Key-Recovery Attacks on `CRAFT` and `WARP` (Full Version)

Ling Sun[1,2,3], Wei Wang[1,3], and Meiqin Wang(✉)[1,3]

[1] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan, China
[2] State Key Laboratory of Cryptology, P. O. Box 5159, Beijing, 100878, China
[3] School of Cyber Science and Technology, Shandong University, Qingdao, China
{`lingsun, weiwangsdu, mqwang`}`@sdu.edu.cn`

Abstract. This paper considers the security of `CRAFT` and `WARP`. We present a practical key-recovery attack on full-round `CRAFT` in the related-key setting with only one differential characteristic, and the theoretical time complexity of the attack is $2^{36.09}$ full-round encryptions. The attack is verified in practice. The test result indicates that the theoretical analysis is valid, and it takes about 15.69 hours to retrieve the key. A full-round key-recovery attack on `WARP` in the related-key setting is proposed, and the time complexity is $2^{44.58}$ full-round encryptions. The theoretical attack is implemented on a round-reduced version of `WARP`, which guarantees validity. Besides, we give a 33-round multiple zero-correlation linear attack on `WARP`, which is the longest attack on the cipher in the single-key attack setting. We note that the attack results in this paper do not threaten the security of `CRAFT` and `WARP` as the designers do not claim security under the related-key attack setting.

Keywords: Differential attack · Zero-correlation linear attack · Related-key · `CRAFT` · `WARP`.

1 Introduction

Differential cryptanalysis [5] is one of the most fundamental cryptanalytic methods regarding symmetric-key primitives. It investigates how an input difference propagates through the cipher. If a particular output difference appears in a non-random way, this observation can be used to construct a distinguisher or recover keys.

For now, security against differential cryptanalysis is a baseline for modern symmetric-key primitives. Consequently, the probability of the distinguisher utilised in the differential attack typically is close to 2^{-n}, where n is the block size. However, for ciphers without related-key security, we can find differential distinguishers with extremely high probabilities. If these ciphers are misused in malicious cases, efficiently retrieving the key is more meaningful than distinguishing attacks. Moreover, according to Kerchhoffs' principle [15], the security of the encryption scheme should rely solely on the secrecy of the key. Hence, the motivation is to accomplish efficient key-recovery attacks on `CRAFT` [3] and `WARP` [1], two ciphers without related-key security, in the related-key setting.

1.1 Our Contributions

This work centres on the security of `CRAFT` and `WARP`, and the results are summarised as follows.

Practical related-key differential attack on full-round `CRAFT`. The previous related-key attack [13] with only one differential characteristic on `CRAFT` [3] has unrealistic time complexity, and we manage to propose a practical one. According to the theoretical analysis, the data complexity of our attack on full-round `CRAFT` is $2^{35.17}$ chosen plaintexts; the time complexity is $2^{36.09}$ full-round encryptions; the memory complexity is negligible. Given that the complexity of the new attack is practical, we try to verify it in practice. The test result indicates that the theoretical analysis is valid, and it takes about 15.69 hours to retrieve the key. A summary of cryptanalytic results on `CRAFT` in the related-key attack setting can be found in Table 1.

Related-key differential attack on full-round `WARP`. Since `WARP` [1] applies the same S-box as `CRAFT`, we wonder about its differential property in the related-key attack setting. With the automatic searching method, we first identify 384 full-round related-key differential characteristics with probability 2^{-40}. One characteristic is employed to accomplish the key-recovery attack. Based on the theoretical analysis, the data complexity of our attack on full-round `WARP` is $2^{44.58}$ chosen plaintexts; the time complexity is $2^{44.58}$ encryptions; the memory complexity is negligible. Because the complexity of the full-round attack is impractical, we attempt to launch an attack on a round-reduced version of `WARP`. We consider a 27-round version of `WARP` and remark that the key enumeration procedures remain the same. The test results demonstrate the validity of the theoretical attack. Excluding the time to collect right pairs, the runtime of the round-reduced key-recovery attack is about 44.50 hours. A summary of cryptanalytic results on `WARP` can be found in Table 1.

33-round multiple zero-correlation linear attack on `WARP`. When analysing the security of `WARP`, we notice that the designers expected that the longest attack on the cipher in the single-key attack setting does not exceed 32 rounds. We propose a 33-round multiple zero-correlation linear attack, and it is the longest attack on `WARP` in the single-key attack setting as far as we know. A sketch of the 33-round attack can be found in Table 1.

Outline. Section 2 reviews two target ciphers and presents the main idea of the differential attack in this paper. The practical key-recovery attack on `CRAFT` in the related-key attack setting is given in Section 3. In Section 4, we propose the full-round related-key differential attack on `WARP`. Section 5 investigates the security of `WARP` against the multiple zero-correlation linear attack. Section 6 concludes the paper.

Table 1. Summary of cryptanalytic results on `CRAFT` and `WARP`.

Cipher	Setting	Attack	Round	Time	Data	Memory	P_S	Ref.
`CRAFT`	RK	Differential	32	$2^{85.00}$	$2^{31.00}$	$2^{41.00}$	97.72%	[13]
				$2^{32.00}$⊛	$2^{35.17}$	2^{6}	-	[13]
				$\mathbf{2^{36.09}}$	$\mathbf{2^{35.17}}$	**neg**	**100%**◎	**Section 3**
`WARP`	SK	Differential	21	$2^{113.00}$	$2^{113.00}$	$2^{72.00}$	-	[17]
		Differential	23	$2^{106.68}$	$2^{106.62}$	$2^{106.62}$	92.09%	[26]
		Rectangle	24	$2^{125.18}$	$2^{126.06}$	$2^{127.06}$	86.20%	[26]
		Rectangle	26	$2^{115.90}$	$2^{120.60}$	$2^{120.60}$	97.67%	[19]
		Integral	32	$2^{127.00}$	$2^{127.00}$	$2^{108.00}$	-	[14]
		Zero-correlation	**33**	$\mathbf{2^{127.01}}$	$\mathbf{2^{97.71}}$	$\mathbf{2^{100.00}}$	**50.00%**	**Section 5**
	RK	Differential	41⁎	$2^{37.00}$	$2^{37.00}$	$2^{9.59}$	-	[26]
		Differential	**41**	$\mathbf{2^{44.58}}$	$\mathbf{2^{44.58}}$	**neg**	**100%**◎	**Section 4**

SK stands for the single-key attack setting. RK stands for the related-key attack setting.

⊛ The attack uses eight related-key characteristics with distinct key differences. *Eight pairs* of keys are required. The time complexity of the attack *might be wrong* since it should not be less than the data complexity.

⁎ The attack recovers 60 bits of the key and is *not a complete* key-recovery attack.

◎ The success probability is estimated with practical tests.

2 Preliminaries

In this section, we first review the two ciphers studied in the paper. Note that some irrelevant details are omitted. See Beierle *et al.* [3] and Banik *et al.* [1] for more information about the ciphers. Following that, the main idea of the differential attack in this paper is introduced.

2.1 Specification of `CRAFT`

`CRAFT` [3] is a lightweight block cipher with 64-bit block, 128-bit key, and 64-bit tweak. In the encryption phase, the internal state is viewed as a 4×4 square array of nibbles, and the nibble located at the i-th row and the j-th column is denoted as $I_{i,j}$, where $0 \leqslant i, j \leqslant 3$. Also, the 4×4 square array can be seen as a vector by concatenating the rows, and I_ℓ stands for the ℓ-th nibble of the vector in this expression, where $0 \leqslant \ell \leqslant 15$. With these definitions, the equation $I_{i,j} = I_{4 \cdot i+j}$ holds.

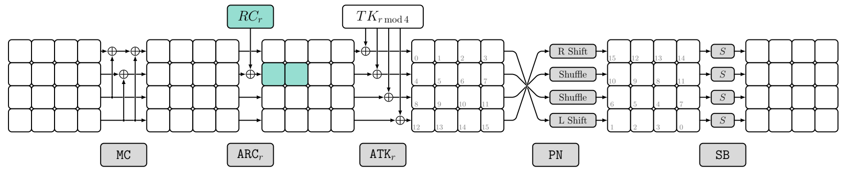


Fig. 1. Full round function $\mathcal{R}_r$ of `CRAFT`.

After initialising the state with the plaintext, the cipher iterates 31 full round functions ($\mathcal{R}_r$, $0 \leqslant r \leqslant 30$) and appends one linear round function ($\mathcal{R}'_{31}$) to yield the ciphertext. Each $\mathcal{R}_r$ applies the following five involutory operations, while $\mathcal{R}'_{31}$ only involves the first three operations. An illustration of $\mathcal{R}_r$ can be found in Figure 1.

`MixColumn(MC)` Each column of the state is multiplied with the following involutory matrix

$$M = \begin{bmatrix} \texttt{0x1} & \texttt{0x0} & \texttt{0x1} & \texttt{0x1} \\ \texttt{0x0} & \texttt{0x1} & \texttt{0x0} & \texttt{0x1} \\ \texttt{0x0} & \texttt{0x0} & \texttt{0x1} & \texttt{0x0} \\ \texttt{0x0} & \texttt{0x0} & \texttt{0x0} & \texttt{0x1} \end{bmatrix}.$$

`AddConstants`$_r$`(ARC`$_r$`)` The round constants are generated with one 4-bit linear feedback shift register (LFSR) and one 3-bit LFSR, whose states are denoted by $a = (a_0, a_1, a_2, a_3)$ and $b = (b_0, b_1, b_2)$. In each round, $a_0 \| a_1 \| a_2 \| a_3$ and $0 \| b_0 \| b_1 \| b_2$ are firstly XORed with the state nibbles $I_{1,0}$ and $I_{1,1}$. After that, the two LFSRs get updated.

`AddTweakey`$_r$`(ATK`$_r$`)` The 128-bit key K is split into two 64-bit keys K_0 and K_1. The formation of tweakeys involves the application of the permutation $\mathcal{Q} = (12, 10, 15, 5,\ 14, 8, 9, 2,\ 11, 3, 7, 4,\ 6, 0, 1, 13)$ on the sixteen nibbles of the 64-bit tweak T, and the i-th nibble of $\mathcal{Q}(T)$ equals the $\mathcal{Q}(i)$-th nibble of T, where $0 \leqslant i \leqslant 15$. Then, four 64-bit tweakeys TK_0, TK_1, TK_2, and TK_3 are derived as

$$TK_0 = K_0 \oplus T,\ TK_1 = K_1 \oplus T,\ TK_2 = K_0 \oplus \mathcal{Q}(T),\ TK_3 = K_1 \oplus \mathcal{Q}(T).$$

In the r-th round, the tweakey $TK_{r \bmod 4}$ is XORed with the cipher state.

`PermuteNibbles(PN)` The involutory permutation $\mathcal{P}$ is utilised to update the nibble positions of the state, i.e., $I_i \leftarrow I_{\mathcal{P}(i)}$ for all $0 \leqslant i \leqslant 15$, where $\mathcal{P} = (15, 12, 13, 14,\ 10, 9, 8, 11,\ 6, 5, 4, 7,\ 1, 2, 3, 0)$.

`SubBox(SB)` The S-box S in this operation is the same as the one exploited in `Midori` [2] and is applied to each nibble of the state. The table for S is given in the following.

x	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xa	0xb	0xc	0xd	0xe	0xf
$S(x)$	0xc	0xa	0xd	0x3	0xe	0xb	0xf	0x7	0x8	0x9	0x1	0x5	0x0	0x2	0x4	0x6

2.2 Description of `WARP`

The overall structure of `WARP` [1] is a variant of the 32-branch Type-2 Generalised Feistel Network (GFN) [27]. It encrypts the 128-bit plaintext $\mathcal{M}$ via the 128-bit key $\mathcal{K}$. Note that $\mathcal{K}$ is denoted as the concatenation of two 64-bit keys $\mathcal{K}_0$ and $\mathcal{K}_1$ as $\mathcal{K} = \mathcal{K}_0 \| \mathcal{K}_1$. Further, $\mathcal{K}_0$ and $\mathcal{K}_1$ are expressed as 16 nibbles, that is, $\mathcal{K}_0 = \mathcal{K}_0[0] \| \mathcal{K}_0[1] \| \cdots \| \mathcal{K}_0[15]$ and $\mathcal{K}_1 = \mathcal{K}_1[0] \| \mathcal{K}_1[1] \| \cdots \| \mathcal{K}_1[15]$, where $\mathcal{K}_i[j] \in \mathbb{F}_2^4$, $i \in \{0, 1\}$, and $j \in \{0, 1, \ldots, 15\}$.

In the encryption phase, the plaintext $\mathcal{M}$ is loaded into a 128-bit internal state $\mathcal{X}_0$, which is represented in nibbles as $\mathcal{X}_0 = \mathcal{X}_0[0] \| \mathcal{X}_0[1] \| \cdots \| \mathcal{X}_0[31]$. As in Figure 15, the round function of `WARP` is composed of the following operations:

- ▷ 16 applications of the S-box S, which also is the same as the one in `Midori`, with outputs being $\mathcal{Z}_r[i] \triangleq S(\mathcal{X}_r[2\cdot i])$, $0 \leqslant i \leqslant 15$;
- ▷ 16 XOR operations, with outputs being $\mathcal{W}_r[i] \triangleq \mathcal{Z}_r[i] \oplus \mathcal{K}_{r \bmod 2}[i]$, $0 \leqslant i \leqslant 15$;
- ▷ 16 XOR operations, with outputs being $\mathcal{Y}_r[2\cdot i+1] \triangleq \mathcal{W}_r[i] \oplus \mathcal{X}_r[2\cdot i+1]$, $0 \leqslant i \leqslant 15$;
- ▷ two XOR operations, with outputs being $\mathcal{Y}_r[2\cdot i+1] \leftarrow \mathcal{Y}_r[2\cdot i+1] \oplus RC_r[i]$, where $RC_r[i]$'s are constants and $i \in \{0,1\}$;
- ▷ a permutation π, which is listed in the following, applied to 32 nibbles $\mathcal{Y}_r[0]$, $\mathcal{Y}_r[1]$, ..., $\mathcal{Y}_r[31]$, where $\mathcal{Y}_r[2\cdot i] \triangleq \mathcal{X}_r[2\cdot i]$ for all $0 \leqslant i \leqslant 15$, with outputs being $\mathcal{X}_{r+1}[\pi(j)] = \mathcal{Y}_r[j]$ for all $0 \leqslant j \leqslant 31$.

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\pi(j)$	31	6	29	14	1	12	21	8	27	2	3	0	25	4	23	10
j	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$\pi(j)$	15	22	13	30	17	28	5	24	11	18	19	16	9	20	7	26

The round function is iterated 41 times from $r = 0$ to 40, and the nibble shuffle operation π is skipped in the last round.

2.3 Differential Attacks with High-Probability Characteristics

Differential cryptanalysis [5] studies how an input difference propagates through the cipher. For the iterated cipher, a *differential characteristic* describes not only the input and output differences but also the internal differences after every round function. The *right pair* for the given differential characteristic should fulfil all differences defined in the characteristic. The proportion of right pairs to all pairs validating the input difference is the *differential probability* of the characteristic. In contrast to differential characteristics, a *differential* only clarifies the input and output differences and should comprise all differential characteristics with input and output differences identical to those of the differential.

The first step in the differential attack is to find differential distinguishers with relatively high probabilities. In theory [18], the distinguisher should be differentials. However, searching for all characteristics and evaluating the differential probability of the differential are not manageable tasks in most cases. Hence, a standard measure is finding some dominating characteristics with significant differential probabilities and employing the sum of these probabilities to approximate the probability of the differential.

As in Figure 2(a), after constructing a round-reduced differential distinguisher, the general key-recovery attack implements the following procedures.

1. Appending several rounds before and after the distinguisher.
2. Collecting multiple pairs of plaintexts with specific differences.
3. Enumerating subkeys involved in partial encryption and decryption phases.
4. Performing the statistical hypothesis testing to sieve subkey candidates.
5. Testing the surviving key candidates with plaintext-ciphertext pairs.

It can be noticed that the general method only exploits the input and output differences of the distinguisher. However, as illustrated in previous literature [12,10,25], given right pairs satisfying the known differential characteristic, more messages about the right pair can be inferred from the internal differences and can be employed to launch more efficient attacks. The differential attacks in this paper are based on this observation.

We start with the discussion on the S-box. Given a possible differential propagation $\delta_{\mathtt{i}} \to \delta_{\mathtt{o}}$ for the S-box S, we can calculate all right pairs validating this propagation. Denote $\mathbb{I}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$ (resp., $\mathbb{O}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$) the set of input (resp., output) values of right pairs. As an instance, for the S-box of `CRAFT` and `WARP`, we have $\mathbb{I}_S(\mathtt{0x5}, \mathtt{0x7}) = \{\mathtt{0x0}, \mathtt{0x5}, \mathtt{0xa}, \mathtt{0xf}\}$ and $\mathbb{O}_S(\mathtt{0x5}, \mathtt{0x7}) = \{\mathtt{0x1}, \mathtt{0x6}, \mathtt{0xb}, \mathtt{0xc}\}$. The right pairs for other possible propagations can be found in Appendix A.

The main idea of the differential attack in this work is shown in Figure 2(b). Firstly, we search for a differential characteristic with a high probability p and confirm that the differential containing this characteristic does not have other significant characteristics with a probability greater than p. Then, $\mathcal{O}(p^{-1})$ random pairs are queried to find right pairs for the differential characteristic. Suppose that the characteristic activates an S-box with propagation $\delta_{\mathtt{i}} \to \delta_{\mathtt{o}}$ in a particular round; the values of the right pair at the input and output of the active S-box are known. Based on this message, we perform subkey enumeration inside the differential distinguisher and check whether the values of the right pair at the input of the active S-box fall into the set $\mathbb{I}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$. The key guess will be accepted if the condition is met by each right pair. Additionally, if more than one active S-box exists in the differential characteristic, this procedure can be done concerning other active S-boxes (cf. Figure 2(c)).

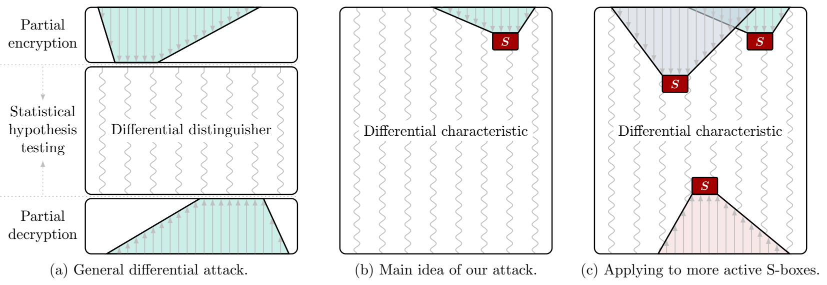

Partial encryption
Statistical hypothesis testing
Differential distinguisher
Partial decryption
(a) General differential attack.
S
Differential characteristic
(b) Main idea of our attack.
S
S
Differential characteristic
S
(c) Applying to more active S-boxes.

Fig. 2. Methods to implementing differential attacks.

3 Practical Related-Key Differential Attack on `CRAFT`

In this section, we first check the differential properties of `CRAFT` in the related-key and related-tweakey settings. After that, a practical key-recovery attack on `CRAFT` is introduced.

Table 2. Upper bounds on differential probabilities of `CRAFT` in RK and RTK settings.

Round	1	2	3	4	5	6	7	8	9	10	11
$-\log_2(\mathrm{P}_{\mathsf{RK}}(r))$	0	0	2	4	4	6	6	8	8	10	10
$-\log_2(\mathrm{P}_{\mathsf{RTK}}(r))$	0	0	0	2	4	4	6	8	8	8	10
Round	12	13	14	15	16	17	18	19	20	21	22
$-\log_2(\mathrm{P}_{\mathsf{RK}}(r))$	12	12	14	14	16	16	18	18	20	20	22
$-\log_2(\mathrm{P}_{\mathsf{RTK}}(r))$	12	12	12	14	16	16	16	18	20	20	20
Round	23	24	25	26	27	28	29	30	31	32	-
$-\log_2(\mathrm{P}_{\mathsf{RK}}(r))$	22	24	24	26	26	28	28	30	30	30	-
$-\log_2(\mathrm{P}_{\mathsf{RTK}}(r))$	22	24	24	24	26	28	28	28	30	30	-

$\mathrm{P}_{\mathsf{RK}}(r)$ is the maximum differential probability for r-round characteristics in the RK setting.
$\mathrm{P}_{\mathsf{RTK}}(r)$ is the maximum differential probability for r-round characteristics in the RTK setting.

3.1 Related-(Twea)Key Differential Properties of `CRAFT`

In [13], the authors created several iterative 2-round related-key differential characteristics for `CRAFT`. These 2-round characteristics were used to construct 28-round differential characteristics with probability 2^{-28}, which enable the authors to realise the full-round attack. However, if only one characteristic is utilised, the time complexity is 2^{85}, and the attack is unrealistic. Given the extremely high probability of the distinguisher, we wonder about the possibility of implementing a practical key-recovery attack on `CRAFT`.

Motivated by this issue, we first search for the complete upper bounds on the differential probability of `CRAFT` in the related-key (RK) and related-tweakey (RTK) settings. The automatic searching method in [25] is applied, and the test results are listed in Table 2. Note that there are full-round differential characteristics with probability 2^{-30} in both settings. Then, we adopt the approach in [25] to find out all full-round differential characteristics in the two settings.

In the RK setting, 384 full-round differential characteristics are returned and can be divided into sixteen groups, denoted as $\mathbb{G}_0^{\mathsf{RK}}$, $\mathbb{G}_1^{\mathsf{RK}}$, …, and $\mathbb{G}_{15}^{\mathsf{RK}}$. Each group is composed of 24 characteristics, and all characteristics in the same group follow the same differential pattern. Note that all of the sixteen full-round differential patterns are iterative. The differential pattern in $\mathbb{G}_7^{\mathsf{RK}}$ is illustrated in Figure 3, where $\delta_{\mathtt{i}}$ and $\delta_{\mathtt{o}}$ are nonzero differences, and $\delta_{\mathtt{i}} \rightarrow \delta_{\mathtt{o}}$ should be a possible differential propagation for the S-box with probability 2^{-2}. The remaining fifteen differential patterns can be found in the Supplementary Material[4]. These results are in accordance with the analysis in [13].

In the RTK setting, we obtain 7680 32-round differential characteristics. These characteristics can be partitioned into 28 groups, and we use $\mathbb{G}_0^{\mathsf{RTK}}$, $\mathbb{G}_1^{\mathsf{RTK}}$, …, $\mathbb{G}_{27}^{\mathsf{RTK}}$ to represent them. All characteristics in the same group share the

[4] https://github.com/SunLing134340/CRAFT-and-WARP

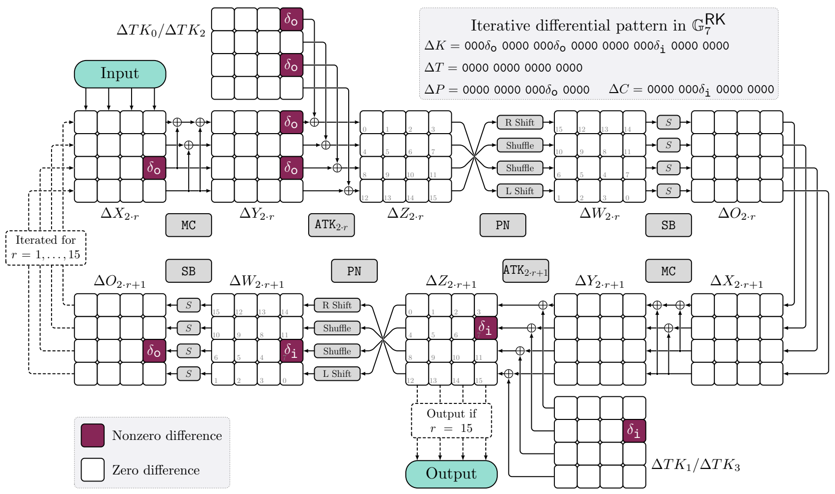

Iterative differential pattern in $\mathbb{G}_7^{\mathsf{RK}}$
$\Delta K = 000\delta_{\mathtt{o}}\ 0000\ 000\delta_{\mathtt{o}}\ 0000\ 0000\ 000\delta_{\mathtt{i}}\ 0000\ 0000$
$\Delta T = 0000\ 0000\ 0000\ 0000$
$\Delta P = 0000\ 0000\ 000\delta_{\mathtt{o}}\ 0000$
$\Delta C = 0000\ 000\delta_{\mathtt{i}}\ 0000\ 0000$
$\Delta TK_0/\Delta TK_2$
Input
$\Delta X_{2\cdot r}$
$\Delta Y_{2\cdot r}$
$\Delta Z_{2\cdot r}$
$\Delta W_{2\cdot r}$
$\Delta O_{2\cdot r}$
MC
ATK$_{2\cdot r}$
PN
SB
R Shift
Shuffle
Shuffle
L Shift
S
Iterated for $r = 1, \ldots, 15$
$\Delta O_{2\cdot r+1}$
$\Delta W_{2\cdot r+1}$
$\Delta Z_{2\cdot r+1}$
$\Delta Y_{2\cdot r+1}$
$\Delta X_{2\cdot r+1}$
ATK$_{2\cdot r+1}$
Output if $r = 15$
Output
$\Delta TK_1/\Delta TK_3$
Nonzero difference
Zero difference

Fig. 3. Iterative differential pattern exploited in $\mathbb{G}_7^{\mathsf{RK}}$. We omit the $\mathtt{ARC}_r$ operation.

same differential pattern. Each of the first sixteen groups $\mathbb{G}_0^{\mathsf{RTK}}$, $\mathbb{G}_1^{\mathsf{RTK}}$, $\ldots$, $\mathbb{G}_{15}^{\mathsf{RTK}}$ is composed of 384 differential characteristics, and all of the 6144 differential characteristics are generated with 2-round iterative characteristics. As an illustration, the differential pattern in $\mathbb{G}_7^{\mathsf{RTK}}$ is provided in Figure 9. The remaining fifteen differential patterns can be found in the Supplementary Material. It can be noticed that for all $0 \leqslant i \leqslant 15$, the differential pattern in $\mathbb{G}_i^{\mathsf{RK}}$ corresponds to the special case of the differential pattern in $\mathbb{G}_i^{\mathsf{RTK}}$, where the tweak has no difference. In each of the remaining twelve groups $\mathbb{G}_{16}^{\mathsf{RTK}}$, $\mathbb{G}_{17}^{\mathsf{RTK}}$, $\ldots$, $\mathbb{G}_{27}^{\mathsf{RTK}}$, there are 128 characteristics. These differential characteristics also utilise iterative structures, while the number of iterative rounds is four. Figure 10 exhibits the differential pattern in $\mathbb{G}_{16}^{\mathsf{RTK}}$. The differential patterns in $\mathbb{G}_{17}^{\mathsf{RTK}}$, $\mathbb{G}_{18}^{\mathsf{RTK}}$, $\ldots$, $\mathbb{G}_{27}^{\mathsf{RTK}}$ are given in Supplementary Material. As far as we know, we are the first to give the differential characteristic in the RTK setting.

3.2 Practical Key-Recovery Attack on CRAFT

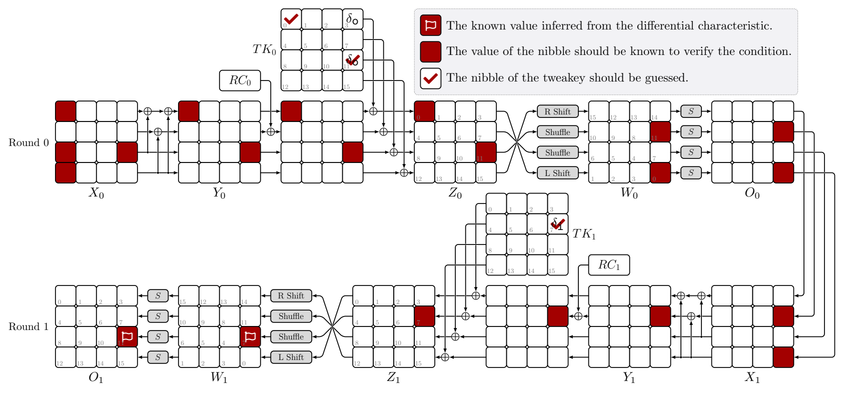
Now, we aim at a practical key-recovery attack with one differential characteristic in the RK setting. The differential characteristics operated in the following discussion belongs to $\mathbb{G}_7^{\mathsf{RK}}$, and we fix $\delta_{\mathtt{i}} = \mathtt{0x5}$ and $\delta_{\mathtt{o}} = \mathtt{0x7}$. With the automatic method [25], we verify that there is no characteristic with a probability larger than 2^{-55} in the same differential apart from the optimal characteristic with probability 2^{-30}.

To facilitate the key-recovery attack, we first query N random pairs of plaintexts (P, P') validating $P \oplus P' = \Delta P = \mathtt{0x0000\ 0x0000\ 0x0007\ 0x0000}$. On average, $N_R = N \cdot 2^{-30}$ right pairs can be identified. So, the data requirement of the attack is $2 \cdot N$ chosen plaintexts. After obtaining right pairs, we execute the subsequent procedures to recover the values of the pair (K, K'). In the following, we use T to denote the tweak. The pair of tweakeys are represented with (TK_0, TK_1, TK_2, TK_3) and $(TK'_0, TK'_1, TK'_2, TK'_3)$. Note that the value of T is known in the attack. Once the j-th nibble of TK_i is recovered, the value of $K_{i \bmod 2}[j]$ can be computed with $TK_i[j]$ and T.

Step 1: recovering partial keys in the first two rounds. Note that the right pair must follow the differential characteristic in Figure 3. Since the unique active S-box in the second round propagates the input difference $\delta_{\mathtt{i}}$ to the output difference $\delta_{\mathtt{o}}$, the values of the right pair at $W_1[11]$ should fall into the set $\mathbb{I}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$. These known values enable us to recover the values of three nibbles of $TK_0 \| TK_1$.

An illustration of this step can be found in Figure 4. To check the value at $W_1[11]$, we enumerate the value of the 12-bit tweakey $TK_0[0, 11] \| TK_1[7]$. With the known information on the related-tweakey, the values of $TK'_0[0] = TK_0[0]$, $TK'_0[11] = TK_0[11] \oplus \delta_{\mathtt{o}}$, and $TK'_1[7] = TK_1[7] \oplus \delta_{\mathtt{i}}$ are obtained. For each right pair (P, P'), partial encryption is performed, and we check whether $W_1[11]$ and $W'_1[11]$ belong to the set $\mathbb{I}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$. The guess for the tweakey will be accepted as a candidate if the verification is passed for all right pairs.

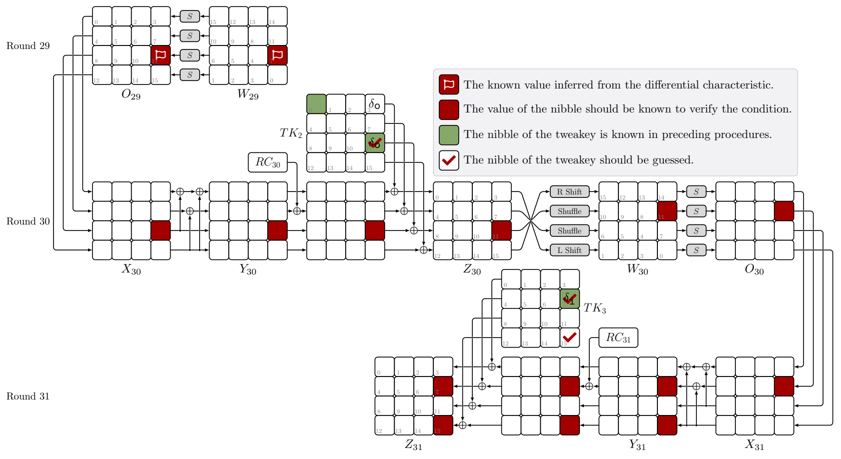
Observing the differential characteristic (cf. Figure 3), we find that the two predictions $W_1[11] \in \mathbb{I}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$ and $W'_1[11] \in \mathbb{I}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$ should be true or false simultaneously. Since the propagation $\delta_{\mathtt{i}} \to \delta_{\mathtt{o}}$ holds with probability 2^{-2}, the set $\mathbb{I}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$ contains four elements. So, given the right pair, the probability that the random guess for the tweakey validates $W_1[11] \in \mathbb{I}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$ is 2^{-2}. Accordingly, on average, we can get the correct value for the tweakey with $N_1 = 6$ right pairs. With the value of T, we retrieve the value of the 12-bit key $K_0[0, 11] \| K_1[7]$.



The known value inferred from the differential characteristic.
The value of the nibble should be known to verify the condition.
The nibble of the tweakey should be guessed.
TK_0
$\delta_{\mathtt{o}}$
RC_0
Round 0
X_0
Y_0
Z_0
W_0
O_0
R Shift
Shuffle
Shuffle
L Shift
S
TK_1
RC_1
Round 1
O_1
W_1
Z_1
Y_1
X_1

Fig. 4. The first step of the attack on `CRAFT`.

Step 2: recovering partial keys in the last two rounds. After investigating the active S-box in the second round, we move to the active S-box in the 29-th round. From Figure 3, the unique active S-box in the 29-th round follows the propagation $\delta_{\mathtt{i}} \to \delta_{\mathtt{o}}$. Therefore, the values of the right pair at $O_{29}[11] = X_{30}[11]$ should belong to the set $\mathbb{O}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$. Figure 5 displays the key-recovery procedure in the last two rounds. Note that the value of $TK_2[0, 11]\|TK_3[7]$ can be derived from the result in the first step, and we only need to enumerate the value of the 4-bit tweakey $TK_3[15]$. For each right pair (P, P'), we check the validity of the condition $X_{30}[11]/X'_{30}[11] \in \mathbb{O}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$. The guess for the tweakey will be saved as a candidate if the verification is passed for all right pairs. The correct tweakey guess can be identified with $N_2 = 2$ right pairs based on a similar analysis as the first step, After that, the value of $K_1[15]$ can be computed with the known tweak T.



Round 29
Round 30
Round 31
O_{29}
W_{29}
TK_2
$\delta_{\mathtt{o}}$
RC_{30}
X_{30}
Y_{30}
Z_{30}
W_{30}
O_{30}
R Shift
Shuffle
Shuffle
L Shift
S
TK_3
RC_{31}
Z_{31}
Y_{31}
X_{31}
The known value inferred from the differential characteristic.
The value of the nibble should be known to verify the condition.
The nibble of the tweakey is known in preceding procedures.
The nibble of the tweakey should be guessed.

Fig. 5. The second step of the attack on `CRAFT`.

Step 3: recovering partial keys in the first four rounds. In this step, we go back to the head of the cipher and implement key-recovery regarding the first four rounds. The study centres on the known value of $W_3[11]$, and Figure 11 shows the key-recovery procedure. With the previous analysis, the value of $TK_0[0, 11]\|TK_1[7, 15]\|TK_2[0, 11]\|TK_3[7, 15]$ is known. We enumerate the value of the 28-bit tweakey $TK_0[2, 3, 5, 8, 12]\|TK_1[1, 6]$ and check whether each right pair (P, P') fulfils the condition $W_3[11] \in \mathbb{I}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$. On average, with $N_3 = 14$ right pairs, we uncover the correct guess of the tweakey. Then, the value of $K_0[2, 3, 5, 8, 12]\|K_1[1, 6]$ can be recovered.

Step 4: recovering partial keys in the last four rounds. To control the time complexity of the attack, we focus on the tail of the cipher in this step. As Figure 12, the analysis is based on the known values of the right pair at $O_{27}[11] = X_{28}[11]$. $TK_0[0, 2, 3, 5, 8, 11, 12]\|TK_1[1, 6, 7, 15]\|TK_2[0, 2, 3, 5, 8, 11, 12]\|TK_3[1, 6, 7, 15]$ is known, and we enumerate the value of the 12-bit tweakey $TK_3[9, 13, 14]$ and check the validity of the condition $X_{28}[11] \in \mathbb{O}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$ for each right pair. The correct guess of the tweakey can be found with approximately $N_4 = 6$ right pairs. Following that, we calculate the value of $K_1[9, 13, 14]$.

Step 5: recovering partial keys in the last six rounds. In this step, we still pay attention to the tail of the cipher and manipulate the known values of the right pair at $O_{25}[11] = X_{26}[11]$. To keep the key-recovery attack practical, we interchange the order of $\mathtt{ATK}_r$ operation and $\mathtt{ARC}_r \circ \mathtt{MC}$ operation for $r \geqslant 30$. Correspondingly, the tweakey involved in $\mathtt{ATK}_r$ operation is replaced from $TK_{r \bmod 4}$ to $ETK_{r \bmod 4} \triangleq M(TK_{r \bmod 4})$ so that the correctness of the encryption can be guaranteed.

The key-recovery procedure can be found in Figure 13. Note that the value of $TK_2[0, 2, 3, 5, 8, 11, 12] \| TK_3[1, 6, 7, 9, 13\text{-}15]$ is known. By the equation between $TK_{r \bmod 4}$ and $ETK_{r \bmod 4}$, we derive $ETK_2[0, 8, 11, 12] \| ETK_3[1, 6, 7, 9, 13\text{-}15]$

$$\begin{aligned}
ETK_2[0] &= TK_2[0] \oplus TK_2[8] \oplus TK_2[12], \\
ETK_2[j] &= TK_2[j] \text{ for } j \in \{8, 11, 12\}, \\
ETK_3[1] &= TK_3[1] \oplus TK_3[9] \oplus TK_3[13], \\
ETK_3[j] &= TK_3[j] \oplus TK_3[j+8] \text{ for } j \in \{6, 7\}, \\
ETK_3[j] &= TK_3[j] \text{ for } j \in \{9, 13, 14, 15\}.
\end{aligned}$$

In order to compute the value of $X_{26}[11]$, we should guess the value of the 28-bit tweakey $ETK_2[2, 3, 5] \| ETK_3[0, 2\text{-}4]$. If the condition $X_{26}[11] \in \mathbb{O}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$ holds by each right pair, the guess for the tweakey will be accepted as a candidate. Therefore, with about $N_5 = 14$ right pairs, we recognise the correct guess. Based on the retrieved value of $ETK_2[3, 5]$, the value of $TK_2[13] = TK_2[5] \oplus ETK_2[5]$ and $TK_2[15] = TK_2[3] \oplus TK_2[11] \oplus ETK_2[3]$ can be determined. Consequently, we recover the value of $K_0[13, 15]$. Beyond that, with the value of $ETK_2[2] \| ETK_3[0, 2\text{-}4]$, we develop the following 20-bit information about the unknown value of the tweakey

$$\begin{aligned}
TK_2[10] \oplus TK_2[14] &= TK_2[2] \oplus ETK_2[2], \\
TK_3[0] \oplus TK_3[8] \oplus TK_3[12] &= ETK_3[0], \\
TK_3[2] \oplus TK_3[10] &= TK_3[14] \oplus ETK_3[2], \\
TK_3[3] \oplus TK_3[11] &= TK_3[15] \oplus ETK_3[3], \\
TK_3[4] \oplus TK_3[12] &= ETK_3[4].
\end{aligned}$$

These messages allow us to guess fewer bits in the next step.

Step 6: recovering partial keys in the first six rounds. In this step, we go back to the head of the cipher and utilise the known values of the right pairs at $W_5[11]$. The key-recovery procedure is exhibited in Figure 14. For now, we know the value of $K_0[0, 2, 3, 5, 8, 11\text{-}13, 15] \| K_1[1, 6, 7, 9, 13\text{-}15]$. To compute the value of $W_5[11]$, we enumerate the value of the 36-bit tweakey $TK_0[1, 4, 6, 7, 10] \| TK_1[0, 2\text{-}4]$. As the value of $TK_2[10] \oplus TK_2[14]$ is uncovered in the fifth step, the value of $TK_0[14]$ is determined after guessing the value of $TK_0[10]$. Then, for each right pair, we inspect the validity of the condition $W_5[11] \in \mathbb{I}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$. On average, with $N_6 = 18$ right pairs, we can locate the correct guess of the tweakey. After that, we get the value of the 40-bit key $K_0[1, 4, 6, 7, 10, 14] \| K_1[0, 2\text{-}4]$.

Step 7: exhaustively searching for the remaining keys. After executing the preceding steps, the value of the 8-bit key $K_0[9] \| K_1[5]$ is unexplored. We exhaustively search for the value of $K_0[9] \| K_1[5]$ with $N_7 = 1$ right pair.

Theoretical complexity analysis. To ensure the success of each step, the number of right pairs for this attack should be no less than $N_R = \max\{N_i \mid 0 \leqslant i \leqslant 7\} = 18$. Accordingly, the data complexity of this attack is about $2 \cdot 18 \cdot 2^{30} \approx 2^{35.17}$ chosen plaintexts. The time complexity T_1 to collect plaintext-ciphertext pairs is about $2^{35.17}$ full-round encryptions. Apart from that, the time complexity T_2 of the sixth step dominates the key-recovery phase. Once the verification condition $W_5[11] \in \mathbb{I}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$ is not passed for at least one right pair, the tweakey guess is dropped. Therefore, T_2 is bounded by $2 \cdot \sum_{i=0}^{17} 2^{36-2 \cdot i}$ 6-round encryptions. Approximately, we have $T_2 = 2^{35}$ full-round encryptions. Overall, the time complexity of this attack is $2^{36.09}$. The memory complexity of this attack is negligible.

Practical runtime. Since the complexity of the attack is practical, we try to confirm the attack in practise. All tests are realised on one AMD EPYC 7302 16-Core Processor, and all programs use a single thread. To begin with, we fix $K \| T$ with the following randomly selected values

$$K = \mathtt{0x6c6b\ 0xd022\ 0x4dc5\ 0x65e3\ 0x1d4d\ 0x5753\ 0xa30f\ 0xa6d8},$$
$$T = \mathtt{0x08a2\ 0x3a1b\ 0x40f6\ 0x376f}.$$

With 2^{35} randomly generated pairs of chosen plaintexts, we collect 49 right pairs for the full-round differential characteristic in $\mathbb{G}_7^{\mathsf{RK}}$ with $\delta_{\mathtt{i}} = \mathtt{0x5}$ and $\delta_{\mathtt{o}} = \mathtt{0x7}$. The runtime for this approach is about 9.2 hours. Although we find more than $N_R = 18$ right pairs, only 18 right pairs are employed in the subsequent tests. The output and the runtime for the seven steps are listed in Table 3. It can be noticed that we cannot determine the unique correct candidate in some steps. We think the reason is that the tweakey is XORed before the S-box operation sometimes. In this case, if the input set $\mathbb{I}_S(\delta_{\mathtt{i}}, \delta_{\mathtt{o}})$ is taken in the verification condition, more than one tweakey candidate can validate the condition. The good thing is that the auxiliary candidates are sieved in the following steps. At last, we catch the unique correct key. To sum up, the practical time complexity of the attack is about 15.69 hours. All source codes can be found in the Supplementary Material.

Remark 1. Although we specify $\delta_{\mathtt{i}} = \mathtt{0x5}$ and $\delta_{\mathtt{o}} = \mathtt{0x7}$ in the attack, each of the remaining 23 characteristics in $\mathbb{G}_7^{\mathsf{RK}}$ can facilitate a practical key-recovery attack in RK setting.

Table 3. Test results about CRAFT.

Step	1	2	3	4	5	6	7	Total
Runtime	0.001s	0.001s	189.30s	0.001s	0.39h	6.05h	0.001s	6.49h
#{Candidates}	4	4	1	16	1	1	1	1

#{Candidates} is the number of tweakey candidates after conducting each step.

Remark 2. We also try characteristics in $\mathbb{G}_i^{\mathsf{RK}}$ with $i \neq 7$ and find that the time complexity might be impractical with characteristics in $\mathbb{G}_0^{\mathsf{RK}}$, $\mathbb{G}_1^{\mathsf{RK}}$, $\mathbb{G}_2^{\mathsf{RK}}$, $\mathbb{G}_3^{\mathsf{RK}}$, $\mathbb{G}_{12}^{\mathsf{RK}}$, $\mathbb{G}_{13}^{\mathsf{RK}}$, $\mathbb{G}_{14}^{\mathsf{RK}}$, and $\mathbb{G}_{15}^{\mathsf{RK}}$.

4 Related-Key Differential Attack on WARP

After realising the practical related-key differential attack on CRAFT, we are curious about the differential property of WARP in the related-key attack setting since it applies the same S-box as CRAFT.

4.1 Related-Key Differential Property of WARP

Table 4. Upper bounds on differential probabilities of WARP in the RK setting.

Round	1	2	3	4	5	6	7	8	9	10	11
$-\log_2(\mathrm{P}_{\mathsf{RK}}(r))$	0	0	0	2	4	6	6	8	8	10	10
Round	12	13	14	15	16	17	18	19	20	21	22
$-\log_2(\mathrm{P}_{\mathsf{RK}}(r))$	12	12	14	14	16	16	18	18	20	20	22
Round	23	24	25	26	27	28	29	30	31	32	33
$-\log_2(\mathrm{P}_{\mathsf{RK}}(r))$	22	24	24	26	26	28	28	30	30	32	32
Round	34	35	36	37	38	39	40	41	-	-	-
$-\log_2(\mathrm{P}_{\mathsf{RK}}(r))$	34	34	36	36	38	38	40	40	-	-	-

$\mathrm{P}_{\mathsf{RK}}(r)$ is the maximum differential probability for r-round characteristics in the RK setting.

We exploit the automatic searching method in [25] to explore the optimal related-key differential characteristics for WARP, and Table 4 presents the test result. Similarly to the case in CRAFT, there is a full-round related-key differential characteristic with a high probability, say 2^{-40}. With a further study, we discover 384 41-round differential characteristics with a probability of 2^{-40} in the RK setting. These characteristics can be divided into sixteen groups, which are denoted as $\mathbb{G}_0^{\mathtt{WARP}}$, $\mathbb{G}_1^{\mathtt{WARP}}$, ..., and $\mathbb{G}_{15}^{\mathtt{WARP}}$. Each group consists of 24 characteristics, and all characteristics in the same group follow the same differential pattern. All of the sixteen full-round differential patterns are iterative. The differential pattern in $\mathbb{G}_6^{\texttt{WARP}}$ is demonstrated in Figure 6, where $\delta_{\texttt{i}}$ and $\delta_{\texttt{o}}$ are nonzero differences, and $\delta_{\texttt{i}} \to \delta_{\texttt{o}}$ should be a possible differential propagation for the S-box with probability 2^{-2}. The remaining fifteen differential patterns can be found in the Supplementary Material.

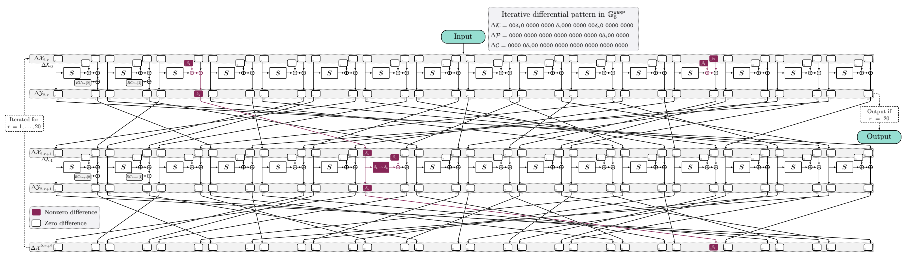

Iterative differential pattern in $\mathbb{G}_6^{\texttt{WARP}}$
$\Delta\mathcal{K} = 00\delta_{\texttt{i}}0\ 0000\ 0000\ \delta_{\texttt{i}}000\ 0000\ 00\delta_{\texttt{o}}0\ 0000\ 0000$
$\Delta\mathcal{P} = 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0\delta_{\texttt{i}}00\ 0000$
$\Delta\mathcal{C} = 0000\ 0\delta_{\texttt{i}}00\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000$
Input
Output
Iterated for $r = 1,\ldots,20$
Output if $r = 20$
Nonzero difference
Zero difference

Fig. 6. Iterative differential pattern used in $\mathbb{G}_6^{\texttt{WARP}}$

4.2 Key-Recovery Attack on WARP

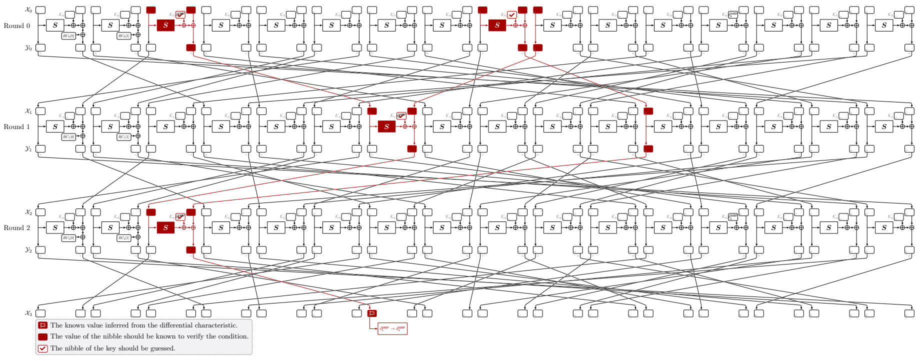
Now, we discuss the key-recovery attack on the full-round WARP with one characteristic in the RK setting. The differential characteristic belongs to $\mathbb{G}_6^{\texttt{WARP}}$, and we fix $\delta_{\texttt{i}}^{\texttt{WARP}} = \texttt{0xa}$ and $\delta_{\texttt{o}}^{\texttt{WARP}} = \texttt{0x5}$. With the automatic method, we confirm that there is no characteristic with a probability higher than 2^{-70} in the same differential apart from the optimal characteristic with probability 2^{-40}.

To implement the key-recovery attack, we first query $N^{\texttt{WARP}}$ random pairs of plaintexts $(\mathcal{P}, \mathcal{P}')$ with

$$\mathcal{P} \oplus \mathcal{P}' = \texttt{0x0000 0x0000 0x0000 0x0000 0x0000 0x0000 0x0a00 0x0000}.$$

Approximately $N_R^{\texttt{WARP}} = N^{\texttt{WARP}} \cdot 2^{-40}$ right pairs can be obtained. Thus, the data complexity of the attack is $2 \cdot N^{\texttt{WARP}}$ chosen plaintexts. After collecting right pairs, we conduct the following steps to retrieve the values of the pair $(\mathcal{K}, \mathcal{K}')$.

Step 1: recovering partial keys in the first three rounds. The right pair must follow the differential characteristic in Figure 6. As the unique active S-box in the fourth round propagates the input difference $\delta_{\texttt{i}}^{\texttt{WARP}}$ to the output difference $\delta_{\texttt{o}}^{\texttt{WARP}}$, the values of the right pair at $\mathcal{X}_3[12]$ should fall into the set $\mathbb{I}_S(\delta_{\texttt{i}}^{\texttt{WARP}}, \delta_{\texttt{o}}^{\texttt{WARP}})$. These known values enable us to recover the values of three nibbles in $\mathcal{K} = \mathcal{K}_0 \| \mathcal{K}_1$. The key-recovery procedure of this step is illustrated in Figure 7. To compute the value at $\mathcal{X}_3[12]$, we guess the value of the 12-bit key $\mathcal{K}_0[2,8] \| \mathcal{K}_1[6]$. For each right pair $(\mathcal{P}, \mathcal{P}')$, we check whether $\mathcal{X}_3[12]$ belongs to the set $\mathbb{I}_S(\delta_{\texttt{i}}^{\texttt{WARP}}, \delta_{\texttt{o}}^{\texttt{WARP}})$. The key guess will be accepted as a candidate if all right pairs pass the verification. On average, with $N_1^{\texttt{WARP}} = 6$ right pairs, we find the correct guess for the key.



$\mathcal{X}_0$
Round 0
$\mathcal{Y}_0$
$\mathcal{X}_1$
Round 1
$\mathcal{Y}_1$
$\mathcal{X}_2$
Round 2
$\mathcal{Y}_2$
$\mathcal{X}_3$
The known value inferred from the differential characteristic.
The value of the nibble should be known to verify the condition.
The nibble of the key should be guessed.

Fig. 7. The first step of the related-key attack on WARP.

Step 2: recovering partial keys in the last three rounds. This step concentrates on the tail of the cipher. From Figure 6, the values of the right pair at $\mathcal{X}_{38}[25]$ should fall into the set $\mathbb{I}_S(\delta_{\mathtt{i}}^{\mathtt{WARP}}, \delta_{\mathtt{o}}^{\mathtt{WARP}})$. Based on this observation, the key-recovery procedure in the last three rounds is demonstrated in Figure 16. We enumerate the value of the 16-bit key $\mathcal{K}_0[1, 6, 12] \| \mathcal{K}_1[5]$ and check whether the condition $\mathcal{X}_{38}[25] \in \mathbb{I}_S(\delta_{\mathtt{i}}^{\mathtt{WARP}}, \delta_{\mathtt{o}}^{\mathtt{WARP}})$ is satisfied by all right pairs. The correct key guess can be identified with roughly $N_2^{\mathtt{WARP}} = 8$ right pairs.

Step 3: recovering partial keys in the first five rounds. After recovering partial keys in the last three rounds, we go back to the head of the cipher and implement key-recovery in the first five rounds. The enumeration is based on known values of the right pair at $\mathcal{X}_5[12]$, and the attack procedure is illustrated in Figure 17. We guess the value of the 20-bit key $\mathcal{K}_0[11, 14, 15] \| \mathcal{K}_1[12, 13]$ and check whether each right pair fulfils the condition $\mathcal{X}_5[12] \in \mathbb{I}_S(\delta_{\mathtt{i}}^{\mathtt{WARP}}, \delta_{\mathtt{o}}^{\mathtt{WARP}})$. With $N_3^{\mathtt{WARP}} = 10$ right pairs, we uncover the correct guess for the key.

Step 4: recovering partial keys in the last five rounds. Again, we move to the tail of the cipher, and the analysis is centred on the known values of the right pair at $\mathcal{X}_{36}[25]$. Figure 18 exhibits the key-recovery procedure. The value of the 24-bit key $\mathcal{K}_0[0, 3\text{-}5] \| \mathcal{K}_1[14, 15]$ is enumerated, and we inspect the validity of the condition $\mathcal{X}_{36}[25] \in \mathbb{I}_S(\delta_{\mathtt{i}}^{\mathtt{WARP}}, \delta_{\mathtt{o}}^{\mathtt{WARP}})$ for each right pair. The correct guess for the key can be identified with about $N_4^{\mathtt{WARP}} = 12$ right pairs.

Step 5: recovering partial keys in the first seven rounds. In this step, we go back to the head of the cipher and exploit the known values of the right pair at $\mathcal{X}_7[12]$. The key-recovery procedure is shown in Figure 19. We guess the value of the 24-bit key $\mathcal{K}_0[13] \| \mathcal{K}_1[2, 3, 8\text{-}10]$ and check the validity of the condition $\mathcal{X}_7[12] \in \mathbb{I}_S(\delta_{\mathtt{i}}^{\mathtt{WARP}}, \delta_{\mathtt{o}}^{\mathtt{WARP}})$ for each right pair. The correct key guess can be identified with about $N_5^{\mathtt{WARP}} = 12$ right pairs.

Step 6: recovering partial keys in the last seven rounds. This step focuses on the tail of the cipher. The analysis relies on known values of the right pair at $\mathcal{X}_{34}[25]$, and the key-recovery procedure is shown in Figure 20. We guess the value of the 16-bit key $\mathcal{K}_0[7,9,10]\|\mathcal{K}_1[1]$ and verify the condition $\mathcal{X}_{34}[25] \in \mathbb{I}_S(\delta_{\mathtt{i}}^{\mathtt{WARP}}, \delta_{\mathtt{o}}^{\mathtt{WARP}})$ for each right pair. The correct key guess can be obtained with about $N_6^{\mathtt{WARP}} = 8$ right pairs.

Step 7: exhaustively searching for the remaining keys. After performing the previous six steps, the value of the 16-bit key $\mathcal{K}_1[0,4,7,11]$ is unexplored. We exhaustively search for the value of $\mathcal{K}_1[0,4,7,11]$ with $N_7^{\mathtt{WARP}} = 1$ right pair.

Theoretical complexity analysis. To facilitate the analysis of each step, the number of right pairs for this attack should be no less than $N_R^{\mathtt{WARP}} = \max\{N_i^{\mathtt{WARP}} \mid 0 \leqslant i \leqslant 7\} = 12$. Hence, the data complexity of this attack is about $2 \cdot 12 \cdot 2^{40} \approx 2^{44.58}$ chosen plaintexts. The time complexity $T_1^{\mathtt{WARP}}$ to gather plaintext-ciphertext pairs is about $2^{44.58}$ full-round encryptions. Since $T_1^{\mathtt{WARP}}$ dominates the whole attack, the time complexity of the attack is $2^{44.58}$. The memory complexity of this attack is negligible.

Experimental runtime for the round-reduced attack. Since the complexity of the full-round attack is impractical, we try to launch an attack on a round-reduced version of `WARP`. We consider a 27-round version of `WARP` and note that the previous analyses also hold. The platform to run the test is the same as `CRAFT`. To begin with, we set $\mathcal{K}$ with the following randomly picked value

$$\mathcal{K} = \mathtt{0x3b09\ 0xfaab\ 0x8d77\ 0xf7f1\ 0x0fa4\ 0xd8f6\ 0x91f7\ 0x6f1e}.$$

As the probability of the 27-round differential characteristic is 2^{-26}, we generate 2^{31} pairs of chosen plaintexts randomly and get 32 right pairs. The runtime for this approach is about 2.02 hours. Due to the relatively weak diffusion of the Feistel structure, we find that using $N_R^{\mathtt{WARP}} = 12$ right pairs will raise the number of surviving key candidates in the test. Thus, we use 17 right pairs to complete the test. The output and the runtime for the seven steps are listed in Table 5. Although the number of remaining candidates exceeds the expected value in several steps, the unique correct key can be recovered after performing the seven steps. All source codes can be found in the Supplementary Material.

Table 5. Test results about `WARP`.

Step	1	2	3	4	5	6	7	Total
Runtime	0.004s	0.28s	0.22h	16.77h	27.51h	0.14s	0.34s	44.50h
#{Candidates}	16	1024	4096	4096	4	4	1	1

#{Candidates} is the number of tweakey candidates after conducting each step.

Performances of characteristics in other groups. The above attack is based on the differential characteristic in $\mathbb{G}_6^{\mathtt{WARP}}$. We also check the performance of the key-recovery attack implemented with the characteristic in other groups. The number of right pairs required in each step is given in Table 6. It can be noticed that the value of $N_R^{\mathtt{WARP}}$ regarding $\mathbb{G}_6^{\mathtt{WARP}}$ is the minimum one.

Table 6. The number of right pairs for the sixteen groups.

Group	$N_1^{\mathtt{WARP}}$	$N_2^{\mathtt{WARP}}$	$N_3^{\mathtt{WARP}}$	$N_4^{\mathtt{WARP}}$	$N_5^{\mathtt{WARP}}$	$N_6^{\mathtt{WARP}}$	$N_7^{\mathtt{WARP}}$	$N_R^{\mathtt{WARP}}$
$\mathbb{G}_0^{\mathtt{WARP}}$	8	8	12	6	16	12	1	16
$\mathbb{G}_1^{\mathtt{WARP}}$	8	8	12	6	12	14	1	14
$\mathbb{G}_2^{\mathtt{WARP}}$	6	8	10	8	14	14	1	14
$\mathbb{G}_3^{\mathtt{WARP}}$	8	6	12	10	10	14	1	14
$\mathbb{G}_4^{\mathtt{WARP}}$	8	8	14	8	12	12	1	14
$\mathbb{G}_5^{\mathtt{WARP}}$	8	6	14	10	16	8	1	16
$\mathbb{G}_6^{\mathtt{WARP}}$	6	8	10	12	12	8	1	12
$\mathbb{G}_7^{\mathtt{WARP}}$	8	8	12	10	8	12	1	12
$\mathbb{G}_8^{\mathtt{WARP}}$	8	8	12	6	16	12	1	16
$\mathbb{G}_9^{\mathtt{WARP}}$	8	8	12	6	12	14	1	14
$\mathbb{G}_{10}^{\mathtt{WARP}}$	6	8	10	8	14	14	1	14
$\mathbb{G}_{11}^{\mathtt{WARP}}$	8	6	12	10	10	14	1	14
$\mathbb{G}_{12}^{\mathtt{WARP}}$	8	8	14	8	12	12	1	14
$\mathbb{G}_{13}^{\mathtt{WARP}}$	8	6	14	10	16	8	1	16
$\mathbb{G}_{14}^{\mathtt{WARP}}$	6	8	10	12	12	8	1	12
$\mathbb{G}_{15}^{\mathtt{WARP}}$	8	8	12	10	8	12	1	12

5 Multiple Zero-Correlation Linear Attack on WARP

When analysing the security of WARP, we notice that the designers expected that the longest attack on the cipher in the single-key attack setting does not exceed 32 rounds. In this section, we propose a 33-round multiple zero-correlation linear attack, and it is the longest attack on WARP as far as we know.

5.1 Brief Review of Multiple Zero-Correlation Linear Attack

Zero-correlation linear cryptanalysis was introduced by Bogdanov and Rijmen [8] and is based on linear approximations with the correlation being zero. It can be viewed as the dual method of impossible differential cryptanalysis [16,4] in the domain of linear cryptanalysis.

Given a linear approximation with zero correlation, the conventional zero-correlation linear attack demands the full-codebook, which blocks up the development of this theory. To overcome this restriction, Bogdanov and Wang [9] proposed multiple zero-correlation linear cryptanalysis. Similarly to multiple linear cryptanalysis [6], multiple zero-correlation linear cryptanalysis relies on numerous linear approximations to accomplish more efficient attacks.

Suppose that the adversary collects N_Z plaintext-ciphertext pairs and discovers ℓ zero-correlation linear approximations for an n-bit block cipher. For each approximation, the adversary calculates the number of times $\mathtt{T}_j$ that the j-th linear approximation is fulfilled by plaintext-ciphertext pairs. Note that $\mathtt{T}_j$ suggests an empirical correlation $\hat{\mathtt{c}}_i = 2 \cdot \frac{\mathtt{T}_j}{N_Z} - 1$ for the j-th approximation. Then, the adversary computes the value of the statistic

$$\mathtt{T}_{\mathrm{MP}} = N_Z \cdot \sum_{i=0}^{\ell-1} \hat{\mathtt{c}}_i^2 = N_Z \cdot \sum_{i=0}^{\ell-1} \left(2 \cdot \frac{\mathtt{T}_j}{N_Z} - 1\right)^2,$$

and compares its value with the predefined value of the threshold Θ. A possible candidate should validate the condition $\mathtt{T}_{\mathrm{MP}} < \Theta$.

Denote the probability that the correct key survives as P_S, which is the success probability of the attack. Let the proportion of keys discarded in the screening process be 2^{-a}, where a is called the advantage [22]. With these notations, in the statistical hypothesis testing, the probabilities for the two types of errors are evaluated as $\alpha_0 = 2^{-a}$ and $\alpha_1 = 1 - P_S$. Note that α_0 is the probability that a wrong key candidate is accepted, and α_1 is the probability that the correct key is rejected.

Given that the number of linear approximations is not enough for the general model [7], we adopt the chi-square-multiple zero-correlation model in [24] to estimate the complexity of the attack. After fixing the threshold with $\Theta = \chi^{(\ell)}_{1-\alpha_0}$, the number of known-plaintexts[5] in the attack is

$$N_Z \approx \frac{2^n \cdot \left(\chi^{(\ell)}_{1-\alpha_0} - \chi^{(\ell)}_{\alpha_1}\right)}{\chi^{(\ell)}_{\alpha_1}}, \tag{1}$$

where $\chi^{(\ell)}_{1-\alpha_0}$ and $\chi^{(\ell)}_{\alpha_1}$ are quantiles of the χ^2-distribution with the degree of freedom being ℓ evaluated at points $1-\alpha_0$ and α_1, respectively.

5.2 21-Round Zero-Correlation Linear Approximations

The automatic method in [20,21,11] is utilised to search for zero-correlation linear approximations. We evaluate the search space such that both the input and output masks activate only one nibble. As a result, we find that the longest zero-correlation linear approximation covers 21 rounds. The thirty 21-round zero-correlation linear approximations returned by the solver can be partitioned into two groups. The first group is composed of 15 approximations

$$\begin{aligned} &\mathtt{0x0000\ 0x0000\ 0x0000\ 0x00}\mathit{\Gamma}_{\mathtt{i}}\mathtt{0\ 0x0000\ 0x0000\ 0x0000\ 0x0000} \\ \xrightarrow{\text{21-round}}\ &\mathtt{0x0000\ 0x0000\ 0x0000\ 0x0000\ 0x0000\ 0x0000\ 0x0000\ 0x000}\mathit{\Gamma}_{\mathtt{o}}, \end{aligned}$$

[5] Note that we only consider the known-plaintext sampling method. The distinct known-plaintexts sampling method [7] is not utilised, as adopting this method increases the memory complexity.

where $\Gamma_{\mathtt{i}}$ and $\Gamma_{\mathtt{o}}$ are nonzero nibbles meeting $\Gamma_{\mathtt{i}} = \Gamma_{\mathtt{o}}$. The approximations in the second group are

$$\begin{aligned} & \mathtt{0x0000\ 0x0000\ 0x0000\ 0x0000\ 0x0000\ 0x0000\ 0x0000\ 0x00}\Gamma_{\mathtt{i}}\mathtt{0} \\ \xrightarrow{\text{21-round}}\ & \mathtt{0x0000\ 0x0000\ 0x0000\ 0x000}\Gamma_{\mathtt{o}}\ \mathtt{0x0000\ 0x0000\ 0x0000\ 0x0000}. \end{aligned}$$

We manually verify the contradiction in the approximations, and the illustration can be found in the Supplementary Material.

5.3 33-Round Multiple Zero-Correlation Linear Attack on WARP

In this section, we propose a multiple zero-correlation linear attack on 33-round WARP. This attack exploits the first group of 21-round zero-correlation approximations in Section 5.2 from round 6 to 26. The key-recovery attack is illustrated in Figure 8. To control the time complexity of the attack, we equivalently move the XOR operations with the key in the r-th round to the $(r+1)$-th round for $0 \leqslant r \leqslant 5$. The tail of the distinguisher applies a similar strategy. The XOR operations with the key in the r'-th round are moved to the $(r'-1)$-th round for $27 \leqslant r' \leqslant 32$. N_Z plaintext-ciphertext pairs are required, and the detailed attack procedure is given in Appendix D.

Complexity analysis. We set the advantage as $a = 1.00$ and the success probability P_S as 50.00%. With Eq. (1), we obtain the data requirement of the attack is $N_Z = 2^{97.71}$. The time complexity of the attack is composed of the time complexity in the key enumeration phase as in **Step 0 - Step 17** and the time to check the remaining 36-bit key exhaustively. Thus, the total time complexity of the attack is $2^{127.01}$. Since the counter $\mathtt{C}_0[z_0]$ constitutes the most significant memory, the memory complexity is roughly 2^{100}. Given that the time complexity achieves $2^{127.01}$, we claim that the success probability of the attack is 50.00%, and it cannot be improved by repeating the entire work as the time complexity will go beyond 2^{128}.

6 Conclusion

This paper first focuses on the security of CRAFT and WARP in the related-key attack setting. For full-round CRAFT, we present a practical key-recovery attack with only one differential characteristic, and the theoretical time complexity is $2^{36.09}$ encryptions. The practical test is conducted and indicates that the theoretical analysis is valid. A full-round key-recovery attack on WARP is proposed, and the time complexity is $2^{44.58}$ encryptions. The theoretical attack is implemented on a round-reduced version of WARP, which guarantees validity. Moreover, we give a 33-round multiple zero-correlation linear attack on WARP, which is the longest attack on the cipher in the single-key attack setting as far as we know.

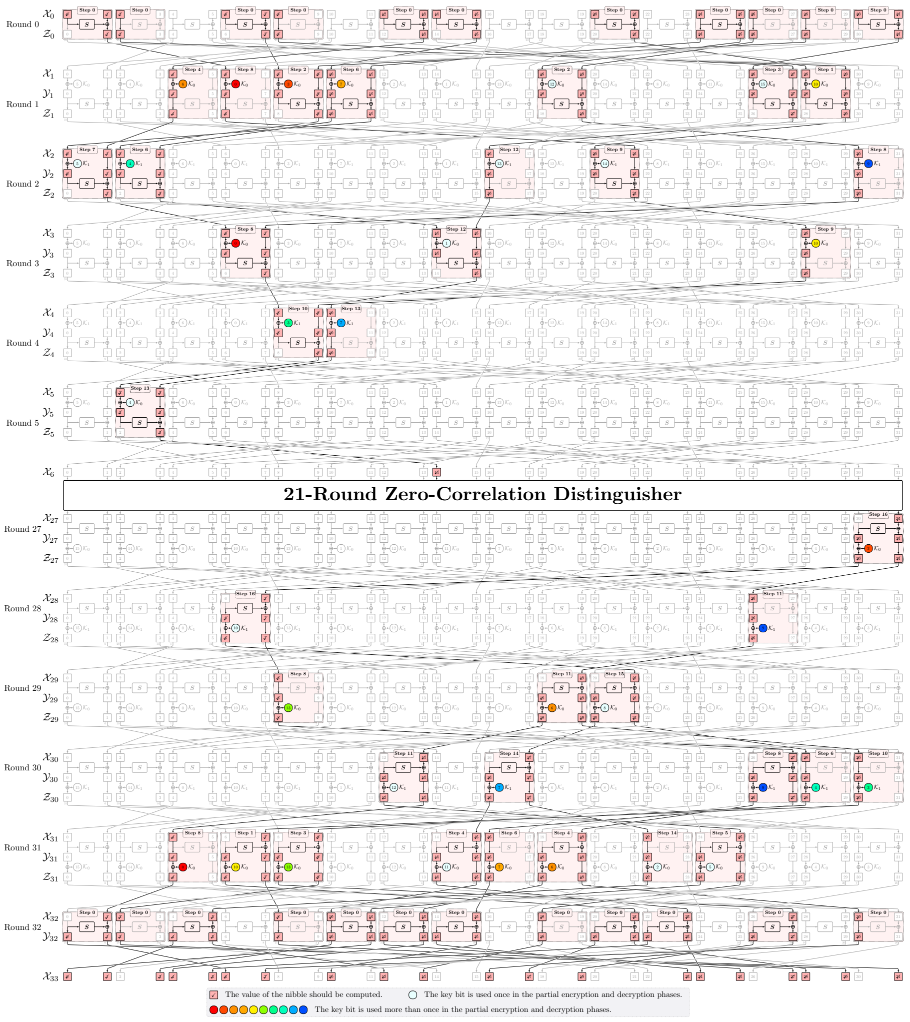

21-Round Zero-Correlation Distinguisher
The value of the nibble should be computed.
The key bit is used once in the partial encryption and decryption phases.
The key bit is used more than once in the partial encryption and decryption phases.

Fig. 8. Multiple zero-correlation linear attack on 33-round `WARP`.

Acknowledgements. The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper. The research leading to these results has received funding from the National Natural Science Foundation of China (Grant No. 62002201, Grant No. 62032014), the National Key Research and Development Program of China (Grant No. 2018YFA0704702), and the Major Basic Research Project of Natural Science Foundation of Shandong Province, China (Grant No. ZR202010220025).

References

1. Banik, S., Bao, Z., Isobe, T., Kubo, H., Liu, F., Minematsu, K., Sakamoto, K., Shibata, N., Shigeri, M.: WARP : Revisiting GFN for lightweight 128-bit block cipher. In: Dunkelman, O., Jr., M.J.J., O'Flynn, C. (eds.) Selected Areas in Cryptography - SAC 2020 - 27th International Conference, Halifax, NS, Canada (Virtual Event), October 21-23, 2020, Revised Selected Papers. Lecture Notes in Computer Science, vol. 12804, pp. 535–564. Springer (2020). https://doi.org/10.1007/978-3-030-81652-0_21
2. Banik, S., Bogdanov, A., Isobe, T., Shibutani, K., Hiwatari, H., Akishita, T., Regazzoni, F.: Midori: A block cipher for low energy. In: Iwata, T., Cheon, J.H. (eds.) Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II. Lecture Notes in Computer Science, vol. 9453, pp. 411–436. Springer (2015). https://doi.org/10.1007/978-3-662-48800-3_17
3. Beierle, C., Leander, G., Moradi, A., Rasoolzadeh, S.: CRAFT: lightweight tweakable block cipher with efficient protection against DFA attacks. IACR Trans. Symmetric Cryptol. **2019**(1), 5–45 (2019). https://doi.org/10.13154/tosc.v2019.i1.5-45
4. Biham, E., Biryukov, A., Shamir, A.: Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In: Stern, J. (ed.) Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding. Lecture Notes in Computer Science, vol. 1592, pp. 12–23. Springer (1999). https://doi.org/10.1007/3-540-48910-X_2
5. Biham, E., Shamir, A.: Differential cryptanalysis of DES-like cryptosystems. In: Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings. pp. 2–21 (1990). https://doi.org/10.1007/3-540-38424-3_1
6. Biryukov, A., Cannière, C.D., Quisquater, M.: On multiple linear approximations. In: Franklin, M.K. (ed.) Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings. Lecture Notes in Computer Science, vol. 3152, pp. 1–22. Springer (2004). https://doi.org/10.1007/978-3-540-28628-8_1
7. Blondeau, C., Nyberg, K.: Joint data and key distribution of simple, multiple, and multidimensional linear cryptanalysis test statistic and its impact to data complexity. Des. Codes Cryptogr. **82**(1-2), 319–349 (2017). https://doi.org/10.1007/s10623-016-0268-6
8. Bogdanov, A., Rijmen, V.: Zero-correlation linear cryptanalysis of block ciphers. IACR Cryptol. ePrint Arch. p. 123 (2011), `http://eprint.iacr.org/2011/123`
9. Bogdanov, A., Wang, M.: Zero correlation linear cryptanalysis with reduced data complexity. In: Canteaut, A. (ed.) Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers. Lecture Notes in Computer Science, vol. 7549, pp. 29–48. Springer (2012). https://doi.org/10.1007/978-3-642-34047-5_3
10. Canteaut, A., Lambooij, E., Neves, S., Rasoolzadeh, S., Sasaki, Y., Stevens, M.: Refined probability of differential characteristics including dependency between multiple rounds. IACR Trans. Symmetric Cryptol. **2017**(2), 203–227 (2017). https://doi.org/10.13154/tosc.v2017.i2.203-227

11. Cui, T., Chen, S., Fu, K., Wang, M., Jia, K.: New automatic tool for finding impossible differentials and zero-correlation linear approximations. Sci. China Inf. Sci. **64**(2) (2021). https://doi.org/10.1007/s11432-018-1506-4
12. Daemen, J., Rijmen, V.: Plateau characteristics. IET Inf. Secur. **1**(1), 11–17 (2007). https://doi.org/10.1049/iet-ifs:20060099
13. ElSheikh, M., Youssef, A.M.: Related-key differential cryptanalysis of full round CRAFT. In: Bhasin, S., Mendelson, A., Nandi, M. (eds.) Security, Privacy, and Applied Cryptography Engineering - 9th International Conference, SPACE 2019, Gandhinagar, India, December 3-7, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11947, pp. 50–66. Springer (2019). https://doi.org/10.1007/978-3-030-35869-3_6
14. Hadipour, H., Eichlseder, M.: Integral cryptanalysis of WARP based on monomial prediction. IACR Trans. Symmetric Cryptol. **2022**(2), 92–112 (2022). https://doi.org/10.46586/tosc.v2022.i2.92-112, https://doi.org/10.46586/tosc.v2022.i2.92-112
15. Kerckhoffs, A.: La cryptographie militaire. Journal des Sciences Militaires pp. 5–38
16. Knudsen, L.: DEAL - a 128-bit block cipher. In: NIST AES Proposal (1998)
17. Kumar, M., Yadav, T.: MILP based differential attack on round reduced WARP. In: Batina, L., Picek, S., Mondal, M. (eds.) Security, Privacy, and Applied Cryptography Engineering - 11th International Conference, SPACE 2021, Kolkata, India, December 10-13, 2021, Proceedings. Lecture Notes in Computer Science, vol. 13162, pp. 42–59. Springer (2021). https://doi.org/10.1007/978-3-030-95085-9_3
18. Lai, X., Massey, J.L., Murphy, S.: Markov ciphers and differential cryptanalysis. In: Davies, D.W. (ed.) Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings. Lecture Notes in Computer Science, vol. 547, pp. 17–38. Springer (1991). https://doi.org/10.1007/3-540-46416-6_2
19. Lallemand, V., Minier, M., Rouquette, L.: Automatic search of rectangle attacks on feistel ciphers: Application to WARP. IACR Trans. Symmetric Cryptol. **2022**(2), 113–140 (2022). https://doi.org/10.46586/tosc.v2022.i2.113-140, https://doi.org/10.46586/tosc.v2022.i2.113-140
20. Sasaki, Y., Todo, Y.: New impossible differential search tool from design and cryptanalysis aspects. IACR Cryptol. ePrint Arch. p. 1181 (2016), http://eprint.iacr.org/2016/1181
21. 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. pp. 185–215 (2017). https://doi.org/10.1007/978-3-319-56617-7_7
22. Selçuk, A.A.: On probability of success in linear and differential cryptanalysis. J. Cryptol. **21**(1), 131–147 (2008). https://doi.org/10.1007/s00145-007-9013-7
23. Soleimany, H., Nyberg, K.: Zero-correlation linear cryptanalysis of reduced-round LBlock. Des. Codes Cryptogr. **73**(2), 683–698 (2014). https://doi.org/10.1007/s10623-014-9976-y
24. Sun, L., Chen, H., Wang, M.: Zero-correlation attacks: statistical models independent of the number of approximations. Des. Codes Cryptogr. **86**(9), 1923–1945 (2018). https://doi.org/10.1007/s10623-017-0430-9
25. Sun, L., Wang, W., Wang, M.: More accurate differential properties of LED64 and Midori64. IACR Trans. Symmetric Cryptol. **2018**(3), 93–123 (2018). https://doi.org/10.13154/tosc.v2018.i3.93-123

26. Teh, J.S., Biryukov, A.: Differential cryptanalysis of WARP. IACR Cryptol. ePrint Arch. p. 1641 (2021), `https://eprint.iacr.org/2021/1641`
27. Zheng, Y., Matsumoto, T., Imai, H.: On the construction of block ciphers provably secure and not relying on any unproved hypotheses. In: Brassard, G. (ed.) Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings. Lecture Notes in Computer Science, vol. 435, pp. 461–480. Springer (1989). https://doi.org/10.1007/0-387-34805-0_42

A Right Pairs for the S-box of CRAFT and WARP

The right pair for the S-box of CRAFT and WARP can be found in Table 7.

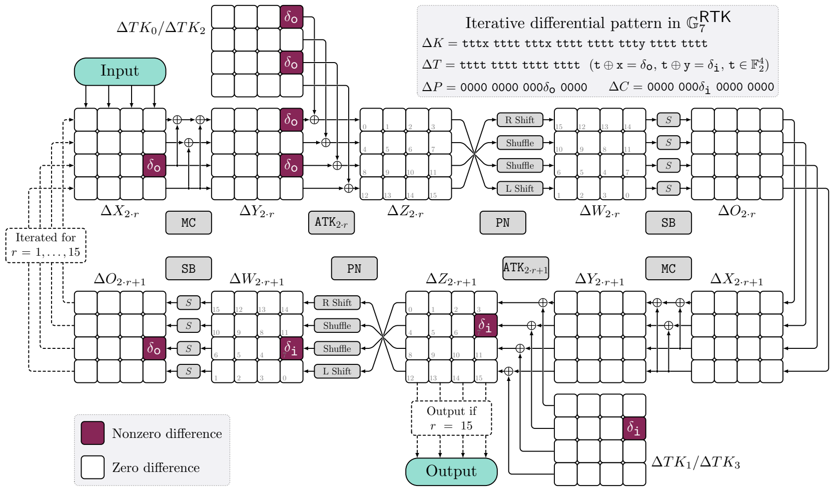
Table 7. Right pairs for the S-box of CRAFT and WARP.

$\delta_\mathtt{i}$	$\delta_\mathtt{o}$	$\mathbb{I}_S(\delta_\mathtt{i},\delta_\mathtt{o})$	$\mathbb{O}_S(\delta_\mathtt{i},\delta_\mathtt{o})$	$\delta_\mathtt{i}$	$\delta_\mathtt{o}$	$\mathbb{I}_S(\delta_\mathtt{i},\delta_\mathtt{o})$	$\mathbb{O}_S(\delta_\mathtt{i},\delta_\mathtt{o})$
0x1	0x1	{0x8, 0x9}	{0x8, 0x9}	0x8	0x9	{0x5, 0xd}	{0x2, 0xb}
0x1	0x2	{0xc, 0xd, 0xe, 0xf}	{0x0, 0x2, 0x4, 0x6}	0x8	0xb	{0x6, 0xe}	{0x4, 0xf}
0x1	0x4	{0xa, 0xb}	{0x1, 0x5}	0x8	0xc	{0x2, 0xa}	{0x1, 0xd}
0x1	0x5	{0x4, 0x5}	{0xb, 0xe}	0x8	0xe	{0x4, 0xc}	{0x0, 0xe}
0x1	0x6	{0x0, 0x1}	{0xa, 0xc}	0x9	0x2	{0x1, 0x3, 0x8, 0xa}	{0x1, 0x3, 0x8, 0xa}
0x1	0x8	{0x6, 0x7}	{0x7, 0xf}	0x9	0x3	{0x7, 0xe}	{0x4, 0x7}
0x1	0xe	{0x2, 0x3}	{0x3, 0xd}	0x9	0x5	{0x0, 0x9}	{0x9, 0xc}
0x2	0x1	{0x0, 0x2, 0x4, 0x6}	{0xc, 0xd, 0xe, 0xf}	0x9	0x8	{0x2, 0xb}	{0x5, 0xd}
0x2	0x4	{0xc, 0xd, 0xe, 0xf}	{0x0, 0x2, 0x4, 0x6}	0x9	0x9	{0x6, 0xf}	{0x6, 0xf}
0x2	0x9	{0x1, 0x3, 0x8, 0xa}	{0x1, 0x3, 0x8, 0xa}	0x9	0xb	{0x5, 0xc}	{0x0, 0xb}
0x2	0xc	{0x5, 0x7, 0x9, 0xb}	{0x5, 0x7, 0x9, 0xb}	0x9	0xc	{0x4, 0xd}	{0x2, 0xe}
0x3	0x4	{0x5, 0x6}	{0xb, 0xf}	0xa	0x5	{0x2, 0x7, 0x8, 0xd}	{0x2, 0x7, 0x8, 0xd}
0x3	0x6	{0xc, 0xd, 0xe, 0xf}	{0x0, 0x2, 0x4, 0x6}	0xa	0xa	{0x3, 0x4, 0x9, 0xe}	{0x3, 0x4, 0x9, 0xe}
0x3	0x7	{0x1, 0x2}	{0xa, 0xd}	0xa	0xd	{0x0, 0x5, 0xa, 0xf}	{0x1, 0x6, 0xb, 0xc}
0x3	0x8	{0x9, 0xa}	{0x1, 0x9}	0xa	0xf	{0x1, 0x6, 0xb, 0xc}	{0x0, 0x5, 0xa, 0xf}
0x3	0x9	{0x4, 0x7}	{0x7, 0xe}	0xb	0x4	{0x2, 0x9}	{0x9, 0xd}
0x3	0xd	{0x8, 0xb}	{0x5, 0x8}	0xb	0x7	{0x7, 0xc}	{0x0, 0x7}
0x3	0xf	{0x0, 0x3}	{0x3, 0xc}	0xb	0x8	{0x4, 0xf}	{0x6, 0xe}
0x4	0x1	{0x1, 0x5}	{0xa, 0xb}	0xb	0x9	{0x0, 0xb}	{0x5, 0xc}
0x4	0x2	{0x0, 0x2, 0x4, 0x6}	{0xc, 0xd, 0xe, 0xf}	0xb	0xb	{0x1, 0x3, 0x8, 0xa}	{0x1, 0x3, 0x8, 0xa}
0x4	0x3	{0xb, 0xf}	{0x5, 0x6}	0xb	0xd	{0x6, 0xd}	{0x2, 0xf}
0x4	0x4	{0x3, 0x7}	{0x3, 0x7}	0xb	0xf	{0x5, 0xe}	{0x4, 0xb}
0x4	0x5	{0xa, 0xe}	{0x1, 0x4}	0xc	0x2	{0x5, 0x7, 0x9, 0xb}	{0x5, 0x7, 0x9, 0xb}
0x4	0x8	{0x8, 0xc}	{0x0, 0x8}	0xc	0x5	{0x3, 0xf}	{0x3, 0x6}
0x4	0xb	{0x9, 0xd}	{0x2, 0x9}	0xc	0x6	{0x4, 0x8}	{0x8, 0xe}
0x5	0x1	{0xb, 0xe}	{0x4, 0x5}	0xc	0x8	{0x1, 0xd}	{0x2, 0xa}
0x5	0x4	{0x1, 0x4}	{0xa, 0xe}	0xc	0x9	{0x2, 0xe}	{0x4, 0xd}
0x5	0x7	{0x0, 0x5, 0xa, 0xf}	{0x1, 0x6, 0xb, 0xc}	0xc	0xc	{0x0, 0xc}	{0x0, 0xc}
0x5	0x9	{0x9, 0xc}	{0x0, 0x9}	0xc	0xe	{0x6, 0xa}	{0x1, 0xf}
0x5	0xa	{0x2, 0x7, 0x8, 0xd}	{0x2, 0x7, 0x8, 0xd}	0xd	0x3	{0x5, 0x8}	{0x8, 0xb}
0x5	0xc	{0x3, 0x6}	{0x3, 0xf}	0xd	0x6	{0x7, 0xa}	{0x1, 0x7}
0x6	0x1	{0xa, 0xc}	{0x0, 0x1}	0xd	0x7	{0x3, 0x4, 0x9, 0xe}	{0x3, 0x4, 0x9, 0xe}
0x6	0x3	{0x0, 0x2, 0x4, 0x6}	{0xc, 0xd, 0xe, 0xf}	0xd	0xa	{0x1, 0x6, 0xb, 0xc}	{0x0, 0x5, 0xa, 0xf}
0x6	0x7	{0xb, 0xd}	{0x2, 0x5}	0xd	0xb	{0x2, 0xf}	{0x6, 0xd}
0x6	0x8	{0x3, 0x5}	{0x3, 0xb}	0xd	0xe	{0x0, 0xd}	{0x2, 0xc}
0x6	0xc	{0x8, 0xe}	{0x4, 0x8}	0xe	0x1	{0x3, 0xd}	{0x2, 0x3}
0x6	0xd	{0x1, 0x7}	{0x7, 0xa}	0xe	0x7	{0x6, 0x8}	{0x8, 0xf}
0x6	0xf	{0x9, 0xf}	{0x6, 0x9}	0xe	0x8	{0x0, 0xe}	{0x4, 0xc}
0x7	0x3	{0xa, 0xd}	{0x1, 0x2}	0xe	0xc	{0x1, 0xf}	{0x6, 0xa}
0x7	0x5	{0x1, 0x6, 0xb, 0xc}	{0x0, 0x5, 0xa, 0xf}	0xe	0xd	{0x2, 0xc}	{0x0, 0xd}
0x7	0x6	{0x2, 0x5}	{0xb, 0xd}	0xe	0xe	{0x5, 0x7, 0x9, 0xb}	{0x5, 0x7, 0x9, 0xb}
0x7	0xb	{0x0, 0x7}	{0x7, 0xc}	0xe	0xf	{0x4, 0xa}	{0x1, 0xe}
0x7	0xd	{0x3, 0x4, 0x9, 0xe}	{0x3, 0x4, 0x9, 0xe}	0xf	0x3	{0x3, 0xc}	{0x0, 0x3}
0x7	0xe	{0x8, 0xf}	{0x6, 0x8}	0xf	0x6	{0x6, 0x9}	{0x9, 0xf}
0x8	0x1	{0x7, 0xf}	{0x6, 0x7}	0xf	0xa	{0x0, 0x5, 0xa, 0xf}	{0x1, 0x6, 0xb, 0xc}
0x8	0x3	{0x1, 0x9}	{0x9, 0xa}	0xf	0xb	{0x4, 0xb}	{0x5, 0xe}
0x8	0x4	{0x0, 0x8}	{0x8, 0xc}	0xf	0xe	{0x1, 0xe}	{0x4, 0xa}
0x8	0x6	{0x3, 0xb}	{0x3, 0x5}	0xf	0xf	{0x2, 0x7, 0x8, 0xd}	{0x2, 0x7, 0x8, 0xd}

B Materials for Related-Key Attack on CRAFT

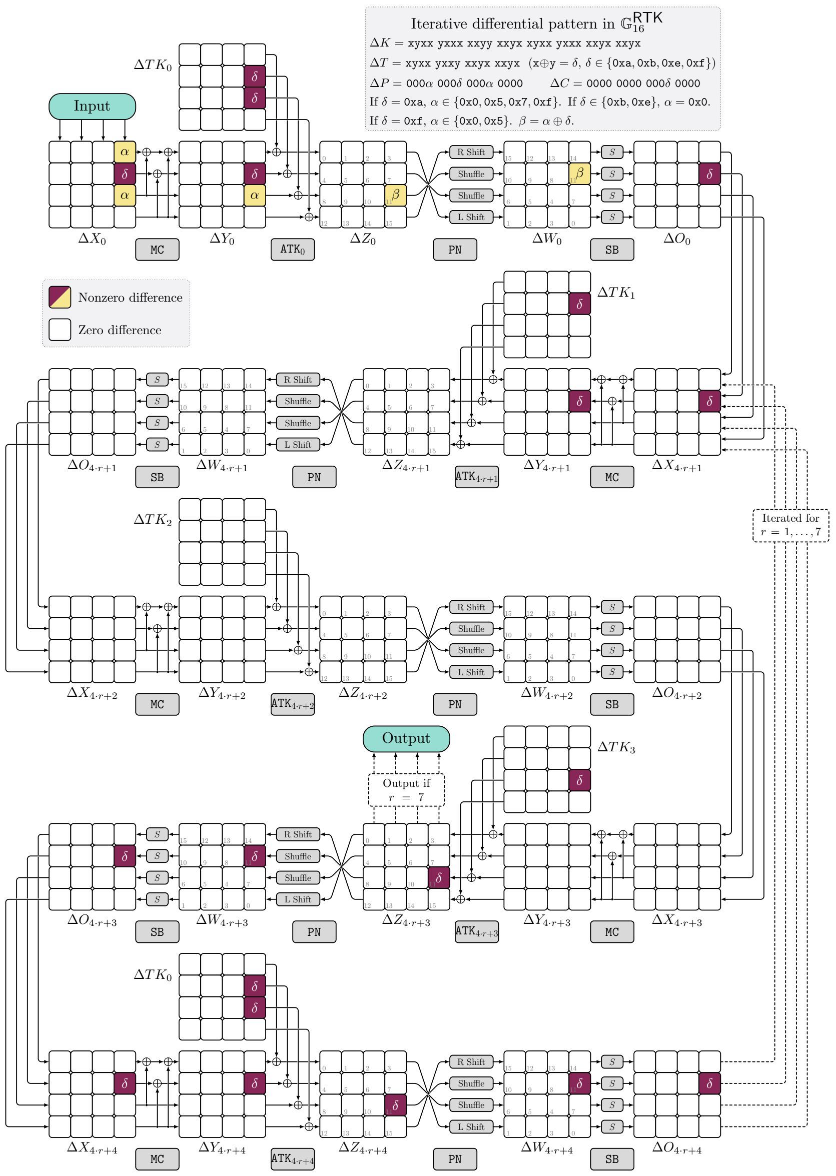
B.1 Illustrations for Differential Patterns in the RTK Setting

Figure 9 and Figure 10 show iterative differential patterns in $\mathbb{G}_7^{\mathsf{RTK}}$ and $\mathbb{G}_{16}^{\mathsf{RTK}}$.



Iterative differential pattern in $\mathbb{G}_7^{\mathsf{RTK}}$
$\Delta K = \texttt{tttx tttt tttx tttt tttt ttty tttt tttt}$
$\Delta T = \texttt{tttt tttt tttt tttt}$ $(\texttt{t} \oplus \texttt{x} = \delta_{\texttt{o}}, \texttt{t} \oplus \texttt{y} = \delta_{\texttt{i}}, \texttt{t} \in \mathbb{F}_2^4)$
$\Delta P = 0000\ 0000\ 000\delta_{\texttt{o}}\ 0000$
$\Delta C = 0000\ 000\delta_{\texttt{i}}\ 0000\ 0000$
$\Delta TK_0/\Delta TK_2$
Input
$\Delta X_{2\cdot r}$
MC
$\Delta Y_{2\cdot r}$
$\mathtt{ATK}_{2\cdot r}$
$\Delta Z_{2\cdot r}$
PN
$\Delta W_{2\cdot r}$
SB
$\Delta O_{2\cdot r}$
R Shift
Shuffle
Shuffle
L Shift
S
Iterated for $r = 1, \ldots, 15$
$\Delta O_{2\cdot r+1}$
$\Delta W_{2\cdot r+1}$
$\Delta Z_{2\cdot r+1}$
$\mathtt{ATK}_{2\cdot r+1}$
$\Delta Y_{2\cdot r+1}$
$\Delta X_{2\cdot r+1}$
$\delta_{\texttt{o}}$
$\delta_{\texttt{i}}$
Output if $r = 15$
Output
$\Delta TK_1/\Delta TK_3$
Nonzero difference
Zero difference

Fig. 9. Iterative differential pattern exploited in $\mathbb{G}_7^{\mathsf{RTK}}$. We omit the $\mathtt{ARC}_r$ operation. Note that $\delta_{\texttt{i}} \rightarrow \delta_{\texttt{o}}$ should be a possible differential propagation for the S-box with probability 2^{-2}.


Iterative differential pattern in $\mathbb{G}_{16}^{\mathsf{RTK}}$

$\Delta K =$ xyxx yxxx xxyy xxyx xyxx yxxx xxyx xxyx

$\Delta T =$ xyxx yxxy xxyx xxyx $(\mathtt{x}\oplus\mathtt{y} = \delta,\ \delta \in \{\mathtt{0xa}, \mathtt{0xb}, \mathtt{0xe}, \mathtt{0xf}\})$

$\Delta P =$ 000α 000δ 000α 0000 $\Delta C =$ 0000 0000 000δ 0000

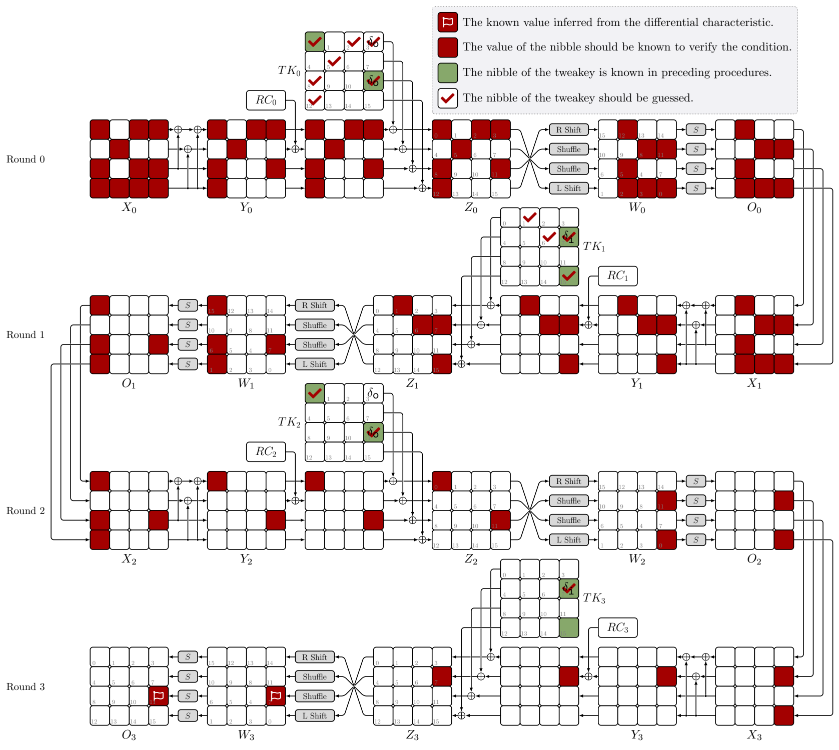
If $\delta = \mathtt{0xa}$, $\alpha \in \{\mathtt{0x0}, \mathtt{0x5}, \mathtt{0x7}, \mathtt{0xf}\}$. If $\delta \in \{\mathtt{0xb}, \mathtt{0xe}\}$, $\alpha = \mathtt{0x0}$.

If $\delta = \mathtt{0xf}$, $\alpha \in \{\mathtt{0x0}, \mathtt{0x5}\}$. $\beta = \alpha \oplus \delta$.

Fig. 10. Iterative differential pattern exploited in $\mathbb{G}_{16}^{\mathsf{RTK}}$. We omit the $\mathtt{ARC}_r$ operation. Note that $\beta \to \delta$ and $\delta \to \delta$ should be possible differential propagations for the S-box with probability 2^{-2}.

B.2 Illustrations for Key-Recovery Procedures

Figure 11 - 14 demonstrate the key-recovery procedures of the related-key differential attack on `WARP`.



The known value inferred from the differential characteristic.
The value of the nibble should be known to verify the condition.
The nibble of the tweakey is known in preceding procedures.
The nibble of the tweakey should be guessed.
Round 0
Round 1
Round 2
Round 3
TK_0
TK_1
TK_2
TK_3
RC_0
RC_1
RC_2
RC_3
X_0
Y_0
Z_0
W_0
O_0
X_1
Y_1
Z_1
W_1
O_1
X_2
Y_2
Z_2
W_2
O_2
X_3
Y_3
Z_3
W_3
O_3
R Shift
Shuffle
L Shift
S
δ_0

Fig. 11. The third step of the attack on `CRAFT`.

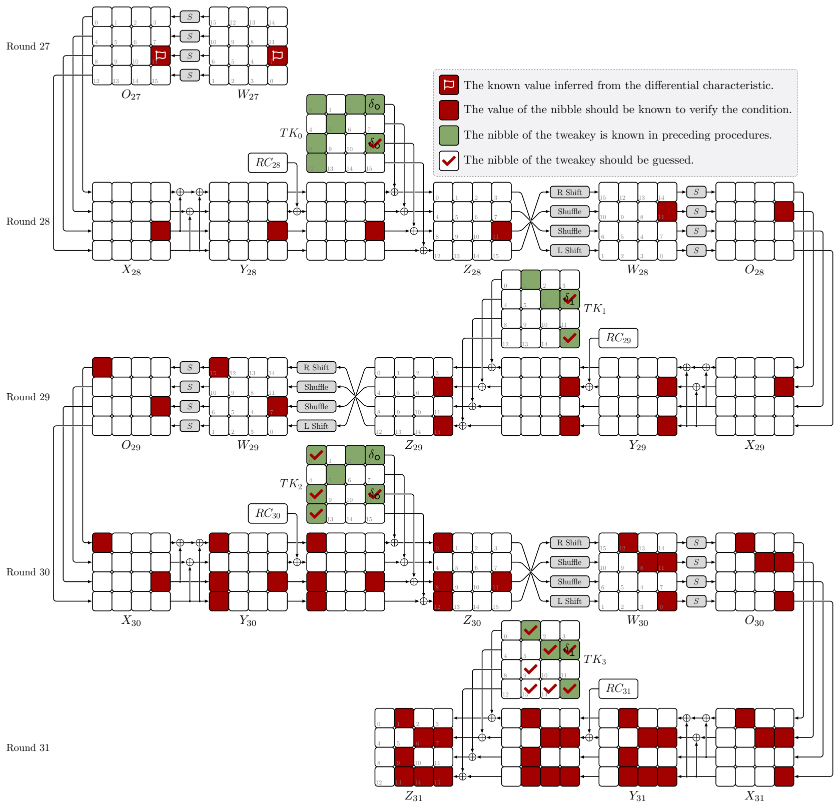


Fig. 12. The fourth step of the attack on `CRAFT`.

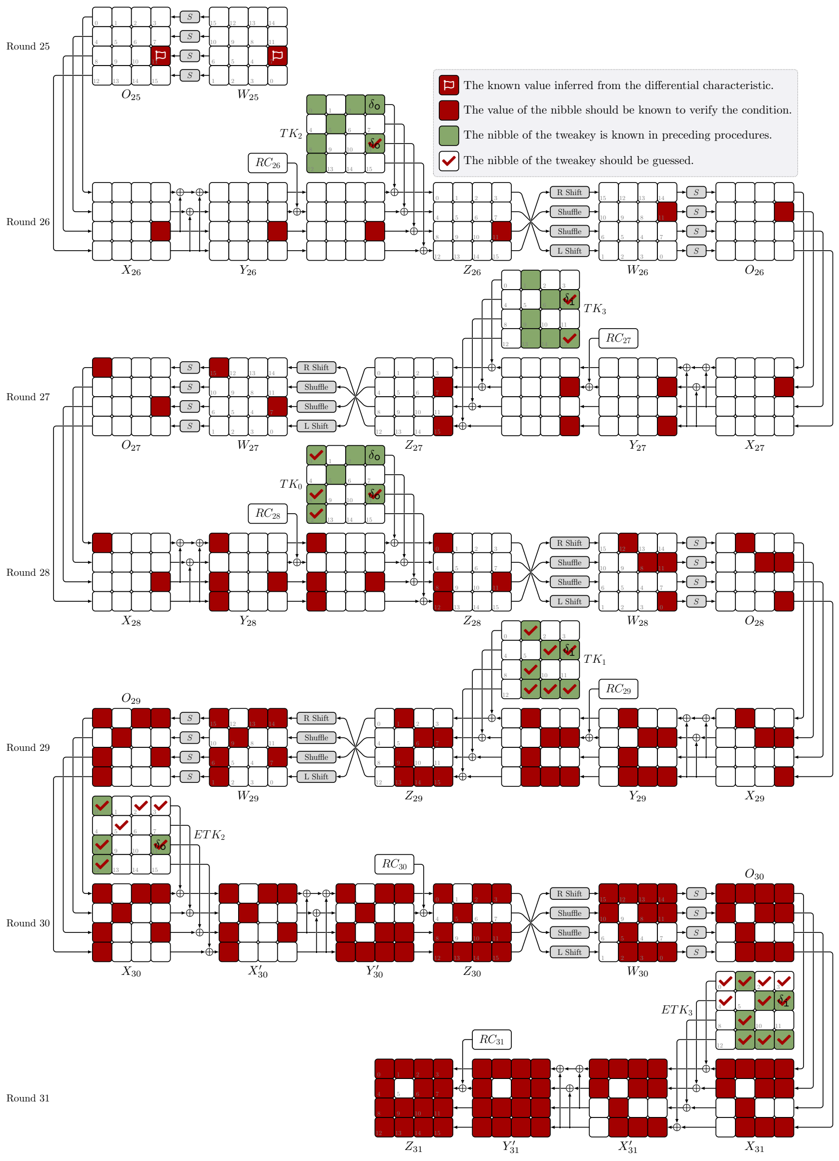


Fig. 13. The fifth step of the attack on `CRAFT`.

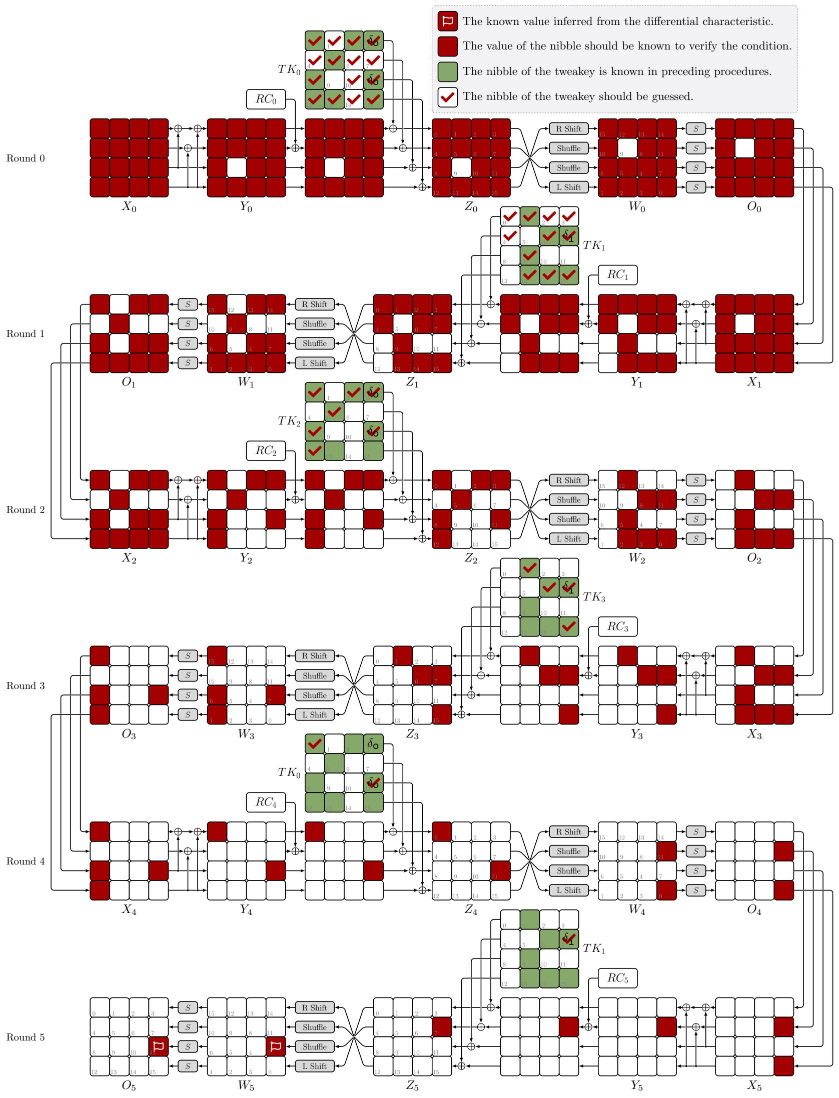


Fig. 14. The sixth step of the attack on CRAFT.

C Materials for Related-Key Attack on `WARP`

C.1 Round Function of `WARP`

Figure 15 shows the round function of `WARP`.

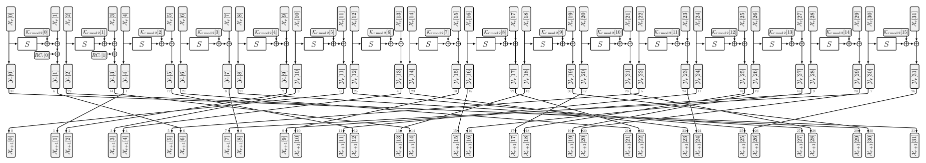
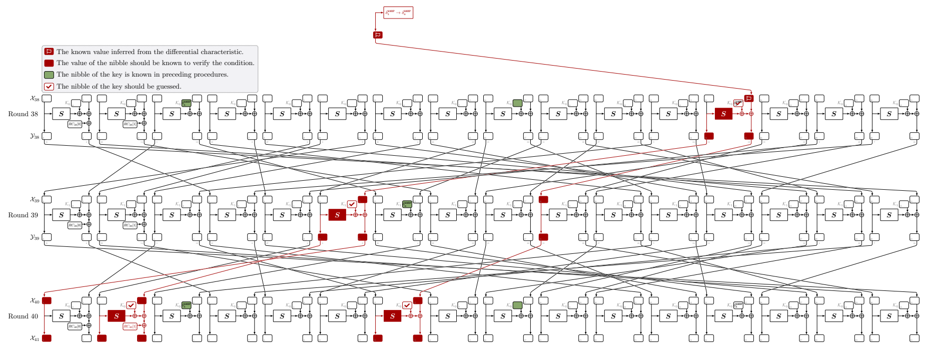


Fig. 15. Round function of `WARP`.

C.2 Key-Recovery Procedures

Figure 16 - 20 demonstrate the key-recovery procedures of the related-key differential attack on `WARP`.



$\delta_1^{\mathtt{WARP}} \to \delta_0^{\mathtt{WARP}}$
The known value inferred from the differential characteristic.
The value of the nibble should be known to verify the condition.
The nibble of the key is known in preceding procedures.
The nibble of the key should be guessed.
$\mathcal{X}_{38}$
Round 38
$\mathcal{Y}_{38}$
$RC_{38}[0]$
$RC_{38}[1]$
$\mathcal{X}_{39}$
Round 39
$\mathcal{Y}_{39}$
$\mathcal{X}_{40}$
Round 40
$\mathcal{X}_{41}$

Fig. 16. The second step of the related-key attack on `WARP`.

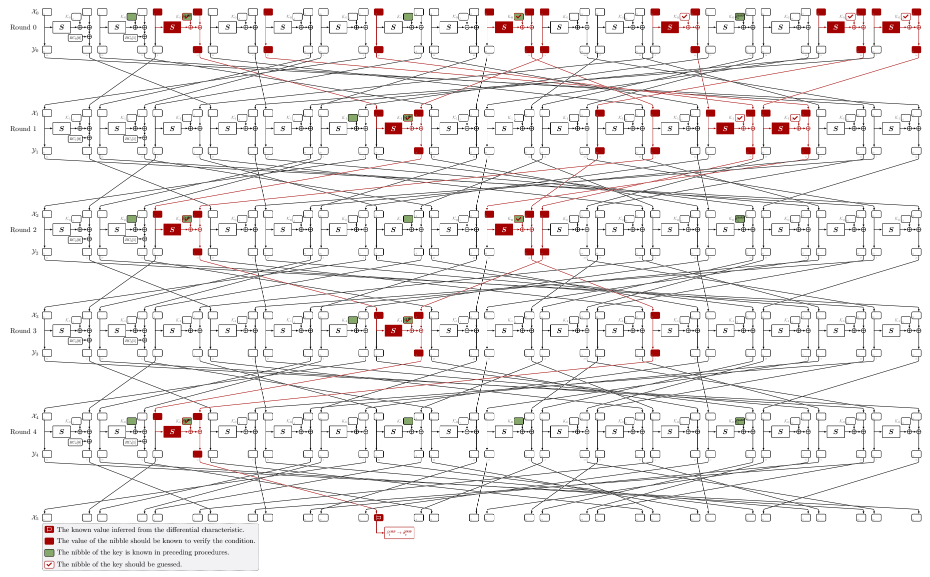

Round 0
Round 1
Round 2
Round 3
Round 4
x_5
🟥 The known value inferred from the differential characteristic.
🟥 The value of the nibble should be known to verify the condition.
🟩 The nibble of the key is known in preceding procedures.
✓ The nibble of the key should be guessed.

Fig. 17. The third step of the related-key attack on `WARP`.

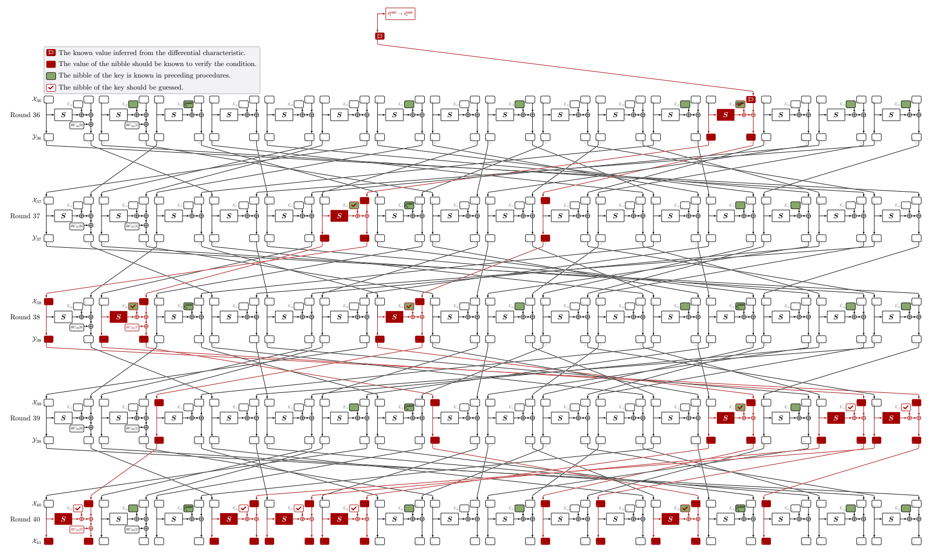

🟥 The known value inferred from the differential characteristic.
🟥 The value of the nibble should be known to verify the condition.
🟩 The nibble of the key is known in preceding procedures.
✓ The nibble of the key should be guessed.
Round 36
Round 37
Round 38
Round 39
Round 40
x_{41}

Fig. 18. The fourth step of the related-key attack on `WARP`.

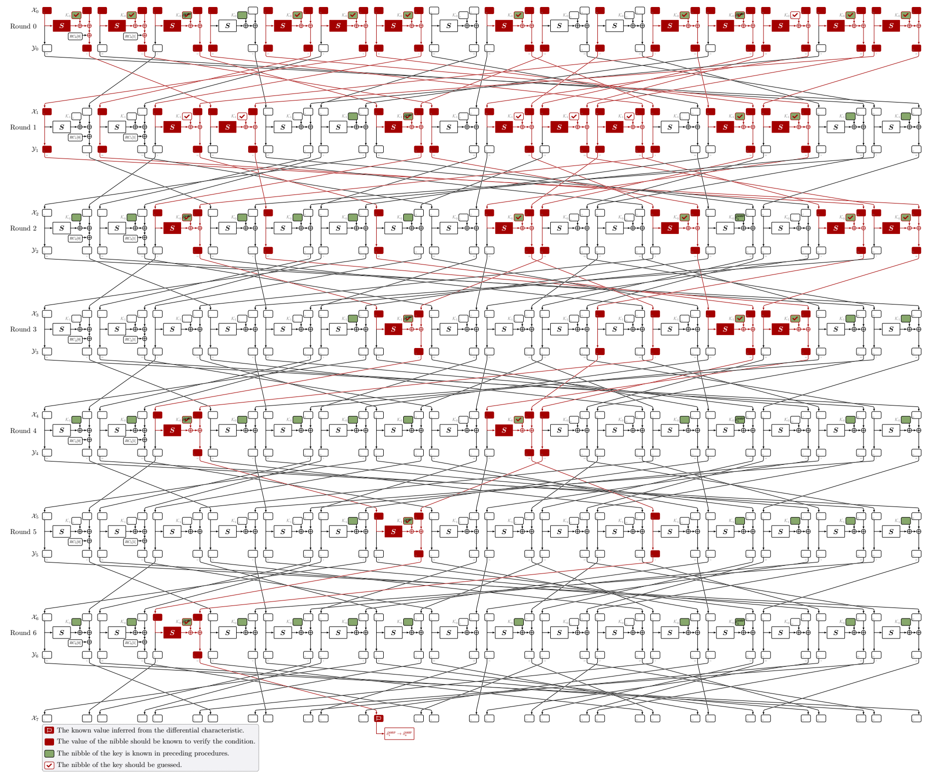


Fig. 19. The fifth step of the related-key attack on `WARP`.

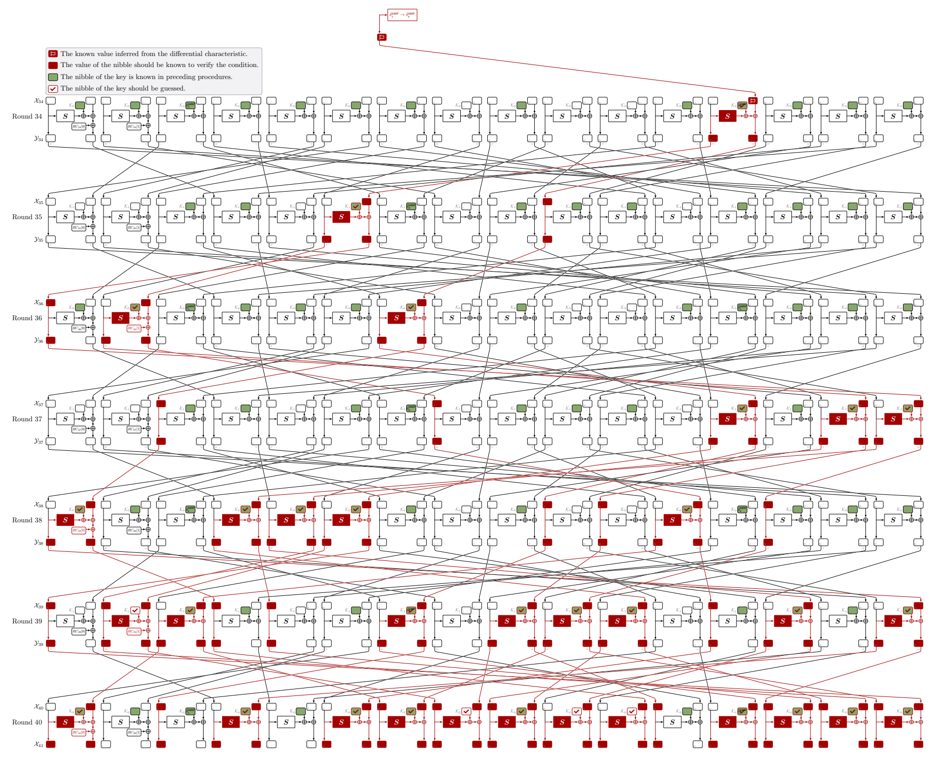

The known value inferred from the differential characteristic.
The value of the nibble should be known to verify the condition.
The nibble of the key is known in preceding procedures.
The nibble of the key should be guessed.
Round 34
Round 35
Round 36
Round 37
Round 38
Round 39
Round 40
S

Fig. 20. The sixth step of the related-key attack on `WARP`.

D Materials for Zero-Correlation Linear Attack on WARP

The 33-round attack proceeds as follows.

Step 0 Allocate a counter $\mathtt{C}_0[z_0]$ for each of 2^{100} possible values of

$$z_0 = \mathcal{X}_1[4, 6, 8\text{-}11, 18, 19, 26\text{-}29] \| \mathcal{Z}_{31}[4, 6\text{-}9, 14\text{-}16, 18, 19, 22, 24, 25].$$

For each of N_Z plaintext-ciphertext pairs, compute the value of z_0 and update $\mathtt{C}_0[z_0]$ with $\mathtt{C}_0[z_0] + 1$. The time complexity of this step is composed of N_Z 33-round encryptions, N_Z memory accesses to a table with 2^{100} elements, $15 \cdot N_Z$ S-box operations, and $\mathcal{O}(15 \cdot N_Z)$ XOR operations. As in [23], we view one memory access to a large table as one 33-round encryption. Therefore, the dominant time complexity is $2 \cdot N_Z$ 33-round encryptions.

Step 1 Allocate a counter $\mathtt{C}_1[z_1]$ for each of 2^{92} possible values of

$$z_1 = \mathcal{X}_1[4, 6, 8\text{-}11, 18, 19, 26, 27] \| \mathcal{Z}_1[29] \| \mathcal{X}_{31}[7] \| \mathcal{Z}_{31}[4, 8, 9, 14\text{-}16, 18, 19, 22, 24, 25].$$

For each possible 4-bit key $\mathcal{K}_0[10]$, compute the value of z_1 and update $\mathtt{C}_1[z_1]$ with $\mathtt{C}_1[z_1] + \mathtt{C}_0[z_0]$. Similarly, the dominant time complexity of this step is $2^{100} \cdot 2^4 = 2^{104}$ memory accesses to a table with 2^{92} elements.

Step 2 Allocate a counter $\mathtt{C}_2[z_2]$ for each of 2^{84} possible values of

$$z_2 = \mathcal{X}_1[4, 6, 10, 11, 26, 27] \| \mathcal{Z}_1[9, 19, 29] \| \mathcal{X}_{31}[7] \| \mathcal{Z}_{31}[4, 8, 9, 14\text{-}16, 18, 19, 22, 24, 25].$$

For each possible 8-bit key $\mathcal{K}_0[3, 12]$, compute the value of z_2 and update $\mathtt{C}_2[z_2]$ with $\mathtt{C}_2[z_2] + \mathtt{C}_1[z_1]$. The dominant time complexity of this step is $2^{92} \cdot 2^8 \cdot 2^4 = 2^{104}$ memory accesses to a table with 2^{84} elements.

Step 3 Allocate a counter $\mathtt{C}_3[z_3]$ for each of 2^{76} possible values of

$$z_3 = \mathcal{X}_1[4, 6, 10, 11] \| \mathcal{Z}_1[9, 19, 27, 29] \| \mathcal{X}_{31}[7, 9] \| \mathcal{Z}_{31}[4, 14\text{-}16, 18, 19, 22, 24, 25].$$

For each possible 8-bit key $\mathcal{K}_0[13, 15]$, compute the value of z_3 and update $\mathtt{C}_3[z_3]$ with $\mathtt{C}_3[z_3] + \mathtt{C}_2[z_2]$. The dominant time complexity of this step is $2^{84} \cdot 2^8 \cdot 2^{12} = 2^{104}$ memory accesses to a table with 2^{76} elements.

Step 4 Allocate a counter $\mathtt{C}_4[z_4]$ for each of 2^{68} possible values of

$$z_4 = \mathcal{X}_1[6, 10, 11] \| \mathcal{Z}_1[4, 9, 19, 27, 29] \| \mathcal{X}_{31}[7, 9, 15, 19] \| \mathcal{Z}_{31}[4, 16, 22, 24, 25].$$

For each possible 8-bit key $\mathcal{K}_0[6, 11]$, compute the value of z_4 and update $\mathtt{C}_4[z_4]$ with $\mathtt{C}_4[z_4] + \mathtt{C}_3[z_3]$. The dominant time complexity of this step is $2^{76} \cdot 2^8 \cdot 2^{20} = 2^{104}$ memory accesses to a table with 2^{68} elements.

Step 5 Allocate a counter $\mathtt{C}_5[z_5]$ for each of 2^{64} possible values of

$$z_5 = \mathcal{X}_1[6, 10, 11] \| \mathcal{Z}_1[4, 9, 19, 27, 29] \| \mathcal{X}_{31}[7, 9, 15, 19, 25] \| \mathcal{Z}_{31}[4, 16, 22].$$

For each possible 4-bit key $\mathcal{K}_0[5]$, compute the value of z_5 and update $\mathtt{C}_5[z_5]$ with $\mathtt{C}_5[z_5] + \mathtt{C}_4[z_4]$. The dominant time complexity of this step is $2^{68} \cdot 2^4 \cdot 2^{28} = 2^{100}$ memory accesses to a table with 2^{64} elements.

Step 6 Allocate a counter $\mathtt{C}_6[z_6]$ for each of 2^{60} possible values of

$$z_6 = \mathcal{X}_1[6]\|\mathcal{Z}_1[4,11,19,27,29]\|\mathcal{Z}_2[3]\|\mathcal{X}_{30}[28]\|\mathcal{X}_{31}[7,15,16,19,25]\|\mathcal{Z}_{31}[4,22].$$

For each possible 8-bit key $\mathcal{K}_0[7]\|\mathcal{K}_1[4]$, compute the value of z_6 and update $\mathtt{C}_6[z_6]$ with $\mathtt{C}_6[z_6] + \mathtt{C}_5[z_5]$. The dominant time complexity of this step is $2^{64} \cdot 2^8 \cdot 2^{32} = 2^{104}$ memory accesses to a table with 2^{60} elements.

Step 7 Allocate a counter $\mathtt{C}_7[z_7]$ for each of 2^{56} possible values of

$$z_7 = \mathcal{X}_1[6]\|\mathcal{Z}_1[19,27,29]\|\mathcal{Z}_2[1,3]\|\mathcal{X}_{30}[28]\|\mathcal{X}_{31}[7,15,16,19,25]\|\mathcal{Z}_{31}[4,22].$$

For each possible 4-bit key $\mathcal{K}_1[5]$, compute the value of z_7 and update $\mathtt{C}_7[z_7]$ with $\mathtt{C}_7[z_7]+\mathtt{C}_6[z_6]$. The dominant time complexity of this step is $2^{60}\cdot 2^4\cdot 2^{40} = 2^{104}$ memory accesses to a table with 2^{56} elements.

Step 8 Allocate a counter $\mathtt{C}_8[z_8]$ for each of 2^{48} possible values of

$$z_8 = \mathcal{Z}_1[6,27,29]\|\mathcal{Z}_2[3]\|\mathcal{Z}_3[7]\|\mathcal{X}_{29}[8]\|\mathcal{X}_{30}[28]\|\mathcal{X}_{31}[4,7,15,25]\|\mathcal{Z}_{31}[22].$$

For each possible 8-bit key $\mathcal{K}_0[0]\|\mathcal{K}_1[9]$, compute the value of z_8 and update $\mathtt{C}_8[z_8]$ with $\mathtt{C}_8[z_8] + \mathtt{C}_7[z_7]$. The dominant time complexity of this step is $2^{56} \cdot 2^8 \cdot 2^{44} = 2^{108}$ memory accesses to a table with 2^{48} elements.

Step 9 Allocate a counter $\mathtt{C}_9[z_9]$ for each of 2^{44} possible values of

$$z_9 = \mathcal{Z}_1[27]\|\mathcal{Z}_2[3]\|\mathcal{Z}_3[7,28]\|\mathcal{X}_{29}[8]\|\mathcal{X}_{30}[28]\|\mathcal{X}_{31}[4,7,15,25]\|\mathcal{Z}_{31}[22].$$

For each possible 4-bit key $\mathcal{K}_1[14]$, compute the value of z_9 and update $\mathtt{C}_9[z_9]$ with $\mathtt{C}_9[z_9] + \mathtt{C}_8[z_8]$. The dominant time complexity of this step is $2^{48} \cdot 2^4 \cdot 2^{52} = 2^{104}$ memory accesses to a table with 2^{44} elements.

Step 10 Allocate a counter $\mathtt{C}_{10}[z_{10}]$ for each of 2^{40} possible values of

$$z_{10} = \mathcal{Z}_1[27]\|\mathcal{Z}_2[3]\|\mathcal{Z}_4[9]\|\mathcal{X}_{29}[8]\|\mathcal{X}_{30}[28,30]\|\mathcal{X}_{31}[4,15,25]\|\mathcal{Z}_{31}[22].$$

For each possible 4-bit key $\mathcal{K}_1[3]$, compute the value of z_{10} and update $\mathtt{C}_{10}[z_{10}]$ with $\mathtt{C}_{10}[z_{10}] + \mathtt{C}_9[z_9]$. The dominant time complexity of this step is $2^{44} \cdot 2^4 \cdot 2^{56} = 2^{104}$ memory accesses to a table with 2^{40} elements.

Step 11 Allocate a counter $\mathtt{C}_{11}[z_{11}]$ for each of 2^{32} possible values of

$$z_{11} = \mathcal{Z}_1[27]\|\mathcal{Z}_2[3]\|\mathcal{Z}_4[9]\|\mathcal{X}_{28}[26]\|\mathcal{X}_{29}[8]\|\mathcal{X}_{30}[28]\|\mathcal{X}_{31}[15]\|\mathcal{Z}_{31}[22].$$

For each possible 4-bit key $\mathcal{K}_1[12]$, compute the value of z_{11} and update $\mathtt{C}_{11}[z_{11}]$ with $\mathtt{C}_{11}[z_{11}] + \mathtt{C}_{10}[z_{10}]$. The dominant time complexity of this step is $2^{40} \cdot 2^4 \cdot 2^{60} = 2^{104}$ memory accesses to a table with 2^{32} elements.

Step 12 Allocate a counter $\mathtt{C}_{12}[z_{12}]$ for each of 2^{28} possible values of

$$z_{12} = \mathcal{Z}_3[15]\|\mathcal{Z}_4[9]\|\mathcal{X}_{28}[26]\|\mathcal{X}_{29}[8]\|\mathcal{X}_{30}[28]\|\mathcal{X}_{31}[15]\|\mathcal{Z}_{31}[22].$$

For each possible 8-bit key $\mathcal{K}_0[1]\|\mathcal{K}_1[13]$, compute the value of z_{12} and update $\mathtt{C}_{12}[z_{12}]$ with $\mathtt{C}_{12}[z_{12}] + \mathtt{C}_{11}[z_{11}]$. The dominant time complexity of this step is $2^{32} \cdot 2^8 \cdot 2^{64} = 2^{104}$ memory accesses to a table with 2^{28} elements.

Step 13 Allocate a counter $\mathtt{C}_{13}[z_{13}]$ for each of 2^{24} possible values of

$$z_{13} = \mathcal{X}_6[14] \| \mathcal{X}_{28}[26] \| \mathcal{X}_{29}[8] \| \mathcal{X}_{30}[28] \| \mathcal{X}_{31}[15] \| \mathcal{Z}_{31}[22].$$

For each possible 8-bit key $\mathcal{K}_0[4] \| \mathcal{K}_1[7]$, compute the value of z_{13} and update $\mathtt{C}_{13}[z_{13}]$ with $\mathtt{C}_{13}[z_{13}] + \mathtt{C}_{12}[z_{12}]$. The dominant time complexity of this step is $2^{28} \cdot 2^8 \cdot 2^{72} = 2^{108}$ memory accesses to a table with 2^{24} elements.

Step 14 Allocate a counter $\mathtt{C}_{14}[z_{14}]$ for each of 2^{20} possible values of

$$z_{14} = \mathcal{X}_6[14] \| \mathcal{X}_{28}[26] \| \mathcal{X}_{29}[8] \| \mathcal{X}_{30}[17, 28].$$

For each possible 4-bit key $\mathcal{K}_0[2]$, compute the value of z_{14} and update $\mathtt{C}_{14}[z_{14}]$ with $\mathtt{C}_{14}[z_{14}] + \mathtt{C}_{13}[z_{13}]$. The dominant time complexity of this step is $2^{24} \cdot 2^4 \cdot 2^{80} = 2^{108}$ memory accesses to a table with 2^{20} elements.

Step 15 Allocate a counter $\mathtt{C}_{15}[z_{15}]$ for each of 2^{16} possible values of

$$z_{15} = \mathcal{X}_6[14] \| \mathcal{X}_{28}[26] \| \mathcal{X}_{29}[8, 21].$$

For each possible 4-bit key $\mathcal{K}_0[8]$, compute the value of z_{15} and update $\mathtt{C}_{15}[z_{15}]$ with $\mathtt{C}_{15}[z_{15}] + \mathtt{C}_{14}[z_{14}]$. The dominant time complexity of this step is $2^{20} \cdot 2^4 \cdot 2^{84} = 2^{108}$ memory accesses to a table with 2^{16} elements.

Step 16 Allocate a counter $\mathtt{C}_{16}[z_{16}]$ for each of 2^4 possible values of $z_{16} = \mathcal{X}_6[14] \oplus \mathcal{X}_{27}[31]$. For each possible 4-bit key $\mathcal{K}_1[10]$, compute the value of z_{16} and update $\mathtt{C}_{16}[z_{16}]$ with $\mathtt{C}_{16}[z_{16}] + \mathtt{C}_{15}[z_{15}]$. The dominant time complexity of this step is $2^{16} \cdot 2^4 \cdot 2^{88} = 2^{108}$ memory accesses to a table with 2^4 elements.

Step 17 Allocate a counter $\mathtt{T}_j$ for each of 15 zero-correlation linear approximations. Update $\mathtt{T}_j$ according to the value of $\mathtt{C}_{16}[z_{16}]$ and compute the statistic $\mathtt{T}_{\mathrm{MP}}$.

The threshold is set as Θ. If the statistic $\mathtt{T}_{\mathrm{MP}}$ validates the condition $\mathtt{T}_{\mathrm{MP}} < \Theta$, the key guess will be accepted as a candidate. All keys compatible with the guessed 92 key bits are tested exhaustively against a maximum of two plaintext-ciphertext pairs.