Related-Tweak Impossible Differential Cryptanalysis of Reduced-Round TweAES

Chao Niu[1,2], Muzhou Li[1,2], Meiqin Wang[1,2(✉)], Qingju Wang[3], and Siu-Ming Yiu[4]

[1] School of Cyber Science and Technology, Shandong University, Qingdao, Shandong, 266237, China. {niuchao, muzhouli}@mail.sdu.edu.cn, mqwang@sdu.edu.cn

[2] Key Laboratory of Cryptologic Technology and Information Security of Ministry of Education, Shandong University, Qingdao, Shandong, 266237, China

[3] SnT, University of Luxembourg, Esch-sur-Alzette L-4364, Luxembourg. qjuwang@gmail.com

[4] Department of Computer Science, the University of Hong Kong, Hong Kong, 999077, China. smyiu@cs.hku.hk

Abstract. We consider the related-tweak impossible differential cryptanalysis of TweAES. It is one of the underlying primitives of Authenticated Encryption with Associated Data (AEAD) scheme ESTATE which was accepted as one of second-round candidates in the NIST Lightweight Cryptography Standardization project. Firstly, we reveal several properties of TweAES, which show what kinds of distinguishers are more effective in recovering keys. With the help of automatic solver Simple Theorem Prover (STP), we achieve many 5.5-round related-tweak impossible differentials with fixed input differences and output differences that just have one active byte. Then, we implement 8-round key recovery attacks against TweAES based on one of these 5.5-round distinguishers. Moreover, another 5.5-round distinguisher that has four active bytes at the end is utilized to mount a 7-round key recovery attack against TweAES, which needs much lower attack complexities than the 6-round related-tweak impossible differential attack of TweAES in the design document. Our 8-round key recovery attack is the best one against TweAES in terms of the number of rounds and complexities so far.

Keywords: TweAES· Tweakable block ciphers · Related-tweak · Impossible differential cryptanalysis

1 Introduction

TweAES is one of the underlying primitives of Authenticated Encryption with Associated Data (AEAD) scheme ESTATE [7], which was accepted as one of second-round candidates in the NIST Lightweight Cryptography Standardization project. As a tweakable variant of AES-128 [11], TweAES is explicitly designed for efficient processing of small tweaks of 4 bits and the tweak is used to provide domain separation. More specifically, TweAES is identical to AES-128 except that it injects a tweak value at an interval of every two rounds.

After two decades of cryptanalysis, various attacks have been carried out on AES-128 [1,4,5,9,10,12,15,18,19]. Among all these attacks, the best attack against it in the single-key setting only reaches 7 out of 10 rounds, and the best-known attack so far is either an impossible differential attack [1,5,15], or a meet-in-the-middle attack [4,9,10]. Moreover, a known-key attack against full AES-128 is proposed in [12]. With the extra freedom brought by the 4-bit tweak, the security of TweAES compared to AES-128 is worthy of further study. In [7], designers of TweAES evaluated its security against differential, impossible differential and boomerang attacks in the chosen-tweak setting. Among all of them, the most effective attack is the impossible differential attack, where the data and time complexity are both 2^{127} while the memory complexity is 2^{96}.

Here, we give the first third-party cryptanalytic result on TweAES utilizing impossible differentials under related tweaks, which is also the best key recovery attack in terms of the number of rounds and complexities according to our knowledge.

Motivations and Contributions. A tweakable block cipher has the advantages of easier to prove models of operation based on it, and respond to the high demand, many tweakable block ciphers have been proposed [3,13,14]. As a tweakable block cipher, TweAES provides a more efficient way to process the demand of domain separation when encrypting a short message. Although the AEAD scheme ESTATE does not enter the finalist of the NIST competition, the security of the newly designed TweAES is still worth noticing.

In this paper, we firstly reveal several interesting properties of TweAES, which show that the related-tweak impossible differential attack on 8-round cipher proposed in the design document [7] is not valid actually. In their attack, same data sets are used under 2^7 different tweak differences trying to filter out wrong keys. However, as shown in Section 2.3, different tweak differences will lead to different involved key bits used before and after this distinguisher. With these facts, their data complexity is corrected as 2^{131}, and then the time complexity of their attack is at least 2^{131} 8-round TweAES encryptions. Thus, their attack is invalid due to the time complexity exceeding the exhaustively search.

To evaluate the security of TweAES against impossible differential, we use the automatic solver STP to search for more effective distinguishers aiming to achieve more rounds in key recovery attacks. As a result, we achieve many 5.5-round distinguishers with fixed input and tweak differences, and the output differences only have one active byte. Then, we implemented 8-round key recovery attack against TweAES based on one of these 5.5-round related-tweak impossible differential distinguishers shown in Section 4. Moreover, we use the 5.5-round impossible differential that has four active bytes at the end to mount a 7-round key recovery attack against TweAES in Section 5 with much lower attack complexities than the 6-round attack proposed in the design document [7].

Our attacks along with others on TweAES are shown in Table 1, from which one can see that our 8-round key recovery attack is the best one against TweAES in terms of the number of rounds and complexities so far.

Table 1. Summary of attacks against TweAES. The time complexity is measured by the unit of encryptions, and the memory complexity is measured in bits.

#Round	Data	Time (EN)	Memory (Bits)	#TK	Attack Type	Ref.
5	2^{5} CP	2^{26}	$2^{28.58}$	2	Trunc. Diff.	[7]
6	2^{5} KP	2^{45}	*negl.*	2^{4}	Integral	[7]
6	2^{119} CP	2^{119}	$2^{78.17}$	2	RTID	[7]
7	2^{99} **CP**	2^{100}	2^{70}	2	**RTID**	**Sect. 5**
8[†]	2^{131} CP	$>2^{131}$	$2^{112.58}$	2^{4}	RTID	[7]
8	$2^{124.28}$ **CP**	$2^{124.36}$	$2^{118.81}$	2	**RTID**	**Sect. 4**

CP: chosen plaintext; KP: known plaintext; #TK: the number of tweak used;
RTID: related-tweak impossible differential;
[†] After correction, the time complexity exceeds the exhaustive search, which makes it invalid.

Outline. In Section 2, we briefly recall the specification of TweAES and reveal several interesting properties of it. With these properties, the attack complexity of related-tweak impossible differential attack in [7] has been corrected. Then an automatic search algorithm with STP for finding a key recovery conducive impossible differential and a new 8-round key recovery attack with 5.5-round related-tweak impossible differential is proposed in Section 3 and Section 4, respectively. Moreover, in Section 5, we mount a key recovery attack against 7-round TweAES with much lower complexities than the 6-round attack in the design document. Finally Section 6 concludes this work.

2 Preliminaries

TweAES, one of the underlying primitives of the AEAD scheme ESTATE [7], is a tweakable variant of AES-128 [11]. In this section, we first recall the specification of TweAES. Then, we reveal several properties of it, which cause the 8-round attack in [7] to be illusive. Actual complexities of this attack are evaluated in the end of this section.

2.1 Specification of TweAES

TweAES is a 128-bit tweakable block cipher with a 4-bit tweak and a 128-bit key, which is a tweakable variant of AES-128. More specifically, TweAES is identical to AES-128 except that it injects a tweak value at an interval of every two rounds.

A 128-bit plaintext states are commonly treated as byte matrices of size 4×4, as shown in Figure 1.

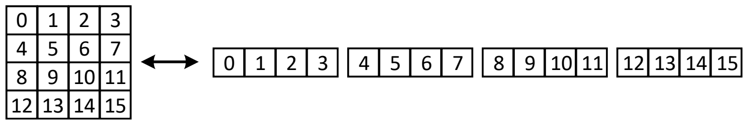

Fig. 1. 4x4 (left) and 1×16 (right) byte indexing of 128-bit data block of `TweAES`

The round function of `TweAES` is composed of five operations, which are implemented sequentially, and we detail them as follows:

- **SubByte** (SB)**:** `TweAES` uses the same 8-bit Sbox as `AES-128`. One can refer to [11] for more details.
- **ShiftRows** (SR)**:** The bytes in the i-th row are cyclically shifted by i place to the left.
- **MixColumn** (MC)**:** Multiply each column with a constant 4×4 matrix

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

  over the finite field $\mathbb{F}_8$, where the irreducible polynomial is $x^8+x^4+x^3+x+1$.
- **AddKey** (AK)**:** XOR the state with a 128-bit subkey, which is generated from the 128-bit master key according to the key schedule of `AES-128`.
- **AddTweak** (AT)**:** The 4-bit tweak is first expanded to an 8-bit value using a linear code, and then the 8-bit value is XORed to the state at an interval of every two rounds. To be specific, let us denote r as the r-th round with $1 \leq r \leq 10,\ r \in \mathbb{Z}$, then the AT operation is only applied when r is odd.

Note that, all the operations except for the AT, are identical to that of `AES-128` including the key schedule. The detail of AT is shown as follows. At first, the 4-bit tweak expanded to an 8-bit value using a linear code. Define (T_0, T_1, T_2, T_3) as the 4-bit tweak, and

$$T_{\oplus} = T_0 \oplus T_1 \oplus T_2 \oplus T_3.$$

Then for each $i \in \{0, 1, 2, 3\}$, we have

$$T_{i+4} \leftarrow T_i \oplus T_{\oplus}.$$

Afterwards, the 8-bit expanded tweak $(T_0, T_1, T_2, T_3, T_4, T_5, T_6, T_7)$ is XORed into the least significant bit of each byte in first two rows of the state. The values of expanded tweak are the same for each AT operation.

X_i	:	state before SB in round i
Y_i	:	state before SR in round i
Z_i	:	state before MC in round i
W_i	:	state before AK in round i
T	:	the four-bit tweak
T_i	:	the i-th bit of T
K_i	:	the subkey in round i
ET	:	the eight-bit expanded tweak
ΔX	:	the difference in a state X
$X_i[m]$	:	the m^{th} byte of a state X in round i, where $0 \leq m \leq 15$
$X_i[p, ..., r]$	:	bytes from p^{th} to r^{th} of state X in round i, where $0 \leq p,\ r \leq 15$

2.2 Notations and Definitions

The following notations are utilized throughout the rest of this paper.

2.3 Properties of `TweAES`

Some interesting properties of `TweAES` that were utilized during our attacks are described as follows:

1. The linear expand code of the 4-bit tweak make sure that the number of the active columns of the state caused by the tweak difference is at least three after the AT operation and one AES round.
2. After the inverse AT operation and one inverse AES round, the tweak difference propagates to the whole state when the active number of tweak bits is odd or 1111_2.
3. Recall that the `TweAES` has a 4-bit tweak, and this 4-bit tweak has 15 kinds of nonzero differences. With the insert of tweak difference and propagation of difference before and after the distinguisher, key bytes involved are different in the key recovery attack.

In general, multi distinguishers can be used to mount a key recovery attack only when the involved key bits in the process are the same. By exploiting the involved key bits when appending one round before and after the distinguisher, we can determine that one cannot use a different tweak difference in the key recovery because of the different involved key bits. Under the 15 different nonzero tweak differences, involved key bits are shown in Table 2. Here, we omit the MC operation in the appending round after the distinguisher.

We utilize the following proposition in our attacks, which is also exploited in [10].

Proposition 1. *(Differential Property of Sbox [10]). Given the nonzero input and output difference pair $(\Delta_{in}, \Delta_{out})$ of an Sbox S, there exists one solution y on average, for which the equation, $S(y) \oplus S(y \oplus \Delta_{in}) = \Delta_{out}$, holds true.*

Table 2. The involved key bits under different non-zero tweak differences, where `*` denotes the key bits involved, and `0` denotes the key bits not involved. The byte order of the involved key is shown in Figure 1.

Tweak Diff.	Expanded Tweak Diff.	Before	After
0001_2	$(0001\ 1110)_2$	`**** **** **** ****`	`000* **0* 0000 0000`
0010_2	$(0010\ 1101)_2$	`**** **** **** ****`	`00*0 *0** 0000 0000`
0100_2	$(0100\ 1011)_2$	`**** **** **** ****`	`0*00 0*** 0000 0000`
1000_2	$(1000\ 0111)_2$	`**** **** **** ****`	`*000 ***0 0000 0000`
1110_2	$(1110\ 0001)_2$	`**** **** **** ****`	`***0 00*0 0000 0000`
1101_2	$(1101\ 0010)_2$	`**** **** **** ****`	`**0* 0*00 0000 0000`
1011_2	$(1011\ 0100)_2$	`**** **** **** ****`	`*0** *000 0000 0000`
0111_2	$(0111\ 1000)_2$	`**** **** **** ****`	`0*** 000* 0000 0000`
0011_2	$(0011\ 0011)_2$	`00** *00* **00 0**0`	`00** 0**0 0000 0000`
0110_2	$(0110\ 0110)_2$	`0**0 00** *00* **00`	`0**0 **00 0000 0000`
1100_2	$(1100\ 1100)_2$	`**00 0**0 00** *00*`	`**00 *00* 0000 0000`
1001_2	$(1001\ 1001)_2$	`*00* **00 0**0 00**`	`*00* 00** 0000 0000`
0101_2	$(0101\ 0101)_2$	`0*0* *0*0 0*0* *0*0`	`0*0* *0*0 0000 0000`
1010_2	$(1010\ 1010)_2$	`*0*0 0*0* *0*0 0*0*`	`*0*0 0*0* 0000 0000`
1111_2	$(1111\ 1111)_2$	`**** **** **** ****`	`**** **** 0000 0000`

2.4 Impossible Differential Attack on `TweAES` Proposed by Designers

In [7], designers proposed a key recovery attack on 8-round `TweAES` utilizing a 6-round related-tweak impossible differential, which is depicted in Appendix A. By adding one round before and after it, they claimed that 8-round `TweAES` can be attacked (see Appendix B). However, there are two main points one should notice.

The first one comes from the properties described in Property 3. As we can see, different tweak differences will lead to different involved key bits. In their attack [7], tweak difference is set to be 1100_2. Thus, there are only $2^2 \times 2^2 \times \frac{1}{2} = 2^3$ tweak pairs satisfying this difference. However, they exploited 2^7 tweak pairs to filter out wrong keys which is not achievable. And then, the attack procedure of [7] in Step 3 will obtain about $2^{3+64+64-96} = 2^{35}$ pairs rather than 2^{39} pairs. Hence, in Step 5, for filtering the wrong key, they need repeat 2^{63} times from the first step to obtain $2^{35+63} = 2^{98}$ wrong-key candidates. The data complexity is $2^4 \times 2^{64} \times 2^{63} = 2^{131}$ chosen plaintexts. And then the time complexity of this attack is at least 2^{131} 8-round `TweAES` encryptions.

The second one is that to recover the master key, the attack needs much higher complexity than the exhaustive search. In the attack [7], after filtering the wrong key candidates, the remaining key space of the involved 12 bytes becomes $2^{96} \times (1 - 2^{-96})^{2^{98}} \approx 2^{96} \times e^{-2} \approx 2^{90.2}$. Due to the complicated key schedule of `TweAES`, the reduced factor of 5.77-bit subkey information cannot be used to recover 5.77-bit master key information. To recover the master key, we should exhaustively search the subkey candidates and the left eight bytes of K_1. Then, the time complexity for recovering the master key is $2^{90.2} \times 2^{64} \approx 2^{154.2}$

8-round encryption units, which costs much higher than the exhaustively search of 128-bit master key.

3 STP-Based Automatic Searching Algorithm for Related-Tweak Impossible Differential

Recently, many cryptanalytic results have been improved with the advent of various automated tools. Among all of them, the Boolean Satisfiability (SAT) [8] and Satisfiability Modulo Theories (SMT) problem [2] solver STP[5] has been playing an important role. The application of STP in cryptanalysis is firstly proposed by Mouha and Preneel [17]. It is a decision procedure to check if there is a solution to a set of equations. These equations must follow the rule of input language parsed by STP[6].

When searching for related-tweak impossible differentials, differences on the input state is often set to be canceled by tweak differences in the first round, as shown in [20], which can lead to longer distinguishers. Moreover, to make them effective when mounting key recovery attacks, the number of active bytes of input and output differences is usually restricted to be as small as possible. And position of active bytes shall be chosen carefully. Motivated by this kind of strategy, we try to find distinguishers that cancel the state difference with tweak at the beginning, but remain one active byte at the end.

With the help of automatic tools, we can accurately characterize the propagation of the difference. More specifically, in the search algorithm, differential propagation properties of operations should be represented by some equations and precisely depicted. In addition to these propagation properties, equations representing the condition for related-tweak impossible differential are also included. Whether these equations have a solution or not can directly help us to confirm whether the expected impossible differential exists.

In practice, if we aim at finding an r-round related-tweak impossible differential, we describe the difference propagation through the round function and tweak schedule (in the case of `TweAES`, the same tweak is used in each round). These constraint equations can be divided into two parts. Part 1 contains equations depicting propagation properties between input and output difference of operations in the round function and the tweak schedule. In Part 2, we describe equations representing the condition we used for related-tweak impossible differentials.

Part 1. Equations for Basic Operations in Block Cipher

In the first part of the model, we describe equations of XOR and branching operations for differential propagation [16]. Then, we give equations to describe

[5]http://stp.github.io/

[6]STP supports two kinds of input languages, here we use the CVC one. For more information about the CVC, please refer to https://stp.readthedocs.io/en/latest/cvc-input-language.html

the ideal Sbox and confusion layer. For clarity, differential propagation of XOR and branching operations are illustrated in Figure 2.

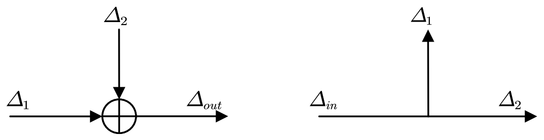

Fig. 2. XOR and branching

Property 1. **(XOR[16])** Let Δ_1 and Δ_2 represent two input differences for the operation XOR, and output difference is Δ_{out}. Then the relation between them is $\Delta_{out} = \Delta_1 \oplus \Delta_2$.

Property 2. **(Three-Branch[16])** Let Δ_{in} denote input difference, and output difference to be decided are Δ_1 and Δ_2. Then the relation between them are $\Delta_{in} = \Delta_1 = \Delta_2$.

In our search model, we explicitly described the linear layer in the bit level. For the Sbox operation, we assume that the AES Sbox can map a nonzero input difference to any nonzero output difference. Under this assumption, when we found an impossible differential, it will still hold for real ciphers since contradictions occur at the bit-level linear layer.

Property 3. **(Ideal Sbox)** Let S be the Sbox used in the round function of the target cipher. The input difference is Δ_{in}, and the corresponding output difference is denoted as Δ_{out}. Then we have $\Delta_{out} = 0$ if $\Delta_{in} = 0$. Otherwise, $\Delta_{out} \neq 0$.

The linear layer of many block ciphers can be represented as matrix multiplication. Then, we have the following property.

Property 4. **(Confusion Matrix)** Let A denote the confusion matrix, $\boldsymbol{\Delta}_{in}$ and $\boldsymbol{\Delta}_{out}$ represent the column-wise input and output difference, respectively. Then, we have $\boldsymbol{\Delta}_{out} = A \cdot \boldsymbol{\Delta}_{in}$.

Part 2. Equations Depicting the Related-Tweak Impossible Differential Condition

The original impossible differential cryptanalysis usually derives the differential propagation in the forward and backward directions, and finds the contradiction in the middle. Our search model is at the bit level, and we just need to add constraints at the beginning and the end of the model. Then, STP will detect the contradiction in the whole model. Note that, the active state or the active bytes of input and output state are heuristically determined. In our search strategy, we restrict the input difference at the beginning of the state equal to the tweak difference. At the end of the state, we remain only one active byte. In the actual search, we found that the less the number of non-zero differences in the output is limited, the longer distinguisher we can get.

Given all these properties, the searching algorithm for related-tweak impossible differential is listed in Algorithm 1.

Algorithm 1: Search RTID $(R, \varDelta_T, \varDelta_0, \varDelta_R)$

```
Input:  $R$ : Number of rounds covered by the expected distinguisher
    $\varDelta_T$ : Active state of tweak difference
    $\varDelta_0$ : Active state of input difference in the state
    $\varDelta_R$ : Active state of output difference in the state
Output:  $R$-round related-tweak impossible differential or "No solution"
1 forall considered difference on the 4-bit tweak do
2     forall considered position of active output difference byte do
          /* Equations in Part 1 describing state update */
3         for $r \leftarrow 0$ to $R-1$ do
4             Use Property 1 $\sim$ 4 to construct equations for the $r$-th round function;
          /* Equations in Part 2 describing the related-tweak impossible differential condition */
5         Construct equations describing the active state of the difference on tweak, input and output according to $\varDelta_T$, $\varDelta_0$ and $\varDelta_R$;
6         Input all these equations into STP and let it solve;
7         if STP return "Invalid" then
8             Output $(\varDelta_T, \varDelta_0, \varDelta_R)$ as a related-tweak impossible differential;
9 return "No Solution";
```

One thing we have to mention is that this algorithm can also be used to search for related-tweak impossible differentials under different strategies of choosing input and output differences, by simply modifying equations in Part 2.

4 Key Recovery Attack on 8-Round TweAES

After revealing that the related-tweak impossible differential distinguisher in [7] cannot perform valid key recovery attacks, we try to find a new related-tweak impossible differential distinguisher and perform a key recovery attack against TweAES. Before the attack is explained, we introduce some more notations which are borrowed from [6]. Suppose an impossible differential ($\varDelta X \nrightarrow \varDelta Y$) has been constructed for $r_\varDelta$ rounds under a pair of related tweaks, and is used to attack $r_{in} + r_\varDelta + r_{out}$ rounds. Through r_{in} and r_{out}, $\varDelta X$ and $\varDelta Y$ propagate to $\varDelta_{in}$ and $\varDelta_{out}$ with probability one, respectively. Let c_{in} (resp. c_{out}) denote the number of bit-conditions that have to be verified to obtain ΔX from Δ_{in} (resp. ΔY from Δ_{out}).

4.1 The 5.5-Round Related-Tweak Impossible Differential Distinguisher of TweAES

With the help of Algorithm 1, we can find lots of 5.5-round impossible differential distinguishers which have only one active byte at the end. These distinguishers are shown in Table 3. To clarify, we visualize the contradiction leading to the

Table 3. Tweak differences and corresponding possible active byte of the output difference. The active byte index at the end is shown as a list $[0, ..., 15]$.

Dist.	Tweak Diff.	Expanded Tweak Diff.	Active Byte Index
D_1	0001_2	$(0001\ 1110)_2$	$[0, 1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_2	0010_2	$(0010\ 1101)_2$	$[0, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_3	0100_2	$(0100\ 1011)_2$	$[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_4	1000_2	$(1000\ 0111)_2$	$[0, 1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_5	1110_2	$(1110\ 0001)_2$	$[0, 1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_6	1101_2	$(1101\ 0010)_2$	$[0, 1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_7	1011_2	$(1011\ 0100)_2$	$[0, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_8	0111_2	$(0111\ 1000)_2$	$[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_9	0011_2	$(0011\ 0011)_2$	$[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_{10}	0110_2	$(0110\ 0110)_2$	$[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_{11}	1100_2	$(1100\ 1100)_2$	$[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_{12}	1001_2	$(1001\ 1001)_2$	$[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_{13}	0101_2	$(0101\ 0101)_2$	$[4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_{14}	1010_2	$(1010\ 1010)_2$	$[4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$
D_{15}	1111_2	$(1111\ 1111)_2$	$[4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$

impossible differential property in Figure 3 for D_{11} with the 12-th byte of the output difference. To illustrate the contradiction that leads to the impossible differential, we depict the propagation of the difference through the encryption data path. First, we derive the difference propagation in the forward direction. Recall the AT operation in the specification of `TweAES`, the 4-bit tweak is firstly expanded to an 8-bit value using a linear code, and the 8-bit value is XORed to the least significant bit of each byte in the first two rows of state. For canceling the difference on the state with a tweak difference, we set the difference on the $W_2[0, 1, 4, 5]$ equal to the active one-bit tweak difference, which means the least significant bit of each byte has an active difference, along with other bits a zero difference. Hence, the difference on each byte of $W_2[0, 1, 4, 5]$ is fixed `0x01`. Since the tweak of `TweAES` is XORed every two rounds, after the difference canceling occurs at the beginning of the distinguisher, the zero difference will propagate through two rounds to W_4.

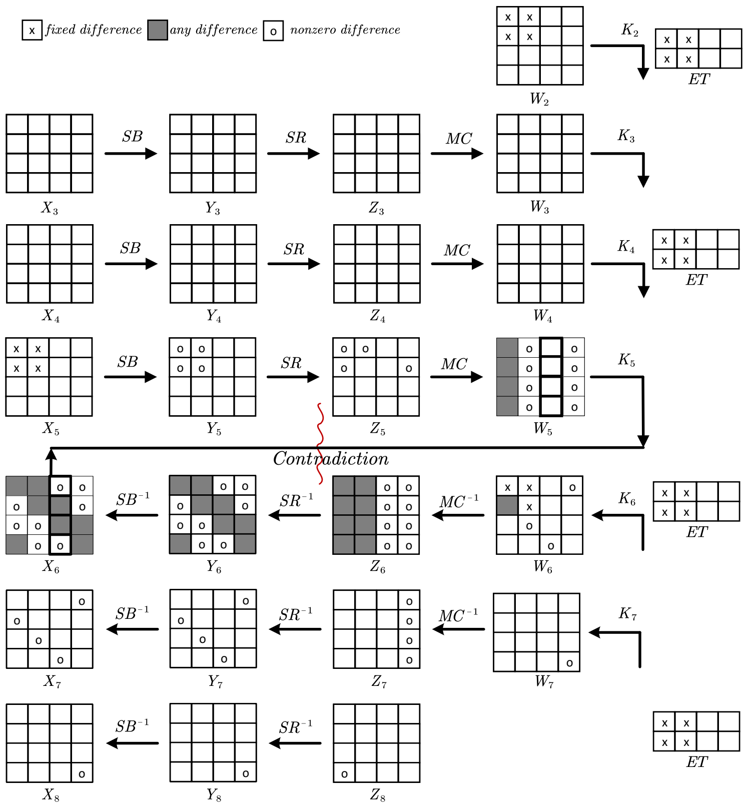

Fig. 3. 5.5-round distinguisher of `TweAES` with tweak difference 1100_2

After the AT operation, the tweak difference inserts into the state. Hence, the difference of $X_5[0,1,4,5]$ has an active difference on the least significant bit of each byte with other bits a zero difference. Then, after SB operation, SR operation, and MC operation, we get the active status of the state W_5 where the third column of it is an inactive difference.

Next, we will derive the difference propagation in the back direction. At the end of the distinguisher, we set $Z_8[12]$ to be an active byte. After inverse SR operation and inverse SB operation, the difference propagates to $W_7[15]$. Then, the active byte is propagating to the state $X_7[3,4,9,14]$ after one inverse AES round. After the AT operation, the difference propagates to $W_6[0,1,3,4,5,9,14]$. Then, after one inverse AES round, the third column of state X_6 has at least two active bytes.

Due to the inactiveness of the second column of W_5, the contradiction occurs in the third column of state W_5 and state X_6.

As shown in Table 3, the end of the distinguisher D_{11} can be any byte of Z_8. Hence, the active byte of W_7 can be anyone of the four bytes of the fourth column. Moreover, we can easily derive that the active column of state Z_7 can also be anyone of the four-column.

Remarks.

1. The beginning of distinguisher D_{11} shows that the possible difference has only one value that equals the expanded tweak difference.
2. The distinguisher D_{11} shows that, in key recovery attacks, anyone of the four bytes in the first column of ΔZ_8 can be used to recover the same key bytes, which means there are $(2^8 - 1) \times \binom{4}{1} \approx 2^{10}$ possible values.

4.2 The Key Recovery Attack on 8-Round `TweAES`

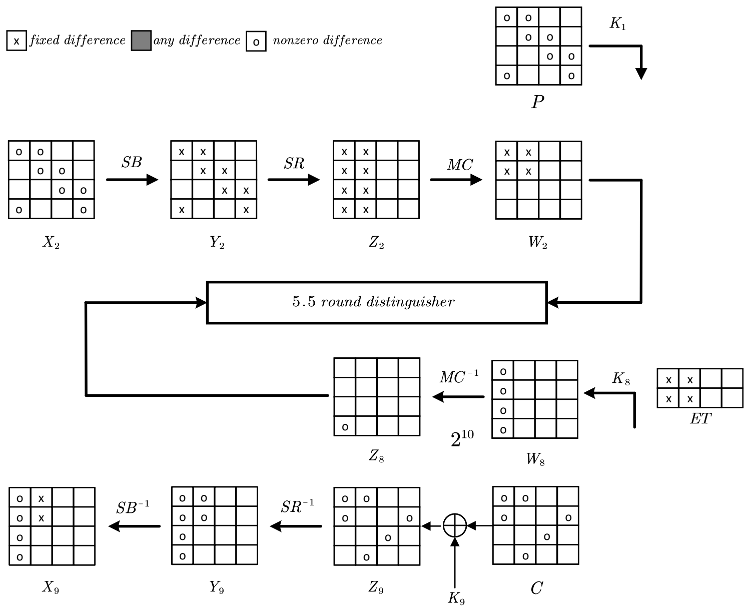

Fig. 4. Key recovery attack on 8-round `TweAES`

By appending one round on the top and another one round at the bottom of the distinguisher D_{11} where one of the four bytes in the first column of Z_8 is active, as illustrated in Figure 3, we mount a key recovery attack on 8-round `TweAES`. As usual, we omit the last round MC operation in the reduced-round version of `TweAES`. Before the attack is explained, we instantiate the notation mentioned before for better comprehension. Here, r_Δ covers 5.5-round, r_{in} and r_{out} are both one round, $c_{in} = |\Delta_{in}| = 64$, $c_{out} = 8 \times (6-1) - 2 = 38$, $|\Delta_{out}| = 8 \times 6 = 48$ and $|k_{in} \cup k_{out}| = 8 \times (8+6) = 112$. In our attack, the number of keys that can be eliminated by one qualified plaintext pair is denoted as 2^{elim}. Then, we have the sieved wrong key with probability $2^{elim}/2^{|k_{in} \cup k_{out}|} = 1/2^{c_{in}+c_{out}} = 2^{-102}$.

The attack procedure is briefly described in Algorithm 2 and illustrated in Figure 4. Detailed attack procedure is shown as follows.

Data Collection. Consider a pair of structures S_1 and S_2, where, each structure consists of $2^{|\Delta_{in}|} = 2^{64}$ plaintexts, and for each plaintext pair $P_1 \in S_1$ and $P_2 \in S_2$, $P_1 \oplus P_2 = (\, * * 0\, 0 \mid 0 * * 0 \mid 0\, 0 * * \mid * 0\, 0 * \,)$, where $*$ denotes any byte value. The total number of possible plaintext pairs is $2^{2\times|\Delta_{in}|} = 2^{128}$. Choose T and $T^{'}$, where $T \oplus T^{'} = 1100_2$. Encrypt the pool S_1 under T and the pool S_2 under $T^{'}$ to obtain the corresponding ciphertexts. For each ciphertext pair, check whether $n - |\Delta_{out}| = 80$ bits, i.e., $\Delta C[2,3,5,6,8,9,11,12,14,15] = 0$ or not, and discard it if false. Generate 2^N such pair of structures and repeat this for each pair of structures. In total, we will get about $M = 2^{N+|\Delta_{in}|\times 2-n+|\Delta_{out}|} = 2^{N+64\times 2-(128-8\times 6)} = 2^{N+48}$ pairs. This step requires a total of $2^{N+|\Delta_{in}|+1} = 2^{N+65}$ encryptions.

Key Recovery. For each one of these M pairs, do the following steps:

1. As mentioned by the Note in the end of Section 4.1, there is only one possible value of $\Delta W_2[0,1,4,5]$ where differences on these four bytes are all `0x01`. By inverse MC and inverse SR operation, we can deduce the difference of $\Delta Y_2[0,1,5,6,10,11,12,15]$. Note that, the value of ΔY_2 is also fixed.
Considering that we can get $\Delta X_2[0,1,5,6,10,11,12,15]$ from ΔP. Then, by using Proposition 1, we can deduce the value of $X_2[0,1,5,6,10,11,12,15]$.
So we can get one possible value of $K_1[0,1,5,6,10,11,12,15]$ as $K_1 = P \oplus X_2$.
This step has a time complexity of $M \cdot 1$ one-round encryptions.
2. As mentioned by the Note in Section 4.1, there are 2^{10} possible values of $\Delta W_8[0,4,8,12]$. For each possible value of $\Delta W_8[0,4,8,12]$, by a AT operation, we can deduce the difference $\Delta X_9[0,1,4,5,8,12]$. Considering that we can get $\Delta Y_9[0,1,4,5,8,12]$ from ΔC, by using Proposition 1, we can deduce the value of $Y_9[0,1,4,5,8,12]$. So we can get 10-bit information of $K_9[0,1,4,5,8,12]$ as $K_9 = SR(Y_9) \oplus C$. Then 10-bit information of K_9 is obtained. These obtained keys are wrong ones since they fulfill the impossible differential distinguisher. Hence, one pair of M can eliminate $2^{elim} = 2^{10}$ keys. This step has a time complexity of $M \cdot 2^{elim}$ one-round encryptions.
3. We can use the above steps to filter out the wrong subkey values.

The time complexity of analyzing M pairs is $M \cdot 2^{elim}$, and the total number of subkey left is:

$$K_{rem} = 2^{|k_{in}\cup k_{out}|} \times (1 - 2^{elim}/2^{|k_{in}\cup k_{out}|})^M = 2^{112} \times (1 - 2^{-102})^{2^{N+48}} \quad (1)$$

Suppose $(1 - 2^{elim-|k_{in}\cup k_{out}|})^M = 2^{-g}$, where $1 < g \le |k_{in} \cup k_{out}|$. It means that g-bit key information is recovered, then we have $M = 2^{|k_{in}\cup k_{out}|-elim} g \ln 2$

since $(1-2^{elim-|k_{in}\cup k_{out}|})^M \approx e^{-M2^{|k_{in}\cup k_{out}|-elim}}$. Moreover we know $M = 2^{N+|\Delta_{in}|\times 2-n+|\Delta_{out}|}$, thus $2^N = 2^{|k_{in}\cup k_{out}|-elim-|\Delta_{in}|\times 2+n-|\Delta_{out}|}\times g\ln 2$. Finally the data complexity is $D = 2^{N+|\Delta_{in}|+1} = 2^{|k_{in}\cup k_{out}|-elim+n+1-|\Delta_{in}|-|\Delta_{out}|}g\ln 2$.

Brute Force. For the subkey candidates that remain, we guess the left key bytes of $K_1[2,3,4,7,8,9,13,14]$ (8 bytes), and exhaustively search the $K_{rem}\times 2^{8\times 8} = 2^{112-g+64} = 2^{176-g}$ keys. For every guessed K_1, deduce the master key using key schedule, and verify this master key by one pair of plaintext and ciphertext.

Complexity Computation. The attack described above requires a data complexity of

$$D = 2^{|k_{in}\cup k_{out}|-elim+n+1-|\Delta_{in}|-|\Delta_{out}|}g\ln 2 = 2^{-elim+n+1}g\ln 2 = 2^{119}g\ln 2$$

chosen plaintexts. The total time complexity is the summation of the time consumption of all the steps:

$$T = D + M\cdot 2^{elim} + K_{rem}\times 2^{64}.$$

The memory complexity is the storage for one structure and wrong keys. We set $g=56$, then $D = 2^{124.28}$, $M\cdot 2^{10} = 2^{117.28}$, $K_{rem}\times 2^{64} = 2^{120}$, $T = 2^{124.36}$ 8-round encryption, and the memory complexity is $2^{112}\times 112 \approx 2^{118.81}$ bits.

5 The Key Recovery Attack on 7-Round `TweAES`

In this section, we evaluate the security of 7-round `TweAES` under related-tweak impossible differential attacks. Moreover, this 7-round attack needs much lower attack complexity than 6-round attack [7] of `TweAES` in design document. This shows that, the security of the cipher needs to be further studied. Our result shows that the security strength of `AES-128` will decrease after adding 4-bit tweak in this cipher.

5.1 The 5.5-Round Impossible Differential with Tweak Difference 1001_2

Applying the automatic search algorithm introduced in Section 3, we can find a 5.5-round related-tweak impossible distinguisher of `TweAES` which is shown in Figure 5. We visualize the contradiction leading to the impossible differential property.

As before, we depict the propagation of the differential with the tweak difference 1001_2. First, we derive the difference propagation in the forward direction. For canceling the difference on the state with a tweak difference, we set the difference on the $W_2[0,3,4,7]$ equal to the active one-bit tweak difference, which means the least significant bit of each byte has an active difference, along with other bits a zero difference. Hence, the difference on each byte of $W_2[0,3,4,7]$

Algorithm 2: Key Recovery Attack on 8-Round TweAES

```
   /* Data collection                                                    */
1  for $2^N$ pairs of structures do
2      Choose $(T, T^{'})$, where $T \oplus T^{'} = 1100_2$;
3      Choose $P_1 \in S_1$, $P_2 \in S_2$, $P_1 \oplus P_2 = (* * 0 0 | 0 * * 0 | 0 0 * * | * 0 0 *)$;
4      forall $2^{|\Delta_{in}|}$ plaintext $P_1$ in $S_1$ do
5          $C \leftarrow$ Encrypt $P_1$ under $(T, K)$;
6      forall $2^{|\Delta_{in}|}$ plaintext $P_2$ in $S_2$ do
7          $C^{'} \leftarrow$ Encrypt $P_2$ under $(T^{'}, K)$;
8      forall $2^{2|\Delta_{in}|}$ pairs do
9          $\Delta C \leftarrow C \oplus C^{'}$;
10         if $\Delta C = (* * 0 0 | * 0 0 * | 0 0 * 0 | 0 * 0 0)$ then
11             Remain the pair of $(P_1, T, C)$ and $(P_2, T^{'}, C^{'})$;
               /* Seive $2^{N+48}$ remaining pairs finally                  */
   /* Key recovery                                                       */
12 forall $2^{N+48}$ remaining pairs do
13     $\Delta X_2 = \Delta P$, $\Delta Y_2 = SR^{-1} \circ MC^{-1}(\Delta W_2)$;
14     Using Prop.1, $X_2 \leftarrow (\Delta X_2, \Delta Y_2)$;
15     $K_1 \leftarrow (X_2 \oplus P \oplus ET)$;
16     forall $2^{10}$ possible $\Delta W_8$ do
17         $\Delta X_9 = \Delta W_8 \oplus \Delta ET$, $\Delta Y_9 = SR^{-1}(\Delta C)$;
18         Using Prop.1, $Y_9 \leftarrow (\Delta X_9, \Delta Y_9)$;
19         $K_9 \leftarrow (SR(Y_9) \oplus C)$;
   /* Exhaustively search the candidate and the remaining key bits       */
20 for $2^{64}$ left key bits of $K_1$ do
21     forall remaining key candidates $K_{rem}$ do
22         Compute the master key $MK$ from $K_1$ and $K_{rem}$ using key schedule;
23         Get a random new pair of $(P, C)$;
24         $C^{'} \leftarrow$ encrypt $P$ under $MK$;
25         if $C^{'} = C$ then
26             Output the $MK$ as the right key.
```

is fixed as 0x01. Since the tweak of TweAES is XORed every two rounds, after the difference canceling occurs at the beginning of the distinguisher, the zero difference will propagate through two rounds to W_4.

After the AT operation, the tweak difference inserts to the state. Hence, the difference of $X_5[0, 3, 4, 7]$ has an active difference on the least significant bit of each byte with other bits a zero difference. Then, after SB operation, SR operation, and MC operation, we get the active status of W_5 where the second column of it is an inactive difference. Next, after the SB operation and SR

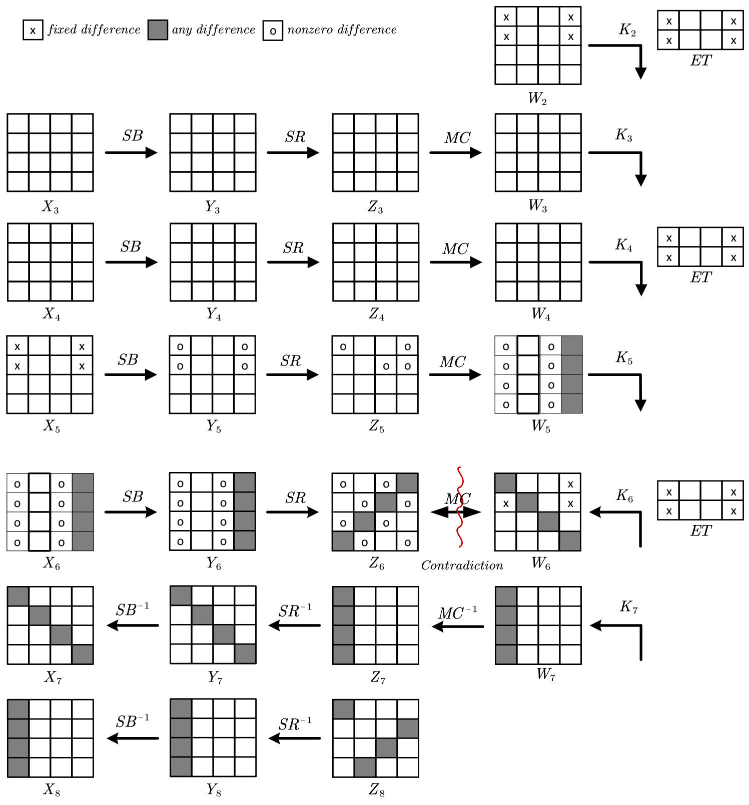

Fig. 5. 5.5-round distinguisher with tweak difference 1001_2

operation we get that the difference on the third column of Z_6 have at most three nonzero bytes.

Then, we will derive the difference propagation in the back direction. At the end of distinguisher, we set the difference on $Z_8[0, 7, 10, 13]$ to be any value. After the inverse SR operation and inverse SB operation, the difference propagates to $W_7[0, 4, 8, 12]$. Then, the difference propagates to the state $X_7[0, 5, 10, 15]$ after one inverse AES round. After the AT operation, the difference propagates to $W_6[0, 3, 4, 5, 7, 10, 15]$. The difference on third column of W_6 has at most one nonzero byte. Since the AES operation Mixcolumns has the branch number as 5, the contradiction occurs on the third column of Z_6 and W_6.

Remarks. The active column of W_7 can be one of the four-column. It's easy to verify that there are still contradictions existing, but the indices of active bytes in Z_8 change.

5.2 The Key Recovery Attack on 7-Round `TweAES`

By appending one round on the top of the distinguisher described in Section 5.1, we mount a key recovery attack on 7-round `TweAES`. Here, r_Δ covers 5.5-round, r_{in} is one round while $r_{out} = 0$, $c_{in} = |\Delta_{in}| = 64$, $c_{out} = 0$, $|\Delta_{out}| = 0$ and $|k_{in} \cup k_{out}| = 8 \times (8 + 0) = 64$.

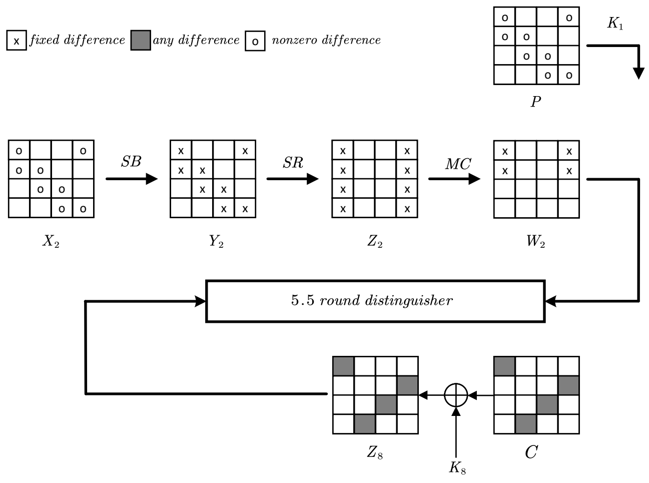

Fig. 6. 7-round key recovery attack of `TweAES`

The attack process is briefly described in Algorithm 3 and illustrated in Figure 5. Detailed attack procedures are as follows.

1. Construct 2^N structure such that each structure is made up of 2^{64} plaintexts. In each structure, we set $\Delta P[0,3,4,5,9,10,14,15]$ the eight active bytes.
2. Choose a pair of $(T, T^{'})$ such that the tweak difference is fixed 1001_2. Encrypt the plaintexts under two tweaks and only choose the ciphertexts pairs satisfying $\Delta Z_8[1,2,3,4,5,6,8,9,11,12,14,15] = 0$ which is one of the 4 cases mentioned in Section 5.1.
   In total, we will get about $2^{N+64\times 2-8\times 12} \times \binom{4}{1} = 2^{N+34}$ pairs.

   For each of the remaining pairs, do the following steps:
3. As mentioned before, $\Delta W_2[0,3,4,7]$ is fixed `0x01` on each byte, after the inverse MC and SR operation, we can deduce $\Delta Y_2[0,3,4,5,9,10,14,15]$. Note the value of ΔY_2 is also fixed.
   Considering that we can get $\Delta X_2[0,3,4,5,9,10,14,15]$ from ΔP. Then, by using Proposition 1, we can deduce the value of $X_2[0,3,4,5,9,10,14,15]$.
   So we can get one possible value of $K_1[0,3,4,5,9,10,14,15]$ as $K_1 = P \oplus X_2$.
4. We can use the above steps to filter out the wrong key values and then exhaustively search the left key bits.

5. For the subkey candidates that remain, guess the left key bytes of K_1 (8 bytes), and exhaustively search the $K_{rem} \times 2^{64}$ keys. For every guessed K_1, deduce the master key using key schedule, and verify this master key by one plaintext and ciphertext pair.

Complexity Computation. In total, the number of deduced key bytes is 8, i.e., 64 bits information of the key. In this case, each pair can eliminate only one value of the 64-bit guessed key information. Then, we choose $N = 35$, and the remaining candidate key is $K_{rem} = 2^{64} \times (1 - 1/2^{64})^{2^{N+34}} \approx 2^{17.8}$.

The memory for storing the key bits is $2^{64}64 \approx 2^{70}$ bit. The data complexity is $2^{35+64} = 2^{99}$ plaintexts under one pair of tweaks, which is 2^{100} chosen plaintexts.

The time complexity of Step 2 for encrypting the plaintexts is $2 \times 2^{35+64} = 2^{100}$. In Step 3, the total number of guesses is $2^{N+34} = 2^{69}$, which is equivalent to $2^{69} \times 8/16 \times 1/7 \times 2 \approx 2^{66.2}$ 7-round encryption. The time complexity of exhaustively search left key candidate and the left key bits of K_1 is $K_{rem} \times 2^{64} = 2^{17.8} \times 2^{64} = 2^{81.8}$. Thus the time complexity is approximately 2^{100} 7-round encryptions.

6 Conclusions

In this paper, we firstly reveal several properties of `TweAES`, which show what kinds of distinguishers are more effective in key recovery phase. Then, we use the automatic solver STP to search more effective related-tweak impossible differentials. As a result, we achieved many 5.5-round distinguishers with fixed input differences and output differences that only have one active byte. Then, based on one of these 5.5-round distinguishers, we implemented 8-round key recovery attack against `TweAES`. Moreover, another 5.5-round distinguisher that has four active bytes at the end is utilized to mount a 7-round key recovery attack against `TweAES`, which needs much lower attack complexities than 6-round attack [7] of `TweAES` in the design document. Our 8-round key recovery attack is the best one against `TweAES` in terms of the number of rounds and complexities so far.

As a tweakable variant of `AES-128`, `TweAES` is identical to `AES-128` except for the addition of the 4-bit tweak. Compared to `AES-128`, the security of `TweAES` against impossible differential has dropped due to the extra freedom brought from the 4-bit tweak discussed in our work. Moreover, the best attack against `AES-128` is meet-in-the-middle attack or impossible differential. With the addition of the 4-bit tweak, the security of `TweAES` against the meet-in-the-middle attack deserves further evaluation.

Acknowledgements

We thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper. This work is supported by the National Natural Science Foundation of China (Grant No. 62002201, Grant No. 62032014), the National Key Research and Development Program of China (Grant No. 2018YFA0704702), the Major Scientific and Technological Innovation Project of Shandong Province, China (Grant No. 2019JZZY010133), the Major Basic Research Project of Natural Science Foundation of Shandong Province, China (Grant No. ZR202010220025). Qingju Wang is funded by Huawei Technologies Co., Ltd (Agreement No.: YBN2020035184).

References

1. Bahrak, B., Aref, M.R.: Impossible differential attack on seven-round AES-128. IET Inf. Secur. **2**(2), 28–32 (2008)
2. Barrett, C.W., Sebastiani, R., Seshia, S.A., Tinelli, C.: Satisfiability modulo theories. In: Biere, A., Heule, M., van Maaren, H., Walsh, T. (eds.) Handbook of Satisfiability, Frontiers in Artificial Intelligence and Applications, vol. 185, pp. 825–885. IOS Press (2009)
3. Beierle, C., Jean, J., Kölbl, S., Leander, G., Moradi, A., Peyrin, T., Sasaki, Y., Sasdrich, P., Sim, S.M.: The SKINNY family of block ciphers and its low-latency variant MANTIS. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology - CRYPTO 2016, Part II. LNCS, vol. 9815, pp. 123–153. Springer (2016)
4. Bonnetain, X., Naya-Plasencia, M., Schrottenloher, A.: Quantum security analysis of AES. IACR Trans. Symmetric Cryptol. **2019**(2), 55–93 (2019)
5. Boura, C., Lallemand, V., Naya-Plasencia, M., Suder, V.: Making the impossible possible. J. Cryptol. **31**(1), 101–133 (2018)
6. Boura, C., Naya-Plasencia, M., Suder, V.: Scrutinizing and improving impossible differential attacks: Applications to clefia, camellia, lblock and simon. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology - ASIACRYPT 2014, Part I. LNCS, vol. 8873, pp. 179–199. Springer (2014)
7. Chakraborti, A., Datta, N., Jha, A., Mancillas-López, C., Nandi, M., Sasaki, Y.: ESTATE: A lightweight and low energy authenticated encryption mode. IACR Trans. Symmetric Cryptol. **2020**(S1), 350–389 (2020)
8. Cook, S.A.: The complexity of theorem-proving procedures. In: Harrison, M.A., Banerji, R.B., Ullman, J.D. (eds.) Proceedings of the 3rd Annual ACM Symposium on Theory of Computing, May 3-5, 1971, Shaker Heights, Ohio, USA. pp. 151–158. ACM (1971)
9. Demirci, H., Selçuk, A.A.: A meet-in-the-middle attack on 8-round AES. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 116–126. Springer (2008)
10. Derbez, P., Fouque, P., 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. LNCS, vol. 7881, pp. 371–387. Springer (2013)
11. Dworkin, M., Barker, E., Nechvatal, J., Foti, J., Bassham, L., Roback, E., Dray, J.: Advanced encryption standard (aes) (2001-11-26 2001). https://doi.org/https://doi.org/10.6028/NIST.FIPS.197
12. Gilbert, H.: A simplified representation of AES. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology - ASIACRYPT 2014, Part I. LNCS, vol. 8873, pp. 200–222. Springer (2014)
13. Jean, J., Nikolic, I., Peyrin, T.: Tweaks and keys for block ciphers: The TWEAKEY framework. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology - ASIACRYPT 2014, Part II. LNCS, vol. 8874, pp. 274–288. Springer (2014)

14. Jérémy Jean, I.N., Peyrin., T.: Deoxys v1.41. Submission to CAESAR, (2016), https://competitions.cr.yp.to/round3/deoxysv141.pdf
15. Leurent, G., Pernot, C.: New representations of the AES key schedule. In: Canteaut, A., Standaert, F. (eds.) Advances in Cryptology - EUROCRYPT 2021, Part I. LNCS, vol. 12696, pp. 54–84. Springer (2021)
16. Liu, Y., Liang, H., Li, M., Huang, L., Hu, K., Yang, C., Wang, M.: STP models of optimal differential and linear trail for s-box based ciphers. IACR Cryptol. ePrint Arch. **2019**, 25 (2019)
17. Mouha, N., Preneel, B.: Towards finding optimal differential characteristics for arx: Application to salsa20. Cryptology ePrint Archive, Report 2013/328 (2013), https://eprint.iacr.org/2013/328
18. Sun, S., Gérault, D., Lafourcade, P., Yang, Q., Todo, Y., Qiao, K., Hu, L.: Analysis of aes, skinny, and others with constraint programming. IACR Trans. Symmetric Cryptol. **2017**(1), 281–306 (2017)
19. Tiessen, T.: Polytopic cryptanalysis. In: Fischlin, M., Coron, J. (eds.) Advances in Cryptology - EUROCRYPT 2016, Part I. LNCS, vol. 9665, pp. 214–239. Springer (2016)
20. Zong, R., Dong, X.: Milp-aided related-tweak/key impossible differential attack and its applications to qarma, joltik-bc. IEEE Access **7**, 153683–153693 (2019)

A Related-Tweak Impossible Differential Distinguisher of TweAES in Design Document [7]

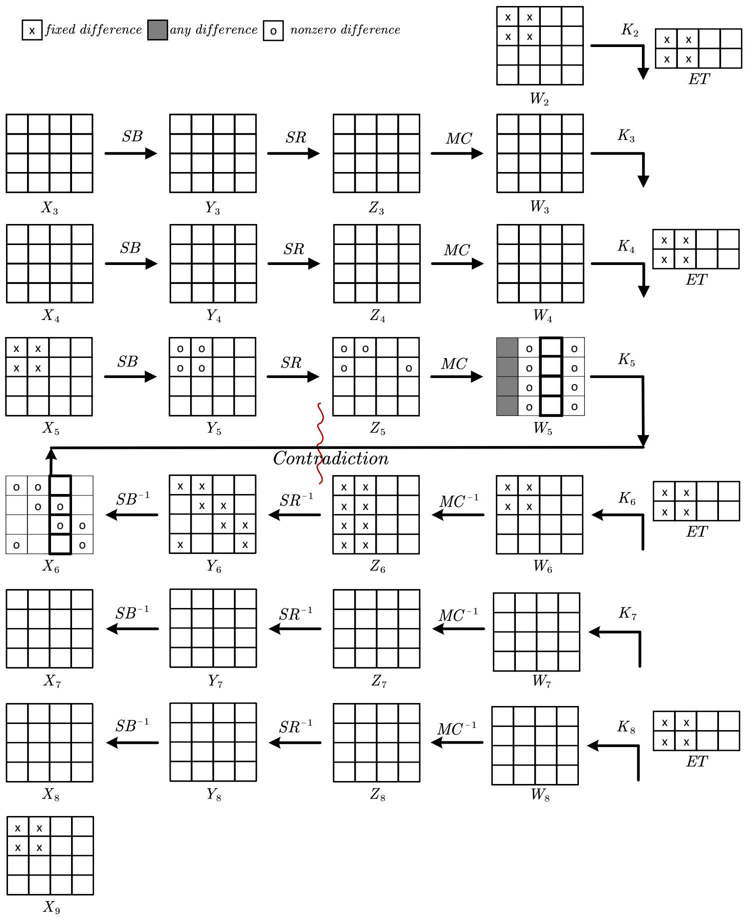

Fig. 7. 6-round related-tweak impossible distinguisher of TweAES [7]

B 8-Round Key Recovery Attack on TweAES in Design Document [7]

By appending one round at the beginning and the end of the distinguisher, they can perform an 8-round key recovery attack. The attack differential is shown in Figure 8. The key recovery attack procedure is as follows.

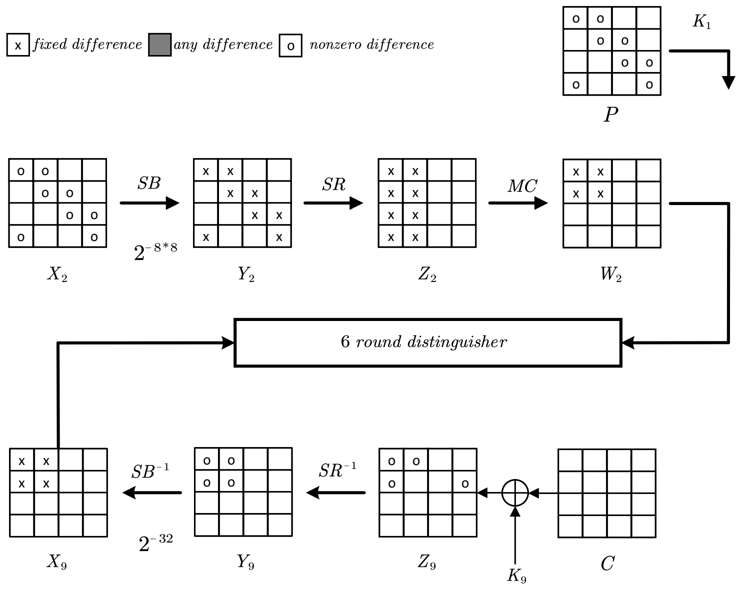

Fig. 8. Key recovery attack against 8-round `TweAES` in [7]

1. Choose all tweak values denoted by T^i where $i = 0, 1, ..., 2^4 - 1$.
2. For each of T^i, fix the value of inactive 8 bytes at the input, choose all 8-byte values at the active byte positions of the input state. Query those 2^{64} values to get the corresponding outputs. Those outputs are stored in the list L^i where $i = 0, 1, ..., 2^4 - 1$.
3. For all $\binom{2^4}{2} \approx 2^7$ pairs of L^i and L^j with $i \neq j$, find the pairs that do not have difference in 12 inactive bytes of the output state. About $2^{7+64+64-96} = 2^{39}$ pairs will be obtained.
4. For each of the obtained pairs, the tweak difference is fixed and the differences at the input and output states are also fixed. Those fix both of input and output differences of each Sbox in the first round and the last round. Hence, each pair suggests a wrong key.
5. Repeat the procedure 2^{59} times from the first step by changing the inactive byte values at the input. After this step, $2^{39+59} = 2^{98}$ wrong-key candidates (including overlaps) will be obtained. The remaining key space of the involved 12 bytes becomes $2^{96} \times (1 - 2^{-96})^{2^{98}} \approx 2^{96} \times e^{-2} \approx 2^{90.2}$. Hence, the key space for the 8 bytes of K_1 and 4 bytes of K_9 will be reduced by a factor of $2^{5.77}$.

The data complexity is $2^4 \times 2^{64} \times 2^{59} = 2^{127}$. The time complexity is also 2^{127} memory accesses. The memory complexity is to record the wrong keys of the 12 bytes, which is 2^{96}.

C The Algorithm for The Key Recovery Attack on 7-Round TweAES

Algorithm 3: Key Recovery Attack on 7-Round TweAES

```
   /* Data collection                                                    */
1  for $2^N$ pairs of structures do
2      Choose $(T, T^{'})$, where $T \oplus T^{'} = 1001_2$;
3      Choose $P_1 \in S_1$, $P_2 \in S_2$, $P_1 \oplus P_2 = (* * 0 0 | 0 * * 0 | 0 0 * * | * 0 0 *)$;
4      forall $2^{|\Delta_{in}|}$ plaintext $P_1$ in $S_1$ do
5          $C \leftarrow$ Encrypt $P_1$ under $(T, K)$;
6      forall $2^{|\Delta_{in}|}$ plaintext $P_2$ in $S_2$ do
7          $C^{'} \leftarrow$ Encrypt $P_2$ under $(T^{'}, K)$;
8      forall $2^{2|\Delta_{in}|}$ pairs do
9          $\Delta C \leftarrow C \oplus C^{'}$;
10         if $\Delta C = (* * 0 0 | * 0 0 * | 0 0 0 0 | 0 0 0 0)$ then
11             Remain the pair of $(P_1, T, C)$ and $(P_2, T^{'}, C^{'})$;
               /* Seive $2^{N+32}$ remaining pairs finally;
                                                                         */
   /* Key recovery                                                       */
12 forall $2^{N+32}$ remaining pairs do
13     $\Delta X_2 = \Delta P$, $\Delta Y_2 = SR^{-1} \circ MC^{-1}(\Delta W_2)$;
14     Using Prop.1, $X_2 \leftarrow (\Delta X_2, \Delta Y_2)$;
15     $K_1 \leftarrow (X_2 \oplus P \oplus ET)$;
   /* Exhaustively search the candidate and the remaining key bits       */
16 for $2^{64}$ left key bits of $K_1$ do
17     for Remaining key candidates $K_{rem}$ do
18         Deduce the master key using key schedule;
19         Verify this master key by one plaintext and ciphertext pair.
```