The ISC Int'l Journal of Information Security
ISeCure

July 2018, Volume 10, Number 2 (pp. 93–105)
http://www.isecure-journal.org

Impossible Differential Cryptanalysis on Deoxys-BC-256

Alireza Mehrdad [1], Farokhlagha Moazami [1,*], and Hadi Soleimany [1]
[1] *Cyberspace Research Institute, Shahid Beheshti University, Tehran, Iran*

ARTICLE INFO.

Article history:
Received: 9 January 2018
First Revised: 20 January 2018
Last Revised: 4 May 2018
Accepted: 30 May 2018
Published Online: 5 July 2018

Keywords:
CAESAR Competition, Deoxys-BC, Impossible Differential Cryptanalysis, Distinguisher.

ABSTRACT

Deoxys is a final-round candidate of the CAESAR competition. Deoxys is built upon an internal tweakable block cipher Deoxys-BC, where in addition to the plaintext and key, it takes an extra non-secret input called a tweak. This paper presents the first impossible differential cryptanalysis of Deoxys-BC-256 which is used in Deoxys as an internal tweakable block cipher. First, we find a 4.5-round ID characteristic by utilizing a miss-in-the-middle-approach. We then present several cryptanalysis based upon the 4.5 rounds distinguisher against round-reduced Deoxys-BC-256 in both single-key and related-key settings. Our contributions include impossible differential attacks on up to 8-round Deoxys-BC-256 in the single-key model. Our attack reaches 9 rounds in the related-key related-tweak model which has a slightly higher data complexity than the best previous results obtained by a related-key related-tweak rectangle attack presented at FSE 2018, but requires a lower memory complexity with an equal time complexity.

1 Introduction

Recent real-world applications that need to protect both confidentiality and authentication have led to a renewed interest in designing novel authenticated encryption. Due to the lack of well-studied authenticated encryption schemes with the desirable level of security and performance, an ongoing CAESAR competition, funded by NIST, plans to identify a promising new portfolio of reliable and efficient authenticated encryptions that are suitable for widespread applications. A total of 58 diverse proposals from international cryptographers have been submitted on March 2014. According to the results of a public evaluation, the CAESAR committee has announced 7 schemes as the final round candidates.

Deoxys is one of the final-round authenticated encryption candidates in the CAESAR competition. Deoxys is built upon an internal tweakable block cipher Deoxys-BC, where in addition to the plaintext and key, it takes an extra non-secret input called a “tweak. Deoxys-BC is an AES-like design with the SPN structure which is based on the TWEAKEY framework. The inner tweakable block cipher Deoxys-BC has two variants, each with a block size of 128 bits and a tweak size of 128 bits, but two different key lengths: 128 and 256 bits. These variants are called Deoxys-BC-256 and Deoxys-BC-384, respectively. We note that the specification of Deoxys-BC has been slightly changed during the competition. In this paper, we study the last version submitted to the CAESAR competition called Deoxys v1.41.

The security of Deoxys-BC was studied against a wide variety of cryptanalyses by the designers and as was proved by them, the cipher is secure against several known attacks. However, impossible differential cryptanalysis was not covered by the designers in

[*] Corresponding author.
Email addresses: a.mehrdad@mail.sbu.ac.ir (A. Mehrdad), f_moazemi@sbu.ac.ir (F. Moazami), h_soleimany@sbu.ac.ir (H. Soleimany)

ISSN: 2008-2045 © 2018 ISC. All rights reserved.

the original proposal, instead, third-party experts are encouraged to investigate the security of Deoxys-BC against impossible differential cryptanalysis in different settings. The aim of this work is to evaluate the security of Deoxys-BC-256 against impossible differential cryptanalysis which is an important class of cryptanalytic techniques applicable to a wide variety of block ciphers. Impossible differential cryptanalysis was proposed by Knudsen and independently by Biham. Impossible differential cryptanalysis exploits differential characteristic with a probability of (exactly) zero to eliminate the wrong key candidates of some key bits involved in outer rounds that lead to such impossible differences.

Previous Work and Our Contributions

Carlos Cid *et al.* [1], present a related-key related-tweak rectangle attack against up to 9 rounds of Deoxys-BC-256 with a data complexity of 2^{117}, a memory complexity of 2^{117} states and a time complexity of 2^{118}. They also present a related-key related-tweak cryptanalysis on a particular variant of 10-round Deoxys-BC-256 in which the key length is greater than 204 and the tweak length is less than 52. The described cryptanalysis is not applicable to the cipher with the key length of 128-bits. In addition, the proposed cryptanalysis on the 10-round Deoxys-BC-256 requires $2^{127.58}$ chosen plainetexts while the maximum permitted amount of data for a given key in the Doexys scheme is 2^{t-4} where t denotes the size of the tweak.

In this paper, we present several impossible differential cryptanalysis on the round-reduced variants of Deoxys-BC-256:

- First, we study the security of Deoxys-BC-256 in the related-tweak single-key model. We describe how to mount an impossible differential attack on the 8 rounds of Deoxys-BC-256 given 2^{118} plaintext-ciphertext pairs and 2^{102} memories.
- After that we propose a related-key related-tweak impossible differential attack on 8-round Deoxys-BC-256 with a memory complexity of 2^{44}, a data complexity of $2^{116.5}$ chosen plaintexts and a time complexity of $2^{116.5}$ full encryptions. Then we exploit a precomputation phase to apply a similar attack on the 9 rounds of Deoxys-BC-256 which comes with the cost of increasing the required memory to 2^{102} states.

The results of our attacks compared with the previous attacks on Deoxys-BC-256 in the single-key and related-key models are summarized in Table 1. The designers presented an upper bound for an efficient related-key related-tweak differential cryptanalysis on 8 rounds of Deoxys-BC-256 without proposing a specific attack. However, our contributions include impossible differential attacks on 8-round Deoxys-BC-256 in the related-tweak single-key model. In addition, we present an impossible differential cryptanalysis on 9-round Deoxys-BC-256 in the related-key related-tweak model in which the required data is two times more than the rectangle attack while the memory complexity is decreased by a factor of 2^{15}.

After the submission of this paper and uploaded our results in Cryptology ePrint Archive, we noticed that Jiang and Jin presented a single-key impossible differential cryptanalysis on 8-Round Deoxys-BC-256 in [2]. In addition, during the preparation the final version of this paper Zong *et al.* published a key-recovery attack on 10 rounds of Deoxys-BC-256 which is applicable only when the key size ≥ 174 and the tweak size ≤ 82 [3]. However, the h permutation is not considered correctly in both published papers and it should be interpreted the other way. So the results are not valid. In addition, our paper includes more results in both single-key and related-key settings.

Outline of the Paper

The paper is organized as follows: Section 2 starts with a short description of Deoxys. This is followed by a brief introduction of the internal tweakable block cipher Deoxys-BC-256 and some notations that are used throughout the paper. After that we introduce a 4.5-round impossible differential characteristic which can be utilized in both single-key and related-key settings. Then we describe related-tweak impossible differential cryptanalysis on 8-round Deoxys-BC-256 in Section 4. We also present impossible differential characteristic of the 8-round and 9-round of the cipher in the related-key related-tweak model in Section 5 and Section 6, respectively. We conclude the paper in Section 7.

2 Description of Deoxys and Deoxys-BC

In this section, we describe Deoxys and Deoxys-BC-256. The section starts with a short description of Deoxys authenticated encryption. This is followed by a specification of the internal tweakable block cipher Deoxys-BC-256. We assume the reader is familiar with the concept of tweaks and keys for block ciphers and also the standard block cipher AES; otherwise, we refer to [5] and [6], respectively for the full specification details.

2.1 Deoxys Authenticated Encryption Scheme

The designers of Deoxys proposed two operating modes, called Deoxys-I and Deoxys-II. The former

Table 1. Results of attacks on Deoxys-BC-256.

Rds	Attack type	Attack mode	Key size	Tweak size	Time	Complexity Date (CP)	Memory	Ref.
8	MitM	SK	128	128	$\leq 2^{128}$	-	-	[4]
8	Differential	SK	128	128	$\leq 2^{128}$	-	-	[4]
8	Imp. dif.	SK	128	128	2^{118}	2^{118}	2^{102}	Section 4
8	Imp. dif.	RK	128	128	$2^{116.5}$	$2^{116.5}$	2^{44}	Section 5
9	Rectangle	RK	128	128	2^{118}	2^{117}	2^{117}	[1]
9	Imp. dif.	RK	128	128	2^{118}	2^{118}	2^{102}	Section 6

CP=chosen plaintext; RK= related-key; SK= single-key.

mode, Deoxys-I, is a nonce-based scheme which is proven to be secure against nonce-respecting adversaries. The latter mode, Deoxys-II, is a nonce-based AEAD scheme that provides security in the nonce misuse model in which the adversary can query different plaintexts while keeping the nonce constant. In this section, we only present a brief description of Deoxys-I. We refer the readers to the original proposal [4] for more details.

The encryption process, in the nonce-respecting mode with no padding is described in Table 2.

Table 2. Encryption algorithm when we have no padding to associated data and message.

```
Processing associated data
divide A to 128-bit blocks $A_1$ to $A_{la}$
Auth ← 0
for i = 0 to $la - 1$ do
    Auth ← Auth ⊕ $E_K(0010||i, A_{i+1})$
end
Message encryption and tag generation
divide M to 128-bit blocks $M_1$ to $M_l$
Checksum ← 0
for j = 0 to $l - 1$ do
    Checksum ← Checksum ⊕ $M_j$
    $C_j$ ← $E_K(0000||N||j, M_{j+1})$
end
Final ← $E_K(0001||N||l, Checksum)$
tag ← Final ⊕ Auth
```

2.2 Deoxys-BC-256

Deoxys utilizes a dedicated tweakable block cipher, Deoxys-BC as its internal encryption. The inner tweakable block cipher Deoxys-BC is an AES-based tweakable block cipher that makes use of the TWEAKEY framework. The TWEAKEY framework is a general method to concatenate the tweak and key as a unified state called tweakey. Deoxys-BC has two variants, each with a block size of 128 bits, but a different tweakey size of 128 and 256 bits which are called Deoxys-BC-256 and Deoxys-BC-384, respectively. Since the aim of this paper is to study the security of to Deoxys-BC-256 against impossible differential cryptanalysis, we only describe Deoxys-BC-256 in this section.

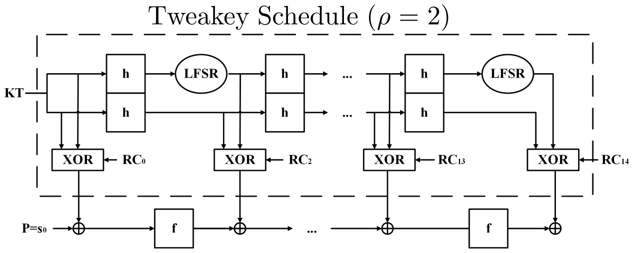

Figure 1. TWEAKEY framework for Deoxys-BC.

Deoxys-BC-256 has 14 rounds. The round function reuses the existing components of AES, with the main differences with the tweakeys that are used every round as the round subkeys. One round of the Deoxys-BC (f-function in Figure 1) consists of the following four transformations:

- **AddRoundTweakey** – XoR the subtweakey and internal state.
- **SubBytes** – Apply the AES S-box to the 16 bytes of the internal state.
- **ShiftRows** – Rotate i-th row left by i positions, where $i = (0, 1, 2, 3)$.
- **MixColumns** – Multiply the four input bytes in each column by the MDS matrix of AES.

To achieve the ciphertext, a final AddRoundTweakey operation is performed after the last round.

Definition of the subtweakeys.

Let KT be the concatenation of key and tweak. In Deoxys-BC-256, we denote the most significant 128-bit of KT by TK_0^1 and the least significant 128-bit of KT by TK_0^2. For Deoxys-BC-256, a subtweakey STK_i is defined as $STK_i = TK_i^1 \oplus TK_i^2 \oplus RC_i$ where TK_i^1 is the most significant 128-bit and TK_i^2 is the least significant 128-bit of the tweakey of round i.

The 128-bit words TK_{i+1}^j produces recursively from TK_i^j by a byte permutation h and an $LFSR$ as follows:

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

where the byte permutation h, is defined as:

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

in which we use the byte indexing as follows:

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

Also, the $LFSR$ function is defined as follows:

$$LFSR: \begin{pmatrix} (x_7 \parallel x_6 \parallel x_5 \parallel x_4 \parallel x_3 \parallel x_2 \parallel x_1 \parallel x_0) \\ \downarrow \\ (x_6 \parallel x_5 \parallel x_4 \parallel x_3 \parallel x_2 \parallel x_1 \parallel x_0 \parallel x_7 \oplus x_5) \end{pmatrix}$$

2.3 Notations

We use the following notations throughout the paper:

- x_i^I: The input of the round i.
- x_i^S: The SubBytes output of the round i.
- x_i^R: The ShiftRows output of the round i.
- x_i^M: The MixColumns output of the round i.
- x_i^O: The AddRoundTweakey output of the round i.
- $x_{i,col(j)}$: The j-th column of x_i, where j= (0,1,2,3).
- STK_i: The Subtweakey of the round i.
- K_i: The Subkey of the round i.
- T_i: The tweak of the round i.
- RC_i: The key schedule round constant of round i.
- TR_i: The result of XoRing of T_i and RC_i.

Also, we use the following enumeration $[0, 1, 2, \cdots, 15]$. By this enumeration, $x[l]$ represents the l-th byte of the x.

Since MixColumns and AddRoundTweakey operations are linear, they can be interchanged, that is, we can first do AddRoundTweakey and then MixColumns. Hence, we first begin by XoRing the internal state with a corresponding subkey and after that use the MixColumns and finally, XoR the obtained value with round tweak and RC_i. We indicate the corresponding subkey by $w_i = MC^{-1}(k_i)$. Let x_i^{Aw} represent the result of the XoRing of x_i^R and w_i of the round i.

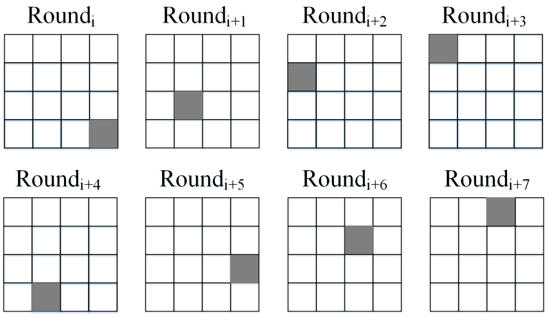

Figure 2. Subtweakeys difference Schedule used for impossible differential characteristic.

3 4.5-round Impossible Differential Characteristic

Impossible differential attack was first introduced at [7] and [8]. This attack mainly composed of two parts. The first section is to find impossible characteristic with maximum length. The second part of the attack is to use this characteristic to recover (part of) round subkeys, which is also called the key filtering step. This attack has become one of the most important attacks in the field of cryptanalysis, and has been used in several articles such as [9–13], to attack a large number of ciphers.

By the subtweakey schedule, one can easily check that if $\triangle STK_i[15]$ is an active byte then the structure of the subtweakey of other rounds is like Figure 2. Since the difference of subtweakeys is only due to the difference between tweaks and keys, the difference values of gray bytes can be zero in special cases.

That is to say, after eight rounds, the subtweakeys difference, is just like the arrangement of first round difference($\triangle STK_i = \triangle STK_{i+8}$). This repetition may be efficient for some future probable attack.

Figure 3 shows an illustration of an impossible differential of 4.5-round Deoxys. The gray boxes denote the (active) bytes in which the pair differs while the white boxes refer to the equal (passive) bytes in the pair. The black boxes refer to the byte that can be active or passive.

In forward direction, we use a tweakey difference with one non-zero difference byte $\triangle STK_i[0] \neq 0$ that leads to one active byte, $x_i^O[0]$. According to the process of producing subtweakey, we know $\triangle STK_i[0] \neq 0$ leads to $\triangle STK_{i+1}[7] \neq 0$. That leads to the five active bytes, $x_{i+1}^O[0, 1, 2, 3, 7]$. This process always gives eleven active bytes, $x_{i+2}^O[0, 1, 2, 3, 4, 5, 6, 7, 12, 13, 15]$, at the end of round i+2.

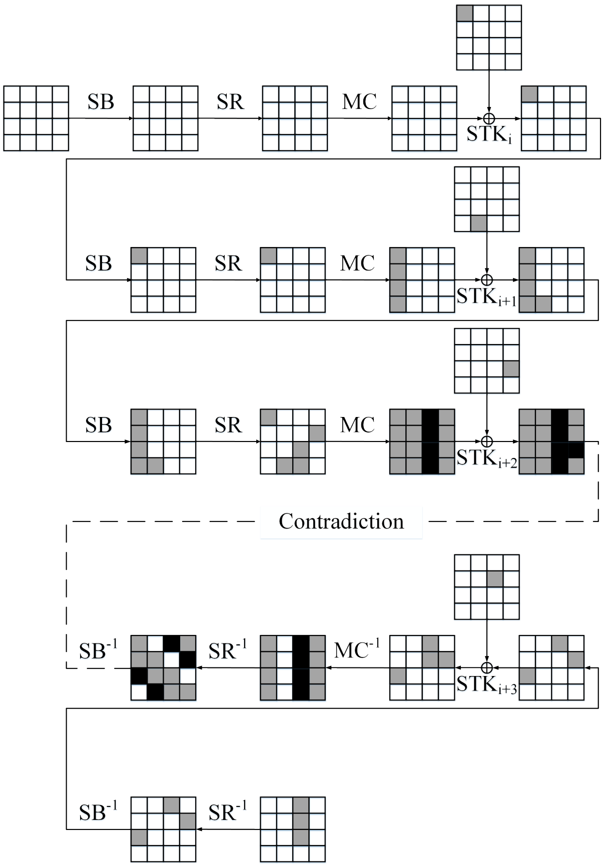

Figure 3. 4.5-round impossible differential characteristic of Deoxys

In backward direction, three active bytes, $x^R_{i+4}[8, 9, 10]$ or $x^R_{i+4}[8, 9, 11]$ or $x^R_{i+4}[8, 10, 11]$ afford one zero difference column in x^R_{i+3}. This passive column brings one zero difference byte at each column of x^I_{i+3} which contradicts with $\triangle x^O_{i+2,col(0,1)} \neq 0$.

Thus, according to this 4.5-round impossible differential, a plaintext pair which is equal at all bytes, after 4.5-round Deoxys encryption cannot convert to the ciphertext pair which is equal at all bytes except three bytes: $[8, 9, 10]$ or $[8, 9, 11]$ or $[8, 10, 11]$.

We will use this 4.5-round impossible characteristic for both the single key mode and related key mode. What's important is that in single key mode, the subtweakey differences are only caused by the difference of the tweaks ($\triangle STK_i = \triangle T_i$), but in the case of the related key attack, the subtweakey differences are due to the both differences of the tweaks and the keys($\triangle STK_i = \triangle T_i \oplus \triangle K_i$).

4 8-round Single-Key Impossible Differential Attack

We achieve a single key impossible differential cryptanalysis of 8-round Deoxys, using impossible differential characteristic of 4.5-round Deoxys as shown in Figure 3. We extend our impossible differential characteristic by one round at the beginning and 2.5-round at the end. Figure 4 shows this attack.

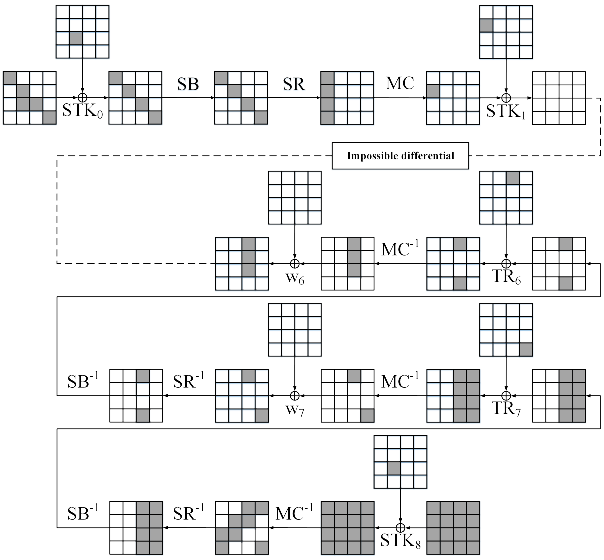

Figure 4. 8-round single key impossible differential trail

In this attack, we use an improvement that is suggested by Lu *et al.* [14] and is based on the following observation.

Observation I: Given a random pair $(\triangle X, \triangle Y)$ as input and output differences of the AES S-box, there is on average one pair of (X, X'), such that $S(X) \oplus S(X') = \triangle Y$.

Before we explain details of the attack, we define the concept of *structure* and *set* of plaintexts.

A structure L consists of 2^{40} plaintexts P_i which all of them are different in the bytes $P_i[0, 5, 6, 10, 15]$ and equal at other bytes. Each 2^{32} plaintexts P_i of one structure that have equal 6^{th} byte value of plaintexts P_i, form a set S. The value of $T^i_0[6]$ is equal to $P_i[6]$, so a dedicated tweak $T^i_0[6]$ (and $P_i[6]$) is assigned to each set. Clearly, there exist 2^8 sets in each structure. In our attack procedure, we need the pair of plaintext-tweaks $((P, T), (P', T'))$ such that $P \oplus P'$ has active bytes in positions $[0, 5, 6, 10, 15]$ and $P[6] \oplus P'[6] = T[6] \oplus T'[6]$. We can build about $2^{32} \times (2^8 - 1)^4 \approx 2^{64}$ distinct pairs that have four active bytes $[0, 5, 10, 15]$. Also, we can choose $\binom{2^8}{2}$ different sets from a structure to be sure that the 6^{th} byte is active too. Totally, we can build about $\binom{2^8}{2} \times 2^{32} \times (2^8 - 1)^4 \approx 2^{79}$ pairs per structure, that have five active bytes in positions $[0, 5, 6, 10, 15]$.

The attack procedure has two phases, online phase and precomputation (offline phase). Algorithm 1 illustrates a high level of the attack procedure. The details of the attack are as follows.

```
Algorithm 1 8-round Single-Key Attack on Deoxys-BC
Precomputation Phase make a list $H_p$ of $2^{40}$ valid first round constructions.
Online Phase
  for all $2^{118}$ chosen plaintexts do
    keep corresponding ciphertexts which have only two active last columns $x^I_{8,col(2,3)}$.
    for all $2^8$ possible values of $\triangle x^{Aw}_7[15]$ that leads to $\triangle x^R_8[3,6,9,12]$ do
      compute $w_8[3,6,9,12]$.
      for all $2^8$ possible values of $\triangle x^{Aw}_7[8]$ that leads to $\triangle x^R_8[2,5,8,15]$ do
        compute $w_8[2,5,8,15]$.
        for all $3 \times 255$ possible values of $\triangle x^{Aw}_{6,col(2)}$ that leads to $\triangle x^R_7[8,15]$ do
          compute $w_7[8,15]$.
          for all $2^{37}$ remaining pairs do
            discard each $STK_0[0,5,10,15]$ that is in $H_p$.
          end for
        end for
      end for
    end for
  end for
  Do exhaustive search for all remaining subkey bits.
```

4.1 Precomputation Phase

The number of pairs $(x^M_{1,col(0)}, {x'}^M_{1,col(0)})$ that are different only in byte position $x^M_1[1]$ and the difference is equal to the difference of two tweaks $T_1[1]$ and $T'_1[1]$ $(\triangle x^M_1[1] = \triangle T_1[1] \neq 0)$, is equal to $^8 \times (2^8 - 1) \times (2^8)^3 \approx 2^{40}$. For all these 2^{40} pairs, compute four bytes [0, 5, 10, 15] of x^I_1 and ${x'}^I_1$:
$x^I_1[0,5,10,15] = SB^{-1} \circ SR^{-1} \circ MC^{-1}(x^M_{1,col(0)})$ and
${x'}^I_1[0,5,10,15] = SB^{-1} \circ SR^{-1} \circ MC^{-1}({x'}^M_{1,col(0)})$.
Then, store the pairs $(x^I_1, {x'}^I_1)$ in a hash table H_p indexed by $(\triangle x^I_1 \mid\mid \triangle T_1[1])$. By considering the fact that $\triangle T_0[6] = T_0[6] \oplus T'_0[6] = \triangle T_1[1]$ the value of $(\triangle x^I_1 \mid\mid \triangle T_1[1])$ is equal to $(\triangle x^I_1 \mid\mid \triangle T_0[6])$. These parameters can take 2^{40} different values (2^{32} distinct values for $\triangle x^I_1$ and 2^8 unequal values for $\triangle T_0[6]$). Each value represents a row in the H_p. Since, we have 2^{40} pairs $(x^M_1, {x'}^M_1)$, then on average H_p has one pair $(x^I_1, {x'}^I_1)$ in each row; where first parameter specifies the difference $\triangle x^I_1$, and second parameter determines the difference $\triangle T_0[6]$.

4.2 Online Phase

(1) We take 2^n structures, which produce about $2^n \times 2^{79} = 2^{n+79}$ possible plaintext pairs. Then we ask for the corresponding ciphertexts: $C_i = E^{T^i}_K(P_i)$. The difference of last round subtweakey $\triangle STK_8$ is only dependent on the difference of tweaks $\triangle T_8$ that we know. So we can invert the final subtweakey XoR and mixcolumn and compute $\triangle x^R_8 = MC^{-1} \circ (\triangle C \oplus \triangle STK_8)$. We just select the pairs that corresponding to $\triangle x^R_8$, have eight active bytes $[2,3,5,6,8,9,12,15]$. Since we must have eight equal bytes $\triangle x^R_8$ in positions [0,1,4,7,10,11,13,14], the expected number of remaining pairs is $2^{n+79} \times 2^{-64} = 2^{n+15}$.

(2) Since we know the value of (C, C') and $\triangle STK_8$, the difference $\triangle x^S_{8,col(3)}$ can be determined. Thus, by knowing the value of $\triangle x^{Aw}_{7,col(3)}$, we can obtain the values of $x^S_{8,col(3)}$ and ${x'}^S_{8,col(3)}$ according to observation I. Since we know $\triangle x^{Aw}_7[15] \neq 0$, we have only $2^8 - 1$ different possible values of $\triangle x^{Aw}_{7,col(3)}$. Therefore, this step can be done as follows:
Initialize 2^{32} empty lists, for each guess of $w_8[3,6,9,12]$.
For each remaining pair (C, C'), and for each possible value of $\triangle x^{Aw}_7[15]$, calculate the key $w_8[3,6,9,12]$ that leads the pair (C, C') to $\triangle x^{Aw}_{7,col(3)}$, and add this pair to the list related to the guessed key.
Due to observation I, for each pair and distinction guess, on average we have one key suggestion. Since these $2^{n+15} \times 255 \approx 2^{n+23}$ suggestions are distributed over all 2^{32} possible keys, we have about $2^{n+23}/2^{32} = 2^{n-9}$ pairs for each guess of $w_8[3,6,9,12]$.

(3) Similar to step 2, we initialize 2^{32} empty lists for each guess of $w_8[2,5,8,15]$.
For each remaining pair (C, C'), and for each possible value of $\triangle x^{Aw}_7[8]$, calculate the key $w_8[2,5,8,15]$ that leads the pair (C, C') to $\triangle x^{Aw}_{7,col(2)}$, and add this pair to the list related to the guessed key.
Due to observation I, for each pair and distinction guess, on average we have one suggested key. Since these $2^{n-9} \times 255 \approx 2^{n-1}$ suggestions are distributed over all 2^{32} possible keys, we have about $2^{n-1}/2^{32} = 2^{n-33}$ pairs for each guess of $w_8[2,5,8,15]$.

(4) We use Lu *et al.* improvement again. Since we want $\triangle x^{Aw}_{6,col(2)}$ to have an active byte in the 8^{th} position and two of three other bytes, there are 3×255^3 possible differences and only 3×255 of these differences lead to a difference of $\triangle x^M_{6,col(2)}$ where only two bytes $\triangle x^M_6[8,11]$ are active. So, for each pair and each guess of $w_7[8,15]$ we must check whether $\triangle x^M_{6,col(2)}$ belongs to these 3×255 differences. According to observation I, when we have $\triangle x^{Aw}_7[8,15]$ as output and $\triangle x^M_6[8,11]$ as input difference of the S-box, we can compute

the values of $x_6^{M}[8,11]$ and ${x'}_6^{M}[8,11]$ and therefore determine the value of $w_7[8,15]$. At this step, we have about $3\times 2^{n-25}$ candidates for $w_7[8,15]$. From the 2^{n-33} pairs and the 3×255 differences which are distributed over the 2^{16} possible values of $w_7[8,15]$. Consequently, for a given guess of $w_7[8,15]$, we have about $3\times 2^{n-25}/2^{16} = 3\times 2^{n-41}$ pairs which for each guess of the considered bytes in w_8 and w_7, lead the input difference to the impossible differential.

(5) First we create a list A of all 2^{32} 4-byte keys $STK_0[0,5,10,15]$ and for all remaining pairs (P_i, P_j), we compute four bytes [0, 5, 10, 15] of x_1^I and ${x'}_1^I$:
$x_1^I = P_i[0,5,10,15] \oplus STK_0[0,5,10,15]$,
${x'}_1^I = P_j[0,5,10,15] \oplus STK_0[0,5,10,15]$.
Note that the STK_0 only has one non-zero difference byte $\triangle STK_0[6] \neq 0$, and $\triangle STK_0[6] = \triangle P[6]$.
From precomputation, we know on average H_p has one pair $(x_1^I, {x'}_1^I)$ in each row. For each tuple $(x_1^I, {x'}_1^I, \triangle T_0[6])$ which is obtained at this stage, we discard the $P \oplus x_1^I$ from the related indexed row of the hash table. Since with respect to the precomputation (offline phase), we are sure that such a a key leads to an impossible differential, resulting in a wrong key.

Finally, if A is not empty, output the remaining value(s) in A with corresponding key guess of $w_7[8,15]$ and $w_8[2,3,5,6,8,9,12,15]$.

4.3 Complexity Analysis

- Data Complexity

  We know $2^{-(c_{in}+c_{out})} = 2^{-32}\times 2^{-48}\times 3\times 2^{-8} \approx 2^{-86.4}$ and we guessed 32-bit of k_{in} and $(64+16)$-bit of K_{out}. So, we can easily compute the data complexity D:

$$(1-2^{-(86.4)})^D < 1/2^{112} \rightarrow e^{-(2^{-86.4}\times D)} < 1/2^{112} \rightarrow$$
$$\rightarrow D \approx 2^{93} = 2^{n+15} \rightarrow n = 78.$$

  Since $n = 78$ then $2^{78}\times 2^{40} = 2^{118}$ chosen plaintexts, are required for the attack.

- Time Complexity

  (1) The **precomputation** requires about $2\times 2^{40}\times 4/16 = 2^{39}$ one-round decryptions, which is equal to $2^{39}/8 = 2^{36}$ 8-round decryptions.

  (2) Since we need n to be equal to 78, **step 1** requires $2^{(78+40)} = 2^{118}$ 8-round encryptions.

  (3) Based on observation I, **step 2** can be done by a look-up table. So, this step needs about $255\times 2^{78+15}\approx 2^{101}$ memory accesses.

  (4) We considered 255 differences of the 32-bit key guesses $w_8[3,6,9,12]$ and 255 differences for 32-bit guesses of key $w_8[2,5,8,15]$. Therefore, **Step 3** requires about $255\times 255\times 2^{78+15}\approx 2^{109}$ memory accesses.

  (5) For each 2^{64} guesses of $w_8[2,3,5,6,8,9,12,15]$, we need $2^{78-33}\times 3\times 255 \approx 2^{78-23.4}$ memory accesses in a lookup table to achieve the guess for $w_7[8,15]$ from the differences $\triangle x_6^{M,col(2)}$. So, **step 4** requires about $2^{64}\times 2^{78-23.4} = 2^{118.6}$ memory accesses.

  (6) For each remaining pair, **step 5** is repeated 2^{80} times (for each possible values of w_7 and w_8), and on average for each repetition, we need to access to hash table Hp and list A. So, this step requires about $2\times 2^{80}\times 2^{78-39.4} = 2^{119.6}$ memory accesses.

  (7) We already have obtained eight bytes of the key w_8 and an **exhaustive search** is needed to achieve the remaining key bytes which cost $2^{8\times 8} = 2^{64}$ encryption. But the time complexity of this step is negligible compared to the other steps.

  As a rule of thumb method adopted in most of the works (e.g. [12]), we assume the time complexity of 16 memory access equals to the time complexity of one round, since in each round the Sbox is called 16 times. Consequently, the 8 rounds are at least equal to $16\times 8 = 2^7$ memory accesses. So we can assume that each memory access is equivalent to $1/(16\times 8) = 2^{-7}$ 8-round encryption.

  Totally, time complexity is about $(2^{36} + 2^{118})Enc + (2^{101} + 2^{109} + 2^{118.6} + 2^{119.6})MA \approx (2^{118} + 2^{120.2}\times 2^{-7})Enc \approx 2^{118}Enc.$

- Memory Complexity

  The **precomputation** phase needs about $2^{40}\times(4+4+1)\approx 2^{43.2}$ bytes of memory for storing $x_1^I[0,5,10,15]$, $x_1^I[0,5,10,15]$ and $\triangle T_0[6]$. With the simple approach, we need $2^{8\times(8+2+4)}$ bytes to store the deleted values of $w_8[2,3,5,6,8,9,12,15]$, $w_7[8,15]$ and $K_0[0,5,10,15]$. But if we use Lu *et al.* improvement, we can apply the attack individually for each guess of the key; and for the remaining bytes of each guess that is not discarded, perform an exhaustive search. So, instead of the simple approach, we can store about $2^{n+23} = 2^{101}$ suggestions that remain after step 2. Each suggestion consists of one pair. So, the memory complexity of the attack is about $2^{43.2} + (2^{101}\times 2\times 16)\approx 2^{106}$ bytes or 2^{102} states.

5 8-round Related-Tweakey Impossible Differential Attack

In this section, we present a related key impossible differential cryptanalysis of 8-round Deoxys by ex-

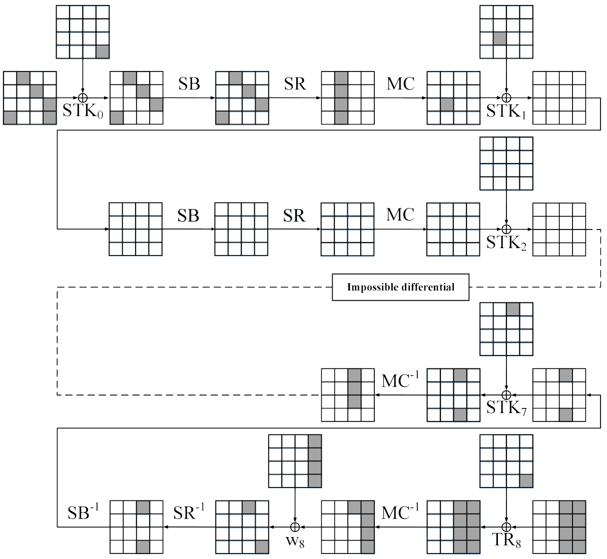

Figure 5. 8-round related key impossible differential trail

tending our impossible differential characteristic by two rounds at the beginning and 1.5-round at the end. This attack on the reduced 8-round Deoxys requires about $2^{116.5}$ chosen plaintexts, 2^{48} words of memory and $2^{116.5}$ 8-round Deoxys encryptions. Figure 5 illustrates this attack.

For our analysis, we consider a situation in which the value of the key difference in round 2 is exactly equal to the value of the tweak difference in the second round ($\triangle K_2[1] = \triangle T_2[1]$). In other words, we assume that two users encrypt data with two different keys, and that these two keys have a non-zero difference of one byte, $\triangle K_0[15]$. In this case, we select the tweaks in a way that the value of the $\triangle K_2[1]$ exactly matches the value of the $\triangle T_2[1]$. Considering this condition, the definition of *structure* and *set* of plaintexts is a little different from what described in Section 4.

A structure L consists of 2^{40} plaintexts P_i all of which are different in bytes $P_i[3,4,9,14,15]$ and equal at other bytes. Each 2^{32} plaintexts P_i of one structure that have equal 15^{th} byte value of plaintexts P_i, form a set S. The value of $STK_0^i[15]$ is equal to $P_i[15]$, so a dedicated $STK_0^i[15]$ (and $P_i[15]$) is assigned to each set. Clearly, there exist 2^8 sets in each structure. In our attack procedure, we need the pair of plaintext-tweakeys $((P, STK),(P', STK'))$ such that $P \oplus P'$ has an active byte in positions $[3, 4, 9, 14, 15]$ and $P[15] \oplus P'[15] = STK[15] \oplus STK'[15]$. We can build about $2^{32} \times (2^8 - 1)^4 \approx 2^{64}$ distinct pairs that have four active bytes $[3, 4, 9, 14]$. Also, we can choose $\binom{2^8}{2}$ different sets from a structure to be sure that the 15^{th} byte is active too. Totally, we can build about $\binom{2^8}{2} \times 2^{32} \times (2^8 - 1)^4 \approx 2^{79}$ pairs per structure, that have five active bytes in positions $[3, 4, 9, 14, 15]$.

5.1 Attack Procedure

Algorithm 2 illustrates a high level of the attack procedure. In what follows, we describe the attack procedure in details:

Algorithm 2 8-round Related-Key Attack on Deoxys-BC

```
for all $K_2[1] = T_2[1]$ & $K_2'[1] = T_2'[1]$ do
    for all $2^{116.5}$ chosen plaintexts do
        keep corresponding ciphertexts which have only two active last columns.
        for all $2^{91.5}$ remaining ciphertext pairs do
            check if only two bytes $x_8^R[8, 15]$ are active.
            for all $2^{16}$ possible values of $w_8[8, 15]$ do
                compute $x^O_{7,col(2)}$.
                for all $2^{43.5}$ remaining pairs do
                    check if $x^R_{7,col(2)}$ have only one passive byte at $x_7^R[9]$ or $x_7^R[10]$ or $x_7^R[11]$.
                    for all remaining pairs do
                        discard each $STK_0[3, 4, 9, 14]$ leads to passive $x_1^O$.
                    end for
                end for
            end for
        end for
    end for
end for
Do exhaustive search for all remaining subkey bits.
```

(1) We take $2^n \times 2^{79} = 2^{n+79}$ possible plaintext pairs. Then we ask for the corresponding ciphertexts: $C_i = E_{K_i}^{T^i}(P_i)$. We just select the pairs, so that corresponding pairs $(x_8^M, {x'}_8^M)$, have just eight active bytes in the last two columns. So the expected number of remaining pairs is $2^{n+79} \times 2^{-64} = 2^{n+15}$.

(2) For all pairs $(x_8^M, {x'}_8^M)$ that passed step 1, we compute $\triangle x^{Aw}_{8,col(2,3)}$: $\triangle x_8^{Aw} = MC^{-1} \circ (\triangle x_8^M)$. We keep pairs where only $\triangle x_8^{Aw}[8, 12, 13, 14, 15]$ are active bytes and also the differences of bytes in the positions $\triangle x_8^{Aw}[12, 13, 14]$ are equal to the differences of bytes in the positions $\triangle w_8[12, 13, 14]$. Since we must have three zero-difference bytes, $\triangle x_8^{Aw}[9, 10, 11] = 0$ and three special difference bytes $\triangle x_8^{Aw}[12, 13, 14] = \triangle w_8[12, 13, 14]$, the number of remaining pairs is $2^{n+15} \times (2^{-8})^3 \times (2^{-8})^3 = 2^{n-33}$.

(3) We guess the 16-bit values of $w_8[8, 15]$. Since we know the relation of subtweakeys, we can compute $w_8'[15]$ easily. Then for each pairs $(C, w_8), (C', w_8')$ that has passed step 2, compute four bytes of $x^O_{7,col(2)}$ and ${x'}^O_{7,col(2)}$:
$x_7^O = SB^{-1} \circ SR^{-1} \circ (w_8 \oplus (MC^{-1} \circ (TR_8 \oplus C)))$,
${x'}_7^O = SB^{-1} \circ SR^{-1} \circ (w_8' \oplus (MC^{-1} \circ (TR_8' \oplus C')))$.

From step 1 we are sure that only two bytes $\triangle x_7^O[8,11]$ are active. So we have no filtering here.

(4) Consider the value of $\triangle STK_7[8]$, check that $\triangle x^R_{7,col(2)}$ has three active bytes in positions $[8,9,11]$ or $[8,9,11]$ or $[8,10,11]$. At the end of this step, the expected number of remaining pairs is about $2^{n-33} \times 2^{-8} \times 3 \approx 2^{n-39.4}$.

(5) We guess 32-bit values of $STK_0[3,4,9,14]$ and for all remaining pairs from the above steps, we compute four-bytes $x^M_{1,col(1)}$ and $x'^M_{1,col(1)}$:
$x_1^M = MC \circ SR \circ SB \circ (P \oplus STK_0)$,
$x'^M_1 = MC \circ SR \circ SB \circ (P' \oplus STK'_0)$.
We only consider the pairs that $\triangle x^M_{1,col(1)} = \triangle STK_{1,col(1)}$. So, we only choose pairs in which $\triangle x_1^M[6] = \triangle STK_1[6]$ and $\triangle x_1^M[4,5,7] = 0$. In other word, we only choose pairs that at the end of round one, we are sure that there is no active byte at $\triangle x^O_{1,col(1)}$. Since, we must have four zero-difference bytes $\triangle x^O_{1,col(1)} = 0$, the number of remaining pairs is about $2^{n-39.4} \times (2^{-8})^4 = 2^{n-71.4}$.
Since we initially considered the difference $\triangle K_2[1]$ equal to $\triangle T_2[1]$, then we are sure that the differential characteristic passes the forward path correctly.
The keys that pass all the above steps and lead such a difference (that is impossible), are wrong keys and must be discarded. We remove such a key K_0 for each 16-bit guess of output corresponding key w_8. Since only one of the keys is the correct key, if we choose proper data complexity and perform the above operation for all remaining pairs of step 4, we can be sure we have reached the correct key.

5.2 Complexity Analysis

- Data Complexity
  The bit conditions are about $2^{-(c_{in}+c_{out})} = 2^{-32} \times 2^{-48} \times 3 \times 2^{-8} \approx 2^{-86.4}$ and $| k_{in} \bigcup k_{out} |$ is equal to $32 + 16 = 48$ so the data complexity D is:
  $$(1 - 2^{-(86.4)})^D < 1/2^{48} \rightarrow e^{-(2^{-86.4} \times D)} < 1/2^{48} \rightarrow$$
  $$\rightarrow D \approx 2^{91.5} = 2^{n+15} \rightarrow n = 76.5.$$
  Since $n = 76.5$ then $2^{76.5} \times 2^{40} = 2^{116.5}$ chosen plaintexts, are required for the attack.
- Time Complexity
  (1) **Step 1** requires $2^{(76.5+40)} = 2^{116.5}$ 8-round encryptions.
  (2) Complexity of **step 3** is about $2 \times 2^{16} \times 2^{(76.5-33)} = 2^{60.5}$ one-round 4/16 decryptions, which means $2^{60.5} \times 4/16 \times 1/8 = 2^{55.5}$ 8-round encryptions.

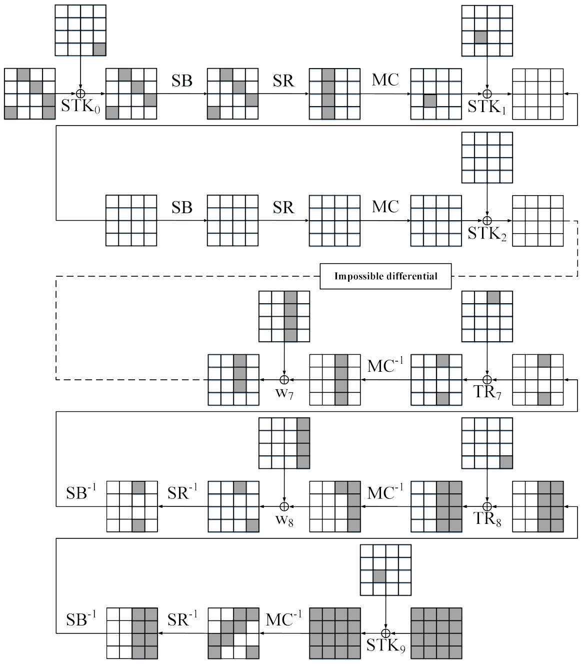

Figure 6. 9-round related key impossible differential trail

  (3) **Step 5** needs about $2 \times 2^{16} \times 2^{32} \times 2^{76.5-39.4} = 2^{86.1}$ one-round 4/16 encryptions, which is equal to $2^{86.1} \times 4/16 \times 1/8 = 2^{81.1}$ 8-round encryptions.
  Consequently, total complexity is about $(2^{116.5} + 2^{55.5} + 2^{81.1})Enc \approx 2^{116.5}Enc$.
- Memory Complexity
  For storing the list of discard keys, we need $2^{8\times(2+4)} = 2^{48}$ bytes of memory for storing the deleted values of $w_8[8,15]$ and $K_0[3,4,9,14]$. Therefore, memory complexity is 2^{48} bytes or 2^{44} states.

6 9-round Related-Tweakey Impossible Differential Attack

Similar to the attack that was applied to the 8-round Deoxys, we can analyse the 9-round Deoxys, using impossible differential characteristic of 4.5-round Deoxys as shown in Figure 3. We extend our impossible differential by two rounds at the beginning and 2.5-round at the end. Figure 6 shows this attack. In this section, we use observation I again.

The attack procedure has two phases, online phase and precomputation (offline phase). Algorithm 3 illustrates a high level of the attack procedure. The details of the attack are as follows.

6.1 Precomputation Phase

The number of pairs $(x^M_{1,col(1)}, x'^M_{1,col(1)})$ that are different only in byte position $x_1^M[6]$ and the difference is equal to the difference of two subtweakeys $STK_1[6]$

Algorithm 3 9-round Related-Key Attack on Deoxys-BC

```
Precomputation Phase make a list $H_p$ of $2^{40}$ valid first round constructions.
Online Phase
  for all $K_2[1] = T_2[1]$ & $K'_2[1] = T'_2[1]$ do
    for all $2^{118}$ chosen plaintexts do
      keep corresponding ciphertexts which have only two active last columns $x^I_{9,col(2,3)}$.
      for all $2^8$ possible values of $\triangle x^{Aw}_8[15]$ that leads to $\triangle x^R_9[3,6,9,12]$ do
        compute $w_9[3,6,9,12]$.
        for all $2^8$ possible values of $\triangle x^{Aw}_8[8]$ that leads to $\triangle x^R_9[2,5,8,15]$ do
          compute $w_9[2,5,8,15]$.
          for all $3 \times 255$ possible values of $\triangle x^{Aw}_{7,col(2)}$ that leads to $\triangle x^R_8[8,15]$ do
            compute $w_8[8,15]$.
            for all $2^{37}$ remaining pairs do
              discard each $STK_0[3,4,9,14]$ that is in $H_p$.
            end for
          end for
        end for
      end for
    end for
  end for
  Do exhaustive search for all remaining subkey bits.
```

and $STK'_1[6]$ ($\triangle x^M_1[6] = \triangle STK_1[6] \neq 0$), is equal to $2^8 \times (2^8-1) \times (2^8)^3 \approx 2^{40}$. For all these 2^{40} pairs, compute four bytes [0, 5, 10, 15] of x^I_1 and x'^I_1:
$x^I_1[3,4,9,14] = SB^{-1} \circ SR^{-1} \circ MC^{-1}(x^M_{1,col(1)})$ and
$x'^I_1[3,4,9,14] = SB^{-1} \circ SR^{-1} \circ MC^{-1}(x'^M_{1,col(1)})$.
Since the $\triangle STK_1[6]$ leads to a special $\triangle STK_0[15]$, we can store the pairs (x^I_1, x'^I_1) in a hash table H_p indexed by $(\triangle x^I_1 \mid\mid \triangle STK_0[15])$. These parameters can take 2^{40} different values (2^{32} distinct values for $\triangle x^I_1$ and 2^8 unequal values for $\triangle STK_0[15]$). Each value represents a row in H_p. Since, we have 2^{40} pairs (x^M_1, x'^M_1), then on average H_p has one pair (x^I_1, x'^I_1) in each row. In which the first parameter specifies the value of $\triangle x^I_1$, and the second parameter determines the value of $\triangle STK_0[15]$.

6.2 Online Phase

(1) We take 2^n structures, which produce about $2^n \times 2^{79} = 2^{n+79}$ possible plaintext pairs. Then we ask for the corresponding ciphertexts: $C_i = E^{T^i}_K(P_i)$. We can invert the final subtweaky XoR and compute $\triangle x^R_9 = MC^{-1} \circ (\triangle C \oplus \triangle STK_9)$. We just select the pairs that corresponding $\triangle x^R_9$, have eight active bytes $[2,3,5,6,8,9,12,15]$. Since we must have eight equal bytes $\triangle x^R_9$ in positions [0,1,4,7,10,11,13,14], the expected number of remaining pairs is $2^{n+79} \times 2^{-64} = 2^{n+15}$.

(2) Since we know the value of (C, C') and $\triangle STK_9$, the difference $\triangle x^S_{9,col(3)}$ can be determined. Thus, by knowing the value of $\triangle x^{Aw}_{8,col(3)}$, we can obtain the values of $x^S_{9,col(3)}$ and $x'^S_{9,col(3)}$ according to observation I. Since we have 2^8-1 different values of $\triangle x^R_8[15]$, because of fix subkey difference notice to the tweak difference, finally we have only 2^8-1 different values of $\triangle x^{Aw}_{8,col(3)}$. Therefore, this step can be done as follows:
Initialize 2^{32} empty lists, for each guess of $w_9[3,6,9,12]$ we can easily obtain the value of $w'_9[3,6,9,12]$, which is different from $w_9[3,6,9,12]$ only at $w_9[6]$ due to the subtweakey difference schedule.
For each remaining pair (C, C'), and for each possible value of $\triangle x^{Aw}_{8,col(3)}$, calculate the key $w_9[3,6,9,12]$ that leads the pair (C, C') to $\triangle x^{Aw}_{8,col(3)}$ and add this pair to the list related to the guessed key.
Due to observation I, for each pair and distinction guess, on average we have one key suggestion. Since these $2^{n+15} \times 2^8 = 2^{n+23}$ suggestions are distributed over all 2^{32} possible keys, we have about $2^{n+23}/2^{32} = 2^{n-9}$ pairs for each guess of $w_9[3,6,9,12]$.

(3) Similar to step 2, we initialize 2^{32} empty lists for each guess of $w_9[2,5,8,15]$.
For each remaining pair (C, C'), and for each possible value of $\triangle x^{Aw}_8[8]$, calculate the key $w_9[2,5,8,15]$ that leads the pair (C, C') to $\triangle x^{Aw}_{8,col(2)}$, and add this pair to the list related to the guessed key.
Due to observation I, for each pair and distinction guess, on average, we have one suggested key. Since these $2^{n-9} \times 255 \approx 2^{n-1}$ suggestions are distributed over all 2^{32} possible keys, we have about $2^{n-1}/2^{32} = 2^{n-33}$ pairs for each guess of $w_9[2,5,8,15]$.

(4) We use Lu *et al.* improvement again. Since we want $\triangle x^R_{7,col(2)}$ to have an active byte in the 8^{th} position and two of other three bytes and also the subkey difference is fix, there are 3×255^3 possible differences for $\triangle x^{Aw}_{7,col(2)}$ and only $3 \times 255^3/255^2 = 3 \times 255$ of these differences lead to a difference $\triangle x^M_{7,col(2)}$ so that only two bytes $\triangle x^M_7[8,11]$ are active.

So, for each pair and each guess of $w_8[8,15]$ we must check whether $\triangle x^M_{7,col(2)}$ belongs to these 3×255 differences. According to observation I, when we have $\triangle x^R_8[8,15]$ as output and $\triangle x^M_7[8,11]$ as input difference of the S-box, we can compute the values of $x^M_7[8,11]$

and ${x'}_7^M[8,11]$ and therefore determine the value of $w_8[8,15]$. At this step, we have about $3 \times 255 \times 2^{-33} = 3 \times 2^{n-25}$ candidates for $w_8[8,15]$. From the 2^{n-33} pairs and the 3×255 differences which are distributed over the 2^{16} possible values of $w_8[8,15]$. Consequently, for a given guess of $w_8[8,15]$, we have about $3 \times 2^{n-25}/2^{16} = 3 \times 2^{n-41}$ pairs which for each guess of the considered bytes in w_9 and w_8, lead the input difference to the impossible differential.

(5) First we create a list A of all 2^{32} 4-byte keys $STK_0[3,4,9,14]$ and for all remaining pairs (P_i, P_j), we compute four bytes [3, 4, 9, 14] of x_1^I and ${x'}_1^I$:
$x_1^I = P_i[3,4,9,14] \oplus STK_0[3,4,9,14]$,
${x'}_1^I = P_j[3,4,9,14] \oplus STK_0[3,4,9,14]$.
Note that the STK_0 only has one non-zero difference byte $\triangle STK_0[15] \neq 0$, which $\triangle STK_0[15] = \triangle P[15]$.
From precomputation, we know on average H_p has one pair $(x_1^I, {x'}_1^I)$ in each row. For each tuple $(x_1^I, {x'}_1^I, \triangle STK_0[15])$, which is obtained at this stage, we discard the $P \oplus x_1^I$ from the related indexed row of the hash table. Since with respect to the precomputation (offline phase), we are sure that such a key leads to the impossible differential, resulting in a wrong key. Since we initially considered the difference $\triangle K_2[1]$ to be equal to $\triangle T_2[1]$, then we are sure that the differential characteristic passes the forward path correctly.

Finally, if A is not empty, output the remaining value(s) in A with corresponding key guess of $w_8[8,15]$ and $w_9[2,3,5,6,8,9,12,15]$.

6.3 Complexity Analysis

- Data Complexity

  The data complexity D is:
  $(1 - 2^{-(86.4)})^D < 1/2^{112} \rightarrow e^{-(2^{-86.4} \times D)} < 1/2^{112} \rightarrow$
  $\rightarrow D \approx 2^{93} = 2^{n+15} \rightarrow n = 78.$

  Where $2^{-(c_{in}+c_{out})} = 2^{-32} \times 2^{-48} \times 3 \times 2^{-8} \approx 2^{-86.4}$ and $\mid k_{in} \bigcup k_{out} \mid$ is equal to $32+64+16 = 112$. Since $n = 78$ then $2^{78} \times 2^{40} = 2^{118}$ chosen plaintexts, are required for the attack.
- Time Complexity

  (1) The **precomputation** requires about $2 \times 2^{40} \times 4/16 = 2^{39}$ one-round decryptions, which is equal to $2^{39}/9 \approx 2^{35.9}$ 9-round decryptions.

  (2) Since n was considered to be 78, **step 1** requires $2^{(78+40)} = 2^{118}$ 9-round encryptions.

  (3) Based on Lu *et al.* method, **step 2** can be done by a look-up table. So, this step needs about $255 \times 2^{78+15} \approx 2^{101}$ memory accesses.

  (4) **Step 3** requires about $255 \times 255 \times 2^{78+15} \approx 2^{109}$ memory accesses.

  (5) For each 2^{64} guesses of $w_9[2,3,5,6,8,9,12,15]$, we need $2^{78-33} \times 3 \times 255 \approx 2^{78-23.4}$ memory accesses in a lookup table to achieve the guess for $w_8[8,15]$ from the differences $\triangle x_7^{M,col(2)}$. So, **step 4** requires about $2^{64} \times 2^{78-23.4} = 2^{118.6}$ memory accesses.

  (6) For each remaining pair, **step 5** is repeated 2^{80} times (for each possible values of w_8 and w_9), and on average for each repetition, we need to access to hash table Hp and list A. So, this step require about $2 \times 2^{80} \times 2^{78-39.4} = 2^{119.6}$ memory accesses.

  (7) **Exhaustive search** is negligible.

  As it is mentioned before in Section 4, the time complexity of 9 rounds is equal to at least 16×9 memory access. So we can estimate each memory access as $1/(16 \times 9) \approx 2^{-7}$ 9-round encryption.

  Totally, time complexity is about $(2^{35.9} + 2^{118})Enc + (2^{101} + 2^{109} + 2^{118.6} + 2^{119.6})MA \approx (2^{118} + 2^{120.2} \times 2^{-7})Enc \approx 2^{118}Enc.$
- Memory Complexity

  The **precomputation** phase needs about $2^{40} \times (4+4+1) \approx 2^{43.2}$ bytes of memory for storing $x_1^I[3,4,9,14]$, $x_1^I[3,4,9,14]$ and $\triangle STK_0[15]$. we apply the attack individually for each guess of the key and for the remaining bytes of each guess that is not discarded, perform an exhaustive search. So, we store about $2^{n+31} = 2^{109}$ suggestions that remain after Step 2. Each suggestion consists of one pair. So, the memory complexity of the attack is about $2^{43.2} + 2^{114} \approx 2^{114}$ bytes or 2^{110} states.

7 Conclusion

This paper describes several impossible differential cryptanalysis on the round-reduced variants of Deoxys-BC-256. As a possible direction for future research, one can investigate the security of Deoxys-BC-256 against impossible differential by considering a beyond full-codebook scenario, since the tweak in Deoxys-BC can provide extra plaintext-ciphertext pairs in contradiction to the classical model.

This paper describes several impossible differential cryptanalysis on the round-reduced variants of Deoxys-BC-256. This work presents the first third-party cryptanalysis of the tweakable block cipher Deoxys-BC-256 in the single-key model. We also propose impossible differential attacks up to the 9-round Deoxys-BC-256 in the related-tweak related-key model which has a lower memory complexity than the best previous attack.

The cryptanalysis presented in this paper cannot be exploited to mount a key-recovery attack on Deoxys-II authenticated encryption scheme. However, as it is discussed in Section 6 of [1] the results can be applied on the Deoxys-I authenticated encryption.

As a possible direction for future research, one can investigate the security of Deoxys-BC-256 against impossible differential by considering a beyond full-codebook scenario, since the tweak in Deoxys-BC can provide extra plaintext-ciphertext pairs in contradiction to the classical model.

8 Acknowledgement

The work of Hadi Soleimany is partly supported by grants from Shahid Beheshti University and by the Iranian National Science Foundation (grant number 95835673).

References

[1] C. Cid, T. Huang, T. Peyrin, Y. Sasaki, and L. Song, “A security analysis of Deoxys and its internal tweakable block ciphers”, IACR Transactions on Symmetric Cryptology, 2017(3):73107, 2017.

[2] Z. Jiang and C. Jin, “Impossible Differential Cryptanalysis of 8-Round Deoxys-BC-256”, IEEE Access, Vol. 6, pp. 8890–8895, 2018.

[3] R. Zong, X. Dong, X. Wang, “Related-Tweakey Impossible Differential Attack on Reduced-Round Deoxys-BC-256”, SCIENCE CHINA Information Sciences.

[4] J. Jean, I. Nikolic, T. Peyrin, and Y. Seurin, “Deoxys v1.41”, Submitted to CAESAR, October 2016.

[5] J. Jean, I. Nikolić, and T. Peyrin, “Tweaks and Keys for Block Ciphers : the TWEAKEY Framework”, Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II, volume 8874 of Lecture Notes in Computer Science, pages 274288. Springer, 2014.

[6] J. Daemen, V. Rijmen, “AES Proposal : Rijndael”, NIST AES proposal, 1998.

[7] E. Biham, A. Biryukov, A. Shamir, “Miss in the middle attacks on IDEA and Khufu”, In L. Knudsen, editor, Fast Software Encryption, 6th international Workshop, Volume 1636 of Lecture Notes in Computer Science, pages 124138, Rome, Italy, Springer-Verlag 1999.

[8] E. Biham, A. Biryukov, A. Shamir, “Cryptanalysis of Skipjack Reduced to 31 Rounds using Impossible Differentials”, in International Conference on the Theory and Applications of Cryptographic Techniques, 1999, pp. 12-23.

[9] M. Minier and M. Naya-Plasencia, “A related key impossible differential attack against 22 rounds of the lightweight block cipher LBlock”, In Information Processing Letters, Volume 112, Issue 16, 2012, Pages 624-629, ISSN 0020-0190.

[10] J. Chen, Y. Wei, Y. Hu, “A New Method for Impossible Differential Cryptanalysis of 7-round Advanced Encryption Standard”, Proceedings of International Conference on Communications, Circuits and Systems Proceedings 2006, Vol. 3, pp. 1577-1579, IEEE, 2006.

[11] C. Boura, M. Naya-Plasencia, and V. Suder,“Scrutinizing and Improving Impossible Differential Attacks: Applications to CLEFIA, Camellia, LBlock and Simon”, In ASIACRYPT 2014, Lecture Notes in Computer Science , volume 8873, pages 179-199, Springer, 2014.

[12] B. Bahrak and M. R. Aref, “Impossible differential attack on seven-round AES-128”, IET Information Security journal, Vol. 2, Number 2, pp. 2832, IET, 2008.

[13] B. Bahrak and M. R. Aref, “A Novel Impossible Differential Cryptanalysis of AES”, proceedings of the Western European Workshop on Research in Cryptology 2007, Bochum, Germany, 2007.

[14] J. Lu, O. Dunkelman, N. Keller, and J. Kim, “New Impossible Differential Attacks on AES”, INDOCRYPT 2008. LNCS, vol. 5365, pp. 279293. Springer, Berlin, 2008.

[15] C. Dobraunig and E. List, “Impossible-Differential and Boomerang Cryptanalysis of Round-Reduced Kiasu-BC”, pp. 207222. Cham: Springer International Publishing, 2017.

Alireza Mehrdad is a master of science. He received his M.S. in secure communications and cryptography from Shahid Beheshti University, Tehran, Iran, in 2018. He had received his bachelor in communication from Noshirvani Institute of Technology, Babol, Iran, in 2015. His main research interests are symmetric cryptography, post-quantum cryptography, and authenticated encryption.

Farokhlagha Moazami is an assistant professor at the Cyber Space Research Institute at Shahid Beheshti University, Iran, Tehran, since 2013. She received B.S. and Ph.D. degrees in mathematics from Alzahra University, Tehran, Iran, in 2004 and 2012, respectively and M.S. degree in mathematics from Sharif University of Technology, Iran, Tehran, in 2006. She was a postdoctoral at Sharif University of Technology, Iran, Tehran, from 2012 to 2013. Her main research interests is theoretical and practical aspects of cryptography.

Hadi Soleimany is an assistant professor at Cyberspace Research Institute at Shahid Beheshti University, Iran, since 2015. He received his Ph.D. in theoretical computer science from Aalto University, Finland, in 2015. He was a postdoctoral researcher at Technical University of Denmark (DTU), Denmark, in summer 2016 and 2017. His main research interests are practical aspects of cryptography.