Decomposing the ASASA Block Cipher Construction

Itai Dinur[1], Orr Dunkelman[2], Thorsten Kranz[3], and Gregor Leander[3]

[1] Département d'Informatique, École Normale Supérieure, Paris, France
[2] Computer Science Department, University of Haifa, Israel
[3] Horst Görtz Institute for IT-Security, Ruhr-Universität Bochum, Germany

Abstract. We consider the problem of recovering the internal specification of a general SP-network consisting of three linear layers (A) interleaved with two Sbox layers (S) (denoted by ASASA for short), given only black-box access to the scheme. The decomposition of such general ASASA schemes was first considered at ASIACRYPT 2014 by Biryukov et al. which used the alleged difficulty of this problem to propose several concrete block cipher designs as candidates for white-box cryptography. In this paper, we present several attacks on general ASASA schemes that significantly outperform the analysis of Biryukov et al. As a result, we are able to break all the proposed concrete ASASA constructions with practical complexity. For example, we can decompose an ASASA structure that was supposed to provide 64-bit security in roughly 2^{28} steps, and break the scheme that supposedly provides 128-bit security in about 2^{41} time. Whenever possible, our findings are backed up with experimental verifications.

Keywords: Block cipher, ASASA, white-box cryptography, integral cryptanalysis, differential cryptanalysis, Boomerang attack.

1 Introduction

In this paper we consider generic substitution permutation networks consisting of linear layers interleaved with Sbox layers. The linear layers considered are arbitrary linear bijections on the full state, whereas the Sbox layers partition the state into several fixed-size parts and apply a non-linear permutation to each part in parallel. Special cases of this general set-up include a large variety of well-known block ciphers, most prominently the AES.

Such constructions are far from being new. The first construction, due to Patarin and Goubin, is the ASAS construction, i.e., two rounds of a linear layer (A) followed by an Sbox layer (S) [14]. The scheme was later broken by Biham, exploiting the fact that the Sboxes were chosen to be non-bijective [1] (which allowed for a simple differential attack).

Later, Biryukov and Shamir explored the general construction, denoted by SASAS (three Sbox layers and two linear layers) [4]. In the context of this system, the cryptanalytic task is to to recover the actual specification of the secret Sboxes

and linear layers, given only black-box access to the SP-network and its inverse. The attack is a generalization of the integral attack [11] (a.k.a. square attack) originally proposed by Knudsen as an attack on the cipher SQUARE [7]. It is thus a structural attack in nature, exploiting the fact that the SASAS construction is composed of only three permutation layers over a smaller alphabet interleaved with linear layers, but ignoring the actual specification of these layers.

Besides the very general results of [4], several papers have focused on special cases of the above scenario where more information about the internal structure is known. Most prominently, a case that has received considerable attention is where only the Sboxes are unknown but the specification of the linear layers is completely revealed to the attacker (see e.g. [5,15]).

Here we focus not on a special case of the most general setting but rather on the natural complement of the SASAS structure, namely ASASA. That is, we consider an n-bit SP-network consisting of three linear (or affine) layers, interleaved with two Sbox layers, where each Sbox is a permutation over b bits. Our main motivation for analyzing this particular set-up was given by a recent proposal from ASIACRYPT 2014 [3] which constructed block ciphers with the additional property of having memory-hard white-box implementations (see [3] for details). Such a block cipher is represented as a black-box (or a few black-boxes) describing how each input is mapped to an output. The goal of the adversary is to find a succinct representation of the given black-box mapping, which in our case implies recovering (or decomposing) the actual specifications of the affine and Sbox layers.

Since the security of generic ASASA constructions was not previously analyzed, the authors of [3] supported their designs by claiming that they resist standard attacks (such as differential and integral cryptanalysis, and boomerang attacks), essentially due to the external affine layers that hide the internal structure of the Sboxes. The best generic attack on the ASASA construction proposed in [3] is a variant of the attack by Biryukov and Shamir on the SASAS structure, and is extremely inefficient, having time complexity of $2^{b(n-b)}$. For example, [3] claims that an ASASA scheme with a block size of $n = 16$ and an Sbox size of $b = 8$ provides security of $2^{8(16-8)} = 2^{64}$.

It should be noted that [3] also uses a special case of the ASASA structure with expanding Sboxes to construct strong white-box cryptography (again we refer to the paper for details) and that this part has been recently broken [8] by algebraic cryptanalysis. However, those attacks do not seem to carry over to the general ASASA case that we consider here.

1.1 Our Contribution

In this paper we describe several attacks that decompose the ASASA structure significantly faster than the $2^{b(n-b)}$ complexity of [3]. Our most efficient attacks have time complexity of roughly $n \cdot 2^{3n/2}$, and when applied to the specific instances proposed in [3], they have practical complexities. For example, the ASASA scheme with $n = 16$ and $b = 8$, which was claimed to provide security of 2^{64}, can in fact be broken in $16 \cdot 2^{(3 \cdot 16)/2} = 2^{28}$ time.

Table 1 shows the effectiveness of our attacks when applied to the instances given in [3] in detail.

ASASA instance	Security claim [3]	Our attack complexity (Section 3.3)
2×8-bit	64 bits	28 bits
2×10-bit	100 bits	35 bits
3×8-bit	128 bits	41 bits

Table 1. Attack complexities for the ASASA instances suggested in [3].

More precisely, we present and analyze three different attack strategies. The first attack follows the general idea of an integral attack and is thus a structural attack. The second and third attacks on the other hand are statistical attacks: the second attack is based on a Boomerang attack, making use of boomerang distinguishers in the known-plaintext setting. The authors of [3] were aware of the existence of such boomerang distinguishers, but left the possibility of exploiting them efficiently in an attack as an open problem, which we address. Finally, the third attack is a variant of a differential attack making use of non-random behaviour of the differential distribution table. The running time of the integral attack is roughly $n \cdot 2^{3n/2}$, the Boomerang has an attack complexity of roughly $2^{3n/2+3b/2}$ and the differential attack a complexity of 2^{2n}..

The integral attack is generally the most efficient attack and should be used whenever possible. However, this attack is applicable with the claimed complexity only to ASASA constructions in which $b > \sqrt{n}$. In cases where $b \leq \sqrt{n}$, the Boomerang attack is generally the most efficient attack, as its time complexity is at most $2^{3n/2+3\sqrt{n}/2}$. However, the success and efficiency of the integral and boomerang attacks are based on some assumptions on ASASA instances in which the linear and Sbox layers are chosen at random. These assumptions seem rather natural and were verified whenever possible on random instances of ASASA with small state sizes. Nevertheless, it is clear that our assumptions do not hold for all ASASA instances, and in such cases (which seem rather sparse), one can try to apply the differential attack which has a higher time complexity of 2^{2n}, but is based on different assumptions. Finally, we note that we focus on ASASA constructions in which all Sboxes have the same size, but our algorithms easily generalize to schemes built with difference Sbox sizes.

The paper is organized as follows. After fixing the notation and some preliminary results outlining the general attack vector in Section 2, we present our integral attack in Section 3. The Boomerang attack is given in Section 4. Finally, the differential attack is described in Section 5.

2 Preliminaries

Let

$$F : \mathbb{F}_2^n \to \mathbb{F}_2^n$$

be the ASASA construction with k Sboxes on b-bits each ($n = kb$). More precisely, F is constructed as

$$F = L_2 \circ S_1 \circ L_1 \circ S_0 \circ L_0$$

where L_i are linear bijections on $\mathbb{F}_2^n$ and S_i consist of parallel applications of k b-bit Sboxes. That is,

$$S_i(x_1, \ldots, x_k) = \left(S_i^{(1)}(x_1), \ldots, S_i^{(k)}(x_k)\right)$$

with $x_i \in \mathbb{F}_2^b$

If we can recover the first linear layer, we are left with the SASA construction that can be decomposed very efficiently as shown in [4]. Thus, the aim of our attacks is to recover L_0 (or L_2 by considering the inverse of F). For this, first note that it is impossible – and unnecessary – to recover L_0 exactly. This is due to the fact that a given ASASA instance is not uniquely determined by F. More precisely, replacing L_0 by

$$L_0' = \begin{pmatrix} T_1 \\ T_2 \\ \vdots \\ T_k \end{pmatrix} \circ L_0$$

and $S_0^{(i)}$ by

$$S'^{(i)}_0 = S_0^{(i)} \circ T_i^{-1},$$

where $T_i : \mathbb{F}_2^b \to \mathbb{F}_2^b$ are linear bijections, results in the same ASASA instance F. This observation motivates the following definition.

Definition 1. *Two linear bijections L and L' on $\mathbb{F}_2^n$ are* ASASA-equivalent *if there exist linear bijections T_i on $\mathbb{F}_2^b$ such that*

$$L' = \begin{pmatrix} T_1 \\ T_2 \\ \vdots \\ T_k \end{pmatrix} \circ L$$

For recovering L_0 (up to equivalence) we are actually going to recover the subspaces

$$V_i = L_0^{-1}(U_i)$$

where

$$U_i = \{(x_1, x_2, \ldots, x_k) \mid x_j \in \mathbb{F}_2^b \text{ and } x_j = 0 \text{ if } j \neq i\}.$$

As formalized in the proposition below, given the set of V_i's basically determines L_0 up to (unavoidable) equivalences.

Proposition 1. *Given k subspaces V_i of dimension b it holds that any linear bijection $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$ such that $L(V_i) = U_i$ with U_i as above is ASASA equivalent to L_0.*

Proof. We consider $T = L \circ L_0^{-1}$ and observe that T maps inputs from U_i to U_i since

$$T(U_i) = L(L_0^{-1}(U_i)) = L(V_i) = U_i.$$

As a result, T represents k parallel linear bijections T_i over $\mathbb{F}_2^b$ and L is ASASA equivalent to L_0:

$$L \circ L_0^{-1} = T \Rightarrow L = T \circ L_0$$

□

Thus, after recovering the spaces V_i we can set-up linear equations for L_0 and any bijective linear mapping fulfilling all those linear equations will be a valid solution for L_0.

More precisely, L_0 can be written as

$$L_0 = \begin{pmatrix} L_0^{(1,1)} & L_0^{(1,2)} & \dots & L_0^{(1,k)} \\ L_0^{(2,1)} & L_0^{(2,2)} & \dots & L_0^{(2,k)} \\ \vdots & \vdots & \ddots & \vdots \\ L_0^{(k,1)} & L_0^{(k,2)} & \dots & L_0^{(k,k)} \end{pmatrix}$$

where $L_0^{(i,j)}$ are $b \times b$ submatrices over $\mathbb{F}_2$. Then, for any element $(x_1, \dots, x_k) \in V_i \subseteq (\mathbb{F}_2^b)^k$ we obtain b equations

$$L_0^{(j,1)} x_1 + L_0^{(j,2)} x_2 + \dots + L_0^{(j,k)} x_k = 0 \quad \forall j \in \{1, \dots, k\},\ j \neq i.$$

Choosing b linearly independent vectors from every V_i leads to a system of

$$k(k-1)b^2 = k^2b^2 - kb^2$$

linearly independent equations over $\mathbb{F}_2$ with k^2b^2 unknowns for L_0. This system of equations has 2^{kb^2} solutions and we know that the invertible solutions are exactly the linear bijections that are ASASA equivalent to L_0. The number of such linear bijections for a given L_0 is N_b^k where

$$N_b = \prod_{i=0}^{b-1} (2^b - 2^i) = 2^{b^2} \prod_{i=1}^{b} (1 - 2^{-i}) = 2^{b^2} p_b$$

is the number of invertible $b \times b$ matrices and p_b is the probability that a randomly chosen $b \times b$ matrix is invertible. Accordingly, the fraction of invertible solutions for the system of linear equations is

$$\frac{2^{kb^2} p_b^k}{2^{kb^2}} = p_b^k.$$

Thus, if we know the set of V_i's for an ASASA constructions with k b-bit Sboxes, we try out p_b^k solutions on average. Since $p_b > 2^{-1.792}$, it holds that

$$p_b^k > 2^{-1.792k}$$

when considering constructions with up to k Sboxes. This step is completely separated from the attacks that find the set of V_i's. Accordingly, the complexities do *not* multiply and as we will see this step does not dominate the overall attack complexity.

Note that it is sufficient to know $V_{\pi(i)}$ for an unknown permutation π. The solution will then be a matrix which is ASASA equivalent to L_0 with its rows permuted according to π. This is also a valid solution as additionally permuting the Sboxes in S_1 and the columns in L_1 results in the same ASASA instance.

3 Integral Attack

In this section we describe our integral attack on the ASASA construction. We start by describing a basic integral attack that recovers the subspaces V_i for $i \in \{1, \ldots, k\}$ in time complexity of about $n \cdot 2^{2n}$. Finally, we optimize the attack and reduce its time complexity to about $n \cdot 2^{3n/2}$.

The analysis of the attack assumes that the parameters of the ASASA construction satisfy $b > k$, or equivalently $b > \sqrt{n}$ (and $k < \sqrt{n}$). Moreover, it assumes that the algebraic degree of the Sbox layers S_0 and S_1 and their inverses is the maximal possible value of $b - 1$.[4] We note that if the Sboxes are selected at random, then this will be the case with high probability. Additionally, the analysis is based on more subtle and heuristic (but natural) assumptions that we specify later in this section.

The starting point of the attack is the following integral property, which was used to devise a related attack in the original ASASA paper [3]. We begin with a short definition, followed by the property and its proof.

Definition 2. *Given a set R and $x \in \mathbb{F}_2^n$, define $R + x \triangleq \{x + y \mid y \in R\}$*

Proposition 2. *Let $x \in \mathbb{F}_2^n$ and $i \in \{1, \ldots, k\}$, then $\sum_{y \in V_i} F(x + y) = \mathbf{0}$.*

Similar properties have been used before in integral and related attacks, most notably in the cryptanalysis of the SASAS structure [4]. It is possible to prove this property combinatorially (as done in [4]), but here we give an algebraic proof, as it is more relevant to the rest of this section. The proof is based on the use of high-order derivatives (see [10,12] for details about high-order differential cryptanalysis).

[4] This condition can be somewhat relaxed, but we do not go into details for the sake of simplicity.

Proof. Due to the linearity of L_0, we have $L_0(x + V_i) = L_0(x) + L_0(V_i) = L_0(x)+U_i$. Namely, Sbox i is active in S_0 (its input attains all 2^b possible values), while all other Sboxes are inactive, as their inputs are fixed to the corresponding b bits of $L_0(x)$. Since $S_0^{(i)}$ is a permutation, then $S_0(L_0(x)+U_i) = S_0(L_0(x))+U_i$, which is an affine subspace of dimension b. We are interested in computing the sum of outputs (over $\mathbb{F}_2$) of this subspace through the remaining layers $L_2 \circ S_1 \circ L_1$, which is equivalent to evaluating a b-order derivative of these layers on their n output bits. Therefore, in order to show that $\sum_{y \in V_i} F(x+y) = \mathbf{0}$, it is sufficient to show that the algebraic degree of every output bit of $L_2 \circ S_1 \circ L_1$ is less than b, which implies that all of the b-order derivatives are zero.

All the b-bit Sboxes in the scheme (and in S_1) are bijective, and therefore, their algebraic degree is at most $b-1$, implying that the degree of S_1 is upper bounded by $b-1$. Since L_1 and L_2 are affine, the degree of $L_2 \circ S_1 \circ L_1$ is less than b, proving the claim. □

In the attack we sum over the encryption values of larger affine subspaces, and use the following (more general) property.

Proposition 3. *Let $x \in \mathbb{F}_2^n$, $i \in \{1, \ldots, k\}$, and let R be a linear subspace such that $V_i \subseteq R$, then $\sum_{y \in R} F(x+y) = \mathbf{0}$.*

Proof. Write R as a direct sum of orthogonal subspaces $R = V_i \oplus R'$. Then

$$\sum_{y \in R} F(x \oplus y) = \sum_{y_1 \in R', y_2 \in V_i} F(x + y_1 + y_2) = \sum_{y_1 \in R'} \Big(\sum_{y_2 \in V_i} F((x + y_1) + y_2) \Big) = \mathbf{0},$$

using the previous property. □

When we select at random a linear subspace R of a fixed dimension d such that $b \leq d < n$, Proposition 3 implies that we can potentially distinguish between the case $V_i \subseteq R$ (where $\sum_{y \in R} F(y) = \mathbf{0}$), and the case $V_i \nsubseteq R$ (assuming $\sum_{y \in R} F(y) \neq \mathbf{0}$). Distinguishing between these two cases is very useful, as it narrows down the search for V_i from the full n-dimensional space to the smaller space R of dimension $d < n$. Assuming that we find two subspaces $R_1 \neq R_2$ such that $V_i \subseteq R_1$ and $V_i \subseteq R_2$, then $V_i \subseteq R_1 \bigcap R_2$, which further narrows down the search for V_i. We then continue the search until we find sufficiently many subspaces such that their intersection gives the b-dimensional subspace V_i. This gives rise to the attack given in Algorithm 1.

The success and complexity of the attack are determined by the two quantities defined below.

Definition 3. *Let R be a subspace chosen at random by Algorithm 1 and fix $i \in \{1, \ldots, k\}$. The covering probability is defined as*

$$p_c \triangleq Pr[V_i \subseteq R].$$

Input : An integer d
Output: Subspaces V_i for $i \in 1, \ldots, k$

```
1  begin
       // The set of candidate subspaces
2      Candidates ⟵ ∅;
3      Found ⟵ 0;
4      while Found < k do
5          Pick a subspace R of dimension d at random;
6          s ⟵ Σ_{y∈R} F(y);
7          if s = 0 then
               // Iterate over found subspaces
8              for j ← 1 to size(Candidates) do
9                  R' ⟵ R ⋂ Candidates[j];
10                 if R' = Candidates[j] then
                       // R contains a previously found subspace
11                     go to 4;
12                 if dim(R') < b then
                       // The dimension of R' is too small
13                     go to 8;
14                 s' ⟵ Σ_{y∈R'} F(y);
15                 if s' = 0 then
                       // Narrow down subspace
16                     Candidates[j] ← R';
17                     if dim(R') = b then
                           // Found a subspace of dimension b
18                         V_Found ← R';
19                         Found ← Found + 1;
20                     go to 4;
               // Store R, as it does not narrow down any previously
                  found subspace
21             NewIndex ← size(Candidates) + 1;
22             Candidates[NewIndex] ← R;
```

Algorithm 1: Integral Attack

Definition 4. *Let R be a subspace chosen at random by Algorithm 1. The false alarm probability is defined as*

$$p_f \triangleq Pr[\sum_{y \in R} F(y) = \mathbf{0} \mid V_i \nsubseteq R \text{ for each } i \in \{1, \ldots, k\}]$$

.

Our analysis is based on the two assumptions below. The validity of these assumptions depends on the concrete ASASA scheme and we cannot prove them in general. The assumptions are discussed at the end of Section 3.2.

Assumption 1 *The are no false alarms in Line 15 of Algorithm 1. Namely, if $s' = \mathbf{0}$, then there exists V_i such that $V_i \subseteq R'$.*

This assumption guarantees that if the algorithm halts, it returns the correct answer. This assumption can be somewhat relaxed, as noted at the end of Section 3.2.

Assumption 2 *For $d = n - k$, the false alarm probability in Line 7 of Algorithm 1 satisfies $p_f < 2^{-n/2}$. Namely,*

$$Pr[\sum_{y \in R} F(y) = \mathbf{0} \mid V_i \nsubseteq R \text{ for each } i \in \{1, \ldots, k\}] < 2^{-n/2}$$

.

Note that if $\sum_{y \in R} F(y)$ is uniformly distributed when $V_i \nsubseteq R$ for each $i \in \{1, \ldots, k\}$, then $p_f = 2^{-n}$. Assumption 2 is weaker and only requires $p_f < 2^{-n/2}$.

The analysis of the algorithm is rather involved, and we start with a high level description.

3.1 Analysis Overview

We estimate the time complexity of the algorithm based on several propositions which are stated below and proved in Section 3.2.

Proposition 4. *Let R be an affine subspace of dimension $d > b$ chosen at random by Algorithm 1, then*

$$p_c \geq 2^{b(d-n)} \cdot (1 - 2^{b-d-1}) \approx 2^{b(d-n)}.$$

Proposition 5. *Let e be the number of subspaces R that Algorithm 1 considers before halting, then*

$$e \approx n \cdot p_c^{-1}.$$

Proposition 6. *The expected total time complexity of Algorithm 1 is about*

$$(2^d \cdot e) \cdot (1 + e \cdot p_f^2).$$

Plugging the value of p_c obtained from Proposition 4 into the value of e, obtained from Proposition 5, we get

$$e \approx n \cdot p_c^{-1} \approx n \cdot 2^{b(n-d)}.$$

Plugging this value of e into the total time complexity of the attack obtained from Proposition 6, we can conclude that it is preferable to select the largest dimension d which does not significantly affect p_f. However, according to the upper-bound of Proposition 7, we cannot take d too large, as this will result in $p_f = 1$, causing the algorithm to fail (or to be extremely inefficient). We note that unlike the previous propositions, the upper bound on d is not directly used in the complexity analysis of the attack, but rather restricts the possible choice of parameters.

Proposition 7. *Assume that the parameters of the ASASA construction satisfy $b > k$ and $d > n - k$, then $p_f = 1$.*

Since we require $p_f < 1$, we select the maximal possible value for which this may occur, namely $d = n - k$. According to Proposition 4,

$$p_c \approx 2^{b(d-n)} = 2^{-bk} = 2^{-n}.$$

Plugging the value $p_c = 2^{-n}$ into the expression obtained in Proposition 5, we get

$$e \approx n \cdot p_c^{-1} \approx n \cdot 2^n.$$

The total time complexity of the attack is estimated in Proposition 6 as $(2^d \cdot e) \cdot (1 + e \cdot p_f^2)$, where $e \approx n \cdot 2^n$. According to Assumption 2, $p_f < 2^{-n/2}$, implying that $1 + e \cdot p_f^2 \approx 1$ and the time complexity of the algorithm is about

$$2^d \cdot e \approx n \cdot 2^{2n},$$

since $d = n - k \approx n$ when $k < \sqrt{n}$.

3.2 Detailed Analysis

In this section we prove the propositions stated above and consider the assumptions 1 and 2.

Proof of Proposition 4 Let $(a_1, \ldots, a_b)$ be an arbitrary basis of V_i. We have $V_i \subseteq R$ if and only if $a_j \in R$ for $j \in \{1, \ldots, b\}$. Since R is a random d-dimensional subspace, then $Pr[a_1 \in R] = 2^{d-n}$. Next, since a_1 and a_2 are linearly independent $Pr[a_2 \in R \mid a_1 \in R] = 2^{d-n} - 2^{-n}$, and $Pr[\{a_1, a_2\} \subseteq R] = Pr[a_1 \in R] \cdot Pr[a_2 \in R \mid a_1 \in R] = 2^{d-n} \cdot (2^{d-n} - 2^{-n})$.

In general, for $j > 1$

$$Pr[\{a_1, \ldots, a_j\} \subseteq R \mid \{a_1, \ldots, a_{j-1}\} \subseteq R] = 2^{d-n} - 2^{j-2-n}$$

and therefore

$$p_c = Pr[\{a_1, \ldots, a_b\} \subseteq R] =$$
$$Pr[\{a_1, \ldots, a_b\} \subseteq R \mid \{a_1, \ldots, a_{b-1}\} \subseteq R] \cdot Pr[\{a_1, \ldots, a_{b-1}\} \subseteq R] =$$
$$Pr[a_1 \in R] \cdot \prod_{j=2}^{b} Pr[\{a_1, \ldots, a_j\} \subseteq R \mid \{a_1, \ldots, a_{j-1}\} \subseteq R] >$$
$$\prod_{j=1}^{b} (2^{d-n} - 2^{j-2-n}) =$$
$$(2^{d-n})^b \cdot \prod_{j=1}^{b} (1 - 2^{j-2-d}) \geq$$
$$2^{b(d-n)} \cdot (1 - 2^{b-d-1}),$$

where the last inequality can be easily proved by induction. □

Proof of Proposition 5 We estimate e as follows. Fix $i \in \{1, \ldots, k\}$ and let $R_1, \ldots, R_t$ be t random d-dimensional subspaces under the restriction that $V_i \subseteq R_j$ for each $j \in \{1, \ldots, t\}$. Since each subspace R_j is of dimension $n-k$, it is easy to see that for $t = n/k = b$, with good probability, $\bigcap_{j=1}^{t} R_j = V_i$ (as every subspace in the sequence is expected to reduce the dimension of the intersection by about k, until the intersection is equal to V_i). Therefore, after about $b \cdot p_c^{-1}$ choices of random subsets, we expect that V_i will be recovered in Line 18. Note that this assumes that there are no false alarms in Line 15, as such false alarms will disrupt the sequence $\bigcap_{j=1}^{t} R_j$, stored in memory.[5] In order to recover all V_i for $i \in \{1, \ldots, k\}$, we need to try about

$$e \approx \log(k) \cdot b \cdot p_c^{-1} < n \cdot p_c^{-1}$$

random subsets of dimension $d = n - k$. □

Proof of Proposition 6 The time complexity of summing over all the d-dimensional subspaces in Line 6 of the attack is $e \cdot 2^d$. Moreover, there are additional operations on the set of candidates that we need to take into account. At the end of the attack, the size of the stored subspaces in $Candidates$ depends on p_f and is about $e \cdot p_f$, in addition to the k subspaces which are not false alarms. Therefore, at the end of the attack $size(Candidates) \approx k + e \cdot p_f \approx e \cdot p_f$. Every candidate subspace R is intersected with all previous subspaces in $Candidates$, and the sum over the intersection R' is computed. In total, the

[5] We also assume that we did not select R such that $V_i \subseteq R$ and $V_j \subseteq R$ for $i \neq j$. This is event is very unlikely and can be ignored.

amount of additional work for the candidates is dominated by Line 14 and is about $(e \cdot p_f)^2 \cdot 2^d$. The total time complexity of the attack is about

$$e \cdot 2^d + (e \cdot p_f)^2 \cdot 2^d = (2^d \cdot e) \cdot (1 + e \cdot p_f^2).$$

□

Proof of Proposition 7 Assume that $d > n-k$, and we want to show that $p_f = 1$. Since R is a linear subspace, the expression $\sum_{y \in R} F(y)$ evaluates a derivative of F of order higher than $n-k$ for each of the n output bits. Therefore, it is sufficient to show that $deg(F) < n-k+1$, which implies that the outcome of the derivation is zero regardless of R. In other words, $Pr[\sum_{y \in R} F(y) = \mathbf{0}] = 1$, and this holds in particular if $V_i \nsubseteq R$ for each $i \in \{1, \ldots, k\}$, implying $p_f = 1$.

The fact that $deg(F) < n-k+1$ is derived from a theorem due to Boura and Canteaut in [6] (stated below in a slightly modified form).

Theorem 1 ([6]). *Let H be a permutation of $\mathbb{F}_2^n$ and let G be a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$. Then we have*

$$deg(G \circ H) < n - \lfloor \frac{n-1-deg(G)}{deg(H^{-1})} \rfloor$$

.

In our case, let $H = L_1 \circ S_0 \circ L_0$ and $G = L_2 \circ S_1$, then $deg(G) = deg(H^{-1}) = b-1$ (as assumed at the beginning of the section). Therefore,

$$\begin{gathered} deg(F) < n - \lfloor \frac{n-1-b-1}{b-1} \rfloor = n - \lfloor \frac{bk-b}{b-1} \rfloor = \\ n - \lfloor \frac{(b-1)(k-1)+(k-1)}{b-1} \rfloor = \\ n - (k-1) + \lfloor \frac{k-1}{b-1} \rfloor = n-k+1, \end{gathered}$$

as $k < b$.

□

Assumptions on False Alarms Assumption 1 states that false alarms do not occur in Line 15 of Algorithm 1. A false alarm in Line 15 occurs in case there are false alarms for both R and the smaller subspace $R' \subseteq R$. This event is significantly less likely than p_f, but may occur nevertheless. Therefore, we relax the assumption and slightly modify the algorithm to deal with such false alarms: in case $s' = 0$ in Line 15, before updating the candidate list we add an additional filtering to test the condition $V_i \subseteq R'$. This is done by selecting at random a vector x which is not in R', and testing whether $\sum_{y \in span(\{x\} \bigcup R')} F(y) = \mathbf{0}$.

The additional filtering will not result in deterioration in performance (again, assuming false alarms in Line 15 do not occur often).

Recall that in the attack we evaluate arbitrary d-order derivatives of F, and therefore the value of p_f in Assumption 2 depends on the density of monomials of degree (at least) d in its algebraic normal form. We select the maximal possible value $d = n - k$ for which it is theoretically possible that $deg(F) \leq d$, or $p_f < 1$ (and in particular, we assume that $p_f < 2^{-n/2}$). Indeed, as shown above, the bound due to [6] implies that $deg(F) < n - k + 1$, but does not rule out the possibility that $deg(F) = n - k$. Moreover, the trivial bound on the algebraic degree of F gives $\deg(S_0) \cdot deg(S_1) = (b-1) \cdot (b-1) \geq (\sqrt{n})^2 = n > n - k$, and does not contradict the possibility that $deg(F) = n - k$.

Our experiments of toy ASASA variants confirm our assumption about p_f. In fact, for ASASA schemes with 2 or 3 Sboxes, $\sum_{y \in R} F(y)$ was almost uniformly distributed when $V_i \nsubseteq R$ for each $i \in \{1, \ldots, k\}$, namely $p_f \approx 2^{-n}$.

3.3 Optimized Integral Attack

The basic integral attack sums the outputs of $e \approx n \cdot 2^n$ subspaces, each containing $2^d = 2^{n-k} \approx 2^n$ elements. Therefore, its total time complexity is about $n \cdot 2^{2n}$. The complexity of summing over the subspaces can be significantly reduced if we choose correlated subspaces instead of picking them at random. More specifically, as we show next, it is possible to sum over the outputs of $2^{n/2}$ carefully chosen subspaces in about 2^n time. This reduces the complexity of the attack to about $n \cdot 2^{3n/2}$ under the assumption that $p_f < 2^{-3n/4}$, which is stronger than the assumption made in the basic attack.

One way to optimize the summation process is to divide the n-bit block into two equal halves (assuming n is even). First, for each value of the $n/2$ most significant bits (MSBs), compute the sums over the outputs of all possible $2^{n/2}$ values of the $n/2$ least significant bits (LSBs). This gives an array of partial sums of size $2^{n/2}$ which is computed in time 2^n. Next, choose a subspace of dimension $d - n/2$ from the $n/2$ MSBs, and sum over the outputs of the bigger subspace of dimension d that includes the $n/2$ LSBs. Using the recomputed array, this can be done in time $2^{d-n/2}$. Repeating the process for $2^{n/2}$ subspaces of dimension $2^{d-n/2}$, the sums over the outputs of all of them can be computed in about 2^d time using the precomputed array. In total, we sum over the outputs of $2^{n/2}$ subspaces in about 2^n time, as claimed.

In general, instead of dividing the bits of the block into two halves, we can work with two orthogonal subspaces of dimension $n/2$. The pseudocode of the general procedure is given in Algorithm 2.

We consider slightly modify definitions 3 and 4 which refer to subspaces selected according to the procedure of Algorithm 2. The analysis at the end of this section shows that the covering probability of the algorithm p_c remains roughly 2^{-n}, and thus the attack still requires evaluating $e \approx n \cdot 2^n$ subspaces. The sums

Input : An integer $d > n/2$
Output: Sums on outputs of $2^{n/2}$ d-dimensional subspaces s_j

begin
  Pick a subspace T of dimension $n/2$ at random;
  `// Array of` $2^{n/2}$ `auxiliary sums`
  $A \longleftarrow A[1, \ldots, 2^{n/2}]$;
  `// Iterate over the` $n/2$`-dimensional subspace orthogonal to` T
  for ***each*** $x \in T_\perp$ **do**
    `// A is accessed using an` $n/2$`-bit representation of` x
    $A[x] \leftarrow \sum_{y \in T} F(x + y)$;
  for $j \leftarrow 1$ **to** $2^{n/2}$ **do**
    Pick a subspace $T' \subset T_\perp$ of dimension $d - n/2$ at random;
    $s_j \leftarrow \sum_{x \in T'} A[x]$;

Algorithm 2: Efficient Sum over Subspaces

over these subspaces are computed by running the procedure of Algorithm 2 about $n \cdot 2^{n/2}$ times.[6]

Recall form Proposition 6 that the work spent on false alarms is about $2^d \cdot (e \cdot p_f)^2 \approx 2^{3n} \cdot (p_f)^2$. Therefore, in order to run in the claimed time complexity of about $n \cdot 2^{3n/2}$, we need to strengthen Assumption 2, and assume that the false alarm probability satisfies $p_f < 2^{-3n/4}$ (and this assumption was supported by our experiments, as noted above).

Analysis of Covering Probability Fix a subspace T of dimension $n/2$ and write $V_i = W_i \oplus W_i'$, where $W_i \subseteq T$ and $W_i' \subseteq T_\perp$. For a random subspace $T' \subseteq T_\perp$ of dimension $d - n/2$, $V_i \subseteq T \oplus T'$ if and only if $W_i' \subseteq T'$. In order to estimate p_c, we distinguish between two cases according to the Sbox size b: either $b = n/2$, or $b \leq n/3$. If $b \leq n/3$, the worst (and likely) case for our attack is $dim(W_i') = dim(V_i) = b$, and in this case (reusing the analysis of Proposition 4) p_c remains about 2^{-n} for our choice of T' of dimension $d' = n/2 - k$ (up to the small multiplicative factor $1 - 2^{b-d'-1} = 1 - 2^{b+k-n/2-1} \approx 1$.

For $b = n/2$ the previous worst-case analysis assumes $dim(W_i') = dim(V_i) = n/2$ and gives $p_c = 0$, as we need to cover the $n/2$-dimensional subspace W_i' with a smaller subspace T' of dimension $n/2 - 2$. It is possible to show that in this case p_c remains roughly 2^{-n} using the randomness in the choice of T. However, it is somewhat simpler to slightly tweak the algorithm, and use subspaces T of dimension $n/2 - 3$, implying $dim(T_\perp) = n/2 + 3$ and $d' = dim(T') = d - (n/2 - 3) = n/2 + 1$. Therefore, even if $dim(W_i') = dim(V_i) = n/2$, then $p_c \geq$

[6] Note that in order to recover the final subspaces V_i, the base $n/2$-dimensional subspace T in the full attack must be randomized frequently enough (as done in our algorithm), since the intersection of d-dimensional subspaces produced with the same choice of T will always contain T.

$2^{-n} \cdot (1 - 2^{b-d'-1}) = 2^{-n} \cdot (1 - 2^{-2}) \approx 2^{-n}$. The tweaked algorithm sums over $2^{n/2}$ subspaces of dimension 2^{n-2} in time $2^{n/2} \cdot 2^{d'} = 2^{n+1}$ which it is slower compared to what is claimed, but only by a small multiplicative factor of 2.

4 Boomerang Attack

4.1 Distinguishing Boomerang Attack on the Core $S_1 \circ L_1 \circ S_0$

The boomerang attack [16], introduced by Wagner in 1999, allows for breaking a cipher with short high-probability differentials (rather than one long low-probability differential). When we consider the core of the ASASA construction, the $S_1 \circ L_1 \circ S_0$ layer, it can be easily identified that very good short differentials exist for the S_0 and the $S_1 \circ L_1$ layers.[7] For each such layer, it is easy to see that many single active Sbox differentials exist.

Consider the first layer S_0, it is easy to see that one can pick any of the $k \cdot (2^b - 1)$ possible input differences composed of a single active Sbox, and obtain differentials (which in this degenerate case correspond to differential characteristics), each with probability of at least $2 \cdot 2^{-b}$. Similarly, one can find $k \cdot (2^b - 1)$ possible output differences composed of a single active Sbox in the second layer. Again, each such differential has a probability of at least $2 \cdot 2^{-b}$.

One advantage of the boomerang attack, is that once the input difference to the first layer is fixed, or once the output difference of the second layer is fixed, one can use multiple differentials (see [2] for the full analysis). So a simple boomerang distinguisher for the core of $S_1 \circ L_1 \circ S_0$ can be easily constructed as follows:

- Pick an input difference α with one active Sbox,
- Pick an output difference δ with one active Sbox,
- Generate $c \cdot 2^{2b}$ boomerang quartets (pick a pair with input difference α, encrypt, XOR each ciphertext with δ, and decrypt the newly obtained ciphertexts), and expect a boomerang quartet.

We note that for a random permutation, in $c \cdot 2^{2b}$ boomerang quartets, we expect $cd \cdot 2^{2b} \cdot 2^{-bk}$ quartets such that the newly decrypted plaintexts satisfy that their difference is α (as this is an bk-bit condition). For the core we discuss the probability of a quartet to become a right one is:

$$
\begin{aligned}
p_{boomerang} &= \sum_{\beta,gamma} \Pr^2[\alpha \xrightarrow{S_0} \beta] \cdot \Pr^2[\gamma \xrightarrow{S_1 \circ L_1} \delta] \\
&= \underbrace{\left(\sum_{\beta} \Pr^2[\alpha \xrightarrow{S_0} \beta] \right)}_{(*)} \cdot \underbrace{\left(\sum_{\gamma} \Pr^2[\gamma \xrightarrow{S_1 \circ L_1} \delta] \right)}_{(**)} \\
&\geq 2^{-b+1} \cdot 2^{-b+1} = 2^{-2b+2}
\end{aligned}
$$

[7] We note that we can decompose the core into $L_1 \circ S_0$ and S_1, and obtain essentially the same results.

The last transition follows the fact that the sums $(*)$ and $(**)$, reach minimal value for Sboxes which are 2-differentially uniform, i.e., for any given non-zero input (or output) difference to (from) the Sbox, there are exactly 2^{b-1} possible output (input) differences, each with probability $2 \cdot 2^{-b}$.

Hence, for $c = 1/4$, we expect one right boomerang quartet for the core $S_1 \circ L_1 \circ S_0$. Moreover, as the actual number of right quartets follows a Poisson distribution, for $c = 1/4$, we indeed expect to get (at least) one quartet with probability 63%. Obviously, increasing c allows a better success rate.

To conclude, there are several (actually, $(k \cdot (2^b - 1))^2$) boomerang distinguishers for the core. Each such distinguisher can be applied using $1/4 \cdot 2^{2b}$ quartets, i.e., 2^{2b} adaptive chosen plaintexts and ciphertexts. The identification of the right quartets is immediate, and the memory complexity is restricted to storing a single quartet each time.

4.2 Extending the Attack to $L_2 \circ S_1 \circ L_1 \circ S_0$

The main problem that prevents a simple adaptation of the above attack to two full layers is the fact that due to the L_2 layer, we cannot identify by which δ we need to XOR the ciphertexts to obtain the new ciphertexts. This prevents an adaptive chosen plaintext and ciphertext attack, and forces us to use a chosen plaintext attack (as we can still control the α difference). To this end, we just transform the above boomerang attack into an amplified boomerang attack [9] (or the rectangle attack [2]).

The amplified boomerang attack starts with pairs of plaintexts $(P_i, P_i' = P_i \oplus \alpha)$, encrypted into (C_i, C_i'). If the amplified boomerang condition holds for some quartet $((P_i, P_i'), (P_j, P_j'))$, then we know that the partial encryption of P_i and P_j through $S_1 \circ L_1 \circ S_0$ have difference δ (of one active Sbox), and that the same holds for P_i', P_j'. Unlike the case of the boomerang attack on the core, we do not have the differences $P_i \oplus P_j$ and $P_i' \oplus P_j'$ exposed to us as δ. However, we know that L_2 is a linear transformation, namely,

$$\begin{aligned}
S_1(L_1(S_0(P_i))) \oplus S_1(L_1(S_0(P_j))) = \delta &\Rightarrow L_2(S_1(L_1(S_0(P_i)))) \oplus L_2(S_1(L_1(S_0(P_j)))) = L_2(\delta) \\
&\Rightarrow C_i \oplus C_j = L_2(\delta) \\
S_1(L_1(S_0(P_i'))) \oplus S_1(L_1(S_0(P_j'))) = \delta &\Rightarrow L_2(S_1(L_1(S_0(P_i')))) \oplus L_2(S_1(L_1(S_0(P_j')))) = L_2(\delta) \\
&\Rightarrow C_i' \oplus C_j' = L_2(\delta) \\
&\Rightarrow C_i \oplus C_j = C_i' \oplus C_j' \Rightarrow C_i \oplus C_i' = C_j \oplus C_j'
\end{aligned}$$

The result, is an amplified boomerang attack which takes $c \cdot 2^{n/2+b}$ pairs with input difference α, and searches for the right quartets, which can be identified by the fact that for right quartets, the above condition (which can be checked by analyzing a pair $(P_i, P_i' = P_i \oplus \alpha)$ and storing in a hash table the value $C_i \oplus C_i'$ of the corresponding ciphertexts).

The analysis of the number of right amplified quartets is relatively straightforward, and we obtain that of the $c^2 \cdot 2^{n+2b}$ possible quartets[8] about $4c^2$ are right quartets. Hence, for the correct value of $L_2(\delta)$ we expect to encounter $4c^2$ quartets (identified as collision in the hash table).

Given the large number of quartets, we expect wrong quartets to offer collisions in the hash table. There are going to be (about) additional $c^2 \cdot 2^{2b}$ such collisions in the table, but as they happen randomly, they are going to be scattered over the 2^n possible $L_2(\delta)$ values. Hence, as long as $4c^2 \cdot \gg c^2 \cdot 2^{2b-n}$, we expect the right value to be identified.

The result is an attack that takes $c \cdot 2^{n/2+b}$ chosen plaintexts, time, and memory, and identifies $L_2(\delta)$. We later show how to transform this knowledge into (partial) key recover attack on L_2.

A Small (and somewhat insignificant) Technicality We note that the probability estimation that we have used assumes that the Sbox in use is differentially 2-uniform. Obviously, if the Sboxes are chosen at random, it is highly unlikely that this condition holds. In these circumstances, the probability of the boomerang $p_{boomerang}$ is actually slightly higher.

We can use [13] to evaluate the way the difference distribution table behaves for an Sbox chosen at random. Namely, an entry in the difference distribution table of an b-bit Sbox (besides those involved with input/output zero), has a value distributed according to $2 \cdot Poi(1/2)$. As a result, the sums $(*)$ and $(**)$ are expected to obtain the value:

$$\begin{aligned}(*) &= \sum_{\beta \neq 0} Pr^2[\alpha \xrightarrow{S_0} \beta] = \left(2^b - 1\right) \cdot 2^{-2b} \cdot \sum_{i=0}^{2^{b-1}} (2 * i)^2 \cdot e^{-1/2} \cdot (1/2)^i / i! \\ &\approx 3 \cdot 2^{-b}\end{aligned}$$

(compared with the value $2 \cdot 2^{-b}$ which is a lower bound).

A Small (but Important) Technicality We note that the amplified boomerang attack is being run in parallel for all $k \cdot (2^b - 1)$ possible values of δ, each resulting in about $4c^2$ quartets. Hence, we can obtain almost an exhaustive list of all the output differences of L_2 that originate from a single active Sbox.

Due to the nature of the amplified attack, we *do not* need additional data to find all these values. They are just suggested by the different boomerangs. The only thing we need to take into account is that we want all $k \cdot (2^b - 1)$ boomerangs to "succeed" (i.e., have enough quartets). In Table 2 we give an example of parameters that follows AES' ones ($k = 16, b = 8, n = 128$).

One additional technicality is the fact that there is some (non-zero) chance that differential characteristics with *two* active Sboxes in the second-layer, may

[8] We refer the interested reader to [2] for the full analysis, e.g., why there are $c^2 \cdot 2^{n+2b}$ rather than $c^2/2 \cdot 2^{n+2b}$ quartets.

also be encountered. A simple analysis reveals that the expected number of quartets for a given two-active-Sboxes differences is $4c^2 \cdot 2^{-b}$. Moreover, there are $\binom{k}{2} \cdot (2^b - 1)^2$ possible such differences. Hence, to avoid too much noise from these cases we must demand that about $k^2/2 \cdot 2^{2b}$ random variables following a Poisson distribution with a mean value of about $4c^2 \cdot 2^{-b+1}$ will not interfere with the correct differences (whose number of right quartets follows a Poisson distribution with a mean value of $4c^2$).[9] Luckily, it is easy to see that for reasonable values of c, the problem is far from causing trouble. Obviously, the problem generalizes to more active Sboxes in the second layer, but it is (again) of no affect on the main result. We list in Table 2 also the expected number of two-active Sboxes differences that might pass for a cipher with AES' parameters.

c	Expected Number of Quartets	Threshold (for ident.)	Discovered differences	
			Single Sbox (out of 4,080)	Two Sboxes (out of 7,803,000)
1	4	1	4,005	240,073
		2	3,706	3,732
		3	3,109	39
		4	2,311	0.3
2	16	3	4,079	2,313
		4	4,079	72
		5	4,078	2
		6	4,074	0.03

Table 2. Number of differences discovered by the (amplified boomerang) attack as a function c for AES' parameters ($k = 16, b = 8, n = 128$).

This leads to the following problem: once we have all these $k \cdot (2^b - 1)$, can we recover all (or part) of L_2? which we address next.

Partial Key Recovery At this point, we obtain $k \cdot (2^b - 1)$ output differences $L_2(\delta_{i,j})$ where $\delta_{i,j}$ is a difference of j in the i'th Sbox (and zero in all other Sboxes). We now show how to divide these difference into the k different Sboxes (as the order between them can be fixed arbitrarily, given that any order can be "adjusted" by using L_1).

[9] We alert the reader that it is easy to overcome a "small" number of two-active-Sboxes differences in the other steps of the attack, and as we see later, they may even be useful for the attack. Moreover, one can start the exploration of the differences with those that are suggested by as many quartets as possible. This values are very likely to be with a single active Sbox in S_2. Using the fact that if all non-zero differences of a single Sbox form a linear subspace, one can check whether differences with a smaller amount of quartets is actually of single active Sbox, and even complete the space for values that received no quartet.

The simplest method to do so, is to actually increase a bit the data/time complexity, and look for two active Sboxes in the second layer. It is easy to see that once we identify all $\delta_{i,j}$'s values, then a difference in two active Sboxes can be written as $\Delta = \delta_{i,j} \oplus \delta_{i',j'}$, i.e., $L_2(\Delta) = L_2(\delta_{i,j} \oplus \delta_{i',j'}) = L_2(\delta_{i,j}) \oplus (\delta_{i',j'})$. This holds only when $i \neq i'$, and thus, we can identify when two of the one active Sbox differences do not share an Sbox (as there will be a corresponding two active Sbox difference for them). Hence, the separation into different Sboxes can be done quite immediately, resulting in an attack that requires $c \cdot 2^{n/2+3b/2}$ chosen plaintexts, and about the same amount of time.

4.3 Attacking the Full Structure

We now turn our attention to the full ASASA construction. The main problem with directly applying the amplified boomerang attack to the full construction is that we cannot have a difference in the first Sbox layer in only a single Sbox.

Luckily, the solution to the problem is to perform another birthday paradox argument, and use a known-plaintext boomerang. The known-plaintext boomerang was first mentioned in [16], and its a natural extension of the boomerang (and the amplified boomerang attack). A random set of $c \cdot 2^{3n/4+b/2}$ known plaintexts, contains $c^2/2 \cdot 2^{3n/2+b}$ pairs at the entrance to S_0, i.e., offers $c^2/2 \cdot 2^{n/2+b}$ pairs with any given input difference, including those with a single active Sbox. As shown earlier, such amount of pairs is sufficient to generate right amplified boomerang quartets.

The only remaining task is the identification of the quartets $((P_1, P_2), (P_3, P_4))$ themselves. If both (P_1, P_2) and (P_3, P_4) are pairs which are part of the quartet, in other words, have the same difference in a single byte after L_0 (i.e., $L_0(P_1 \oplus P_2) = L_0(P_3 \oplus P_4)$ with a single active Sbox), then $P_1 \oplus P_2 = P_3 \oplus P_4$. As before, we can also detect that $L_2(C_1) \oplus L_2(C_2) = L_2(C_3) \oplus L_2(C_4)$. This suggests that finding the quartets $((P_1, P_2), (P_3, P_4))$ can be easily done by:

- For all pairs of plaintext (P_i, P_j) store in a hash table $P_i \oplus P_j || C_i \oplus C_j$ (along with P_i and P_j).
- Collect all collisions in the table as candidate quartets, and analyze them as before.

Given $c \cdot 2^{3n/4+b/2}$ known plaintexts, we expect $c^2/2 \cdot 2^{3n/2+b}$ pairs, and in total $c^4/2 \cdot 2^{3n+2b}$ quartets.[10] Hence, we expect $c^4/2 \cdot 2^{n+2b}$ quartets to be suggested by the collisions in the table, out of which $c^4/2$ are right quartets that suggest the correct differences in the plaintexts and in the ciphertexts. These differences are of course, differences that become an active single Sbox (either through L_0 or the inverse of L_2). Hence, once again, it is possible to identify all the differences that are transformed into a single active Sbox (on both sides of the scheme). Again, increasing the data complexity a bit (to $c \cdot 2^{3n/4+3b/4}$ known

[10] We note that the quartet $((P_1, P_2), (P_3, P_4))$ differs from the quartet $((P_1, P_2), (P_4, P_3))$.

plaintexts), allows finding all the differences that go to a single active Sbox, by working with differences of two active Sboxes.

To conclude, one can easily identify the $k \cdot (2^b - 1)$ differences that lead to a single active Sbox thorough L_0 or L_2, using a known plaintext boomerang. The attack takes $c \cdot 2^{3n/4+3b/4}$ plaintexts, and has memory and time complexity of $c^2/2 \cdot 2^{3n/2+3b/2}$.

5 Differential Attack: Using the DDT

We denote by

$$\delta(\alpha) := |\{F(x) + F(x+\alpha) \mid x \in \mathbb{F}_2^n\}|$$

the number of possible output differences for a given input difference.

The basic idea is that the number of possible output differences depends on the number of active Sboxes in the first layer of Sboxes. More precisely, we rely on the following assumption.

Assumption 3 *The number of possible output differences is (expected to be) smaller if only one Sbox is active in the first layer compared to the case when more than one Sbox is active in the first layer.*

So the attack starts by computing $\delta(\alpha)$ for all non-zero α values. This takes time 2^{2n} and will be the bottleneck of the attack. Afterwards the list of all $\delta(\alpha)$ is sorted in increasing order. We denote the sorted list by $\mathcal{T}$ with $\mathcal{T}_0$ corresponding to α with the smallest $\delta(\alpha)$ value.

The assumption is now that the top of the list contains mostly input-differences α such that

$$L(\alpha) = (\beta_1, \ldots, \beta_k)$$

with only one $\beta_i \in \mathbb{F}_2^b$ non-zero. In other words, the top of the list contains mainly elements α such that $\alpha \in V_i$ for some (unknown) $1 \leq i \leq k$.

For recovering V_i we next have to sort those α values into different bins such that two α values are in the same bin iff they are both in the same subspace V_i.

We explain how to recover the first V_i, without loss of generality V_0 , in detail. Recovering the remaining ones is very similar and will only be briefly sketched.

We start by considering the first t entries in the sorted table $\mathcal{T}$. Choosing t such that $\binom{t}{b}$ is smaller than 2^{2n} makes sure that this step is not the bottleneck of the attack. In most cases (as long as $b \leq 2^{2k}$) it would be sufficient to choose $t \leq 2^{2k}$, as

$$\binom{2^{2k}}{b} \leq \left(2^{2k}\right)^b = 2^{2n}.$$

However, our experiments show that the success probability of the attack is reduced for high values of t. The best results have been obtained when t was set to a small multiple of b.

The idea now is to identify a basis of V_0. For any choice of b (linearly independent) elements $\alpha_1, \ldots, \alpha_b$ among the first t elements of $\mathcal{T}$ we compute a penalty value

$$P(\{\alpha_1, \ldots, \alpha_b\}) := \sum_{u \in \text{span}(\{\alpha_1, \ldots, \alpha_b\}), u \neq 0} \delta(u) \ ,$$

According to assumption 3 there are now two cases to be considered. First, if all α_i values actually belong to V_0 the penalty $P(\{\alpha_1, \ldots, \alpha_b\})$ is simply the sum over all $\delta(u)$ values for $u \in V_i$ and thus expected to be small. Second, if at least one α_j does not belong to V_0, at least $(2^{b-1} - 1)$, that is roughly half, of the elements in $\text{span}(\{\alpha_1, \ldots, \alpha_b\})$ do not belong to any V_i and thus the penalty value is expected to be significantly higher.

Thus, this procedure allows to identify a basis of V_0. As will be shown below in the experimental results, this is successful with very good probability.

After recovering V_0 we simply remove all elements in V_0 from the list $\mathcal{T}$ and repeat the same procedure again. Experimentally, the overall success probability, i.e., the probability that we recover all V_i correctly is again rather high, as shown in Table 3 below.

b \ k	2	3	4	5
3	0.16	0.06	0.02	0
4	0.78	0.92	1	-
5	1	1	-	-
6	1	-	-	-
7	1	-	-	-
8	1	-	-	-

Table 3. Result of 100 runs of the attack for various parameters. In each run a new random ASASA instance has been generated. The numbers correspond to the success probability to recover all V_i correctly. The dashes correspond to experiments we did not run due to limited resources.

It can be seen that the success probality increases for increasing k and b with the exception of $b = 3$. The reason for this exception might be linear components that occur in the randomly chosen 3-bit Sboxes.

6 Conclusion

In this paper we analyzed efficient decomposition algorithms for the ASASA structure. In particular, we described three different attack vectors, two based on differential properties and one on an integral - i.e. a structural - property. In all the attacks, the difficulty lies not in exploiting the properties, but rather

in locating them as they are hidden by the external linear layers. Reciprocally, as soon as such a property has been detected, information on the linear layers can be easily deduced, and combining the data gathered from several properties allows to peel off the external linear layers efficiently. While our integral attack is the most efficient, it is not always applicable, and in such cases the other attacks should be considered. Moreover, we feel that the boomerang and differential attack vectors provide additional value, which could be further increased in the future if they are improved.

As our most efficient attacks have time complexity of roughly $2^{3n/2}$, it seems that ASASA schemes must have a large block size n in order to be considered as secure candidates for white-box cryptography. However, this requires their black-box representations to be extremely large and inappropriate for practical use (e.g., in order to guarantee a minimal security level of 64 bits, an ASASA scheme requires storage of about 2^{43} words, which is more than 10 terabytes). A natural future work item is to study the security of general SP-networks with more layers (e.g. ASASAS), and in particular, investigate if any of the attacks presented here can be generalized or improved on these constructions.

References

1. Eli Biham. Cryptanalysis of Patarin's 2-Round Public Key System with S Boxes (2R). In Bart Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, volume 1807 of *Lecture Notes in Computer Science*, pages 408–416. Springer, 2000.
2. Eli Biham, Orr Dunkelman, and Nathan Keller. The Rectangle Attack - Rectangling the Serpent. In Birgit Pfitzmann, editor, *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, volume 2045 of *Lecture Notes in Computer Science*, pages 340–357. Springer, 2001.
3. Alex Biryukov, Charles Bouillaguet, and Dmitry Khovratovich. Cryptographic Schemes Based on the ASASA Structure: Black-Box, White-Box, and Public-Key (Extended Abstract). In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, volume 8873 of *Lecture Notes in Computer Science*, pages 63–84. Springer, 2014.
4. Alex Biryukov and Adi Shamir. Structural Cryptanalysis of SASAS. *J. Cryptology*, 23(4):505–518, 2010.
5. Julia Borghoff, Lars R. Knudsen, Gregor Leander, and Søren S. Thomsen. Slender-Set Differential Cryptanalysis. *J. Cryptology*, 26(1):11–38, 2013.
6. Christina Boura and Anne Canteaut. On the Influence of the Algebraic Degree of F^{-1} on the Algebraic Degree of G $\circ$ F. *IEEE Transactions on Information Theory*, 59(1):691–702, 2013.
7. Joan Daemen, Lars R. Knudsen, and Vincent Rijmen. The Block Cipher Square. In Eli Biham, editor, *Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings*, volume 1267 of *Lecture Notes in Computer Science*, pages 149–165. Springer, 1997.

8. Henri Gilbert, Jérôme Plût, and Joana Treger. Key-Recovery Attack on the ASASA Cryptosystem with Expanding S-boxes. In *CRYPTO 2015*, Lecture Notes in Computer Science. Springer, 2015. to appear.
9. John Kelsey, Tadayoshi Kohno, and Bruce Schneier. Amplified Boomerang Attacks Against Reduced-Round MARS and Serpent. In Bruce Schneier, editor, *Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings*, volume 1978 of *Lecture Notes in Computer Science*, pages 75–93. Springer, 2000.
10. Lars R. Knudsen. Truncated and Higher Order Differentials. In Bart Preneel, editor, *Fast Software Encryption: Second International Workshop. Leuven, Belgium, 14-16 December 1994, Proceedings*, volume 1008 of *Lecture Notes in Computer Science*, pages 196–211. Springer, 1994.
11. Lars R. Knudsen and David Wagner. Integral Cryptanalysis. In Joan Daemen and Vincent Rijmen, editors, *Fast Software Encryption, 9th International Workshop, FSE 2002, Leuven, Belgium, February 4-6, 2002, Revised Papers*, volume 2365 of *Lecture Notes in Computer Science*, pages 112–127. Springer, 2002.
12. Xuejia Lai. Higher Order Derivatives and Differential Cryptanalysis. In *"Symposium on Communication, Coding and Cryptography", in honor of James L. Massey on the occasion of his 60'th birthday*, pages 227–233, 1994.
13. Luke O'Connor. On the Distribution of Characteristics in Bijective Mappings. *J. Cryptology*, 8(2):67–86, 1995.
14. Jacques Patarin and Louis Goubin. Asymmetric cryptography with S-Boxes. In Yongfei Han, Tatsuaki Okamoto, and Sihan Qing, editors, *Information and Communication Security, First International Conference, ICICS'97, Beijing, China, November 11-14, 1997, Proceedings*, volume 1334 of *Lecture Notes in Computer Science*, pages 369–380. Springer, 1997.
15. Tyge Tiessen, Lars R. Knudsen, Stefan Kölbl, and Martin M. Lauridsen. Security of the AES with a Secret S-box. In *FSE 2015*, Lecture Notes in Computer Science. Springer, 2015. to appear.
16. David Wagner. The Boomerang Attack. In Lars R. Knudsen, editor, *Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings*, volume 1636 of *Lecture Notes in Computer Science*, pages 156–170. Springer, 1999.