An Efficient Structural Analysis of SAS and its Application to White-Box Cryptography

Hyoungshin Yim
Dept. of Financial information security, Kookmin University
Seoul, South Korea
kuunh2@kookmin.ac.kr

Ju-Sung Kang
Dept. of Information Security, Cryptology, and Mathematics/ Financial information security, Kookmin University
Seoul, South Korea
jskang @ kookmin.ac.kr

Yongjin Yeom
Dept. of Information Security, Cryptology, and Mathematics/ Financial information security, Kookmin University
Seoul, South Korea
salt @ kookmin.ac.kr

***Abstract*—Structural analysis is the study of finding component functions for a given function. In this paper, we proceed with structural analysis of structures consisting of the S (nonlinear Substitution) layer and the A (Affine or linear) layer. Our main interest is the S_1AS_2 structure with different substitution layers and large input/output sizes. The purpose of our structural analysis is to find the functionally equivalent oracle F^* and its component functions for a given encryption oracle $F(= S_2 \circ A \circ S_1)$. As a result, we can compute the decryption oracle F^{*-1} explicitly and break the one-wayness of the building blocks used in White-box Cryptography. Our attack consists of two steps: S layer recovery using multiset properties and A layer recovery using differential properties. We present the attack algorithm for each step and estimate the time complexity. Finally, we discuss the applicability of S_1AS_2 structural analysis in a White-box Cryptography environment.**

***Keywords*— cryptography; structural analysis; SAS strcuture; white-box; security**

I. Introduction

Structural analysis is the study of revealing internal components of a function provided by the structure and unknown component functions. In other words, A given cryptographic oracle in a block-box model can be viewed only in terms of input/output values without knowledge of internal functions. The structural analysis is informally defined as:

- *Structural analysis.* Given an encryption oracle F and its structure, that is, the assumption that the structure of the oracle F is known as illustrated in Fig. 1, we define the structural analysis of F by the analysis to find an equivalent oracle F^* by revealing all internal components of F^* explicitly.

When a function is used as a building block of a cryptographic algorithm, it may include subfunctions that are key-dependent or protected secret components during the cipher operation. For more than two decades, cryptographic study on the structural analysis has been developed particularly on the layered structure which alternates S(nonlinear Substitution) and A(Affine or linear) layers such as SAS, ASASA, and SASAS. These structures are shown in Fig. 1. S layer consists of nonlinear S-boxes in parallel and A layer represents a bitwise linear transformation. We focus on SAS structure with different size S-boxes in the first and the last S layers.

White-box cryptography (WBC) is a method of mixing and hiding encryption key with other components in software implementations of a cryptographic algorithm. That is, it is a technique to perform a cryptographic algorithm without revealing encryption keys or sensitive security parameters. The WBC offers one-wayness properties and key hiding techniques. One-wayness means the unidirectional operation of F in that it is infeasible to execute a decryption oracle F^{-1} given an encryption oracle F. The key hiding technique enables WBC implementation to cloak the key during the encryption process by merging key-dependent operations in the lookup tables. SAS structure is widely used as one of the main building blocks in white-box cryptography.

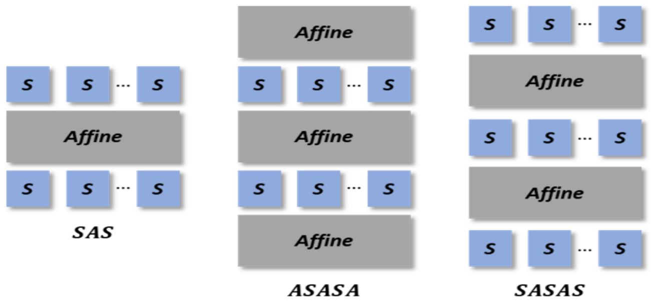


Fig. 1. Examples of substitution/affine structures

In this paper, we suggest an efficient inverting method of SAS structure, which consists of alternating substitution and affine structure. Particularly, we analyze the generalized structure S_1AS_2 where the sizes of the components in two substitution layers are different (5 bits for S_1 and 4 bits for S_2). Our analysis works for SAS with large input/output size. For a WBC construction with 60-bit oracles, we can construct an attack on the main building blocks by inverting oracle with structural analysis.

II. Related Work

In this section, we present previous research for structural analysis and explain several properties of *Multisets.*

A. *Researches on alternating Substitution/affine structures*

Structural analysis consisting of substitution/affine functions has been studied for two decades, as listed in Table I.

In 2001, Biryukov and Shamir [1] proposed how to efficiently attack the substitution/affine layer in a SASAS structure with the same size of S-boxes for all substitution layers. In 2003, Biryukov et al. [2] has presented a detailed analysis method for the affine layer or linear layer. An improved method for finding an affine equivalent function was recently proposed in [8]. In 2014, analysis of ASASA as black-box, white-box, and public-key cryptography was conducted in [3]. Research on this structure has been ongoing [4][6] and even SASASASAS structure was conducted in [5].

TABLE I. Papers Related To The Subtitution/Affine Structure

Year	Topic	Authors
2001	Structural cryptanalysis of SASAS [1]	A. Biryukov, A. Shamir, et al.
2003	Affine Equivalence Algorithms [2]	A. Biryukov, B. Preneel, et al.
2014	Cryptographic Schemes Based on the ASASA [3]	A. Biryukov, C. Bouillauet, et al.
2015	Structural cryptanalysis of ASASA [4]	I. Dinur, O. Dunkelman, et al.
2015	Decomposition attack on SASASASAS [5]	A. Biryukov, D. Khovratovich
2015	Key-Recovery Attack on the ASASA [6]	H. Gilbert, J. Plut, et al.
2016	Analytic Tools for White-box Cryptography [7]	CH. Baek
2018	An Improved Affine Equivalence Algorithm [8]	I. Dinur

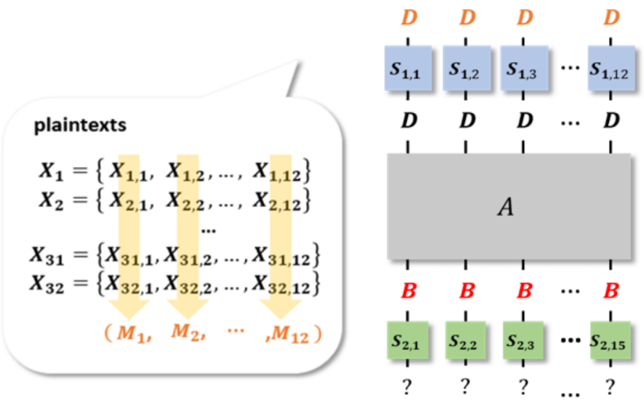
B. *Multiset properties*

We introduce *Multiset* properties that allow characterizing the intermediate values of an encryption structure, even if none of the actual functions is known [1].

Multiset allows for multiple instances for each of its elements. Properties of a *Multiset* are shown below :

- *Property P* (Permutation): all element must appear exactly once.
- *Property E* (Even): each element included an even number of times.
- *Property D* (Dual): it is either *property P* or *property E*.
- *Property B* (Balanced): the *XOR* of all elements is the zero vector.

Multisets may have specific properties for substitution/affine structure. For simplicity, consider a 60-bit oracle with S_1AS_2. We define *Multisets* as follows: Let input $X_i = \{X_{i,1}, X_{i,2}, \dots, X_{i,12}\} \in GF(2^5)^{12}$, for $i \in \{1,2,\dots,32\}$. The input X_i is expressed as input for each of the S-boxes in the first substitution layer. A *Multiset* M_j defined for the input X_i as $M_j = \{X_{1,j}, X_{2,j}, \dots, X_{32,j}\} \in GF(2^5)^{32}, j \in \{1,2,\dots,12\}$, where M_j means a *Multiset* corresponding to each of the S-boxes in the first substitution layer, and when observing this property, look at $(M_1, M_2, \dots, M_{12})$. When the *Multiset* is applied to the S_1AS_2 structure in this paper, each layer meets the following conditions, as shown in Fig. 2.



plaintexts
$X_1 = \{X_{1,1}, X_{1,2}, \dots, X_{1,12}\}$
$X_2 = \{X_{2,1}, X_{2,2}, \dots, X_{2,12}\}$
$X_{31} = \{X_{31,1}, X_{31,2}, \dots, X_{31,12}\}$
$X_{32} = \{X_{32,1}, X_{32,2}, \dots, X_{32,12}\}$
$(M_1, M_2, \cdots, M_{12})$
D D D D
$S_{1,1}$ $S_{1,2}$ $S_{1,3}$ $S_{1,12}$
D D D D
A
B B B B
$S_{2,1}$ $S_{2,2}$ $S_{2,3}$ $S_{2,15}$
? ? ? ?

Fig. 2. *Multiset* properties in $\boldsymbol{S_1AS_2}$ structure

Through Fig. 2, if the *Multiset* for the input satisfies (*D, D, ..., D*), the input *Multiset* of the second substitution layer satisfies (*B, B, ..., B*). Note that the input *Multiset* of the second substitution layer becomes a 15-dimensional vector as $(M_1, M_2, \dots, M_{15})$ because the S-boxes have a 4-bit input. This feature contributes to the recovery of the second S-boxes. The following section details how to attack S_1AS_2 structures.

III. Attack of SAS Structure

This section describes a detailed attack for analysis on S_1AS_2 structure, which has substitution layers of different sizes. Therefore, the goal of structural analysis in this paper is as follows:

GOAL (Efficient structural analysis of S_1AS_2)

Suppose that an encryption oracle F is given with S_1AS_2 structure and different substitution layers. That is, we know the construction of F with its unknown components and we can compute $y = F(x)$ for any input x. The goal of the structural analysis is to construct an equivalent oracle F^* explicitly by determining all its components so that we can compute $x = F^{-1}(y) = F^{*-1}(y)$ for any y as well.

As with the above goal, we obtain encryption oracle F^* and decryption oracle F^{*-1}, which are functionally equivalent to encryption oracle F. Oracle F and the obtained oracle F^* depict in Fig. 3. For any input x, $y = F(x)$ and $y^* = F^*(x)$. The figure shows that the input and output are identical, but the functions constructed inside the oracle are different. Note that, F^* is not uniquely determined. We successfully find one of the several F^*'s. The attack method proposed in this paper consists of two steps. The first step is to convert S_1AS_2 into $S_1\tilde{A}$, a combination of linear and affine, to find S_2^*. The second step finds A in the $S_1\tilde{A}$ structure obtained in step 1. Finally, oracle F^*, which is functionally equivalent to F, can be found. The attack proceeds in two steps described below.

Step1) **Algorithm1** ($S_1AS_2 \rightarrow S_1\tilde{A}$):
Find S_2^* such that $S_2^{*-1} \circ (S_2 \circ A \circ S_1) = \tilde{A} \circ S_1$
Step2) **Algorithm2** ($S_1\tilde{A} \rightarrow S_1^*$):
Find A^* such that $A^{*-1} \circ (\tilde{A} \circ S_1) = S_1^*$

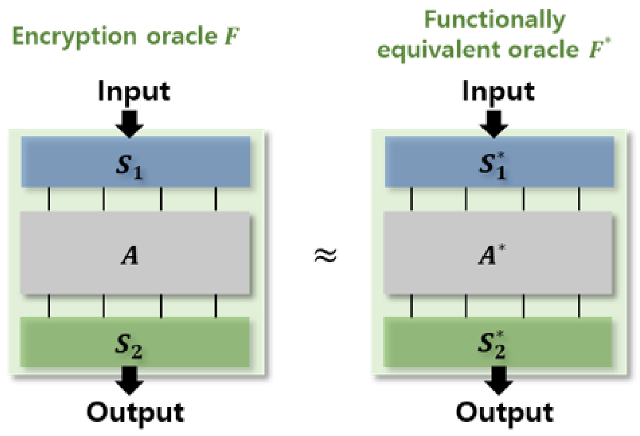


Fig. 3. Encryption oracle and functionally equivalent oracle

A. Components in structural analysis of oracle F

Each function and component defined in this paper shown below:

- $F = S_2 \circ A \circ S_1$ encryption oracle scheme.
- $S_1: GF(2^5)^{12} \rightarrow GF(2^5)^{12}$ invertible nonlinear function.

$$S_1 = \{S_{1,1}, \dots, S_{1,12}\}. \tag{1}$$

- $S_2: GF(2^4)^{15} \rightarrow GF(2^4)^{15}$ invertible nonlinear function.

$$S_2 = \{S_{2,1}, \dots, S_{2,15}\}. \tag{2}$$

- $A: GF(2)^{60} \rightarrow GF(2)^{60}$ invertible affine function.

$$A = (A_1 \,||\, A_2 \,||\, \dots \,||\, A_{12}). \tag{3}$$

B. Components in structural analysis of target oracle F^*

F^* is the target oracle that we have to construct. S_1^*, A^*, and S_2^* are the internal functions. The components obtained at each step shown below:

- $F^* = S_2^* \circ A^* \circ S_1^*$ encryption target oracle.
- $L_1: GF(2^5)^{12} \rightarrow GF(2^5)^{12}$ invertible linear function.
- $L_2: GF(2^4)^{15} \rightarrow GF(2^4)^{15}$ invertible linear function.
- $S_1^*: GF(2^5)^{12} \rightarrow GF(2^5)^{12}$ invertible nonlinear function.

$$\begin{aligned} S_1^* &= L_1^{-1} \circ S_1 \\ &= \{L_{1,1}^{-1} \circ S_{1,1}, \dots, L_{1,12}^{-1} \circ S_{1,12}\} = \{S_{1,1}^*, \dots, S_{1,12}^*\}. \end{aligned} \tag{4}$$

- $S_2^*: GF(2^4)^{15} \rightarrow GF(2^4)^{15}$ invertible nonlinear function.

$$\begin{aligned} S_2^* &= S_2 \circ L_2 \\ &= \{S_{2,1} \circ L_{2,1}, \dots, S_{2,15} \circ L_{2,15}\} = \{S_{2,1}^*, \dots, S_{2,15}^*\}. \end{aligned} \tag{5}$$

- $A^*: GF(2)^{60} \rightarrow GF(2)^{60}$ invertible affine function.

$$\begin{aligned} A^* = L_2^{-1} \circ A \circ L_1 = \tilde{A} \circ L_1 \ (\tilde{A} = L_2^{-1} \circ A). \\ A^* = (A_1^* \,||\, A_2^* \,||\, \dots \,||\, A_{12}^*). \end{aligned} \tag{6}$$

The component functions S_1^*, S_2^* and A^* are not uniquely determined since invertible linear functions in L_1 and L_2 can be arbitrarily chosen. We describe the attack against S_1AS_2 structures using fixed parameters (60-bit oracle with 4-bit/5-bit S-boxes).

C. Recovering S-box layer S_2^* from S_1AS_2

Algorithm 1 recovers the second S-box S_2^* layer from encryption oracle F. In other words, "$S_1AS_2 \rightarrow S_1\tilde{A}$", it recovers S_2^* from S_1AS_2 and reconstructs it into an $S_1\tilde{A}$ structure. The process removes the last S layer as: $S_2 \circ A \circ S_1 \longrightarrow S_2^{*-1} \circ (S_2 \circ A \circ S_1) = L_2^{-1} \circ A \circ S_1 = \tilde{A} \circ S_1$.

Algorithm 1. Recovering S-boxes layer algorithm

Input: encryption oracle $F(= S_2 \circ A \circ S_1)$
Output: S_2^* nonlinear function such that $S_2^{*-1} \circ (S_2 \circ A \circ S_1) = \tilde{A} \circ S_1$

```
1. For each $S_{2,j}$ in ($j = 1, 2, \dots, 15$) do
2.     Choose $2^5$ input values that satisfies properties (D, D, ..., D)
3.     Obtain the corresponding output values of $F$
4.     Indexing the input values of $S_2$ by output values
5.     Establish a linear equation using Multiset properties (B, B, ..., B)
6.     Construct a total of $2^4$ linear equation
7.     Use Gaussian elimination to obtain a non-zero solution $S_{2,j}^{*\,-1}$
8. $S_2^{*-1} \leftarrow (S_{2,1}^{*\,-1} \,||\, S_{2,2}^{*\,-1} \,||\, \dots \,||\, S_{2,15}^{*\,-1})$
9. $S_2^* \leftarrow (S_{2,1}^* \,||\, S_{2,2}^* \,||\, \dots \,||\, S_{2,15}^*)$ // inverse of $S_2^{*-1}$ ($S_2^* \circ S_2^{*-1} = I$)
```

This attack carries out in order of each S-boxes on the second layer of substitution. Suppose that we attack the first S-box in the second substitution layer. We choose 2^5 plaintexts that satisfy the properties (*D, D, ..., D*) and 2^5 corresponding ciphertexts. As explained in Section II, if the plaintext satisfies (*D, D, ..., D*) properties, then the *Multisets* of S_2^* inputs have (*B, B, ..., B*) properties. We proceed with the attack using the sum of all elements, which are balanced properties, zero. First, index the input corresponding to all variables that can come to the S-box output. We set Z to the input of the S-box, shown in Table II.

TABLE II. INDEXING THE INPUT VALUES **Z** OF S_2^*

Z	Z_0	Z_1	…	Z_{15}
Y	0	1	…	15

Since the S-box in the second layer is a nonlinear function from 4bits onto 4bits, 2^4 indexes are used. Next, a linear equation creates using the corresponding indexes of elements constructed from the *Multiset* of the obtained ciphertexts. That is, when the *Multiset* of the output for one S-box is M_{output}, the equation is as follows: $\sum_{t \in M_{output}} S^{-1}(t) = 0$. Let's take a look at the detailed explanation with the toy example. Let us set ciphertexts to $Y_i = \{Y_{i,1}, Y_{i,2}, \dots, Y_{i,15}\}, i \in \{1, 2, \dots, 32\}$. We obtain a *Multiset* for the first S-box when the 2^5 ciphertexts we obtain are equal to the following $Y_1 = \{3, 31, \dots, 19\}$, $Y_2 = \{5, 23, \dots, 10\}$, …, $Y_{32} = \{23, 2, \dots, 11\}$. As a result, $M_{output} = \{3, 5, \dots, 23\}$ and produces a linear equation that satisfies balanced properties.

$$\sum_{t \in M_{output}} S^{-1}(t) = Z_3 \oplus Z_5 \oplus \cdots \oplus Z_{23} = 0. \qquad (7)$$

To find all solutions Z_j ($j = 0, 1, \dots, 15$), we need 16 equations like (7). Thus, the process performs 16 times to obtain an equation. Then, given solution vector $\boldsymbol{Z} := (Z_0, Z_1, \dots, Z_{15})$, 16 linear equations are expressed as $B \cdot \boldsymbol{Z} = 0$ (B: 16×16 matrix over $GF(2)$). However, matrix B has a kernel of dimension 5, and rank is 11. That is, it cannot have a full rank. Consequently, it has multiple solutions, as suggested in [1]. After all, we obtain 11 linearly independent equations after applying Gaussian elimination, which cannot determine a single solution. Therefore, as shown in Fig. 4, the obtained S-box in S_2^* possibly contains a linear function. When attacking using Algorithm 1, we consider the S-box and the affine function which contain parts of linear functions canceled out as in Fig. 4. The following section describes the process of recovering the affine layer from the obtained $S_1\tilde{A}$ structure.

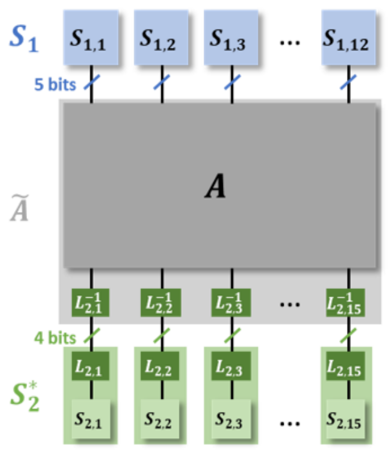


Fig. 4. Modified $\boldsymbol{S_1\tilde{A}S_2^*}$ structure in the S-box recovery phase

D. *Recovering affine layer A^* from $S_1\tilde{A}$*

Algorithm 2 recovers the affine A^* layer. In other words, "$S_1\tilde{A} \to S_1^*$" it recovers A^* from $S_1\tilde{A}$, and we can obtain an S_1^*. The process is formulated as follows: $\tilde{A} \circ S_1 \longrightarrow A^{*-1} \circ (\tilde{A} \circ S_1) = L_1^{-1} \circ \tilde{A}^{-1} \circ A \circ S_1 = L_1^{-1} \circ S_1 = S_1^*$.

Algorithm 2. Recovering affine layer algorithm

Input: oracle $\tilde{A} \circ S_1$

Output: A^* affine function such that $A^{*-1} \circ (\tilde{A} \circ S_1) = S_1^*$

```
1.  $U_{in}$ is the set of possible input differences at $GF(2)^5$
2.  For each $S_{1,j}$ in ($j = 1, 2, \dots, 12$) do
3.      $\Delta_{in} \leftarrow \emptyset$        // $\Delta_{in}$ is the set of 5-bit input differences
4.      $\Delta_{out} \leftarrow \emptyset$       // $\Delta_{out}$ is the set of 5-bit output differences
6.      Repeat
7.          $d_{in} \overset{\$}{\leftarrow} U_{in}$      // Select one randomly from set $U_{in}$
8.          $D_{in} \leftarrow (0, 0, \dots, d_{in}, \dots, 0)$    // $d_{in}$ is the i-th difference of $S_{1,j}$
9.          if $dim(F(\Delta_{in})) < dim(F(\Delta_{in} \cup \{D_{in}\}))$ then
10.             $\Delta_{in} \leftarrow \Delta_{in} \cup \{D_{in}\}$
11.             $\Delta_{out} \leftarrow F(\Delta_{in})$        //Update the difference set $\Delta_{out}$
12.         end if
13.     until $dim(\Delta_{out}) = 5$      //Calculate dimension of the subspace generated by $\Delta_{out}$
14.     Find the basis $< w_1, w_2, w_3, w_4, w_5 >$ of the $\Delta_{out}$
15.     Obtain affine layer $A_j^* := [w_1\ w_2\ w_3\ w_4\ w_5]$, where each basis $w_i$ ($i = 1, 2, \dots, 5$) is a column vector
16. $A^* \leftarrow (A_1^* \parallel A_2^* \parallel \dots \parallel A_{12}^*)$
```

Attacks proceed in order from the first S-box. Suppose that we attack the first S-box. The key idea of this attack is to calculate the difference (Δ in Fig. 5) by selecting two plaintexts from the corresponding S-box. The input difference d_{in} of the S-box, the attack location, should not be zero ($d_{in} \neq 0$). The input difference of the other location, S-box, should be zero. Set the difference between each S-box to one input difference $D_{in} = (d_{in}, 0, , , 0)$.

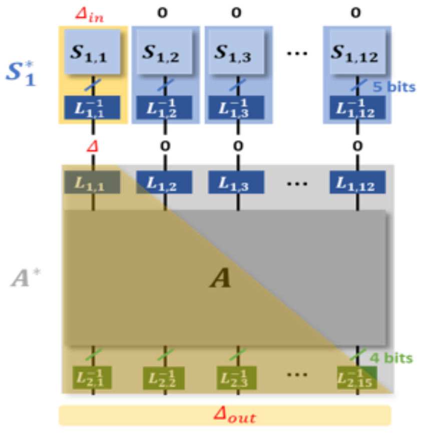


Fig. 5. Modified $\boldsymbol{S_1^*A^*}$ structure in the affine recovery phase

As depicted in Fig. 5, only the difference set Δ_{in} corresponding to the first S-box puts an input Δ that affects the output difference Δ_{out} $(F(\Delta_{in}) = \Delta_{out})$. The reason for this setting is that the difference characteristics of the first S-box corresponding to the location of the attack affect the output. As such, outputs obtained using the difference condition are attacked by using the basis transformation in the output difference values Δ from $S^*_{1,1}$. To recover the affine layer A^*_1, the important point is that the elements of set Δ_{out} must satisfy their independence from each other. The verification method confirms the changes in the dimension of set $F(\Delta_{in} \cup \{D_{in}\})$, which adds the calculated difference D_{in} with $F(\Delta_{in})$. If dimension $dim(\ F(\Delta_{in} \cup \{D_{in}\})\ )$ is greater than dimension $dim(F(\Delta_{in}))$, then the difference values in set Δ_{in} and D_{in} are independent of each other. Therefore, the calculated differences under that condition are stored in Δ_{in}, a set of input differences. Through the above process, we construct an independent five-difference set Δ_{in}. If the $S^*_{1,1}$ output dimension becomes 5, the dimension of the final output dimension becomes 5 or less. Therefore, if we find a case where the confirm the independent because only the case where the dimension of the $S^*_{1,1}$ output values is five.

The affine layer is the basis transformation of the sets obtained, and the process shows below. Although we do not know what the basis is for the difference output values of the original $S_{1,1}$, by adding a linear function, we assume that five linearly independent output differences of $S^*_{1,1}$ form the standard basis. The standard basis expresses as $V = < e_1, e_2, e_3, e_4, e_5 >$ and affine layer performs, the following equation is satisfied:

$$\begin{aligned} A^*_1(V) &= < A^*_1(e_1), A^*_1(e_2), A^*_1(e_3), A^*_1(e_4), A^*_1(e_5) > \\ &= < w_1, w_2, w_3, w_4, w_5 >. \end{aligned} \tag{8}$$

The w_i is the basis for ciphertexts. Eventually, the affine layer is represented by a matrix of w_i, which is a column vector.

$$A^*_1 \coloneqq [w_1\ \ w_2\ \ w_3\ \ w_4\ \ w_5]. \tag{9}$$

If we will use these features, the part of the affine layer recovered. Like the above process, recover the affine layer A^* by performing the same procedure for other S-box.

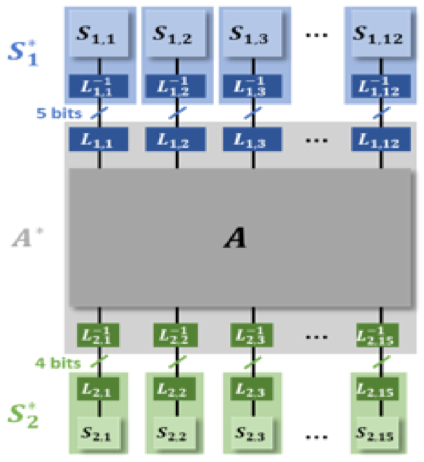


Fig. 6. An $\boldsymbol{F}^*$ function functionally equivalent to encryption oracle $\boldsymbol{F}$.

Through Algorithm1 and Algorithm2, we were able to obtain S^*_1, S^*_2, and A^* from the S_1AS_2 structure. The F^* oracle consisting of the obtained functions is functionally equivalent to the encryption oracle F. After all, since we know S^* and A^*, we can obtain each function's inverse function and generate a decryption oracle F^{*-1}. Fig. 6 shows an F^* function that recovered the $S^*_1A^*S^*_2$ structure.

IV. Complexity of Attack

In this section, we present the complexity based on the algorithms presented in Section Ⅲ.

A. *Complexity of recovering S-box layer*

The main operation in the S-box recovery phase is Gaussian elimination. First, since a Gaussian elimination operation performs for each S-box, a total of 15 times achieve. To be more specific, when we set up the linear equations in each S-box, it is represented as $B \cdot \boldsymbol{Z} = 0$, and B becomes a 16×16 matrix. Using Gaussian elimination in this matrix results in the time complexity of $O(2^{12})$. The reason is in the process of proceeding with the Gaussian elimination method. The first phase in Gaussian elimination is 16 equations with 16 variables, so 16 pivots are required, with a complexity of $O(16)$. The next phase is to subtract multiple rows from each other's rows for each pivot, for which the complexity is $O(16^2)$. Thus, the total runtime is $O(16 \cdot 16^2) = O(2^{12})$. However, since we have a total of 15-second layer S-boxes, it becomes $O(15 \cdot 2^{12})$. By expressing it in general, it describes as follows:

$$O(k_2 \cdot 2^{3 \cdot m_2}). \tag{10}$$

k_2 means the number of S-boxes in the second substitution layer, and m_2 means the input/output size of each S-box. The calculated complexity is the same as the complexity presented in [1] paper.

B. *Complexity of recovering Affine layer*

Like the S-box recovery phase, the main operation of the affine layer recovery phase is Gaussian elimination. Five 60-bit outputs of the $S_1\tilde{A}$ structure form a 5×60 matrix. It was using Gaussian elimination in this matrix that results in the time complexity of $O(5\ \cdot 60^2)$. First, since there are 5 equations, it needs 5 pivots, and the time complexity for this is $O(5)$. Next, the multiplication and subtraction operations perform using each row for each pivot calculation. The time complexity for this becomes $O(60^2)$. But, since this process has to be performed for each S-box, the time complexity is equal to $O(12 \cdot 5\ \cdot 60^2) = O(2^6 \cdot 3^3 \cdot 5^3)$. By expressing it in general, it describes as follows:

$$O(k_1 \cdot m_1 \cdot n^2). \tag{11}$$

k_1 means the number of S-boxes in the first substitution layer, and m_1 means the input/output size of each S-box. $n = k_1 * m_1$, which means the input/output size of the entire oracle. As a result, time complexity of S-box recovery phase is $O(15 \cdot 2^{12}) \approx 2^{16}$ and time complexity of affine recovery phase is $O(2^6 \cdot 3^3 \cdot 5^3) \approx 2^{18}$. Thus, the attack on the entire S_1AS_2

structure is approximately 2^{19}, and the phase to recover affine layer is more time complexity than the S-box recovery phase.

V. Application of SAS Structure in White-Box Cryptography

This section discusses how S_1AS_2 structural analysis, introduced in Section Ⅲ, applies to the WBC. S_1AS_2 structural analysis can generate a functionally equivalent decryption oracle when given an encryption oracle. This does not satisfy the one-wayness of the WBC. However, research on the WBC with SAS structure is still in progress. In 2019, Shi presented the WBC scheme with a SAS structure [9]. Shi's WB model has a Feistel structure and a SAS structure consisting of TBOX. The detailed structure for this Shi's WB model summarizes in TABLE III.

TABLE III. Structure Form of Shi's Wb Model

Shi's WB model	
Structures	Feistel structure, SAS structure
Total length	120 bits (60bits, 60bits)
Round	16
S-box layers	$S_1 : GF(2^5)^{12} \rightarrow GF(2^5)^{12}$, $S_2 : GF(2^4)^{15} \rightarrow GF(2^4)^{15}$
Affine layer	$A : GF(2)^{60} \rightarrow GF(2)^{60}$
Security claimed level	3×2^{60}

As shown in Table III, we can confirm that it has the same class as the S_1AS_2 structure presented in this paper. In addition, substitution/affine is composed of a single T-box to provide oracle characteristics. The Feistel structure, Shi's WBC, also exposes the input/output values of the TBOX since the input/output values expose by round. Furthermore, the internal function is an undisclosed environment. A detailed structure is shown in Fig. 7.

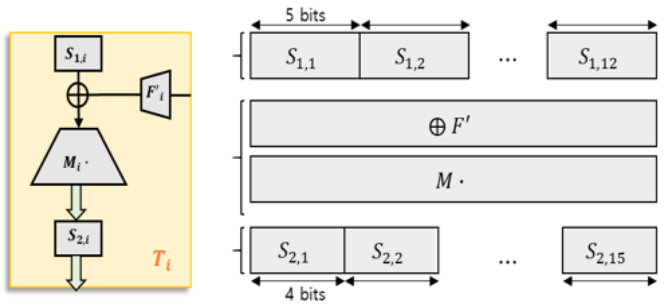


Fig. 7. SAS structure in Shi's WB model

The security claimed in Shi's [9] base on components of different widths and an appropriate number of rounds. This paper suggested that the time complexity for the attack is 3×2^{60} for 2.5 rounds. The reasons for security time complexity are as follows:

- The sizes of S-boxes in two layers are different 4 bits and 5 bits, respectively.
- (4, 5) are coprime so that it can obtain a relatively large least common multiplier for defeating the *Multiset* attack.

However, in this paper, we showed successful structural analysis in the SAS structures with the same class. In Section Ⅳ, we calculate the time complexity 2^{19} for this attack, which we can confirm is much lower than the security level 3×2^{60} claimed by Shi. We implemented the attack method proposed in this paper and the execution time measured within 1 second. Therefore, our result shows that Shi's WBC model is prone to be attacked.

VI. Conclusion

This paper proposes an efficient structural analysis for S_1AS_2 with large input/output sizes and different S layers. The purpose of structural analysis is to find a functionally equivalent decryption oracle F^{*-1} for a given encryption oracle F. We analyze the time complexity for the proposed attack algorithm and prove that it has an extremely low level of security. Furthermore, we also experimentally confirm that the attack is possible in less than 1 second with our implementation. Therefore, when applied to WBCs with one-wayness properties, the attack can successfully execute attacks within a few seconds depending on the number of look-up tables. As a future work, the structural analysis presented in this paper is likely to break one-wayness of WBC scheme with the SAS structure.

Acknowledgment

This work has supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021M1A2A2043893).

References

[1] A. Biryukov, A. Shamir, "Structural Cryptanalysis of SASAS," Advances in Cryptology - EUROCRYPT 2001, LNCS 2045, Springer-Verlag, pp. 394–405, 2001.

[2] A. Biryukov, C.D. Cannière, A. Braeken, and B. Preneel, "A toolbox for cryptanalysis: linear and affine equivalence algorithms," Advances in Cryptology - EUROCRYPT 2003, LNCS 1267, Springer-Verlag, pp. 33–50, 2003.

[3] A. Biryukov, C. Bouillaguet, and D. Khovratovich, "Cryptographic schemes based on the ASASA structure: Black-box, white-box, and public-key (extended abstract)," ASIACRYPT 2014, Part I, LNCS 8873, pp. 63–84, Dec. 7–11, 2014.

[4] I. Dinur, O. Dunkelman, T. Kranz, and G. Leander, "Decomposing the ASASA Block Cipher Construction," Cryptology ePrint Archive, Report 2015/507, 2015.

[5] A. Biryukov and D. Khovratovich, "Decomposition attack on SASASASAS," Cryptology ePrint Archive, Report 2015/646, 2015.

[6] H. Gilbert, Jérôme Plût, and J. Treger, "Key-Recovery Attack on the ASASA Cryptosystem with Expanding S-boxes," Advances in Cryptology – CRYPTO 2015, Springer, pp. 475-490, 2015.

[7] Chung Hun Baek, "Analytic Tools for White-box and Lattice Cryptography," Ph. D. Thesis, Department of Mathematical Sciences, Seoul National University, 2016.

[8] I. Dinur, "An Improved Affine Equivalence Algorithm for Random Permutation," Advanceds in Cryptology –EUROCRYPT 2018, Springer, pp. 413-442, 2018.

[9] Y. Shi, W. Wei, H. Fan, M.H. Au, X. Luo, "A Light-Weight White-Box Encryption Scheme for Securing Distributed Embedded Devices," IEEE Transaction on Computers, pp. 1411-1427, 2019.