Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.32, NO.3, Jun. 2022 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2022.32.3.465

AND 연산자 축적을 통한 경량 S-boxes 생성 방법*

전 용 진,[1†] 김 종 성[2‡]
[1,2]국민대학교 (대학원생, 교수)

How to Generate Lightweight S-Boxes by Using AND Gate Accumulation*

Yongjin Jeon,[1†] Jongsung Kim[2‡]
[1,2]Kookmin University (Graduate student, Professor)

요 약

코로나19의 영향으로 사람들은 편리함, 건강 등에 관심을 두게 되었고, 이를 도와주는 IoT 기기의 사용량은 늘어나고 있다. 리소스가 제한적이지만 민감한 정보를 다뤄야 하는 IoT 기기들에 경량 보안요소를 내장하기 위해서는 경량 S-box의 개발이 필수적이다. 2021년 이전까지 경량 4-bit S-box는 휴리스틱 방법으로 개발하고, 더 큰 크기의 경량 S-box는 확장구조 혹은 같은 연산을 반복하여 개발하는 것이 일반적이었다. 그러나 2022년 1월, MISTY 확장구조로 생성한 S-box보다 더 좋은 차분 균일성(Differential uniformity)과 선형성(Linearity)을 갖는 8-bit S-box를 찾을 수 있는 휴리스틱 알고리즘을 제안한 논문이 게재되었다[1]. 해당 논문에서 제안한 휴리스틱 알고리즘은 AND 연산자를 한 개씩 추가하면서 S-box를 생성하는데, AND 연산자를 추가할 때마다 차분 균일성을 계산하여 원하는 기준에 도달할 수 없는 S-box를 사전에 제거하는 방식을 사용한다. 본 논문에서는 이 휴리스틱 알고리즘의 성능을 향상한다. 차분 균일성뿐만 아니라 또 다른 차분성질을 사용하여 사전제거하는 양을 늘리고, 선형성을 계산하여 사전제거하는 프로세스를 추가함으로써 차분안전성뿐만 아니라 선형안전성까지도 동시에 만족할 수 있게 한다.

ABSTRACT

Due to the impact of COVID-19, people are paying attention to convenience and health, and the use of IoT devices to help them is increasing. In order to embed a lightweight security element in IoT devices that need to handle sensitive information even with limited resources, the development of a lightweight S-box is essential. Until 2021, it was common to develop a lightweight 4-bit S-box by a heuristic method, and to develop an extended structure or repeat the same operation for a larger size lightweight S-box. However, in January 2022, a paper that proposed a heuristic algorithm to find an 8-bit S-box with better differential uniformity and linearity than the S-box generated with an MISTY extended structure, although non-bijective, was published [1]. The heuristic algorithm proposed in this paper generates an S-box by adding AND operations one by one. Whenever an AND operation is added, they use a method that pre-removes the S-box for which the calculated differential uniformity does not reach the desired criterion. In this paper, we improve the performance of this heuristic algorithm. By increasing the amount of pre-removal using not only differential uniformity but also other differential property, and adding a process of calculating linearity for pre-removing, it is possible to satisfy not only differential security but also linear security.

Keywords: Lightweight S-box, Differential uniformity, Linearity, Multiplicative complexity

Received(02. 22. 2022), Modified(04. 25. 2022), Accepted(04. 26. 2022)

* 이 논문은 2022년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2017-0-00520, SCR-Friendly 대칭키 암호 및 응용모드 개발)

† 주저자, idealtop18@kookmin.ac.kr
‡ 교신저자, jskim@kookmin.ac.kr(Corresponding author)

I. 서 론

코로나19의 영향으로 사람들은 주거공간에 머무는 시간이 늘어나고 편리함과 건강, 생활필수품의 운송 방법에도 많은 관심을 두게 되었다. 이로 인해 스마트 가전제품 등을 활용한 스마트홈이 발달하고, PC와 스마트폰 같은 범용적 디지털기기 외에도 웨어러블이 가능한 의료 IoT, 운송 전용 무인 드론 등과 같이 기능과 리소스가 제한적인 IoT 기기들의 사용량이 늘어나고 있다. 사용자에게 밀접한 기기일수록 민감한 개인정보를 다루는데, 개인정보의 악용을 막기 위해서는 경량 보안요소를 반드시 내장해야 한다. 경량화되지 않는 보안요소의 탑재는 성능저하를 만들거나 급기야는 기기의 작동을 멈출 수 있다. 이러한 문제를 해결하기 위하여 경량암호의 개발은 필수적이다. 국제적인 지지를 받고 있으며 미국 보안표준을 다루는 NIST에서는 경량암호표준개발을 위한 Lightweight Cryptography (LWC) 프로젝트가 진행 중이며, 2021년 3월에 final round 후보들이 발표되었다[3,4].

S-box는 암호 설계에 필수적이며 중요한 요소이다. 중요성에 대한 예로, LWC 프로젝트에서 final round로 발표된 암호의 절반 이상이 S-box를 사용한다. S-box를 사용하는 암호의 암호학적 안전성은 S-box에 크게 기인한다. 특히, 암호분석법 중 차분분석과 선형분석에 사용되는 경로는 각각 S-box의 DDT(differential distribution table)와 LAT(linear approximation table)에 많은 영향을 받는다. 경로생성에는 각 표의 가장 큰 값(혹은 가장 큰 절댓값)을 주로 사용하며, 이는 각각 차분균일성(Differential uniformity)과 선형성(linearity)으로 불린다. S-box는 주로 연접해서 사용되는 경우가 많으므로 S-box의 bijective 성질은 활용 면을 극대화한다.

논리연산(AND, XOR 등)으로 생성되는 경량 S-box는 확장구조 혹은 같은 연산을 반복하여 생성하거나, 연산자를 휴리스틱 알고리즘을 통해 한 개씩 추가하여 생성하는 방법이 있다. 확장구조나 연산을 반복하여 생성하는 방법은 큰 크기의 S-box 생성이 쉬우나, 안전성의 한계가 존재한다[5]. 휴리스틱 알고리즘을 사용하는 방법은 큰 크기의 S-box 생성이 어려우나, 안전성과 효율성의 균형에 대한 섬세한 작업이 가능하다. 2021년 이전까지 큰 크기의 S-box는 휴리스틱 생성법이 없었으나, 2021년 김기윤 외 4인은 휴리스틱 방법에 강화학습을 도입해 MISTY 확장구조로 생성한 8-bit S-box와 같은 차분 균일성과 선형성을 갖는 8-bit S-box 생성에 성공했다[2]. 2022년 전용진 외 4인은 비록 bijective는 아니지만, 같은 선형성을 가지며 비선형연산자를 1개 줄이고 절반의 차분 균일성를 갖는 8-bit S-box를 생성하는 휴리스틱 알고리즘을 제안했다[1]. 해당 논문에서는 제안한 알고리즘으로 각 비선형연산자 개수에 대한 차분 균일성과 선형성의 하한도 제안했다.

[1]의 알고리즘은 AND 연산자를 한 개씩 추가하면서 S-box를 생성하는데, AND 연산자를 추가하기 전에 차분 균일성을 계산하여 원하는 기준에 도달할 수 없는 S-box를 사전에 제거하는 방식을 사용한다. 이는 선형분석관점에서의 안전성은 S-box 생성이 완료된 후 확인할 수 있다는 단점이 있다. 본 논문에서는 이를 보완하여 선형분석관점에서 S-box를 사전에 제거할 방법을 추가하고, 차분분석관점에서도 조건을 추가하여 더 많은 S-box를 제거하는 알고리즘을 제안한다.

본 논문은 2장에서는 S-box와 동치류, A-box 등 논리 전개에 필요한 사전지식을 소개한다. 3장에서는 향상된 경량 S-box 조사방법들을 소개하며, 4장에서는 알고리즘을 사용하여 실제로 S-box를 조사했을 때의 결과를 소개한다. 이후, 5장에서 결론을 짓는다.

II. 사전준비

2.1 S-box의 암호학적 성질

(n,m)-bit S-box는 다변수 부울함수 $S:\{0,1\}^n \rightarrow \{0,1\}^m$로 정의된다. 차분분석에 대한 S-box의 안전성에 대한 대표적인 지표로는 DDT와 차분 균일성 δ가 있다. DDT는 각 입출력차분쌍의 발생분포를 나타내는 표이며, 차분 균일성은 DDT에서 0 입출력차분을 제외할 때 가장 큰 값을 말한다. 이때의 0은 0을 연접한 적당한 비트열 $0\|\cdots\|0$을 뜻한다. 본 논문에서는 가독성의 이유로 숫자 0과 영비트열을 모두 0으로 표기한다. 여기서 Δa는 n-bit이고, Δb는 m-bit이다.

$$DDT_S(\Delta a,\Delta b) = \#\{x\in\{0,1\}^n : S(x)\oplus S(x\oplus\Delta a)=\Delta b\},$$
$$\delta(S) = \max_{\Delta a(\neq 0),\Delta b} DDT_S(\Delta a,\Delta b).$$

LAT와 선형성 λ는 선형분석에 대한 S-box의 안전성 지표이다. LAT는 각 입출력마스킹에 대한 선형방정식이 성립할 경우의 수의 편차를 말하며, 선형성은 LAT에서 0 입출력마스킹을 제외할 때 가장 큰 편차의 절댓값을 말한다.

$$LAT_S(\Lambda a,\Lambda b) = \#\{x\in\{0,1\}^n : x\cdot\Lambda a\oplus S(x)\cdot\Lambda b\}-2^{n-1},$$
$$\lambda(S)=\max_{\Lambda a,\Lambda b(\neq 0)}|LAT_S(\Lambda a,\Lambda b)|.$$

$n=m$인 (n,m)-bit S-box는 n-bit S-box라고 작성한다. 만약 n-bit S-box S의 치역이 정의역과 같다면 S를 bijective라 하고, bijective가 아니면 non-bijective라고 한다. 또한, bijective 성질은 다음과 같은 필요충분조건이 존재한다.

- S는 bijective이다.
- 0이 아닌 모든 n-bit 입력차분 Δa에 대해 $DDT_S(\Delta a,0)=0$이다.
- 0이 아닌 모든 n-bit 출력마스킹 Λb에 대해 $LAT_S(0,\Lambda b)=0$이다.

2.2 동치류에 대한 불변성질

차분 균일성과 선형성은 선형동치류(linear equivalent class)와 확장선형동치류(extended linear equivalent class)에 대해 보존된다. 선형동치와 확장선형동치의 정의는 다음과 같다.

정의 1. 두 다변수 부울함수 $F,G:\{0,1\}^n\rightarrow\{0,1\}^m$에 대해, 두 선형순열 $A:\{0,1\}^n\rightarrow\{0,1\}^n$과 $B:\{0,1\}^m\rightarrow\{0,1\}^m$가 $F(x)=(B\circ G\circ A)(x)$를 만족한다면 F와 G는 서로 선형동치라고 한다. 또한, 어떤 선형함수 $C:\{0,1\}^n\rightarrow\{0,1\}^m$에 대해

$$F(x)=(B\circ G\circ A)(x)\oplus C(x)$$

를 만족하면 F와 G는 서로 확장선형동치라고 한다.

동치의 변환은 각 동치에 해당하는 식으로 변환하는 것을 말한다. 예를 들어, 다변수 부울함수 G를 $F(x)=(B\circ G\circ A)(x)\oplus C(x)$꼴로 변환하는 것을 확장선형변환이라고 한다. 확장선형변환에 대해 DDT와 LAT는 아래와 같이 변형된다. 모든 선형순열은 행렬표현으로 유일하게 표현할 수 있는데, 아래첨자로 쓰인 B^T는 선형순열 B의 행렬표현과 전치인 행렬표현을 갖는 선형순열을 뜻하며, A^{-T}는 선형순열 A^{-1}의 행렬표현과 전치인 행렬표현을 갖는 선형순열을 뜻한다.

$$DDT_G(\Delta a,\Delta b)=DDT_F(A(\Delta a),B^{-1}(\Delta b)\oplus C(\Delta a))$$
$$LAT_G(\Lambda a,\Lambda b)=LAT_F(A^{-T}(\Lambda a\oplus C^T(\Lambda b)),B^T(\Lambda b))$$

위의 식을 통해 확장선형변환에서는 차분 균일성과 선형성이 보존된다는 것은 쉽게 알 수 있다. C를 0만 출력하는 상수함수로 취급하면 단순 선형변환이 되며, 마찬가지로 차분 균일성과 선형성이 보존된다. ($\delta_G=\delta_F$, $\lambda_G=\lambda_F$.) Bijective 성질은 선형변환에서는 보존되나 확장선형변환에서는 보존되지 않는다.

2.3 S-box와 A-box의 관계성

임의의 (n,m)-bit S-box는 $S(0)=0$인 S와 m-bit 벡터 v를 XOR하여 만들어진다. $S(0)=0$를 만족하는 S는 XOR 연산자와 AND 연산자만으로도 구현할 수 있다[6]. 이때, 모든 AND 연산자의 출력을 순서대로 연접한 함수를 A-box라고 정의한다[1]. 예를 들어, k개의 AND 연산자를 사용했다면 이때의 A-box는 입력 n-bit와 출력 k-bit를 갖는다. S의 암호학적 안전성은 비선형연산자인 AND 연산에 크게 기인하므로 A-box는 S-box의 안전성을 좌우한다.

AND 연산자는 2-bit를 입력받아 1-bit 출력을 하는 연산이다. S-box 구현 시 AND 연산자의 2-bit 입력값은 전체 S-box의 입력값 n-bit와 이전 AND 연산자 출력값들의 선형결합으로 이루어진다. 즉, S-box의 입력값을 x, AND 연산자 출력값을 처음부터 순서대로 $y_0,y_1,\cdots$ 라고 한다면 i번째 AND 연산자의 두 입력값은 x와 $y_0,y_1,\cdots,y_{i-2}$의 선형결합으로 이루어진다. 입력 2-bit는 각각 어떤 $(n+i)$-bit 벡터 b_{2i}와 b_{2i+1}을 통해 아래와 같은 식으로 표현할 수 있다.

$$b_{2i}\cdot(y_{i-2}\|y_{i-3}\|\cdots\|y_0\|x),$$
$$b_{2i+1}\cdot(y_{i-2}\|y_{i-3}\|\cdots\|y_0\|x).$$

선형결합의 표현에 사용되는 벡터 b_{2i}와 b_{2i+1}을 파트너 벡터(partner vector)라고 하고, 파트너 벡터를 순서대로 모은 쌍 $(b_0,b_1,\cdots)$을 파트너쌍(partner tuple)이라고 한다. 파트너쌍이 정해지면 모든 AND 연산자들의 입력값이 정해지는 것이므로 A-box의 값이 정해진다. 그러므로 A-box의 수학적 정의는 파트너쌍을 사용하여 다음과 같이 연역적으로 정의된다.

정의 2. (n,k)-bit S-box S_A의 입력과 출력을 $x=(x_{n-1},\cdots,x_0)\in\{0,1\}$, $y=(y_{k-1},\cdots,y_0)\in\{0,1\}$라고 하자. $2k$쌍인 파트너쌍 $(b_0,\cdots,b_{2k-1})$가 다음의 연역적 성질을 만족하면 S_A를 (n,k)-bit A-box라고 한다.

- $y_0=(b_0\cdot x)(b_1\cdot x)$,
- 1 이상이고, k보다 작은 자연수 i에 대해, $y_i=(b_{2i}\cdot(y_{i-1}\|\cdots\|y_0\|x))(b_{2i+1}\cdot(y_{i-1}\|\cdots\|y_0\|x))$.

Multiplicative complexity (MC)는 S-box를 XOR, AND, NOT 연산자만으로 구현했을 때 필요한 AND 연산자의 최소 개수를 뜻한다. MC는 상수 XOR 및 NOT 연산에 대해 불변하므로[7] $S(0)=0$을 만족하는 S-box S만을 고려한다.

임의의 S-box는 MC 개수의 AND 연산자를 사용한 구현 방법이 존재한다. A-box 출력값의 크기는 AND 연산자의 개수와 같으므로 이런 구현 방법에서의 A-box는 MC와 같은 크기의 출력값을 갖는다. A-box를 제외하면 전부 XOR 연산자로 이루어져 있으므로 행렬로 표현된다. MC가 k이고 $S(0)=0$인 (n,m)-bit S-box S는 다음을 만족하는 $m\times k$ 기약다항식꼴행렬 M, $m\times m$ 가역행렬 D, $m\times n$ 행렬 C, (n,k)-bit A-box S_A를 갖는다[1].

$$S=D\circ(M\circ S_A\oplus C).$$

이때, S는 $M\circ S_A$와 확장선형동치이며, $M\circ S_A\oplus C$와는 선형동치인 것은 자명하게 알 수 있다. $S=M\circ S_A\oplus C$의 DDT와 LAT는 다음과 같다.

$$\begin{aligned}DDT_S(\Delta a,\Delta b)&=DDT_{M\circ S_A\oplus C}(\Delta a,\Delta b)\\&=DDT_{M\circ S_A}(\Delta a,\Delta b\oplus C(\Delta a)),\end{aligned}$$

$$\begin{aligned}LAT_S(\Lambda a,\Lambda b)&=LAT_{M\circ S_A\oplus C}(\Lambda a,\Lambda b)\\&=LAT_{M\circ S_A}(\Lambda a\oplus C^T(\Lambda b),\Lambda b).\end{aligned}$$

III. 향상된 경량 S-box 생성 방법

3.1 부분 A-box와 부분 행렬

S가 $D\circ(M\circ S_A\oplus C)$로 표현될 때, S와 $M\circ S_A$는 서로 확장선형동치이므로 차분 균일성과 선형성이 같다. 그러므로 $M\circ S_A$가 적절한 차분 균일성과 선형성을 만족하는 S_A와 M을 생성하는 것으로 S-box의 조사를 대체할 수 있다. 적절한 C까지 생성하여 $M\circ S_A\oplus C$를 구하면 bijective 성질도 얻을 수 있다.

k개의 AND 연산자를 사용하여 S-box를 생성한다고 가정하자. A-box의 크기는 (n,k)-bit가 된다. (n,k)-bit A-box S_A의 출력을 $(y_{k-1},y_{k-2},\cdots,y_0)$로 표기하자. A-box의 정의에 따라 (n,k)-bit A-box를 만들기 위해서는 첫 번째 AND 연산자로 생성되는 y_0부터 순서대로 생성해야 한다. A-box 생성과정에서 나타나는 A-box들을 부분 A-box라고 부르고 다음과 같이 정의한다.

$$\begin{aligned}&S_A^i:\{0,1\}\to\{0,1\},\\&S_A^1=y_0,\\&S_A^2=(y_1,y_0),\\&\cdots,\\&S_A^k=(y_{k-1},y_{k-2},\cdots,y_0)=S_A.\end{aligned}$$

$k<m$이면 [1]의 정리 5에 의해 최종 S-box의 선형성이 2^n가 되므로 선형분석에 저항성을 갖지 못한다. 그러므로 $k\geq m$을 가정한다. $k\times m$ 기약행사다리꼴행렬 M을 생각해보자. 제일 아래의 행을 0번째 행이라 하고, 제일 오른쪽의 열을 0번째 열이라고 할 때, i번째 행과 j번째 열에 있는 항을 $M_{i,j}$라고 하자. 기약행사다리꼴행렬의 성질로 인해, $i<j+m-k$일 때 $M_{i,j}=0$이다. 이제, $M\circ S_A=(z_{m-1},z_{m-2},\cdots,z_0)$라고 하자. z_i는 다음과 같은 형태를 보인다.

$$z_i=M_{i,0}y_0\oplus M_{i,1}y_1\oplus\cdots\oplus M_{i,k-m-1+i}y_{k-m-1+i}.$$

즉, z_i는 y_{k-m+i}부터 y_{k-1}의 값에 영향을 받지 않는다. 그러므로 부분 A-box인 S_A^{k-m+i}을 생성하고, M의 밑에서부터 $i+1$개의 행을 생성하면 z_0부터 z_i까지의 값을 구할 수 있다. M의 밑에서부터 i개의 행만 추출한 $k\times i$ 부분행렬을 M^i라 정의하자. $M^m=M$임은 쉽게 알 수 있다.

이제 부분 A-box S_A^{k-m+i}와 부분행렬 M^i를 합성한 $M^i\circ S_A^{k-m+i}$를 S^i라고 하자. S^i는 (n,i)-bit S-box이며 다음과 같은 성질이 존재한다.

$$DDT_{S^i}(\Delta a,\Delta b)=DDT_{S^{i+1}}(\Delta a,0\|\Delta b)+DDT_{S^{i+1}}(\Delta a,1\|\Delta b),$$
$$LAT_{S^i}(\Lambda a,\Lambda b)=LAT_{S^{i+1}}(\Lambda a,0\|\Lambda b).$$

위의 성질을 통해 우리는 다음의 성질도 얻을 수 있다.

$$\frac{1}{2}\delta(S^i)\leqq\delta(S^{i+1})\leqq\delta(S^i),\lambda(S^i)\leqq\lambda(S^{i+1}).$$

이를 통해 차분 균일성 δ와 선형성 λ를 만족하는 S는 다음의 식을 만족한다.

$$\frac{1}{2^{m-1}}\delta(S^1)\leqq\frac{1}{2^{m-2}}\delta(S^2)\leqq\cdots\leqq\delta(S^m)=\delta(S)=\delta,$$
$$\lambda(S^1)\leqq\lambda(S^2)\leqq\cdots\leqq\lambda(S^m)=\lambda(S)=\lambda.$$

따라서 i를 증가시키면서 다음 두 식을 만족하지 않는 M^i와 S_A^{k-m+i}를 사전에 제거한다면 더 빠른 조사가 가능하다.

$$\delta(S^i)=\delta(M^i\circ S_A^{k-m+i})\leqq 2^{m-i}\delta, \quad (1)$$
$$\lambda(S^i)=\lambda(M^i\circ S_A^{k-m+i})\leqq\lambda. \quad (2)$$

A-box를 생성하는 중에는 S_A^i의 파트너쌍에 2개의 파트너 벡터들을 추가하여 (즉, 1개의 AND 연산자를 추가하여) S_A^{i+1}을 생성하는데, 이를 S_A^i를 S_A^{i+1}로 확장한다고 한다. 또, M^i의 위에 한 행을 추가하여 M^{i+1}을 생성하는 것을 M^i를 M^{i+1}로 확장한다고 한다.

3.2 A-box 전수조사

A-box는 작은 크기의 A-box로 큰 크기의 A-box를 생성할 수 있는 성질이 있다. 작은 크기의 A-box를 미리 전수조사하고 조사된 A-box들의 크기를 늘려 다양한 크기를 갖는 A-box로 만든다면, 더 빠른 조사를 할 수 있다. A-box의 크기를 늘리는 방법은 입력 크기를 늘리는 방법과 출력 크기를 늘리는 방법 두 가지가 있다. A-box의 출력 크기를 늘리는 방법은 전 장에서 다룬 바와 같이 AND 연산자의 개수를 늘리는 방식으로 A-box를 확장하는 것이다. 이와 반대로, 파트너 벡터에 값을 추가하여 A-box의 입력 크기를 늘리는 방법이 있다.

파트너 벡터들은 A-box 입력에 관여되는 n개의 성분과 A-box 출력에 관여되는 나머지 성분들을 갖는다. 이 성분들 사이에 j-bit 0 비트열을 주입하면 크기가 $(n+j)$-bit인 A-box의 파트너 벡터가 된다. 이때, y_0를 생성하는 파트너 벡터 b_0와 b_1은 출력에 관여되는 성분이 없으므로 입력에 관여되는 성분의 MSB (Most Significant Bit)에 추가하여 $0\|b_0$과 $0\|b_1$로 생성한다. 이렇게 0 비트열을 주입하여 A-box의 입력 크기를 늘리는 것을 파트너 벡터를 확장한다고 한다.

작은 크기라도 A-box를 모두 전수조사하는 것은 어려우므로 동치류를 구한다. A-box의 동치류 중 A-box의 출력에 행렬을 연산하는 방식의 동치류 생성은 M과의 상호작용으로 S-box 생성에 장애가 될 수 있다. 그러므로 A-box의 전수조사에는 입력선형동치를 사용한다. 두 (n,k)-bit A-box S_A와 S_A'이 어떤 선형순열 L에 대해 $S_A=S_A'\circ L$을 만족하면 S_A와 S_A'은 서로 입력선형동치라고 한다. 입력선형동치인 S_A와 S_A'은 다음과 같은 관계를 갖는다.

정리 1. (n,k)-bit A-box S_A와 선형순열 $L:\{0,1\}\rightarrow\{0,1\}$에 대해, $S_A'=S_A\circ L$를 만족하는 A-box S_A'를 생각하자. 만약 S_A의 파트너 쌍이 $(b_{2k-1},b_{2k-2},\cdots,b_0)$이면, 다음을 만족하는 $(b_{2k-1}',b_{2k-2}',\cdots,b_0')$는 S_A'의 파트너 쌍이 된다.

$$\begin{cases}b_{2i}'=L_i(b_{2i}),\\ b_{2i+1}'=L_i(b_{2i+1}).\end{cases}$$

이때, $0 \le i < k$인 i에 대해서 $L_i = \begin{pmatrix} I_i & 0 \\ 0 & L \end{pmatrix}$이고, I_i는 $i \times i$인 항등행렬이다[1].

정리 1의 b_{2i}'와 b_{2i+1}'은 다음과 같은 꼴을 갖는다.

$$\begin{aligned} b_{2i}' &= L_i(b_{2i}^{n+i-2}, b_{2i}^{n+i-3}, \cdots, b_{2i}^{0}) \\ &= b_{2i}^{n+i-2} \oplus b_{2i}^{n+i-3} \oplus \cdots \oplus b_{2i}^{n} \oplus L(b_{2i}^{n-1}, \cdots, b_{2i}^{0}), \\ b_{2i+1}' &= L_i(b_{2i+1}^{n+i-2}, b_{2i+1}^{n+i-3}, \cdots, b_{2i+1}^{0}) \\ &= b_{2i+1}^{n+i-2} \oplus b_{2i+1}^{n+i-3} \oplus \cdots \oplus b_{2i+1}^{n} \oplus L(b_{2i+1}^{n-1}, \cdots, b_{2i+1}^{0}). \end{aligned}$$

$n > 2k$라고 가정하자. 그러면 서로 일차독립인 $b_0, b_1, \cdots, b_{2k-1}$에 대해, 다음과 같은 규칙을 갖는 L을 정할 수 있다.

$$\begin{aligned} &L(b_0^{n-1}, \cdots, b_0^0) = (0,0,\cdots,0,0,0,0,0,0,1), \\ &L(b_1^{n-1}, \cdots, b_1^0) = (0,0,\cdots,0,0,0,0,0,1,0), \\ &L(b_2^{n-1}, \cdots, b_2^0) = (0,0,\cdots,0,0,0,0,1,0,0), \\ &L(b_3^{n-1}, \cdots, b_3^0) = (0,0,\cdots,0,0,0,1,0,0,0), \\ &\cdots, \\ &L(b_{2k-2}^{n-1}, \cdots, b_{2k-2}^0) = (0,0,\cdots,0,0,1,0,\cdots,0), \\ &L(b_{2k-1}^{n-1}, \cdots, b_{2k-1}^0) = (0,0,\cdots,0,1,0,0,\cdots,0). \end{aligned}$$

이러한 L로 만들어진 A-box의 모든 파트너 벡터들은 $(n-2k)$-bit MSB 값이 0이 된다. 만약 $b_0, b_1, \cdots, b_{2k-1}$이 서로 일차종속이라면 더 많은 비트의 값을 0으로 만들 수 있다. 이렇게 만들어진 0 성분들을 모두 제거하고 남은 성분들만 모은 벡터들은 $(2k,k)$-bit A-box의 파트너 벡터가 된다. 그러므로 $n > 2k$이면 n의 크기에 상관없이 (n,k)-bit A-box의 입력선형동치류들은 모두 $(2k,k)$-bit A-box의 입력선형동치류들과 일대일 대응 관계가 된다.

만약 b_{2k-2}와 b_{2k-1}로 생성되는 k번째 AND 연산자의 출력값 y_{k-1}이 A-box의 입력값 x와 이전 AND 연산자들의 출력값인 $y_{k-2}, y_{k-3}, \cdots, y_0$의 선형결합으로 만들어질 수 있으면 사전에 제거한다. 이러한 출력값은 AND 연산자의 필요 없이 XOR 연산자만으로 생성할 수 있기 때문이다. 이 과정으로 인해, A-box는 A-box의 MC과 같은 개수의 AND 연산자만 사용하게 된다.

본 논문에서는 (2,1)-bit, (4,2)-bit A-box를 조사하고, 이를 통해 (4,3)-bit A-box, (5,3)-bit A-box, (6,3)-bit A-box의 전수조사를 진행했다. Table 1.은 그에 대한 표이다.

Table 1. Number of representative elements of A-box

(n,k)	(2,1)	(4,2)	(4,3)	(5,3)	(6,3)
#A-boxes	1	14	2686	3375	3401

(2,1)-bit A-box는 특별한 경우로, 부울 함수 $(x_1,x_0) \mapsto x_0x_1$의 동치류만 존재한다. 아래는 (4,2)-bit A-box 대표원소들의 예시이다. 단, $y_0 = x_0x_1$이다.

$$\begin{aligned} &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, x_0x_2), \\ &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, (x_1 \oplus x_0)x_2), \\ &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, x_2(x_2 \oplus x_0)), \\ &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, x_2(x_2 \oplus x_1 \oplus x_0)), \\ &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, x_2x_3), \\ &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, x_2y_0), \\ &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, x_2(y_0 \oplus x_0)), \\ &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, x_2(y_0 \oplus x_1 \oplus x_0)), \\ &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, x_2(y_0 \oplus x_2)), \\ &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, x_2(y_0 \oplus x_2 \oplus x_0)), \\ &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, x_2(y_0 \oplus x_2 \oplus x_1 \oplus x_0)), \\ &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, x_2(y_0 \oplus x_3)), \\ &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, (y_0 \oplus x_1 \oplus x_0)(y_0 \oplus x_2)), \\ &(x_3,x_2,x_1,x_0) \mapsto (x_0x_1, (y_0 \oplus x_2)(y_0 \oplus x_3)). \end{aligned}$$

3.3 A-box의 차분성

S-box의 차분 균일성은 A-box의 차분 균일성보다 작아질 수 없다[1]. 그러므로 행렬 M^i를 조사하기 전에 A-box S_A^i의 차분 균일성을 먼저 검사한다. 식 (1)의 성질을 사용해 S_A^i의 차분 균일성이 $2^{m-i}\delta$이하인지 검사하고 $2^{m-i}\delta$보다 크다면 제거한다. 만약 i가 작아서 $2^{m-i}\delta \geqq 2^n$를 만족하면 식(1)을 항상 만족하므로 검사는 건너뛴다.

또한, S_A^i와 S_A^i로 확장된 S_A^{i+1}의 DDT는 다음과 같은 관계를 갖는다.

$$DDT_{S_A^i}(\Delta a,\Delta b) \quad (3)$$
$$= DDT_{S_A^{i+1}}(\Delta a,0\|\Delta b) + DDT_{S_A^{i+1}}(\Delta a,1\|\Delta b)$$

$y_i = (b_{2i} \cdot (S_A^i\|x)) \cdot (b_{2i+1} \cdot (S_A^i\|x))$를 S_A^{i+1}의 새로운 출력값이라고 하자. 식 (3)에서 Δb의 앞에 붙는 1-bit 값은 y_i의 차분을 뜻한다. $S_A(x)\oplus S_A(x\oplus\Delta a)=\Delta b$를 만족하는 x에 대해 y_i의 Δa 차분은 다음과 같이 주어진다.

$$\begin{aligned}
&y_i(x)\oplus y_i(x\oplus\Delta a)\\
&= (b_{2i}\cdot S_A^i(x)\|x)(b_{2i+1}\cdot S_A^i(x)\|x)\\
&\quad\oplus(b_{2i}\cdot S_A^i(x\oplus\Delta a)\|(x\oplus\Delta a))\\
&\qquad(b_{2i+1}\cdot S_A^i(x\oplus\Delta a)\|(x\oplus\Delta a))\\
&= (b_{2i}\cdot S_A^i(x)\|x)(b_{2i+1}\cdot S_A^i(x)\|x)\\
&\quad\oplus(b_{2i}\cdot(S_A^i(x)\|x)\oplus b_{2i}\cdot(\Delta b\|\Delta a))\\
&\qquad(b_{2i+1}\cdot(S_A^i(x)\|x)\oplus b_{2i+1}\cdot(\Delta b\|\Delta a))\\
&= (b_{2i}\cdot S_A^i(x)\|x)(b_{2i+1}\cdot\Delta b\|\Delta a)\\
&\quad\oplus(b_{2i}\cdot\Delta b\|\Delta a)(b_{2i+1}\cdot S_A^i(x)\|x)\\
&\quad\oplus(b_{2i}\cdot\Delta b\|\Delta a)(b_{2i+1}\cdot\Delta b\|\Delta a)
\end{aligned}$$

위 식을 통해 다음 식을 만족하면 y_i의 차분도 0이 됨을 쉽게 알 수 있다.

$$b_{2i}\cdot\Delta b\|\Delta a = b_{2i+1}\cdot\Delta b\|\Delta a = 0.$$

즉, b_{2i}와 b_{2i+1}이 $\Delta b\|\Delta a$와 직교한다면 y_i의 Δa 차분은 0이 된다. 이러한 Δa와 Δb에 대해 DDT는 다음과 같은 식을 만족한다.

$$DDT_{S_A^i}(\Delta a,\Delta b) = DDT_{S_A^{i+1}}(\Delta a,0\|\Delta b). \quad (4)$$

$2^{m-i-1}\delta < DDT_{S_A^i}(\Delta a,\Delta b)$를 만족하는 Δa와 Δb에 대해, 새로 만들어진 파트너 벡터 b_{2i}와 b_{2i+1}가 $\Delta b\|\Delta a$와 직교한다면 S_A^{i+1}의 차분 균일성은 $2^{m-i-1}\delta$보다 크므로 제거된다. 따라서 이러한 파트너 벡터들은 조사하지 않는다.

3.4 A-box의 선형성

식 (2)와 같이 최종 S-box S의 선형성이 λ이기 위해서는 S^i의 선형성도 λ여야 한다. S^i의 LAT는 아래와 같은 관계식이 존재한다.

$$\begin{aligned}
&LAT_{S^i}(\Lambda a,\Lambda b)\\
&= LAT_{M^i\circ S_A^{k-m+i}}(\Lambda a,\Lambda b)\\
&= LAT_{S_A^{k-m+i}}(\Lambda a,(M^i)^T(\Lambda b)).
\end{aligned}$$

즉, S^i의 LAT는 S_A^{k-m+i}의 LAT의 일부분이다. M^i에 따라 LAT의 원하는 행을 선택할 수 있으므로 λ을 넘는 값을 모두 제거하는 M^i를 선택한다.

3.2장에서 전수조사했던 바와 같이, 어떠한 A-box를 선택하든 y_0는 $(x_0,x_1)\mapsto x_0x_1$과 같은 동치류에 속한다. $1\times(k-1)$ 행렬인 M^1이 다음과 같은 형태임을 상기하자.

$$M^1 = (0\ \cdots\ 0\, M_{0,k-m-1} M_{0,k-m-2}\ \cdots\ M_{0,0})$$

만약 $M_{0,k-m-1}=M_{0,k-m-2}=\cdots=M_{0,1}=0$이라면, $S^1=z_0$는 $M_{0,0}y_0$와 같으므로 $M_{0,0}$에 따라 0이거나 y_0이다. 이때의 선형성은 각각 2^n이거나 2^{n-1}이다. $\lambda(S^1)\le\lambda(S)$이므로 2^{n-1}보다 낮은 선형성을 얻을 수 없다. 그러므로 $M_{0,k-m-1},M_{0,k-m-2},\cdots,M_{0,1}$ 중 하나는 0이 아니라는 조건을 추가한다면 2^{n-1}보다 더 낮은 선형성을 갖는 S-box를 찾을 수 있다.

3.5 S-box 조사 알고리즘

k개의 AND 게이트를 사용하고, 차분 균일성이 δ이고, 선형성이 λ인 (n,m)-bit S-box를 구한다고 하자. $S^i=M^i\circ S_A^{k-m+i}$에 대해, $\delta(S^i)\le 2^{m-i}\delta$과 $\lambda(S^i)=\lambda$를 만족하는 M^i와 S_A^{k-m+i}가 주어졌을 때, M^{i+1}과 $S_A^{k-m+i+1}$의 집합을 구하는 알고리즘은 다음과 같다. 작성의 편의상 $r=k-m$으로 치환한다.

1. A-box를 확장하기 위하여 파트너 벡터의 쌍 $(b_{2r+2i},b_{2r+2i+1})\in\{0,1\}^{n+r+i}\times\{0,1\}^{n+r+i}$의 집합을 생성한다.

1-1. $b_{2r+2i}\ge b_{2r+2i+1}$이면 사전에 제거한다.

1-2. S_A^{r+i}의 마지막 파트너 벡터의 쌍 $(b_{2r+2i-2},b_{2r+2i-1})$에 대해, $b_{2r+2i-2}\ge b_{2r+2i}$이고 $b_{2r+2i-1}\ge b_{2r+2i+1}$이면 사전에 제거한다.

1-3. 식 (4)를 만족하는 Δa와 Δb에 대해 $\Delta b \| \Delta a$와 직교하는 파트너 벡터는 사전에 제거한다.

1-4. 새로운 파트너 벡터로 생성한 출력이 S_A^{r+i} 입출력의 선형결합으로 생성될 수 있으면 제거한다.

2. $\delta(S_A^{k-m+i+1}) > 2^{m-i+1}\delta$를 만족하는 $S_A^{k-m+i+1}$를 제거한다.

3. M^i를 확장하여 $\lambda(M^{i+1} \circ S_A^{k-m+i+1}) = \lambda$를 만족하는 M^{i+1}를 만든다.

4. $\delta(M^{i+1} \circ S_A^{k-m+i+1}) > 2^{m-i+1}\delta$를 만족하는 $(M^{i+1}, S_A^{k-m+i+1})$쌍을 제거한다.

5. 남은 $(M^{i+1}, S_A^{k-m+i+1})$쌍을 저장한다.

위의 알고리즘에서 1-1번, 2번, 5번 단계만 취하면 [1]의 알고리즘이 된다. 또한, [1]의 알고리즘은 안전성 하한에 맞는 S-box만을 생성할 수 있었지만, 본 논문의 알고리즘은 그렇지 않은 S-box까지도 생성할 수 있다.

[1]의 알고리즘은 A-box가 완성되고 나서 M을 생성한다. [1]와의 알고리즘 성능 비교를 위해 M^i와 S_A^{k-m+i}를 동시에 생성하는 관점에서 본다. [1]에서는 M^i에 대한 제약이 없으므로 각 S_A^{k-m+i}에 대해 모든 경우를 생성한다. Table 2.는 $k=8$, $n=m=6$, $\delta=2$, $\lambda=8$인 경우에 대해 S^i의 개수를 비교한 것이다. 이때 S_A^3는 Table 1.에서 사전조사한 A-box를 사용한다. 본 논문의 방법으로 20배 이상의 검색공간 감소 효율을 보이는 것을 알 수 있다.

Table 2. The number of (S_A^{2+i}, M^i) with $k=8$, $n=m=6$, $\delta=2$, $\lambda=8$.

	$\#S_A^3$	$\#(S_A^3, M^1)$	$\#(S_A^4, M^2)$
This paper	3 401	1 223	16 855 779
[1]	3 401	23 807	268 423 969

본 논문에서는 다음과 같은 세 개의 경우에 대해 조사를 진행했으며, 그 결과는 4장에 작성되어 있다.

- $k=8$, $n=m=6$, $\delta=2$, $\lambda=8$
- $k=8$, $n=m=7$, $\delta=8$, $\lambda=32$
- $k=8$, $n=m=8$, $\delta=16$, $\lambda=64$

Table 3. Existed 6-bit S-box

Reference of S-box	differential uniformity	linearity	#AND gates
SPEEDY[8]	8	12	-
[1]	4	8	7
[9]	4	8	9
SC2000[10]	4	8	-
x^3[11]	2	8	9
Dillon's[12]	2	8	-
FIDES[13]	2	8	-

IV. 각 크기별 S-box 조사

4.1 6-bit S-box

Table 3.은 기존에 발표된 6-bit S-box의 성질을 보여준다. AND 연산자의 개수가 제시되어있지 않은 항목은 '-'라고 표시했다.

[1]는 7개의 AND 연산자를 사용하며 차분 균일성이 4를 만족하는 S-box를 발견했다. AND 연산자가 하나 늘었을 때, 차분 균일성은 최대 절반으로 감소하므로 8개를 사용했을 때는 2가 될 가능성이 존재한다. 6-bit S-box의 선형성은 [14]에서 제시하는 논리적인 하한에 따르면 최소 6이 될 수 있지만, 실제 발견된 S-box들의 선형성을 보아 8이 최소값임을 추정할 수 있다. 8개의 AND 연산자를 사용하는 6-bit S-box를 조사하므로 $k=8$, $n=m=6$, $\delta=2$, $\lambda=8$이다. 조사의 단계는 다음과 같다.

- 3.2장에서 전수조사한 각 S_A^3에 대해, $\lambda(M^1 \circ S_A^3) = 8$을 만족하는 M^1을 조사하여 (S_A^3, M^1)쌍을 저장한다.
- 각 i에 대해 랜덤하게 (S_A^{2+i}, M^i)쌍을 선택하여 3.5장의 알고리즘을 반복한다.
- (옵션) $M^6 \circ S_A^8 \oplus C$가 bijective가 되는 C를 조사한다.

해당 조건을 만족하는 S-box는 찾지 못했다. 이 결과를 통해 6-bit S-box는 차분 균일성이 2가 되기 위해서는 최소 9개의 AND 연산자를 사용해야

Table 4. 7-bit S-box

Reference of S-box	differential uniformity	linearity	#AND gates
This Paper	8	32	8
WAGE[15]	8	20	-
[9]	8	16	11
[1]	4	16	10
MISTY[16]	2	8	-
KASUMI[17]	2	8	-

한다고 추정할 수 있다.

4.2 7-bit S-box

Table 4.는 기존에 발표된 7-bit S-box의 성질을 보여준다. AND 연산자의 개수가 제시되어있지 않은 항목은 '-'라고 표시했으며, 본 논문의 조사를 통해 발견한 S-box를 추가하였다.

[1]는 7개의 AND 연산자를 사용하며 차분 균일성이 16를 만족하는 S-box를 발견했다. 그러므로 8개의 AND 연산자를 사용하여 8의 차분 균일성을 갖는 S-box를 찾는 것을 목표로 한다. 발견된 S-box들의 최소 선형성은 8이며, [14]에서 제시하는 선형성의 하한값과 같다. 조사방법은 6-bit S-box와 비슷하나 첫 번째 단계에 파트너 벡터의 확장이 추가된다. 본 장에서는 $k=8$, $n=m=7$, $\delta=8$, $\lambda=32$의 조사단계를 묘사한다. 조사의 단계는 다음과 같다.

- 3.2장에서 전수조사한 각 (6,3)-bit A-box의 파트너 벡터를 확장하여 (7,3)-bit A-box를 만든다. 이를 S_A^3이라고 한다.
- 각 S_A^3에 대해, $\lambda(M^2 \circ S_A^8)=32$을 만족하는 M^2을 조사하여 (S_A^3, M^2)쌍을 저장한다.
- 각 i에 대해 랜덤하게 (S_A^{1+i}, M^i)쌍을 선택하여 3.5장의 알고리즘을 반복한다.
- (옵션) $M^7 \circ S_A^8 \oplus C$가 bijective가 되는 C를 조사한다.

본 논문에서는 차분 균일성이 8이고, 선형성이 32이고 non-bijective인 7-bit S-box를 찾았다.

Table 5. Existed 8-bit S-box

Reference of S-box	differential uniformity	linearity	#AND gates
SKINNY[19]	64	64	8
Fantomas[20]	16	32	11
[1]	8	32	10
AES[18]	4	16	32

4.3 8-bit S-box

8-bit S-box는 블록암호 AES[18]를 비롯하여 SKINNY[19], Fantomas[20]등 수많은 블록암호에서 사용되는 크기이다. [14]에서 제시하는 선형성의 하한은 12이나, 현존하는 S-box들의 최소값은 16이다. 기존에 발표된 8-bit S-box는 Table 5.와 같다.

[1]는 7개의 AND 연산자를 사용하여 차분 균일성이 32를 만족하는 S-box를 발견했다. 8개의 AND 연산자를 사용했을 때, 차분 균일성 16을 달성할 수 있는지는 알려지지 않았다. 이때의 선형성은 최소 64이다. $k=8$, $n=m=8$, $\delta=16$, $\lambda=64$이며, 방법은 이전 장에서 묘사한 방법과 유사하다.

해당 조건을 만족하는 S-box는 찾지 못했다. 이를 통해, 8-bit S-box는 차분 균일성이 16이 되기 위해서는 최소 9개의 AND 연산자를 사용해야 한다고 추정할 수 있다.

V. 결 론

본 논문에서는 A-box와 S-box의 관계성을 살펴보고 이를 통해 경량 S-box를 생성하는 방법을 제안한다. 임의의 S-box는 $S(0)=0$을 만족하는 S-box S와 XOR 동치이며, S는 A-box S_A와 기약행사다리꼴행렬 M으로 이루어진 $M \circ S_A$와 확장 선형동치를 이룬다. 동치류의 특성상 S-box의 차분 균일성과 선형성은 $M \circ S_A$와 같다. 그러므로 S-box의 차분 균일성은 A-box의 차분 균일성보다 작을 수 없으며, A-box의 LAT로 S-box의 선형성을 계산할 수 있다.

S_A와 $M \circ S_A$를 다음과 같이 표기하자.

$$S_A=(y_{k-1}, y_{k-2}, \cdots, y_0),$$

$M \circ S_A = (z_{m-1}, z_{m-2}, \cdots, z_0)$.

A-box는 y_0부터 순차적으로 생성되며, 기약행사다리꼴행렬의 특성으로 z_i는 y_{k-m+i}부터 y_{k-1}의 값에 영향을 받지 않는다. 이로 인해 A-box의 생성과정 중 $M \circ S_A$의 일부를 계산할 수 있으므로 $M \circ S_A$의 부분적인 DDT와 LAT를 생성하여 불가능한 조건을 갖는 쌍을 사전에 제거한다. 만약 δ의 차분 균일성과 λ의 선형성을 갖는 S-box를 원한다면, z_i가 생성될 때 $2^{m-i}\delta$의 차분 균일성과 λ의 선형성을 가져야 한다. 이 외에도 차분의 특성을 사용한 추가적인 방법으로 더 많은 쌍을 사전에 제거할 수 있다.

또한, 본 논문에서는 제안한 방법으로 8개의 AND 연산자를 사용하여 6,7,8-bit 크기의 S-box를 조사하였다. 발견된 7-bit S-box는 같은 차분 균일성을 갖는 기존에 발견된 S-box보다 적은 AND 연산자를 사용하는 것을 볼 수 있다. 반면, 선형성은 더 높으며 non-bijective 특성을 갖는다. 6,8-bit 크기의 S-box는 기존보다 더 좋은 안전성을 갖는 S-box를 찾지 못했다.

본 논문에서는 3개의 AND 연산자를 사용하는 A-box를 전수조사하고, 이것을 기반으로 8개의 AND 연산자를 사용하는 S-box를 조사하였다. 전수조사가 가능한 AND 연산자의 개수를 늘린다면, 더 많은 AND 연산자를 사용하는 S-box를 조사할 수 있을 것이다.

References

[1] Y. Jeon, S. Baek, H. Kim, G. Kim, and J. Kim, "Differential uniformity and linearity of S-boxes by multiplicative complexity," Cryptography and Communications (2022), Jan. 2022.

[2] G. Kim, H. Kim, Y. Heo, Y. Jeon, and J. Kim, "Generating Cryptographic S-Boxes Using the Reinforcement Learning," IEEE Access, vol. 9, pp. 83092-83104, Jun. 2021.

[3] K. McKay, L. Bassham, M. Sönmez Turan, and N. Mouha, "Report on lightweight cryptography," NISTIR 8114, Mar. 2017.

[4] M. S. Turan, K. McKay, D. Chang, C. Calik, L. Bassham, J. Kang, and J. Kelsey, "Status Report on the Second Round of the NIST Lightweight Cryptography Standardization Process," NISTIR 8369, Jul. 2021.

[5] A. Canteaut, S. Duval, and G. Leurent, "Construction of lightweight S-boxes using Feistel and MISTY structures," SAC'15, LNSC 9566, pp. 373-393, 2015.

[6] P. Zajac, and M. Jókay, "Multiplicative complexity of bijective 4×4 S-boxes," Cryptography and Communications, vol. 6, pp. 255-277, May 2014.

[7] M. Turan Sönmez, and R. Peralta, "The multiplicative complexity of Boolean functions on four and five variables," LightSec'14, LNSC 8898, pp. 21-33, Mar. 2015.

[8] G. Leander, T. Moos, A. Moradi, and S. Rasoolzadeh, "The SPEEDY Family of Block Ciphers," Engineering an Ultra Low-Latency Cipher from Gate Level for Secure Processor Architectures, vol. 2021(4), pp. 510-545, Aug. 2021.

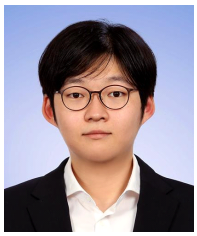
[9] H. Kim, Y. Jeon, G. Kim, J. Kim, B. Sim, D. Han, H. Seo, S. Kim, S. Hong, J. Sung, and D. Hong, "A new method for designing lightweight S-boxes with high differential and linear branch numbers, and its application," IEEE Access, vol. 9, pp. 150592-150607, Oct. 2021.

[10] T. Shimoyama, H. Yanami, K. Yokoyama, M. Takenaka, K. Itoh, J. Yajima, N. Torii, and H. Tanaka, "The block cipher SC2000," FSE'00, LNCS 2355, pp. 312-327, Jun. 02.

[11] B. Bilgin, L. De Meyer, S. Duval, I. Levi, and F. X. Standaert, "Low AND depth and efficient inverses: a guide on s-boxes for low-latency masking," IACR Transactions on Symmetric Cryp

tology, vol. 2020(1), pp. 144-184, May 2020.
[12] K. A. Browning, J. F. Dillon, M. T. McQuistan, and A. J. Wolfe, "An APN permutation in dimension six," Finite Fields: theory and applications, vol. 518, pp. 33-42, 2010
[13] B. Bilgin, A. Bogdanov, M. Knežević, F. Mendel, and Q. Wang, "Fides: Lightweight authenticated cipher with side-channel resistance for constrained hardware," CHES'13, pp. 142-158, Aug. 2013
[14] C. Carlet and C. Ding, "Nonlinearities of S-boxes," Finite fields and their applications, vol. 13(1), pp. 121-135, Jan. 2007
[15] M. Aagaard, R. AlTawy, G. Gong, K. Mandal, R. Rohit, and N. Zidaric, "WAGE: an authenticated cipher," Submission to NIST Lightweight Cryptography Standardization Project, Aug. 2019.
[16] M. Matsui, "New block encryption algorithm MISTY," FSE'97, pp. 54-68, Jan. 1997
[17] J. Wallén, "Design principles of the kasumi block cipher," In Proceedings of the Helsinki University of Technology Seminar on Network Security, 2000
[18] J. Daemen and V. Rijmen, "The Rijndael block cipher: AES proposal," In First candidate conference (AeS1), pp. 343-348, Mar. 1999.
[19] C. Beierle, J. Jean, S. Kölbl, G. Leander, A. Moradi, T. Peyrin, Y. Sasaki, P. Sasdrich, and S. M. Sim, "The SKINNY family of block ciphers and its low-latency variant MANTIS," CRYPTO'16, LNSC 9815, pp. 123-153, Jul. 2016.
[20] V. Grosso, G. Leurent, F. X. Standaert, and K. Varıcı, "LS-designs: Bitslice encryption for efficient masked software implementations," FSE'14, LNSC 8540, pp. 18-37, Apr. 2015.

〈저자소개〉



전 용 진 (Yongjin Jeon) 학생회원
2018년 8월: 국민대학교 정보보안암호수학과 졸업
2020년 8월: 국민대학교 금융정보보안학과 석사
2020년 9월~현재: 국민대학교 금융정보보안학과 박사과정
〈관심분야〉 정보보호, 암호 알고리즘

김 종 성 (Jongsung Kim) 종신회원
2006년 11월: K.U.Leuven, ESAT/SCD-COSIC 정보보호 공학박사
2007년 2월: 고려대학교 정보보호대학원 공학박사
2009년 9월~2013년 2월: 경남대학교 e-비즈니스학과 교수
2013년 9월~2017년 2월: 국민대학교 수학과 교수
2017년 3월~현재: 국민대학교 정보보안암호수학과/금융정보보안학과 교수
〈관심분야〉 정보보호, 암호 알고리즘, 디지털 포렌식