MILP-Aided Related-Tweak/Key Impossible Differential Attack and Its applications to QARMA, Joltik-BC

Rui Zong[1] and Xiaoyang Dong[2]

[1] Tsinghua University, zongrui3@163.com
[2] Tsinghua University

Abstract. In this paper, we study the relation of single-key impossible differentials with the related-tweakey/key ones and propose an interesting algorithm that can efficiently derive longer related-tweakey/key impossible differentials from single-key ones. With application of the MILP technique, the algorithm can be converted an automatic tool for searching related-tweakey/key impossible differentials.
We use this automatic tool to analyze QARMA-64 and give a 11-round key recovery attack, which attacks one more round than the best previous result. Moreover, we also analyze Joltik-BC-128, a internal tweakable block cipher of an authenticated encryption candidate of the CAESAR competition Joltik and our result can attack two more rounds than the result given by the cipher designers.

Keywords: Tweakable block cipher · Impossible differential attack · RelatedTweakey · MILP · Tweakey framework.

1 First Section

In the last decades, a lot of block ciphers have been proposed. A key point for these ciphers to be accepted and used by industry is to provide a reliable security evaluation. Recently, cryptanalysts find many classical cryptanalysis methods could be converted to mathematical optimization problems which aims to achieve the minimal or maximal value of an objective function under certain constraints. Mixed-integer Linear Programming (MILP) is the most widely studied technique to solve these optimization problems. One of the most successful applications of MILP is to search differential and linear trails. Mouha *et al.*[1] first applied MILP method to count active Sboxes of word-based block ciphers. Then, at Asiacrypt 2014, Sun et al. [2] extended this technique to search differential and linear trails, whose key idea is to derive some linear inequalities through the H-Representation of the convex hull of all differential patterns of S-box. Sun *et al.* also provided a greedy algorithm to select certain number of linear inequalities from hundreds of linear inequalities produced by SageMath [3]. Based on H-Representation of the convex hull of S-box and Sun *et al.*'s greedy algorithm, Xiang *et al.* [4] introduced a MILP model to search integral distinguisher, Sasaki *et al.* [5] and

Cui *et al.* [6] gave the MILP-based impossible differential search model independently. There are many more MILP-based tools proposed recently, such as MILP-based differential/linear search model for ARX ciphers [7], MILP-based conditional cube attacks [8, 9] on Keccak [10], etc. Since MILP-based automated evaluation tools could help evaluating various complicated designs in short term, it is being explored intensively by worldwide researchers. In the aspect of searching impossible differential, Sasaki *et al.* [5]'s MILP model could not cover the search of related-tweakey/key impossible differential. Cui emphet al. [6] studied a special case of related-key impossible differential of LBlock. However, it is not a general model and could not be applied to other ciphers trivially. In all, to find the related-tweakey/key impossible differentials using MILP-based method is still an open problem.

Our Contributions: We propose an interesting method that can derive longer related-tweakey/key impossible differentials from single-key ones. The method is heavily reliant on the key schedule. By using the MILP technique, we convert this method to be three MILP modeling process and make it into an automatic tool for searching related-tweakey/key impossible differentials.

We successfully apply this tool to QARMA-64 and Joltik-BC-128 and give the best analysis results of both these two block ciphers.

2 MILP Model to Search Related-Key Impossible Differential

In 2014 [11], the TWEAKEY framework with goal to unify the design of tweakable block ciphers and of block ciphers resistant to related-key attacks was proposed. Since then, many proposals, for example, Deoxys-BC, Joltik-BC, SKINNY and QARMA [12], have followed the TWEAKEY framework and thus take a unified tweakey input instead of a pair key/tweak. For these proposals, the related-key setting is more risky. There are two reasons:

1. One original feature of a tweakable block cipher is that the extra cost of making a block cipher "tweakable" is small. To satisfy this requirement, many tweakable block ciphers adopt a very simple (fully linearity) tweakey schedule. The attacker can utilize the tweakey schedule to mount a related-key attack efficiently;
2. What's more, the tweak part can be public, thus it can be totally controlled by the attacker. Obtaining a specific tweakey differential is more easier, for example, we can just get all nonzero difference from the the tweak part and set the difference of the key part to zero.

Considering the popular future that more tweakable block ciphers will adopt the TWEAKEY framework, the related-tweakey analysis result will be a significant evaluation criteria for these ciphers.

This section introduces a method that can automatically derive longer related-key[3] impossible differentials from single-key impossible differentials. General idea of this method is that by importing tweak/key differences that are equal to the input and output difference of the original single-key impossible differential, check whether the tweak/key differential and the related-key impossible differential still holds. In order to describe this idea more concretely, we introduce the following notations, as shown in Fig. 1:

n the block size of the block cipher, in this paper we assume the bit length of the round tweak/key is also n;

Δ_i the input difference (before key addition) of the i-th round cipher;

Δ'_i the input difference (after key addition) of the i-th round cipher;

Δk_i the difference of the i-th round key or tweak k_i, in Tweakable Block cipher, it is the Tweakey.

Fig. 1 shows a framework to construct a new related tweak/key impossible differential from a r-round single-key impossible differential. We summarise the procedures in Algorithm 1.

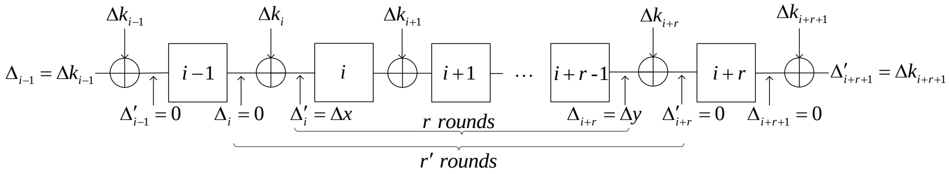

Fig. 1. Framework of Our Work

In Algorithm 1, there are three MILP models to be solved, i.e. $\mathcal{M}_1$, $\mathcal{M}_2$ and $\mathcal{M}_3$.

- In step 1-6, we follow the ideas to find single-key impossible differential from Sasaki et al. [5] and Cui et al. [13]. If $\mathcal{M}_1$ is infeasible, an r-round single-key impossible differential is found with input and output differences ($\Delta_i{}' = \Delta x$, $\Delta_{i+r} = \Delta y$).
- In step 7-9, a MILP model $\mathcal{M}_2$ is constructed to test whether the r'-round tweak/key differential of the cipher's key schedule is valid with $\Delta k_i = \Delta x$ and $\Delta k_{i+r} = \Delta y$. We are going to use the tweak/key differences to introduce the input and output differences of the r-round cipher. If $\mathcal{M}_2$ is feasible, that means there is no contradiction in the tweak/key differential of the r'-round key/tweak Schedule. If $\mathcal{M}_2$ is infeasible, that means there is not any tweak/key pair that satisfy $\Delta k_i = \Delta x$ and $\Delta k_{i+r} = \Delta y$.

  For linear tweakey schedule, such as Joltik-BC, Deoxys-BC, QARMA, given Δx and Δy:

[3] We use "related-key" and "related-tweakey" equivalently as the method can be applied to both block ciphers and tweakable block ciphers.

Algorithm 1: Related-key impossible differential searching algorithm based on MILP technique

```
Input: Block cipher $\mathcal{E}$, $i$ and $r$
Output: Related-tweak/key impossible differential
for All differences $\Delta x$ do
    for All differences $\Delta y$ do
        Construct MILP model $\mathcal{M}_1$ describing the differential behaviour of the $r$-round cipher shown in Fig. 1 in the single-key setting;
        Add constraints to $\mathcal{M}_1$ by setting $\Delta_i{}' = \Delta x$ and $\Delta_{i+r} = \Delta y$;
        if $\mathcal{M}_1$ is infeasible then
            //An $r$-round single-key impossible differential obtained [5, 13];
            Construct MILP model $\mathcal{M}_2$ describing the differential behaviour of the $r'$-round tweak/key schedule;
            Add constraints to $\mathcal{M}_2$ by setting $\Delta k_i = \Delta x$ and $\Delta k_{i+r} = \Delta y$;
            if $\mathcal{M}_2$ is feasible then
                Construct MILP model $\mathcal{M}_3$ describing the related-tweak/key differential behaviour of $r'$-round cipher;
                Add constraints to $\mathcal{M}_3$ by setting $\Delta k_i = \Delta x$ and $\Delta k_{i+r} = \Delta y$, setting $\Delta_i{}' = \Delta x$ and $\Delta_{i+r} = \Delta y$;
                if $\mathcal{M}_3$ is infeasible then
                    Compute $\Delta k_{i-1}$ and $\Delta k_{i+r+1}$ from the tweak/key schedule by setting $\Delta k_i = \Delta x$ and $\Delta k_{i+r} = \Delta y$;
                    if $\Delta k_{i-1} \neq 0$ and $\Delta k_{i+r+1} \neq 0$ then
                        Return $(r+2)$-round related tweak/key impossible differential with input and output differences $(\Delta_{i-1} = \Delta k_{i-1},\ \Delta'_{i+r+1} = \Delta k_{i+r+1})$
                    else
                        If $\Delta k_{i-1} = 0$, continue to compute $\Delta k_{i-2}$, and if $\Delta k_{i-2} \neq 0$, return $(r+3)$-round impossible differential $(\Delta_{i-2} = \Delta k_{i-2},\ \Delta'_{i+r+1} = \Delta k_{i+r+1})$;
                        ......
```

- If the tweakey size is n (equals to the block size), one of the two conditions $\Delta k_i = \Delta x$ and $\Delta k_{i+r} = \Delta y$ is satisfied, thus we could only extend the trail in one direction, and $r+1$-round related-tweakey impossible differential is expected to find.
- If the tweakey size is $2n$, it is expected that there is only one tweakey differential characteristic, whose probability is 1, that satisfy $\Delta k_i = \Delta x$ and $\Delta k_{i+r} = \Delta y$ on average. Thus an $r+2$-round related-tweakey impossible differential is expected to find.
- If the tweakey size is $3n$, not only the two conditions $\Delta k_i = \Delta x$ and $\Delta k_{i+r} = \Delta y$ could be satisfied, but also one of $\Delta k_{i-1} = 0$ and $\Delta k_{i+r+1} = 0$ conditions could be satisfied in step 17. Thus we could extend one more round to get an $r+3$-round related-tweakey impossible differential.

For nonlinear tweakey schedules, we have more freedom. Since an input key difference of an S-boxes will get many more possible output differences, that means if we fixed the above two constrains in the input and output sides, there will be a key differential characteristic on average with certain probability smaller than 1. And $\mathcal{M}_2$ will output the feasible solution. However, it is more like a weak-key setting, that the valid keys must belongs to a given very small subset of the full key space. Thus for nonlinear tweakey schedules, our algorithm still work, but the distinguisher will be weaker.

- In step 10-13, $\mathcal{M}_3$ is constructed and solved.
  - If $\mathcal{M}_3$ is feasible, that means the added tweak/key differences to the r-round single-key differential model $\mathcal{M}_2$ makes it feasible. Thus the r'-round related tweak/key differential is possible. item If $\mathcal{M}_3$ is infeasible, we get a r'-round related tweak/key impossible differential.
- In step 14-18, we extend the r'-round related tweak/key impossible differential to $r+2$ rounds or more.

3 The Application to QARMA Block Cipher

QARMA is a lightweight tweakable block cipher recently accepted by FSE 2017 which has been used by the ARMv8 architecture to support a software protection feature. It contains two versions: QARMA-64 and QARMA-128. In this paper, we focus on QARMA-64, for more information, we refer to [14].

3.1 The Tweakable Block Cipher QARMA-64

QARMA-64 is a SPN structure with 14 rounds and the central construction (two-round functions and a *Pseudo-Reflector* construction). The encryption process can be seen as a sequence of operations on the 64-bit internal state together with a tweak and the key. Internal state size of QARMA-64 can be represented as sixteen 4-bit cells, which are indexed in big endian order, while the bits in a cell are ordered in little endian order, *e.g.*, a plaintext P can be expressed as:

$$P = p^0 \| p^1 \| p^2 \| \cdots \| p^{15} = \begin{pmatrix} p^0 & p^1 & p^2 & p^3 \\ p^4 & p^5 & p^6 & p^7 \\ p^8 & p^9 & p^{10} & p^{11} \\ p^{12} & p^{13} & p^{14} & p^{15} \end{pmatrix} \tag{1}$$

The forward round function F includes 4 operations:

KeyAddition($\boldsymbol{K}$): The i^{th} 64-bit round key K_i is XORed to the state S with the round tweak T_i and round constant c_i.

ShuffleCell(τ): This operation is a simple cell permutation, i.e.,

$$S^{0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15} \rightarrow S^{0,11,6,13,10,1,12,7,5,14,3,8,15,4,9,2}. \tag{2}$$

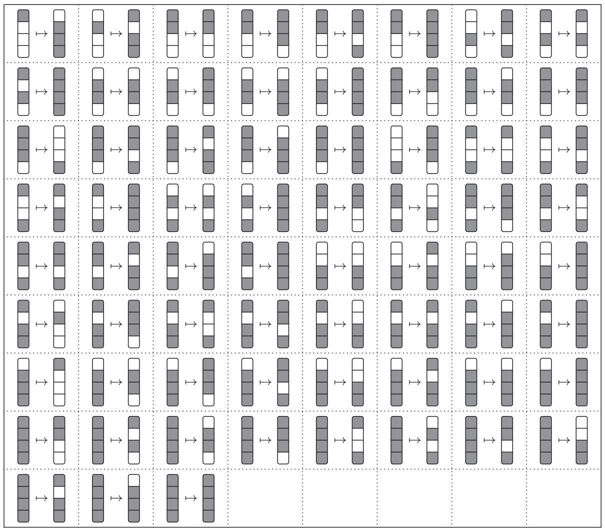

Fig. 2. The Column-wise Active State Transitions for the Matrice in QARMA-64

MixColumn($\boldsymbol{M}$): Each column of the cipher internal state array is multiplied by an involutory matrix M. All possible transitions of M is depicted in Fig. 2.

$$M = \begin{pmatrix} 0 & \rho & \rho^2 & \rho \\ \rho & 0 & \rho & \rho^2 \\ \rho^2 & \rho & 0 & \rho \\ \rho & \rho^2 & \rho & 0 \end{pmatrix}. \tag{3}$$

The multiplication of an element of the array with ρ^i is just a simple left circular rotation of the element by i bits.

Table 1. Sbox of QARMA-64

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
δ_0	0	14	2	10	9	15	8	11	6	4	3	7	13	12	1	5
δ_1	10	13	14	6	15	7	3	5	9	8	0	12	11	1	2	4
δ_2	11	6	8	15	12	0	9	14	3	7	4	5	13	2	1	10

SubCell(S): Apply the non-linear 4×4 S-box in parallel on each nibble of the state. The designer propose three kinds of Sbox shown in 1.

The backward round function is the inverse of the forward round function. Notice that a short version of the forward round function exists in the first forward round and the last backward round which omits the ShuffleCell and MixColumn operation.

The central construction is made up of one forward round, one backward round and a *Pseudo-Reflector* constructions. The *Pseudo-Reflector* construction includes four parts which is essentially a ShuffleCell-MixColumn-KeyAddition-Inverse ShuffleCell operation.

For convenience, we use T_i^j to denote the j_{th} nibble of the tweak state inside the i_{th} round, similarly, for the cipher internal state: $X_i \xrightarrow{\tau} Y_i \xrightarrow{M} Z_i \xrightarrow{S} W_i$.

The Tweak (T) Update Function

There are two operations for the tweak during every round: $\mathcal{W}$ and $\mathcal{H}$. The $\mathcal{H}$ operation is a nibble permutation: $\mathcal{H} = [6, 5, 14, 15, 0, 1, 2, 3, 7, 12, 13, 4, 8, 9, 10, 11]$; the $\mathcal{W}$ operation is essentially a LFSR that updated the 4-bit tweak cells from $(b3, b2, b1, b0)$ to $(b0 \oplus b1, b3, b2, b1)$ with indexes $(0, 1, 3, 4, 8, 11, 13)$.

The Key Schedule

In the attack on QARMA-64 of this paper, all used round keys($K_{5,6,13,14,15}$) are the same value. They are decided by the same 64 bits of the master key. For more information about the key schedule, we refer to [14].

3.2 Proposition

Proposition 1 (Differential Property of Sbox, [15]).

Given the nonzero input and output differences of an Sbox, there exists only one pair of actual values on average to satisfy these two differences.

3.3 A 7-Round Related-tweak Impossible Distinguisher for QARMA-64

Apply Algorithm 1 to QARMA-64, we obtain a related-tweak impossible distinguisher of QARMA that contains 7 round functions (6-12) and the $Pseudo-Reflector$ construction, shown in Fig. 3. The distinguisher is explained in a miss-in-the-middle method.

Distinguisher:

The differential: $(00000000000\Delta W_5^{11}0000) \rightarrow (000000\Delta W_{13}^6000000000)$ is impossible when the following conditions are satisfied:

(1) ΔW_5^{11} is the only active nibble of ΔW_5; (2) ΔT_6^{11} is the only active nibble of ΔT_6; (3) $\Delta W_5^{11} = \Delta T_6^{11}$; (4) ΔW_{13}^6 is the only active nibble of ΔW_{13}; (5) ΔT_{12}^6 is the only active nibble of ΔT_{12}; (6) $\Delta W_{13}^6 = \Delta T_{12}^6$; (7) Choose ΔT_{11}^0 that $\text{Sbox}(\rho \cdot \Delta T_{11}^0 \oplus x) \oplus \text{Sbox}(x) = \mathcal{W}(\Delta T_{11}^0)$ has no solutions[4].

Proof:

In the forward direction, as shown in Fig. 3, when condition (1)-(3) are satisfied, according to the transition property of the ShuffleCell operation and the MixColumn operation, $\Delta X_8^{0,5,10,15}$ will be inactive nibbles with probability 1.

And when condition (2) is satisfied, according to tweak update function, ΔT_8^{15} will be an active nibble. Thus, after an KeyAddition operation, ΔW_9^{10} is still inactive.

In the backward direction, when condition (4)-(6) are satisfied, it's easy to verify that $\Delta Y_{12}, \Delta Z_{12}$ and ΔW_{12} will be all inactive. The only active nibble ΔX_{11}^0 will come from ΔT_{11}^0. After a round function, $\Delta W_{11}^{4,8,12}$ will be all active.

Notice that $\Delta Z_{11}^4 = \rho \cdot \Delta Y_{11}^0 = \rho \cdot \Delta X_{11}^0 = \rho \cdot \Delta T_{11}^0$ and $\Delta W_{11}^4 = \Delta T_{10}^4 \oplus \Delta X_{10}^4 = \mathcal{W}(\Delta T_{11}^0) \oplus \Delta X_{10}^4$, thus condition (7) will insure that ΔX_{10}^4 is an active nibble as $\text{Sbox}(\Delta Z_{11}^4) = (\Delta W_{11}^4)$ must have solutions.

After another two backward rounds, the active nibbles $(\Delta X_{10}^{4,8,12}, \Delta T_9^{11})$ will insure that $\Delta Y_9^{1,4,5,7,8,9,10,12,14,15}$ are all active nibbles and the other 6 nibbles in ΔY_9 are all inactive nibbles.

By now, as the contradiction is only related to $\Delta Y_9^{10,14}$, we only consider the transition situation of the third column. According to Fig. 2, when $\Delta Y_9^{10,14}$ are the only two active nibbles in the third column of ΔY_9, $\Delta Z_9^{10,14}$ will also be two active nibbles with probability 1.

Thus, in the backward direction, ΔZ_9^{10} is active; after a SubCell operation, in the forward direction, ΔW_9^{10} is inactive.

This is a contradiction, thus when all 7 conditions are satisfied, the differential is impossible.

3.4 Attack on 11-Round QARMA-64

Based on the distinguisher in 3.3, we mount an attack on 11-round QARMA-64 with the $Pseudo-Reflector$ construction from round 5 to round 15. The distinguisher used in the attack is from round 7 to round 12.

Attack Procedure

(1) Choose two tweaks $(T, \overline{T})$ satisfying that $\text{Sbox}(\rho \cdot \Delta T_{11}^0 \oplus x) \oplus \text{Sbox(x)} = \mathcal{W}(\Delta T_{11}^0)$ has no solutions.

[4] There are many values of ΔT_{11}^0 that satisfy condition (7), e.g., $\Delta T_{11}^0 = 0001$ for QARMA-64 employing Sbox δ_0 or δ_1 and $\Delta T_{11}^0 = 0010$ for QARMA-64 employing Sbox δ_2.

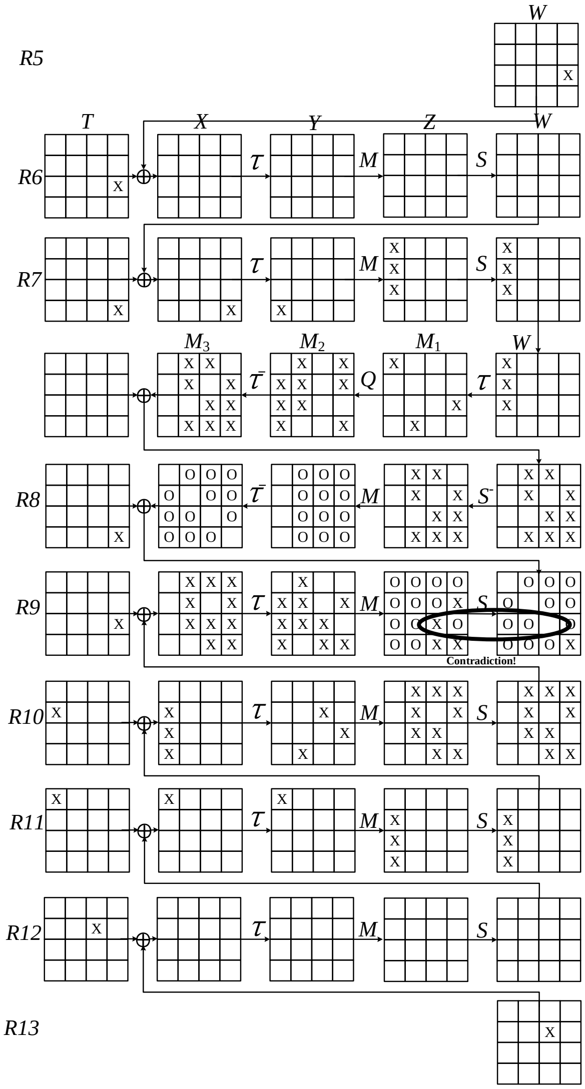

Fig. 3. 7-Round Related-tweak Impossible Distinguisher of QARMA-64. White nibbles have zero difference, nibbles with x have non-zero difference, nibbles with o can be zero or non-zero difference.

(2) Under T, construct 2^n structures that each structure traverse 4 nibbles: $P_5^{7,8,11,13}$ and the other 12 nibbles are constants; under $\overline{T}$, construct 2^n structures that each structure satisfies that a): $\overline{P_5^{7,8,11,13}}$ traverses all possible values, b): $\overline{P_5^4} = \Delta T_5^4 \bigoplus Y_5^4$, and c): the values of the other 11 nibbles are the same as the corresponding 11 nibbles of P_5.

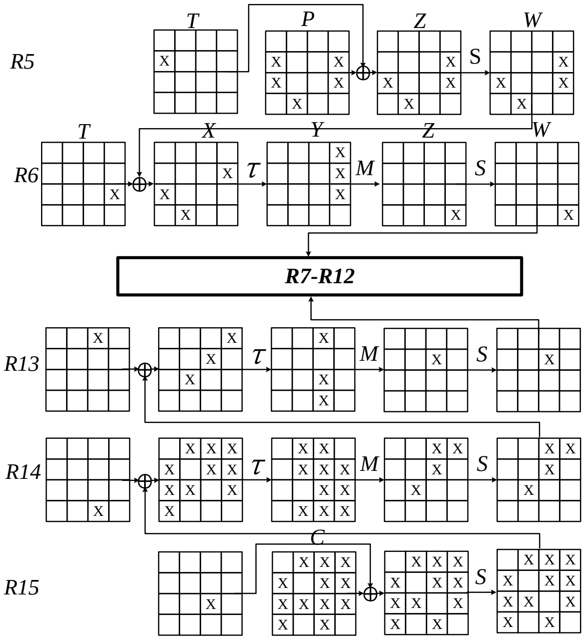

Fig. 4. Related-tweak Impossible Attack on 11 Rounds of QARMA-64

Totally, we can get 2^{n+32} pairs.

(3) Encrypt the plaintexts and only choose the pairs that satisfy $\Delta C^{0,5,13,15} = 0$ and $\Delta C^{10} = \Delta T_{15}^{10}$.
This step performs a 20-bit filter, there are 2^{n+12} pairs remaining.

(4) For the remaining pairs, do the following steps:

(4.a) Guess the value of $K_5^{7,8,13}$ and encrypt the pairs to deduce the value of $(W_5^{7,8,13}, \overline{W_5^{7,8,13}})$.
According to the key schedule, we can also known the value of $K_6^{7,8,13}$, so we can deduce $\Delta X_6^{7,8,13}$. As $Y_6^{3,7,11} = X_6^{13,7,8}$, we also get the value of $\Delta Y_6^{3,7,11}$. After a MixColumn operation, 2^{n+4} pairs that satisfy $\Delta Z_6^{3,7,11} = 0$ remain.

(4.b) According to key schedule, $K_{15}^{7,8} = K_5^{7,8}$. By guessing the value of K_{15}^2, we can know the value of $K_{15}^{2,7,8}$ and get the value of $(Z_{15}^2, \overline{Z_{15}^2})$ by decrypting $(C^2, \overline{C^2})$. After a SubCell operation, we get $(W_{15}^2, \overline{W_{15}^2})$.
Also as $K_{14}^{2,7,8} = K_{15}^{2,7,8}$, we can get the value of $(X_{14}^{2,7,8}, \overline{X_{14}^{2,7,8}})$. After another ShuffleCell and MixColumn operation, we can get the value of $\Delta Z_{14}^{3,7,11,15}$. 2^{n-4} pairs that satisfy $\Delta Z_{14}^{7,11,15} = 0$ remain. The value of $(W_{14}^3, \overline{W_{14}^3})$ can also be deduced.

(4.c) Guess the value of K_5^{11}, encrypt the pairs to get the value of ΔW_5^{11}. About 2^{n-8} pairs that satisfy $\Delta W_5^{11} = \Delta T_6^{11}$ remain.
Till now, we get the pairs that satisfy the input difference of the distinguisher: ΔZ_6^{15} is the only active nibble in ΔZ_6.

(4.d) By guessing $K_{15}^{1,4}$, together with the value of K_{15}^{11} as $K_{15}^{11} = K_5^{11}$, we decrypt $(C^{1,4,11}, \overline{C^{1,4,11}})$. We can get the value of $(W_{15}^{1,4,11}, \overline{W_{15}^{1,4,11}})$.

As $K_{14}^{1,4,11} = K_{15}^{1,4,11}$, we can also get the value of $(X_{14}^{1,4,11}, \overline{X_{14}^{1,4,11}})$. After a ShuffleCell and Mixcolumn operation, we can get the value of $\Delta Z_{14}^{1,5,9,13}$. About 2^{n-16} pairs that satisfy $\Delta Z_{14}^{1,5,13} = 0$ remain. For the left pairs, $(W_{14}^9, \overline{W_{14}^9})$ is also known.

(4.e) Guess the value of K_{15}^{14} and decrypt $(C^{14}, \overline{C^{14}})$ to deduce the value of $(Z_{15}^{14}, \overline{Z_{15}^{14}})$. After a SubCell operation, we can get the value of ΔW_{15}^{14}. About 2^{n-20} pairs that satisfy $\Delta W_{15}^{14} = \Delta T_{14}^{14}$ remain.

(4.f) Guess the value of $K_{15}^{3,6,9,12}$ and decrypt $(C^{3,6,9,12}, \overline{C^{3,6,9,12}})$ to deduce the value of $(X_{14}^{3,6,9,12}, \overline{X_{14}^{3,6,9,12}})$ as $K_{14}^{3,6,9,12} = K_{15}^{3,6,9,12}$.

After another ShuffleCell and MixColumn operation, we can get the value of $(Z_{14}^{2,6,10,14}, \overline{Z_{14}^{2,6,10,14}})$. To get the output difference of the distinguisher, only pairs that satisfy $\Delta Z_{14}^{10,14} = 0$ and $\Delta W_{14}^2 = \Delta T_{13}^2$ remain. For the pairs, the value of $(W_{14}^6, \overline{W_{14}^6})$ can also be deduced.

By now, only $\Delta X_{13}^{3,6,9}$ can be active nibbles in ΔX_{13}. As $K_{13}^{3,6,9} = K_{15}^{3,6,9}$, we can get the value of $(X_{13}^{3,6,9}, \overline{X_{13}^{3,6,9}})$. After a ShuffleCell and MixColumn and SubCell operation, we can get the value of $\Delta Z_{13}^{2,6,10,14}$. Only pairs that satisfy $\Delta Z_{13}^{2,10,14} = 0$ and $Sbox(Z_{13}^6) \bigoplus Sbox(\overline{Z_{13}^6}) = \Delta T_{12}^6$ remain.

This step performs a 24-bit filter, about 2^{n-44} pairs remain.

(5) Eliminate all the wrong key values and get the correct value of $k_0^{1,4,11,14,3,6,9,12,2,7,8,13}$. Finally, we exhaustively search the other 80-bit value of the master key.

Complexity: For each 48-bit value of $k_0^{1,4,11,14,3,6,9,12,2,7,8,13}$, the expected number of left pairs is $N = 2^{n-44}$. To ensure that $N > 1$, we choose $n = 44$. Then, the date complexity is $2^{44+16+1} = 2^{61}$ plaintexts. The time complexity of the attack procedure is shown in 2. E denotes a 11-round encryption unit from round 5 to round 15.

Table 2. Time Complexity of Attack on 11-Round QARMA-64

Step	Time Complexity
(3)	$2^{61}E$
(4.a)	$2^{69} \times 2 \times \frac{3}{16} \times \frac{1}{12} \approx 2^{64}E$
(4.b)	$2^{65} \times 2 \times \frac{3}{16} \times \frac{1}{12} \approx 2^{60}E$
(4.c)	$2^{61} \times 2 \times \frac{1}{16} \times \frac{1}{12} \approx 2^{54.5}E$
(4.d)	$2^{69} \times 2 \times \frac{3}{16} \times \frac{1}{12} \approx 2^{60}E$
(4.e)	$2^{61} \times 2 \times \frac{1}{16} \times \frac{1}{12} \approx 2^{54.5}E$
(4.f)	$2^{69} \times 2 \times \frac{1}{16} \times \frac{1}{12} + 2^{49} \times 2 \times \frac{1}{16} \times \frac{1}{12} \approx 2^{62.5}E$
SUM	$2^{64.4}E$

4 The Application to Joltik-BC Block Cipher

4.1 The Tweakable Block Cipher Joltik-BC

In this section, we recall the details of Joltik-BC-128 block cipher. We assume that the reader is familiar with the AES block cipher [16]. Fig. 5 shows the structure of Joltik-BC-128.

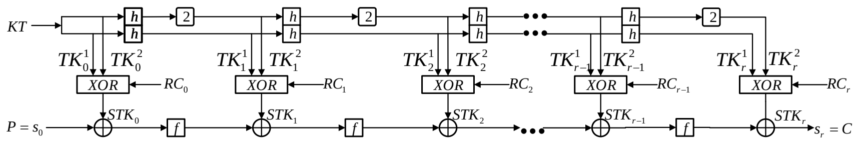

Fig. 5. Structure of Joltik-BC-128

Joltik-BC is the internal ad-hoc tweakable block cipher of the Joltik authenticated encryption scheme, conforming to the TWEAKEY framework [11]. Except the two standard inputs of a block cipher, a plaintext P and a key K, it adopts a third input called a tweak T, i.e., $E_K(T, P) = C$. According to the TWEAKEY framework, we can use a single input, called the tweakey, to unified view the the tweak and the key. The length of the tweakey is the cumulative size of the key and the tweak. For Joltik-BC-128, the tweakey size is 128-bit; for Joltik-BC-192, the tweakey size is 192-bit. In this paper, we focus on Joltik-BC-128. For more information, we refer to [17].

Joltik-BC is an AES-like design, i.e. it is an iterative substitution-permutation network that transforms the initial plaintext through series of round functions (that depend on the key and the tweak) to a ciphertext. As most AES-like designs, the state of Joltik-BC is seen as 4×4 matrix of nibbles, where each nibble is a 4-bit word. We denote the base field by K as $GF(16)$ defined by the irreducible polynomial $x^4 + x + 1$ (sometimes noted in hexadecimal display 0x13). The number r of rounds is 24 for Joltik-BC-128. One round, similarly to a round in AES, has the following four transformations applied to the internal state.

Firstly we stress that to be consistent with the Joltik document, the index of the internal state is in column major order, *e.g.*, a plaintext P can be expressed as:

$$P = p^0 \| p^1 \| p^2 \| \cdots \| p^{15} = \begin{pmatrix} p^0 & p^4 & p^8 & p^{12} \\ p^1 & p^5 & p^9 & p^{13} \\ p^2 & p^5 & p^{10} & p^{14} \\ p^3 & p^6 & p^{11} & p^{15} \end{pmatrix}, \tag{4}$$

which is different from the index rule of the QARMA block cipher.

Joltik-BC-128 round function. The round function, similar with AES, has four operations applied to the internal state as follows:

- **AddRoundTweakey(AK)** - XOR the 64-bit round subtweakey (defined further) to the internal state,
- **SubNibbles(SB)** - Apply the 4-bit Sbox **S** odefined below to the 16 nibbles of the internal state,
- **ShiftRows(SR)** - Rotate the 4-nibble i-th row left by $\rho[i]$ positions, where ρ=(0,1,2,3),
- **MixColumns(MC)** - Multiply the internal state by the 4×4 constant MDS matrix **M** defined below.

After the last round, a final **AddRoundTweakey** operation is performed to produce the ciphertext.

Table 3. Sbox of Joltik-BC-128

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

The 4-bit SBox **S** we use in Joltik-BC is the one selected for the Piccolo block cipher [18], and is exhaustively defined by: The MDS matrix M with coefficients in **K** we use in Joltik-BC is non-circulant, MDS and involutory:

$$\mathbf{M} = \begin{pmatrix} 1 & 4 & 9 & 13 \\ 4 & 1 & 13 & 9 \\ 9 & 13 & 1 & 4 \\ 13 & 9 & 4 & 1 \end{pmatrix} = \overline{\mathbf{M}}.$$

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

Definition of the Subtweakey. The structure of the tweakey schedule distinguish Joltik-BC from the classical construction of an AES-like block cipher.

Let us denote with STK_i the subtweakey (a 64-bit word) that is added to the state at round i of the cipher with the **AddRoundTweakey** operation. For Joltik-BC-128, subtweakey is defined as:

$$STK_i = TK_i^1 \oplus TK_i^2 \oplus RC_i.$$

The 64-bit words TK_i^1, TK_i^2 are outputs produced by a special key schedule algorithm. A single instance of this algorithm, denotedas $KS(W, \alpha)$, takes as inputs a 64-bit word W and a nibble α and produces subkeys $TK_0, TK_1, \ldots$ The subkeys are produced sequentially, one from another (where $TK_0 = W$), by applying two permutations: a nibble permutation h, and a finite field multiplication g:

$$TK_{i+1} = g(h(TK_i)).$$

The nibble permutation h is defined as:

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

Furthermore, g is a finite field multiplication $\mathbf{K}$ of each nibble by α (recall that α is input to the key schedule algorithm).

Let us define the inputs W and α. Denote the concatenation of the key K and the tweak T as KT, i.e. $KT = K||T$. Then, in Joltik-BC-128, the size of KT is 128 bits. The first (most significant) 64 bits of KT is W_1, while the second W_2. Then, TK_i^1 are the output words of the key scheduling algorithm $KS(W_1, 1)$, and TK_i^2 are the output words of the key scheduling algorithm $KS(W_2, 2)$.

Finally, RC_i are the key schedule round constants, and are defined as:

$$RC_i = \begin{pmatrix} 1 & (rc_5||rc_4||rc_3) & 0 & 0 \\ 2 & (rc_2||rc_1||rc_0) & 0 & 0 \\ 4 & (rc_5||rc_4||rc_3) & 0 & 0 \\ 8 & (rc_2||rc_1||rc_0) & 0 & 0 \end{pmatrix}$$

where rc_i are fixed constants.

4.2 Proposition

Proposition 2 (Subtweakey Difference Cancellation).

As noticed by the designers [17], a single subtweakey difference cancellation can happen every 15 rounds for Joltik-BC-128. Suppose that a single cell of TK^1 and TK^2 are active. Let a1 and a2 be differences of the active cells respectively. Then the subtweakey difference of the first round is $a2 \oplus a1$ at this cell, and in the $i_t h$ round, the subtweakey difference is $KS^i(a2, 2) \oplus a1$ ignoring the position permutation h. Since a1 and a2 are both nonzero differences, $KS^i(a2, 2) \oplus a1 = 0$ can happen no more than one time.

4.3 A 6-round Related-Tweakey Impossible Distinguisher

Apply Algorithm 1 to Joltik-BC-128, we get a 6-round related-tweak impossible distinguisher as shown in Fig. 6.

Similarly, we will explain the distinguisher in a miss-in-the-middle method.

Distinguisher

In Fig. 6, it is easy to verify that the differential from X_1 to Z_4 is impossible in the single-tweakey scenario. $X_1[11]$ is the only active nibble of X_1, after a 2-round encryption, all nibbles of Z_2 are active. $Z_4[0, 1, 2]$ are the only 3 active nibbles of Z_4, after a 2-round decryption, X_3 only have 12 active nibbles. As $Z_2 = X_3$, we get contradictions in 4 nibbles.

In the related-tweakey setting, the differential:

$(00000000\mathit{\Delta} W_0[8]\mathit{\Delta} W_0[9]0\mathit{\Delta} W_0[11]0000) \rightarrow (000000\mathit{\Delta} X_7[6]00000000\mathit{\Delta} X_7[15])$ is impossible when the following conditions are satisfied:

(1) $\mathit{\Delta} W_0[8, 9] = \mathit{\Delta} STK_1[8, 9]$;

(2) $\Delta X_7[6,15] = \Delta STK_7[6,15]$;
(3) ΔSTK_6 are inactive in all 16 nibbles;
(4) $\Delta Z_4[3] = 0$.
Proof:

In the forward direction, when condition (1) is satisfied, $X_1[11]$ will be the only active nibble of X_1. After a 1-round encryption, the fourth column of W_1 will be active in all 4 nibbles. In the second round, $STK_2[9,14]$ are the two active tweakey nibbles, after xoring the internal state difference with ΔSTK_2 and the second round function, W_2 will be active in all 12 nibbles in the 1st, 2nd and 4th column.

In the backward direction, when conditions (2) and (3) are satisfied, all internal state nibbles in $R5$ and $R6$ are inactive. The difference of W_4 will be imported from STK_5. Considering condition (4), $Z_4[0,1,2]$ are the only 3 active nibble of Z_4. After a $\overline{SR}$ and $\overline{MC}$ operation, X_4 will be active only in $X_4[0,5,10]$. Xor ΔX_4 with ΔSTK_4, as the 4th column of W_3 is inactive, after a $\overline{MC}$, $\overline{SR}$ and $\overline{SB}$ operation, the value of $\Delta X_3[1,6,11,12]$ will be 0. As $\Delta STK_3[1,6,11,12] = 0$, $\Delta W_2[1,6,11,12]$ will also be 0.

So there are contradictions in $W_2[1,11,12]$ when considering both the forward and backward direction, thus when all 4 conditions are satisfied, the differential is impossible.

Note. In fact, the differential before the contradictions can have more possibilities, for example, the active nibble of X_1 can be anyone of the four nibbles of the third column. It's easy to verify that there are still contradictions existing, but the index of contradict nibbles changes, the corresponding active nibbles of W_0 also change, respectively. In total, there are $2^4 \times 4 = 2^6$ possible related-twekaey impossible differential applicable for the distinguisher.

4.4 The 9-round Key Recovery Attack

By adding one round on the top and two rounds on the bottom of the distinguisher in Sect. 4.3, we mount a 9-round key recovery attack on Joltik-BC-128. The attack differential is shown in Fig. 7. The attack process is as follows:

1. Construct 2^n structures that each structure is made up of 2^{16} plaintexts. In each structure, we set $\Delta P[15] = \Delta STK_0[15]$ (a fixed value) and $\Delta P[2,7,8,13]$ the 4 active bytes.
2. Choose (KT, KT') that the tweakey difference satisfy the subtweakey difference trail. Encrypt the plaintexts under two tweakeys and only choose the pairs that satisfy $\Delta Z_8[1,2,4,5,11,14,15]\overline{MC}(\Delta C \oplus \Delta STK_9)[1,2,4,5,11,14,15] = 0$.

   In total, we will get about $2^{n+16\times 2-4\times 7} = 2^{n+4}$ pairs.

   For each of the remaining pairs, do the following steps:
3. As mentioned by the *Note* in the end of Sect. 4.3, there are 2^6 possible values of $\Delta Z_0[8,9,10,11]$. For each possible value of $\Delta Z_0[8,9,10,11]$, by a $\overline{MC}$ and $\overline{SR}$ operation, we can deduce the difference value $\Delta Y_0[2,7,8,13]$.

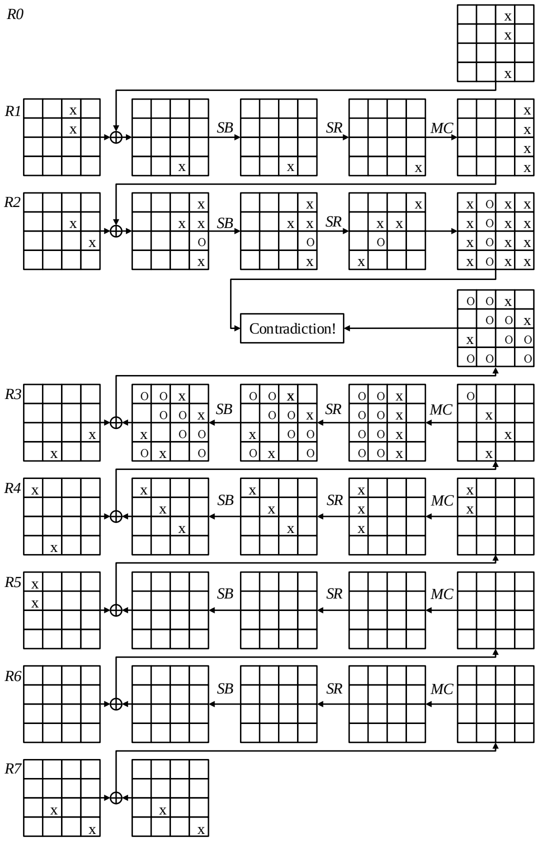

Fig. 6. The 6-round related-tweakey impossible distinguisher of Joltik-BC-128.

Considering that we can get the value of $\Delta X_0[2,7,8,13]$ from ΔP and ΔSTK_0 (known and fixed), by using Proposition. 2, we can deduce the value of $X_0[2,7,8,13]$. So we can get 6-bit information of $STK_0[2,7,8,13]$ as $STK_0[2,7,8,13] = P[2,7,8,13] \oplus X_0[2,7,8,13]$.

4. Guess the value of $\Delta Z_7[3,14]$.

   We can straightly deduce $\Delta W_7[0,1,2,3,12,13,14,15]$ by a MC operation. By xoring ΔW_7 with ΔSTK_8, we get the value of $\Delta X_8[0,1,2,3,8,12,13,14,15]$.

   In the backward direction, from ΔC and ΔSTK_9, we can get ΔW_8. After a $\overline{MC}$ and $\overline{SR}$ operation, we get the differences of active nibbles of Y_8:$\Delta Y_8[0,1,2,3,8,12,13,14,15]$. By using Proposition. 2, we can get the value of $X_8[0,1,2,3,8,12,13,14,15]$ and $Y_8[0,1,2,3,8,12,13,14,15]$. As $Y_8[0,1,2,3,8,12,13,14,15]$ $= \overline{SR}Z_8[0,13,10,7,8,12,9,6,3] = \overline{MC}STK_9 \oplus \overline{MC}C$. So we get 8-bit information of STK_9.

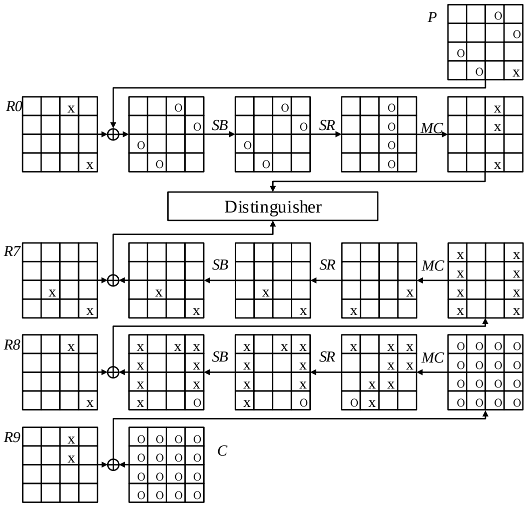

Fig. 7. The 9-round attack on Joltik-BC-128.

What's more, as $\Delta X_7[6,15] = \Delta STK_7[6,15]$ and $\Delta Y_7[6,15] = \Delta Z_7[3,14]$, by using Proposition. 2, we can get the value of $Y_7[6,15]$ and $Z_7[3,14] = Y_7[6,15]$. As $Z_7[3,14] = \overline{MC}(X_8)[3,14] \oplus \overline{MC}(STK_8)[3,14]$. So we also get 8-bit information of

5. We can use as the above steps to filter the wrong key values and then exhaustively search the left key bits.

Complexity Computation:

In total, the number of deduced key nibbles is $4 + 2 + 9 = 15$, i.e., 60 bits information of the tweakey. As we guess 2^8 values of $\Delta Y_7[6,15]$ and there are 2^6 possible values of $\Delta W_0[8,9,10,11]$, each pair can eliminate 2^{14} values of the 60-bit guessed tweakey information. To balance the time complexity and the data complexity, we do not eliminate all wrong values of the 60-bit key information but only filter 2^{-4} of it. So, to satisfy $2^{60} \times (1 - 2^{14}/2^{60})^{2^{n+4}} \ll 2^{56}$, we choose $n = 44$.

The data complexity is $2^{44+16} = 2^{60}$ plaintexts. The time complexity of step 3 for encrypting the plaintexts is $2 \cdot 2^{44+16} = 2^{61}$. In step (3), the total number of guesses is $2^{4+n+14} = 2^{62}$, which is equivalent to $2^{62} \cdot (3/16 + 2/16 + 9/16) \cdot 1/9 \cdot 2 \approx 2^{60.4}$ 9-round encryptions. Thus the time complexity is approximately $2^{61.7}$ 9-round encryptions.

4.5 The 10-round Key Recovery Attack

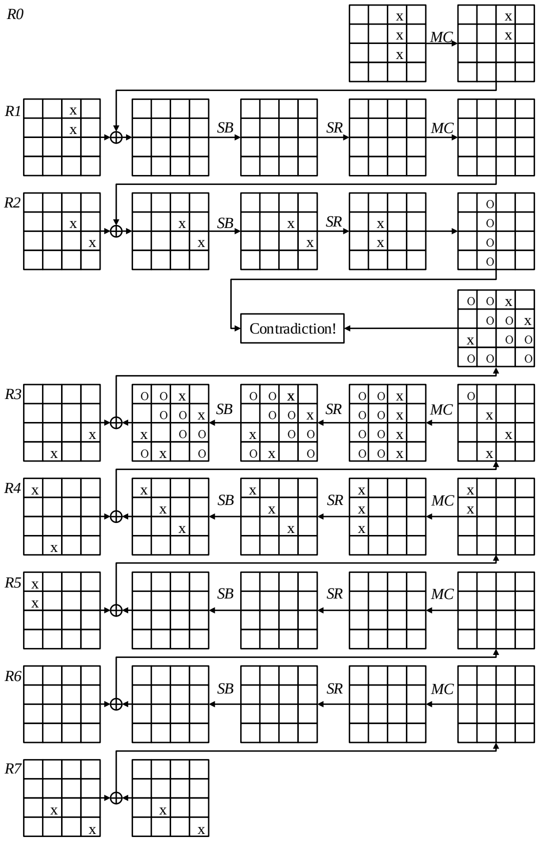

Fig. 8. The 6-round distinguisher for attacking 10-round Joltik-BC-128.

Distinguisher We also seek out another distinguisher and based on which we propose a 10-round key recovery attack. The distinguisher is depicted in Fig. 8.

The original single-key impossible differential used to derive the related-key one is from X_2 to Z_4. It's a 2.5-round impossible differential and easy to verify.

By utilizing Proposition. 2 and extending rounds both on the top and the bottom of the single-key impossible differential, we get this 6-round related-tweakey impossible distinguisher. As it is very similar with the one in Sect. 4.3, we do not provide the detailed introduction. The only area of note is that once the MC operation in $R4$ is a $3-to-2$ transformation, so is the MC operation in $R0$. This is because that $\Delta W_4[0,1] = \Delta STK_5[0,1]$ and $\Delta W_0[0,1] = \Delta STK_1[0,1]$ and the correlation of ΔSTK_1 and ΔSTK_5. So we include Z_0 into the distinguisher. The full derivation process is given in 5.

Attack Process By adding two rounds both on the top and the bottom of the distinguisher in Fig. 8, we successfully mount a 10-round key recovery attack on Joltik-BC-128, shown in Fig. 9.

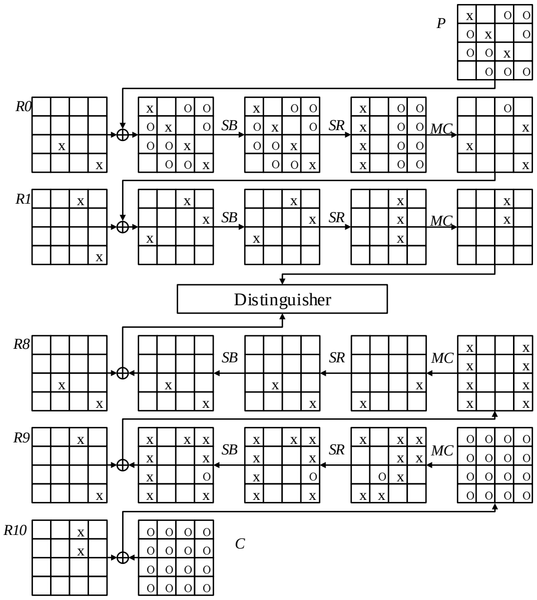

Fig. 9. The 10-round attack on Joltik-BC-128.

The attack process is as follows:

1. Construct 2^n structures that each structure is made up of 2^{48} plaintexts. In each structure, $P[0, 1, 2, 5, 6, 7, 8, 10, 11, 12, 13, 15]$ are the 12 active nibbles.
2. Choose (KT, KT') that the tweakey difference satisfy the subtweakey difference trail. Encrypt the plaintexts under two tweakeys and choose the pairs that satisfy $\overline{MC}(\Delta C \oplus \Delta STK_{10})[1, 2, 4, 5, 11, 14, 15] = 0$.
   In total, we will get about $2^{n+48\times2-4\times7} = 2^{n+68}$ pairs.
   For each of the remaining pairs, do the following steps:
3. Guess the value of $\Delta W_0[2, 8, 13]$. Since $\Delta STK_1[8, 15]$ is known, we get the value of $\Delta X_1[2, 8, 13]$. From $\Delta W_1[8, 9]$, after a $\overline{MC}$ and $\overline{SR}$ operation, we get the value of $\Delta Y_1[2, 8, 13]$. By using Proposition. 1, we can deduce the value of $X_1[2, 8, 13]$.

   What's more, the difference value of all 4 active nibbles of W_0 is also known, by a $\overline{MC}$ and $\overline{SR}$ operation, we can know the value of $\Delta Y_0[0,1,2,5,6,7,8,10,11,12,13,15]$.

   According to plaintext difference and $\Delta STK_0[6,15]$, we can get the difference value of $X_0[0,1,2,5,6,7,8,10,11,12,13,15]$. Using Proposition. 1, we get the value of $X_0[0,1,2,5,6,7,8,10,11,12,13,15]$ and $Y_0[0,1,2,5,6,7,8,10,11,12,13,15]$.

   Since $STK_0[0,1,2,5,6,7,8,10,11,12,13,15] = X_0[0,1,2,5,6,7,8,10,11,12,13,15] \oplus Y_0[0,1,2,5,6,7,8,10,11,12,13,15]$, we can get 12-bit information of the 12 nibbles tweakey information.

   From $Y_0[0,1,2,5,6,7,8,10,11,12,13,15]$, we can deduce the value of $W_0[2,8,13]$, together with the value of $X_1[2,8,13]$, we can get the value of $STK_1[2,8,13]$.

4. To recover the 2^8-bit of 11-nibble tweakey information of STK_9 and STK_10, the guessing and deducing process is totally same with Step. 4 as the differential of $R8$ and $R9$ in Fig. 9 is same with the differential of $R7$ and $R8$ in Fig. 7.
5. Once there are pairs left, the guessing values are wrong. Eliminate all wrong tweakey values and exhaustively search the left key bits and recover the whole tweakey.

Complexity Computation:

In total, we can deduce $12+3+9+2=26$ nibbles, 104 bits information of the tweakey. As we guess 2^{20} values of $(\Delta W_0[2,8,13], \Delta Y_8[14,7])$, each pair can eliminate 2^{20} values of the 104-bit guessed tweakey information. Also, to balance the time and data complexity, $2^{104} \times (1-2^{20}/2^{104})^{2^{68+n}} \ll 1$, we choose $n=23$.

The data complexity is $2^{48+23} = 2^{71}$ plaintexts. The time complexity of step (1) for encrypting the plaintexts is $2\cdot 2^{48+23} = 2^{72}$. The total number of guesses is $2^{68+n+20} = 2^{111}$, which is equivalent to $2^{111}\cdot(12/16+3/16+2/16+9/16)\cdot 1/10\cdot 2 \approx 2^{109.5}$ 10-round encryptions. Thus the time complexity is approximately $2^{109.5}$ 10-round encryptions.

5 Conclusion

We propose an algorithm that can derive longer related-tweakey/key impossible differentials from single-key ones. By utilizing the MILP technique, we convert this algorithm into three MILP models and propose an automatic tool for searching relate-tweakey/key impossible differentials.

The analysis results of QARMA-64 and Joltik-BC-128 give proofs of the validity of this tool. In fact, the results of these two block ciphers are both best results as far as we know. What's more, we can apply this method to more ciphers with similar key schedules, for example, SKINNY and Deoxys-BC.

We welcome other researcher using this method to analyze other ciphers and give improvements.

Appendix

Considering the tweakey schedule,

$$TK_6^1[i] = TK_5^1[i] = TK_1^1[i],$$
$$TK_6^2[i] = 32 \cdot TK_1^2[i],$$
$$TK_5^2[i] = 16 \cdot TK_1^2[i].$$

As we set the MC in $R4$ a 3-to-2 transformation, then:

$$13 \cdot \mathit{\Delta} W_4[0] \oplus 9 \cdot \mathit{\Delta} W_4[1] = 13 \cdot \mathit{\Delta} STK_5[0] \oplus 9 \cdot \mathit{\Delta} STK_4[1] = 0,$$
$$STK_5[0] = TK_5^1[0] \oplus TK_5^2[0],$$
$$STK_5[1] = TK_5^1[1] \oplus TK_5^2[1].$$

STK_6 is inactive in all 16 nibbles, so:

$$\mathit{\Delta} TK_1^1[8] = 32 \cdot \mathit{\Delta} TK_1^2[8],$$
$$\mathit{\Delta} TK_1^1[9] = 32 \cdot \mathit{\Delta} TK_1^2[9].$$

Combing all above equations, we get:

$$13 \cdot \mathit{\Delta} TK_1[8] \oplus 9 \cdot \mathit{\Delta} TK_2[9] = 0.$$

Then, as $\mathit{\Delta} W_0[8,9] = \mathit{\Delta} STK_1[8,9]$, after a $\overline{MC}$ operation, as shown in Fig. 8, the third column of Z_1 will active only in the first three nibbles.

References

1. Mouha, N., Wang, Q., Gu, D., Preneel, B.: Differential and linear cryptanalysis using mixed-integer linear programming. In: International Conference on Information Security and Cryptology, Springer (2011) 57–76
2. Sun, S., Hu, L., Wang, P., Qiao, K., Ma, X., Song, L.: Automatic security evaluation and (related-key) differential characteristic search: application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers. In: International Conference on the Theory and Application of Cryptology and Information Security, Springer (2014) 158–178
3. http://www.sagemath.org/.
4. Xiang, Z., Zhang, W., Bao, Z., Lin, D.: Applying MILP method to searching integral distinguishers based on division property for 6 lightweight block ciphers. In: Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I. (2016) 648–678
5. Sasaki, Y., Todo, Y.: New impossible differential search tool from design and cryptanalysis aspects - revealing structural properties of several ciphers. In: Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III. (2017) 185–215

6. Cui, T., Jia, K., Fu, K., Chen, S., Wang, M.: New automatic search tool for impossible differentials and zero-correlation linear approximations. IACR Cryptology ePrint Archive **2016** (2016) 689
7. Fu, K., Wang, M., Guo, Y., Sun, S., Hu, L.: MILP-based automatic search algorithms for differential and linear trails for speck. In: Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers. (2016) 268–288
8. Li, Z., Bi, W., Dong, X., Wang, X.: Improved conditional cube attacks on Keccak keyed modes with MILP method. Cryptology ePrint Archive, Report 2017/804 (2017) http://eprint.iacr.org/2017/804.
9. Song, L., Guo, J., Shi, D.: New MILP modeling: Improved conditional cube attacks to keccak-based constructions. Cryptology ePrint Archive, Report 2017/1030 (2017) http://eprint.iacr.org/2017/1030.
10. Berton, G., Daemen, J., Peeters, M., Assche, G.V.: The Keccak sponge function family http://keccak.noekeon.org/.
11. Jean, J., Nikolic, I., Peyrin, T.: Tweaks and keys for block ciphers: The TWEAKEY framework. In: 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. (2014) 274–288
12. 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: Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II. (2016) 123–153
13. Cui, T., Jia, K., Fu, K., Chen, S., Wang, M.: New automatic search tool for impossible differentials and zero-correlation linear approximations. Cryptology ePrint Archive, Report 2016/689 (2016) https://eprint.iacr.org/2016/689.
14. Avanzi, R.: The qarma block cipher family. almost mds matrices over rings with zero divisors, nearly symmetric even-mansour constructions with non-involutory central rounds, and search heuristics for low-latency s-boxes. IACR Transactions on Symmetric Cryptology **2017**(1) (2017) 4–44
15. Derbez, P., Fouque, P., Jean, J.: Improved key recovery attacks on reduced-round AES in the single-key setting. In: Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings. (2013) 371–387
16. NIST: Aes: the advanced encryption standard
17. Jean, J., Nikolic, I., Peyrin, T.: Joltik v1.3. In: The CAESAR Competition
18. Shibutani, K., Isobe, T., Hiwatari, H., Mitsuda, A., Akishita, T., Shirai, T.: Piccolo: An ultra-lightweight blockcipher. In: Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings. (2011) 342–357