IET Information Security

Research Article

IET Journals
The Institution of Engineering and Technology

Meet-in-the-middle attacks on round-reduced tweakable block cipher Deoxys-BC

ISSN 1751-8709
Received on 18th March 2018
Revised 20th May 2018
Accepted on 21st June 2018
E-First on 25th September 2018
doi: 10.1049/iet-ifs.2018.5091
www.ietdl.org

Rongjia Li[1] ✉, *Chenhui Jin*[1]

[1]*Zhengzhou Information Science and Technology Institute, Department of Applied Mathematics, Zhengzhou 450000, People's Republic of China*
✉ *E-mail: lirongjia1991@163.com*

Abstract: Deoxys-BC is a tweakable block cipher designed by Jean *et al.* at ASIACRYPT 2014 within the TWEAKEY framework. Then Deoxys-BC is used in the CAESAR finalist Deoxys. In this study, the authors consider the security of Deoxys-BC against meet-in-the-middle attack in the single-key setting. Using the idea that a chosen tweak difference allows to cancel a difference in the state, they can construct 5-round meet-in-the-middle distinguisher on Deoxys-BC-128-128 which can be extended to attack on 8-round Deoxys-BC-128-128. Moreover, they construct 6-round meet-in-the-middle distinguisher on Deoxys-BC-256-128 which can be extended to attack on 10-round Deoxys-BC-256-128. As far as the authors know, these are the best attacks against Deoxys-BC in the single-key setting.

Nomenclature

$\|\|$	concatenation of two strings of bits
$\oplus$	bit-wise logical operation for XOR
Δx	difference of state x
P	plaintext
C	ciphertext
$x[i]$	byte in position i of state x
$(x \oplus y)[i]$	byte in position i of state $x \oplus y$
$x[i, \ldots, j]$	bytes in position $i, \ldots, j$ of state x

1 Introduction

Liskov *et al.* [1, 2] formalised the definition of tweakable block cipher and showed that tweakable block cipher is a valuable building block if changing the tweak value is less costly than changing its secret key. Since then, tweakable block cipher has found many different applications in cryptographic schemes, such as disk encryption (e.g. [3]), format-preserving encryption (e.g. [4]) or authenticated encryption (e.g. [5]). Formally, a tweakable block cipher is a map $E: K \times T \times \{0,1\}^n \to \{0,1\}^n$ where for every key $k \in K$ and every tweak $t \in T$, $E(k, t, \cdot)$ is a permutation on $\{0,1\}^n$. The key is usually secret, but the tweak is completely public or even chosen by the attacker.

When one designs a tweakable block cipher, Liskov *et al.* proposed to separate the roles of the secret key from that of the tweak. At ASIACRYPT 2014, Jean *et al.* [6] argued that these two inputs should be considered almost the same. They brought together key schedule design and tweak input handling for block ciphers in a commmon framework called TWEAKEY which unifies the tweak and key input. The TWEAKEY framework is to build a n-bit tweakable block cipher with $(k + t)$-bit tweakey (the concatenation of k-bit key and t-bit tweak). The term *tweakey* refers to an input that can be both tweak or key material, without distinction.

They also gave a subclass of TWEAKEY for AES-like ciphers, named STK, where the key and the tweak materials are treated almost the same, except for the small difference between the linear key and the tweak schedules. Two concrete tweakable block ciphers based on the STK construction were proposed: Deoxys-BC and Joltik-BC [7]. Deoxys-BC comes with two versions. One version is denoted by Deoxys-BC-256 which uses a 256-bit tweakey, and another is denoted by Deoxys-BC-384 which uses a 384-bit tweakey. Both versions process plaintext blocks of 128 bits.

Especially, the CAESAR candidate Deoxys [8] is designed based on the Deoxys-BC-256 and Deoxys-BC-384 block ciphers. With the recommended parameters given in [8], two schemes of Deoxys are based on the internal block cipher Deoxys-BC-256 with 128-bit key and 128-bit tweak (denoted by Deoxys-BC-128-128), while the other two schemes of Deoxys are based on the internal block cipher Deoxys-BC-384 with 256-bit key and 128-bit tweak (denoted by Deoxys-BC-256-128).

The existing public security analysis of Deoxys-BC focuses mainly on related-key related-tweak differential attacks and meet-in-the-middle attacks. In [8], the designers of Deoxys-BC provided an upper bound on the probability of the best round-reduced related-key related-tweak differential paths for Deoxys-BC, which was greatly improved in [9] by performing a special MILP-based search. Moreover, Cid *et al.* proposed related-key related-tweak rectangle attacks against 10-round Deoxys-BC-256 and 14-round Deoxys-BC-384 in [9]. As for meet-in-the-middle attacks, the designers considered attacks in the single-key single-tweak setting and claimed that 'a first analysis shows that the meet-in-the-middle technique can attack up to 8 rounds'. Recently, Mehrdad *et al.* presented the first impossible differential cryptanalysis of Deoxys-BC-256 in [10].

At FSE 2008, a new type of meet-in-the-middle attack was proposed by Demirci and Selçuk [11] in the attacks on AES. Then, Dunkelman *et al.* [12, 13] improved the attack using *multi-set technique*, *key-bridging technique* and *differential enumeration technique*. Finally, Derbez *et al.* [14, 15] found that the attack could be more efficient and described the best attack on AES-128 in single-key setting, then Li *et al.* [16] gave a meet-in-the-middle attack on 9-round AES-192 and Li and Jin [17] gave a meet-in-the-middle attack on 10-round AES-256.

In contrast to standard block cipher, tweakable block cipher provides a public tweak, which can be fully controlled by the user. Consequently, differences can be injected into the internal state via tweak to cancel the state differences even in the single-key setting. This idea was first introduced by Dobraunig *et al.* [18] in the square attack on Kiasu-BC, and then used in the impossible differential and boomerang cryptanalysis [19], meet-in-the-middle attack [20] and differential cryptanalysis [21]. In the sequel, we call this idea *tweak difference cancellation technique*.

Our contributions: In this paper, we provide (to the best of our knowledge) the best attacks against Deoxys-BC in the single-key setting and all our attacks are based on meet-in-the-middle attacks. Similar to prior works [18–21], we fully use the freedom introduced by the tweak, and choose a non-zero tweak difference

Table 1 Cryptanalytic results on Deoxys-BC in the single-key setting

Cipher	Rounds	Time	Data (CP)	Memory	Technique	Reference
Deoxys-BC-128-128	8	$\leq 2^{128}$	—	—	MITM	[8]
	8	$2^{116.5}$	$2^{116.5}$	2^{48}	ID	[10]
	8	2^{113}	2^{113}	2^{97}	MITM	this paper
Deoxys-BC-256-128	8	$\leq 2^{256}$	—	—	MITM	[8]
	10	2^{228}	2^{113}	2^{226}	MITM	this paper

CP: chosen plaintext, ID: impossible differential, MITM: meet-in-the-middle.

Table 2 Two LFSRs used in Deoxys-BC tweakey schedule

$LFSR_2$	input byte	$(b_7 \| b_6 \| b_5 \| b_4 \| b_3 \| b_2 \| b_1 \| b_0)$
	output byte	$(b_6 \| b_5 \| b_4 \| b_3 \| b_2 \| b_1 \| b_0 \| b_7 \oplus b_5)$
$LFSR_3$	input byte	$(b_7 \| b_6 \| b_5 \| b_4 \| b_3 \| b_2 \| b_1 \| b_0)$
	output byte	$(b_0 \oplus b_6 \| b_7 \| b_6 \| b_5 \| b_4 \| b_3 \| b_2 \| b_1)$

which allows to cancel a difference in the state. Combined with the *tweak difference cancellation technique* and *differential enumeration technique*, we can, respectively, construct 8-round and 10-round attacks for Deoxys-BC-128-128 and Deoxys-BC-256-128, which are based on 5-round and 6-round distinguishers, respectively. The time complexity of the attack on 8-round Deoxys-BC-128-128 is 2^{113}, the memory complexity is 2^{97} and the data complexity is 2^{113} chosen plaintext-tweak combinations. The time complexity of the attack on 10-round Deoxys-BC-256-128 is 2^{228}, the memory complexity is 2^{226} and the data complexity is 2^{113} chosen plaintext-tweak combinations. Table 1 summarises our results along with previous attack results of Deoxys-BC in the single-key setting.

Organisations of this paper: In Section 2, we provide the specification of Deoxys-BC. In Sections 3 and 4, we give the attacks on 8-round Deoxys-BC-128-128 and 10-round Deoxys-BC-256-128, respectively. We conclude this paper in Section 5.

2 Preliminary

2.1 Descriptions of Deoxys-BC

Deoxys-BC [8] adopts an AES-like design, i.e. an iterative substitution permutation network with 128-bit block size. The internal state of Deoxys-BC can be seen as a 4×4 matrix of bytes, where each byte represents a value in $GF(2^8)$ defined by the irreducible polynomial $x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1$. Hereinafter, we number the 16 bytes of an internal state by the usual ordering

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

Like AES, one round Deoxys-BC consists of four operations:

- AddRoundTweakey (ART): XOR the 128-bit round subtweakey to the internal state,
- SubBytes (SB): Apply the 8-bit Sbox S to each byte of the internal state,
- ShiftRows (SR): Rotate the 4-byte ith row of the state matrix left by i bytes ($i = 0, 1, 2, 3$),
- MixBytes (MB): Multiply the internal state by the 4×4 constant MDS matrix from AES.

After the final round, an additional AddRoundTweakey operation is performed. The number of rounds is 14 for Deoxys-BC-256 and 16 for Deoxys-BC-384.

So far, the description of Deoxys-BC is the same as AES. The operation of ART, and in particular the production of the subtweakeys, is where Deoxys-BC differs from AES. We denote the concatenation of the key K and the tweak T as KT, i.e. $KT = K||T$. The tweakey is then divided into 128-bit words. More precisely, in Deoxys-BC-256 the size of KT is 256 bits with the first (most significant) 128 bits of KT being denoted by W_2; the second word is denoted by W_1. For Deoxys-BC-384, the size of KT is 384 bits, and we denote the first (most significant), second and third 128-bit words of KT by W_3, W_2 and W_1, respectively. Finally, we denote with STK_i the subtweakey (a 128-bit word) that is added to the state at round i of the cipher during the AddRoundTweakey operation. For Deoxys-BC-256, a subtweakey is defined as $STK_i = TK_i^1 \oplus TK_i^2 \oplus RC_i$, whereas for Deoxys-BC-384 it is defined as $STK_i = TK_i^1 \oplus TK_i^2 \oplus TK_i^3 \oplus RC_i$. Here RC_i are the key schedule round constants and we will omit them, as they have little impact on our analysis.

The 128-bit words TK_i^1, TK_i^2, TK_i^3 are outputs produced by a special tweakey schedule algorithm, initialised with $TK_0^1 = W_1$ and $TK_0^2 = W_2$ for Deoxys-BC-256 and with $TK_0^1 = W_1$, $TK_0^2 = W_2$ and $TK_0^3 = W_3$ for Deoxys-BC-384. The tweakey schedule algorithm is defined as

$$\begin{aligned} TK_{i+1}^1 &= h(TK_i^1), \\ TK_{i+1}^2 &= h(LFSR_2(TK_i^2)), \\ TK_{i+1}^3 &= h(LFSR_3(TK_i^3)), \end{aligned}$$

where the byte permutation h is defined as (see equation below) with the 16 bytes of a 128-bit tweakey word numbered by the usual AES byte ordering.

The $LFSR_2$ and $LFSR_3$ functions are simply the application of an LFSR to each of the 16 bytes of a tweakey 128-bit word. More precisely, the two LFSRs are given in Table 2 (b_0 stands for the LSB of the byte).

Tweakey schedule of Deoxys-BC-128-128: For Deoxys-BC-128-128, since it uses a 128-bit key K and a 128-bit tweak T, W_2 (the first 128 bits of KT) equals to K and W_1 (the second 128 bits of KT) equals to T. For simplicity, we denote TK_i^2 with k_i and denote TK_i^1 with t_i. Thus, the tweakey schedule algorithm takes as inputs the 128-bit key K and the 128-bit tweak T, and sequentially produces the 128-bit words k_i and t_i

$$\begin{aligned} t_{i+1} &= h(t_i), \\ k_{i+1} &= h(LFSR_2(k_i)), \end{aligned}$$

where $k_0 = K$ and $t_0 = T$.

Then a subtweakey of Deoxys-BC-128-128 is defined as

$$STK_i = k_i \oplus t_i \oplus RC_i.$$

Tweakey schedule of Deoxys-BC-256-128: For Deoxys-BC-256-128, since it uses a 256-bit key K and a 128-bit tweak T, where K is divided into words of 128 bits, i.e. $K = K^1 \| K^2$, W_3 (the

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

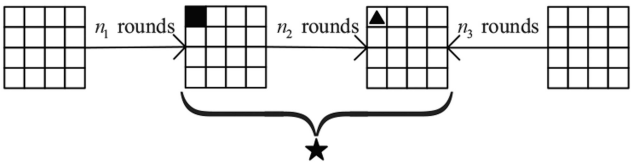

Fig. 1 *General scheme of meet-in-the-middle attack, where 'star' represents a certain property that some messages may verify, 'filled square' represents the active byte of the δ-set and 'filled triangle' represents the chosen byte on which we want to build sequences*

first 128 bits of KT) equals to K^1, W_2 (the second 128 bits of KT) equals to K^2, and W_1 (the third 128 bits of KT) equals to T. For simplicity, we denote TK_i^3 with k_i^1, TK_i^2 with k_i^2 and TK_i^1 with t_i. Thus, the tweakey schedule algorithm takes as inputs the 256-bit key K and the 128-bit tweak T, and the 128-bit words k_i^1, k_i^2 and t_i are sequentially produced

$$t_{i+1} = h(t_i),$$
$$k_{i+1}^2 = h(LFSR_2(k_i^2)),$$
$$k_{i+1}^1 = h(LFSR_3(k_i^1)),$$

where $k_0^1 = K^1$, $k_0^2 = K^2$ and $t_0 = T$.

Then a subtweakey of Deoxys-BC-256-128 is defined as

$$STK_i = k_i^1 \oplus k_i^2 \oplus t_i \oplus RC_i.$$

Throughout the paper, the numbering of the rounds starts with round 0, and the intermediate state after the ART, SB, SR and MB operations of round i in the encryption of the plaintext P^j is denoted by x_i^j, y_i^j, z_i^j and w_i^j, respectively. Sometimes, the key addition with k_i in round i and the MB operation in round $i-1$ should be swapped without affecting the result. They can be swapped if the key addition with the equivalent round key $u_i = MB^{-1}(k_i)$ is performed instead. In this case, we denote the intermediate state before MB operation by s_{i-1}.

2.2 Meet-in-the-middle attack

In this section, we first give the definition and property used in the attacks. Then, we present the general scheme of meet-in-the-middle attack.

Definition 1: A δ-set is a set of 2^8 state values that are all different in one state byte (the active byte) and are all equal in the remaining state bytes (the inactive bytes) [22].

Property 1: Given Δ_i and Δ_j two non-zero differences of S-box S, the equation $S(x) \oplus S(x \oplus \Delta_i) = \Delta_o$, has one solution in average [14].

As shown in Fig. 1, in the meet-in-the-middle attack on an n-round block cipher, the block cipher is split into three consecutive parts of n_1, n_2 and n_3 rounds, $n = n_1 + n_2 + n_3$, such that a particular message satisfies a certain property in the middle n_2 rounds.

The general attack uses the following two phases [14]:

Precomputation phase

i. In this phase, we first choose a byte in the end of the middle n_2 rounds and then build a hash table H containing all the possible sequences of 256 values of that byte constructed from a δ-set such that one message satisfies the $\star$ property.

Online phase

ii. In the online phase, we first identify a δ-set containing a message satisfying the $\star$ property.

iii. Then, we partially decrypt the associated δ-set through the last n_3 rounds and check whether it belongs to H.

In the online phase of the attack, we need to guess some key bytes. The goal of this attack is to filter some of their values. In the best case, only the right key can pass the test.

For meet-in-the-middle attack with *differential enumerate technique*, the $\star$ property is set to be an expected-probability truncated differential characteristic. Usually, the active byte of the truncated differential characteristic at the beginning of the middle n_2 rounds is exactly the active byte of the δ-set, and the active byte of the truncated differential characteristic at the end of the middle n_2 rounds is exactly the chosen byte on which we want to build sequences.

3 Meet-in-the-middle attack on 8-round Deoxys-BC-128-128

In this section, we launch an 8-round attack on Deoxys-BC-128-128 combined with the *differential enumerate technique* and *tweak difference cancellation technique*. By cancelling the state difference with the tweak difference, we can make one blank round and then construct 5-round meet-in-the-middle distinguisher. Since the active byte of the δ-set can take 16 different positions and so does the byte on which to build the sequence, we can construct $16 \times 16 = 256$ different distinguishers. By adding one round on top of a distinguisher and two rounds beneath it, 8-round attack can be constructed. The introduced tweak difference may activate a few additional state bytes in the later rounds and the number of active state bytes varies according to different attacks based on different distinguishers. As a result, the complexities of the 256 attacks are different. We exhaust all the 256 attacks and find eight attacks with the lowest complexity. In the following, we pick one of the eight attacks and describe the detailed procedure of it.

3.1 5-Round distinguisher on Deoxys-BC-128-128

Theorem 1: Let $\{w_0^0, w_0^1, \ldots, w_0^{255}\}$ and $\{t_1^0, t_1^1, \ldots, t_1^{255}\}$ be two δ-sets where $w_0^d[1] = t_1^d[1] \oplus t_1^0[1] = d$ for $0 \le d \le 255$. Consider the encryption of w_0^a $(0 \le a \le 31)$ under tweak t_1^a through 5-round Deoxys-128-128, in the case of that a pair of message-tweak combination $\{(w_0^i, t_1^i), (w_0^j, t_1^j)\}$ $(0 \le i, j \le 255)$ conforms to the related-tweak truncated differential characteristic outlined in Fig. 2, then the corresponding 248-bit ordered sequence $(x_6^1[4] \oplus x_6^0[4], x_6^2[4] \oplus x_6^0[4], \ldots, x_6^{31}[4] \oplus x_6^0[4])$ only takes about 2^{96} values (out of the 2^{248} theoretically values).

Proof: Firstly, we prove that the sequence $(x_6^1[4] \oplus x_6^0[4], x_6^2[4] \oplus x_6^0[4], \ldots, x_6^{31}[4] \oplus x_6^0[4])$ is determined by 26 bytes, namely

$$w_0^i[1], x_2^i[6], x_3^i[12, 13, 14, 15], x_4^i, x_5^i[3, 4, 9, 14].$$

In the following proof, we denote $\Delta t^a, \Delta x^a, \Delta y^a, \Delta w^a$ the difference between t^a and t^i, x^a and x^i, y^a and y^i, w^a and w^i, respectively. With the knowledge of $t_1^a[1] \oplus t_1^0[1]$ and $t_1^i[1] \oplus t_1^0[1]$ which equals to $w_0^i[1]$, we can deduce Δt_s^a $(1 \le s \le 6)$ because of the linear tweakey schedule. Since $\Delta w_0^a[1] = \Delta t_1^a[1] = a \oplus w_0^i[1]$, then $\Delta x_1^a = 0$. Hence, $\Delta x_2^a[6] = \Delta t_2^a[6]$. Then we can get $\Delta x_3^a[12, 13, 14, 15]$ linearly from $\Delta t_3^a[15]$ and $\Delta y_2^a[6]$, which can be deduced from $\Delta x_2^a[6]$ and $x_2^i[6]$. Similarly, Δx_4^a is deduced with the knowledge of $x_3^i[12, 13, 14, 15]$, $\Delta x_5^a[3, 4, 9, 14]$ is deduced with the knowledge of x_4^i and $\Delta x_6^a[4]$ is deduced with the knowledge of $x_5^i[3, 4, 9, 14]$. Then by XORing $\Delta x_6^a[4]$ with $\Delta x_6^0[4]$, the sequence $(x_6^1[4] \oplus x_6^0[4], x_6^2[4] \oplus x_6^0[4], \ldots, x_6^{31}[4] \oplus x_6^0[4])$ is obtained.

If a pair of message-tweak combination $\{(w_0^i, t_1^i), (w_0^j, t_1^j)\}$ conforms to the related-tweak truncated differential characteristic outlined in Fig. 2, we can determine the above 26 bytes only with the following 12 bytes:

$$w_0^i[1], \Delta x_2^j[6], x_2^i[6], x_3^i[12, 13, 14, 15], x_5^i[3, 4, 9, 14], \Delta w_5^j[4].$$

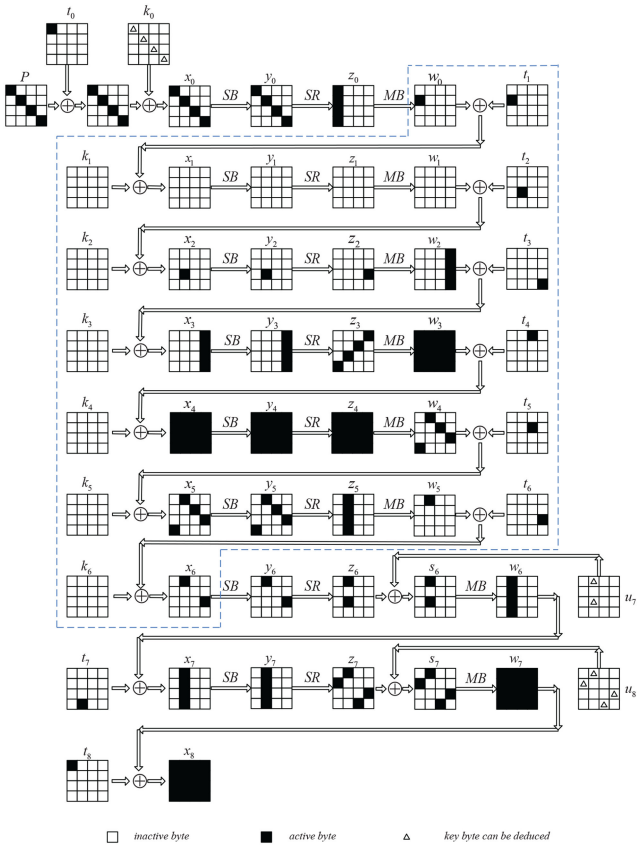

Fig. 2 *Related-tweak truncated differential characteristic of the Deoxys-128-128*

With the knowledge of $\Delta x_2^j[6]$ which equals to $\Delta t_2^j[6]$, we can deduce Δt_s^j. Since $\Delta x_2^j[6]$ and $x_2^j[6], x_3^j[12, 13, 14, 15]$ are known, Δx_4^j is deduced. On the other hand, Δy_4^j can be deduced with the knowledge of $\Delta w_5^j[4]$ and $x_5^j[3, 4, 9, 14]$. According to Property 1, we get on average one value of x_4^j for a fixed difference $\Delta x_4^j, \Delta y_4^j$.

As a result, the sequence $(x_6^1[4] \oplus x_6^0[4], x_6^2[4] \oplus x_6^0[4], \ldots, x_6^{31}[4] \oplus x_6^0[4])$ is determined by 12 bytes, so only takes about 2^{96} values. □

3.2 Attack on 8-round Deoxys-BC-128-128

The attack is composed of two phases: precomputation phase and online phase.

In the precomputation phase, we compute all the 2^{96} possible values of the sequence $(x_6^1[4] \oplus x_6^0[4], x_6^2[4] \oplus x_6^0[4], \ldots, x_6^{31}[4] \oplus x_6^0[4])$ as described in Theorem 1, and then store them in a hash table H.

The attack procedure of the online phase is as follows.

Structures: We will consider sets and structures of plaintexts. Define a set S of 2^{32} plaintexts where $P[0, 5, 10, 15]$ take all the possible 2^{32} values, and the remaining 12 bytes are fixed to some constants. A sub-structure consists of a set S and a concrete tweak T. Then, a structure L consists of 2^8 sub-structures, where each sub-structure in L differs only in the tweak byte $T[0]$. Thus, there are in total $2^{32} \times 2^8 = 2^{40}$ plaintext-tweak combinations in a structure. We can generate $2^{40} \times (2^{40} - 1)/2 = 2^{79}$ pairs of plaintext-tweak combination for one structure.

Step 1: Choose 2^{73} structures, which can yield about $2^{73} \times 2^{79} = 2^{152}$ pairs. Then, for each structure, do the following steps:

i. Ask for the encryption of the structure and obtain the corresponding ciphertexts $C_i \leftarrow E_K^{T_i}(P_i)$.
ii. Invert the final tweak XOR and MixBytes operation and store all states s_7 into a hash table indexed by bytes $s_7[0, 2, 3, 5, 6, 7, 8, 9, 10, 12, 13, 15]$.
iii. For each row with more than one ciphertext, consider all possible combinations of pairs therein. We expect to have $2^{79} \times 2^{-8 \times 12} = 2^{-17}$ pairs. So for all the 2^{73} structures we can obtain $2^{-17} \times 2^{73} = 2^{56}$ such pairs that are equal in bytes $s_7[0, 2, 3, 5, 6, 7, 8, 9, 10, 12, 13, 15]$.

Step 2: For the 2^{56} remaining pairs, look for pairs satisfying the following two conditions:

i. The difference in $T[0]$ is non-zero,
ii. The difference in $P[a] \oplus T[a]$ is non-zero for $a = 0, 5, 10, 15$.

About $2^{56} \times (255/256)^5 \simeq 2^{56}$ pairs are expected to remain. Among the 2^{56} pairs, we expect that about $2^{56} \times 2^{-8 \times 7} = 1$ pair satisfies the truncated differential trail in Fig. 2 because of the $4 \rightarrow 1$ transition in MixBytes of round 0, $1 \rightarrow 0$ transition in AddRoundTweakey of round 1, $2 \rightarrow 1$ transition in AddRoundTweakey of round 6 and $4 \rightarrow 2$ transition in MixBytes of round 6. For each of the 2^{56} pairs, we suppose it follows the truncated differential trail and do the following steps:

i. Compute $\Delta t_1[1], \Delta t_7[7]$ and $\Delta t_8[0]$, with the knowledge of $\Delta t_0[0]$. Since $\Delta w_0[1]$ equals to $\Delta t_1[1]$, we can then deduce $\Delta y_0[0, 5, 10, 15]$. Deduce the difference $\Delta x_0[0, 5, 10, 15]$ with the plaintexts. According to Property 1, $x_0[0, 5, 10, 15]$ is obtained. Hence, we get the subkey $k_0[0, 5, 10, 15]$.
ii. Guess $\Delta y_6[4, 14]$ and then deduce $\Delta x_7[4, 5, 6, 7]$. Deduce $\Delta y_7[4, 5, 6, 7]$ with the ciphertexts. Therefore, according to Property 1 we can compute $y_7[4, 5, 6, 7]$ which can be used to deduce the subkey $u_8[1, 4, 11, 14]$.
iii. Take one of the members of the pair and encrypt it to get the value $w_0[0, 1, 2, 3]$ and $t_1[1]$, then let $i = w_0[1]$. Next, we construct the first 32 values of the two δ-sets defined in Theorem 1. Let the value of $(w_0^0[1], w_0^1[1], \ldots, w_0^{31}[1])$ be $(0, 1, \ldots, 31)$ and the value of $(t_1^0[1], t_1^1[1], \ldots, t_1^{31}[1])$ be $(t_1[1] \oplus i, t_1[1] \oplus i \oplus 1, \ldots, t_1[1] \oplus i \oplus 31)$. Therefore, we can obtain the 32 plaintext-tweak combinations $\{(P^0, T^0), \ldots, (P^{31}, T^{31})\}$ through partial decryption.
iv. Guess $u_7[4]$. Query the corresponding ciphertexts of the 32 plaintext-tweak combinations and partially decrypt them with the knowledge of $u_7[4], u_8[1, 4, 11, 14]$ to obtain the sequence $(x_6^1[4] \oplus x_6^0[4], x_6^2[4] \oplus x_6^0[4], \ldots, x_6^{31}[4] \oplus x_6^0[4])$.
v. Check whether the sequence lies in the hash table H. If not, we discard the subkey $k_0[0, 5, 10, 15], u_7[4], u_8[1, 4, 11, 14]$. Since the probability for a wrong subkey to pass this test is $2^{96} \times 2^{-248} = 2^{-152}$, there are about $1 + 2^{56} \times 2^{8 \times 3} \times 2^{-152} \simeq 1$ subkey left in the end.

Step 3: The master key can be retrieved by exhaustive key search of the rest 12 bytes of k_0. Note that by using the linear key schedule, <12 subkey bytes need to be guessed, since a lot of subkey bytes are already known. However, this will not influence the complexity of the whole attack.

Complexity analysis: In the precomputation phase, the memory complexity is $2^{96} \times 248/128 \simeq 2^{97}$ 128-bit blocks of memory required for storing the hash table H. The time complexity is about $2^{96} \times 32 \times 2^{-2.4} = 2^{98.6}$ 8-round Deoxys-BC encryptions since each value of the δ-set needs about $2^{-2.4}$ 8-round Deoxys-BC computations. In the online phase, the time complexity is dominated by encrypting $2^{73} \times 2^{40} = 2^{113}$ plaintext-tweak combinations. All in all, the time complexity of the attack is 2^{113} 8-round Deoxys-BC encryptions, the memory complexity is 2^{97} 128-bit blocks and the data complexity is 2^{113} plaintext-tweak combinations.

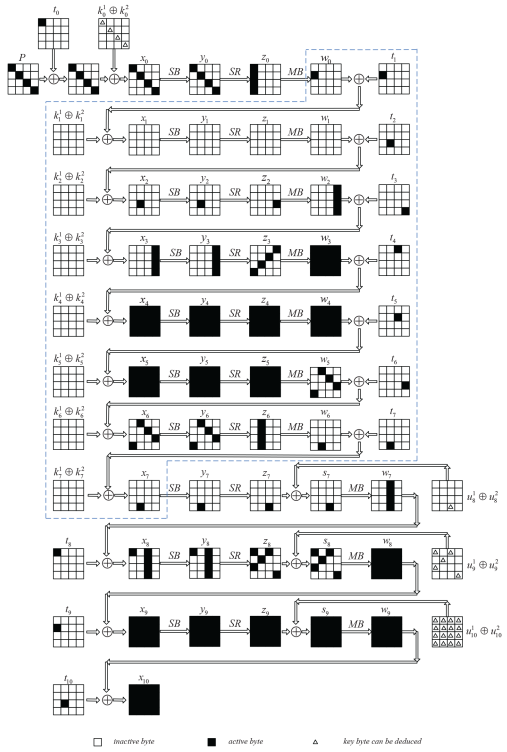

Fig. 3 *Related-tweak truncated differential characteristic of the Deoxys-256-128*

4 Meet-in-the-middle attack on 10-round Deoxys-BC-256-128

In this section, we present a 10-round attack on Deoxys-BC-256-128 based on 6-round meet-in-the-middle distinguisher. The detailed procedure of the attack is as follows.

4.1 6-Round distinguisher on Deoxys-BC-256-128

Theorem 2: Let $\{w_0^0, w_0^1, \ldots, w_0^{255}\}$ and $\{t_1^0, t_1^1, \ldots, t_1^{255}\}$ be two δ-sets where $w_0^d[1] = t_1^d[1] \oplus t_1^0[1] = d$ for $0 \le d \le 255$. Consider the encryption of w_0^a $(0 \le d \le 63)$ under tweak t_1^a through 6-round Deoxys-256-128, in the case of that a pair of message-tweak combination $\{(w_0^i, t_1^i), (w_0^j, t_1^j)\}$ $(0 \le i, j \le 255)$ conforms to the related-tweak truncated differential characteristic in Fig. 3, then the corresponding 504-bit ordered sequence $(x_7^1[7] \oplus x_7^0[7], x_7^2[7] \oplus x_7^0[7], \ldots, x_7^{63}[7] \oplus x_7^0[7])$ only takes about 2^{224} values (out of the 2^{504} theoretically values).

Proof: First, we can prove that the sequence $(x_7^1[7] \oplus x_7^0[7], x_7^2[7] \oplus x_7^0[7], \ldots, x_7^{63}[7] \oplus x_7^0[7])$ is determined by 42 bytes, namely

$$w_0^i[1], x_2^i[6], x_3^i[12, 13, 14, 15], x_4^i, x_5^i, x_6^i[3, 4, 9, 14].$$

Second, if a pair of message-tweak combination $\{(w_0^i, t_1^i), (w_0^j, t_1^j)\}$ conforms to the related-tweak truncated differential characteristic outlined in Fig. 3, we can determine the above 42 bytes only with the following 28 bytes:

$$w_0^i[1], \Delta x_2^j[6], x_2^j[6], x_3^i[12, 13, 14, 15], x_4^i, x_6^i[3, 4, 9, 14], \Delta w_6^j[7].$$

As a result, the sequence $(x_7^1[7] \oplus x_7^0[7], x_7^2[7] \oplus x_7^0[7], \ldots, x_7^{63}[7] \oplus x_7^0[7])$ is determined by 28 bytes, so only takes about 2^{224} values. □

4.2 Attack on 10-round Deoxys-BC-256-128

The attack is composed of two phases: precomputation phase and online phase.

In the precomputation phase, we compute all the 2^{224} possible values of the sequence $(x_7^1[7] \oplus x_7^0[7], x_7^2[7] \oplus x_7^0[7], \ldots, x_7^{63}[7] \oplus x_7^0[7])$ as described in Theorem 2, and then store them in a hash table H.

The attack procedure of the online phase is as follows:

Structures: Define a set S of 2^{32} plaintexts where $P[0, 5, 10, 15]$ takes all the possible 2^{32} values, and the remaining 12 bytes are fixed to some constants. A sub-structure consists of a set S and a concrete tweak T. Then, a structure L consists of 2^8 sub-structures, where each sub-structure in L differs only in the tweak byte $T[0]$. Thus, there are in total 2^{40} plaintext-tweak combinations in a structure. We can generate $2^{40} \times (2^{40} - 1)/2 = 2^{79}$ pairs of plaintext-tweak combination for one structure.

Step 1: Choose 2^{73} structures, which can yield about $2^{73} \times 2^{79} = 2^{152}$ pairs. For the 2^{152} pairs, look for pairs satisfying the following two conditions:

i. The difference in $T[0]$ is non-zero,
ii. The difference in $P[a] \oplus T[a]$ is non-zero for $a = 0, 5, 10, 15$.

About $2^{152} \times (255/256)^5 \simeq 2^{152}$ pairs are expected to remain. Among the 2^{152} pairs, we expect that about $2^{152} \times 2^{-8 \times 19} = 1$ pair satisfies the truncated differential trail in Fig. 3. For each of the 2^{152} pairs, we suppose it follows the truncated differential trail and do the following steps:

i. Ask for the encryption of the pair and obtain the corresponding ciphertexts.
ii. Compute $\Delta t_1[1], \Delta t_8[0]$ and $\Delta t_9[1]$ with the knowledge of $\Delta t_0[0]$. Since $\Delta w_0[1]$ equals to $\Delta t_1[1]$, we can then deduce $\Delta y_0[0, 5, 10, 15]$. Deduce the difference $\Delta x_0[0, 5, 10, 15]$ with the plaintexts. According to Property 1, $x_0[0, 5, 10, 15]$ is obtained. Hence, we get the subkey $(k_0^1 \oplus k_0^2)[0, 5, 10, 15]$.
iii. Guess $\Delta y_8[0, 8, 9, 10, 11]$ and then deduce Δx_9. Deduce Δy_9 with the ciphertexts. Therefore, according to Property 1 we can compute x_9 and y_9 which can be used to deduce the subkey $u_{10}^1 \oplus u_{10}^2$. Guess $\Delta y_7[7]$ and then deduce $\Delta x_8[8, 9, 10, 11]$. Since $\Delta x_8[0]$ equals to $\Delta t_8[0]$, $y_8[0, 8, 9, 10, 11]$ can be deduced according to Property 1. Thus, we can deduce $(u_9^1 \oplus u_9^2)[0, 2, 5, 8, 15]$.
iv. Take one of the members of the pair and encrypt it to get the value $w_0[0, 1, 2, 3]$ and $t_1[1]$, then let $i = w_0[1]$. Next, we construct the first 64 values of the two δ-sets defined in Theorem 2. Let the value of $(w_0^0[1], w_0^1[1], \ldots, w_0^{63}[1])$ be $(0, 1, \ldots, 63)$ and the value of $(t_1^0[1], t_1^1[1], \ldots, t_1^{63}[1])$ be $(t_1[1] \oplus i, t_1[1] \oplus i \oplus 1, \ldots, t_1[1] \oplus i \oplus 63)$. Therefore, we can obtain the 64 plaintext-tweak combinations $\{(P^0, T^0), \ldots, (P^{63}, T^{63})\}$ through partial decryption.
v. Guess $(u_8^1 \oplus u_8^2)[11]$. Query the corresponding ciphertexts of the 64 plaintext-tweak combinations and partially decrypt them with the knowledge of $(u_8^1 \oplus u_8^2)[11], (u_9^1 \oplus u_9^2)[2, 5, 8, 15], u_{10}^1 \oplus u_{10}^2$ to obtain the sequence $(x_7^1[7] \oplus x_7^0[7], x_7^2[7] \oplus x_7^0[7], \ldots, x_7^{63}[7] \oplus x_7^0[7])$.
vi. Check whether the sequence lies in the hash table H. If not, we discard the subkey $k_0[0, 5, 10, 15], (u_8^1 \oplus u_8^2)[11], (u_9^1 \oplus u_9^2)[0, 2, 5, 8, 15], u_{10}^1 \oplus u_{10}^2$. Since the probability for a wrong subkey to pass this test is $2^{224} \times 2^{-504} = 2^{-280}$, there are about $1 + 2^{152} \times 2^{8 \times 7} \times 2^{-280} \simeq 1$ subkey left in the end.

Step 2: The master key can be retrieved by exhaustive key search of the 16 bytes of u_{10}^1.

Complexity analysis: In the precomputation phase, the memory complexity is $2^{224} \times 504/128 \simeq 2^{226}$ 128-bit blocks of memory required for storing the hash table H. The time complexity is about $2^{224} \times 64 \times 2^{-2} = 2^{228}$ 10-round Deoxys-BC encryptions. In the online phase, the time complexity is dominated by step 1.5, whose time complexity equals to $2^{152} \times 2^{8\times7} \times 64 \times 2^{-3} = 2^{211}$ 10-round Deoxys-BC encryptions. All in all, the time complexity of the attack is 2^{228} 10-round Deoxys-BC encryptions, the memory complexity is 2^{226} 128-bit blocks and the data complexity is $2^{73} \times 2^{40} = 2^{113}$ plaintext-tweak combinations.

5 Conclusion

This paper considers the security of Deoxys-BC against meet-in-the-middle attacks in the single-key setting. Using the freedom introduced by the tweak, we can choose a non-zero tweak difference which allows to cancel a difference in the state. With this method, we construct 5-round meet-in-the-middle distinguisher on Deoxys-BC-128-128. Then we extend one round on the top and two rounds on the bottom to present attack on 8-round Deoxys-BC-128-128. We construct 6-round meet-in-the-middle distinguisher on Deoxys-BC-256-128 and then extend one round on the top and three rounds on the bottom to present attack on 10-round Deoxys-BC-256-128.

6 Acknowledgments

The authors like to thank anonymous referees for their helpful comments and suggestions. This work was supported by the National Natural Science Foundation of China (grant nos. 61772547, 61402523, 61272488).

7 References

[1] Liskov, M., Rivest, R.L., Wagner, D.A.: 'Tweakable block ciphers'. CRYPTO 2002, Santa Barbara, USA, August 2002, pp. 31–46

[2] Liskov, M., Rivest, R.L., Wagner, D.A.: 'Tweakable block ciphers', *J. Cryptol.*, 2011, **24**, (3), pp. 588–613

[3] Crowley, P.: 'Mercy: A fast large block cipher for disk sector encryption'. FSE 2000, New York, USA, April 2000, pp. 49–63

[4] Lee, J., Koo, B., Roh, D., *et al.*: 'Format-preserving encryption algorithms using families of tweakable blockciphers'. ICISC 2014, Seoul, Korea, December 2014, pp. 132–159

[5] Peyrin, T., Seurin, Y.: 'Counter-in-tweak: authenticated encryption modes for tweakable block ciphers'. CRYPTO 2016, Part I, Santa Barbara, USA, August 2016, pp. 33–63

[6] Jean, J., Nikolic, I., Peyrin, T.: 'Tweaks and keys for block ciphers: The TWEAKEY framework'. ASIACRYPT 2014, Part II, Taiwan, R.O.C., December 2014, pp. 274–288

[7] Jean, J., Nikolic, I., Peyrin, T.: 'Tweaks and keys for block ciphers: the TWEAKEY framework'. IACR Cryptology ePrint Archive, Report 2014/831. Available at http://eprint.iacr.org/2014/831

[8] Jean, J., Nikolic, I., Peyrin, T., *et al.*: 'Deoxys v1.4', Submission to the CAESAR competition. Available at http://www1.spms.ntu.edu.sg/ syllab/ Deoxys

[9] Cid, C., Huang, T., Peyrin, T., *et al.*: 'Cryptanalysis of Deoxys and its internal tweakable block ciphers', *IACR Trans. Symmetric Cryptol.*, 2017, **3**, pp. 73–107

[10] Mehrdad, A., Moazami, F., Soleimany, H.: 'Impossible differential cryptanalysis on Deoxys-BC-256'. IACR Cryptology ePrint Archive, Report 2018/048. Available at https://eprint.iacr.org/2018/048

[11] Demirci, H., Selçuk, A.A.: 'A meet-in-the-middle attack on 8-round AES'. FSE 2008, Lausanne, Switzerland, February 2008, pp. 116–126

[12] Dunkelman, O., Keller, N., Shamir, A.: 'Improved single-key attacks on 8-round AES-192 and AES-256'. ASIACRYPT 2010, Singapore, December 2010, pp. 158–176

[13] Dunkelman, O., Keller, N., Shamir, A.: 'Improved single-key attacks on 8-round AES-192 and AES-256', *J. Cryptol.*, 2015, **28**, (3), pp. 397–422

[14] Derbez, P., Fouque, P.A., Jean, J.: 'Improved key recovery attacks on reduced-round AES in the single-key setting'. EUROCRYPT 2013, Athens, Greece, May 2013, pp. 371–387

[15] Derbez, P., Fouque, P.A.: 'Exhausting demirci-selçuk meet-in-the-middle attacks against reduced-round AES'. FSE 2013, Singapore, March 2013, pp. 541–560

[16] Li, L., Jia, K., Wang, X.: 'Improved single-key attacks on 9-round AES-192/256'. FSE 2014, London, UK, March 2014, pp. 127–146

[17] Li, R., Jin, C.: 'Meet-in-the-middle attacks on 10-round AES-256', *Des. Codes Cryptogr.*, 2016, **80**, (3), pp. 459–471

[18] Dobraunig, C., Eichlseder, M., Mendel, F.: 'Square attack on 7-round Kiasu-BC'. ACNS 2016, Guildford, UK, June 2016, pp. 500–517

[19] Dobraunig, C., List, E.: 'Impossible-differential and boomerang cryptanalysis of round-reduced Kiasu-BC'. CT-RSA 2017, San Francisco, USA, February 2017, pp. 207–222

[20] Tolba, M., Abdelkhalek, A., Youssef, A.M.: 'A meet in the middle attack on reduced-round Kiasu-BC', *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 2016, **E99-A**, (10), pp. 1888–1890

[21] Dobraunig, C., Eichlseder, M., Kales, D., *et al.*: 'Practical key-recovery attack on MANTIS5', *IACR Trans. Symmetric Cryptol.*, 2016, **2**, pp. 248–260

[22] Daemen, J., Rijmen, V.: 'AES proposal: Rijndael', 1998