Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.33, NO.6, Dec. 2023 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2023.33.6.869

6 라운드로 축소된 Sparkle384와 7 라운드로 축소된 Sparkle512에 대한 새로운 구별 공격*

홍 득 조,[1†] 장 동 훈[2‡]
[1]전북대학교 스마트그리드연구센터 (교수), [2]IIIT-Delhi (교수)

New Distinguishing Attacks on Sparkle384 Reduced to 6 Rounds and Sparkle512 Reduced to 7 Rounds*

Deukjo Hong,[1†] Donghoon Chang[2‡]
[1]Smart Grid Research Center at Jeonbuk National University (Professor),
[2]IIIT-Delhi (Professor)

요 약

Sparkle은 NIST에서 최근까지 진행한 경량 암호 표준화 프로세스의 최종 후보 알고리즘 중 하나로서, 비선형 퍼뮤테이션이며, 인증 암호화 알고리즘 Schwaemm 및 해시함수 Esch의 핵심 구성 요소이다. 본 논문에서는 Sparkle의 두 버전 Sparkle384의 6 라운드와 Sparkle512의 7 라운드에 대해 특정한 형태의 입력 차분과 출력 차분을 제시하고, 그것을 만족시키는 입력쌍을 찾는 복잡도에 관한 공식을 제시한다. 또한, 같은 입출력 크기를 갖는 랜덤 퍼뮤테이션에 대한 동일 작업 보다 복잡도가 훨씬 낮을 가능성이 매우 크다는 것을 보인다. 그러므로, 이것들은 유효한 구별 공격이 된다. 공격되는 라운드 수(6과 7)는 실제 사용되는 라운드 수의 최소값(7과 8)과 매우 가깝다.

ABSTRACT

Sparkle is one of the finalists in the Lightweight Cryptography Standardization Process conducted by NIST. It is a nonlinear permutation and serves as a core component for the authenticated encryption algorithm Schwaemm and the hash function Esch. In this paper, we provide specific forms of input and output differences for 6 rounds of Sparkle384 and 7 rounds of Sparkle512, and make formulas for the complexity of finding input pairs that satisfy these differentials. Due to the significantly lower complexity compared to similar tasks for random permutations with the same input and output sizes, they can be valid distinguishing attacks. The numbers(6 and 7) of attacked rounds are very close to the minimum numbers(7 and 8) of really used rounds.

Keywords: Sparkle, Sparkle384, Sparkle512, Distinguishing Attack

I. 서 론

Sparkle은 NIST 경량 암호 표준화 프로세스 (Lightweight cryptography standardization process)[1]의 최종 후보 알고리즘 중 하나로서, 비선형 퍼뮤테이션이며, 인증 암호화 알고리즘 Schwaemm 및 해시함수 Esch의 핵심 구성 요소이다. Saprkle에 사용된 비선형 함수인 Alzette는

Received(10. 19. 2023), Accepted(11. 02. 2023)
* This work was supported by the National Research Foundation of Korea (NRF) funded by the Korean Government [Ministry of Science and ICT (MSIT)], South Korea, under Grant 2021R1A2C1005946.
† 주저자, deukjo.hong@jbnu.ac.kr
‡ 교신저자, donghoon.chang@nist.gov(Corresponding author)

64비트 입출력을 가지며, 32비트 덧셈, 로테이션, XOR로 구성된 ARX 구조이다. 또한 입출력 크기가 256비트인 Sparkle256, 384비트인 Sparkle384, 512비트인 Sparkle512의 세 가지 버전이 있다.

Sparkle에 대한 안전성 분석 결과는 알려진 것이 많지 않다. Bierle 등의 Sparkle 설계자들은 Sparkle256, Sparkle384, Sparkle512에 대하여 각각 4 라운드, 5 라운드, 6 라운드만으로는 차분 공격에 대해 b/2-비트 안전성을 보장할 수 없음을 보였고, 또한, 선형 공격에 대해서도 각각 5 라운드, 6 라운드, 6 라운드만으로는 b/2-비트 안전성을 보장할 수 없음을 보였다[2]. CRYPTO 2022에서는 Schrottenloher와 Stevens가 Sparkle256의 4 라운드와 Sparkle384의 4 라운드, Sparkle512의 5 라운드에 대한 guess-and- determine 구별자(Distinguisher)를 제시하였다[3].

본 논문에서는 Sparkle384와 Sparkle512의 6 라운드 및 7 라운드에 대하여 특정한 형태의 입력 차분과 출력 차분을 다음과 같이 설정하고,

Sparkle384의 6 라운드:
$$\Delta_I=(0,0,0,\alpha,0,0),\Delta_O=(0,\varepsilon,\varepsilon,\varepsilon,0,\varepsilon) \quad (1)$$

Sparkle512의 7 라운드:
$$\Delta_I=(0,0,0,0,\alpha,0,0,0),\Delta_O=(\xi,0,\zeta,\eta,\zeta,0,0,\zeta) \quad (2)$$

(1)과 (2)의 입력 차분 Δ_I와 출력 차분 Δ_O을 만족하는 쌍을 찾는 복잡도를 제시한다. 여기서, 입력 차분 Δ_I와 출력 차분 Δ_O에 있는 α, ε, ζ, η는 0이 아닌 어떠한 64비트 값도 될 수 있다. 반면에, ξ는 0을 포함한 어떠한 64비트 값도 될 수 있다. 6 라운드로 구성된 Sparkle384를 Sparkle384_6, 7 라운드로 구성된 Sparkle512를 Sparkle512_7로 표현하자.

Sparkle384_6 및 Sparkle512_7에 대해 각각 (1)과 (2)를 만족시키는 쌍을 찾는 각 과정의 복잡도는 $2^{66.8}$과 $2^{193.2}$으로 계산되며, 이것은 랜덤 퍼뮤테이션에 대하여 동일한 작업을 하는 복잡도 2^{257}보다 훨씬 낮다. 그러므로, 본 논문에서 제시되는 방법들은 각각 Sparkle384의 6 라운드와 Sparkle512의 7 라운드에 대한 유효한 구별 공격이 된다. 인증 암호화 알고리즘 Schwaemm256-128과 Schwaemm192-192에 핵심 구성요소로서 Sparkle384_{11}과 Sparkle384_7이 사용되고, 인증 암호화 알고리즘 Schwaemm256-256에 핵심 구성요소로서 Sparkle512_{12}와 Sparkle512_8이 사용되며, 안전성 증명에 랜덤 퍼뮤테이션 가정이 사용된다. 이런 점들을 고려할 때, 본 논문의 분석 대상이 실제 제안된 알고리즘에 매우 근접한다고 볼 수 있다.

논문의 나머지 부분은 다음과 같이 구성된다. 2장에서는 Sparkle384 및 Sparkle512의 구조에 대한 설명이 주어진다. 3장에서는 384비트 입출력 크기를 갖는 랜덤 퍼뮤테이션에 대하여 (1)을 만족시키는 쌍을 찾는 복잡도와 512비트 입출력 크기를 갖는 랜덤 퍼뮤테이션에 대하여 (2)를 만족시키는 쌍을 찾는 복잡도가 모두 2257임이 설명된다. 4장에서는 Sparkle384_6에 대하여 (1)을 만족시키는 쌍을 찾는 과정과 Sparkle 512_7에 대하여 (2)를 만족시키는 쌍을 찾는 과정이 설명되고 복잡도가 제시된다. 5장에서는 이 결과의 의미 및 향후 연구의 필요성에 대해 설명한다.

II. 배경 지식

2.1 정의 및 기호

같은 길이의 비트열 x와 y에 대한 비트 연산 XOR(배타적 논리합)을 $x\oplus y$로 표현한다. $\{0,1\}^n$을 모든 n비트 비트열들의 집합으로 정의한다. 본 논문에서 $\{0,1\}^n$은 GF(2) 상의 n차 벡터 공간으로 다루어지기도 한다.

2.2 Sparkle384

Sparkle384는 집합 $\{0,1\}^{384}$에 대한 비선형 퍼뮤테이션이며 라운드 반복 구조로 설계되어 있다. Sparkle384의 384-비트 입력값은 여섯 개의 64-비트 입력 워드 z_0^0, z_1^0, …, z_5^0 로 쪼개어진다. $i\geq 0$ 번째 라운드 함수(Round i)는 입력 워드 z_0^i, z_1^i, …, z_5^i 로부터 출력 워드 z_0^{i+1}, z_1^{i+1}, …, z_5^{i+1} 를 생성한다. 라운드 함수는 세 가지 계층 π, θ, ρ로 구성된다. Round i에서 π 계층은 32-비트 라운드 상수를 z_0^i에 XOR하고, 32-비트 라운드 카운터 i를 z_1^i에 더한다. π에서 수행되는 연산은 본 논문의 연구에서 고려되는 차분 전파에 거의 영향을 미치지 않기 때문에 이후로는 생략한다. 따라서, 다

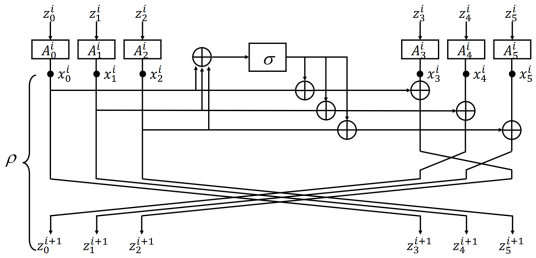


Fig. 1. Round function of Sparkle384 (Round i)

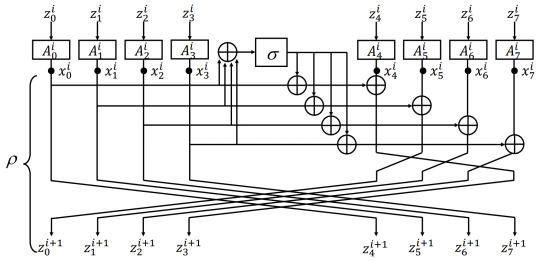


Fig. 2. Round function of Sparkle512 (Round i)

음으로 적용되는 θ 계층의 입력도 z_0^i, z_1^i, ..., z_5^i으로 표시한다. θ 계층은 비선형 Alzette 연산, A_j^i를 $x_j^i \leftarrow A_j^i(z_j^i)$ for $0 \le j \le 5$ 와 같이 적용한다. [2]에서 Sparkle 설계자들은 해당 Alzette A의 내부에서 사용되는 상수값 c_0, c_1, ...을 아래첨자로 사용하여 A_{c_0}, A_{c_1}, ...와 같이 표현하였는데, 본 논문에서는 각 Alzette 연산의 내부에 사용된 상수값 보다 해당 연산의 위치가 더 중요하게 다루어지기 때문에 i번째 라운드의 j번째 위치에 적용된 Alzette 연산이라는 의미의 A_j^i와 같이 표현한다. 본 논문에서는 Alzette가 64-비트 입출력 크기를 가진 비선형 퍼뮤테이션이라는 성질만 이용하기 때문에 Alzette의 구조에 대한 자세한 설명은 생략한다. ρ 계층은 64-비트 입출력을 갖는 간단한 선형 퍼뮤테이션 σ를 사용하여 다음과 같이 $(x_0^i, x_1^i, \ldots, x_5^i)$로부터 $(z_0^{i+1}, z_1^{i+1}, \ldots, z_5^{i+1})$을 계산한다:

$$t^i \leftarrow x_0^i \oplus x_1^i \oplus x_2^i;$$
$$z_{j-1 \bmod 3}^{i+1} \leftarrow \sigma(t^i) \oplus x_j^i \oplus x_{j+3}^i \text{ for } 0 \le j \le 2;$$
$$z_{j+3}^{i+1} \leftarrow x_j^i \text{ for } 0 \le j \le 2.$$

2.3 Sparkle512

Sparkle512는 집합 $\{0,1\}^{512}$에 대한 비선형 퍼뮤테이션이며 라운드 반복 구조로 설계되어 있다. 또한, Sparkle384와 동일한 Alzette 및 σ 연산을 사용하며 유사한 구조를 갖는다. Sparkle512의 512-비트 입력값은 여덟 개의 64-비트 입력 워드 z_0^0, z_1^0, ..., z_7^0 로 쪼개어진다. $i \ge 0$ 번째 라운드 함수(Round i)는 입력 워드 z_0^i, z_1^i, ..., z_7^i 로부터 출력 워드 z_0^{i+1}, z_1^{i+1}, ..., z_7^{i+1} 를 생성한다. 라운드 함수는 세 가지 계층 π, θ, ρ로 구성되는데, 2.2절에서와 같은 이유로 π 계층은 생략한다. Round i에서 θ 계층의 입력도 z_0^i, z_1^i, ..., z_7^i으로 표시한다. θ 계층은 비선형 Alzette 연산, A_j^i를 $x_j^i \leftarrow A_j^i(z_j^i)$ for $0 \le j \le 7$ 과 같이 적용한다. ρ 계층은 선형 퍼뮤테이션 σ를 사용하여 다음과 같이 $(x_0^i, x_1^i, \ldots, x_7^i)$로부터 $(z_0^{i+1}, z_1^{i+1}, \ldots, z_7^{i+1})$을 계산한다:

$$t^i \leftarrow x_0^i \oplus x_1^i \oplus x_2^i \oplus x_3^i;$$
$$z_{j-1 \bmod 4}^{i+1} \leftarrow \sigma(t^i) \oplus x_j^i \oplus x_{j+4}^i \text{ for } 0 \le j \le 3;$$
$$z_{j+4}^{i+1} \leftarrow x_j^i \text{ for } 0 \le j \le 3.$$

III. 랜덤 퍼뮤테이션에 대하여 입력 차분과 출력 차분을 만족시키는 쌍 찾기

$\{0,1\}^{384}$에 대한 랜덤 퍼뮤테이션에 대해 (1)의 입력 차분과 출력 차분을 만족시키는 쌍을 찾는 과정이 3.1 절에서 설명되고 복잡도가 퍼뮤테이션 오라클 질의 횟수로 계산된다. 유사한 논리와 방법으로, $\{0,1\}^{512}$에 대한 랜덤 퍼뮤테이션에 대해 (2)의 입력 차분과 출력 차분을 만족시키는 쌍을 찾는 과정이 3.2 절에서 설명되고 복잡도가 계산된다.

3.1 $\{0,1\}^{384}$에 대한 랜덤 퍼뮤테이션에 대해 (1)의 입력 차분과 출력 차분을 만족시키는 쌍 찾기

P384를 $\{0,1\}^{384}$에 대한 모든 퍼뮤테이션들의 집합이라고 하자. P384에서 랜덤하게 선택된 P에 대하여, P 오라클과 P^{-1} 오라클에 대한 접근 권한이 주어진다고 가정한다. 이 때, P에 대해 (1)을 만족시키는 쌍을 찾는 방법은 다음과 같이 설명될 수 있다.

서로 다른 두 384-비트 값 X와 X'에 대해, 두 값의 XOR에 무관하게, $P(X)\oplus P(X')=\Delta_O$ 가 성립할 확률은 약 2^{-320}이다. 그러므로, Δ_I 를 만족시키는 입력쌍 2^{320}개를 모은다면 평균적으로 (1)을 만족시키는 쌍 1개를 기대할 수 있다. Δ_I의 α에 대한 제한 조건은 단지 그것들이 0이 아니라는 것뿐임을 이용하면 효율적으로 쌍들을 모을 수 있다.

384비트 값 X에 대해 집합 $T(X)$를 다음과 같이 정의하자.

$$W=\{(0,0,0,a,0,0)\mid a\in\{0,1\}^{64}\};$$
$$T(X)=X\oplus W=\{X\oplus w\mid w\in W\}.$$

W는 $V=\{0,1\}^{384}$의 64차 선형 부분공간이며, $T(X)$는 W의 잉여류(coset)이다. 따라서, $T(X)$는 V/W의 원소이고, V/W에는 2^{320}개의 서로 다른 $T(X)$ 집합이 존재한다. 각 $T(X)$는 2^{64}개의 원소를 가지며, $T(X)$의 임의의 서로 다른 두 원소 $X\oplus w$와 $X\oplus w'$은 입력 차분 Δ_I를 만족시킨다. 그러나, 서로 다른 두 집합 $T(X)$와 $T(X')$으로부터의 어떠한 두 원소 $X\oplus w$와 $X'\oplus w'$도 Δ_I를 만족시키지 않는다. 그러므로, 각 $T(X)$에 대하여 Δ_I를 만족시키는 쌍을 약 $2^{127}(\approx 2^{64}(2^{64}-1)/2)$개 기대할 수 있으며, 2^{193}개의 서로 다른 $T(X)$ 집합을 이용하면 2^{320}개의 쌍을 모을 수 있다. 그러므로, 복잡도는 P에 대한 $2^{193+64}=2^{257}$번의 질의로 계산된다.

P^{-1}에 대한 질의를 이용하는 방법 또한 동일한 복잡도를 소요한다.

3.2 $\{0,1\}^{512}$에 대한 랜덤 퍼뮤테이션에 대해 (2)의 입력 차분과 출력 차분을 만족시키는 쌍 찾기

P_{512}를 $\{0,1\}^{512}$에 대한 모든 퍼뮤테이션들의 집합이라고 하자. P_{512}에서 랜덤하게 선택된 P에 대하여, P 오라클과 P^{-1} 오라클에 대한 접근 권한이 주어진다고 가정한다. 이 때, P에 대해 (2)을 만족시키는 쌍을 찾는 방법은 다음과 같이 설명될 수 있다.

서로 다른 두 512-비트 값 X와 X'에 대해, 두 값의 XOR에 무관하게, $P(X)\oplus P(X')=\Delta_O$ 가 성립할 확률은 약 2^{-320}이다. 그러므로, Δ_I 를 만족시키는 입력쌍 2^{320}개를 모은다면 평균적으로 (2)를 만족시키는 쌍 1개를 기대할 수 있다. 3.1절에서와 비슷한 방법으로, 512비트 값 X에 대해 집합 $T(X)$를 다음과 같이 정의하자.

$$W=\{(0,0,0,0,a,0,0,0)\mid a\in\{0,1\}^{64}\};$$
$$T(X)=X\oplus W=\{X\oplus w\mid w\in W\}.$$

W는 $V=\{0,1\}^{512}$의 64차 선형 부분공간이며, $T(X)$는 W의 잉여류(coset)이다. 따라서, $T(X)$는 V/W의 원소이고, V/W에는 2^{448}개의 서로 다른 $T(X)$ 집합이 존재한다. 각 $T(X)$는 2^{64}개의 원소를 가지며, $T(X)$의 임의의 서로 다른 두 원소 $X\oplus w$와 $X\oplus w'$은 입력 차분 Δ_I를 만족시킨다. 그러나, 서로 다른 두 집합 $T(X)$와 $T(X')$으로부터의 어떠한 두 원소 $X\oplus w$와 $X'\oplus w'$도 Δ_I를 만족시키지 않는다. 그러므로, 각 $T(X)$에 대하여 Δ_I를 만족시키는 쌍을 약 $2^{127}(\approx 2^{64}(2^{64}-1)/2)$개 기대할 수 있으며, 2^{193}개의 서로 다른 $T(X)$ 집합을 이용하면 2^{320}개의 쌍을 모을 수 있다. 그러므로, 복잡도는 P에 대한 $2^{193+64}=2^{257}$번의 질의로 계산된다.

P^{-1}에 대한 질의를 이용하는 방법은 약 두 배의 질의 횟수를 소요하기 때문에, 위의 방법이 조금 더 효율적이다.

IV. Sparkle에 대하여 입력 차분과 출력 차분을 만족시키는 쌍 찾기

4.1 Sparkle384$_6$에 대해 (1)의 입력 차분과 출력 차분을 만족시키는 쌍 찾기

Δz_i^j와 Δx_i^j를 각각 z_i^j와 x_i^j의 차분이라고 하자. Sparkle3846에 대해 (1)의 입력 차분과 출력 차분을 만족시키는 쌍은 다음과 같은 단계들(Step 1 ~ Step 4)을 통해 찾을 수 있다.

Step 1: Round 2의 왼쪽 세 입력 워드들의 차분을 다음과 같이 설정한다.

$$(\Delta z_0^2,\Delta z_1^2,\Delta z_2^2)=(\sigma(\beta),\beta\oplus\sigma(\beta),\sigma(\beta)) \qquad (3)$$

그리고 다음과 같은 식 (4)을 만족시키는 (z_0^2,z_1^2,z_2^2)에 대한 쌍을 찾는다.

$$\Delta x_0^2 \oplus \sigma(\Delta t^2) = 0 \quad (4)$$

(4)가 성립할 확률은 2^{-64}이다. 그러한 쌍이 발견된다면 (z_3^3, z_4^3, z_5^3)의 값과 차분이 모두 결정되며 또한 차분 $\Delta z_0^3 = \gamma_0$와 $\Delta z_2^3 = 0$도 결정된다. 게다가, $\Delta z_2^3 = 0$는 $\Delta x_2^3 = \Delta z_5^4 = 0$을 의미한다. (4)를 만족시키는 쌍으로부터 차분 Δx_3^3, Δx_4^3, Δx_5^3를 계산한 다음, $\Delta z_0^4 = \Delta z_1^4 = \Delta z_2^4 = 0$을 가정하면 $\sigma(\Delta t^3) = \Delta x_5^3$, $\Delta x_0^3 = \Delta x_3^3 \oplus \Delta x_5^3$, $\Delta x_1^3 = \Delta x_2^3 \oplus \Delta x_5^3$를 얻게 된다. 마지막으로, 다음 식 (5)가 확률 2^{-64}로 성립하기를 기대한다.

$$\sigma(\Delta x_0^3 \oplus \Delta x_1^3) = \Delta x_5^3 \quad (5)$$

그러므로, (3)을 만족시키는 쌍 2^{128}개가 필요하다. 여기서 β에 대한 조건은 0이 아닌 것뿐임을 이용하여 쌍을 효율적으로 모을 수 있다. 192비트 값 X에 대해 집합 $S(X)$를 다음과 같이 정의하자.

$$W = \{(\sigma(a), a \oplus \sigma(a), \sigma(a)) \mid a \in \{0,1\}^{64}\};$$
$$S(X) = X \oplus W = \{X \oplus w \mid w \in W\}.$$

$S(X)$는 (3)을 만족시키는 $2^{64}(2^{64}-1)/2 \approx 2^{127}$개의 쌍을 만들어낸다. 그러므로 두 개의 서로 다른 집합 $S(X)$와 $S(X')$이 있으면 2^{128}개의 쌍을 얻게 된다. 이것으로 (4)와 (5)를 모두 만족시키는 1개의 쌍을 기대할 수 있다.

Step 1의 복잡도 C_1를 계산해보자. 복잡도는 Alzette 연산 시간을 단위로 계산된다. 그밖의 선형 연산들은 Alzette 연산에 비해 훨씬 가볍기 때문이다. $S(X)$와 $S(X')$의 각 원소에 대해 먼저 A_0^2, A_1^2, A_2^2에 해당하는 세 번의 Alzette 연산이 적용된다. 그 다음 2^{128}개 쌍 중에 몇 쌍이 (4)를 만족시키는지 체크한다. 평균적으로는 2^{64} 쌍이 (4)를 만족시킨다. 그 쌍들 중 (5)를 만족시키는 것이 있는지 확인하기 위해 각 쌍마다 A_3^3, A_4^3, A_5^3이 적용되는데, 최대 여섯 번의 Alzette 연산이 요구된다. 그러므로, C_1은 다음과 같이 계산된다.

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

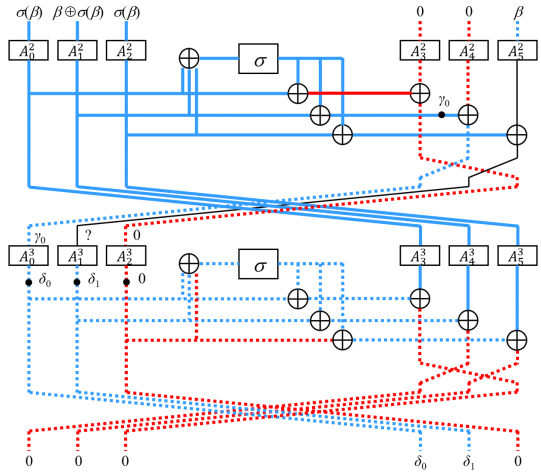


Fig. 3. Situation in Rounds 3 and 4 after Step 1 in finding a right pair of (1) for Sparkle384$_6$

Fig.3은 Step 1 이후의 Round 2와 Round 3에서의 상황을 표현하고 있다. 굵은 빨간 점선은 값은 미정(undetermined)이지만 차분은 0임을 의미한다. 굵은 파란 점선은 값은 미정이지만 차분이 결정되었고 0이 아님을 의미한다. 굵은 빨간 실선은 값이 결정되어 있고 차분이 0임을 의미한다. 굵은 파란 실선은 값이 결정되어 있고 차분이 0이 아님을 의미한다. 검은 실선은 값과 차분 모두 미정임을 의미한다.

Step 2: Fig.3과 같이, $\Delta z_0^3 = \gamma_0$이고 $\Delta x_0^3 = \delta_0$라고 하자. A_0^3에 대해, 입력 차분 γ_0을 만족시키는 모든 2^{64}개의 입력쌍을 시도하여 출력 차분 δ_0을 만족시키는 쌍이 있는지 확인한다. 만약 그러한 쌍을 찾지 못한다면 다시 Step 1을 수행한다. 따라서, Step 2의 복잡도 C_2는 $C_2 = 2^{65}A$으로 계산된다.

Step 3: Fig.3에서와 같이, $\Delta x_1^3 = \delta_1$이라고 하자. Step 1에서 찾은 쌍은 $x_2^2 \oplus \sigma(t^2)$의 값들을 결정한다. 그 값들을 k와 k'이라고 하자. $z_5^2 \oplus z_5^{2'} = \beta$를 만족시키는 2^{64}개의 모든 $(z_5^2, z_5^{2'})$쌍에 대해

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

을 만족시키는 쌍이 있는지 확인한다. 만약 그러한 쌍을 찾지 못한다면 다시 Step 1을 수행한다. 따라

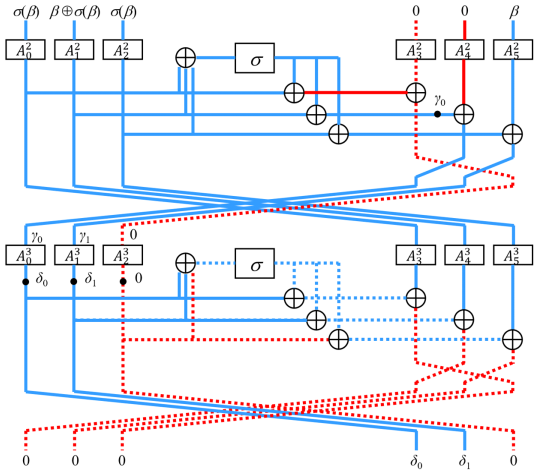


Fig. 4. Situation in Rounds 3 and 4 after Steps 2 and 3 in finding a right pair of (1) for Sparkle384$_6$

서, Step 3의 복잡도 C_3는 $C_3 = 2^{66}A$으로 계산된다. Fig.4는 Step 2와 Step 3이 성공한 후의 Round 2와 Round 3의 상황을 보여준다.

Step 4: 이전의 Step들이 모두 성공했다면, 이 단계에서는 z_2^3를 제외한 Round 3의 모든 입력 워드들의 값들이 결정되어있게 된다. 따라서, 유일한 미정 워드인 $x_2^3 = A_2^3(z_2^3)$는 다른 모든 미정 워드들과 연관되어 있기 때문에, x_2^3의 64 자유도를 이용하여 x_0^5과 x_2^5의 차분들이 일치하도록(즉, $\Delta x_0^5 = \Delta x_2^5$) 할 수 있다. 이것에는 x_2^3의 가능한 모든 2^{64} 가지 값들을 통한 시도가 평균적으로 요구된다. 이전 Step들의 성공으로부터, $(x_i^3, x_i^{3'})$ 값들이 $i = 0, 1, 3, 4, 5$에 대하여 확보되어 있다면, x_2^3의 각 값에 대해 다음의 계산들을 수행함으로써 $\Delta x_0^5 = \Delta x_2^5$인지 확인할 수 있다:

$$
\begin{aligned}
&t^3 = x_0^3 \oplus x_1^3 \oplus x_2^3;\\
&x_i^4 = A_i^4(x_{i+1 \bmod 3}^3 \oplus \sigma(t^3)) \text{ for } i \in \{0,1,2\};\\
&t^4 = x_0^4 \oplus x_1^4 \oplus x_2^4;\\
&x_0^5 = A_0^5(A_4^4(x_1^3) \oplus x_1^4 \oplus \sigma(t^4));\\
&x_0^{5'} = A_0^5(A_4^4(x_1^{3'}) \oplus x_1^4 \oplus \sigma(t^4));\\
&x_2^5 = A_2^5(A_3^4(x_0^3) \oplus x_0^4 \oplus \sigma(t^4));
\end{aligned}
$$

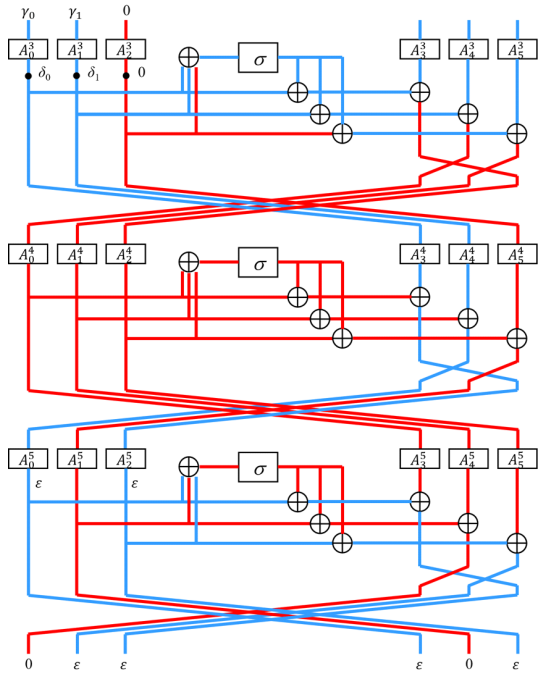


Fig. 5. Forward propagation of differences from Round 3 to Round 5 after Step 4 in finding a right pair of (1) for Sparkle384$_6$

$$x_2^{5'} = A_2^5(A_3^4(x_0^{3'}) \oplus x_0^4 \oplus \sigma(t^4)).$$

이와 같은 과정에 따라, Step 4의 복잡도 C_4는 $C_4 = 2^{64} \cdot 11A$로 계산된다.

만약 위 식들을 만족시키는 x_2^3를 찾는다면, 나머지 미정 워드들로의 전파(propagation)는 쉽게 계산된다. Fig.5는 $(z_0^3, \ldots, z_5^3)$에서 출력 차분 Δ_O까지의 정방향 전파, Fig.6은 $(z_0^2, \ldots, z_5^2)$에서 입력 차분 Δ_I까지의 역방향 전파의 결과를 보여준다.

4.2 Sparkle512$_7$에 대해 (2)의 입력 차분과 출력 차분을 만족시키는 쌍 찾기

Sparkle512$_7$에 대해 (2)의 입력 차분과 출력 차분을 만족시키는 쌍은 다음과 같은 단계들(Step 1 ~ Step 4)을 통해 찾을 수 있다.

Step 1: Round 2의 왼쪽 네 입력 워드들의 차분을 다음과 같이 설정한다.

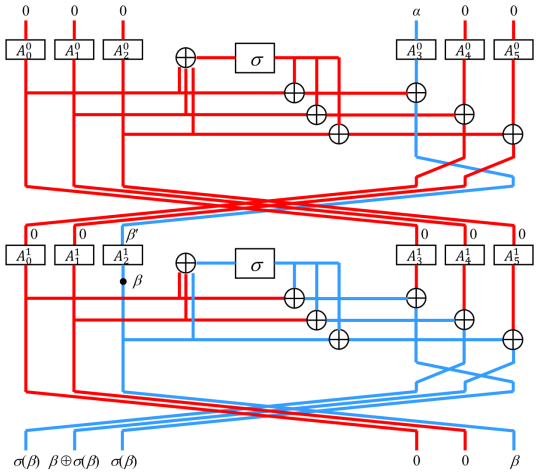


Fig. 6. Backward propagation of differences from Round 1 to Round 0 after Step 4 in finding a right pair of (1) for Sparkle384$_6$

$$(\Delta z_0^2,\Delta z_1^2,\Delta z_2^2,\Delta z_2^2) = (\sigma(\beta),\sigma(\beta),\beta\oplus\sigma(\beta),\sigma(\beta)) \quad (6)$$

그리고 다음과 같은 식 (7)을 만족시키는 $(z_0^2,\ z_1^2,\ z_2^2,\ z_3^2)$에 대한 쌍을 찾는다.

$$(\Delta x_0^2\oplus\sigma(\Delta t^2),\Delta x_1^2\oplus\sigma(\Delta t^2)) = (0,0). \quad (7)$$

(7)이 성립할 확률은 2^{-128}이다. 그러한 쌍이 발견된다면 $(z_4^2,z_5^2,z_6^2,z_7^2)$의 값과 차분이 모두 결정되며 또한 차분 $\Delta z_0^3 = \Delta z_3^3 = 0$와 $\Delta z_1^3 = \gamma_0$도 결정된다. 게다가, $\Delta z_0^3 = \Delta z_3^3 = 0$은 $\Delta x_0^3 = \Delta z_4^4 = 0$와 $\Delta x_3^3 = \Delta z_7^4 = 0$을 의미한다. (7)를 만족시키는 쌍으로부터 차분 Δx_4^3, Δx_5^3, Δx_6^3, Δx_7^3를 계산한 다음에, $\Delta z_0^4 = \cdots = \Delta z_3^4 = 0$을 가정하면 $\sigma(\Delta t^3) = \Delta x_4^3$, $\Delta x_1^3 = \Delta x_4^3\oplus\Delta x_5^3$, $\Delta x_2^3 = \Delta x_4^3\oplus\Delta x_6^3$를 얻게 된다. 마지막으로, 다음 식 (8)과 (9)가 각각 확률 2^{-64}로 성립하기를 기대한다.

$$\Delta x_4^3 = \Delta x_7^3; \quad (8)$$

$$\sigma(\Delta x_1^3\oplus\Delta x_2^3) = \Delta x_4^3. \quad (9)$$

그러므로, (6)을 만족시키는 쌍 2^{128}개가 필요하다.

여기서 β에 대한 조건은 0이 아닌 것뿐임을 이용하여 쌍을 효율적으로 모을 수 있다. 256비트 값 X에 대해 집합 $S(X)$를 다음과 같이 정의하자.

$$W = \{(\sigma(a),\sigma(a),a\oplus\sigma(a),\sigma(a)) \mid a\in\{0,1\}^{64}\};$$
$$S(X) = X\oplus W = \{X\oplus w \mid w\in W\}.$$

$S(X)$는 (6)을 만족시키는 $2^{64}(2^{64}-1)/2 \approx 2^{127}$개의 쌍을 만들어낸다. 그러므로 2^{129}개의 서로 다른 $S(X)$ 집합이 있으면 2^{256}개의 쌍을 얻게 된다. 이것으로 (7), (8), (9)를 모두 만족시키는 1개의 쌍을 기대할 수 있다.

Step 1의 복잡도 C_1를 계산해보자. 각 $S(X)$의 각 원소에 대해 먼저 A_0^2, A_1^2, A_2^2, A_3^2에 해당하는 네 번의 Alzette 연산이 적용된다. 그리고 이 2^{256}개 쌍 중에 몇 쌍이 (7)을 만족시키는지 체크한다. 평균적으로는 2^{128} 쌍이 (7)을 만족시킨다. 그 쌍들 중 (8)을 만족시키는 것이 있는지 확인하기 위해 각 쌍마다 A_4^3, A_7^3이 적용된다. 평균적으로는 남은 쌍들 중 2^{64} 쌍이 (8)을 만족시킨다. 마지막으로, 그 쌍들 중 (9)를 만족시키는 것이 있는지 확인하기 위해 각 쌍마다 A_5^3, A_6^3이 적용된다. 그러므로, C_1은 다음과 같이 계산된다.

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

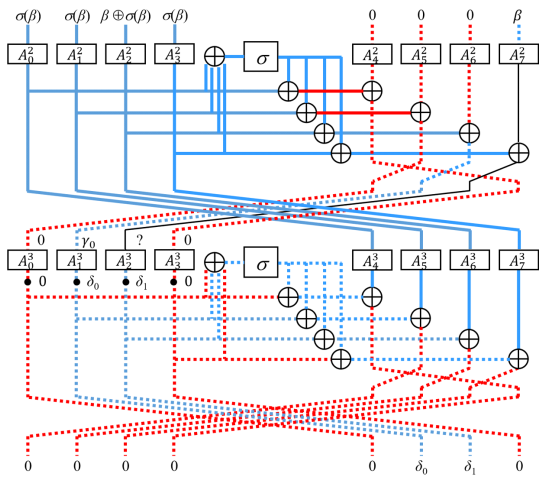


Fig. 7. Situation in Rounds 3 and 4 after Step 1 in finding a right pair of (2) for Sparkle512$_7$

Fig.7은 Step 1 이후의 Round 2와 Round 3에서의 상황을 표현하고 있다.

Step 2: Fig.7과 같이, $\Delta z_1^3=\gamma_0$이고 $\Delta x_1^3=\delta_0$라고 하자. A_1^3에 대해, 입력 차분 γ_0을 만족시키는 모든 2^{64}개의 입력쌍을 시도하여 출력 차분 δ_0을 만족시키는 쌍이 있는지 확인한다. 만약 그러한 쌍을 찾지 못한다면 다시 Step 1을 수행한다. 따라서, Step 2의 복잡도 C_2는 $C_2=2^{65}A$으로 계산된다.

Step 3: Fig.7에서와 같이, $\Delta x_2^3=\delta_1$이라고 하자. Step 1에서 찾은 쌍은 $x_3^2\oplus\sigma(t^2)$의 값들을 결정한다. 그 값들을 k와 k'이라고 하자. $z_7^2\oplus z_7^{2'}=\beta$를 만족시키는 2^{64}개의 모든 $(z_7^2,z_7^{2'})$쌍에 대해

$$A_2^3(A_7^2(z_7^2)\oplus k)\oplus A_2^3(A_7^2(z_7^{2'})\oplus k')=\delta_1$$

을 만족시키는 쌍이 있는지 확인한다. 만약 그러한 쌍을 찾지 못한다면 다시 Step 1을 수행한다. 따라서, Step 3의 복잡도 C_3는 $C_3=2^{66}A$으로 계산된다. Fig.8은 Step 2와 Step 3이 성공한 후의 Round 2와 Round 3의 상황을 보여준다.

Step 4: 이전의 Step들이 모두 성공했다면, 이 단계에서는 z_0^3과 z_3^3을 제외한 Round 3의 모든 입력 워드들의 값들이 결정되어있게 된다. 따라서, 유이한 미정 워드인 $x_0^3=A_0^3(z_0^3)$과 $x_3^3=A_3^3(z_3^3)$는 다른 모든 미정 워드들과 연관되어 있기 때문에, (x_0^3,x_3^3)의 128 자유도를 이용하여 $\Delta x_0^5=\Delta x_1^5$과 $\Delta x_0^6=\Delta x_3^6$이 성립하게 할 수 있다. 이것에는 (x_0^3,x_3^3)의 가능한 모든 2^{128} 가지 값들을 통한 시도가 평균적으로 요구된다. 이전 Step들의 성공으로부터, $(x_i^3,x_i^{3'})$ 값들이 $i=1,\ 2,\ 4,\ 5,\ 6,\ 7$에 대해 확보되어 있다면, $(x_0^3,\ x_3^3)$의 각 값에 대해 다음의 계산들을 수행함으로써 $\Delta x_0^5=\Delta x_1^5$인지 확인할 수 있다:

$$\begin{aligned}
&t^3=x_0^3\oplus x_1^3\oplus x_2^3\oplus x_3^3;\\
&x_i^4=A_i^4(x_{i+1\bmod 4}^3\oplus\sigma(t^3))\ \text{ for }\ i\in\{0,1,2,3\};\\
&t^4=x_0^4\oplus x_1^4\oplus x_2^4\oplus x_3^4;\\
&x_0^5=A_0^5(A_5^4(x_1^3)\oplus x_1^4\oplus\sigma(t^4));\\
&x_0^{5'}=A_0^5(A_5^4(x_1^{3'})\oplus x_1^4\oplus\sigma(t^4));\\
&x_1^5=A_1^5(A_6^4(x_2^3)\oplus x_2^4\oplus\sigma(t^4));\\
&x_1^{5'}=A_1^5(A_6^4(x_2^{3'})\oplus x_2^4\oplus\sigma(t^4)).
\end{aligned}$$

평균적으로, 2^{64}개의 $(x_0^3,\ x_3^3)$ 값들이 $\Delta x_0^5=\Delta x_1^5$을 만족시킨다. $\Delta x_2^5=\Delta x_3^5=0$이므로, $\Delta x_0^5=\Delta x_1^5$이면 $\Delta t^6=0$이다. 남아있는 $(x_0^3,\ x_3^3)$의 각 값에 대하여 다음의 계산들을 수행함으로써, $\Delta x_0^6=\Delta x_3^6$인지 확인할 수 있다:

$$\begin{aligned}
&x_2^5=A_3^5(A_7^4(x_3^3)\oplus x_3^4\oplus\sigma(t^4));\\
&x_3^5=A_2^5(A_4^4(x_0^3)\oplus x_0^4\oplus\sigma(t^4));\\
&t^5=x_0^5\oplus x_1^5\oplus x_2^5\oplus x_3^5;\\
&x_0^6=A_0^6(A_5^5(x_1^4)\oplus x_1^5\oplus\sigma(t^5));\\
&x_0^{6'}=A_0^6(A_5^5(x_1^{4'})\oplus x_1^5\oplus\sigma(t^5));\\
&x_3^6=A_3^6(A_4^5(x_0^4)\oplus x_0^5\oplus\sigma(t^5));\\
&x_3^{6'}=A_3^6(A_4^5(x_0^4)\oplus x_0^{5'}\oplus\sigma(t^5)).
\end{aligned}$$

이와 같은 과정에 따라, Step 4의 복잡도 C_4는 $C_4=2^{128}\cdot 12A+2^{64}\cdot 12A\simeq 2^{130}\cdot 3A$로 계산된다.

위 식들을 만족시키는 $(x_0^3,\ x_3^3)$를 구한다면, 나머

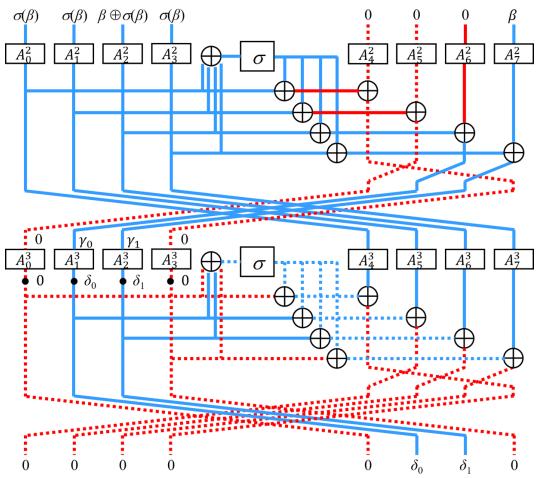


Fig. 8. Situation in Rounds 3 and 4 after Steps 2 and 3 in finding a right pair of (2) for Sparkle512$_7$

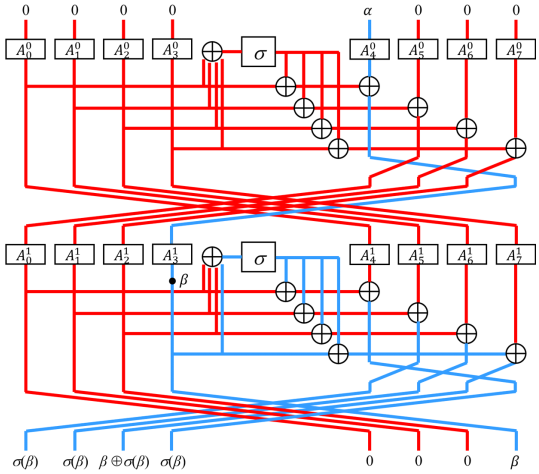


Fig. 9. Backward propagation of differences from Round 1 to Round 0 after Step 4 in finding a right pair of (2) for Sparkle512$_7$

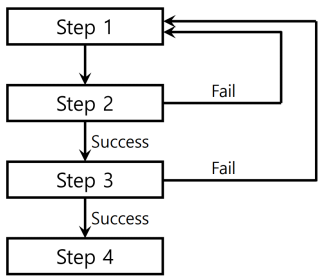


Fig. 10. Flow from Step 1 to Step 4 in finding right pairs for Sparkle permutations

지 미정 워드들로의 전파(propagation)는 쉽게 계산된다. Fig.9는 $(z_0^3,...,z_7^3)$에서 출력 차분 Δ_O까지의 정방향 전파, Fig.10은 $(z_0^2,...,z_7^2)$에서 입력 차분 Δ_I까지의 역방향 전파의 결과를 보여준다.

4.3 복잡도

Fig.11은 4.1절과 4.2절에서 설명된 방법들에 대하여 Step 1부터 Step 4까지의 간단한 순서도를 보여준다. p와 q를 각각 Step 2와 Step 3의 성공 확률이라고 하자. 그러면 각 방법의 총 복잡도 C는

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

와 같이 표현된다.

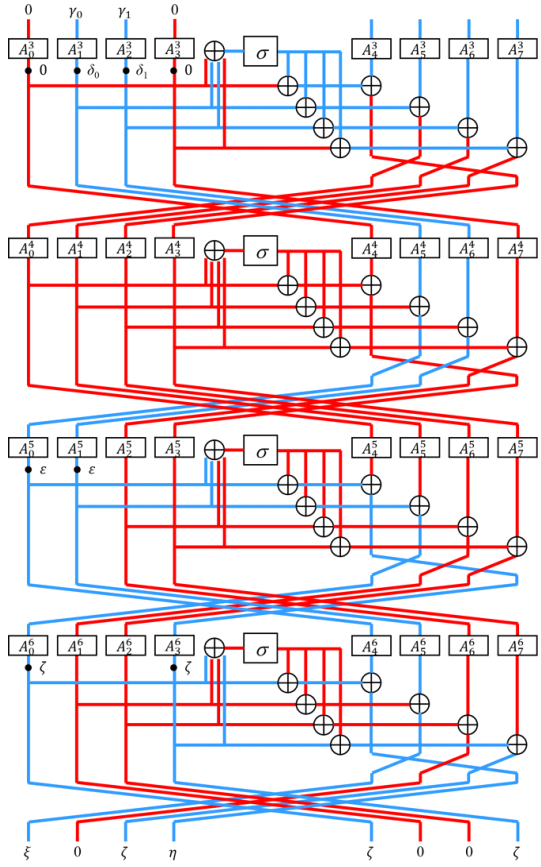


Fig. 11. Forward propagation of differences from Round 3 to Round 6 after Step 4 in finding a right pair of (2) for Sparkle512$_7$

p는 Alzette 연산의 차분 분포 테이블(Difference Distribution Table)에서 0이 아닌 성분의 비율과 같다. Alzette는 64비트 크기의 입력과 출력을 갖는 연산이므로 p에 대한 정확한 값을 구하는 것은 어렵다. 그런데, Alzette는 64비트 4-라운드 ARX 구조를 갖고 있으며[1], 적절한 로테이션 연산들을 사용하고 있기 때문에 32비트 덧셈에서 발생할 수 있는 차분 성질이 출력의 각 비트에 고르게 확산된다. 따라서, p의 값은 1/4과 1/2 사이에 있을 것으로 추측된다. 마찬가지로, q의 값 또한 1/4과 1/2 사이에 있을 것으로 추측된다. 작은 크기(8비트~12비트)의 입출력을 갖도록 축소된 형태의 Alzette에 대하여 테스트한 결과들은 이러한 추측을 뒷받침한다.

먼저, 4.1절에서 설명된 방법의 복잡도를 계산해 보자. 식 (10)에서 p와 q를 모두 1/4로 가정하고,

Step 1부터 Step 4까지 계산된 복잡도 C_1, C_2, C_3, C_4를 대입하면, C는 다음과 같이 계산된다:

$$C = ((2^{66} \cdot 3A + 2^{65}A)2^2 + 2^{66}A)2^2 + 2^{64} \cdot 11A$$
$$= 2^{64} \cdot 251A.$$

한 번의 Sparkle384_6 연산은 36번의 Alzette 연산을 요구하므로, 위의 값은 다시 $C \simeq 2^{66.8}$로 환산될 수 있다. 이 복잡도는 3.1절에서 제시된, 랜덤 퍼뮤테이션에 대한 복잡도 2^{257} 보다 훨씬 작기 때문에, 4.1절에서 설명되는 방법은 Sparkle384_6에 대한 유효한 구별 공격이 된다.

4.2절에서 설명된 방법에서는 Step 1의 복잡도가 다른 단계의 복잡도에 비해 월등히 높다. 그러므로, 식 (10)은 $C \simeq C_1 p^{-1} q^{-1}$로 수정될 수 있다. p와 q를 모두 1/4로 가정하고, $C_1 = 2^{195}A$를 대입하면, $C \simeq 2^{199}A$로 계산된다. 한 번의 Sparkle512_7 연산이 56번의 Alzette 연산을 소요한다는 것을 고려하면 이것은 $C \simeq 2^{193.2}$로 환산된다. 이 복잡도 또한 3.2절에서 제시된 랜덤 퍼뮤테이션에 대한 복잡도 2^{257} 보다 훨씬 작기 때문에, 4.2절에서 Sparkle512_7에 대해 설명되는 방법은 유효한 구별 공격이 된다.

V. 결 론

Sparkle은 NIST에서 진행한 경량 암호 표준화 프로세스의 10 가지 최종 후보 알고리즘 중 하나이다. 본 논문에서는 Sparkle384의 6 라운드와 Sparkle512의 7 라운드에 대하여 새로운 구별 공격을 제시하였다. 공격된 라운드 수는 실제 제안된 알고리즘에서 사용된 라운드 수와 불과 1 라운드 차이이다.

Alzette는 64비트 4 라운드 ARX 구조로 되어 있으며, 몇 가지 분석 결과들이 제시되어 있으나 [1,4-8], 이러한 ARX 구조의 차분 분포를 정확하게 파악하는 것은 매우 어려운 작업이다. 본 논문에서 제시된 구별 공격들의 복잡도 추정에는 Alzette 차분 분포에 대한 특별한 가정이 고려되었는데, 만약 향후 연구에서 Alzette 차분 분포에 대한 성질들이 좀 더 자세하게 분석된다면, 본 논문에서 제시된 구별 공격의 복잡도가 더욱 명확해질 것으로 기대한다.

References

[1] 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 Internal Report, NIST IR 8454, June 2023.

[2] Christof Beierle, Alex Biryukov, Luan Cardoso dos Santos, Johann Großschädl, Léo Perrin, Aleksei Udovenko, Vesselin Velichkov, and Qingju Wang, "Lightweight AEAD and Hashing using the Sparkle Permutation Family," IACR Transactions on Symmetric Cryptology, Vol. 2020, No. S1, pp. 208-261, Jun. 2020.

[3] André Schrottenloher and Marc Stevens, "Simplified MITM Modeling for Permutations: New (Quantum) Attacks," Advances in Cryptology - CRYPTO 2022, LNCS 13509, pp. 717-747, Springer, 2022.

[4] Mingjiang Huang, Zhen Xu, and Liming Wang, "On the Probability and Automatic Search of Rotational-XOR Cryptanalysis on ARX Ciphers," The Computer Journal, Vol. 65, Issue 12, December 2022, pp. 3062-3080.

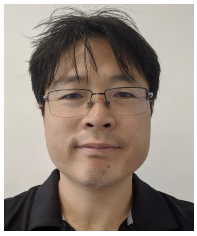
[5] Yunwen Liu, Siwei Sun, and Chao Li, "Rotational Cryptanalysis from a Differential-Linear Perspective," Advanced in Cryptology - EUROCRYPT 2021, LNCS 12696, pp. 741-770, Springer, 2021.

[6] Zhongfeng Niu, Siwei Sun, Yunwen Liu, and Chao Li, "Rotational Differential-Linear Distinguishers of ARX Ciphers with Arbitrary Output Linear Masks," Advanced in

Cryptology - CRYPTO 2022, LNCS 13507, pp. 3-32, Springer, 2022.
〔7〕 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," Cryptology ePrint Archive, 2022/135.
〔8〕 Ties Speel, "Cryptanalysis of Sparkle's ARX-Box Alzette," Bachelor Thesis, Radboud University, June 2022.

〈저자소개〉

홍 득 조 (Deukjo Hong) 종신회원
1999년 8월: 고려대학교 수학과 학사
2001년 8월: 고려대학교 수학과 석사
2006년 2월: 고려대학교 정보보호대학원 박사
2006년 3월~2007년 12월: 고려대학교 정보보호기술연구센터 연구교수
2007년 12월~2015년 8월: 국가보안기술연구소 선임연구원
2015년 9월~현재: 전북대학교 컴퓨터인공지능학부 부교수
〈관심분야〉 암호 알고리즘 설계 및 분석, 네트워크 및 시스템 보안



장 동 훈 (Donghoon Chang) 정회원
2021년 3월: 고려대학교 수학과 학사
2003년 2월: 고려대학교 정보보호대학원 석사
2008년 8월: 고려대학교 정보보호대학원 박사
2008년 8월~2008년 10월: 고려대학교 정보보호기술연구센터 연구교수
2008년 11월~2009년 9월: 미국 콜롬비아대학 포닥연구원
2009년 10월~2012년 5월: 미국 NIST 연구원
2012년 6월~현재: 인도 IIIT-Delhi 컴퓨터학과 부교수
2019년 5월~현재: 미국 NIST 연구원
〈관심분야〉 암호 알고리즘 설계 및 분석