 3, 8, 12, 13, 14, 15] \oplus ruX_9[0, 1, 2, 3, 8, 12, 13, 14, 15]$ .

(3.4) In step (3.3), we also get the value of  $X_9[0, 1, 2, 3, 8, 12, 13, 14, 15]$  using Proposition 1. As  $ruW_8 = X_9$ , the value of the corresponding bytes of  $ruW_8$  is also known. After a  $\overline{MC}$  operation, we can deduce the value  $ruZ_8[3, 14]$ . Thus, the value of  $ruX_8[15, 6]$  is known as  $ruX_8[15, 6] = ruZ_8[3, 14]$ .

As  $\Delta X_8[15, 6] = \Delta STK_8[15, 6]$  is known, by combining it with the guessed value of  $\Delta Y_8[15, 6]$ , we can deduce the value of  $Y_8[15, 6]$  using Proposition 1.

Thus, we get another 2 bytes of information of the tweakey:  $ruK_9[15, 6] = Y_8[15, 6] \oplus ruX_8[15, 6]$ .

(4) We exhaustively search the left key bits and recover the entire tweakey.

#### Complexity Computation:

In total, we can deduce  $8 + 2 + 9 + 2 = 21$  bytes, or 168 bits of information of the tweakey. As we guess  $2^{32}$  values of  $(\Delta W_0[8, 13], \Delta Y_8[15, 6])$ , each pair can eliminate  $2^{32}$  values of the 168-bit guessed tweakey information. To satisfy  $2^{168} \times (1 - 2^{32}/2^{168})^{2^{72+n}} \ll 1$ , we choose  $n = 71$ .

The data complexity is  $2^{64+71} = 2^{135}$  plaintexts. The time complexity of step (1) for encrypting the plaintexts is  $2 \cdot 2^{64+71} = 2^{136}$ . In step (3), the total number of guesses is  $2^{72+n+32} = 2^{175}$ , which is equivalent to  $2^{175} \cdot (8/16 + 2/16 + 2/16 + 9/16) \cdot 1/10 \cdot 2 \approx 2^{173.1}$  10-round encryptions. Thus, the time

complexity is approximately  $2^{173.1}$  10-round encryptions.

#### **Impact on Deoxys Authenticated Encryption:**

We stated that the analysis result has no impact on Deoxys when it uses  $r$ -round ( $r \geq 10$ ) Deoxys-BC-256 as its primitive with the recommended parameters in [2]: the key size of both Deoxys-I and Deoxys-II based on Deoxys-BC-256 is 128 bits, as the time complexity of our attack is  $2^{173.1}$ , which is larger than  $2^{128}$ .

However, Deoxys with nine-round Deoxys-BC-256 can be attacked. An attack against nine-round Deoxys-BC-256 can be mounted by removing the last round of the 10-round attack. As the used techniques are similar, we skip the details and only present the main attack results for Deoxys-BC-256. For this nine-round attack, the relevant tweak nibbles is  $8 + 2 + 2 = 12$ . To recover the 96-bit tweak information, we need both a  $2^{119}$  data complexity and time complexity. In fact, after filtering the pairs, the time complexity to recover the key by guessing  $(\Delta W_0[8, 13], \Delta Y_8[15, 6])$  is about  $2^{101}$ , so the time complexity is dominated by encrypting the  $2^{119}$  plaintexts. As the data complexity is less than  $2^{124}$  and the time complexity is less than  $2^{128}$ , this implies Deoxys with nine-round Deoxys-BC-256 can be attacked.

## 4 Conclusion

In this paper, we introduced a method that can seek out longer related-key impossible differentials from single-key impossible differentials. Using this method, we find a six-round related-key impossible distinguisher of Deoxys-BC-256 by applying the MILP method. Based on this distinguisher, we mount a 10-round attack that can attack the cipher with a wider range of key sizes compared with previous results.

**Acknowledgements** This work was supported by the National Key Research and Development Program of China (Grant No. 2017YFA0303903), the National Natural Science Foundation of China (No. 61672019), National Cryptography Development Fund (No. MMJJ20170121), Zhejiang Province Key R&D Project (No. 2017C01062), the Fundamental Research Funds of Shandong University (No. 2016JC029), and the China Postdoctoral Science Foundation (Grant No. 2017M620807).

## References

- 1 <http://competitions.cr.yp.to/caesar-submissions.html>
- 2 Jean J, Nikolić I, Peyrin T, et al. Deoxys v1.41. Submitted to CAESAR, October 2016
- 3 Daemen J, Rijmen V. The Design of Rijndael. AES - the advanced encryption standard. Berlin: Springer, 2002
- 4 Liskov M, Rivest R L, Wagner D. Tweakable Block Ciphers. *Journal of Cryptology*, 2011, 24(3): 588-613
- 5 Beierle C, Jean J, Kölbl S, et al. The SKINNY family of block ciphers and its low-latency variant MANTIS. In: Robshaw M, Katz J, eds. *Advances in Cryptology - CRYPTO 2016. Lecture Notes in Computer Science*, Vol 9815. Berlin: Springer 2016. 123-153
- 6 Borghoff J, Canteaut A, Gneysu T, et al. PRINCE - a low-latency block cipher for pervasive computing applications. In: Wang X, Sako K, eds. *Advances in Cryptology - ASIACRYPT 2012. Lecture Notes in Computer Science*, Vol 7658. Berlin: Springer, 2012. 208-225
- 7 Avanzi R. The QARMA block cipher family - almost MDS matrices over rings with zero divisors, nearly symmetric Even-Mansour constructions with non-involutory central rounds, and search heuristics for low-latency S-Boxes. *IACR Transactions on Symmetric Cryptology*, 2017, 1: 4-44
- 8 Jean J, Nikolić I, Peyrin T. Tweaks and keys for block ciphers: The TWEAKE framework. In: Sarkar P, Iwata T, eds. *Advances in Cryptology - ASIACRYPT 2014. Lecture Notes in Computer Science*, Vol 8874. Berlin: Springer, 2014. 274-288
- 9 Cid C, Huang T, Peyrin T, et al. A security analysis of Deoxys and its internal tweakable block ciphers. *IACR Transactions on Symmetric Cryptology*, 2017, 3, 73-107
- 10 Bellare M, Hoang V, Tessaro S. Message-recovery attacks on Feistel-based format preserving encryption. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. Vienna, Austria, 2016. 444-455
- 11 Ankele R, Banik S, Chakraborti A, et al. Related-key impossible-differential attack on reduced-round SKINNY. In: Gollmann D, Miyaji A, Kikuchi H, eds. *Applied Cryptography and Network Security - ACNS 2017. Lecture Notes in Computer Science*, Vol 10355. Berlin: Springer, 2017. 208-228

- 12 Derbez P, Fouque P-A, Jean J. Improved key recovery attacks on reduced-round AES in the single-key setting. In: Johansson T, Nguyen P-Q, eds. *Advances in Cryptology - EUROCRYPT 2013*. Lecture Notes in Computer Science, Vol 7881. Berlin, Springer, 2013. 371-387
- 13 Biham E, Keller N. Cryptanalysis of reduced variants of Rijndael. In: *Third AES Candidate Conference*, 2000.
- 14 Mouha N, Wang Q J, Gu D W, et al. Differential and linear cryptanalysis using mixed-integer linear programming. In: Wu C K, Lin D, eds. *Information Security and Cryptology - Inscrypt 2011*. Lecture Notes in Computer Science, Vol 7537. Berlin: Springer, 2011. 57-76
- 15 Wu S B, Wang M S. Security evaluation against differential cryptanalysis for block cipher structures. *Cryptology ePrint Archive*, Report 2011/551. <https://eprint.iacr.org/2011/551>
- 16 Sun S W, Hu L, Wang P, et al. Automatic security evaluation and (related-key) differential characteristic search: Application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers. In: Sarkar P, Iwata T, eds. *Advances in Cryptology - ASIACRYPT 2014*. Lecture Notes in Computer Science, Vol 8873. Berlin: Springer, 2014. 158-178
- 17 Sun S W, Hu L, Song L, et al. Automatic security evaluation of block ciphers with S-bP structures against related-key differential attacks. In: Lin D, Xu S, Yung M, eds. *Information Security and Cryptology - Inscrypt 2013*. Lecture Notes in Computer Science, Vol 8567. Berlin: Springer, 2013. 39-51
- 18 Sun S W, Hu L, Wang M Q, et al. Towards finding the best characteristics of some bit-oriented block ciphers and automatic enumeration of (related-key) differential and linear characteristics with predefined properties. *Cryptology ePrint Archive*, Report 2014/747. <https://eprint.iacr.org/2014/747>
- 19 Fu K, Wang M Q, Guo Y H, et al. MILP-based automatic search algorithms for differential and linear trails for Speck. In: Peyrin T, eds. *Fast Software Encryption - FSE 2016*. Lecture Notes in Computer Science, Vol 9783. Berlin: Springer, 2016. 268-288
- 20 Sasaki Y, Todo Y. New impossible differential search tool from design and cryptanalysis aspects. In: Coron J S, Nielsen J, eds. *Advances in Cryptology - EUROCRYPT 2017*. Lecture Notes in Computer Science, Vol 10212. Berlin: Springer, 2017. 185-215
- 21 Cui T T, Jia K T, Fu K, et al. New automatic search tool for impossible differentials and zero-correlation linear approximations. *Cryptology ePrint Archive*, Report 2016/689. <http://eprint.iacr.org/2016/689>