# Differential Cryptanalysis of WARP

Je Sen Teh[a,b,*], Alex Biryukov[b]

[a] *School of Computer Sciences, Universiti Sains Malaysia, 11800 Gelugor, Malaysia*
[b] *Interdisciplinary Centre for Security, Reliability and Trust (SnT), Universiti of Luxembourg, 4365 Esch-sur-Alzette, Luxembourg*

**Abstract**

WARP is an energy-efficient lightweight block cipher that is currently the smallest 128-bit block cipher in terms of hardware. It was proposed by Banik et al. in SAC 2020 as a lightweight replacement for AES-128 without changing the mode of operation. This paper proposes key-recovery attacks on WARP based on differential cryptanalysis in single and related-key settings. We searched for differential trails for up to 20 rounds of WARP, with the first 19 having optimal differential probabilities. We also found that the cipher has a strong differential effect, whereby 16 to 20-round differentials have substantially higher probabilities than their corresponding individual trails. A 23-round key-recovery attack was then realized using an 18-round differential distinguisher. Next, we formulated an automatic boomerang search using SMT that relies on the Feistel Boomerang Connectivity Table to identify valid switches. We designed the search as an add-on to the CryptoSMT tool, making it applicable to other Feistel-like ciphers such as TWINE and LBlock-s. For WARP, we found a 21-round boomerang distinguisher which was used in a 24-round rectangle attack. In the related-key setting, we describe a family of 2-round iterative differential trails, which we used in a practical related-key attack on the full 41-round WARP.

*Keywords:* , Differential cryptanalysis, Rectangle attack, Related-key, WARP, GFN

## 1. Introduction

Lightweight cryptography is currently one of the most heavily researched areas in recent years. This is due in part to the widespread use of resource-constrained devices such as smart or IoT devices which transmit sensitive information on a daily basis. Compared to other symmetric-key primitives, lightweight block ciphers have received the most attention in terms of development and cryptanalytic efforts. The first generation of lightweight block ciphers (e.g. PRESENT [1], KATAN [2] and LED [3]) focused on minimizing hardware requirements while the next emphasized latency (PRINCE [4]) and energy (MIDORI [5]) without sacrificing hardware requirements. Although most lightweight block ciphers have block sizes of 64 bits there were a number of 128-bit block ciphers with lower area and/or power requirements than AES such as MIDORI [5] and GIFT-128 [6]. These 128-bit lightweight block ciphers are usually based on the Substitution-Permutation Network (SPN) design paradigm which generally takes up more hardware space due to the inversion of their confusion and diffusion layers.

To overcome this hurdle, Banik et al. adopted the Type-2 Generalized Feistel Network [7] in their 128-bit block cipher called WARP which was proposed in SAC 2020 [8]. The design team consisted of the minds behind multiple well-known lightweight block ciphers such as GIFT, MIDORI and TWINE [9]. The motivation behind designing WARP as a 128-bit cipher with a 128-bit key was to realize a direct replacement for AES-128 without having to change the underlying mode of operation. By adopting MIDORI's S-box (for reduced latency and area) and a simple alternating key schedule, the designers found that WARP only requires 763 Gate Equivalents (GE) for a bit-serial encryption-only circuit and has better energy consumption than MIDORI, which is widely considered the current state-of-the-art in terms of 128-bit low-energy ciphers.

**Related Work.** To the best of our knowledge, the only prior third-party cryptanalysis result for WARP was an attack reported by Kumar and Yadav [10]. By using an 18-round differential trail, the authors were able to perform a key recovery attack on 21 rounds. WARP's designers analyzed its security against differential and linear cryptanalysis based on the number of active S-boxes. They found that WARP has more than 64 active S-boxes after 19 rounds. They also found a 21-round impossible differential distinguisher and a 20-round integral distinguisher for the cipher. A meet-in-the-middle attack is expected to be feasible for at most 32 rounds of WARP. Although no concrete attacks were described, 41 rounds of WARP is expected to be secure against these attacks.

**Our Contributions.** In this paper, we cryptanalyze WARP using differential cryptanalysis [11]. By using an SMT-aided differential search, we found differential trails for up to 20 rounds of WARP, with the first 19 guaranteed to be optimal. These differential trails confirm that the lower bounds provided by the designers cannot be improved. We then performed a differential cluster search for each of these trails and found that WARP has a strong differential effect from round 13 onward. Notably,

*Corresponding author
*Email address:* jesen_teh@usm.my (Je Sen Teh)

| R | Method | Time | Data | Mem | Ref. |
|---|---|---|---|---|---|
| 21 | SK Diff. | $2^{113}$ | $2^{113}$ | $2^{72}$ | [10] |
| 23 | SK Diff. | $2^{106.68}$ | $2^{106.62}$ | $2^{106.62}$ | Sec. 4.1 |
| 24 | SK Rect. | $2^{125.18}$ | $2^{126.06}$ | $2^{127.06}$ | Sec. 4.2 |
| 41 | RK Diff. | $2^{37*}$ | $2^{37}$ | $2^{9.59}$ | Sec. 5.2 |

*Time complexity to recover 60 bits of the key

Table 1: Summary of key-recovery attacks on WARP (SK/RK denotes single-key/related-key)

differentials for 16 to 20 rounds have higher probabilities than their corresponding individual trails by at least a factor of $2^{13}$.

Next, we implemented an automatic search for boomerang (or more specifically, rectangle) distinguishers which includes the Feistel Boomerang Connectivity Table (FBCT) [12]. The boomerang search was written as a new module for the CryptoSMT tool [13] rather than one that was specifically catered to WARP. We showcase its flexibility by also applying it to TWINE and LBlock-s [14]. Using our tool, we were able to find a 21-round boomerang distinguisher for WARP with a differential probability, $\mathsf{DP} = 2^{-121.11}$.

We also performed a search for related-key differential trails for WARP. As a result, we found that WARP has a family of 2-round iterative related-key differential trails with low weight. These iterative trails can be concatenated to form distinguishers for the full 41-round WARP with $\mathsf{DP} = 2^{-40}$. These trails exist due to the interaction between the cipher's nibble-wise permutation, simple alternating key schedule and subkey XOR operation performed *after* the S-box. The interaction between these design elements also led to another interesting observation whereby knowledge of the input difference for a Feistel-subround can be propagated to the next round without having to guess its corresponding subkey. This property was leveraged in all of our key recovery attacks to target specific subkeys.

Finally, we proposed key-recovery attacks on WARP based on the differential distinguishers that were found. In the single-key setting, we have a 23-round differential attack using an 18-round differential distinguisher that has time $T$, data $D$ and memory $M$ complexities of $(T, D, M) = (2^{106.68}, 2^{106.62}, 2^{106.62})$, followed by a 24-round rectangle attack using a 21-round distinguisher with $(T, D, M) = (2^{125.18}, 2^{125.06}, 2^{126.06})$. In the related-key setting, we formulated a 25-round attack using a 19-round related-key differential distinguisher for the purpose of computational verification. 16 bits of the secret key were recovered within 2.5 minutes[1]. We extended the same key-recovery framework and introduced a practical attack on all 41 rounds of WARP using a 35-round related-key distinguisher with $(T, D, M) = (2^{37}, 2^{37}, 2^{9.59})$. Our cryptanalytic results are summarized in Table 1.

## 2. Preliminaries

Notations and abbreviations used in this paper are summarized in Table 2. The rightmost (least significant) bits or nibbles have an index of 0.

| Symbol | Meaning |
|---|---|
| $n$ | Block size in bits |
| $k$ | Key size in bits |
| $\Delta P$ | Plaintext XOR difference |
| $\Delta C$ | Ciphertext XOR difference |
| $\alpha, \beta, \delta, \gamma$ | $n$-bit input and output |
| $\alpha_i^j$ | The $i$-th nibble of an $n$-bit XOR difference, $\alpha$ in round $j$ |
| $X_i^j$ | The $i$-th nibble of an $n$-bit binary variable, $X$ in round $j$ |
| $\#AS$ | Number of active S-boxes |
| $\oplus$ | Binary exclusive-OR (XOR) |
| $\|$ | Binary concatenation |
| DP | Differential probability |
| R | Number of rounds |
| $DDT(x, y)$/ $FBCT(x, y)$ | An entry in the DDT/FBCT for an input $x$ and output $y$ |

Table 2: Symbols and notation

### 2.1. Differential Cryptanalysis

A block cipher maps a set of plaintexts to a set of ciphertexts using a key-dependent round function, $f_j$, where $j \in \mathsf{R}$. The goal of differential cryptanalysis is to find pairs of plaintexts $(P_1, P_2)$ and ciphertexts $(C_1, C_2)$ with a strong correlation between their differences $\alpha = P_1 \oplus P_2$ and $\beta = C_1 \oplus C_2$. The propagation pattern of an input difference $\alpha$ to an output difference $\beta$ is known as a differential characterisitic or trail. A differential trail consists of a sequence of differences,

$$\alpha \xrightarrow{f_1} \alpha^1 \xrightarrow{f_2} \dots \xrightarrow{f_{\mathsf{R}-1}} \alpha^{\mathsf{R}-1} \xrightarrow{f_{\mathsf{R}}} \beta. \tag{1}$$

An adversary must find a differential trail with sufficiently high differential probability,

$$\mathsf{DP} = \mathsf{Pr}(\alpha \xrightarrow{f_1} \dots \xrightarrow{f_{\mathsf{R}}} \beta). \tag{2}$$

Based on the Markov assumption [15] which allows treating a cipher's rounds independently, the differential probability can be computed as

$$\mathsf{DP} \approx \prod_{j=1}^{\mathsf{R}} \mathsf{Pr}(\alpha^{j-1} \xrightarrow{f_j} \alpha^j), \tag{3}$$

where $\alpha^0 = \alpha$ and $\alpha^{\mathsf{R}} = \beta$. A better estimate of the differential probability can be obtained by collecting differential trails that share the same input and output differences,

$$\mathsf{DP} = \mathsf{Pr}(\alpha \to \beta) = \sum_{\alpha^1 \dots \alpha^{\mathsf{R}\text{-}1}} (\alpha \xrightarrow{f_1} \dots \xrightarrow{f_{\mathsf{R}}} \beta). \tag{4}$$

When cryptanalyzing a block cipher, an adversary maximizes the probability of the differential by enumerating as many differential trails as possible, which can be automated using methods such as branch-and-bound algorithms [16, 17, 18], MILP

[1] Verification of our related-key attack and boomerang search tool is publicly available at `https://github.com/jesenteh/warp-attacks`

[19, 20, 21], boolean satisfiability problem (SAT) [22, 23] and satisfiability modulo theory (SMT) solvers [24] as well as a graph-based approach [25].

In the related-key setting, an adversary is allowed to also have a difference in the encryption key, and not only in the plaintext. However, the adversary cannot specify the value of the key itself and the attack must be valid for any pair of keys with the given difference. The related-key model has been used to theoretically cryptanalyze the full rounds of various block ciphers over the years [26, 27, 28, 29, 30, 31, 32]. In the past, there have been practical attacks that rely on the related-key property [33]. Ciphers that are vulnerable to related-key attacks are not recommended for use in protocols where key integrity is not guaranteed [34, 35].

### *2.2. Boomerang and Rectangle Attacks*

The boomerang attack proposed by Wagner [36] is a variant of differential cryptanalysis that concatenates two shorter differentials to form a longer distinguisher. The classical boomerang attack involves decomposing a target cipher, $E$ into two subciphers, $E = E_1 \circ E_0$. We denote the input and output differences of the first or top half of the cipher, $E_1$ as $\alpha$ and $\beta$ while for the lower half, these differences are denoted as $\gamma$ and $\delta$. We denote the probability that $\alpha \xrightarrow{E_0} \beta$ as $p$ and $\gamma \xrightarrow{E_1} \delta$ as $q$. This boomerang structure is illustrated in Figure 1. The expected probability of a boomerang differential is $p^2q^2$, which requires an adversary to make $(pq)^2$ adaptive chosen plaintext and ciphertext queries to distinguish $E$ from an ideal cipher.

The boomerang attack was later reformulated as a chosen plaintext attack called the amplified boomerang [37] or rectangle attack [38] by encrypting many pairs with the input difference $\alpha$ and searching for a quartet which satisfies $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$ when $P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha$. Although the probability of a quartet to be a right quartet is reduced to $2^{-n}p^2q^2$, counting over all possible $\beta$'s and $\delta$'s as long as $\beta \neq \delta$ improves the probability to $2^{-n}\hat{p}^2\hat{q}^2$, where $\hat{p} = (\sum_i \mathsf{Pr}^2(\alpha \xrightarrow{E_0} \beta_i)^{\frac{1}{2}}$ and $\hat{q} = (\sum_j \mathsf{Pr}^2(\gamma_j \xrightarrow{E_1} \delta))^{\frac{1}{2}}$.

Independently chosen $E_1$ and $E_0$ trails may turn out to be incompatible [39]. With the introduction of the sandwich attack [28, 29], the boomerang connectivity table (BCT) [40] and its Feistel counterpart [12], we have a systematic means of enumerating these differential trails while guaranteeing their compatibility. The sandwich attack decomposes the cipher into 3 components, $E = E_1 \circ E_m \circ E_0$, where $E_m$ is the transition in the middle round with a switching probability denoted by $r$. We can calculate $r$ with the help of BCT or FBCT, similar to how the differential probability can be calculated based on the difference distribution table (DDT). The connectivity tables already cover the various switches that have been used in the past to improve the probability of boomerang distinguishers such as the ladder, S-box and Feistel switches [28]. The probability of obtaining a right quartet is now

$$\hat{p}^2\hat{q}^2 = \sum_{i,j}(\hat{p}_i{}^2\hat{q}_j{}^2 r_{i,j}), \tag{5}$$

where $p_i = \mathsf{Pr}(\alpha \xrightarrow{E_0} \beta_i)$, $q_j = \mathsf{Pr}(\gamma_j \xrightarrow{E_1} \delta)$ and $r_{i,j} = \mathsf{Pr}(\beta_i \xrightarrow{E_m} \gamma_j)$. This sandwich distinguisher is illustrated in Figure 2. It is also possible to evaluate $E_m$ that consists of more than 1 round as described by Song et al. [41].

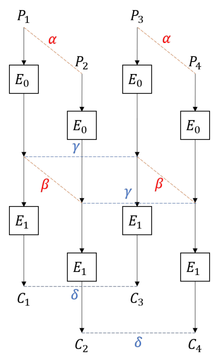



Figure 1: Boomerang attack

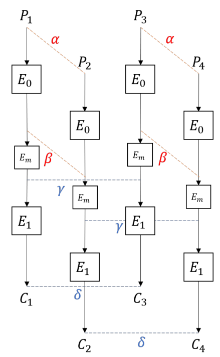



Figure 2: Sandwich attack

| $x$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| $S(x)$ | C | A | D | 3 | E | B | F | 7 |
| $x$ | 8 | 9 | A | B | C | D | E | F |
| $S(x)$ | 8 | 9 | 1 | 5 | 0 | 2 | 4 | 6 |

Table 3: WARP 4-bit S-box

### *2.3. Specification of* WARP

The block cipher WARP is a 41-round, 128-bit block cipher with a 128-bit key designed based on a 32-nibble Type-2 GFN. The $i$-th round's state is divided into 32 nibbles, $X^i = X^i_{31}\|X^i_{30}\|...\|X^i_1\|X^i_0$, where $X^i_j \in \{0,1\}^4$. It has a simple key schedule that first divides the secret key into two 64-bit round keys, $K = K^1\|K^0$, then alternates between them (starting from $K^0$). Each 64-bit round key is divided into 16 nibbles, $K^i = K^i_{15}\|K^i_{14}\|...\|K^i_1\|K^i_0$, where $K^i_j \in \{0,1\}^4$, $i \in \{0,1\}$. The round function is illustrated in Figure 3 while the S-box and permutation pattern, $\pi$ are shown in Tables 3 and 4 respectively. Apart from using the inverse permutation, $\pi^{-1}$, the decryption algorithm is the same.

To avoid the complement property of Feistel-type ciphers [42], the designers of WARP opted for the key XOR operation to be after the S-box, similar to Piccolo [43]. However, this design decision leads to the following property:

**Property 1 (Subround Filters).** *Since XOR with the key is done after the S-box in the Feistel-subround which works on two nibbles, it allows to partially decrypt and propagate the knowledge of the difference to the next round. This can be done for both the top and bottom rounds.*

Based on Figure 4, we can see that partially encrypting $P_1$ and $P_2$ that correspond to the input difference, $\alpha$ allows to immediately check if the given pair is valid if the left nibble of the output difference, $\beta_L$ is known. We can do this without having to guess the corresponding key nibble, $K^j_i$ because the output

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

Table 4: WARP Permutation

difference of the S-box, which we denote as $\gamma$, can be directly computed from the known values of $x_R^1$ and $x_R^2$. Thus, we can check if $\alpha_L \oplus \gamma = \beta_L$ because the effect of the round key has been negated by the XOR operation. The same property exists for the bottom rounds whereby partially decrypting known values of $C_1$ and $C_2$ when $\alpha_L$ is a known difference allows to check if $\beta_L \oplus \gamma = \alpha_L$. We will use these subround filters in our key recovery attacks to filter invalid pairs and determine key candidates associated with valid pairs.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 2 | 4 | 0 | 2 | 2 | 2 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 2 | 0 |
| 2 | 0 | 4 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 4 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 2 | 0 | 4 | 2 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | 2 |
| 4 | 0 | 2 | 4 | 2 | 2 | 2 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 | 0 | 0 |
| 5 | 0 | 2 | 0 | 0 | 2 | 0 | 0 | 4 | 0 | 2 | 4 | 0 | 2 | 0 | 0 | 0 |
| 6 | 0 | 2 | 0 | 4 | 0 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 2 | 0 | 2 |
| 7 | 0 | 0 | 0 | 2 | 0 | 4 | 2 | 0 | 0 | 0 | 0 | 2 | 0 | 4 | 2 | 0 |
| 8 | 0 | 2 | 0 | 2 | 2 | 0 | 2 | 0 | 0 | 2 | 0 | 2 | 2 | 0 | 2 | 0 |
| 9 | 0 | 0 | 4 | 2 | 0 | 2 | 0 | 0 | 2 | 2 | 0 | 2 | 2 | 0 | 0 | 0 |
| A | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 4 | 0 | 4 |
| B | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 2 | 2 | 0 | 4 | 0 | 2 | 0 | 2 |
| C | 0 | 0 | 4 | 0 | 0 | 2 | 2 | 0 | 2 | 2 | 0 | 0 | 2 | 0 | 2 | 0 |
| D | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 4 | 0 | 0 | 4 | 2 | 0 | 0 | 2 | 0 |
| E | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 2 | 4 | 2 |
| F | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 | 0 | 4 | 2 | 0 | 0 | 2 | 4 |

Table 5: Difference Distribution Table of WARP

## 3. Searching for WARP Distinguishers

### 3.1. Differential Distinguishers

We use CryptoSMT [13] to search for both differential trails and differentials for WARP. First, a script was written to generate the SMT model that describes its differential propagation. Then, we enumerate the optimal differential trails for each round and perform differential clustering. Our findings are summarized in Table 6 where $\#AS$ refers to the number of active S-boxes and the weight of a differential trail is calculated as $W = -\log_2$ DP. To find optimal trails (lowest possible weight) for R rounds, we first set the target weight to $\#AS \cdot 2$ for each round, where $\#AS$ is set to the minimum value according to WARP's specification and $-\log_2 \frac{4}{16} = 2$ is the smallest weight for a single S-box, calculated from its difference distribution table (DDT) in Table 5. If a trail was found, we repeat the search by reducing the weight by 1 to confirm its optimality. If no other solution with lower weight can be found by the solver (unsatisfiable), the R-round trail is already optimal. If no trail was found with the minimum weight, we increment the target weight and look for another trail. The first trail found is guaranteed to be optimal.

For up to 19 rounds, we verified that the minimum number of active S-boxes mentioned in WARP's design specification was indeed the lower bound and also found the optimal differential trails for each of these rounds. The time required to find differential trails increased sharply with the number of rounds, compounded with the fact that we are dealing with a 128-bit block size. Finding a trail for Round 17 onward would take up to half a day, longer if a trail did not exist for a particular weight. We managed to find a differential trail for 20 rounds of WARP with a weight of 140 but could not verify its optimality.

Next, we clustered these differentials with a time limit of 24 hours. The results in Table 6 show that for the first 12 rounds, all differentials either had 1 or very few trails each. From Round 13 onward, however, there was a sharp increase in the number of trails. We managed to find all trails for the 13-round to

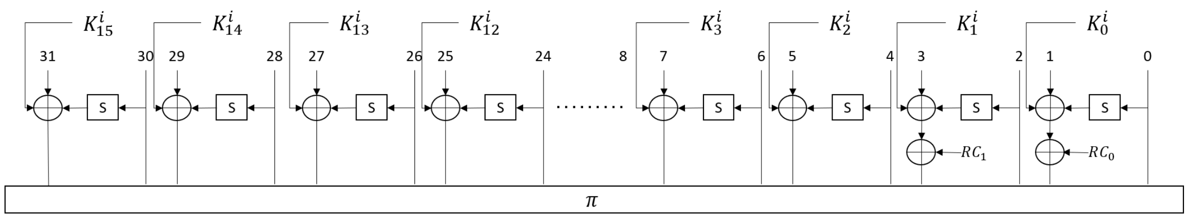
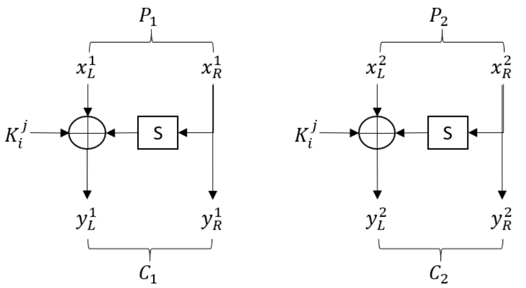



Figure 3: Round Function of WARP

| R | #$AS$ | $W_{opt}$ | Trail $\alpha$ | Trail $\beta$ | Differential $W_{diff}$ | Differential #Trails |
|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0000 0000 0000 0000 0000 0000 0000 0010 | 0000 0000 0000 0000 0000 0000 0100 0000 | 0 | 1 |
| 2 | 1 | 2 | 0000 0000 0000 0000 0000 0000 4000 0000 | 0000 4000 0000 0000 0000 0000 0000 0200 | 2 | 1 |
| 3 | 2 | 4 | 0000 2000 0021 0000 0000 0000 0000 0000 | 0000 0000 0000 0000 0000 0200 0000 0000 | 4 | 1 |
| 4 | 3 | 6 | 0000 0000 002C 0000 0000 0000 0000 0000 | 0000 0000 0000 0400 000C 2000 0000 0000 | 6 | 1 |
| 5 | 4 | 8 | 0000 A000 00AA 0000 0000 0000 0000 0000 | F000 0000 0000 0000 0A00 0000 0F00 0000 | 8 | 1 |
| 6 | 6 | 12 | 0092 0000 0000 0000 0000 0000 9000 0042 | 0000 0000 0000 9002 0000 0002 0000 0000 | 12 | 1 |
| 7 | 8 | 16 | 0000 7DA0 FF00 0000 0000 0000 0000 0000 | 0000 0000 0000 A005 000A 000F 0000 00F0 | 16 | 1 |
| 8 | 11 | 22 | 0000 00FA 5A00 0000 00A0 0000 0000 0000 | 0000 0000 0000 A705 000A 500A 0000 A005 | 22 | 1 |
| 9 | 14 | 28 | 0000 0000 0000 1000 0000 C2C0 4200 0012 | 2900 0020 0000 0000 0120 0000 0104 0001 | 28 | 1 |
| 10 | 17 | 34 | E000 00EE EE00 0000 00E0 00EE 0000 0000 | E000 0000 0E0E 0000 0E0E 00E0 0E00 E00E | 33.19 | 7 |
| 11 | 22 | 44 | 0012 0000 1000 1290 1212 0000 1000 0042 | 2000 0000 0101 0000 0101 0020 0100 2004 | 43.19 | 7 |
| 12 | 28 | 56 | 1212 0000 4000 0042 0012 0000 4000 4240 | 0200 0202 0212 0200 1002 0212 4040 0010 | 55.42 | 5 |
| 13 | 34 | 68 | 0020 2000 0024 2000 0000 0020 2121 0021 | 0010 0202 1000 0000 1200 0240 4000 1202 | 62.37 | 16k |
| 14 | 40 | 80 | 0000 0010 1292 0012 0010 1000 0042 C000 | 0000 1002 0200 4202 40C0 0002 C002 4202 | 72.14 | 22k |
| 15 | 47 | 94 | 0000 00A0 5A5A 005A 00A0 5000 0057 5000 | A500 A005 000A 0700 0AA5 55A5 057A 0AA0 | 85.54 | 497k |
| 16 | 52 | 104 | A000 5AAA 0000 0000 0000 A05A 005A 0000 | 0A00 000A 000A 0000 0057 0A50 005A 500A | 90.52 | 800k |
| 17 | 57 | 114 | 0000 A000 0000 0075 0000 A500 0000 7000 | 000A 5000 0550 0000 AA00 000A 0000 0A00 | 95.66 | 734k |
| 18 | 91 | 122 | 0000 A0AF 005A 0000 A000 AA75 0000 0000 | 000A 5000 0AA0 0000 5A00 000A 0000 0A00 | 104.62 | 627k |
| 19 | 66 | 132 | 5000 A55A 0000 0000 0000 70AA 00A5 0000 | 0500 0050 00A0 0A00 00A5 A00A 5007 000A | 118.07 | 594k |
| 20 | 70* | 140* | 0000 50AA 0057 0000 F000 5AAF 0000 0000 | 0A00 A000 0000 500A 0000 050A 0000 F50A | 122.71 | 545k |

*Number of active S-boxes and/or differential probability not confirmed to be optimal

Table 6: WARP differentials for rounds 1 to 20




$$\alpha = (\alpha_L, \alpha_R) = (x_L^1 \oplus x_L^2, x_R^1 \oplus x_R^2)$$
$$\beta = (\beta_L, \beta_R) = (y_L^1 \oplus y_L^2, y_R^1 \oplus y_R^2)$$

Figure 4: Difference propagation for a pair of nibbles

15-round differentials, which ranged from 16000 to just under 500000 trails. The remaining cluster sizes were bounded by the time limit. The results show that WARP has a significant differential effect at higher rounds, whereby rounds 16 to 20 have an improvement to their differential probabilities by a factor of at least $2^{13.48}$.

### 3.2. Boomerang Distinguishers

To find boomerang distinguishers for WARP, we formulated an automatic boomerang search based on CryptoSMT's differential search functionality. The overall goal of the automatic search is to maximize $\hat{p}^2\hat{q}^2 = \sum_{i,j}(\hat{p}_i{}^2\hat{q}_j{}^2 r_{i,j})$ by finding as many $E_0$ and $E_1$ trails that are compatible. The compatibility of the upper and lower trails is determined using the FBCT[2].

[2]For more information about the FBCT, readers can refer to the work by Boukerrou et al. [12]

WARP's FBCT shown in Table 7 already includes scenarios such as the ladder switch (first row/column) and the Feistel switch (diagonal) where the switching occurs with a probability of 1. The proposed boomerang search procedure is as follows:

1. Search for an $E_0$ trail with $\mathsf{R}_{E_0}$ rounds for up to a weight limit of $W_{upper}$.
2. Search for an $E_1$ trail with $\mathsf{R}_{E_1}$ rounds for up to a weight limit of $W_{lower}$. Limit the search to only compatible trails by propagating $\beta$ from $E_0$ through $E_m$, then including blocking constraints in the SMT model for each of its S-boxes based on entries in the FBCT. If a valid $E_1$ trail is found then:
   (a) If this is the first iteration, fix the input and output differences of the boomerang distinguisher to $\alpha$ and $\delta$ for all future iterations.
   (b) Calculate the switching probability, $r_{i,j}$ based on $\beta$, $\gamma$, the linear layer, $\pi$ and FBCT as
   $$r_{i,j} = \prod_{\substack{k = \\ \{0,2,4,\ldots, \\ 28,30\}}} \frac{FBCT(\beta_k, \gamma_{\pi(k)})}{16}, \tag{6}$$
   where only the even nibbles are involved in the FBCT calculations since WARP is a Feistel cipher.
   (c) For the clustering process, limit the search to $W_{init} + \frac{n}{l}$ where $W_{init}$ is the weight of the initial trail and $l$ controls the upper limit of the search, e.g. for $l = 64$, the upper weight limit of the clustering process is $W_{init} + 2$. Set individual limits for $E_0$ and $E_1$.
   (d) Perform differential clustering for $E_0$ if it has not yet been done. Denote the resulting differential probability as $\hat{p}_i$.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 |
| 1 | 16 | 16 | 4 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 16 | 4 | 16 | 4 | 4 | 0 | 4 | 0 | 0 | 4 | 0 | 4 | 4 | 0 | 4 | 0 |
| 3 | 16 | 4 | 4 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 16 | 0 | 4 | 0 | 16 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 16 | 0 | 0 | 0 | 0 | 16 | 0 | 0 | 0 | 0 | 8 | 0 | 0 | 0 | 0 | 8 |
| 6 | 16 | 0 | 4 | 0 | 4 | 0 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 16 | 0 | 0 | 8 | 0 | 0 | 8 | 0 | 0 |
| 8 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 16 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 16 | 0 | 4 | 0 | 0 | 0 | 0 |
| A | 16 | 0 | 0 | 0 | 0 | 8 | 0 | 8 | 0 | 0 | 16 | 0 | 0 | 8 | 0 | 8 |
| B | 16 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 16 | 0 | 0 | 0 | 0 |
| C | 16 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 16 | 0 | 4 | 0 |
| D | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 8 | 0 | 0 | 8 | 0 | 0 | 16 | 0 | 0 |
| E | 16 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 16 | 0 |
| F | 16 | 0 | 0 | 0 | 0 | 8 | 0 | 0 | 0 | 0 | 8 | 0 | 0 | 0 | 0 | 16 |

Table 7: Feistel Boomerang Connectivity Table of WARP

(e) Perform differential clustering for $E_1$. Denote the resulting differential probability as $\hat{q}_j$

(f) Update the current boomerang probability with $\hat{p}_i{}^2\hat{q}_j{}^2 r_{i,j}$.

(g) Add blocking constraints to the SMT model to prevent the current $E_1$ trail from being found again, then repeat Step 2.

3. If no more valid $E_1$ trails can be found, clear all blocking constraints for $E_1$, add constraints to the SMT model to block the current $E_0$ trail from being found again, then repeat Step 1.

The search itself is generic to Feistel-like ciphers and can be adapted to other design paradigms such as SPN. Several examples of boomerang distinguishers for TWINE and LBlock-s found using the automated search are shown in Table 8[3]. The best boomerang distinguishers that found for WARP are summarized in Table 9.

### 3.3. Related-key Differential Distinguishers

Although its designers do not claim any security in the related-key setting, WARP could possibly be used to design other primitives such as hash functions or used in certain applications for which resilience against related-key attacks are important[4]. We found that WARP has a family of 2-round iterative related-key differential trails:

**Property 2 (2-round Related-key Trails).** *Let $i$ be an odd-numbered index (1,3,...,29,31) of a nonzero nibble in the input difference and $x$ be the nibble's difference. The input difference $\alpha$ consists of all zero nibbles except $\alpha_i = x$. When $K^1_{\frac{\pi^{-1}(i)}{2}} = y$, $K^0_{\frac{(\pi^{-1})^2(i)-1}{2}} = x$ and $K^0_{\frac{i-1}{2}} = x$, we have a 2-round related-key differential trail from $\alpha \rightarrow \alpha$ with $\mathsf{DP} = \frac{DDT(x,y)}{16}$.*

Depending on the DDT (Table 5), these trails can either have a differential probability of $\frac{4}{16} = 2^{-2}$ or $\frac{2}{16} = 2^{-3}$. Figure 5 illustrates two examples of trails described in Property 2. For a more concrete example, we set $i = 3$, $x = 1$ and $y = 2$ and have the following differential propagation that follows the red trail in Figure 5:

$$\texttt{0000...00001000} \xrightarrow[2^{-2}]{2r} \texttt{0000...00001000},$$

where the key difference is $\Delta K = \{\Delta K_1 = \texttt{0000000000200000}, \Delta K_0 = \texttt{0000000010000010}\}$. The trail's differential probability is $\frac{DDT(1,2)}{16} = 2^{-2}$. We can then concatenate this iterative related-key differential trail 20.5 times to construct a 41-round distinguisher $\mathsf{DP} = 2^{-40}$.

## 4. Differential Attacks on WARP

We denote an R-round cipher, $E$ as $E = E_f \circ E' \circ E_b$, where $E'$ is our differential distinguisher. The $\mathsf{R}_b$-round $E_b$ and $\mathsf{R}_f$-round $E_f$ are rounds added before and after the distinguisher, respectively. The input difference of $E_b$ and the output difference of $E_f$ are denoted as $\Delta P$ and $\Delta C$. We denote the number of active or unknown bits of $\Delta P$ as $r_b$ while the $n - r_b$ inactive or fixed bits are denoted as $\hat{r}_b$ and $\bar{r}_b$ for 0s and 1s, respectively. Analogously, these bits are denoted as $r_b$, $\hat{r}_b$ and $\bar{r}_b$ for $\Delta C$. We adopt a targeted approach for the key counting procedure by strategically guessing and filtering $m$ bits of keys involved in subround filters described in Property 1.

### 4.1. 23-round Attack using 18-round Differential

We use the 18-round differential from Table 6 with $\mathsf{DP} = 2^{-104.62}$ to mount an attack on 23-round WARP by adding 2 rounds at the beginning and 3 rounds at the end. The 23-round key recovery model is depicted in Table 10, where we have ($r_b = 56$, $\hat{r}_b = 56$, $\bar{r}_b = 16$) and ($r_f = 72$, $\hat{r}_f = 46$, $\bar{r}_f = 10$). We guess a total of $m = 56$ subkey bits, corresponding to $K^0_i$ and $K^1_j$, where $i = \{0, 1, 4, 5, 7, 8, 9, 11, 14\}$ and $j = \{2, 4, 11, 13, 14\}$.

**Data Preparation.** We let $s = 2$ and collect $y = 2 \cdot 2^{-56} \cdot \frac{2}{2^{-104.62}} \approx 2^{50.62}$ structures of $2^{56}$ plaintexts each. The plaintexts traverse all possible values for the active $r_b$ bits while the $\hat{r}_b$ and $\bar{r}_b$ bits are assigned suitable constants. Notably, half of the plaintexts should have the $\bar{r}_b$ bits set to 0 while the other half has these set to 1. We encrypt all $2^{56}$ plaintexts to obtain $2^{56}$ corresponding ciphertexts of all structures that are stored in a hash table $H$, according to the 46 $\hat{r}_f$ bits set to 0. For each pair of structures, we have $2^{111}$ pairs at the beginning.

[3] These distinguishers only serve to showcase the flexibility of the proposed boomerang search, and may not be the best boomerang distinguishers found for these ciphers.

[4] Other block ciphers such as GIFT have also been extensively cryptanalyzed using related-key attacks despite not claiming any security in this setting [44, 45].

| Cipher | R | $\alpha$ | $\delta$ | $\sum_{i,j}(\hat{p}_i{}^2\hat{q}_j{}^2 r_{i,j})$ |
|---|---|---|---|---|
| TWINE | 15 (7+1+7) | 3890 0000 0097 0000 | 0DB0 0010 0D00 0C00 | $2^{-58.92}$ |
| TWINE | 16 (8+1+7) | A250 0000 0056 0000 | A000 0702 0050 0002 | $2^{-61.62}$ |
| LBlock-s | 15 (7+1+7) | 0420 0004 0600 0004 | 6600 0000 4020 0004 | $2^{-58.64}$ |

Table 8: Boomerang distinguishers for other ciphers

| R | $\alpha$ | $\delta$ | $p^2q^2r$ | $\sum_{i,j}(\hat{p}_i{}^2\hat{q}_j{}^2 r_{i,j})$ |
|---|---|---|---|---|
| 20 (9+1+10) | 0000 0000 0000 1000 0000 C2C0 4200 0012 | 0202 0040 0200 1002 4000 0000 0202 0000 | $2^{-124}$ | $2^{-114.24}$ |
| 21 (10+1+10) | E000 00EE EE00 0000 00E0 00EE 0000 0000 | 2000 0000 0104 0000 0404 0020 0100 2004 | $2^{-142}$ | $2^{-121.11}$ |

Table 9: Boomerang distinguishers for WARP

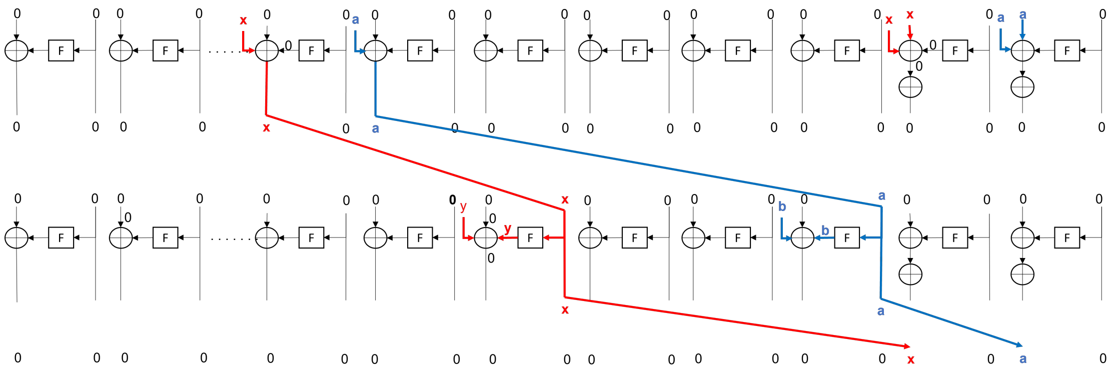


Figure 5: Two examples of the 2-round iterative related-key differential trails for WARP where $i = 3$ and $i = 1$ are represented by the red and blue trails respectively (Property 2)

| R | Input Difference ($\Delta P$) | ???? A000 A0?? ???A ?F?? 0000 A0A0 ??50 |
|---|---|---|
| 1 | After S-box ($\Delta X^1$)<br>After $\pi$ ($\Delta Y^1$) | 0?0? A000 A07? 0?0A 0F0? 0000 A0A0 0?50<br>00?7 00?A F000 00?A A0?A 00?A ?500 0000 |
| 2 | After S-box ($\Delta X^2$)<br>After $\pi$ ($\Delta Y^2$) | 00A7 000A F000 000A A00A 000A 5500 0000<br>0000 A0AF 005A 0000 A000 AA75 0000 0000 |
| 20 | Differential distinguisher,<br>$\alpha \to \beta$ | 0000 A0AF 005A 0000 A000 AA75 0000 0000<br>000A 5000 0AA0 0000 5A00 000A 0000 0A00 |
| 21 | After S-box ($\Delta X^{21}$)<br>After $\pi$ ($\Delta Y^{21}$) | 00?A 5000 ?AA0 0000 ?A00 00?A 0000 ?A00<br>00AA A00? A00? 0005 0?00 0?A0 00A0 0?00 |
| 22 | After S-box ($\Delta X^{22}$)<br>After $\pi$ ($\Delta Y^{22}$) | 00?A A0?? A0?? 00?5 ??00 ??A0 00A0 ??00<br>00?? 000A ??0? 0??A 5?0A ??A0 0000 ?A0? |
| 23 | After S-box ($\Delta X^{23}$)/<br>Output Difference ($\Delta C$) | 00?? 00?A ???? ???A ???A ??A0 0000 ?A?? |

Additional Notes: ? denotes an undetermined nibble. Red text denotes subround filters based on Property 1.

Table 10: The 23-round key recovery model for WARP using an 18-round differential

**Key Recovery.** We initialize a list of $2^{56}$ counters then:

1. We filter wrong pairs using inactive bits of $\Delta C$, leaving $2^{111-56} = 2^{55}$ pairs per structure or $2^{50.62+55} = 2^{105.62}$ pairs in total.
2. The number of filters (Property 1) in the first and final rounds are $v_b = 8$ and $v_f = 6$, respectively as indicated in red in Table 10. Thus, the number of valid pairs would be reduced to $2^{105.62-4\times 8-4\times 6} = 2^{49.62}$.
3. **In Round 23:** We can propagate knowledge of the output difference to Round 22 without having to guess any keys (Property 1).
4. **In Round 22:** In this round, guess and filter subkeys for each remaining pair based on Property 1. For example, guess $K^0_{14}$ to partially decrypt $\Delta X^{23}_{29}$ to calculate $\Delta X^{22}_2 = \Delta Y^{22}_{29}$. Directly compute $\Delta X^{22}_3 = \Delta Y^{22}_{14} = \Delta X^{23}_{14}$ from the ciphertext pairs. Check if $\Delta X^{22}_3 \oplus S(\Delta X^{22}_2) = \Delta Y^{21}_3 = 0$. If the equality holds, keep the guessed value of $K^0_{14}$ and the pair, otherwise discard them. There will be around $2^{49.62} \cdot 2^4 \cdot 2^{-4} = 2^{49.62}$ combinations of the remaining pairs associated with the guessed $K^0_{14}$ values. We have 6 more of these filters in Round 22, for which we can guess and filter $K^0_1$, $K^0_4$, $K^0_5$, $K^0_7$, $K^0_8$ and $K^0_{11}$ candidates. We expect to have $2^{49.62}$ combinations of the remaining pairs associated with 28-bit key candidates.
5. **In Round 21:** Guess $K^1_{14}$, $K^1_{11}$ and $K^1_4$ to calculate $\Delta X^{21}_2$, $\Delta X^{21}_{14}$ and $\Delta X^{21}_{28}$ respectively, while the remaining differences can be calculated based on the previous key guesses. After going through these filters, there will be $2^{49.62}$ combinations of the remaining pairs associated with 40-bit key candidates. For the remaining two filters, guess $(K^0_9, K^1_{13})$ to calculate $\Delta X^{21}_8$ and $(K^0_0, K^1_2)$ to calculate $\Delta X^{21}_{22}$. Since there are $2^8$ possible subkey candidates involved in each of these 4-bit filters, this will increase the number of combinations to $2^{49.62} \cdot 2^{4\times 2} = 2^{57.62}$ pairs associated with 56-bit keys.
6. **In Round 1:** For all the remaining pairs, we propagate knowledge of the input difference to Round 2.
7. **In Round 2:** Differences $\Delta Y^1_6$ and $\Delta Y^1_{24}$ can be calculated using $K^0_0$ and $K^0_{11}$ candidates already associated with each remaining pair. $\Delta Y^1_7$ and $\Delta Y^1_{25}$ can be calculated from the plaintext pairs. We then discard combinations of pairs and keys based on the known differences, $\Delta X^2_7 = 5$ and $\Delta X^2_{25} = 0$ due to Property 1. This reduces the number of possible combinations to $2^{57.62-4\times 2} = 2^{49.62}$.
8. Increment the key counters based on the $2^{49.62}$ remaining combinations of pairs associated with the 56 bits of guessed keys. We expect 2 pairs to vote for the right key while the remaining pairs will vote for a random key with a probability of $2^{49.62-56} = 2^{-6.38}$.
9. We select the top $2^{m-a} = 2^{56-52} = 16$ hits in the counter to be candidates that deliver an $a$-bit or higher advantage [46], then brute-force the 72 remaining bits of the secret key.

**Complexity Estimation.** The data and memory complexities are $N = 2^{50.62} \cdot 2^{56} \approx 2^{106.62}$ plaintexts and $2^{106.62} + 2^{56} \cdot \frac{56}{128} \approx 2^{106.62}$ 128-bit blocks, respectively. The time complexity of the key recovery is dominated by the final round filtering in Step 2, in which the $2^{105.62}$ pairs need to be partially decrypted. This requires $2^{105.62} \cdot \frac{2}{23} \approx 2^{102.09}$ 23-round WARP encryptions. The brute force complexity is $2^{128-a} = 2^{76}$. Therefore, the time complexity of the 23-round differential attack, including data preparation, is about $2^{106.62} + 2^{102.09} + 2^{76} \approx 2^{106.68}$ 23-round WARP encryptions when $a = 52$.

**Success Probability.** We calculate the probability of success, $\mathsf{Pr_S}$ of our attack based on the method proposed by Selçuk [46]:

$$\mathsf{Pr_S} = \Phi\left(\frac{\sqrt{s \cdot S_N} - \Phi^{-1}(1 - 2^{-a})}{\sqrt{S_N + 1}}\right), \tag{7}$$

where the signal-to-noise ratio is calculated as $S_N = \frac{\mathsf{DP}}{2^{-n}}$. With $a = 52$, the probability that the attack succeeds is 92.09%.

### 4.2. *24-round Rectangle Attack using 21-round Boomerang Distinguisher*

We use the 21-round boomerang distinguisher from Table 9 where $\mathsf{R}_{E_0} = 10$, $\mathsf{R}_{E_1} = 10$ and $\sum_{i,j} \hat{p}_i{}^2 \hat{q}_j{}^2 r_{i,j}) = 2^{-121.11}$ to mount a rectangle attack on 24-round WARP by appending 1 round at the beginning and 2 rounds at the end. The 24-round key recovery model is depicted in Table 11, which has ($r_b = 20$, $\hat{r}_b = 84$, $\bar{r}_b = 24$) and ($r_f = 60$, $\hat{r}_f = 61$, $\bar{r}_f = 7$). The number of subkey bits that will be guessed are $m_f = 16$, corresponding to $K^1_j$ where $j = \{2, 8, 10, 15\}$. Details of our rectangle attack are as follows:

**Data Preparation.** For $s = 2$ right quartets, collect $y = \frac{\sqrt{2} \cdot 2^{64-20}}{\sqrt{2^{-121.11}}} = 2^{105.06}$ structures of $2^{20}$ plaintexts each. The plaintexts are assigned all possible combinations of the $r_b$ active bits while the other bits are assigned suitable constants. Encrypt $2^{20}$ plaintexts of each structure to obtain $2^{20}$ corresponding ciphertexts, which are stored in a hash table, $H_1$ indexed by the $r_b$ bits of the plaintext.

**Key Recovery.** Initialize a list of $2^{16}$ counters then:

1. Construct a set $S = \{(P_1, C_1, P_2, C_2) : E_b(P_1) \oplus E_b(P_2) = \alpha\}$ without having to guess any keys in $E_b$ as follows:
   (a) For every plaintext $P_1$ in a structure, determine the known $\hat{r}_b + \bar{r}_b$ bits in $P_2$ by calculating $P_2 = P_1 \oplus \Delta P$.
   (b) The unknown nibbles in $P_2$, which are all left input nibbles to Feistel-subrounds, can be calculated from their corresponding right input nibbles. Let the pairs of nibbles for $P_1$ and $P_2$ be denoted as ($x^1_L$, $x^1_R$) and ($x^2_L$, $x^2_R$) respectively (see Figure 4). We already know the values for the right halves ($x^1_R$, $x^2_R$) after Step 1(a) and we also know the value of $x^1_L$ from $P_1$. We can then calculate the remaining unknown value as

$$x^2_L = x^1_L \oplus S(x^1_R) \oplus S(x^2_R).$$

| R | Input Difference ($\Delta P$) | 00?E 0000 E000 ?EE0 ?E?E 0000 E000 00?E |
|---|---|---|
| 1 | After S-box ($\Delta X^1$)<br>After $\pi$ ($\Delta Y^1$) | 000E 0000 E000 0EE0 0E0E 0000 E000 000E<br>E000 00EE EE00 0000 00E0 00EE 0000 0000 |
| 22 | Boomerang distinguisher,<br>$\alpha \to \delta$ | E000 00EE EE00 0000 00E0 00EE 0000 0000<br>2000 0000 0104 0000 0404 0020 0100 2004 |
| 23 | After S-box ($\Delta X^{23}$)<br>After $\pi$ ($\Delta Y^{23}$) | 2000 0000 ?1?4 0000 ?4?4 0020 ?100 20?4<br>400? 024? 4010 0040 0200 0?0? 0?1? 0200 |
| 24 | After S-box ($\Delta X^{24}$)/<br>Output Difference ($\Delta C$) | 40?? ?2?? 4010 0040 ?200 ???? ???? ?200 |

? denotes an undetermined nibble. Red text denotes subround filters based on Property 1.

Table 11: 24-round key recovery model of the rectangle attack using a 21-round (10+1+10) distinguisher

   (c) After calculating all the unknown bits of $P_2$, check $H_1$ to find the corresponding plaintext-ciphertext pair indexed by the $r_b$ bits of $P_2$. Since $v_b = 5$, we expect $2^{20\times 2-1} \cdot 2^{-4\times v_b} = 2^{19}$ pairs in $S$.

2. The size of $S$ is $N = 2^{105.06} \cdot 2^{(19+1)\times 2^{-1}} = 2^{115.06}$ chosen plaintexts. Insert $S$ into a hash table $H_2$ indexed by the 61 $\hat{r}_f$ bits of $C_1$ and $C_2$. For each element of $S$, check $H_2$ to find $(P_1, C_1, P_2, C_2)$ where $(C_1, C_3)$ and $(C_2, C_4)$ collide in the $\hat{r}_f + \bar{r}_f = 68$ known bits. There will be $(2^{115.06})^2 \cdot 2^{-2\times 68} = 2^{94.12}$ quartets remaining.
3. Since the number of subround filters is $v_f = 9$, the number of valid quartets would be reduced to $2^{94.12-8\times 9} = 2^{22.12}$. The filtering effect due to Property 1 is twofold since it is applicable to both pairs in a quartet.
4. **In Round 24:** Propagate the knowledge of the output difference to Round 23 (Property 1).
5. **In Round 23:** Perform the guess-and-filter procedure for subkeys in Round 23. For example, $\Delta X_0^{23}$ can be directly computed from the ciphertext pairs. Guess $K_{15}^1$ and partially decrypt $(C_1, C_3)$ and $(C_2, C_3)$ to obtain $\Delta X_1^{23}$ for each pair. Check if each pair in the quartet fulfills $\Delta X_1^{23} \oplus S(\Delta X_0^{23}) = \delta_1 = 0$. If the equality holds for both pairs in the quartet, keep the guessed key and the quartet, otherwise, discard them. There will be around $2^{22.12} \cdot 2^4 \cdot 2^{-8} = 2^{18.12}$ combinations of the remaining quartets associated with the guessed $K_{15}^1$ values. Guess and filter 3 more subkeys, $K_2^1$, $K_8^1$ and $K_{10}^1$, which leaves $2^{18.12} \cdot 2^{-4\times 3} = 2^{6.12}$ combinations of the remaining quartets associated with the guessed keys.
6. Increment the key counters based on the $2^{6.12}$ remaining combinations of quartets associated with the 16 bits of guessed keys. On average, 2 quartets will vote for the right key while the remaining quartets will vote for a random key with a probability of $2^{6.12-16} = 2^{-7.88}$.
7. Select the top $2^{16-12} = 16$ hits in the counter and brute force the 112 remaining bits of the secret key.

**Complexity Estimation.** The data complexity of the attack is $N = 2^{105.06} \cdot 2^{20} \approx 2^{125.06}$ chosen plaintexts. The memory complexity includes the space required to store the hash tables and the key counters, which is $2 \cdot 2^{125.06} + 2^{24} \cdot \frac{24}{128} \approx 2^{126.06}$ 128-bit blocks. To prepare the quartets, we require around $2^{125.06}$ 24-round encryptions and $2N$ memory accesses, for which we make a conservative assumption is equivalent to 1 encryption round. The time complexity of the key recovery is dominated by the final round filtering in Step 3, which is approximately $\theta = 2^{94.12} \cdot \frac{4}{24} \approx 2^{91.54}$. The overall time complexity of the 24-round attack is $2^{125.06} + 2 \cdot 2^{125.06} \cdot \frac{1}{24} + 2^{93.54} + 2^{116} \approx 2^{125.18}$ 24-round WARP encryptions when $a = 12$.

**Success Probability.** When $a = 12$ and $S_N = \frac{\sum_{i,j}(\hat{p}_i{}^2 \hat{q}_j{}^2 r_{i,j})}{2^{-n}} = 2^{-121.11}$, the probability that the attack succeeds is 86.2%.

## 5. Related-key Differential Attacks on WARP

### 5.1. 25-round Related-key Differential Attack

We concatenate 9 instances of the 2-round iterative related-key differential trail described in subsection 3.3 and append 1 more round to form a 19-round distinguisher with DP= $2^{-18}$. After appending 6 rounds to the end of this distinguisher, we have a 25-round key-recovery model depicted in Table 12 where $r = 19$. We guess a total of 16 subkey bits, corresponding to $K_i^0$ where $i = \{4, 7, 10, 14\}$. Although it may be possible to guess more key bits to reduce the computational complexity of the final brute force step, we stick with 16 bits so we can computationally verify the attack efficiently.

**Data Preparation.** Encrypt $2^{19}$ pairs of plaintexts, $(P_1, P_2)$ using a pair of related keys, $(K, K \oplus \Delta K)$. We expect $s = 2$ right pairs. There is a strong filtering effect at the output difference $\Delta C$, which has 60 inactive bits and 5 subround filters (Property 1). The probability of a wrong pair surviving is $2^{20} \cdot 2^{-60} \cdot 2^{-4\times 5} = 2^{-60}$, which implies that only the right pairs remain.

**Key Recovery.** For all the remaining (right) pairs:

1. **In Round 25:** Propagate knowledge of the output difference to Round 24 without having to guess any keys (Property 1).
2. **In Round 24:** Guess $K_{14}^0$ to calculate $2_2^{24}$, then derive $2_3^{24}$ from the ciphertext pairs. Since all pairs are valid,

| R ($\Delta K$) | | |
|---|---|---|
| $0 - r$ | RK distinguisher,<br>$\alpha \rightarrow \beta$ | 0000 0000 0000 0000 0000 0000 0000 1000<br>0000 0000 0000 0000 0000 0100 0000 0000 |
| $r+1$<br>($\Delta K_1$) | After S-box ($\Delta X^{r+1}$)<br>After $\pi$ ($\Delta Y^{r+1}$) | 0000 0000 0000 0000 0000 ?100 0000 0000<br>0000 0000 0000 0000 0000 0000 0000 100? |
| $r+2$<br>($\Delta K_0$) | After S-box ($\Delta X^{r+2}$)<br>After $\pi$ ($\Delta Y^{r+2}$) | 0000 0000 0000 0000 1000 0000 0000 00??<br>?000 0000 0000 0000 0000 0100 0?00 0000 |
| $r+3$<br>($\Delta K_1$) | After S-box ($\Delta X^{r+3}$)<br>After $\pi$ ($\Delta Y^{r+3}$) | ?000 0000 0000 0000 0000 ?100 ??00 0000<br>0000 0?00 00?0 0000 0000 000? 0000 100? |
| $r+4$<br>($\Delta K_0$) | After S-box ($\Delta X^{r+4}$)<br>After $\pi$ ($\Delta Y^{r+4}$) | 0000 ??00 00?0 0000 1000 00?? 0000 00??<br>?00? ?000 0000 ?00? 0000 0100 0?00 0?00 |
| $r+5$<br>($\Delta K_1$) | After S-box ($\Delta X^{r+5}$)<br>After $\pi$ ($\Delta Y^{r+5}$) | ?0?? ?000 0000 ?0?? 0000 ?100 ??00 ??00<br>0??0 0?00 0??? 000? ??00 00?? 0000 100? |
| $r+6$<br>($\Delta K_0$) | After S-box ($\Delta X^{r+6}$)<br>Output Difference ($\Delta C$) | ???0 ??00 ???? 00?? ??00 00?? 0000 00?? |

? denotes an undetermined nibble. Red text denotes subround filters based on Property 1.

Table 12: Key recovery model using an $r$-round related key distinguisher where $r \in \{3, 5, ..., 31\}$

each pair will be associated with at least one possible 4-bit key candidate. Guess $K^0_{10}$, $K^0_7$ and $K^0_4$ associated with the remaining subround filters.

**Complexity Estimation.** The data complexity of the attack is $N = 2 \cdot 2^{19} = 2^{20}$ chosen plaintexts and it finds correctly 16-bits of the key, which is sufficient for verification of the attack correctness. The memory requirement of the attack is negligible.

**Computational Verification.** We first calculated the average probability of the 19-round distinguisher. Using 10 randomly selected keys and plaintext pairs, the average differential probability was $2^{18.1}$. We then execute the 25-round attack 10 times on a PC with an Intel Core i7-9700K 3.60GHz processor and 32GB of RAM. The correct subkey always has the highest count of $s$. The correct 16-bit key will be among the top 2 candidates 70% of the time. On average, the attack completes in under 2.5 minutes using an unoptimized Python implementation of WARP that performs around $2^{12.68}$ encryptions per second. Thus the attack time complexity is around $2^{20}$ WARP encryptions, which is dominated by time required to encrypt the chosen plaintexts.

### 5.2. *41-round (Full) Related-key Differential Attack*

The key recovery model for a 41-attack using a 35-round distinguisher with DP $= 2^{-34}$ is depicted in Table 12 where $r = 35$. We generate $2^{35}$ pairs and expect 2 right pairs. From Round 40 to Round 36, we guess a total of 60 subkey bits (16 in Round 40, 12 in Round 39, 16 bits in 38, 12 in Round 37 and 4 in Round 36). There are subround filters in Rounds 37 and 38 that require guessing 12 key bits, key counters that can accommodate $2^{12}$ possibilities are required. The memory requirement is $(6 \cdot 2^4 \cdot \frac{4}{128} + 2 \cdot 2^{12} \cdot \frac{12}{128}) \approx 2^{9.59}$ 128-bit blocks. The time complexity for the guess-and-determine procedure is negligible, therefore recovering 60 bits of the key comes mainly from encrypting the $2^{36}$ chosen plaintexts. We can either brute force the remaining 68 bits, which would then dominate the time complexity or use faster auxiliary techniques to find the rest of the key.

## 6. Conclusion

This paper described cryptanalytic attacks on the lightweight block cipher WARP. We show that its first 20 rounds have high-probability differentials due to a strong differential effect. Then, by using an automatic search for boomerang distinguishers, boomerang distinguishers for up to 21 rounds were found. We also described a family of 2-round iterative related-key trails which can be concatenated to form a full 41-round distinguisher for WARP. Key-recovery attacks were then demonstrated using the identified distinguishers. In the single-key setting, we attacked 23 and 24 rounds of WARP using an 18-round differential and 21-round boomerang distinguisher, respectively. Next, we computationally verified a 25-round related-key attack on WARP using a 19-round distinguisher, where 16 subkey bits were recovered in around 2.5 minutes. Using the same framework, a practical related-key attack on the full WARP was introduced. All attack complexities were summarized in Table 1. To the best of our knowledge, these are the best 3rd party cryptanalysis results for WARP.

## CRediT Authorship Contribution Statement

**Je Sen Teh:** Conceptualization, Methodology, Validation, Investigation, Resources, Data Curation, Writing - Original Draft, Writing - Review and Editing, Visualization; **Alex Biryukov:** Resources, Validation, Writing - Review and Editing, Project Administration, Supervision;

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.


## Acknowledgements

Je Sen Teh was supported by the Luxembourgish Fonds National de la Recherche (FNR) and the German Research Foundation (DFG) project APLICA (C19/IS/13641232). The final version of this paper has been published in the Journal of Information Security and Applications which can be accessed at `https://doi.org/10.1016/j.jisa.2022.103316`.


## References


[1] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, C. Vikkelsoe, PRESENT: An ultra-lightweight block cipher, in: P. Paillier, I. Verbauwhede (Eds.), CHES 2007, Vol. 4727 of LNCS, Springer, Heidelberg, 2007, pp. 450–466. doi:10.1007/978-3-540-74735-2_31.

[2] C. De Cannière, O. Dunkelman, M. Knežević, KATAN and KTANTAN - a family of small and efficient hardware-oriented block ciphers, in: C. Clavier, K. Gaj (Eds.), CHES 2009, Vol. 5747 of LNCS, Springer, Heidelberg, 2009, pp. 272–288. doi:10.1007/978-3-642-04138-9_20.

[3] J. Guo, T. Peyrin, A. Poschmann, M. J. B. Robshaw, The LED block cipher, in: B. Preneel, T. Takagi (Eds.), CHES 2011, Vol. 6917 of LNCS, Springer, Heidelberg, 2011, pp. 326–341. doi:10.1007/978-3-642-23951-9_22.

[4] J. Borghoff, A. Canteaut, T. Güneysu, E. B. Kavun, M. Knežević, L. R. Knudsen, G. Leander, V. Nikov, C. Paar, C. Rechberger, P. Rombouts, S. S. Thomsen, T. Yalçin, PRINCE - A low-latency block cipher for pervasive computing applications - extended abstract, in: X. Wang, K. Sako (Eds.), ASIACRYPT 2012, Vol. 7658 of LNCS, Springer, Heidelberg, 2012, pp. 208–225. doi:10.1007/978-3-642-34961-4_14.

[5] S. Banik, A. Bogdanov, T. Isobe, K. Shibutani, H. Hiwatari, T. Akishita, F. Regazzoni, Midori: A block cipher for low energy, in: T. Iwata, J. H. Cheon (Eds.), ASIACRYPT 2015, Part II, Vol. 9453 of LNCS, Springer, Heidelberg, 2015, pp. 411–436. doi:10.1007/978-3-662-48800-3_17.

[6] S. Banik, S. K. Pandey, T. Peyrin, Y. Sasaki, S. M. Sim, Y. Todo, GIFT: A small present - towards reaching the limit of lightweight encryption, in: W. Fischer, N. Homma (Eds.), CHES 2017, Vol. 10529 of LNCS, Springer, Heidelberg, 2017, pp. 321–345. doi:10.1007/978-3-319-66787-4_16.

[7] Y. Zheng, T. Matsumoto, H. Imai, On the construction of block ciphers provably secure and not relying on any unproved hypotheses, in: G. Brassard (Ed.), CRYPTO'89, Vol. 435 of LNCS, Springer, Heidelberg, 1990, pp. 461–480. doi:10.1007/0-387-34805-0_42.

[8] S. Banik, Z. Bao, T. Isobe, H. Kubo, F. Liu, K. Minematsu, K. Sakamoto, N. Shibata, M. Shigeri, WARP : Revisiting GFN for lightweight 128-bit block cipher, in: O. Dunkelman, M. J. J. Jr., C. O'Flynn (Eds.), Selected Areas in Cryptography - SAC 2020 - 27th International Conference, Halifax, NS, Canada, October 21-23, 2020, Vol. 12804 of Lecture Notes in Computer Science, Springer, 2020, pp. 535–564. doi:10.1007/978-3-030-81652-0_21.
URL https://doi.org/10.1007/978-3-030-81652-0_21

[9] T. Suzaki, K. Minematsu, S. Morioka, E. Kobayashi, TWINE : A lightweight block cipher for multiple platforms, in: L. R. Knudsen, H. Wu (Eds.), SAC 2012, Vol. 7707 of LNCS, Springer, Heidelberg, 2013, pp. 339–354. doi:10.1007/978-3-642-35999-6_22.

[10] M. Kumar, T. Yadav, MILP based differential attack on round reduced WARP, Cryptology ePrint Archive, Report 2020/1598, Version 20210802:090929, https://ia.cr/2020/1598 (2020).

[11] E. Biham, A. Shamir, Differential cryptanalysis of DES-like cryptosystems, Journal of Cryptology 4 (1) (1991) 3–72. doi:10.1007/BF00630563.

[12] H. Boukerrou, P. Huynh, V. Lallemand, B. Mandal, M. Minier, On the feistel counterpart of the boomerang connectivity table introduction and analysis of the FBCT, IACR Trans. Symmetric Cryptol. 2020 (1) (2020) 331–362. doi:10.13154/tosc.v2020.i1.331-362.
URL https://doi.org/10.13154/tosc.v2020.i1.331-362

[13] Stefan Kölbl, CryptoSMT: An easy to use tool for cryptanalysis of symmetric primitives, https://github.com/kste/cryptosmt.

[14] L. Zhang, W. Wu, Y. Wang, S. Wu, J. Zhang, LAC, Tech. rep., available at https://competitions.cr.yp.to/round1/lacv1.pdf (2014).

[15] X. Lai, J. L. Massey, S. Murphy, Markov ciphers and differential cryptanalysis, in: D. W. Davies (Ed.), EUROCRYPT'91, Vol. 547 of LNCS, Springer, Heidelberg, 1991, pp. 17–38. doi:10.1007/3-540-46416-6_2.

[16] M. Matsui, On correlation between the order of S-boxes and the strength of DES, in: A. D. Santis (Ed.), EUROCRYPT'94, Vol. 950 of LNCS, Springer, Heidelberg, 1995, pp. 366–375. doi:10.1007/BFb0053451.

[17] A. Biryukov, V. Velichkov, Automatic search for differential trails in ARX ciphers, in: J. Benaloh (Ed.), CT-RSA 2014, Vol. 8366 of LNCS, Springer, Heidelberg, 2014, pp. 227–250. doi:10.1007/978-3-319-04852-9_12.

[18] J. Chen, J. Teh, Z. Liu, C. Su, A. Samsudin, Y. Xiang, Towards accurate statistical analysis of security margins: New searching strategies for differential attacks, IEEE Trans. Computers 66 (10) (2017) 1763–1777. doi:10.1109/TC.2017.2699190.
URL https://doi.org/10.1109/TC.2017.2699190

[19] N. Mouha, Q. Wang, D. Gu, B. Preneel, Differential and linear cryptanalysis using mixed-integer linear programming, in: C. Wu, M. Yung, D. Lin (Eds.), Information Security and Cryptology - 7th International Conference, Inscrypt 2011, Beijing, China, November 30 - December 3, 2011., Vol. 7537 of Lecture Notes in Computer Science, Springer, 2011, pp. 57–76. doi:10.1007/978-3-642-34704-7_5.
URL https://doi.org/10.1007/978-3-642-34704-7_5

[20] S. Sun, L. Hu, P. Wang, K. Qiao, X. Ma, L. Song, Automatic security evaluation and (related-key) differential characteristic search: Application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers, in: P. Sarkar, T. Iwata (Eds.), ASIACRYPT 2014, Part I, Vol. 8873 of LNCS, Springer, Heidelberg, 2014, pp. 158–178. doi:10.1007/978-3-662-45611-8_9.

[21] B. Zhu, X. Dong, H. Yu, MILP-based differential attack on round-reduced GIFT, in: M. Matsui (Ed.), CT-RSA 2019, Vol. 11405 of LNCS, Springer, Heidelberg, 2019, pp. 372–390. doi:10.1007/978-3-030-12612-4_19.

[22] N. Mouha, B. Preneel, Towards finding optimal differential characteristics for ARX: Application to Salsa20, Cryptology ePrint Archive, Report 2013/328, https://eprint.iacr.org/2013/328 (2013).

[23] L. Sun, W. Wang, M. Wang, Accelerating the search of differential and linear characteristics with the SAT method, IACR Trans. Symmetric Cryptol. 2021 (1) (2021) 269–315. doi:10.46586/tosc.v2021.i1.269-315.
URL https://doi.org/10.46586/tosc.v2021.i1.269-315

[24] R. Ankele, S. Kölbl, Mind the gap - A closer look at the security of block ciphers against differential cryptanalysis, in: C. Cid, M. J. Jacobson Jr: (Eds.), SAC 2018, Vol. 11349 of LNCS, Springer, Heidelberg, 2019, pp. 163–190. doi:10.1007/978-3-030-10970-7_8.

[25] M. Hall-Andersen, P. S. Vejre, Generating graphs packed with paths, IACR Trans. Symm. Cryptol. 2018 (3) (2018) 265–289. doi:10.13154/tosc.v2018.i3.265-289.

[26] E. Biham, O. Dunkelman, N. Keller, A related-key rectangle attack on the full KASUMI, in: B. K. Roy (Ed.), ASIACRYPT 2005, Vol. 3788 of LNCS, Springer, Heidelberg, 2005, pp. 443–461. doi:10.1007/11593447_24.

[27] K. Jeong, C. Lee, J. Sung, S. Hong, J. Lim, Related-key amplified boomerang attacks on the full-round Eagle-64 and Eagle-128, in: J. Pieprzyk, H. Ghodosi, E. Dawson (Eds.), ACISP 07, Vol. 4586 of LNCS, Springer, Heidelberg, 2007, pp. 143–157.

[28] A. Biryukov, D. Khovratovich, Related-key cryptanalysis of the full AES-192 and AES-256, in: M. Matsui (Ed.), ASIACRYPT 2009, Vol. 5912 of LNCS, Springer, Heidelberg, 2009, pp. 1–18. doi:10.1007/978-3-642-10366-7_1.

[29] O. Dunkelman, N. Keller, A. Shamir, A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3G telephony, in:

T. Rabin (Ed.), CRYPTO 2010, Vol. 6223 of LNCS, Springer, Heidelberg, 2010, pp. 393–410. doi:10.1007/978-3-642-14623-7_21.

[30] T. Ashur, O. Dunkelman, A practical related-key boomerang attack for the full MMB block cipher, in: M. Abdalla, C. Nita-Rotaru, R. Dahab (Eds.), CANS 13, Vol. 8257 of LNCS, Springer, Heidelberg, 2013, pp. 271–290. doi:10.1007/978-3-319-02937-5_15.

[31] J. Lu, W.-S. Yap, Y. Wei, Weak keys of the full MISTY1 block cipher for related-key differential cryptanalysis, in: E. Dawson (Ed.), CT-RSA 2013, Vol. 7779 of LNCS, Springer, Heidelberg, 2013, pp. 389–404. doi:10.1007/978-3-642-36095-4_25.

[32] Y. Sasaki, Related-key boomerang attacks on full ANU lightweight block cipher, in: B. Preneel, F. Vercauteren (Eds.), ACNS 18, Vol. 10892 of LNCS, Springer, Heidelberg, 2018, pp. 421–439. doi:10.1007/978-3-319-93387-0_22.

[33] G. Tsudik, E. Van Herreweghen, On simple and secure key distribution, in: D. E. Denning, R. Pyle, R. Ganesan, R. S. Sandhu, V. Ashby (Eds.), ACM CCS 93, ACM Press, 1993, pp. 49–57. doi:10.1145/168588.168594.

[34] J. Kelsey, B. Schneier, D. Wagner, Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA, in: Y. Han, T. Okamoto, S. Qing (Eds.), ICICS 97, Vol. 1334 of LNCS, Springer, Heidelberg, 1997, pp. 233–246.

[35] E. Biham, O. Dunkelman, N. Keller, A unified approach to related-key attacks, in: K. Nyberg (Ed.), FSE 2008, Vol. 5086 of LNCS, Springer, Heidelberg, 2008, pp. 73–96. doi:10.1007/978-3-540-71039-4_5.

[36] D. Wagner, The boomerang attack, in: L. R. Knudsen (Ed.), FSE'99, Vol. 1636 of LNCS, Springer, Heidelberg, 1999, pp. 156–170. doi:10.1007/3-540-48519-8_12.

[37] J. Kelsey, T. Kohno, B. Schneier, Amplified boomerang attacks against reduced-round MARS and Serpent, in: B. Schneier (Ed.), FSE 2000, Vol. 1978 of LNCS, Springer, Heidelberg, 2001, pp. 75–93. doi:10.1007/3-540-44706-7_6.

[38] E. Biham, O. Dunkelman, N. Keller, New results on boomerang and rectangle attacks, in: J. Daemen, V. Rijmen (Eds.), FSE 2002, Vol. 2365 of LNCS, Springer, Heidelberg, 2002, pp. 1–16. doi:10.1007/3-540-45661-9_1.

[39] S. Murphy, The return of the cryptographic boomerang, IEEE Trans. Inf. Theory 57 (4) (2011) 2517–2521. doi:10.1109/TIT.2011.2111091. URL https://doi.org/10.1109/TIT.2011.2111091

[40] C. Cid, T. Huang, T. Peyrin, Y. Sasaki, L. Song, Boomerang connectivity table: A new cryptanalysis tool, in: J. B. Nielsen, V. Rijmen (Eds.), EUROCRYPT 2018, Part II, Vol. 10821 of LNCS, Springer, Heidelberg, 2018, pp. 683–714. doi:10.1007/978-3-319-78375-8_22.

[41] L. Song, X. Qin, L. Hu, Boomerang connectivity table revisited, IACR Trans. Symm. Cryptol. 2019 (1) (2019) 118–141. doi:10.13154/tosc.v2019.i1.118-141.

[42] A. Biryukov, I. Nikolic, Complementing Feistel ciphers, in: S. Moriai (Ed.), FSE 2013, Vol. 8424 of LNCS, Springer, Heidelberg, 2014, pp. 3–18. doi:10.1007/978-3-662-43933-3_1.

[43] K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita, T. Shirai, Piccolo: An ultra-lightweight blockcipher, in: B. Preneel, T. Takagi (Eds.), CHES 2011, Vol. 6917 of LNCS, Springer, Heidelberg, 2011, pp. 342–357. doi:10.1007/978-3-642-23951-9_23.

[44] B. Zhao, X. Dong, W. Meier, K. Jia, G. Wang, Generalized related-key rectangle attacks on block ciphers with linear key schedule: applications to SKINNY and GIFT, Des. Codes Cryptogr. 88 (6) (2020) 1103–1126. doi:10.1007/s10623-020-00730-1.
URL https://doi.org/10.1007/s10623-020-00730-1

[45] R. Zong, X. Dong, H. Chen, Y. Luo, S. Wang, Z. Li, Towards key-recovery-attack friendly distinguishers: Application to GIFT-128, IACR Trans. Symmetric Cryptol. 2021 (1) (2021) 156–184. doi:10.46586/tosc.v2021.i1.156-184.
URL https://doi.org/10.46586/tosc.v2021.i1.156-184

[46] A. A. Selçuk, On probability of success in linear and differential cryptanalysis, Journal of Cryptology 21 (1) (2008) 131–147. doi:10.1007/s00145-007-9013-7.