Related-Tweak Statistical Saturation Cryptanalysis and Its Application on QARMA

Muzhou Li, Kai Hu and Meiqin Wang[*]

Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education,
Shandong University, China

Abstract. Statistical saturation attack takes advantage of a set of plaintext with some bits fixed while the others vary randomly, and then track the evolution of a non-uniform plaintext distribution through the cipher. Previous statistical saturation attacks are all implemented under single-key setting, and there is no public attack models under related-key/tweak setting. In this paper, we propose a new cryptanalytic method which can be seen as related-key/tweak statistical saturation attack by revealing the link between the related-key/tweak statistical saturation distinguishers and KDIB (Key Difference Invariant Bias) / TDIB (Tweak Difference Invariant Bias) ones. KDIB cryptanalysis was proposed by Bogdanov *et al.* at ASIACRYPT'13 and utilizes the property that there can exist linear trails such that their biases are deterministically invariant under key difference. And this method can be easily extended to TDIB distinguishers if the tweak is also alternated. The link between them provides a new and more efficient way to find related-key/tweak statistical saturation distinguishers in ciphers. Thereafter, an automatic searching algorithm for KDIB/TDIB distinguishers is also given in this paper, which can be implemented to find word-level KDIB distinguishers for S-box based key-alternating ciphers. We apply this algorithm to QARMA-64 and give related-tweak statistical saturation attack for 10-round QARMA-64 with outer whitening key. Besides, an 11-round attack on QARMA-128 is also given based on the TDIB technique. Compared with previous public attacks on QARMA including outer whitening key, all attacks presented in this paper are the best ones in terms of the number of rounds.

Keywords: Related-Tweak Statistical Saturation · KDIB · Conditional Equivalence · QARMA

1 Introduction

Linear cryptanalysis [Mat93], proposed by Matsui at Eurocrypt'93, has been playing an important role in evaluating the security of block ciphers. Since then, many interesting results in this area have been introduced including correlation matrices [DGV94], multiple linear cryptanalysis [KR94], linear hull effect [Nyb94], multidimensional cryptanalysis [HCN08], zero-correlation cryptanalysis [BR14] and its extensions [BLNW12, BW12, SCW18].

The basis of linear cryptanalysis is a linear approximation of a given block cipher H. If the linear approximation holds with probability p, then the value $p - \frac{1}{2}$ is called its bias ε. Since the probability of the linear approximation is related to the value of user-supplied key κ used in the target cipher, the bias ε is dependent on κ. However, the entire linear hull is notoriously difficult to analyze for the immense number of linear trails comprising it. In [BBR$^+$13], Bogdanov *et al.* introduced a way to analyze the entire linear hull for key alternating ciphers by utilizing the property that the bias of a linear hull can be actually

[*] Corresponding author, mqwang@sdu.edu.cn

Licensed under Creative Commons License CC-BY 4.0.

IACR Transactions ISSN XXXX-XXXX, Vol. 0, No. 0, pp. 1–27
DOI:XXXXXXXX

invariant under the modification of key. By looking at the composition of the fixed-key linear hull from individual trails, they derive a sufficient condition on linear trails and the keys such that the bias remains unaffected by a change of key. The technique proposed by them is called the key difference invariant bias, or KDIB cryptanalysis for short. One thing we have to remark is that this cryptanalytic method can be extended into TDIB (*tweak difference invariant bias*) attack for block ciphers with tweak alternated, since the tweak can be seen as a kind of key and has the same effect on the bias of linear hull.

Integral cryptanalysis is another important cryptanalytic technique for block ciphers, which was firstly introduced by Daemen *et al.* as a dedicated attack against Square cipher [DKR97]. Later, Knudsen and Wagner unified it as integral attack [KW02], which also known as saturation attacks [HLL$^+$02]. To reduce data complexity, statistical integral attack was proposed in FSE'16 [WCC$^+$16]. All these attacks exploit the propagation of well chosen sets of plaintexts through the cipher. In practice, they often fix a part of plaintext bits to some constant value, and then track the evolution of the variable bits in the cipher state. In [DEM16], Dobraunig *et al.* proposed a related-tweak Square attack on KIASU-BC that extends the single-key attack by one round.

Statistical saturation attack is different from integral attack, as proposed by Collard and Standaert in [CS09]. It also takes advantage of a set of plaintext with some bits fixed while the others vary randomly, but track the evolution of a non-uniform plaintext distribution through the cipher. However, the current statistical saturation attack can only work under single-key/tweak settings and there is no public attack models under related-key/tweak setting. In this paper, we will propose a new cryptanalytic method which actually is related-key/tweak statistical saturation attack. For the related-key/tweak statistical saturation distinguisher, if we fix a part of the plaintext and take all possible values for the other plaintext bits, then the relation between the distribution of a part of the ciphertext value under related-key/tweak pairs will be considered.

The contributions of this paper are shown as follows.

Related-Tweak Statistical Saturation Distinguisher and its Link with TDIB. In Sect. 3, we introduce this new cryptanalytic method, where one fixes a part of the plaintext and takes all possible values for the other plaintext bits and then considers the value distribution of a part of ciphertext under related-key/tweak pairs (z, z'). To obtain this related-key/tweak invariant distribution, we reveal the conditional equivalent property between KDIB/TDIB and related-key/tweak statistical saturation attack. This equivalent property demonstrated that if the bias under z equals to that under z' for all possible input and output mask pairs contained in the KDIB/TDIB distinguisher, then one can obtain a related-key/tweak statistical saturation one. On the other hand, a related-key/tweak statistical saturation distinguisher can derive a KDIB/TDIB distinguisher. More precisely, consider a KDIB/TDIB distinguisher for an n-bit block cipher where (without loss of generality) each composed linear hull has non-zero input mask with zeros in the last s bits and non-zero output mask with zeros in the last $n - t$ bits, and the bias is invariant under different z and z'. We prove that this setting is equivalent to a related-key/tweak statistical saturation distinguisher where fixing the first $n - s$ bits in the input leads to identical distribution for the first t bits output under different z and z'.

Automatically Searching for KDIB Distinguishers for Key-Alternating Ciphers. Automatic tools have been playing a more and more important role in the design and cryptanalysis of symmetric ciphers. In recent years, algorithms to search distinguishers for ciphers with STP have been proposed [KLT15, LWR16, MP13]. Seeing that the known KDIB cryptanalysis has only been utilized to attack word-level key-alternating ciphers with S-boxes, such as LBlock [WZ11] and TWINE [SMMK12], we introduce an algorithm in Sect. 4, which can be implemented to search word-level KDIB distinguishers for S-box based key-alternating ciphers. Notice that this algorithm can also be used to search for TDIB distinguishers seeing that tweak can be seen as a kind of key. With this algorithm, we can obtain 8-round TDIB distinguishers for both versions of QARMA illustrated in Sect. 5.1, which are transformed into related-tweak statistical saturation distinguishers in Sect. 5.2.

Related-Tweak Statistical Saturation and TDIB Attacks on QARMA. QARMA [Ava17] is a family of lightweight tweakable block ciphers designed by Avanzi at ToSC'17. It supports block sizes with 64 and 128 bits, denoted as QARMA-64 and QARMA-128, separately.

Since its proposal, there have been several attacks such as meet-in-the-middle attacks [LJ18, ZD16] and impossible differential attacks [YQC18, ZDW18]. In [YQC18], Yang *et al.* proposed single-key single-tweak impossible differential attacks on 10/11-round QARMA-64 and -128. Unfortunately, their attacks are all invalid ones since the complexity of them are beyond the designer's security claims that the multiplication of time and data complexity for QARMA-64 and -128 should be less than $2^{128-\epsilon}$ and $2^{256-\epsilon}$ for a small ϵ (*e.g.* 2), separately. Besides, attacks proposed in [ZD16] and [ZDW18] didn't consider outer whitening key. According to the number of rounds, the best known valid attack considering outer whitening key can work on 9-round QARMA-64 and 10-round QARMA-128 [LJ18].

We mount related-tweak statistical saturation attacks on 10-round QARMA-64 in Sect. 6.1. Besides, a key recovery attack on 11-round QARMA-128 utilizing those 8-round TDIB distinguishers is proposed in Sect. 6.2 based on the TDIB cryptanalysis. In fact, we found that the complexity of TDIB attack on 10 rounds QARMA-64 is higher than that of related-tweak statistical saturation attack. On the other hand, the related-tweak statistical saturation attack on 11-round QARMA-128 has higher complexity than the TDIB attack. It means that the results of key recovery attacks based on the equivalent TDIB and related-tweak statistical saturation distinguisher are very different. Therefore, the proposition of related-tweak statistical saturation distinguisher provides an additional cryptanalytic method to evaluate the security of block ciphers. All our results are presented in Table 1 along with those introduced in [LJ18]. From Table 1, our attacks for both versions of QARMA are the best ones considering outer whitening key according to the number of rounds and they all satisfy the security claim.

Table 1: Summary of Attacks on Reduced-Round QARMA with Outer Whitening Key

Block	Attacks	Rounds	Data	Time*	Memory	#*tks*	Reference
64	MITM	8	2^{16} CPT	2^{33}	2^{89} 64-bit	1	[LJ18]
	MITM	9	2^{16} CPT	2^{48}	2^{89} 64-bit	1	[LJ18]
	RT SS	10	2^{59} CPT	2^{59}	$2^{29.6}$ bits	8	Sect. 6.1
128	MITM	10	2^{88} CPT	2^{156}	2^{145} 128-bit	1	[LJ18]
	TDIB	11	$2^{126.1}$ KPT	$2^{126.1}$	2^{71} bits	4	Sect. 6.2

MITM: Meet-in-the-Middle; RT SS: Related-Tweak Statistical Saturation.
CPT/KPT: Chosen/Known Plaintext-Tweak Pairs.
#*tks*: the number of different tweaks used in the corresponding attack.
* Evaluated by encryption units.

2 Preliminaries

2.1 Key Difference Invariant Bias in Key-Alternating Ciphers

Daemen and Rijmen proposed the concept of key-alternating cipher in [DR02], which forms a special but important subset of the modern block ciphers. Many block ciphers can be classified into this set, like almost all SPN ciphers and some Feistel ciphers. Here we restate this conception as follows.

Definition 1. ***(Key-Alternating Block Cipher [DR02])*** *Let k_i represent the n-bit round key in round i of an iterative block cipher with $1 \leq i \leq r$. The block cipher is key-alternating, if k_i is XORed into the state at the end of the i-th round. And there also exists a subkey k_0 which is introduced by XORing with the plaintext before the first round.*

A linear approximation of iterative ciphers (*e.g.* key-alternating block ciphers) is called a linear hull [Nyb94]. A linear hull (Γ, Λ) consists of all possible linear trails with input mask Γ and output mask Λ. And it is said to be *trivial* if either Γ or Λ is zero. Otherwise, it is *non-trivial.* Assuming that there is a linear trail θ of an r-round iterative block cipher, the input mask of round i is θ_{i-1} and the output mask is θ_i with $1 \leq i \leq r$. Then we can denote the trail by a $n(r+1)$ bits column vector $\theta = (\theta_0, \theta_1, \ldots, \theta_r)$. The linear hull (Γ, Λ) contains all θ which satisfy $\theta_0 = \Gamma$ and $\theta_r = \Lambda$.

Denote $\mathbb{F}_2$ as the field with two elements $\{0, 1\}$ and $\mathbb{F}_2^n$ as the space of n-dimensional binary vectors over $\mathbb{F}_2$. The inner product of binary vectors is $\Gamma \cdot x = \oplus_{j=0}^{n-1} \Gamma_j \cdot x_j$ with x_0 be the rightmost bit of x, and the bias of the i-th round can be defined as

$$\varepsilon_{\theta_{i-1},\theta_i} = Pr[\theta_{i-1} \cdot x \oplus \theta_i \cdot f(x) = 0] - \frac{1}{2},$$

where $f : \mathbb{F}_2^n \to \mathbb{F}_2^n$ represents the round function. And then the bias of the linear trail θ under κ for key-alternating cipher is

$$\varepsilon_\theta(\kappa) = 2^{r-1}(-1)^{\theta_0 \cdot k_0} \prod_{i=1}^{r} (-1)^{\theta_i \cdot k_i} \varepsilon_{\theta_{i-1},\theta_i}.$$

For key-alternating cipher, the bias ε of a linear hull can be computed if we can know all biases of linear trails comprising the linear hull with the condition that they are estimated under the same fixed key value.

Proposition 1. ***([DR02])*** *For a key-alternating cipher, the bias ε of a non-trivial linear hull (Γ, Λ) under the user-supplied key κ is*

$$\varepsilon(\kappa) = \sum_{\theta:\theta_0=\Gamma,\theta_r=\Lambda} \varepsilon_\theta(\kappa) = \sum_{\theta:\theta_0=\Gamma,\theta_r=\Lambda} (-1)^{\theta^t \cdot K} \varepsilon_\theta(0) = \sum_{\theta:\theta_0=\Gamma,\theta_r=\Lambda} (-1)^{d_\theta + \theta^t \cdot K} |\varepsilon_\theta|,$$

where $\varepsilon_\theta(\kappa)$ is the bias of the linear trail θ under κ, $|\varepsilon_\theta|$ is the absolute value of $\varepsilon_\theta(0)$ with $d_\theta \in \{0, 1\}$ as its sign. And K is a $n(r+1)$ bits column vector $(k_0, k_1, \ldots, k_r)$ derived by κ using the key schedule.

But the truth is that we cannot know all biases of linear trails in the linear hull due to their high number. To fully utilizing the entire linear hull for key-alternating ciphers, Bogdanov *et al.* proposed the key difference invariant bias technique, or KDIB cryptanalysis, due to the fact that *the bias of a linear hull can be actually invariant under the modification of key.* Their main result is shown as follows.

Proposition 2. ***(KDIB Condition, [BBR+13], Theorem 1)*** *Let (Γ, Λ) be a non-trivial linear hull of a key-alternating cipher. Then $\varepsilon(\kappa) = \varepsilon(\kappa')$ if $\theta^t \cdot K = \theta^t \cdot K'$ holds for all θ with $\varepsilon_\theta \neq 0$ in the linear hull.*

To find linear hulls with corresponding key difference $\Delta = K \oplus K'$ satisfying the KDIB condition[1], they proposed a sufficient condition of it. Let $\theta(j)$ be the j-th bit of the column vector θ. If $\theta(j) = 1$, the j-th bit of Δ is restricted to be zero. Otherwise, the j-th bit of Δ can be 0 or 1. Thus, we can assure that the condition $\theta^t \cdot K = \theta^t \cdot K'$ holds for every θ in the linear hull[2].

[1] To simplify notation, we call this the KDIB distinguisher.

[2] Obviously, the condition holds for any θ if $K = K'$. Since this is useless to our key recovery attack, we will require that $K \neq K'$.

Suppose that we have obtained an r-round KDIB distinguisher comprised of λ non-trivial linear hulls, where λ is high enough, we can use it to mount a key recovery attack as follows. At first, we collect N plaintext-ciphertext pairs (P, C) under the user-supplied key κ and another N pairs (P', C') under κ', where κ and κ' satisfies $K \oplus K' = \Delta$. Secondly, partial state value x and x' covered by these linear hulls can be obtained respectively after guessing corresponding key bits. After that, for each linear hull, we compute S_i and S'_i with $1 \leq i \leq \lambda$ to record the total number of times x and x' satisfies this linear hull among all these N pairs, separately. And then we compute the statistic

$$s = \sum_{i=1}^{\lambda} \left[\left(\frac{S_i}{N} - \frac{1}{2} \right) - \left(\frac{S'_i}{N} - \frac{1}{2} \right) \right]^2 .$$

Finally, if the value of s is larger than some threshold s_τ, we'll discard the corresponding key and choose a different one to do this again. Otherwise, we will accept it and check exhaustively all the possible keys by utilizing several plaintext-ciphertext pairs.

Proposition 3. ***([BBR+13], Subsection 4.1)*** *Assuming that one have obtained a KDIB distinguisher for a key-alternating block cipher which contains λ non-trivial linear hulls under the same fixed key difference Δ. Denote α_0 as the probability to reject the right key and α_1 as the probability to accept a wrong key. For sufficiently large N and λ, the data complexity N is*

$$N = \frac{2^{n+0.5}}{\sqrt{\lambda} - q_{1-\alpha_1}\sqrt{2}} (q_{1-\alpha_0} + q_{1-\alpha_1}),$$

and the decision threshold s_τ is

$$s_\tau = \frac{\sqrt{\lambda}}{N\sqrt{2}} q_{1-\alpha_0} + \frac{\lambda}{2N},$$

where $q_{1-\alpha_0}$ and $q_{1-\alpha_1}$ represent the lower quantiles of the standard normal distribution $\mathcal{N}(0, 1)$, respectively.

At the last part of this subsection, we have to mention that the KDIB cryptanalysis proposed for key-alternating ciphers can be simply extended to TDIB or TKDIB (*tweak or tweakey difference invariant bias*) attack for block ciphers with tweak or tweakey alternated, since the tweak or tweakey can be seen as a kind of key and has the same effect on the bias of linear hull. In order to mount TDIB or TKDIB attacks, we only have to replace the key with the tweak or tweakey in Proposition 2. Since methods proposed for TDIB attack can be easily applied to TKDIB attack, we only use the notation of TDIB in the rest part of our paper to simplify our description.

2.2 Brief Description of QARMA

QARMA block cipher [Ava17] is a family of lightweight tweakable block ciphers. It supports two kinds of block sizes with $n = 64$ and $n = 128$, denoted as QARMA-64 and QARMA-128, respectively. And the corresponding size of tweak is equal to n, while the key has $2n$ bits. Its structure is described in Figure 1, which implies that it belongs to the class of key-alternating SPN ciphers.

QARMA-64 is a 14-round block cipher with a central construction composed of two central rounds and a *Pseudo-Reflector* construction, while QARMA-128 has 22 rounds with a same central function. All n-bit values can be represented as arrays of 16 m-bit cells or 4×4 matrices, *i.e.*,

$$IS = s_0||s_1||s_2||\cdots||s_{15} = \begin{bmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{bmatrix},$$

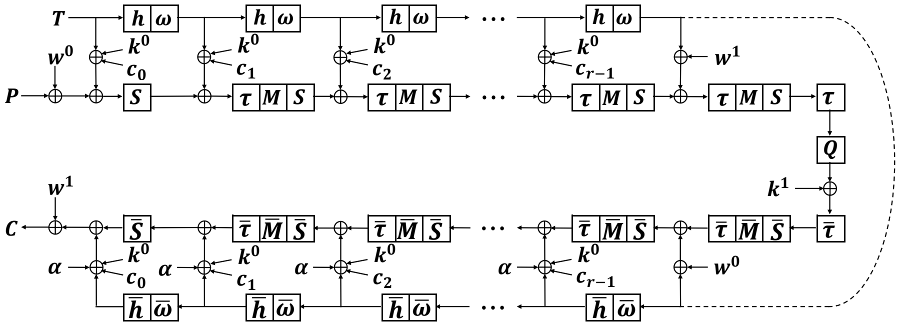

Figure 1: The Structure of $(2r+2)$-Round QARMA

so that 4×4 matrices operate column-wise on these values by left multiplication.

The $2n$-bit key is separated into two parts $w^0||k^0$, where w^0 and k^0, the whitening and core keys, have the same length. And we have $w^1 = o(w^0) = (w^0 \ggg 1) \oplus (w^0 \gg (n-1))$ and $k^1 = k^0$. The tweak update function includes two operations h and ω. h is a permutation $h(T) = t_{h(0)}||t_{h(1)}|| \cdots ||t_{h(15)}$ with $h = [6, 5, 14, 15, 0, 1, 2, 3, 7, 12, 13, 4, 8, 9, 10, 11]$. And ω is a LFSR updating cells with index 0, 1, 3, 4, 8, 11 and 13. For QARMA-64, it maps (b_3, b_2, b_1, b_0) to $(b_0 \oplus b_1, b_3, b_2, b_1)$. But for QARMA-128, it maps $(b_7, b_6, \ldots, b_0)$ to $(b_0 \oplus b_2, b_7, b_6, \ldots, b_0)$. As shown in Figure 1, the round tweakey is the XORed value of core key, round tweak and some constants.

Every forward round function except for the first round, which only consists of AddRoundTweakey and SubCells(S), is composed by four operations: AddRoundTweakey, ShuffleCells(τ), MixColumns(M) and SubCells(S). The operation τ is same for both kinds of QARMA, and $(\tau(IS))_i = s_{\tau(i)}$ holds for $0 \leq i \leq 15$ with $\tau = [0, 11, 6, 13, 10, 1, 12, 7, 5, 14, 3, 8, 15, 4, 9, 2]$. Denote this following matrix by $circ(0, \rho^a, \rho^b, \rho^c)$:

$$\begin{bmatrix} 0 & \rho^a & \rho^b & \rho^c \\ \rho^c & 0 & \rho^a & \rho^b \\ \rho^b & \rho^c & 0 & \rho^a \\ \rho^a & \rho^b & \rho^c & 0 \end{bmatrix},$$

then the matrix M used in QARMA-64 and QARMA-128 can be represented by $circ(0, \rho, \rho^2, \rho)$ and $circ(0, \rho, \rho^4, \rho^5)$, respectively. The multiplication of an element in IS with ρ^i is just a simple left circular rotation of the element by i bits. And the i-th column of internal state after MixColumns is the corresponding column of $M \cdot IS$. The backward round function is totally the inverse of the forward round function. Therefore, we omit it here. The *Pseudo-Reflector* construction contains four operations which are τ, a matrix multiplication(Q), AddRoundTweakey and the inverse of τ. In both versions of QARMA, we have $Q = M$.

3 Related-Tweak Statistical Saturation Cryptanalysis

In this section, we start from KDIB and TDIB distinguishers to respectively convert them into related-key and related-tweak statistical saturation ones. And the converting method for KDIB distinguishers has nothing different with the one used for TDIB distinguishers, which can be realized below. Therefore, we only focus on how to covert TDIB distinguishers into related-tweak statistical saturation ones since we will utilize these distinguishers to attack QARMA.

Related-tweak statistical saturation cryptanalysis (Related-tweak SS) fixes a part of the plaintext and takes all possible values for the other plaintext bits, and then considers the distribution of a part of the ciphertext value under related-tweak pairs (z, z'), where $z' = z \oplus \Delta$ and Δ is a fixed value for all possible values of z. Our result shows that the distribution of a part of the ciphertext value encrypted under z can be the same as the one obtained under z' if the bias under z is equal to that under z' for all possible linear trails of the linear hull in the TDIB distinguisher (See Theorem 1 for details.). This method can be regarded as an extension of statistical saturation cryptanalysis in the related-tweak setting.

To make it clear, we denote $H : \mathbb{F}_2^n \times \mathbb{F}_2^k \to \mathbb{F}_2^n$ as the target block cipher with block size n and tweak size k. And then we split the input of H into two parts (x, y), where x is the part fixed during our attack and y is the part taking all possible values. Similarly, the output of H is also divided into two parts $(H_1(x, y, z), H_2(x, y, z))$ and we only focus on the value distribution of $H_1(x, y, z)$. So we have

$$H : \mathbb{F}_2^r \times \mathbb{F}_2^s \times \mathbb{F}_2^k \to \mathbb{F}_2^t \times \mathbb{F}_2^u,\ H(x, y, z) = (H_1(x, y, z), H_2(x, y, z)).$$

The function T_I defined by

$$T_I : \mathbb{F}_2^s \times \mathbb{F}_2^k \to \mathbb{F}_2^t,\ T_I(y, z) = H_1(I, y, z)$$

is actually the function H when the r bits in the first part of its input are fixed to I and only the t bits in the first part of its output are taken into account.

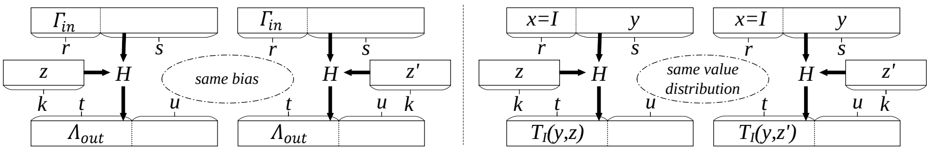

Figure 2: Equivalent between TDIB and Related-Tweak SS

Using these above notations, we introduce the conditional equivalent property between TDIB and related-tweak statistical saturation distinguisher as follows.

Theorem 1. *Let (Γ, Λ) be the linear hull of the target block cipher with $\Gamma = (\Gamma_{in}, 0)$ and $\Lambda = (\Lambda_{out}, 0)$, where $\Gamma_{in} \in \mathbb{F}_2^r$ and $\Lambda_{out} \in \mathbb{F}_2^t \backslash \{0\}$. Given a fixed Δ, if the bias is invariant under related-tweak pairs $(z, z' = z \oplus \Delta)$ for all possible mask pairs $(\Gamma_{in}, \Lambda_{out})$, then $T_I(y, z)$ has the same value distribution with $T_I(y, z')$ and vice versa, i.e., for any $I \in \mathbb{F}_2^r$, if one fixes x as $I \in \mathbb{F}_2^r$, and takes all possible values for y, then we have*

$$\#\{y \in \mathbb{F}_2^s \mid T_I(y, z) = c\} = \#\{y \in \mathbb{F}_2^s \mid T_I(y, z') = c\}$$

for any $c \in \mathbb{F}_2^t$.

To prove this theorem, we have to recall the theory of multidimensional linear cryptanalysis [HCN08].

If X is a random variable in $\mathbb{F}_2^m$, the probability distribution $p = (p_0, p_1, \ldots, p_{2^m-1})$ of X means that the probability that X takes value η is p_η, where $\eta \in \mathbb{F}_2^m$. The bias of the linear hull (Γ, Λ) for the block cipher H under the tweak z is

$$\varepsilon(z) = Pr[\Gamma \cdot (x||y) \oplus \Lambda \cdot H(x, y, z) = 0] - \frac{1}{2},$$

where the probability is taken over all choices of inputs $x||y$. And then the correlation of the linear hull can be represented as $Cor_z(\Gamma, \Lambda) = 2\varepsilon(z)$.

The function $f = (f_1, f_2, \ldots, f_n) : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is called a vectorial Boolean function, where $f_i : \mathbb{F}_2^n \to \mathbb{F}_2$ is a Boolean function. For a fixed tweak z, H can be seen as a vectorial Boolean function from $\mathbb{F}_2^n$ to itself. Suppose that there are m linearly independent binary mask pairs (α_i, β_i), $i = 0, 1, \ldots, m-1$. For each mask pair, there is one linear approximation g_i^z for H, where g_i^z is denoted as

$$g_i^z(x, y) = \alpha_i \cdot (x||y) \oplus \beta_i \cdot H(x, y, z).$$

The m independent linear approximations form the base linear approximations. Let $Cor(g_i^z)$ be the correlation of g_i^z and $g^z = (g_0^z, g_1^z, \ldots, g_{m-1}^z)$ be the target m-dimensional value, which is a vectorial Boolean function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$ for a fixed tweak z.

Let $a \in \mathbb{F}_2^m$ be a combined mask and the correlation of the combined linear approximation $a \cdot g^z$ is denoted as $Cor(a \cdot g^z)$. Suppose that the probability distribution of g^z is $p^z = (p_0^z, p_1^z, \ldots, p_{2^m-1}^z)$, the following corollary introduced in [HCN08] gives the relation between the probability p_η^z and the correlations for all 2^m linear approximations.

Corollary 1. *([HCN08]) Let $g^z : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a vectorial Boolean function with probability distribution $p^z = (p_0^z, p_1^z, \ldots, p_{2^m-1}^z)$. Then there is*

$$p_\eta^z = 2^{-m} \sum_{a \in \mathbb{F}_2^m} (-1)^{a \cdot \eta} Cor(a \cdot g^z), \forall \eta \in \mathbb{F}_2^m.$$

By applying the inverse Walsh-Hadamard transform to the above equality, we can achieve another corollary.

Corollary 2. *Let $g^z : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a vectorial Boolean function with probability distribution $p^z = (p_0^z, p_1^z, \ldots, p_{2^m-1}^z)$. Then there is*

$$Cor(a \cdot g^z) = \sum_{\eta \in \mathbb{F}_2^m} (-1)^{a \cdot \eta} p_\eta^z, \forall a \in \mathbb{F}_2^m.$$

Proof. By using Corollary 1, we can find that

$$\begin{aligned}\sum_{\eta \in \mathbb{F}_2^m} (-1)^{a \cdot \eta} p_\eta^z &= \sum_{\eta \in \mathbb{F}_2^m} (-1)^{a \cdot \eta} \left(2^{-m} \sum_{a' \in \mathbb{F}_2^m} (-1)^{a' \cdot \eta} Cor(a' \cdot g^z) \right) \\ &= 2^{-m} \sum_{\eta \in \mathbb{F}_2^m} \left(\sum_{a' \in \mathbb{F}_2^m} (-1)^{(a' \oplus a) \cdot \eta} Cor(a' \cdot g^z) \right) \\ &= Cor(a \cdot g^z) + \sum_{a' \neq a} Cor(a' \cdot g^z) \left(\sum_{\eta \in \mathbb{F}_2^m} (-1)^{(a' \oplus a) \cdot \eta} \right) \\ &= Cor(a \cdot g^z)\end{aligned}$$

holds for any $a \in \mathbb{F}_2^m$. □

Following these two corollaries, we can prove our Theorem 1 in the following way.

Proof. Denote the concatenation value $\Gamma_{in}||\Lambda_{out}$ as V, then $V \in \mathbb{F}_2^{r+t}$. Let $V^i = \Gamma_{in}^i||\Lambda_{out}^i$ be unit vector $(0 \ldots 010 \ldots 0)$ with 1 in the i-th position, where $0 \leq i \leq r+t-1$. Then these $(r+t)$ V^i are independent with each other. For each mask pair $(\Gamma_{in}^i, \Lambda_{out}^i)$, there is a linear approximation g_i^z for the target block cipher H, where g_i^z is

$$g_i^z(x, y) = \Gamma_{in}^i \cdot x \oplus \Lambda_{out}^i \cdot H_1(x, y, z).$$

Hence, $(r+t)$ g_i^z consist of the base linear approximations, which implies that $a \cdot g^z$ with $a \in \mathbb{F}_2^{r+t} \backslash \{0\}$ contains all the possible mask pairs $(\Gamma_{in}, \Lambda_{out})$. Recall that $V^i = \Gamma_{in}^i||\Lambda_{out}^i$

is the unit vector, then we have $g_0^z(x,y) = H_1(x,y,z)_0$, $g_1^z(x,y) = H_1(x,y,z)_1$, …, $g_{t-1}^z(x,y) = H_1(x,y,z)_{t-1}$, $g_t^z(x,y) = x_0$, $g_{t+1}^z(x,y) = x_1$, …, $g_{t+r-1}^z(x,y) = x_{r-1}$, where $H_1(x,y,z)_i$ represents the i-th bit of $H_1(x,y,z)$.

Since $\varepsilon(z) = \varepsilon(z')$ holds for all possible mask pairs $(\Gamma_{in}, \Lambda_{out})$, we know that

$$Cor(a \cdot g^z) = Cor(a \cdot g^{z'}), \forall a \in \mathbb{F}_2^{r+t} \backslash \{0\}.$$

Let $p^z = (p_0^z, p_1^z, \dots, p_{2^m-1}^z)$ represent the probability distribution of g^z. Then we have

$$2^{r+t} p_\eta^z - 1 = \sum_{a \in \mathbb{F}_2^{r+t} \backslash \{0\}} (-1)^{a \cdot \eta} Cor(a \cdot g^z), \forall \eta \in \mathbb{F}_2^{r+t}$$

according to Corollary 1. Therefore, $p_\eta^z = p_\eta^{z'}$ holds for any $\eta \in \mathbb{F}_2^{r+t}$.

In terms of the definition of g_i^z, we can obtain

$$\begin{aligned} p_\eta^z &= 2^{-n} \#\{(x,y) \in \mathbb{F}_2^n \mid g^z(x,y) = \eta\} \\ &= 2^{-n} \#\{(x,y) \in \mathbb{F}_2^n \mid g_0^z(x,y) = \eta_0,\ g_1^z(x,y) = \eta_1,\ \dots,\ g_{r+t-1}^z(x,y) = \eta_{r+t-1}\} \\ &= 2^{-n} \#\{(x,y) \in \mathbb{F}_2^n \mid x || H_1(x,y,z) = \eta\} \end{aligned}$$

From $p_\eta^z = p_\eta^{z'}$, we have for any $\eta \in \mathbb{F}_2^{r+t}$,

$$\#\{(x,y) \in \mathbb{F}_2^n \mid x || H_1(x,y,z) = \eta\} = \#\{(x,y) \in \mathbb{F}_2^n \mid x || H_1(x,y,z') = \eta\}.$$

Let $\eta = I || c$ with $I \in \mathbb{F}_2^r$ and $c \in \mathbb{F}_2^t$, then we have

$$\#\{y \in \mathbb{F}_2^s \mid x = I,\ H_1(x,y,z) = c\} = \#\{y \in \mathbb{F}_2^s \mid x = I,\ H_1(x,y,z') = c\}.$$

Hence, for any $I \in \mathbb{F}_2^r$,

$$\#\{y \in \mathbb{F}_2^s \mid T_I(y,z) = c\} = \#\{y \in \mathbb{F}_2^s \mid T_I(y,z') = c\}$$

holds for any $c \in \mathbb{F}_2^t$.

That is to say, if one fixes x to be $I \in \mathbb{F}_2^r$ and takes all possible values for y, then $T_I(y,z)$ has the same value distribution with $T_I(y,z')$.

As for the converse, since $T_I(y,z)$ has the same value distribution with $T_I(y,z')$, we can see that $p_\eta^z = p_\eta^{z'}$ holds for any $\eta \in \mathbb{F}_2^{r+t}$ according to the previous proof. With the help of Corollary 2, we can obtain that $Cor(a \cdot g^z) = Cor(a \cdot g^{z'})$ holds for any $a \in \mathbb{F}_2^{r+t}$. □

One thing we have to mention is that the restriction to masks of the form $(\Gamma_{in}, 0)$ and $(\Lambda_{out}, 0)$, where the last bits are fixed to zero, is solely for the simplicity of notations. And according to the proof, we can see that positions of zero bits will not influence the applicability of our theorem.

Assume that we have obtained a related-tweak statistical saturation distinguisher where $T_I(y,z)$ has the same value distribution with $T_I(y,z')$ if x is fixed to be some $I \in \mathbb{F}_2^r$ and y takes all possible values in $\mathbb{F}_2^s$. We can utilize it to mount a key recovery attack by adding several rounds after it. At first, we choose a set of plaintexts $P = (x,y)$ satisfying that $x = I$ and y takes all possible values in $\mathbb{F}_2^s$. Then we can get two sets of ciphertexts C and C' by encrypting these plaintexts under z and z', separately. After guessing corresponding key bits, we can obtain partial state value $T_I(y,z)$ and $T_I(y,z')$ covered by the distinguisher. If $T_I(y,z)$ and $T_I(y,z')$ have the same value distribution, these guessed key bits will be taken as right key bits. Otherwise, they will be discarded. From Theorem 1, we can see that for right key guess, $T_I(y,z)$ has the same value distribution with $T_I(y,z')$. Hence the probability to reject the right key α_0 is zero. To evaluate the probability of accepting a wrong key α_1, we provide the following theorem.

Theorem 2. *Following Theorem 1, the probability to accept a wrong key fulfills*

$$\log_2(\alpha_1) \le \left(2^t - 1 - t\right) 2^{s+1} - 2^{s(2^t-1)/2}.$$

Proof. Denote $V_c = \#\{y \in \mathbb{F}_2^s \mid T_I(y,z) = c\}$ and $V'_c = \#\{y \in \mathbb{F}_2^s \mid T_I(y,z') = c\}$, where $c \in \mathbb{F}_2^t$. If the guessed key is wrong, V_c and V'_c will be two independent random variables satisfying that $\sum_{c=0}^{2^t-1} V_c = 2^s = \sum_{c=0}^{2^t-1} V'_c$. It follows that the probability to accept a wrong key is

$$\begin{aligned}
\alpha_1 &= Pr[V_0 = V'_0, V_1 = V'_1, \ldots, V_{2^t-1} = V'_{2^t-1}] \\
&= \sum_{x_0+x_1+\cdots+x_{2^t-1}=2^s} \left(Pr[V_0 = x_0, V_1 = x_1, \ldots, V_{2^t-1} = x_{2^t-1}]\right)^2 \\
&= \sum_{x_0+x_1+\cdots+x_{2^t-1}=2^s} \left[\left(\frac{1}{2^t}\right)^{2^s} \binom{2^s}{x_0}\binom{2^s - x_0}{x_1} \cdots \binom{2^s - \sum_{j=0}^{2^t-2} x_j}{x_{2^t-1}}\right]^2 \\
&= \frac{1}{2^{t2^{s+1}}} \sum_{x_0+x_1+\cdots+x_{2^t-1}=2^s} \left[\binom{2^s}{x_0}\binom{2^s - x_0}{x_1} \cdots \binom{2^s - \sum_{j=0}^{2^t-2} x_j}{x_{2^t-1}}\right]^2 .
\end{aligned}$$

According to Lemma 1 and 2 introduced in Appendix B, we have

$$\begin{aligned}
\alpha_1 \le \frac{1}{2^{t2^{s+1}}} \left[\binom{2^{s+1}}{2^s}\right]^{2^t-1} &\approx \frac{1}{2^{t2^{s+1}}} \left(\frac{2^{2^{s+1}}}{\sqrt{\pi 2^s}}\right)^{2^t-1} \\
&= \left(\frac{1}{\sqrt{\pi}}\right)^{2^t-1} 2^{(2^t-1-t)2^{s+1} - (2^{s(2^t-1)/2})} \\
&\le 2^{\left(2^t-1-t\right)2^{s+1} - \left(2^{s(2^t-1)/2}\right)}.
\end{aligned}$$

It follows that $\log_2(\alpha_1) \le (2^t - 1 - t)\, 2^{s+1} - 2^{s(2^t-1)/2}$. □

4 Searching for KDIB Distinguishers with STP

In this section, we will introduce how to find KDIB distinguishers for block ciphers. Like what we pointed out in the last part of Sect. 2.1, one can also find TDIB distinguishers by following the way illustrated in this section. To be simple, we will only introduce how to find KDIB distinguishers here.

For ciphers which have been attacked using KDIB distinguishers such as LBlock [WZ11] and TWINE [SMMK12], we found that this method is suitable for word-level key-alternating ciphers with S-boxes. Hence, we targets at searching word-level KDIB distinguishers for S-box based key-alternating ciphers.

Recently, many cryptanalytic results have been proposed by utilizing various kinds of automatic searching tools. Among all of them, the Boolean Satisfiability Problem (SAT) [Coo71]/Satisfiability Modulo Theories (SMT) problem [BSST09] solver STP[3] has been playing an important role. The application of STP for cryptanalysis was firstly suggested by Mouha and Preneel in [MP13]. It is a decision procedure to confirm whether there is a solution to a set of equations. These equations must follow the rule of input language parsed by STP[4].

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

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

Actually, finding KDIB distinguishers can be converted into an existence problem. *Word-level* mask propagation properties of an operation in the round function and *bit-level* difference propagation properties for the key schedule, which can both be represented by some equations, should be precisely depicted. Considering mask propagation property in word-level, we actually described the propagation of necessary conditions on the family of consistent trails, which means that not all the KDIB distinguishers can be found by utilizing our algorithm. In the original paper of KDIB cryptanalysis [BBR$^+$13], KDIB distinguishers for LBlock and TWINE are derived at bit-level for key and word-level for data. In this way, longer distinguishers could be obtained and that is why we consider the key at bit-level. In addition to these propagation properties, equations representing the condition for KDIB distinguishers are also included. And extra equations, such as those restricting that at least one round key is non-zero, will be included in order to exclude trivial distinguishers. Whether these equations have a solution can directly help us to confirm whether the expected KDIB distinguisher exists.

In practice, if we aim at finding R-round KDIB distinguishers covered by R_1 forward rounds and $R - R_1 = R_2$ backward rounds, then we should describe mask propagation properties operations in the encryption and decryption procedure. Besides, equations describing difference propagation properties for R rounds of the key schedule shall be included, as well as some extra equations. These constraint equations can be divided into four parts. Part 1 contains equations depicting propagation properties between input and output mask of an operation in the round function at word-level. Part 2 is composed of equations describing the difference propagation property of key schedule at bit-level. To make our searching algorithm more general, we also describe the difference propagation property of S-box in this part to cover ciphers containing S-box in their key schedule. And then the propagation of key difference will have probability which leads to weak-key attacks. In Part 3, we describe equations representing the condition for KDIB distinguishers which is illustrated in Proposition 2. The last part, Part 4, comprises some extra but necessary equations.

Part 1. Equations for Basic Operations in Round Function

In this part, we utilize the theta variable to represent the active state of a word. The value of theta variable is 0 means this word isn't active. And theta=1 means that this word is definitely active or potentially active.

Property 1. ***(Substitution)*** *Let S be the S-box used in the round function of the target cipher. The active state of input mask is θ_{in}, and the corresponding active state of output mask is denoted as θ_{out}. Then we have $\theta_{out} = \theta_{in}$.*

Property 2. ***(XOR)*** *Let θ_{in_1} and θ_{in_2} represent active states of two input masks for the operation XOR, and the active state of output mask is θ_{out}. Then the relation between them is $\theta_{out} = \theta_{in_1} = \theta_{in_2}$.*

When deriving the mask propagation property of the branching operation, we always have to decide the mask active state of one of these three branches according to mask active states of the other two branches. Thus, we have the following property.

Property 3. ***(Three-Branch)*** *Let θ_1 and θ_2 denote two known mask active states, and the mask active state to be decided is θ_3. Then $\theta_3 = 1$, which means that the corresponding branch is potentially active, if either $\theta_1 = 1$ or $\theta_2 = 1$ holds.*

The linear layer can often be represented as matrix multiplication. To specify the word-level mask propagation property of this operation, we introduce the following definition.

Definition 2. ***(Deterministic Pattern)*** *Let the column vector M_{in} and M_{out} respectively represent the column-wise active state of input and output mask of M. Then the* *pair (M_{in}, M_{out}) is called* ***deterministic pattern*** *if the active state of output mask M_{out} is unique given M_{in}.*

Define G as the set $\{M_{in} \mid (M_{in}, M_{out})$ *is a deterministic pattern*$\}$, and then we have:

Property 4. ***(Matrix-Based Linear Layer)*** *Let θ_{in} and θ_{out} represent the column-wise active state of input and output mask for M, separately. Then all words corresponding to θ_{out} are potentially active if $\theta_{in} \notin G$. Otherwise, θ_{out} equals to the corresponding M_{out}.*

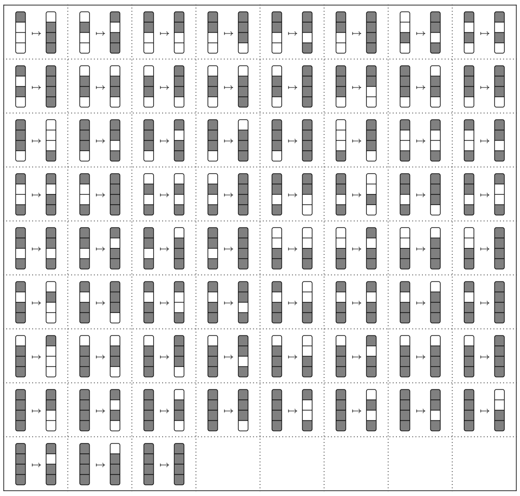

Figure 3: Column-Wise Active State Transitions for M Used in QARMA-64

To make it clear, we take the matrix M used in QARMA-64 [Ava17] as an example. The word-level column-wise active state transition for M is shown in Figure 3, where gray nibbles represent the active ones. Assume that the column vector $M_{in} = (x_0, x_1, x_2, x_3)^t$ and $M_{out} = (y_0, y_1, y_2, y_3)^t$ denote the active state of input and output mask for M, respectively. By observing all these possible transitions, there exist some *deterministic patterns* in Table 2, which can be used to produce the set G. Then we can use this set to give the mask propagation property for the matrix M used in QARMA-64. Let θ_{in} and θ_{out} respectively represent the column-wise active state of mask before and after M. Then $\theta_{out} = (1, 1, 1, 1)^t$ if $\theta_{in} \notin G$. Otherwise, it equals to the corresponding M_{out} shown in Table 2.

Table 2: All *Deterministic Patterns* (M_{in}, M_{out})

M_{in}	$(0,0,0,0)^t$	$(1,0,0,0)^t$	$(0,1,0,0)^t$	$(0,0,1,0)^t$	$(0,0,0,1)^t$
M_{out}	$(0,0,0,0)^t$	$(0,1,1,1)^t$	$(1,0,1,1)^t$	$(1,1,0,1)^t$	$(1,1,1,0)^t$

Notice that when describing the mask propagation property of matrix-based linear layer, we only describe propagation from the input mask. To obtain the mask propagation property from the output, we only have to generate the set G for M^{-1}, the inverse matrix of M, and use Property 4 to derive corresponding equations.

Part 2. Equations for Basic Operations in Key Schedule

Property 5. ***(Substitution)*** *Let S be the S-box used in the key schedule and DDT represents its differential distribution table. The input and output difference are δ_{in} and δ_{out}, respectively. If the corresponding differential propagation probability is denoted as p, we have $p = DDT(\delta_{in}, \delta_{out})$. Then the relation is $p \neq 0$.*

Property 6. ***(XOR)*** *Let δ_{in_1} and δ_{in_2} represent the input differences, and the output difference is denoted as δ_{out}. Then the relation between them is $\delta_{out} = \delta_{in_1} \oplus \delta_{in_2}$.*

Property 7. ***(Three-Branch)*** *Let δ_{in} represent the input difference of the operation, while δ_{out_1} and δ_{out_2} are the output differences. Then the relation between them is $\delta_{out_2} = \delta_{out_1} = \delta_{in}$.*

Part 3. Equations Depicting the KDIB Condition illustrated in Proposition 2
Given an r-round linear hull (θ_0, θ_r) and the corresponding difference on key $\{\delta_0, \delta_1, \ldots, \delta_r\}$, we have the KDIB condition that $\oplus_{j=0}^{r} \theta_j \cdot \delta_j = 0$ holds for all possible linear trails $\{\theta_0, \theta_1, \ldots, \theta_r\}$ with $\varepsilon_\theta \neq 0$ in this linear hull. Seeing that we only care about the active state of mask, it is hard for us to directly use this condition when searching for distinguishers. Hence, we will describe the KDIB condition under word-level.

Property 8. ***(Word-Level KDIB Condition)*** *Given an r-round linear hull (θ_0, θ_r) and the corresponding difference on round key $\{\delta_0, \delta_1, \ldots, \delta_r\}$. Then the difference of the i-th word $\delta_j[i]$ must be zero if the active state of mask of it is 1 for all $0 \leq j \leq r$.*

Part 4. Extra Equations
In order to exclude trivial solutions to these equations, we have to add the constraints that at least one round key is non-zero. And equations describing the active state of input and output mask are also included in this part. For ciphers containing S-box in their key schedule, equations restricting the total propagation probability are included in this part.

Given all these properties, the searching algorithm for KDIB distinguishers is listed in Algorithm 1.

Algorithm 1: `SearchKDIB(`R_1`,`R_2`,`θ_0`,`θ_R`)`

```
input : $R_1$: Number of forward rounds covered by the expected distinguisher
        $R_2$: Number of backward rounds covered by the expected distinguisher
        $\theta_0$: Active state of input mask in the linear hull
        $\theta_R$: Active state of output mask in the linear hull
output: An $(R_1 + R_2)$-round KDIB distinguisher or "No solution"

for all considered active input and output mask words do
    //Equations in Part 1
    for $r \leftarrow 0$ to $R_1 - 1$ do
        Use Property 1∼4 to construct equations for the $r$-th forward round function;
    for $r \leftarrow 0$ to $R_2 - 1$ do
        Use Property 1∼4 to bulid equations for the $r$-th backward round function;
    //Equations in Part 2
    for $r \leftarrow 0$ to $R_1 + R_2 - 1$ do
        Use Property 5∼7 to describe equations for the $r$-th round of key schedule;
    //Equations in Part 3
    Use Property 8 to construct equations describing the KDIB condition;
    //Equations in Part 4
    Construct equations restricting that at least one round key is non-zero;
    Construct equations describing the active state of input and output mask according to
      $\theta_0$ and $\theta_R$;
    Input all these equations into STP and let it solve;
    if STP return a solution then
        Return the solution as the KDIB distinguisher;
Return "No Solution";
```

5 TDIB and Related-Tweak Statistical Saturation Distinguishers for QARMA

Our target cipher QARMA is briefly introduced in Sect. 2.2. In the specification of it [Ava17], the designer claimed that *the attacker does not have control on the key, but she may have full control on the tweak*. Therefore, we focus on related-tweak attacks on QARMA. In this section, we utilize the searching algorithm given in Sect. 4 to find TDIB distinguishers for QARMA.

Under the restriction that there is only one active word in both the input and output mask, we have obtained many 6-round distinguishers for QARMA-64 and -128. To find longer distinguishers, we increase the number of active words in both input and output mask, and finally find 7 different kinds of 8-round TDIB distinguishers, which will be utilized to mount a key recovery attack on 11-round QARMA-128 in Sect. 6.2. And then, several 8-round related-tweak statistical saturation distinguishers for QARMA-64 are presented which are transformed from these 8-round TDIB distinguishers benefiting from Theorem 1. These related-tweak statistical saturation distinguishers will be used to mount key recovery attacks on 10-round QARMA-64 in Sect. 6.1.

5.1 TDIB Distinguishers for 8-Round QARMA

As we can see from Figure 1, QARMA has a central construction consisting of two central rounds and a *Pseudo-Reflector* construction in the middle of the encryption procedure. Thus, we have to construct equations for this part as well as those for all the other operations in the round function and tweak update function. Besides, since we only focus on related-tweak attacks, the difference of user-supplied key should be restricted to zero, while the difference on tweak is non-zero. Here, we set the number of active words in both the input and output mask to be 1.

Adding all the above extra equations into Algorithm 1, we obtained many 6-round distinguishers with 2 rounds before the central construction and another 2 rounds after for both versions of QARMA. However, if we release the restriction with one active word in both input and output mask, longer distinguishers may be obtained. As a result, we achieved 8-round TDIB distinguishers by setting two active words in both input and output mask. And these two active words in the input/output mask are restricted to be in the same column after the operation τ in the first/last round of our expected distinguisher, and they will be transfered into two active words in the same position after the operation M, which forces us to make some additional restriction on the mask value of them.

To be more specific, we denote these active words in the linear hull (Γ, Λ) as $\Gamma[in_0]$, $\Gamma[in_1]$, $\Lambda[out_0]$ and $\Lambda[out_1]$. All possible combinations of (in_0, in_1) satisfying the above restriction are shown in Table 3. Notice that the restriction on (out_0, out_1) is actually the same as that on (in_0, in_1). Thus, Table 3 can also be used to show all the possible combinations of (out_0, out_1).

Table 3: All Possible Combinations of Active Words

(in_0, in_1)	Type	(in_0, in_1)	Type	(in_0, in_1)	Type	(in_0, in_1)	Type
(0,10)	I	(1,11)	I	(6,12)	I	(7,13)	I
(0,5)	II	(11,14)	II	(3,6)	II	(8,13)	II
(0,15)	I	(4,11)	I	(6,9)	I	(2,13)	I
(5,10)	I	(1,14)	I	(3,12)	I	(7,8)	I
(10,15)	II	(1,4)	II	(9,12)	II	(2,7)	II
(5,15)	I	(4,14)	I	(3,9)	I	(2,8)	I

In order to get the expected distinguishers, we have to restrict the value of the input and output masks. For Type-I combinations shown in Table 3, the restriction of mask value is shown in Restriction 1. And Restriction 2 describes the constraint for Type-II combinations.

Restriction 1. *For both versions of `QARMA`,* $\Gamma[in_0] = \Gamma[in_0] \lll 2$*,* $\Gamma[in_1] = \Gamma[in_1] \lll 2$*,* $\Gamma[in_1] = \Gamma[in_0] \lll 1$*,* $\Lambda[out_0] = \Lambda[out_0] \lll 2$*,* $\Lambda[out_1] = \Lambda[out_1] \lll 2$ *and* $\Lambda[out_1] = \Lambda[out_0] \lll 1$*.*

Restriction 2. *For `QARMA`-64,* $\Gamma[in_0] = \Gamma[in_1]$ *and* $\Lambda[out_0] = \Lambda[out_1]$*. For `QARMA`-128,* $\Gamma[in_0] = \Gamma[in_1] \lll 4$*,* $\Gamma[in_1] = \Gamma[in_0] \lll 4$*,* $\Lambda[out_0] = \Lambda[out_1] \lll 4$ *and* $\Lambda[out_1] = \Lambda[out_0] \lll 4$*.*

Under Restriction 1, the number of possible value of $(\Gamma[in_0], \Gamma[in_1], \Lambda[out_0], \Lambda[out_1])$ is 9 for both versions of `QARMA`. Therefore, the expected TDIB distinguisher only contains small number of non-trivial linear hulls, which doesn't fulfill the condition of Proposition 3 and thus the statistical model will not suitable here. Hence we choose linear hulls satisfying Restriction 2. Since the tweak update function is symmetric, we set $in_0 = out_0$ and $in_1 = out_1$ for the purpose of reducing the conditions on the difference of tweak.

To construct TDIB distinguishers based on linear hulls satisfying Restriction 2, it is necessary for us to determine whether there exists a same difference of tweak for linear hulls with the same position of active words. For both versions of `QARMA`, we found the corresponding difference of tweak for almost all Type-II combinations except for $(in_0, in_1) = (10, 15)$ with the help of STP. And the number of non-trivial linear hulls contained in the 8-round distinguisher is $(2^4 - 1)(2^4 - 1)$ for `QARMA`-64 and $(2^8 - 1)(2^8 - 1)$ for `QARMA`-128.

Hence, we have obtained 7 different kinds of TDIB distinguishers for both versions of `QARMA` containing linear hulls satisfying Restriction 2. To be specific, the 8-round distinguisher with $(in_0, in_1) = (0, 5)$ for `QARMA`-64 is shown in Figure 4, while the concrete figure of the distinguisher with $(in_0, in_1) = (0, 5)$ for `QARMA`-128 is omitted due to the similarity between them. And we list the difference of tweak of these two distinguishers in Table 4. As for the other 6 different kinds of TDIB distinguishers, we will not show the concrete figure or the difference of tweak here due to the similarity with these two distinguishers and the limits of paper length.

Table 4: Difference of Round Tweak for 8-Round `QARMA` with $(in_0, in_1) = (0, 5)$

round	Δt_i for `QARMA`-64	Δt_i for `QARMA`-128
5	0x0000000040000000	0x00000000000000001600000000000000
6	0x0000000000004000	0x00000000000000000000000016000000
7	0x0000000004000000	0x00000000000000000016000000000000
8	0x0000000000000200	0x000000000000000000000000008B0000
9	0x0000000000000200	0x000000000000000000000000008B0000
10	0x0000000004000000	0x00000000000000000016000000000000
11	0x0000000000004000	0x00000000000000000000000016000000
12	0x0000000040000000	0x00000000000000001600000000000000

5.2 Related-Tweak Statistical Saturation Distinguishers for `QARMA`-64

Here, we will transform these 8-round TDIB distinguishers into related-tweak statistical saturation (SS) ones by utilizing Theorem 1. Since we mount attacks by only adding several rounds on the bottom of these distinguishers, the first round of them should be a reduced one. Notice that a reduced first round of `QARMA` is only composed of `AddRoundTweakey` and `SubCells`. One of such related-tweak SS distinguisher transformed from the TDIB distinguisher is shown in Figure 4 circled by the dotted line. Since the output mask cannot

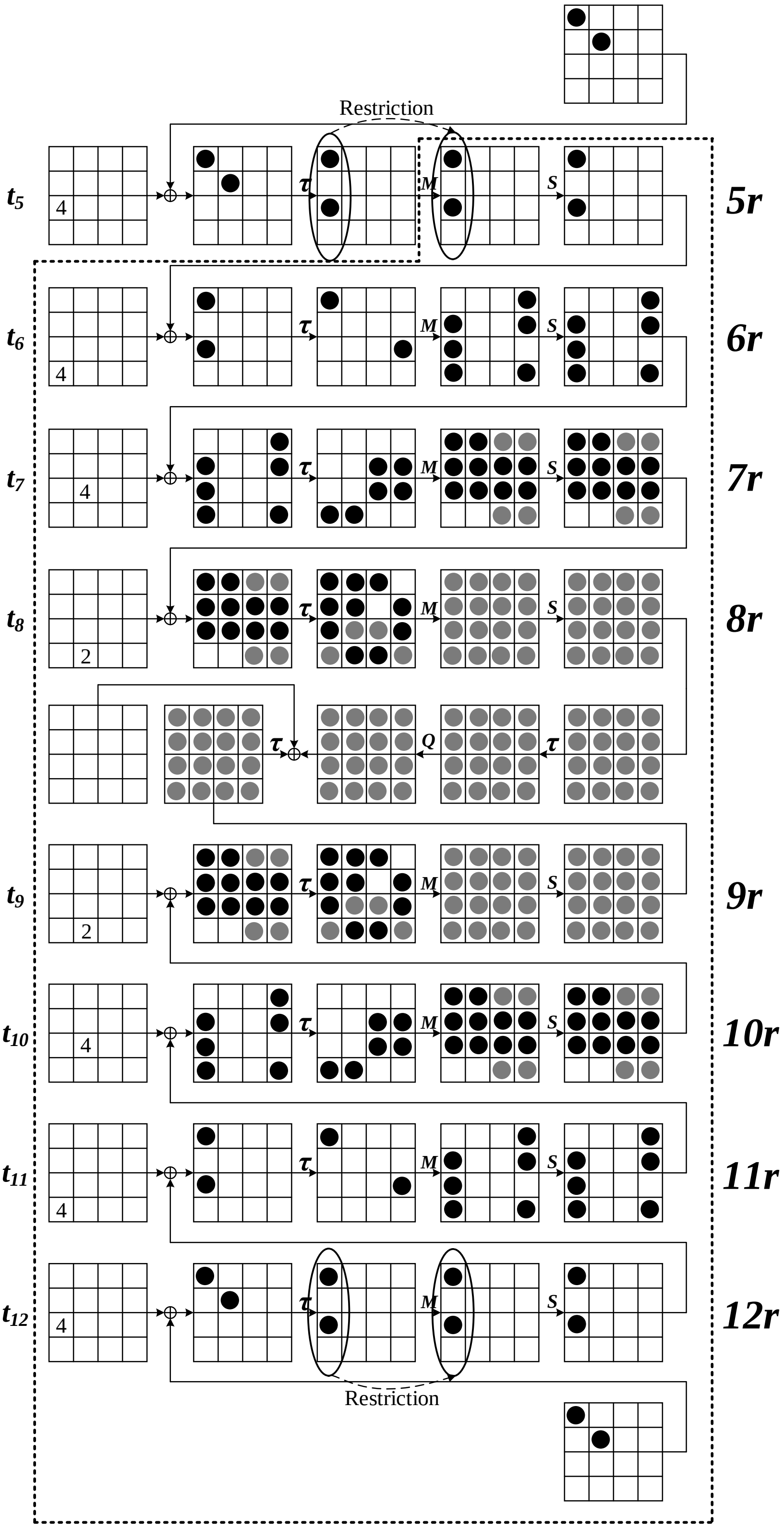

Figure 4: TDIB Distinguisher for 8-Round QARMA-64. White words are non-active ones, while black ones are active words. And gray words can be active or non-active.

take all possible values in $\mathbb{F}_2^8\backslash\{0\}$ due to $\Lambda[out_0] = \Lambda[out_1]$, Theorem 1 cannot be directly used to transform such TDIB distinguishers into related-tweak SS ones. But we can achieve it after changing the output of H and obtain the following theorem, which can be proved in a similar way with the one proposed for the Lemma 1 in [HCGW18].

Theorem 3. *Let (Γ, Λ) be the linear hull contained in the TDIB distinguishers of the block cipher H with $\Gamma = (\Gamma_{in}, 0)$ and $\Lambda = (\Lambda_{out}, 0)$, where $\Gamma_{in} = \Gamma[in_0]||\Gamma[in_1]$, $\Lambda_{out} = \Lambda[out_0]||\Lambda[out_1]$ and $\Lambda[out_0] = \Lambda[out_1]$. If we take all possible values of plaintext P by fixing $P[in_0]||P[in_1]$ as some constant $I \in \mathbb{F}_2^8$, and respectively encrypt them under (z, κ) and (z', κ). Denote the corresponding ciphertext as C and C', separately, then*

$$\begin{aligned}&\#\{P \mid P[in_0]||P[in_1] = I,\ C[out_0] \oplus C[out_1] = c\}\\ =&\#\{P \mid P[in_0]||P[in_1] = I,\ C'[out_0] \oplus C'[out_1] = c\}\end{aligned}$$

holds for any $c \in \mathbb{F}_2^4$.

Proof. We rewrite the cipher H with four inputs and three outputs:

$$H(x, y, z, \kappa) = (H_1(x, y, z, \kappa), H_2(x, y, z, \kappa), H_3(x, y, z, \kappa)),$$

where $x = P[in_0]||P[in_1]$, y is the concatenated value of other 14 nibbles of P, $H_1(x, y, z, \kappa) = C[out_0]$, $H_2(x, y, z, \kappa) = C[out_1]$ and $H_3(x, y, z, \kappa)$ is the concatenated value of other 14 nibbles. Then we change the output of H and produce a new function H' as follows:

$$H'(x, y, z, \kappa) = (H_1(x, y, z, \kappa) \oplus H_2(x, y, z, \kappa), H_3(x, y, z, \kappa)).$$

Recall that the bias of the linear hull (Γ, Λ) under (z, κ) can be represented by

$$\begin{aligned}\varepsilon(z, \kappa) &= Pr[\Gamma \cdot (x||y) \oplus \Lambda \cdot H(x, y, z, \kappa) = 0] - \frac{1}{2}\\ &= Pr[\Gamma_{in} \cdot x \oplus \Lambda[out_0] \cdot C[out_0] \oplus \Lambda[out_1] \cdot C[out_1]] - \frac{1}{2}\\ &= Pr[\Gamma_{in} \cdot x \oplus \Lambda[out_0] \cdot (C[out_0] \oplus C[out_1])] - \frac{1}{2}\\ &= Pr[\Gamma \cdot (x||y) \oplus \Lambda' \cdot H'(x, y, z, \kappa) = 0] - \frac{1}{2},\end{aligned}$$

where $\Lambda' = (\Lambda'_{out}, 0)$ with $\Lambda'_{out} = \Lambda[out_0]$. Hence for the function H', the bias of (Γ, Λ') under (z, κ) is the same as that under (z', κ). In other words, an 8-round TDIB distinguisher for H implies an 8-round TDIB distinguisher for H'. Therefore, we can utilize Theorem 1 on H' to obtain the following related-tweak invariant distribution property:

$$\begin{aligned}&\#\{P \mid P[in_0]||P[in_1] = I,\ C[out_0] \oplus C[out_1] = c\}\\ =&\#\{P \mid P[in_0]||P[in_1] = I,\ C'[out_0] \oplus C'[out_1] = c\}\end{aligned}$$

□

To make it clear, we list all these 8-round related-tweak SS distinguishers in Table 5, which utilize the related-tweak invariant distribution illustrated in Theorem 3. Besides, tweak differences of these distinguishers are listed in Appendix C.

Table 5: Related-Tweak SS Distinguishers for 8-Round `QARMA`-64

No.	1	2	3	4	5	6	7
(in_0, in_1)	(0,8)	(1,9)	(5,13)	(2,10)	(6,14)	(3,11)	(7,15)
(out_0, out_1)	(0,5)	(11,14)	(1,4)	(3,6)	(9,12)	(8,13)	(2,7)

6 Key Recovery Attacks on Reduced-Round QARMA

In this section, we will proceed related-tweak SS attack on 10-round QARMA-64 and TDIB attack on 11-round QARMA-128. In fact, we have also tried to recover the key for QARMA-64 with 8-round equivalent TDIB distinguishers and mount key recovery attack on QARMA-128 with related-tweak SS distinguishers. As a result, the complexity of TDIB attack on 10 rounds QARMA-64 is higher than that of related-tweak SS attack. On the other hand, the related-tweak SS attack on 11-round QARMA-128 has higher complexity than the TDIB attack. Due to the limits of paper length, we will not provide the concrete key recovery procedures of these two attacks here.

In our following attacks, we will guess equivalent keys ek^0, sk^0 and sk^1 instead of k^0 and w^0, where $ek^0 = M(\tau(k^0))$, $sk^0 = k^0 \oplus w^0$ and $sk^1 = k^0 \oplus w^1$.

6.1 Related-Tweak SS Attacks on 10-Round QARMA-64

6.1.1 Attack Procedure

During this attack, we will utilize 4 different related-tweak SS distinguishers presented in Table 5, which are No. 1, No. 3, No. 4 and No. 7. By adding two rounds after these 8-round distinguishers, we can give a key recovery attack on 10-round cipher, which is described in Algorithm 2. To make it clear, we present the detailed attack procedure with No. 1 distinguisher in Figure 5 and Algorithm 3.

Algorithm 2: Key Recovery Procedure of 10-Round QARMA-64

```
Proceed with Algorithm 3 and obtain 32 guessed key bits, which are
    $sk^1[4,5,10,11,14,15]$ and $ek^0[0,5]$;
Proceed with a similar procedure with No. 3 distinguisher to recover 32 key bits
    $sk^1[0,1,4,5,14,15]$ and $ek^0[1,4]$;
Use No. 4 distinguisher to recover 32 key bits $sk^1[2,3,6,7,8,9]$ and $ek^0[3,6]$;
32 key bits $sk^1[2,3,8,9,12,13]$ and $ek^0[2,7]$ can be got with No. 7 distinguisher;
for $2^{32}$ $ek^0[8,9,10,11,12,13,14,15]$ do
    Recover $k^0$ with $ek^0 = M(\tau(k^0))$;
    Compute $w^1$ by using the relation $w^1 = sk^1 \oplus k^0$;
    Obtain $w^0$ according to $w^1 = o(w^0)$;
    Use one $\{plaintext, ciphertext, tweak\}$ triple to check whether it is right;
```

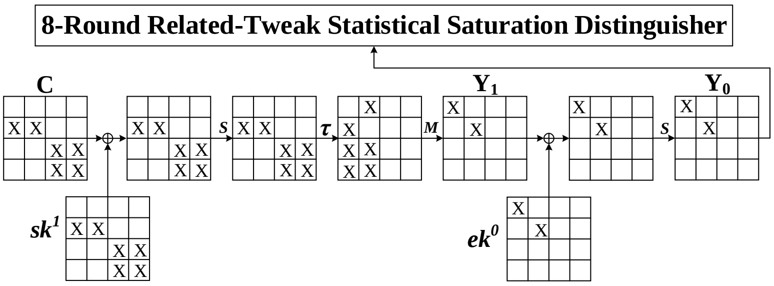

Figure 5: Key Recovery Attack on 10-Round QARMA-64 with No. 1 Distinguisher

Algorithm 3: Key Recovery of 10-Round `QARMA`-64 with No. 1 Distinguisher

```
Randomly choose two values $v_1, v_2 \in \mathbb{F}_2^4$ and set $P[0] = v_1$, $P[8] = v_2$;
Allocate two arrays $V_1[x_1]$ and $V_1'[x_1']$ with $|x_1| = 24 = |x_1'|$, and initialize them to zeros;
for all possible values of plaintext $P$ satisfying $P[0] = v_1$ and $P[8] = v_2$ do
    Query the ciphertexts $C$ and $C'$ under $(z, \kappa)$ and $(z \oplus \Delta z, \kappa)$ separately;
    Let $x_1 = C[4,5,10,11,14,15]$ and $V_1[x_1] \leftarrow V_1[x_1] + 1$;
    Let $x_1' = C'[4,5,10,11,14,15]$ and $V_1'[x_1'] \leftarrow V_1'[x_1'] + 1$;
for $2^{24}$ $sk^1[4,5,10,11,14,15]$ do
    Allocate two arrays $V_2[x_2]$ and $V_2'[x_2']$ with $|x_2| = 8 = |x_2'|$, and initialize them to zeros;
    for $2^{24}$ $x_1$ and $x_1'$ do
        Decrypt one-round for $x_1$ and $x_1'$ to get $Y_1[0,5]$ and $Y_1'[0,5]$;
        Let $x_2 = Y_1[0,5]$ and $V_2[x_2] \leftarrow V_2[x_2] + V_1[x_1]$;
        Let $x_2' = Y_1'[0,5]$ and $V_2'[x_2'] \leftarrow V_2'[x_2'] + V_1'[x_1']$;
    for $2^8$ $ek^0[0,5]$ do
        Allocate and initialize two arrays $V_3[x_3]$ and $V_3'[x_3']$ with $|x_3| = 4 = |x_3'|$;
        for $2^8$ $x_2$ and $x_2'$ do
            Decrypt $x_2$ and $x_2'$ to get $Y_0[0,5]$ and $Y_0'[0,5]$;
            Let $x_3 = Y_0[0] \oplus Y_0[5]$ and $V_3[x_3] \leftarrow V_3[x_3] + V_2[x_2]$;
            Let $x_3' = Y_0'[0] \oplus Y_0'[5]$ and $V_3'[x_3'] \leftarrow V_3'[x_3'] + V_2'[x_2']$;
        if $V_3[x_3] = V_3'[x_3]$ holds for all $2^4$ $x_3$ then
            return the guessed key bits;
        else
            Discard this key;
```

6.1.2 Attack Complexity

According to Theorem 2, we can see that the probability to accept a wrong key is $\log_2(\alpha_1) \leq \left(2^4 - 1 - 4\right) 2^{56+1} - 2^{56(2^4-1)/2} \approx -2.7 \times 10^{126}$. By running Algorithm 3, we can obtain 32 guessed key bits. Hence, the number of wrong keys left is $2^{32} \times \alpha_1 \approx 0$, which means that the 32 guessed key bits left are actually the right ones. Data complexity of Algorithm 3 is 2^{57} chosen plaintext-tweak pairs, while the memory requirements are $2^{29.6}$ bits needed for these arrays. The main time cost of Algorithm 3 is 2^{57} querying ciphertexts, which is 2^{57} 10-round encryptions. Obviously, the data complexity, memory requirements and total time complexity of procedures with No. 3, No. 4 and No. 7 distinguishers are the same as those of Algorithm 3. It follows that the total data complexity of this key recovery attack is $N = 2^{59}$ chosen plaintext-tweak pairs, while the memory requirements are $M = 2^{29.6}$ bits since these arrays can be reused for different procedures. And the total time complexity is $T \approx 2^{59}$ 10-round encryptions. Note that $TN = 2^{118} \leq 2^{126}$, which means that this attack is a valid one.

6.2 TDIB Attack on 11-Round `QARMA`-128

6.2.1 Attack Procedure

To be more specific, we only utilize two distinguishers, which are $(in_0, in_1) = (0, 5)$ and $(in_0, in_1) = (1, 4)$ presented in Table 3. To simplify our clarification, we denote the one with $(in_0, in_1) = (0, 5)$ as No. 1 distinguisher and the other one as No. 3 distinguisher. By adding one round before these distinguishers and another two rounds after, we can proceed with a key recovery attack on 11-round QARMA-128, which is described in Algorithm 4. In order to make our attack procedure clear, we present the detailed attack procedure with No. 1 distinguisher in Figure 6 and Algorithm 5.

```
Algorithm 4: Key Recovery Procedure of 11-Round QARMA-128
1 Proceed with Algorithm 5 and obtain 80 guessed key bits, which are $sk^0[0,5]$,
    $sk^1[4,5,10,11,14,15]$ and $ek^0[0,5]$;
2 Utilize No. 3 distinguisher to obtain 80 key bits $sk^0[3,6]$, $sk^1[2,3,6,7,8,9]$ and
    $ek^0[3,6]$;
3 for $2^{128}$ $sk^0[1,2,4,7,8,9,10,11,12,13,14,15]||sk^1[0,1,12,13]$ do
4     Recover $k^0$ and $w^0$ by using $sk^0 = k^0 \oplus w^0$, $sk^1 = k^0 \oplus w^1$ and $w^1 = o(w^0)$;
5     Compute $cek^0 = M(\tau(k^0))$;
6     if $cek^0[0,3,5,6] = ek^0[0,3,5,6]$ then
7         Use one $\{plaintext, ciphertext, tweak\}$ triple to check whether it is right;
```

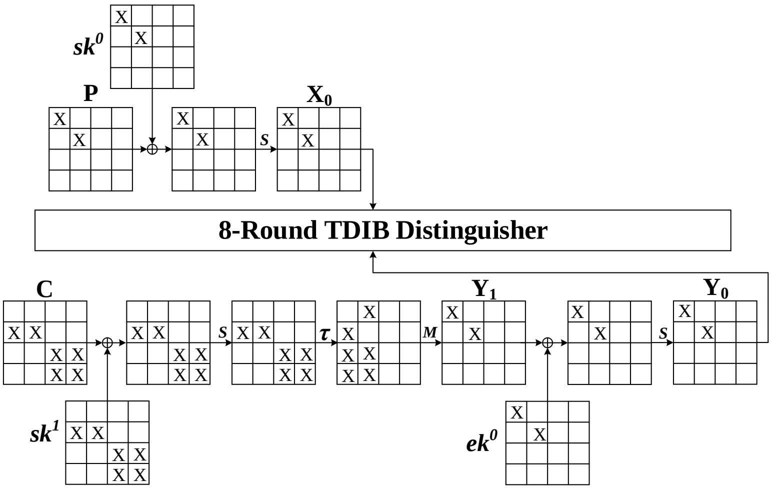

Figure 6: Key Recovery Attack on 11-Round QARMA-128 with No. 1 Distinguisher

6.2.2 Attack Complexity

Since λ is sufficiently large, we can utilize Proposition 3 to evaluate the data complexity of Algorithm 5. According to the proposition, we have $N_1 = \frac{2^{128+0.5}}{\sqrt{\lambda}-q_{1-\alpha_1}\sqrt{2}}(q_{1-\alpha_0}+q_{1-\alpha_1})$ hold for QARMA-128 with $\lambda \approx 2^{15.98}$. Thus, after choosing the value of α_0 and α_1, we can compute the value N_1. Here, we set $\alpha_0 = 2^{-3.7}$, $\alpha_1 = 2^{-81.1}$ and then $N_1 \approx 2^{124.1}$. And the decision threshold is $s_\mathcal{T} = \frac{\sqrt{\lambda}}{N_1\sqrt{2}}q_{1-\alpha_0} + \frac{\lambda}{2N_1} \approx 2^{-109.1}$. Then the data complexity of Algorithm 5 is $2^{125.1}$ known plaintext-tweak pairs. The total time complexity of Algorithm 5 is mainly determined by Step 3∼Step 5, which costs $2N_1$ MA equivalent to $2^{125.1}$ 11-round encryptions. And the memory requirements are 2^{71} bits needed for these arrays. Since $\alpha_1 = 2^{-81.1}$, the 80 guessed key bits left after Step 1 in Algorithm 4 are the right ones. Similarly, the other 80 key bits obtained after Step 2 are all right key bits. It follows that the total time complexity of Algorithm 4 is $T \approx 2^{126.1}$ 11-round encryptions. And the total data complexity is $N = 2^{125.1} \times 2 = 2^{126.1}$ known plaintext-tweak pairs with memory requirements 2^{71} bits. Since $TN = 2^{252.2} < 2^{254}$, this key recovery attack for QARMA-128 is a valid one.

References

[Ava17] Roberto Avanzi. 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 Trans. Symmetric Cryptol.*, 2017(1):4–44, 2017. URL: https://doi.org/10.13154/tosc.v2017.i1.4-44, doi:10.13154/tosc.v2017.i1.4-44.

[BBR$^+$13] Andrey Bogdanov, Christina Boura, Vincent Rijmen, Meiqin Wang, Long Wen, and Jingyuan Zhao. Key difference invariant bias in block ciphers. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part I*, volume 8269 of *Lecture Notes in Computer Science*, pages 357–376. Springer, 2013. URL: https://doi.org/10.1007/978-3-642-42033-7_19, doi:10.1007/978-3-642-42033-7_19.

[BLNW12] Andrey Bogdanov, Gregor Leander, Kaisa Nyberg, and Meiqin Wang. Integral and multidimensional linear distinguishers with correlation zero. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *Lecture Notes in Computer Science*, pages 244–261. Springer, 2012. URL: https://doi.org/10.1007/978-3-642-34961-4_16, doi:10.1007/978-3-642-34961-4_16.

[BR14] Andrey Bogdanov and Vincent Rijmen. Linear hulls with correlation zero and linear cryptanalysis of block ciphers. *Des. Codes Cryptography*, 70(3):369–383, 2014. URL: https://doi.org/10.1007/s10623-012-9697-z, doi:10.1007/s10623-012-9697-z.

[BSST09] Clark W. Barrett, Roberto Sebastiani, Sanjit A. Seshia, and Cesare Tinelli. Satisfiability modulo theories. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 825–885. IOS Press, 2009. URL: https://doi.org/10.3233/978-1-58603-929-5-825, doi:10.3233/978-1-58603-929-5-825.

[BW12] Andrey Bogdanov and Meiqin Wang. Zero correlation linear cryptanalysis with reduced data complexity. In Anne Canteaut, editor, *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, volume 7549 of *Lecture Notes in Computer Science*, pages 29–48. Springer, 2012. URL: https://doi.org/10.1007/978-3-642-34047-5_3, doi:10.1007/978-3-642-34047-5_3.

[Coo71] Stephen A. Cook. The complexity of theorem-proving procedures. In Michael A. Harrison, Ranan B. Banerji, and Jeffrey D. Ullman, editors, *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing, May 3-5, 1971, Shaker Heights, Ohio, USA*, pages 151–158. ACM, 1971. URL: http://doi.acm.org/10.1145/800157.805047, doi:10.1145/800157.805047.

[CS09] Baudoin Collard and François-Xavier Standaert. A statistical saturation attack against the block cipher PRESENT. In Marc Fischlin, editor, *Topics in Cryptology - CT-RSA 2009, The Cryptographers' Track at the RSA Conference 2009, San Francisco, CA, USA, April 20-24, 2009. Proceedings*, volume 5473 of *Lecture Notes in Computer Science*, pages 195–210.

Springer, 2009. URL: https://doi.org/10.1007/978-3-642-00862-7_13, doi:10.1007/978-3-642-00862-7_13.

[DEM16] Christoph Dobraunig, Maria Eichlseder, and Florian Mendel. Square attack on 7-round kiasu-bc. In Manulis et al. [MSS16], pages 500–517. URL: https://doi.org/10.1007/978-3-319-39555-5_27, doi:10.1007/978-3-319-39555-5_27.

[DGV94] Joan Daemen, René Govaerts, and Joos Vandewalle. Correlation matrices. In Bart Preneel, editor, *Fast Software Encryption: Second International Workshop. Leuven, Belgium, 14-16 December 1994, Proceedings*, volume 1008 of *Lecture Notes in Computer Science*, pages 275–285. Springer, 1994. URL: https://doi.org/10.1007/3-540-60590-8_21, doi:10.1007/3-540-60590-8_21.

[DKR97] Joan Daemen, Lars R. Knudsen, and Vincent Rijmen. The block cipher Square. In Eli Biham, editor, *Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings*, volume 1267 of *Lecture Notes in Computer Science*, pages 149–165. Springer, 1997. URL: https://doi.org/10.1007/BFb0052343, doi:10.1007/BFb0052343.

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard.* Information Security and Cryptography. Springer, 2002. URL: https://doi.org/10.1007/978-3-662-04722-4, doi:10.1007/978-3-662-04722-4.

[HCGW18] Kai Hu, Tingting Cui, Chao Gao, and Meiqin Wang. Towards key-dependent integral and impossible differential distinguishers on 5-round AES. In Carlos Cid and Michael J. Jacobson Jr., editors, *Selected Areas in Cryptography - SAC 2018 - 25th International Conference, Calgary, AB, Canada, August 15-17, 2018, Revised Selected Papers*, volume 11349 of *Lecture Notes in Computer Science*, pages 139–162. Springer, 2018. URL: https://doi.org/10.1007/978-3-030-10970-7_7, doi:10.1007/978-3-030-10970-7_7.

[HCN08] Miia Hermelin, Joo Yeon Cho, and Kaisa Nyberg. Multidimensional linear cryptanalysis of reduced round serpent. In *Information Security and Privacy, 13th Australasian Conference, ACISP 2008, Wollongong, Australia, July 7-9, 2008, Proceedings*, pages 203–215, 2008. URL: https://doi.org/10.1007/978-3-540-70500-0_15, doi:10.1007/978-3-540-70500-0_15.

[HLL$^+$02] Kyungdeok Hwang, Wonil Lee, Sungjae Lee, Sangjin Lee, and Jongin Lim. Saturation attacks on reduced round skipjack. In *Fast Software Encryption, 9th International Workshop, FSE 2002, Leuven, Belgium, February 4-6, 2002, Revised Papers*, pages 100–111, 2002. URL: https://doi.org/10.1007/3-540-45661-9_8, doi:10.1007/3-540-45661-9_8.

[KLT15] Stefan Kölbl, Gregor Leander, and Tyge Tiessen. Observations on the SIMON block cipher family. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 161–185. Springer, 2015. URL: https://doi.org/10.1007/978-3-662-47989-6_8, doi:10.1007/978-3-662-47989-6_8.

[KR94] Burton S. Kaliski Jr. and Matthew J. B. Robshaw. Linear cryptanalysis using multiple approximations. In Yvo Desmedt, editor, *Advances in Cryptology*

- *CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*, volume 839 of *Lecture Notes in Computer Science*, pages 26–39. Springer, 1994. URL: https://doi.org/10.1007/3-540-48658-5_4, doi:10.1007/3-540-48658-5_4.

[KW02] Lars R. Knudsen and David A. Wagner. Integral cryptanalysis. In *Fast Software Encryption, 9th International Workshop, FSE 2002, Leuven, Belgium, February 4-6, 2002, Revised Papers*, pages 112–127, 2002. URL: https://doi.org/10.1007/3-540-45661-9_9, doi:10.1007/3-540-45661-9_9.

[LJ18] Rongjia Li and Chenhui Jin. Meet-in-the-middle attacks on reduced-round QARMA-64/128. *The Computer Journal*, 2018. URL: https://doi.org/10.1093/comjnl/bxy045.

[LWR16] Yunwen Liu, Qingju Wang, and Vincent Rijmen. Automatic search of linear trails in ARX with applications to SPECK and Chaskey. In Manulis et al. [MSS16], pages 485–499. URL: https://doi.org/10.1007/978-3-319-39555-5_26, doi:10.1007/978-3-319-39555-5_26.

[Mat93] Mitsuru Matsui. Linear cryptanalysis method for DES cipher. In Tor Helleseth, editor, *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397. Springer, 1993. URL: https://doi.org/10.1007/3-540-48285-7_33, doi:10.1007/3-540-48285-7_33.

[MP13] Nicky Mouha and Bart Preneel. Towards finding optimal differential characteristics for ARX: Application to Salsa20. Cryptology ePrint Archive, Report 2013/328, 2013. https://eprint.iacr.org/2013/328.

[MSS16] Mark Manulis, Ahmad-Reza Sadeghi, and Steve Schneider, editors. *Applied Cryptography and Network Security - 14th International Conference, ACNS 2016, Guildford, UK, June 19-22, 2016. Proceedings*, volume 9696 of *Lecture Notes in Computer Science*. Springer, 2016. URL: https://doi.org/10.1007/978-3-319-39555-5, doi:10.1007/978-3-319-39555-5.

[Neu14] Thorsten Neuschel. A new proof of Stirling's formula. *The American Mathematical Monthly*, 121(4):350–352, 2014. URL: http://www.jstor.org/stable/10.4169/amer.math.monthly.121.04.350.

[Nyb94] Kaisa Nyberg. Linear approximation of block ciphers. In Alfredo De Santis, editor, *Advances in Cryptology - EUROCRYPT '94, Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, May 9-12, 1994, Proceedings*, volume 950 of *Lecture Notes in Computer Science*, pages 439–444. Springer, 1994. URL: https://doi.org/10.1007/BFb0053460, doi:10.1007/BFb0053460.

[SCW18] Ling Sun, Huaifeng Chen, and Meiqin Wang. Zero-correlation attacks: statistical models independent of the number of approximations. *Des. Codes Cryptography*, 86(9):1923–1945, 2018. URL: https://doi.org/10.1007/s10623-017-0430-9, doi:10.1007/s10623-017-0430-9.

[SMMK12] Tomoyasu Suzaki, Kazuhiko Minematsu, Sumio Morioka, and Eita Kobayashi. TWINE: A lightweight block cipher for multiple platforms. In Lars R. Knudsen and Huapeng Wu, editors, *Selected Areas in Cryptography, 19th International Conference, SAC 2012, Windsor, ON, Canada, August 15-16, 2012, Revised*

Selected Papers, volume 7707 of *Lecture Notes in Computer Science*, pages 339–354. Springer, 2012. URL: https://doi.org/10.1007/978-3-642-35999-6_22, doi:10.1007/978-3-642-35999-6_22.

[WCC+16] Meiqin Wang, Tingting Cui, Huaifeng Chen, Ling Sun, Long Wen, and Andrey Bogdanov. Integrals go statistical: Cryptanalysis of full skipjack variants. In Thomas Peyrin, editor, *Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers*, volume 9783 of *Lecture Notes in Computer Science*, pages 399–415. Springer, 2016. URL: https://doi.org/10.1007/978-3-662-52993-5_20, doi:10.1007/978-3-662-52993-5_20.

[WZ11] Wenling Wu and Lei Zhang. LBlock: A lightweight block cipher. In Javier López and Gene Tsudik, editors, *Applied Cryptography and Network Security - 9th International Conference, ACNS 2011, Nerja, Spain, June 7-10, 2011. Proceedings*, volume 6715 of *Lecture Notes in Computer Science*, pages 327–344, 2011. URL: https://doi.org/10.1007/978-3-642-21554-4_19, doi:10.1007/978-3-642-21554-4_19.

[YQC18] Dong Yang, Wenfeng Qi, and Huajin Chen. Impossible differential attack on QARMA family of block ciphers. Cryptology ePrint Archive, Report 2018/334, 2018. https://eprint.iacr.org/2018/334.

[ZD16] Rui Zong and Xiaoyang Dong. Meet-in-the-middle attack on QARMA block cipher. Cryptology ePrint Archive, Report 2016/1160, 2016. https://eprint.iacr.org/2016/1160.

[ZDW18] Rui Zong, Xiaoyang Dong, and Xiaoyun Wang. MILP-aided related-tweak/key impossible differential attack and its applications to QARMA, Joltik-BC. Cryptology ePrint Archive, Report 2018/142, 2018. https://eprint.iacr.org/2018/142.

A Algorithm 5 in the Attack on 11-round QARMA-128

B Lemmas Used in Proving Theorem 2

Lemma 1. ***(Stirling Formula, [Neu14])*** *$n! \approx n^n e^{-n}\sqrt{2\pi n}$.*

Lemma 2. *If $m \geq 2$ and $m \in \mathbb{Z}$, then*

$$\sum_{x_0+x_1+\cdots+x_{m-1}=n}\left[\binom{n}{x_0}\binom{n-x_0}{x_1}\cdots\binom{n-\sum_{j=0}^{m-2}x_j}{x_{m-1}}\right]^2 \leq \left[\binom{2n}{n}\right]^{m-1}.$$

Proof. The reliability of this lemma can be easily confirmed by induction.

(1) When $m = 2$, the left side is

$$\sum_{x_0+x_1=n}\left[\binom{n}{x_0}\binom{n-x_0}{x_1}\right]^2 = \sum_{x_0=0}^{n}\left[\binom{n}{x_0}\right]^2 = \binom{2n}{n} \leq \left[\binom{2n}{n}\right]^{2-1}.$$

Since the polynomial $(1+y)^{2n} = (1+y)^n(1+y)^n$, we can derive the last equality of the above formula by comparing the coefficient of y^n for both sides.

Algorithm 5: Key Recovery of 11-Round QARMA-128 with No. 1 Distinguisher

```
Gather $N_1$ plaintext-ciphertext pairs $(P, C)$ and $(P', C')$ under $(z, \kappa)$ and
  $(z \oplus \Delta z, \kappa)$, respectively;
Allocate and initialize two arrays $V_1[x_1]$ and $V_1'[x_1']$ with $|x_1| = 64 = |x_1'|$;
for $N_1$ $(P, C)$ and $(P', C')$ do
    Let $x_1 = P[0,5]||C[4,5,10,11,14,15]$ and $V_1[x_1] \leftarrow V_1[x_1] + 1$;
    Let $x_1' = P'[0,5]||C'[4,5,10,11,14,15]$ and $V_1'[x_1'] \leftarrow V_1'[x_1'] + 1$;
for $2^{48}$ $sk^1[4,5,10,11,14,15]$ do
    Allocate and initialize two arrays $V_2[x_2]$ and $V_2'[x_2']$ with $|x_2| = 32 = |x_2'|$;
    for $2^{64}$ $x_1$ and $x_1'$ do
        Decrypt $x_1$ and $x_1'$ to get $P[0,5]||Y_1[0,5]$ and $P'[0,5]||Y_1'[0,5]$;
        Let $x_2 = P[0,5]||Y_1[0,5]$ and $V_2[x_2] \leftarrow V_2[x_2] + V_1[x_1]$;
        Let $x_2' = P'[0,5]||Y_1'[0,5]$ and $V_2'[x_2'] \leftarrow V_2'[x_2'] + V_1'[x_1']$;
    for $2^{32}$ $ek^0[0,5]||sk^0[0,5]$ do
        Allocate and initialize two arrays $V_3[x_3]$ and $V_3'[x_3']$ with $|x_3| = 16 = |x_3'|$;
        for $2^{32}$ $x_2$ and $x_2'$ do
            Decrypt $x_2$ and $x_2'$ to get $X_0[0] \oplus (X_0[5] \lll 4))||(Y_0[0] \oplus (Y_0[5] \lll 4)$ and
              $X_0'[0] \oplus (X_0'[5] \lll 4))||(Y_0'[0] \oplus (Y_0'[5] \lll 4)$;
            Let $x_3 = (X_0[0] \oplus (X_0[5] \lll 4))||(Y_0[0] \oplus (Y_0[5] \lll 4))$ and
              $V_3[x_3] \leftarrow V_3[x_3] + V_2[x_2]$;
            Let $x_3' = (X_0'[0] \oplus (X_0'[5] \lll 4))||(Y_0'[0] \oplus (Y_0'[5] \lll 4))$ and
              $V_3'[x_3'] \leftarrow V_3'[x_3'] + V_2'[x_2']$;
        Allocate a counter $s$;
        for $\lambda \approx (2^8 - 1)(2^8 - 1)$ linear hulls $(\Gamma, \Lambda)$ do
            Allocate two counters $S$ and $S'$, and initialize them to zeros;
            for $2^{16}$ $x_3$ and $x_3'$ do
                if $\Gamma[0] \cdot (X_0[0] \oplus (X_0[5] \lll 4)) = \Lambda[0] \cdot (Y_0[0] \oplus (Y_0[5] \lll 4))$ then
                    $S \leftarrow S + V_3[x_3]$;
                if $\Gamma[0] \cdot (X_0'[0] \oplus (X_0'[5] \lll 4)) = \Lambda[0] \cdot (Y_0'[0] \oplus (Y_0'[5] \lll 4))$ then
                    $S' \leftarrow S' + V_3'[x_3']$;
            $s \leftarrow s + \left[\left(\frac{S}{N_1} - \frac{1}{2}\right) - \left(\frac{S'}{N_1} - \frac{1}{2}\right)\right]^2$;
        if $s \leq s_\tau$ then
            return the guessed subkey bits;
```

(2) When $m = 3$, the left side is

$$\sum_{x_0+x_1+x_2=n} \left[\binom{n}{x_0}\binom{n-x_0}{x_1}\binom{n-x_0-x_1}{x_2}\right]^2 = \sum_{x_0=0}^{n}\sum_{x_1=0}^{n-x_0}\left[\binom{n}{x_0}\binom{n-x_0}{x_1}\right]^2$$

$$= \sum_{x_0=0}^{n}\left[\binom{n}{x_0}\right]^2\left(\sum_{x_1=0}^{n-x_0}\left[\binom{n-x_0}{x_1}\right]^2\right)$$

$$\leq \sum_{x_0=0}^{n}\left[\binom{n}{x_0}\right]^2\left(\sum_{x_1=0}^{n}\left[\binom{n}{x_1}\right]^2\right)$$

$$= \binom{2n}{n}\binom{2n}{n} = \left[\binom{2n}{n}\right]^{3-1}.$$

(3) Assuming that our conclusion holds when $m = k$, we have to prove that it still holds when $m = k + 1$.

$$\begin{aligned}
&\sum_{x_0+x_1+\cdots+x_k=n} \left[\binom{n}{x_0}\binom{n-x_0}{x_1}\cdots\binom{n-\sum_{j=0}^{k-1}x_j}{x_k}\right]^2 \\
&= \sum_{x_k=0}^{n}\left[\binom{n}{x_k}\right]^2 \sum_{\sum_{i=0}^{k-1}x_i=n-x_k}\left[\binom{n-x_k}{x_0}\binom{n-x_k-x_0}{x_1}\cdots\binom{n-x_k-\sum_{j=0}^{k-2}x_j}{x_{k-1}}\right]^2 \\
&\leq \sum_{x_k=0}^{n}\left[\binom{n}{x_k}\right]^2 \sum_{\sum_{i=0}^{k-1}x_i=n}\left[\binom{n}{x_0}\binom{n-x_0}{x_1}\cdots\binom{n-\sum_{j=0}^{k-2}x_j}{x_{k-1}}\right]^2 \\
&\leq \binom{2n}{n}\left[\binom{2n}{n}\right]^{k-1} = \left[\binom{2n}{n}\right]^{(k+1)-1}.
\end{aligned}$$

Combining all the above analysis, we can see that our conclusion holds for any $m \geq 2$ and $m \in \mathbb{Z}$. □

C Tweak Difference of Distinguishers in Table 5

Table 6: Difference of Round Tweak for No. 1 Distinguisher

round	Δt_i for QARMA-64	Δt_i for QARMA-128
5	0x0000000040000000	0x00000000000000001600000000000000
6	0x0000000000004000	0x00000000000000000000000016000000
7	0x0000000004000000	0x00000000000000000016000000000000
8	0x0000000000000200	0x000000000000000000000000008B0000
9	0x0000000000000200	0x000000000000000000000000008B0000
10	0x0000000004000000	0x00000000000000000016000000000000
11	0x0000000000004000	0x00000000000000000000000016000000
12	0x0000000040000000	0x00000000000000001600000000000000

Table 7: Difference of Round Tweak for No. 2 Distinguisher

round	Δt_i for QARMA-64	Δt_i for QARMA-128
5	0x0020000000000000	0x00001600000000000000000000000000
6	0x0000002000000000	0x00000000000016000000000000000000
7	0x9000000000000000	0x8B000000000000000000000000000000
8	0x0000C00000000000	0x00000000C50000000000000000000000
9	0x0000C00000000000	0x00000000C50000000000000000000000
10	0x9000000000000000	0x8B000000000000000000000000000000
11	0x0000002000000000	0x00000000000016000000000000000000
12	0x0020000000000000	0x00001600000000000000000000000000

Table 8: Difference of Round Tweak for No. 3 Distinguisher

round	Δt_i for QARMA-64	Δt_i for QARMA-128
5	0x0000000000110000	0x00000000000000000000010100000000
6	0x0000000000000011	0x00000000000000000000000000000101
7	0x0018000000000000	0x00001800000000000000000000000000
8	0x0000001800000000	0x00000000000018000000000000000000
9	0x0000001800000000	0x00000000000018000000000000000000
10	0x0018000000000000	0x00001800000000000000000000000000
11	0x0000000000000011	0x00000000000000000000000000000101
12	0x0000000000110000	0x00000000000000000000010100000000

Table 9: Difference of Round Tweak for No. 4 Distinguisher

round	Δt_i for QARMA-64	Δt_i for QARMA-128
5	0x0000020000000000	0x00000000001600000000000000000000
6	0x0900000000000000	0x008B0000000000000000000000000000
7	0x0000090000000000	0x00000000008B00000000000000000000
8	0x0C00000000000000	0x00C50000000000000000000000000000
9	0x0C00000000000000	0x00C50000000000000000000000000000
10	0x0000090000000000	0x00000000008B00000000000000000000
11	0x0900000000000000	0x008B0000000000000000000000000000
12	0x0000020000000000	0x00000000001600000000000000000000

Table 10: Difference of Round Tweak for No. 5 Distinguisher

round	Δt_i for QARMA-64	Δt_i for QARMA-128
5	0x0008000000000000	0x00000000240000000000000000000000
6	0x0000000000040000	0x00000000000000000000009200000000
7	0x0000000000000004	0x00000000000000000000000000000092
8	0x0002000000000000	0x00000049000000000000000000000000
9	0x0002000000000000	0x00000049000000000000000000000000
10	0x0000000000000004	0x00000000000000000000000000000092
11	0x0000000000040000	0x00000000000000000000009200000000
12	0x0008000000000000	0x00000000240000000000000000000000

Table 11: Difference of Round Tweak for No. 6 Distinguisher

round	Δt_i for QARMA-64	Δt_i for QARMA-128
5	0x0000000400000000	0x00000000000000160000000000000000
6	0x0000000020000000	0x00000000000000008B00000000000000
7	0x0000000000002000	0x0000000000000000000000008B000000
8	0x0000000002000000	0x0000000000000000008B000000000000
9	0x0000000002000000	0x0000000000000000008B000000000000
10	0x0000000000002000	0x0000000000000000000000008B000000
11	0x0000000020000000	0x00000000000000008B00000000000000
12	0x0000000400000000	0x00000000000000160000000000000000

Table 12: Difference of Round Tweak for No. 7 Distinguisher

round	Δt_i for QARMA-64	Δt_i for QARMA-128
5	0x0000000000004000	0x00000000000000000000000016000000
6	0x0000000004000000	0x00000000000000000016000000000000
7	0x0000000000000200	0x000000000000000000000000008B0000
8	0x0000000000200000	0x000000000000000000008B0000000000
9	0x0000000000200000	0x000000000000000000008B0000000000
10	0x0000000000000200	0x000000000000000000000000008B0000
11	0x0000000004000000	0x00000000000000000016000000000000
12	0x0000000000004000	0x00000000000000000000000016000000