IEICE TRANSACTIONS

on Fundamentals of Electronics, Communications and Computer Sciences

DOI:10.1587/transfun.2024EAP1001

Publicized:2024/07/12

This advance publication article will be replaced by the finalized version after proofreading.

A PUBLICATION OF THE ENGINEERING SCIENCES SOCIETY

The Institute of Electronics, Information and Communication Engineers

Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN

PAPER

New Distinguishing Attacks on Round-Reduced Sparkle384 and Sparkle512 Permutations

Donghoon CHANG[†,††,†††a)], ***Nonmember***, **Deukjo HONG**[††††b)], ***Member***, ***and*** **Jinkeon KANG**[†c)], ***Nonmember***

SUMMARY The Sparkle permutation family is used as an underlying building block of the authenticated encryption scheme Schwaemm, and the hash function Esch which are a part of one of finalists in the National Institute of Standards and Technology (NIST) lightweight cryptography standardization process. In this paper, we present distinguishing attacks on 6-round Sparkle384 and 7-round Sparkle512. We used divide-and-conquer approach and the fact that Sparkle permutations are keyless, as a different approach from designers' long trail strategy. Our attack on Sparkle384 requires much lower time complexity than existing best one; our attack on Sparkle512 is best in terms of the number of attacked rounds, as far as we know. However, our results do not controvert the security claim of Sparkle designers.

key words: *Sparkle384, Sparkle512, Distinguishing Attack*

1. Introduction

Sparkle, designed by Beierle et al. [1], is a family of ARX-based cryptographic permutations, including three members corresponding to three sizes: Sparkle256 for 256-bit block, Sparkle384 for 384-bit block, and Sparkle512 for 512-bit block. The Sparkle permutation family is used as an underlying building block of the authenticated encryption scheme Schwaemm and the hash function Esch which are a part of one of finalists in the National Institute of Standards and Technology (NIST) lightweight cryptography standardization process [11]. The Sparkle permutations have a round-based iterative structure. Variants of Schwaemm and Esch use 10-round Sparkle256, 11-round Sparkle384, and 12-round Sparkle512 as big instances, and 7-round Sparkle256, 7-round Sparkle384, and 8-round Sparkle512 as slim instances.

The designers [1], applied the long trail strategy (LTS) [3] to the design of Sparkle to get resistance against differential cryptanalysis [2] and linear cryptanalysis [7], and analyzed the security of Sparkle permutations from various perspectives. Especially, they showed that the maximum numbers of rounds, for which the security level of $b/2$ bits against differential and linear cryptanalysis is broken, where b is the block size, are 5 for Sparkle256, 6 for Sparkle384, and 6 for Sparkle512, respectively, and claimed that 10-round Sparkle256, 11-round Sparkle384, and 12-round Sparkle512 have no distinguishers with both time and data lower than $2^{b/2}$. Additionally, they presented birthday-differential state-recovery attacks on 4.5-round Schwaemm128-128 (without whitening) with 2^{96} time and 2^{96} memory, 4.5-round Schwaemm192-192 (without whitening) with 2^{128} time and 2^{128} memory, and 4.5-round Schwaemm256-256 (without whitening) with 2^{192} time and 2^{160} memory. Note that these state recovery attacks can be used to recover the key as well, because the attacker can reverse the recovered state up to the initial state containing the key.

Schrottenloher and Stevens [9] noted that these birthday-differential attacks could be used to construct distinguishing attacks on Sparkle256, Sparkle384, Sparkle512 with half more round extension and without any further extra cost. They also provided guess-and-determine distinguishing attacks on 4-round Sparkle256 with negligible time and negligible memory, 4-round Sparkle384 with negligible time and negligible memory, 5-round Sparkle512 with time less than 2^{32} and negligible memory.

Our Contribution. There are few analysis results for Sparkle except [1] and [9]. We present new divide-and-conquer distinguishing attacks on 6-round Sparkle384 and 7-round Sparkle512, which fix specific forms of input and output differences, and find right pairs satisfying the differences. These attacks are devised based on the fact that Sparkle permutations are keyless. The time complexities of the distinguishing attacks on 6-round Sparkle384 and 7-round Sparkle512 are $2^{65.1}$ and $2^{191.4}$, far less than 2^{192} and 2^{256}, which are the $b/2$-bit security level for 384-bit and 512-bit block sizes, respectively.

The generic attacks corresponding to them are regarded as ones to find right pairs satisfying the same differences for random permutations with 384-bit or 512-bit blocks. We show that the generic attacks require 2^{257} queries for both 384-bit and 512-bit block sizes. It implies that our attacks are valid because they are much more efficient than generic ones.

Table 1 provides summary of existing distinguishing attacks on Sparkle permutations. Our attack on Sparkle384 works for the same number of rounds as the linear one in [1], but the time complexity of ours is much lower. Our attack on Sparkle512 works for more rounds than any other ones, as far as we know. However, our results do not controvert the security claim of Sparkle designers.

[†]National Institute of Standards and Technology, Gaithersburg, Maryland, USA
[††]Strativia, Largo, Maryland, USA
[†††]Department of Computer Science, Indraprastha Institute of Information Technology Delhi(IIIT-Delhi), Delhi, India
[††††]Smart Grid Research Center, Jeonbuk National University
a) E-mail: donghoon.chang@nist.gov
b) E-mail: deukjo.hong@jbnu.ac.kr(Correspongding author)
c) E-mail: jinkeon.kang@nist.gov

Copyright © 200x The Institute of Electronics, Information and Communication Engineers

Table 1 Summary of distinguishing attacks on Sparkle$n$ permutations, where 'T' and 'M' mean time and memory complexities, respectively.

n	Attack Type	Rounds	Complexity T	Complexity M	Ref.
256	Guess-and-Determine	4 / 10	negl.	negl.	[9]
	Linear	5 / 10	2^{114}	negl.	[1]
	Birthday-Differential	5 / 10	2^{96}	2^{96}	[1]
384	Guess-and-Determine	4 / 11	negl.	negl.	[9]
	Birthday-Differential	5 / 11	2^{128}	2^{128}	[1]
	Linear	6 / 11	2^{178}	negl.	[1]
	Divide-and-Conquer	6 / 11	$2^{65.1}$	2^{64}	Sec. 3.2
512	Guess-and-Determine	5 / 12	$< 2^{32}$	negl.	[9]
	Birthday-Differential	5 / 12	2^{192}	2^{160}	[1]
	Linear	6 / 12	2^{212}	negl.	[1]
	Divide-and-Conquer	7 / 12	$2^{191.4}$	2^{64}	Sec. 3.3

Organization. This paper is organized as follows. Section 2 describes Sparkle384 and Sparkle512 permutations in addition to some definitions and notations. Section 3 presents our distinguishing attacks on 6-round Sparkle384 and 7-round Sparkle512. Section 4 presents generic attacks to find right pairs for random permutations. Finally, in Section 5, we conclude the paper.

2. Preliminaries

2.1 Definitions and Notations

Let x and y be bitstrings of the same length. Bitwise XOR of x and y is denoted by $x \oplus y$. A nm-bit string x can be regarded as a length-n vector $(x_0, x_1, ..., x_{n-1})$ of m-bit strings or a length-m vector $(x_0, x_1, ..., x_{m-1})$ of n-bit strings. $\{0,1\}^n$ is the set of all n-bit strings. We often regard $\{0,1\}^n$ as a n-dimensional vector space over $GF(2)$.

2.2 Sparkle384 Permutation

First of all, we describe the Alzette operation. It is the only nonlinear operation in Sparkle, and is a 64-bit nonlinear ARX-based permutation. When it uses a 32-bit constant c, it is denoted as A_c. The 64-bit input z to A_c is split into two 32-bit words z_L and z_R. That is, $z = (z_L, z_R)$. Then, $x = A_c(z)$ is computed as follows:

1 : $(u, v) \leftarrow (z_L, z_R)$	8 : $u \leftarrow u + v$
2 : $u \leftarrow u + (v \ggg 31)$	9 : $v \leftarrow v \oplus (u \ggg 31)$
3 : $v \leftarrow v \oplus (u \ggg 24)$	10 : $u \leftarrow u \oplus c$
4 : $u \leftarrow u \oplus c$	11 : $u \leftarrow u + (v \ggg 24)$
5 : $u \leftarrow u + (v \ggg 17)$	12 : $v \leftarrow v \oplus (u \ggg 16)$
6 : $v \leftarrow v \oplus (u \ggg 17)$	13 : $u \leftarrow u \oplus c$
7 : $u \leftarrow u \oplus c$	14 : $x \leftarrow (u, v)$

σ is a simple linear permutation on $\{0,1\}^{64}$, used in Sparkle. The 64-bit input t to σ is split into four 16-bit words t_0, t_1, t_2, and t_3. Then, $\sigma(t) = \sigma(t_0, t_1, t_2, t_3)$ is defined by

$$\sigma(t) = (t_1, t_0 \oplus t_1, t_3, t_2 \oplus t_3).$$

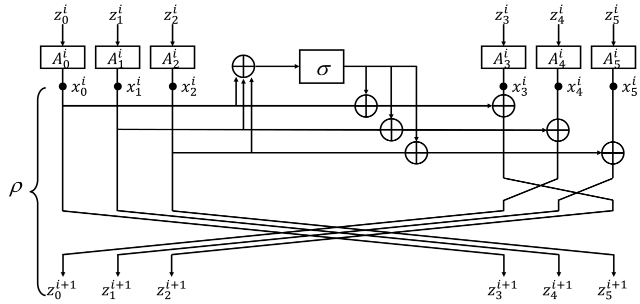


Fig. 1 The i-th round function Round i of Sparkle384

Sparkle384 is a permutation on $\{0,1\}^{384}$ and has a round iterative structure. Sparkle384$_r$ means that it consists of r round functions. The 384-bit input is split into six 64-bit words $z_0^0, z_1^0, ..., z_5^0$. As depicted in Fig. 1, for $0 \le i < r$, the i-th round function Round i takes the input words $z_0^i, z_1^i, ..., z_5^i$ and produces the output words $z_0^{i+1}, z_1^{i+1}, ...z_5^{i+1}$. The round function consists of three layers π, θ, and ρ.

In Round i, the π layer adds a 32-bit round constant and a 32-bit round counter value i to z_0^i and z_1^i, respectively. For simplicity, we omit π in the description of the round function because it has no impact on explaining our results in this paper. So, the input words of the θ layer are still represented as $z_0^i, z_1^i, ..., z_5^i$.

The θ layer consists of six Alzette operations. Instead of A_c, we use the notation of A_j^i, which means the j-th Alzette operation in Round i, because the position of each Alzette operation is given more significance than the constant values used within the operation. So, the θ layer is described as $x_j^i \leftarrow A_j^i(z_j^i)$ for $0 \le j < 6$.

The ρ layer linearly transforms $(x_0^i, ..., x_5^i)$ to $(z_0^{i+1}, ..., z_5^{i+1})$ as follows:

$$t^i \leftarrow x_0^i \oplus x_1^i \oplus x_2^i;$$
$$z_{j-1 \bmod 3}^{i+1} \leftarrow \sigma(t^i) \oplus x_j^i \oplus x_{j+3}^i \text{ for } 0 \le j < 3;$$
$$z_{j+3}^{i+1} \leftarrow x_j^i \text{ for } 0 \le j < 3.$$

See [1] for more details.

2.3 Sparkle512 Permutation

Sparkle512 is a permutation on $\{0,1\}^{512}$ and a round iterative structure. It has a very similar structure to that of Sparkle384, and uses the same Alzette and σ operations. Sparkle512$_r$ means that it consists of r round functions. The 512-bit input is split into eight 64-bit words $z_0^0, z_1^0, ..., z_7^0$. As depicted in Fig. 2, for $0 \le i < r$, the i-th round function Round i takes the input words $z_0^i, z_1^i, ..., z_7^i$ and produces the output words $z_0^{i+1}, z_1^{i+1}, ...z_7^{i+1}$.

Similarly to Sparkle384, the round function uses three layers π, θ, and ρ. We omit π, again. The θ layer uses the 64-bit nonlinear ARX-box Alzette A_j^i as $x_j^i \leftarrow A_j^i(z_j^i)$ for $0 \le j < 8$. The ρ layer linearly transforms $(x_0^i, ..., x_7^i)$ to $(z_0^{i+1}, ..., z_7^{i+1})$ is computed as follows:

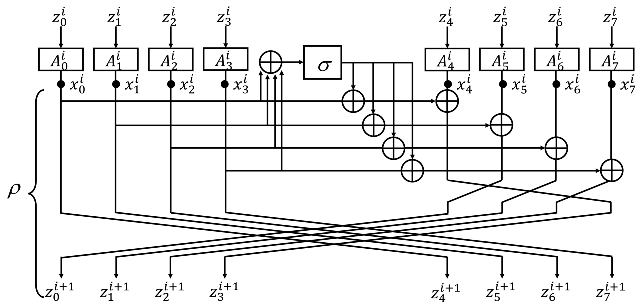


Fig. 2 The i-th round function Round i of Sparkle512

$$t^i \leftarrow x_0^i \oplus x_1^i \oplus x_2^i \oplus x_3^i;$$
$$z_{j-1 \bmod 4}^{i+1} \leftarrow \sigma(t^i) \oplus x_j^i \oplus x_{j+4}^i \text{ for } 0 \le j < 4;$$
$$z_{j+4}^{i+1} \leftarrow x_j^i \text{ for } 0 \le j < 4.$$

See [1] for more details.

3. Finding Right Pairs for Sparkle permutations

In Section 3.1, we give two kinds of probabilities for differential property of Alzette. Sections 3.2 and 3.3 present how to find right pairs for specific forms of input and output differences of Sparkle384_6 and Sparkle512_7, respectively. In Section 3.4, we provide total complexities of our right-pair-finding methods for Sparkle384_6 and Sparkle512_7, by using the probabilities explained in Section 3.1.

3.1 Differential Properties of Alzette

We define p and q for Alzette operation as follows.

- Let c be a 32-bit constant, and let $n(\Delta, \nabla) = \#\{z \mid A_c(z) \oplus A_c(z \oplus \Delta) = \nabla\}$. The value of $\mathcal{D}(\Delta, \nabla)$ is defined 0 if $n(\Delta, \nabla) = 0$, and 1 if $n(\Delta, \nabla) \neq 0$. Then, p is defined as

$$p = \frac{1}{(2^{64}-1)^2} \sum_{\Delta \neq 0, \nabla \neq 0} \mathcal{D}(\Delta, \nabla).$$

- Let c and c' be distinct 32-bit constants, and let k and k' be 64-bit values. Let $m(\Delta, \nabla) = \#\{z \mid A_{c'}(A_c(z) \oplus k) \oplus A_{c'}(A_c(z \oplus \Delta) \oplus k') = \nabla\}$. The value of $\mathcal{T}$ is defined 0 if $m(\Delta, \nabla) = 0$, and 1 if $m(\Delta, \nabla) \neq 0$. Then q is defined as

$$q = \frac{1}{(2^{64}-1)^2} \sum_{\Delta \neq 0, \nabla \neq 0} \mathcal{T}(\Delta, \nabla).$$

Here, we assume that c and c' are the constants with an appropriate number of '0' bits and '1' bits, and that the possibility of $k = k'$ is negligible. p is the ratio of nonzero entries in the difference distribution table of A_c excluding the cases of $\Delta = 0$ or $\nabla = 0$.

The addition modulo 2^{32} makes a strong propagation of difference from least significant bit to most significant bit, adopting a proper bitwise rotation delivers the effect of such propagation to least significant bits, and Alzette is a 64-bit-block nonlinear permutation and designed to alternate bitwise rotation, addition modulo 2^{32}, XOR, and constant addition operations for 4 rounds. Due to these factors, we anticipate that the difference distribution of Alzette will be relatively uniform.

Several analysis results for Alzette have been published and known [4], [6], [8], [10], [12], but it is computationally difficult to compute p and q, because the exact computation of p requires $O(2^{128})$ time and the exact computation of q requires $O(2^{256})$ time. Alternatively, we have considered the experiments on Alzette. In each trial of the experiments for p, we randomly chose two nonzero values as Δ and ∇ and tested whether $\mathcal{D}(\Delta, \nabla) = 1$. In each trial of the experiments for q, we randomly chose two nonzero values as Δ and ∇ and two different values as k and k' and tested whether $\mathcal{T}(\Delta, \nabla) = 1$. However, Each of the tests still remains computationally demanding on a PC because it requires $O(2^{64})$ Alzette operations. So, we performed experiments on reduced variants of Alzette with input sizes of 16, 24, 32, and 40 bits, and with reasonable rotation amounts and constants.

For example, The rotation amounts (31, 17, 24, 16) in Alzette were replaced with (7, 5, 6, 4) in 16-bit-block variant, (11, 7, 9, 6) in 24-bit-block variant, (15, 9, 12, 8) in 32-bit-block variant, and (19, 11, 15, 10) in 40-bit-block variant, respectively. For each variant, the number of the success cases where $\mathcal{D}(\Delta, \nabla) = 1$ was counted, and the number of the success cases where $\mathcal{T}(\Delta, \nabla) = 1$ was counted. The 16-bit version was tested 100,000 times, the 24-bit version was tested 10,000 times, the 32-bit version was tested 1,000 times, and the 40-bit version was tested 100 times. Based on those experiments, we conjecture $p \approx 0.36$ and $q \approx 0.62$.

Table 2 summarizes the success ratios $\tilde{p}$ and $\tilde{q}$ in our experiments, which imply that our conjecture is reasonable. The details and source code used in the experiments can be found in the Github repository [5].

Table 2 Success ratios in the experiments for reduced Alzette with the block size b = 16, 24, 32, and 40.

b	$\tilde{p}$	$\tilde{q}$
16	37,197/100,000	63,281/100,000
24	3,664/10,000	6,252/10,000
32	353/1,000	610/1,000
40	39/100	62/100

3.2 Finding A Right Pair for input and output differences of Sparkle384_6

We consider the input difference Δ_I and the output difference Δ_O as follows.

$$\begin{cases} \Delta_I = (0, 0, 0, \alpha, 0, 0); \\ \Delta_O = (0, \varepsilon, \varepsilon, \varepsilon, 0, \varepsilon), \end{cases} \tag{1}$$

where α and ε are any 64-bit nonzero values. Let Δz_i^j and

Δx_i^j be differences on z_i^j and x_i^j, respectively. Through the following steps, we explain how to find a right pair satisfying (1) for Sparkle384$_6$.

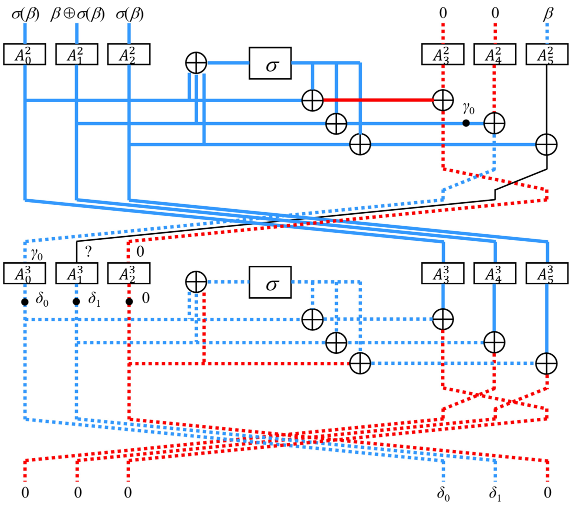


Fig. 3 Situation in Rounds 2 and 3 after Step 1 in finding a right pair for the differential of Sparkle384$_6$

Step 1: We set the difference of three left input words in Round 2 as follows.

$$(\Delta z_0^2, \Delta z_1^2, \Delta z_2^2) = (\sigma(\beta), \beta \oplus \sigma(\beta), \sigma(\beta)). \tag{2}$$

Let $t^j = x_0^j \oplus x_1^j \oplus x_2^j$. We search for a pair for (z_0^2, z_1^2, z_2^2) satisfies

$$\Delta x_0^2 \oplus \sigma(\Delta t^2) = 0, \tag{3}$$

where (3) holds with the probability of 2^{-64}. The found pair determines the values and differences of (z_3^3, z_4^3, z_5^3), and also determines the differences $\Delta z_0^3 = \gamma_0$ and $\Delta z_2^3 = 0$. Moreover, $\Delta z_2^3 = 0$ implies $\Delta x_2^3 = \Delta z_5^4 = 0$. So, we get the differences Δx_3^3, Δx_4^3, and Δx_5^3 by the computation with the pair satisfying (3). Then, letting $\Delta z_0^4 = \Delta z_1^4 = \Delta z_2^4 = 0$, we have $\sigma(\Delta t^3) = \Delta x_5^3$, $\Delta x_0^3 = \Delta x_3^3 \oplus \Delta x_5^3$, and $\Delta x_1^3 = \Delta x_2^3 \oplus \Delta x_5^3$. Finally, we expect (4) holds with the probability of 2^{-64}.

$$\sigma(\Delta x_0^3 \oplus \Delta x_1^3) = \Delta x_5^3. \tag{4}$$

Therefore, we need 2^{128} pairs satisfying (2). Note that the only requirement for the difference β is 'nonzero'. For efficient collection of pairs, we consider the set $\mathcal{S}(X)$ with a 192-bit value X and $\mathcal{U} = \{0,1\}^{64}$, defined as

$$\mathcal{W} = \{(\sigma(a), a \oplus \sigma(a), \sigma(a)) \mid a \in \mathcal{U}\};$$
$$\mathcal{S}(X) = X \oplus \mathcal{W} = \{X \oplus w \mid w \in \mathcal{W}\}.$$

$\mathcal{S}(X)$ can derive $2^{64}(2^{64}-1)/2 \simeq 2^{127}$ pairs satisfying (2). Two distinct sets $\mathcal{S}(X_1)$ and $\mathcal{S}(X_2)$ are enough to get 2^{128} pairs. We expect one of the pairs to satisfy (3) and (4) on average.

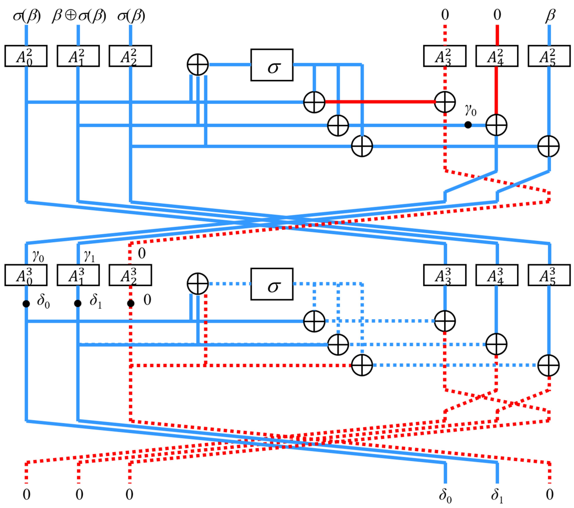


Fig. 4 Situation in Rounds 2 and 3 after Steps 2 and 3 in finding a right pair for the differential of Sparkle384$_6$

We estimate the complexity C_1 of Step 1, using Alzette operation time as the unit. For simplicity, we denote the cost of one Alzette operation by A. A_0^2, A_1^2, and A_2^2 (three Alzette operations) are applied to each element in $\mathcal{S}(X)$ and $\mathcal{S}(X')$. We count how many pairs out of 2^{128} satisfy (3). On average, 2^{64} pairs do. Then, we apply A_3^3, A_4^3, and A_5^3 (at most six Alzette operations) to each among those 2^{64} pairs, in order to check whether (4) holds. Therefore, C_1 is estimated as

$$C_1 = 2^{65} \cdot 3A + 2^{64} \cdot 6A = 2^{66} \cdot 3A.$$

Fig. 3 depicts the situation in Rounds 2 and 3 after Step 1. Bold red dotted lines mean that the values are undetermined but the differences are zero. Bold blue dotted lines mean that values are undetermined but the differences are determined and nonzero. Bold red solid lines mean that values are determined and the differences are zero. Bold blue solid lines mean that values are determined and differences are nonzero. Plain black lines mean that both values and differences are undetermined.

Let the determined differences Δz_0^3, Δx_0^3, and Δx_1^3 be γ_0, δ_0, and δ_1, respectively.

Step 2: For A_0^3, we try all 2^{64} possible input pairs with the input difference γ_0 to find one satisfying the output difference δ_0. If we fail to find it, we go back to Step 1. The complexity C_2 of Step 2 is estimated as $C_2 = 2^{65}A$.

Step 3: The pair found in Step 1 determines the values of $x_2^2 \oplus \sigma(t^2)$. Let k and k' be the values. We try all 2^{64} possible pairs $(z_5^2, z_5^{2\prime})$ with $z_5^2 \oplus z_5^{2\prime} = \beta$ to find one satisfying

$$A_1^3(A_5^2(z_5^2) \oplus k) \oplus A_1^3(A_5^2(z_5^{2\prime}) \oplus k') = \delta_1.$$

If we fail to find it, we go to Step 1. The complexity C_3 of Step 3 is estimated as $C_3 = 2^{66}A$. Fig. 4 shows the situation

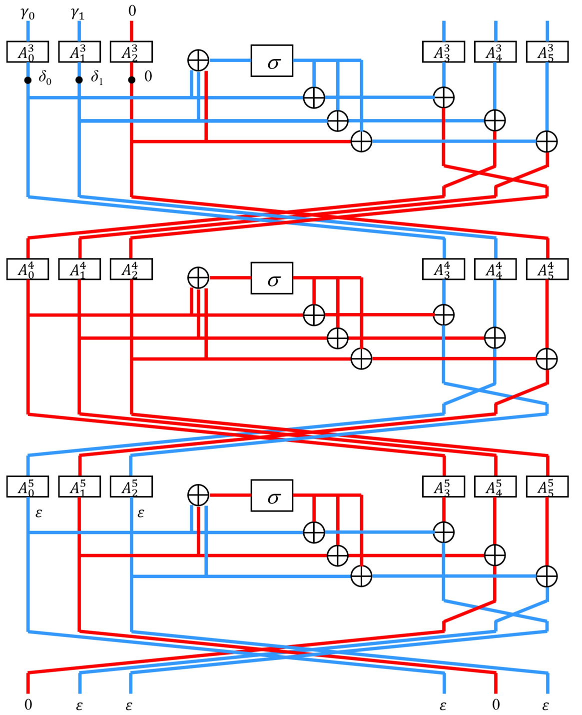


Fig. 5 Forward propagation from Round 3 to Round 5 after Step 4 in finding a right pair for the differential of Sparkle384_6

after Steps 2 and 3 succeed.

Step 4: If the previous steps are successful, the values of the input words in Round 3, excluding z_2^3, are determined and fixed. So, the only undetermined word $x_2^3 = A_2^3(z_2^3)$ is related to the other undetermined words in whole rounds. We use 64 degrees of freedom of x_2^3 to make the differences of x_0^5 and x_2^5 matched, i.e., $\Delta x_0^5 = \Delta x_2^5$. It requires 2^{64} trials on average. From previous steps, we have $(x_i^3, x_i^{3\prime})$ for $i =$ 0, 1, 3, 4, and 5. For each candidate for x_2^3, we can check whether $\Delta x_0^5 = \Delta x_2^5$ by computing the followings:

$$
\begin{aligned}
&t^3 = x_0^3 \oplus x_1^3 \oplus x_2^3;\\
&x_i^4 = A_i^4(x_{i+1 \bmod 3}^3 \oplus \sigma(t^3)) \text{ for } i \in \{0,1,2\};\\
&t^4 = x_0^4 \oplus x_1^4 \oplus x_2^4;\\
&x_0^5 = A_0^5(A_4^4(x_1^3) \oplus x_1^4 \oplus \sigma(t^4));\\
&x_0^{5\prime} = A_0^5(A_4^4(x_1^{3\prime}) \oplus x_1^4 \oplus \sigma(t^4));\\
&x_2^5 = A_2^5(A_3^4(x_0^3) \oplus x_0^4 \oplus \sigma(t^4));\\
&x_2^{5\prime} = A_2^5(A_3^4(x_0^{3\prime}) \oplus x_0^4 \oplus \sigma(t^4)).
\end{aligned}
$$

The complexity C_4 of Step 4 is estimated as $C_4 = 2^{64} \cdot 11A$.

If we find such a value of x_2^3, the propagation to the other undetermined words is easily computed – the forward propagation from $(z_0^3, \ldots, z_5^3)$ to the output difference Δ_O as depicted in Fig. 5 and the backward propagation from $(z_0^2, \ldots, z_5^2)$ to the input difference Δ_I as depicted in Fig. 6. By letting $\Delta x_0^5 = \varepsilon$ and $\Delta z_3^0 = \alpha$, it is easy to see that the found pair satisfies (1).

To verify the correctness of the steps from Step 1 to Step 4, we ran experiments on small-scale version, Sparkle48_6, with reduced variant of Alzette with input size of 8 bits. The rotation amounts used in 8-bit Alzette is (3, 1, 2, 2). The following is one example of inputs I and I' and outputs O and O' found through this experiment, where $\Delta I = I \oplus I'$ and $\Delta O = O \oplus O'$:

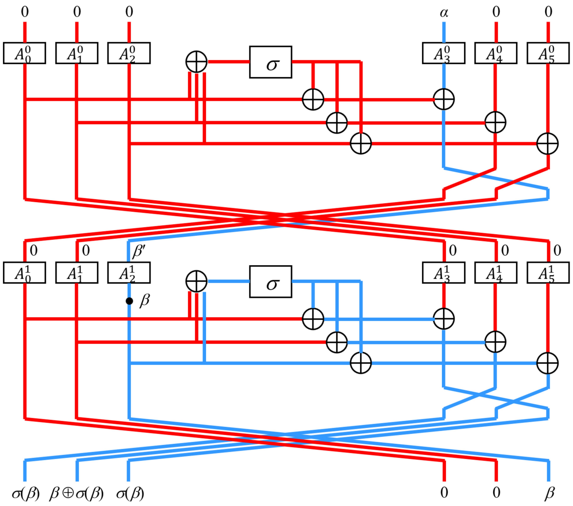


Fig. 6 Backward propagation from Round 1 to Round 0 after Step 4 in finding a right pair for the differential of Sparkle384_6

$$
\begin{aligned}
I &= (\texttt{0x65}, \texttt{0x12}, \texttt{0xed}, \texttt{0xc1}, \texttt{0x50}, \texttt{0xd7}),\\
I' &= (\texttt{0x65}, \texttt{0x12}, \texttt{0xed}, \texttt{0x75}, \texttt{0x50}, \texttt{0xd7}),\\
\Delta_I &= (\texttt{0x00}, \texttt{0x00}, \texttt{0x00}, \texttt{0xb4}, \texttt{0x00}, \texttt{0x00}),\\
O &= (\texttt{0xc6}, \texttt{0xba}, \texttt{0x19}, \texttt{0xdb}, \texttt{0x79}, \texttt{0x7c}),\\
O' &= (\texttt{0xc6}, \texttt{0x0d}, \texttt{0xae}, \texttt{0x6c}, \texttt{0x79}, \texttt{0xcb}), \text{ and}\\
\Delta_O &= (\texttt{0x00}, \texttt{0xb7}, \texttt{0xb7}, \texttt{0xb7}, \texttt{0x00}, \texttt{0xb7}).
\end{aligned}
$$

The details and source code used in the experiments can be found in the Github repository [5].

3.3 Finding A Right Pair for input and output differences of Sparkle512_7

We consider the input difference Δ_I and the output difference Δ_O as follows.

$$
\begin{cases}
\Delta_I = (0,0,0,0,\alpha,0,0,0);\\
\Delta_O = (\xi,0,\zeta,\eta,\zeta,0,0,\zeta),
\end{cases} \tag{5}
$$

where α, ζ, and η are 64-bit nonzero values, and ξ is a 64-bit value.

Through the following steps, we explain how to find a right pair satisfying (5) for Sparkle512_7.

Step 1: We set the difference of three left input words in Round 2 as follows.

$$
\begin{aligned}
&(\Delta z_0^2, \Delta z_1^2, \Delta z_2^2, \Delta z_3^2)\\
&\quad = (\sigma(\beta), \sigma(\beta), \beta \oplus \sigma(\beta), \sigma(\beta)).
\end{aligned} \tag{6}
$$

Let $t^j = x_0^j \oplus x_1^j \oplus x_2^j \oplus x_3^j$. We search for a pair for $(z_0^2, z_1^2, x_2^2, z_3^2)$ satisfying

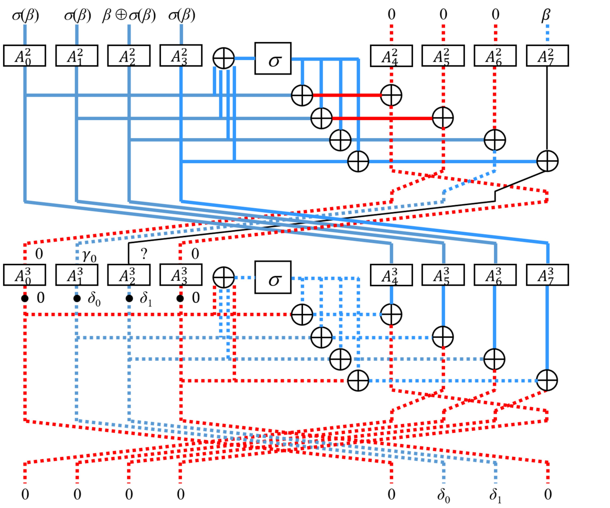


Fig. 7 Situation in Rounds 2 and 3 after Step 1 in finding a right pair for the differential of Sparkle512$_7$

$$(\Delta x^2_0 \oplus \sigma(\Delta t^2), \Delta x^2_1 \oplus \sigma(\Delta t^2)) = (0, 0), \quad (7)$$

where (7) holds with the probability of 2^{-128}. (7) implies $\Delta x^2_0 = \Delta x^2_1 = \sigma(\Delta t^2)$. The found pair determines the values and differences of $(z^3_4, \ldots, z^3_7)$, and also determines the differences $\Delta z^3_0 = \Delta z^3_3 = 0$ and $\Delta z^3_1 = \gamma_0$. Moreover, $\Delta z^3_0 = \Delta z^3_3 = 0$ implies $\Delta x^3_0 = \Delta z^4_4 = 0$ and $\Delta x^3_3 = \Delta z^4_7 = 0$. We get the differences Δx^3_4, Δx^3_5, Δx^3_6, and Δx^3_7 by the computation with the pair satisfying (7). Assuming $\Delta z^4_0 = \cdots = \Delta z^4_3 = 0$, we compute $\sigma(\Delta t^3) = \Delta x^3_4$, $\Delta x^3_1 = \Delta x^3_4 \oplus \Delta x^3_5$, and $\Delta x^3_2 = \Delta x^3_4 \oplus \Delta x^3_6$. Then, we expect that (8) holds with the probability of 2^{-64} and that (9) hold with the probability of 2^{-64}.

$$\Delta x^3_4 = \Delta x^3_7 \text{ and} \quad (8)$$
$$\sigma(\Delta x^3_1 \oplus \Delta x^3_2) = \Delta x^3_4. \quad (9)$$

Therefore, we need 2^{256} pairs satisfying (6). Note that the only requirement of the difference β is 'nonzero'. For efficient collection of pairs, we consider the set $\mathcal{S}(X)$ with a 256-bit value X and $\mathcal{U} = \{0, 1\}^{64}$, defined as

$$\mathcal{W} = \{(\sigma(a), \sigma(a), a \oplus \sigma(a), \sigma(a)) \mid a \in \mathcal{U}\};$$
$$\mathcal{S}(X) = X \oplus \mathcal{W} = \{X \oplus w \mid w \in \mathcal{W}\}.$$

$\mathcal{S}(X)$ can derive $2^{64}(2^{64} - 1)/2 \simeq 2^{127}$ pairs satisfying (6). 2^{129} distinct $\mathcal{S}(X)$sets are enough to get 2^{256} pairs. We expect one of the pairs to satisfy (7), (8), and (9) on average.

The complexity C_1 of Step 1 is estimated as follows. Step 1 uses 2^{129} $\mathcal{S}(X)$ sets. Each element in $\mathcal{S}(X)$ incurs eight Alzette operations (A^2_i for $0 \le i \le 3$ and A^3_j for $4 \le j \le 7$) and one σ operation. Additionally, we anticipate that among all tried pairs, 2^{128} will satisfy (7), and among the surviving pairs, we expect 2^{64} to fulfill (8), with one pair among the last survivors expected to meet (9). Therefore, C_1 is estimated as

$$C_1 = 2^{129+64} \cdot 4A + 2^{128} \cdot 4A + 2^{64} \cdot 4A$$
$$\simeq 2^{195} A.$$

Let the determined differences Δz^3_1, Δx^3_1, and Δx^3_2 be γ_0, δ_0, and δ_1, respectively.

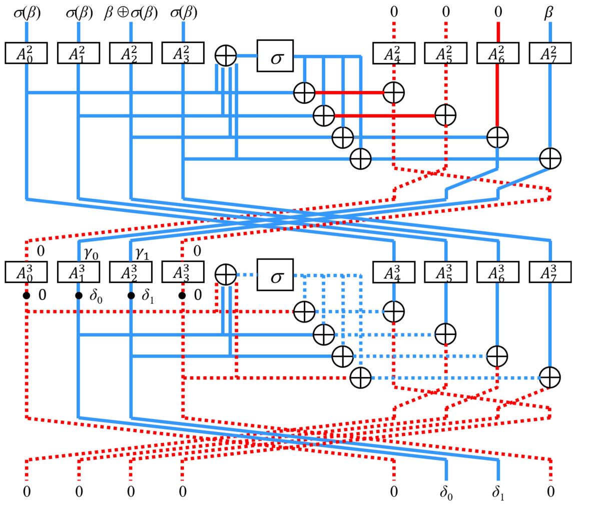


Fig. 8 Situation in Rounds 2 and 3 after Steps 2 and 3 in finding a right pair for the differential of Sparkle512$_7$

Step 2: For A^3_1, we try all 2^{64} possible input pairs with the input difference γ_0 to find one satisfying the output difference δ_0. If we fail to find it, we go back to Step 1. The complexity C_2 of Step 2 is estimated as $C_2 = 2^{65} A$.

Step 3: The pair found in Step 1 determines the values of $x^2_3 \oplus \sigma(t^2)$. Let k and k' be the values. We try all 2^{64} possible pairs $(z^2_7, z^{2\prime}_7)$ with $z^2_7 \oplus z^{2\prime}_7 = \beta$ to find one satisfying

$$A^3_2(A^2_7(z^2_7) \oplus k) \oplus A^3_2(A^2_7(z^{2\prime}_7) \oplus k') = \delta_1.$$

If we fail to find it, we go to Step 1. The complexity C_3 of Step 3 is estimated as $C_3 = 2^{66} A$. Fig. 8 shows the situation after Steps 2 and 3 succeed.

Step 4: If the previous steps are successful, the input word variables for Round 3, excluding z^3_0 and z^3_3, are determined. The only undetermined words $x^3_0 = A^3_0(z^3_0)$ and $x^3_3 = A^3_3(z^3_3)$ are related to the other undetermined words in whole rounds. We use 128 degrees of freedom of (x^3_0, x^3_3) to make $\Delta x^5_0 = \Delta x^5_1$ and $\Delta x^6_0 = \Delta x^6_3$. It requires 2^{128} trials on average. From previous steps, we have $(x^3_i, x^{3\prime}_i)$ for $i = 1, 2, 4, 5, 6$, and 7. For each of (x^3_0, x^3_3), we check whether $\Delta x^5_0 = \Delta x^5_1$ by computing the followings:

$$t^3 = x^3_0 \oplus x^3_1 \oplus x^3_2 \oplus x^3_3;$$
$$x^4_i = A^4_i(x^3_{i+1 \bmod 4} \oplus \sigma(t^3)) \text{ for } 0 \le i \le 3;$$
$$t^4 = x^4_0 \oplus x^4_1 \oplus x^4_2 \oplus x^4_3;$$
$$x^5_0 = A^5_0(A^4_5(x^3_1) \oplus x^4_1 \oplus \sigma(t^4));$$
$$x^{5\prime}_0 = A^5_0(A^4_5(x^{3\prime}_1) \oplus x^4_1 \oplus \sigma(t^4));$$
$$x^5_1 = A^5_1(A^4_6(x^3_2) \oplus x^4_2 \oplus \sigma(t^4));$$
$$x^{5\prime}_1 = A^5_1(A^4_6(x^{3\prime}_2) \oplus x^4_2 \oplus \sigma(t^4)).$$

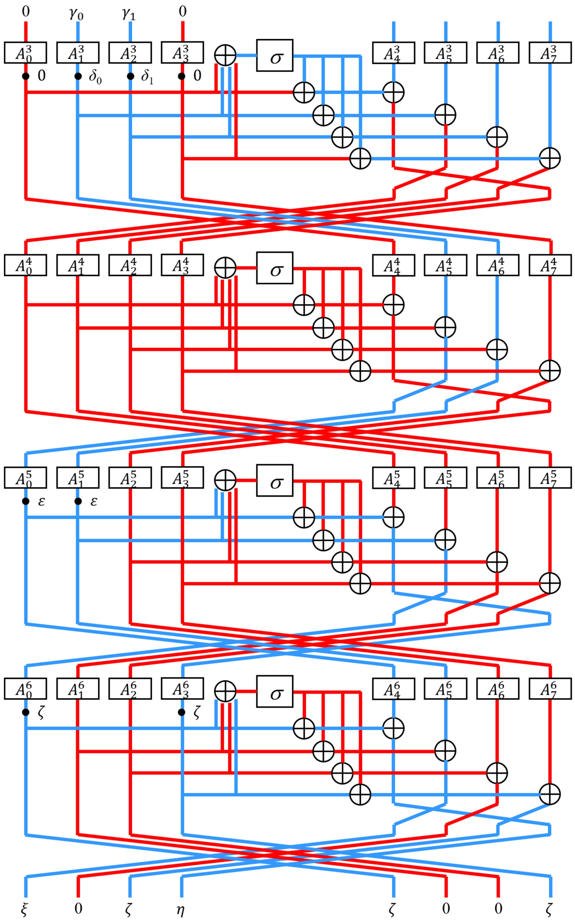


Fig. 9 Forward propagation from Round 3 to Round 6 after Step 4 in finding a right pair for the differential of Sparkle512_7

On average, we expect 2^{64} values of (x_0^3, x_3^3) survive. Since $\Delta x_2^5 = \Delta x_3^5 = 0$, $\Delta x_0^5 = \Delta x_1^5$ implies $\Delta t^5 = 0$. For the surviving values of (x_0^3, x_3^3), we check whether $\Delta x_0^6 = \Delta x_3^6$ by computing the followings:

$$
\begin{aligned}
x_2^5 &= A_2^5(A_4^4(x_0^3) \oplus x_0^4 \oplus \sigma(t^4));\\
x_3^5 &= A_3^5(A_7^4(x_3^3) \oplus x_3^4 \oplus \sigma(t^4));\\
t^5 &= x_0^5 \oplus x_1^5 \oplus x_2^5 \oplus x_3^5;\\
x_0^6 &= A_0^6(A_5^5(x_1^4) \oplus x_1^5 \oplus \sigma(t^5));\\
x_0^{6\prime} &= A_0^6(A_5^5(x_1^4) \oplus x_1^{5\prime} \oplus \sigma(t^5));\\
x_3^6 &= A_3^6(A_4^5(x_0^4) \oplus x_0^5 \oplus \sigma(t^5));\\
x_3^{6\prime} &= A_3^6(A_4^5(x_0^4) \oplus x_0^{5\prime} \oplus \sigma(t^5)).
\end{aligned}
$$

Therefore, the complexity C_4 of Step 4 is estimated as $C_4 = 2^{128} \cdot 12A + 2^{64} \cdot 12A \approx 2^{128} \cdot 12A$.

If we find such a value of (x_0^3, x_3^3), the propagation to the other undetermined words is easily computed – the forward propagation from $(z_0^3, \ldots, z_7^3)$ to the output difference Δ_O as depicted in Fig. 9 and the backward propagation from $(z_0^2, \ldots, z_7^2)$ to the input difference Δ_I as depicted in Fig. 10. By letting $\Delta x_0^5 = \varepsilon$, $\Delta x_0^6 = \zeta$, $\Delta x_0^6 \oplus \Delta x_4^6 = \eta$, $\Delta x_5^6 = \xi$ and $\Delta z_3^0 = \alpha$, it is easy to see that the found pair satisfies (1).

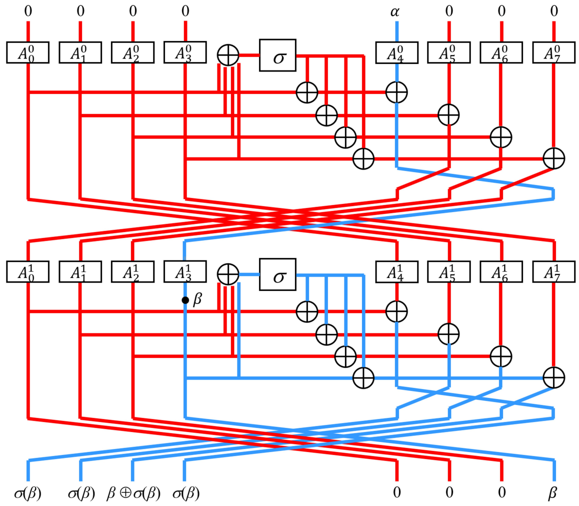


Fig. 10 Backward propagation from Round 1 to Round 0 after Step 4 in finding a right pair for the differential of Sparkle512_7

3.4 Complexity

The right-pair-finding methods explained in both Sections 3.2 and 3.3 consist of Steps 1, 2, 3, and 4. The flow from Step 1 to Step 4 is depicted in Fig. 11. The success probabilities of Steps 2 and 3 are p and q, explained in Section 3.1. Therefore, the total complexity C is computed as

$$C = ((C_1 + C_2)p^{-1} + C_3)q^{-1} + C_4. \tag{10}$$

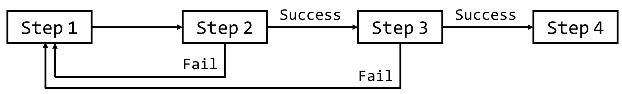


Fig. 11 Flow from Step 1 to Step 4 in finding a right pair for Sparkle permutations

Based on the conjecture and observation explained in Section 3.1, we let $p = 0.36$ and $q = 0.62$. Then, for Sparkle384_6, C is estimated as follows.

$$
\begin{aligned}
C &= ((2^{66} \cdot 3A + 2^{65}A)p^{-1} + 2^{66}A)q^{-1} + 2^{64} \cdot 11A\\
&= 2^{64}((14/0.36 + 4)/0.62 + 11)A\\
&= 2^{64+6.3}A = 2^{70.3}A.
\end{aligned}
$$

Since one Sparkle384_6 operation requires 36 Alzette operations, C is converted into $C \simeq 2^{65.1}$.

In the case of Sparkle512_7, we have $C \approx C_1 p^{-1} q^{-1}$ because C_1 is much larger than C_2, C_3, and C_4. By substituting $2^{195}A$, 0.36, and 0.62 to C_1, p, and q, respectively, C is estimated as $C \simeq 2^{197.2}A$. Since one Sparkle512_7 operation requires 56 Alzette operations, C is converted into $C \simeq 2^{191.4}$.

We computed the time complexities based on the observation through our experiments in Section 3.1, and recognize that the real values of p and q can be slightly different from the conjectured ones. So, we address that the time complexities should be regarded as $O(2^{65.1})$ and $O(2^{191.4})$ rather than $2^{65.1}$ and $2^{191.4}$. The attacks explained in Sections 3.2, 3.3, 4.1, and 4.2 require memory of 2^{64} because each of them uses 64-dimensional linear subspaces over $GF(2)$ to collect pairs.

4. Generic Attacks on Random Permutations

In this section, we describe two generic attacks corresponding to the right-pair-finding ones described in Sections 3.2 and 3.3. One targets the random permutation with 384-bit block as the ideal version of $\mathsf{Sparkle384}_6$, and the other one targets the random permutation with 512-bit block as the ideal version of $\mathsf{Sparkle512}_7$. Although each generic attack was devised to require as few queries as possible, its query complexity is much more than the time complexity of the corresponding one. It implies that the methods in Section 3 are more efficient and work as valid distinguishing attacks.

4.1 Random Permutations on $\{0,1\}^{384}$

We define $\mathcal{P}_{384}$ as the set of all permutations on $\{0,1\}^{384}$. Let P is a permutation randomly chosen from $\mathcal{P}_{384}$. Assuming that we have access to P and P^{-1} oracles, we describe how to find a right pair satisfying (1) for P. We consider the input difference Δ_I and output difference Δ_O in (1).

For any two different 384-bit values X and X', the probability that $P(X) \oplus P(X') = \Delta_O$ is 2^{-320}, whatever $X \oplus X'$ is. So, we need 2^{320} input pairs satisfying Δ_I to expect a right pair. We can efficiently collect them based on the fact that the only requirements for α in Δ_I and ε in Δ_O are 'nonzero'.

The set $\mathcal{A}(X)$ is defined with a 384-bit value X and $\mathcal{U} = \{0,1\}^{64}$ as follows.

$$\mathcal{W} = \{(0,0,0,a,0,0) \mid a \in \mathcal{U}\};$$
$$\mathcal{A}(X) = X \oplus \mathcal{W} = \{X \oplus w \mid w \in \mathcal{W}\}.$$

$\mathcal{W}$ is a 64-dimensional linear subspace of $\mathcal{V} = \{0,1\}^{384}$ and $\mathcal{A}(X)$ is a coset of $\mathcal{W}$. Namely, $\mathcal{A}(X) \in \mathcal{V}/\mathcal{W}$ and there are 2^{320} distinct $\mathcal{A}(X)$ sets in $\mathcal{V}/\mathcal{W}$. Each $\mathcal{A}(X)$ has 2^{64} elements and any two different elements $X \oplus w$ and $X \oplus w'$ in $\mathcal{A}(X)$ satisfies Δ_I: $(X \oplus w) \oplus (X \oplus w') = w \oplus w' \in \mathcal{W}$. It is trivial that $X \oplus w \in \mathcal{A}(X)$ and $X' \oplus w' \in \mathcal{A}(X')$ does not satisfy Δ_I if $\mathcal{A}(X) \neq \mathcal{A}(X')$. We can use $2^{127} (\approx 2^{64}(2^{64}-1)/2)$ pairs for each $\mathcal{A}(X)$. 2^{193} distinct $\mathcal{A}(X)$ sets are enough to collect 2^{320} pairs. Therefore, the complexity is $2^{193+64} = 2^{257}$ P-queries.

Likewise, for any two different 384-bit values Y and Y', the probability that $P^{-1}(Y) \oplus P^{-1}(Y') = \Delta_I$ is 2^{-320}. So, we need 2^{320} pairs, and collect them efficiently by using the set $\mathcal{B}(Y)$ defined with a 384-bit value Y and $\mathcal{U} = \{0,1\}^{64}$ as follows.

$$\mathcal{W} = \{(0,b,b,b,0,b) \mid b \in \mathcal{U}\};$$
$$\mathcal{B}(Y) = Y \oplus \mathcal{W} = \{Y \oplus w \mid w \in \mathcal{W}\}.$$

Each $\mathcal{B}(Y) \in \mathcal{V}/\mathcal{W}$ has 2^{64} elements and derives 2^{127} pairs satisfying the difference Δ_O. Therefore, we use 2^{193} distinct $\mathcal{B}(Y)$ sets to expect a right pair. The complexity is $2^{193+64} = 2^{257}$ P^{-1}-queries.

4.2 Random Permutations on $\{0,1\}^{512}$

We define $\mathcal{P}_{512}$ as the set of all permutations on $\{0,1\}^{512}$. Let P be a permutation randomly chosen from $\mathcal{P}_{512}$. We consider the input difference Δ_I and output difference Δ_O of (5). For any two different 512-bit values X and X', the probability that $P(X) \oplus P(X') = \Delta_O$ is 2^{-320}, whatever $X \oplus X'$ is. So, we need 2^{320} input pairs satisfying Δ_I to expect a right pair. We can efficiently collect them based on the fact that the only requirements for α, ζ, η, and ξ in Δ_I and Δ_O are nonzero.

The set $\mathcal{A}(X)$ is defined with a 512-bit value X and $\mathcal{U} = \{0,1\}^{64}$ as follows.

$$\mathcal{W} = \{(0,0,0,0,a,0,0,0) \mid a \in \mathcal{U}\};$$
$$\mathcal{A}(X) = X \oplus \mathcal{W} = \{X \oplus w \mid w \in \mathcal{W}\}.$$

$\mathcal{W}$ is a 64-dimensional linear subspace of $\mathcal{V} = \{0,1\}^{512}$, and there are 2^{488} distinct $\mathcal{A}(X)$ sets in $\mathcal{V}/\mathcal{W}$. Each $\mathcal{A}(X) \in \mathcal{V}/\mathcal{W}$ has 2^{64} elements and derives around 2^{127} pairs satisfying difference Δ_I. Therefore, we use 2^{193} distinct $\mathcal{A}(X)$ sets to expect a right pair. The complexity is $2^{193+64} = 2^{257}$ P-queries.

Likewise, for any two different 512-bit values Y and Y', the probability that $P^{-1}(Y) \oplus P^{-1}(Y') = \Delta_I$ is 2^{-448}. So, we need 2^{448} pairs, and collect them efficiently by using the set $\mathcal{B}(Y)$ defined with a 512-bit value Y and $\mathcal{U} = \{0,1\}^{64}$ as follows.

$$\mathcal{W} = \{(b,0,c,d,c,0,0,c) \mid b,c,d \in \mathcal{U}\};$$
$$\mathcal{B}(Y) = Y \oplus \mathcal{W} = \{Y \oplus w \mid w \in \mathcal{W}\}.$$

Each $\mathcal{B}(Y) \in \mathcal{V}/\mathcal{W}$ has 2^{192} elements and derives $2^{383} (\approx 2^{192}(2^{64}-1)(2^{64}-1)(2^{64}-1)/2)$ pairs satisfying the difference Δ_O. Therefore, we use 2^{65} distinct $\mathcal{B}(Y)$ sets to expect a right pair. The complexity is $2^{65+192} = 2^{257}$ P^{-1}-queries.

5. Conclusion

In this paper, we presented divide-and-conquer distinguishing attacks on 6-round Sparkle384 and 7-round Sparkle512. Our attacks were devised based on the fact that Sparkle permutations are keyless, differently from designers' approaches on design and analysis. Our attack on Sparkle384 requires much lower time complexity than the best existing attack, and our attack on Sparkle512 is best in terms of the number of attacked rounds, as far as we know. However, our results do not controvert the security claim of Sparkle designers.

Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1A2C1005946). We greatly appreciate

Meltem Sönmez Turan for providing valuable advices.

References

[1] Christof Beierle, Alex Biryukov, Luan Cardoso dos Santos, Johann Großschädl, Léo Perrin, Aleksei Udovenko, Vesselin Velichkov, and Qingju Wang. Schwaemm and Esch: Lightweight Authenticated Encryption and Hashing using the Sparkle Permutation Family (Version 1.2). Submission to the NIST Lightweight Cryptography Standardization Process, 2021. `https://csrc.nist.gov/Projects/lightweight-cryptography/finalists`.

[2] Eli Biham and Adi Shamir. *Differential Cryptanalysis of the Data Encryption Standard*. Springer, 1993.

[3] Daniel Dinu, Léo Perrin, Aleksei Udovenko, Vesselin Velichkov, Johann Großschädl, and Alex Biryukov. Design strategies for ARX with provable bounds: Sparx and LAX. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, volume 10031 of *Lecture Notes in Computer Science*, pages 484–513, 2016.

[4] Mingjiang Huang, Zhen Xu, and Liming Wang. On the probability and automatic search of rotational-xor cryptanalysis on ARX ciphers. *Comput. J.*, 65(12):3062–3080, 2022.

[5] Jinkeon Kang. Finding Right Pairs for Sparkle Permutations. GitHub repository. `https://github.com/JinkeonKang/Sparkle`.

[6] Yunwen Liu, Siwei Sun, and Chao Li. Rotational cryptanalysis from a differential-linear perspective - practical distinguishers for round-reduced friet, xoodoo, and alzette. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I*, volume 12696 of *Lecture Notes in Computer Science*, pages 741–770. Springer, 2021.

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

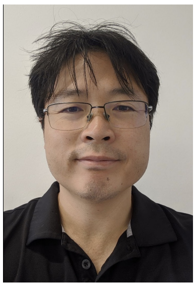
[8] Zhongfeng Niu, Siwei Sun, Yunwen Liu, and Chao Li. Rotational differential-linear distinguishers of ARX ciphers with arbitrary output linear masks. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part I*, volume 13507 of *Lecture Notes in Computer Science*, pages 3–32. Springer, 2022.

[9] André Schrottenloher and Marc Stevens. Simplified MITM Modeling for Permutations: New (Quantum) Attacks. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part III*, volume 13509 of *Lecture Notes in Computer Science*, pages 717–747. Springer, 2022.

[10] Ties Speel. Cryptanalysis of sparkle's arx-box alzette. Bachelor Thesis, Radboud University, June 2022. `https://www.cs.ru.nl/bachelors-theses/2022/Ties_Speel___1020150___Cryptanalysis_of_SPARKLE_ARX-Box_Alzette.pdf`.

[11] Meltem Sönmez Turan, Kerry McKay, Donghoon Chang, Lawrence E. Bassham, Jinkeon Kang, Noah D. Waller, John M. Kelsey, and Deukjo Hong. Status report on the final round of the nist lightweight cryptography standardization process. NIST IR 8454, June 2023. `https://doi.org/10.6028/NIST.IR.8454`.

[12] Zheng Xu, Yongqiang Li, Lin Jiao, Mingsheng Wang, and Willi Meier. Do NOT misuse the markov cipher assumption - automatic search for differential and impossible differential characteristics in ARX ciphers. *IACR Cryptol. ePrint Arch.*, page 135, 2022.



Donghoon Chang received B.S. degree in mathematics from Korea University in 2001, and M.S. and Ph.D. degrees in information security from Korea University in 2003 and 2008, respectively. From 2009 to 2012, he was a guest researcher of NIST, USA. He was an assistant professor (2012-2016) and is an associate professor (2017-present) of Indraprastha Institute of Information Technology Delhi (IIIT-Delhi), India. From May 2019 to July 2021, he was a guest researcher of NIST, USA. Since August 2021, he has been working as a research scientist at Strativia, USA. His research interests are cryptanalysis, provable security of cryptographic algorithms, and biometric security.

Deukjo Hong received B.S. and M.S. degrees in mathematics from Korea University in 1999 and 2002, respectively, and a Ph.D. degree in information security from Korea University in 2006. From 2007 to 2015, he was employed at ETRI. He currently holds the position of Associate Professor in the Department of Information Technology & Engineering at Jeonbuk National University. He has been a guest researcher of NIST from September 2021 to February 2023. His research interests are cryptography and network & system security.

Jinkeon Kang received B.S. degrees in industrial engineering from Korea University in 2007, and a Ph.D. degree in information security from Korea University in 2021. He is a guest researcher of NIST since October 2019. His research interests include symmetric cryptography.