2020, Volume 7, Number 2 (pp. 103–117)
http://www.jcomsec.org

Journal of Computing and Security

Toward A More Efficient Gröbner-based Algebraic Cryptanalysis

Hossein Arabnezhad-Khanoki [a], Babak Sadeghiyan [a,*]

[a] *Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran.*

ARTICLE INFO.

Article history:
Received: 24 June 2020
Revised: 15 July 2020
Accepted: 16 August 2020
Published Online: 4 October 2020

Keywords:
Algebraic Cryptanalysis, Gröbner Basis, Universal Proning, S-Box Representation

ABSTRACT

In this paper, we propose a new method to launch a more efficient algebraic cryptanalysis. Algebraic cryptanalysis aims at finding the secret key of a cipher by solving a collection of polynomial equations that describe the internal structure of the cipher. Chosen correlated plaintexts, as what appears in *higher order differential* cryptanalysis and its derivatives such as cube attack or integral cryptanalysis, forces many linear relations between intermediate state bits in the cipher. In this paper, we take these polynomial relations into account, so it becomes possible to simplify the equation system arising from algebraic cryptanalysis, and consequently, solve the polynomial system more efficiently. We take advantage of the Universal Proning technique to provide an efficient method to recover such linear polynomials. Another important parameter in the algebraic cryptanalysis of ciphers is to effectively describe the cipher. We employ the so-called *Forward-Backward* representation of S-boxes together with Universal Proning to help provide a more powerful algebraic cryptanalysis based on Gröbner-basis computation. We show our method is more efficient than doing algebraic cryptanalysis with MQ representation, and also than employing MQ together with Universal Proning. To show the effectiveness of our approach, we applied it for the cryptanalysis of several lightweight block ciphers. By this approach, we managed to mount algebraic attack on 12-round LBlock, 6-round MIBS, 7-round PRESENT and 9-round SKINNY light-weight block ciphers, so far.

1 Introduction

Algebraic cryptanalysis aims at finding the secret key of the cipher by solving the collection of polynomial equations that describes the cipher, usually in a known plaintext or chosen plaintext scenario. In general, the algebraic analysis takes two stages. In the first stage, a cipher is described by a system of equations. In the second stage, the system is solved using an ”appropriate” algorithm. There are many algorithms to solve such a system of equations, where the computation of Gröbner basis is one of such an approach. It is already well-known that the way a cipher is represented with a system of equations has impacts on the running time to obtain its solution employing a Gröbner-basis computation [1].

Algebraic cryptanalysis of block ciphers in the chosen plaintext scenario, leads to a more efficient cryptanalysis. In [1–4] a series of algebraic attacks on block ciphers were proposed, which all are based on highly

* Corresponding author.

Email addresses: arabnezhad@aut.ac.ir (H. Arabnezhad), basadegh@aut.ac.ir (B. Sadeghiyan)

https://dx.doi.org/10.22108/jcs.2020.123673.1050
ISSN: 2322-4460 © 2020 JComSec. All rights reserved.

correlated plaintexts. Some other successful cryptanalysis techniques of block ciphers are also based on correlated plaintexts, such as differential cryptanalysis [5], integral cryptanalysis or square attack[6], cube attack [7] and recently division cryptanalysis [8].

It is already known that highly structured plaintexts such as what appears in integral or cube attacks, impose some correlation between intermediate state bits with different plaintexts in the structure. For example, the multi-set of correlated plaintexts in integral cryptanalysis, or cubes in cube attack, cause the sum of some intermediate states bits overall plaintexts be a constant value, for some number rounds.

The idea in this paper is to use such relations to improve algebraic attacks that are based on computation of Gröbner basis. In integral cryptanalysis, these relations are computed by a specific algebra that is defined for the propagation of these relations on a multi-set through the block cipher. In the cube attack, these relations are described as Boolean polynomials. With the probabilistic BLR linearity test [9] or its generalized form [7, 10], it is possible to mark off them. Then if such a relation has been found to exist, the polynomial is recovered using other algorithms introduced in [7, 10]. *Balancedness* is one of the properties that integral cryptanalysis examines. Balancedness defines that the sum of some intermediate variables for all vectors in the muli-set is equal to zero. This property is attained by constant superpoly in a cube attack. Instead of the conventional methods to recover such polynomials for these attacks, we use the Universal Proning technique [11].

After recovering polynomials, we add them to the system that describe the block cipher. We found that using these polynomials in combination with FWBW representation of S-boxes allows a more efficient algebraic cryptanalysis.

In this paper, we propose an improved Gröbner basis based algebraic cryptanalysis, with employing FWBW representation together with Universal Proning technique to achieve a more efficient algebraic cryptanalysis.

Contributions: To show the efficiency of our proposed method, we also employed our improved Gröbner basis based algebraic cryptanalysis on LBlock [12], MIBS [13], PRESENT [14] and SKINNY [15]. The main contributions of the work are as follows:

(a) proposing a new method to launch a more efficient algebraic cryptanalysis, with FWBW representation of S-boxes and Universal Proning.

(b) proposing a framework for evaluation of algebraic attacks on light-weight ciphers.

(c) presenting first algebraic attack on 12 rounds of LBlock

(d) presenting first algebraic attack on 8 and 9 rounds of SKINNY.

(e) presenting first algebraic attack on 7 rounds of PRESENT.

(f) finding some unbalanced algebraic property for encryption and decryption of SKINNY family of ciphers.

The paper is organized as follows: In Section 2, we review the higher-order differential cryptanalysis and its derivations, *i.e.* integral cryptanalysis and cube attacks. In Section 3, we review some different S-box representations for algebraic cryptanalysis. In section4, we discuss Universal polynomials and Universal Proning. In Section 5, we review some algebraic attacks in the literature, and report our results for cryptanalysis for four light-weight ciphers. We give conclusions and future research directions in Section 6.

2 Higher Order Differential Cryptanalysis

Higher order differential is a generalization of ordinary differential cryptanalysis and it is introduced in [16]. Let define XOR as the group operation, then higher-order derivative of binary functions is defined as follows:

Proposition 1 ([16]). *Let $L[a_1, a_2, \dots, a_i]$ be the list of all possible linear combinations of a_1,a_2,...,a_i. Then,*

$$\Delta^{(i)}_{a_1,a_2,\dots,a_i} f(x) = \sum_{c\in L[a_1,a_2,\dots,a_i]} f(x \oplus c)$$

defines the higher order derivative of f on L.

Integral cryptanalysis and cube attack methods somehow take advantage of higher-order derivative of binary functions.

2.1 Integral cryptanalysis

The square or integral attack [17] is first proposed as a dedicated attack for the Square cipher [17]. The technique study propagation of the sum of intermediate values through the block cipher. The name integral cryptanalysis coined by Knudsen et. al in [6]. In [8], Todo introduced generalized integral property as division property, which not only considers summation of variables but also summation of monomials of higher degree for example two.

We just review the idea of integral property. Suppose intermediate values during the computation of block cipher are represented by a Boolean vector. Let S be a multi-set of vectors v. The integral over the multi-set S is defined as the sum of all vectors in S. Considering word-based block ciphers such as AES, the intermediate state is divided into n words. Attacker aims to predict the integrals after some number of rounds of encryption. Three cases may be distinguished for the word i of the intermediate state vectors.

case 1. For all v in S, we have $v_i = c$. where c is a fixed value (known or unknown). This condition is denoted by $\mathcal{C}$

case 2. The set of v_i's takes all possible values, for all v in S . This condition is denoted by $\mathcal{A}$

case 3. The sum v_i always lead to fixed value, usually zero. This is denoted by $\mathcal{S}$

The polynomial expression of the first case would be the set of polynomials such that:

$$\{\forall j : v_i^0 = v_i^j\}$$

The second and third cases could be expressed by the polynomial such that:

$$\sum_{j=0}^{m} v_i^j = 0$$

2.2 AIDA/Cube Cryptanalysis

In block ciphers, any bit of ciphertext could be represented with a polynomial p on plaintext bits and key variables.

$$c_i = p_i(x_1, ..., x_n, k_1, ..., k_m)$$

where variables x_i denote plaintext bits and variables k_i denote key bits. These polynomials are of high degree and have an enormous number of monomials. Let I be a set of indexes for plaintext variables and t_I be the monomial from product of the variables with indexes in I. Then polynomial p_i could be rewritten as follows:

$$p_i(x_1, ..., x_n, k_1, ..., k_m) = t_I p_{S(I)} + q(x_1, ..., x_n, \\ k_1, ..., k_m)$$

where $p_{S(I)}$ is called superpoly of t_I in p, and monomials in p does not share a common variable with t_I. The polynomial q does not have a monomial that contains all of the variables in t_I.

Cube attack main observation. Let p_I denote the sum of polynomial p over all possible 0/1 assignment to variables with indexes in I. Then we have $p_I = p_{S(I)}$ (Theorem 2 in [7]).

Given the explicit description of p_i, it would be easy to factor the term t_I, but usually due to the cryptographic properties of block ciphers the polynomial is already unknown or have an exponential length representation. The cube attack provides an efficient way to manipulate these polynomials implicitly or as a black box.

If the $p_{S(I)}$ is linear polynomial or of small degree, we could easily compute the $p_{S(I)}$ through the computation of p_I.

Attacker fixes the public variables that does not appear in t_I and then sum p_i over all possible 0/1 assignments to variables in t_I.

One problem that arises here is that degree of $p_{S(I)}$ is not known a priori. Hopefully, with BLR test [9] it is possible to check the linearity of a polynomial with implicit description.

The test for linearity of the polynomial $p_{S(I)}$ is as follows: If for a random assignments x and y to secret variables, the following test is satisfied with a good probability (> 0.5), the polynomial $p_{S(I)}$ is linear with high probability.

$$p_I[\mathbf{0}] + p_I[\mathbf{x}] + p_I[\mathbf{y}] + p_I[\mathbf{x} + \mathbf{y}] = 0$$

If we repeat the test for sufficiently many times and the test is satisfied in all cases, we ensure that $p_{S(I)}$ is linear with probability near to one. In [7] a generalized version of BLR test was proposed for detecting polynomials of degree 2.

$$p_I[\mathbf{0}] + p_I[\mathbf{x}] + p_I[\mathbf{y}] + p_I[\mathbf{z}] + p_I[\mathbf{x}+\mathbf{y}] + p_I[\mathbf{y}+\mathbf{z}] + \\ p_I[\mathbf{x}+\mathbf{z}] + p_I[\mathbf{x}+\mathbf{y}+\mathbf{z}] = 0$$

It could be also generalized to degree D [10].

Cube attack consists of two parts: Off-line or preprocessing phase and on-line phase. In the off-line phase, the attacker collects a set of polynomials in public (plaintext or ciphertext) and secret (key) variables. In the on-line phase, attacker evaluates these polynomials to derive a system of equations to recover some or all of key bits.

3 S-Box Representation

To attack a block cipher with Gröbner basis, first we need to describe the whole encryption operation with a set of polynomials. The block ciphers design usually consists of applying a simple mixing function (round functions) repeatedly to achieve security. The round function usually consists of a non-linear layer which is also called substitution layer and a diffusion layer which consists of linear transformations [18, 19].

The algebraic description of S-boxes have a direct effect on the efficiency of algebraic attack. $n \times m$ S-boxes are vectorial Boolean functions which translate a Boolean vector of dimension n to a Boolean vector of dimension m.

S-boxes are usually implemented using look-up tables in software and logical gates in hardware. Yet, it is possible to derive a set of polynomials that relates input bits of S-box to its output bits, from hardware implementation. Courtois et al. [20] observe that the AES S-box could have low degree representation with overdefined *Multivariate Quadratic* equations. For example, the 4-bit S-box of PRESENT may be represented by 21 linear-independent equations of degree two [21].

In [1], the polynomials that relate each output bit of the S-box to some of the input bits, are called Forward Equations. The following system of polynomials is forward equations for PRESENT cipher S-box.

$$\begin{aligned}
&y_0 + \; x_0 + x_1x_2 + x_2 + x_3 = 0\\
&y_1 + \; x_0x_1x_2 + x_0x_1x_3 + x_0x_2x_3 + x_1x_3 + x_1 +\\
&\qquad x_2x_3 + x_3 = 0\\
&y_2 + \; x_0x_1x_3 + x_0x_1 + x_0x_2x_3 + x_0x_3 + x_1x_3 +\\
&\qquad x_2 + x_3 + 1 = 0\\
&y_3 + \; x_0x_1x_2 + x_0x_1x_3 + x_0x_2x_3 + x_0 + x_1x_2 +\\
&\qquad x_1 + x_3 + 1 = 0
\end{aligned} \tag{1}$$

Conversely, the polynomials that relate each input bit to some of the output bits, are called Backward Equations. The following system of polynomials expresses backward equations for PRESENT cipher S-box.

$$\begin{aligned}
&x_0 + \; y_0 + y_1y_3 + y_2 + 1 = 0\\
&x_1 + \; y_0y_1y_2 + y_0y_1y_3 + y_0y_2y_3 + y_0y_2 + y_0 +\\
&\qquad y_1y_3 + y_1 + y_2y_3 + y_3 = 0\\
&x_2 + \; y_0y_1y_2 + y_0y_1y_3 + y_0y_1 + y_0y_2y_3 + y_0y_2 +\\
&\qquad y_0y_3 + y_1y_2 + y_1y_3 + y_3 + 1 = 0\\
&x_3 + \; y_0y_1y_2 + y_0y_1 + y_0y_2y_3 + y_0 + y_1 + y_2 +\\
&\qquad y_3 = 0
\end{aligned} \tag{2}$$

The aggregation of Forward and Backward equations is called FWBW representation [1]. It is experimentally shown in [1] that FWBW representation would lead to more efficient algebraic cryptanalysis with Gröbner basis computation. Efficient attacks are also reported on 11-Round LBlock, 6-Round MIBS and 6-Round PRESENT, following FWBW representation. In this paper, we follow the Forward-Backward approach for algebraic representation of S-boxes.

4 Extracting All Linear Equations

As mentioned in Sections 2.1 and 2.2, both multiset of plaintexts in integral cryptanalysis and cubes in cube attack may impose some linear equations in intermediate states of cipher. In a cube attack, these linear polynomials can be recovered by BLR test. In integral cryptanalysis, these polynomials are derived from its specific algebra. Utilizing linear algebra, it is possible to recover a set of all linear polynomials containing above-mentioned polynomials.

In [4, 11], Universal Proning is proposed to derive the set of all linear equations that arises from a set of plaintexts/ciphertexts and the structure of the cipher. The technique is similar to derivation of ANF for an S-box from its lookup table definition [22]. In this technique, all variables appearing in the system of equations are assigned to rows of a matrix. For each column, a key is assigned. Then we extract the values of variables from an encryption oracle, where they are evaluated under the corresponding key.

In [1, 11], it is reported that choosing plaintext samples that are already employed in a successful cube attack and/or integral cryptanalysis improves the efficiency of algebraic cryptanalysis. This may be due to the fact that these samples have a special algebraic structure that causes many linear polynomials to appear (explicitly or implicitly) in the system of equations. So, we apply the Universal Proning technique on each set of plaintexts/ciphertexts to extract all linear equations, that simplify the polynomial system describing the cipher.

Universal Polynomials For this issue, we adopt notations of [11]. Informally, a Universal Polynomial is a polynomial that describes a relation in the cipher and evaluates to zero for all choices of encryption keys. Universal Proning is a technique for finding all such polynomials, however, we are interested specifically in linear ones. These polynomials allow us to simplify the system for algebraic cryptanalysis. By $S_{X,Y,k}$ we denote the polynomial system that describes the cipher for a specific set of plaintexts X and the set of corresponding ciphertexts Y under the key k.

Therefore, the following ideals can be defined [11]:

- The ideal of universal polynomials for encryption under all keys, considering all plaintexts x in the set X, is defined as $\mathcal{P}_X = \bigcap_k \langle S_{X,\star,k} \rangle$.
- The ideal of universal polynomials for decryption under all keys, considering all ciphertexts y in the set Y, is defined as $\mathcal{C}_Y = \bigcap_k \langle S_{\star,Y,k} \rangle$.
- $\mathcal{B}_{X,Y} = \langle S_{X,Y,\star} \rangle$: This ideal contains set of all linear polynomials that may relate intermediate state bits of encryption operation for plaintexts x in the set X and intermediate state bits of decryption operation for ciphertexts y in the set Y, when the encryption and decryption are described by equations on different set of variables, considering the same key. For more detail please refer to [11].

We extract all linear polynomials belonging to the above three sets by linear algebra. Algorithm 1 is used to obtain linear polynomials from ideals P_X and C_Y [11]. This algorithm is a modified version of the Universal Proning algorithm in [11].

Algorithm 1 Universal Forward/Backward Proning.

```
Input: $B \subseteq V$ : a subset of variables ;
Input: $Oracle \leftarrow\ Oracle^{Enc}$ or $Oracle^{Dec}$ ;
Output: $F$: collection of linear polynomials ;
  $K \leftarrow$ random subset of key space such that $|K| = |B| + constant.value$ ;
  $M \leftarrow$ matrix of dimension $|B| \times |K|$ ;
  for all $k \in K$ do
    for all $b \in B$ do
      $M_{i_b,j_k} \leftarrow Oracle_k(b)$ ;
    end for
  end for
  $ker \leftarrow$ find left kernel of matrix $M$ ;
  $F \leftarrow ker \times B$ ;
  return $F$
```

In Algorithm 1, the subset B is selected from the set of all variables V. The elements of the subset K are chosen randomly from the set all of the keys.

Then, a matrix M of dimension $|B| \times |K|$ is created. For each variable $b \in B$, the unique index i_b is assigned, which refers to a unique row of the matrix. For key $k \in K$, the j_k column is assigned. The entry M_{i_b,j_k} is the value of variable b in the encryption/decryption operation under the key k. Then, the basis for left nullspace of A is computed with Gaussian elimination. The set F, which contains the linear polynomials that reside in $\mathcal{P}_X$ or $\mathcal{C}_Y$, is calculated with the multiplication $ker \times B$.

The original algorithm for Universal Proning requires to iterate over all possible keys, which is not practical. Hence, as mentioned in[11], a small subset of key space is used and it is expected that with a high probability the recovered polynomials to be Universal. In our experiments, the number of random samples is just slightly more than the size of B, which is $|K| = |B| + constant.value$. It should be noted that in [11], $|K| = 50 \times |B|$, which may lead to unnecessary computations. Insufficient number of keys may lead the Algorithm to detect a non-universal polynomial as a universal one [11]. We selected the number $constant.value = 256$ experimentally. With this number, we did not encounter any inconsistency in the system of equations. It should be noted that it might require further research to find a formula or a rigorous bound for $constant.value$.

To recover the linear polynomials from ideal $\mathcal{B}_{X,Y}$, we use Algorithm 2 [11]. In this algorithm, for each variable two rows are assigned, one for the value of variables in the encryption and another row for the decryption. So, the indexes of variables have following relation: $i'_b = i_b + |B|$. In other words, the matrix M is comprised of two $|B| \times |K|$ matrices, where the rows of one of them is indexed by i_b and the rows of other one indexed by i'_b.

Algorithm 2 Universal Proning [11].

```
Input: $B \subseteq V$ : set of allowed variables ;
Output: $F$: collection of linear polynomials ;
  $K \leftarrow$ random subset of key space such that $|K| = |B| + constant.value$ ;
  $M \leftarrow$ matrix of dimension $2|B| \times |K|$ ;
  for all $k \in K$ do
    for all $b \in B$ do
      $M_{i_b,j_k} \leftarrow\ Oracle_k^{Enc}(b)$ ;
      $M_{i'_b,j_k} \leftarrow\ Oracle_k^{Dec}(b)$ ;
    end for
  end for
  $ker \leftarrow$ left kernel of matrix $M$ ;
  $F \leftarrow ker \times (B||B)$ ;
  return $F$
```

We break the Proning into three steps:

Step 1. Universal Forward Proning: In this step, Algorithm 1 is run with encryption oracle and recovers linear polynomials resides in $\mathcal{P}_X$.

Step 2. Universal Backward Proning: In this step, Algorithm 1 is run with decryption oracle and recovers linear polynomials resides in $\mathcal{C}_Y$.

Step 3. Universal Proning: In this step, Algorithm 2 is run and recovers linear polynomials resides in $\mathcal{B}_{X,Y}$.

We join the recovered linear polynomials to form the set of universal polynomials.

5 Algebraic Cryptanalysis of Light-Weight Ciphers

To present effectiveness of our method, we selected four light-weight ciphers LBlock, MIBS, PRESENT and SKINNY for algebraic cryptanalysis. The two first ciphers follow Feistel structure but the two latter are designed based on SPN.

Table 1 present some results on algebraic attack reported in the literature on these ciphers [1, 4, 8, 23]. In this table, g denotes the number of guessed key bits that might have been used in the attack.

In this section, we report how to algebraic cryptanalyze the above-mentioned ciphers with our method and compare the efficiency of our proposed method. Experiments are conducted on a desktop computer

Table 1. Algebraic Attacks on LBlock, MIBS and PRESENT, w.r.t Rounds.

N_r	g	$RunTime$	$Data$	$note$	$work$
LBlock					
8	0/80	Not Reported	8 CP	ElimLin	[4]
9	0/80	$O(2^{47})$	1184 CP	Cube Attack Recover 33 bit	[23]
10	0/80	Not Reported	16 CP	ElimLin	[4]
11	0/80	10106 s	128 CP	PolyBoRi-FWBW	[1]
PRESENT					
6	0/80	2009.03 s	32 CP	PolyBoRi-FWBW	[1]
MIBS					
6	0/80	68.46 s	12 CP	PolyBoRi-FWBW	[1]

Table 2. LBock S-Boxes.

s0	14, 9, 15, 0, 13, 4, 10, 11, 1, 2, 8, 3, 7, 6, 12, 5
s1	4, 11, 14, 9, 15, 13, 0, 10, 7, 12, 5, 6, 2, 8, 1, 3
s2	1, 14, 7, 12, 15, 13, 0, 6, 11, 5, 9, 3, 2, 4, 8, 10
s3	7, 6, 8, 11, 0, 15, 3, 14, 9, 10, 12, 13, 5, 2, 4, 1
s4	14, 5, 15, 0, 7, 2, 12, 13, 1, 8, 4, 9, 11, 10, 6, 3
s5	2, 13, 11, 12, 15, 14, 0, 9, 7, 10, 6, 3, 1, 8, 4, 5
s6	11, 9, 4, 14, 0, 15, 10, 13, 6, 12, 5, 7, 3, 8, 1, 2
s7	13, 10, 15, 0, 14, 4, 9, 11, 2, 1, 8, 3, 7, 5, 12, 6
s8	8, 7, 14, 5, 15, 13, 0, 6, 11, 12, 9, 10, 2, 4, 1, 3
s9	11, 5, 15, 0, 7, 2, 9, 13, 4, 8, 1, 12, 14, 10, 3, 6

with 32 GB of RAM, clocked by a Core i7 4770 processor, and running a single core.

We use PolyBoRi library for comupting Gröbner basis [24]. Instead of using its recommended Python interface, we call it from our C++ environment. As the Python implementation for computing the Gröbner basis is more efficient than its implementation in C++, we also re-implemented the Python version in C++. We use M4RI C++ package [25] for operation on Boolean matrices. Our laboratory implementation of the tool can handle the set of plaintexts with up to 512 texts. We also slightly modified the Gröbner basis computation algorithm for finding keys, such that it is returned as soon as all key variables have been found.

After describing a cipher with FWBW representation of S-boxes, we simplify the system of equations by recovering linear polynomials with employing Proning technique. Then, we solve the final system with Gröbner basis computation. So, the following steps are taken after description of the cipher: First, all linear polynomials are found with the Universal Proning technique. Then, many of variables are eliminated from the system of equations, with recovered linear polynomials for the cipher. At the end, PolyBoRi is used to solve the resulting system and find the key.

For efficiency, the Universal Proning step is applied in several stages. The first type of Proning, *i.e.*, Universal Forward Proning, is applied before the on-line phase. Since the attack is a kind of chosen plain-text, some polynomials might also be derived without access to encryption oracle. The Universal Backward Proning and Universal Proning are applied after finding the corresponding ciphertexts. After each stage of Proning, we can eliminate some of the variables from the system. Therefore, the final system of equations has fewer variables.

In our experiments, in chosen-plaintext scenario attacks, the plaintexts are selected based on the integral characteristic for ciphers, except MIBS cipher. For MIBS, we just selected highly correlated message with a cube structure. For other ciphers, integral distinguishers are found by the method proposed in [26].

5.1 Attacking LBlock

Description of LBlock. LBlock [12] is a light-weight 64-bit block cipher with key sizes of 64/80 bits. The cipher consists of 32 rounds. It is presented in ACNS 2011 and had been under much algebraic cryptanalysis [1, 4, 27, 28]. The cipher uses 10 different S-boxes, where 8 S-boxes in round function and 2 in key schedule algorithm. Its round function consists of an S-box layer and a permutation layer. The right branch is rotated 8 bits to right in each round. The S-box layer

Table 3. Algebraic Attacks on LBlock Using FWBW Description of S-Boxes and Universal Proning.

N_r	*Data*	#vars	#lin	#fw	#bw	#pr	#orph	#eqs	T_U	T_G
Higher-Order Chosen Plaintext Scenario - FWBW representation										
9	4 CP	1040	538	473	60	5	61	2493	0.09	6.43
10	16 CP	4284	3088	2547	536	6	509	10893	0.39	4.42
11	16 CP	4768	3112	2536	564	12	267	11691	0.52	2135.77
12	256 CP	82088	71205	50853	20324	28	9656	206440	123.91	5948.16
Higher-Order Chosen Plaintext Scenario - MQ representation										
12	256 CP (2/2)	82088	71248	50883	20342	22	9650	526208	161.71	54934.55
Known Plaintext Scenario - FWBW representation										
6	64 KP	8312	7211	3644	3555	11	186	24842	2.70	4444.47
6	96 KP	12408	11311	5963	5336	11	1356	38300	4.68	1981.30
6	128 KP	16504	15404	8764	6628	12	2467	51699	12.63	78.28
7	256 KP	41088	37975	21409	16521	25	1553	116337	70.98	8907.58
7	512 KP	82048	78923	45984	32915	24	1258	230730	226.66	978.47
Known Plaintext Scenario - MQ representation										
6	96 KP	12408	11309	5963	5337	12	1364	98342	5.70	110.00
6	128 KP	16504	15406	8769	6625	12	2486	131720	12.13	67.31
7	512 KP (13/15)	82048	78915	45985	32904	25	1250	603614	165.00	6303.87

consists of the application of 8 different 4-bit S-boxes over the 32-bit word of the left branch. Table 2, shows the definition of S-boxes of LBlock.

The polynomial system for LBlock cipher is generated by following:

$$\begin{gathered}
L_{0,j} \oplus X_j|_{[0:31]} \\
L_{1,j} \oplus X_j|_{[32:63]} \\
SbxPol(L_{i,j} \oplus K_i, P^{-1}(L_{i-1,j} \lll 8 \oplus L_{i,j+1})) \\
L_{N_r,j} \oplus Y_j|_{[32:63]} \\
L_{N_r+1,j} \oplus Y_j|_{[0:31]}
\end{gathered} \tag{3}$$

In equation (3), and later equations (4),(5) and (6), that describe the above mentioned ciphers with system of polynomials, we use the following notations. j denotes the index of the plaintext that is being encrypted or index of the ciphertext that is being decrypted, where $1 \leq j \leq N_m$. i denotes the round number where $1 \leq i \leq N_r$. $L_{i,j}$ denotes the intermediate state vector in encryption of j-th plaintext, in the i-th round of the cipher. P denotes bit or nibble oriented permutation. M denotes Matrix multiplication operation in round functions. $SbxPol$ denotes the system of equation that describes the relationship between input vector and output vector of the substitution layer. For LBlock and MIBS, $L_{i,j}$ is a vector of dimension 32. For PRESENT and SKINNY, its dimension is 64.

We managed to attack the cipher in both chosen plaintext and known-plaintext scenarios. In a chosen-plaintext scenario, we could attack 9, 10, 11 and 12 rounds of LBlock cipher, but in a known-plaintext scenario, we were able to break 6 and 7-round versions of LBlock. Table 3 shows the results.

In Table 3, *Data* denotes the number of plaintexts used in the attack. $\#vars$, shows the number of variables in the system of equations before elimination. $\#lin$ denotes the total number of linear polynomials that are recovered by Proning Techniques. $\#fw$, $\#bw$ and $\#pr$ present the number of linear polynomials that recovered from $\mathcal{P}_X$, $\mathcal{C}_Y$ and $\mathcal{B}_{X,Y}$, respectively.

#orph denotes the number of linear polynomials that their leading terms have appeared in other polynomials. Therefore, we need to add them to the system of equations. T_U denotes the average running time of Proning step and elimination of variables. T_G denotes the average running time for solving the final system.

In a chosen-plaintext scenario, the plaintexts are chosen and already known. Hence, we can recover polynomials in $\langle S_{x,\star,\star}\rangle$ before retrieving samples from target instance of the cipher. As this step needs to be accomplished only once, we did not include the time of Universal Forward Proning in T_U. The combination of FWBW representation of S-boxes with Universal Proning enabled us to successfully attack 12 rounds of LBlock with 256 chosen plaintexts, in the average of 6,072.02 seconds. To our best knowledge, this is the first algebraic attack on 12-round LBlock. The average running time for Universal Proning is 123.91 seconds and for solving the system and finding the key is about 5948.11 seconds.

We also managed to break 11 rounds of LBlock with 16 chosen plaintexts and a solving time of 2135.77 seconds on average. This is better than previous results reported in [1] with 128 plaintexts and 10106 seconds on average.

To investigate whether these results are due to Universal Proning or efficiency of S-boxes algebraic description, we also did some experiments with MQ representation, in both chosen plaintext and known-plaintext attack scenarios. In a chosen-plaintext attack scenario, we tried to cryptanalyze 12 rounds of LBlock with Universal Proning and MQ representation of S-boxes. Experiments on only two instances yield an average running time of 54934 seconds for solving the system, which is 10 times worse than the results with FWBW representation.

In a known-plaintext attack scenario, the polynomial system of 6-round LBlock with 96 known plaintexts with FWBW is solved on average 1981.30 seconds, and the same with MQ is solved in just 110 seconds on average. For 128 known plaintexts the two representations, are near to each other in terms of running time of computation of Gröbner basis. However, for 64 plaintexts with MQ representation the tool was not able to solve the polynomial system. For FWBW, the polynomial system is solved in 4444.47 seconds on average.

Considering a known-plaintext attack scenario for 7-round LBlock, the polynomial system with FWBW representation and 256 known plaintexts is solved in 8907.58, averagely. But, MQ did not yield any result with the same number of plaintexts. With 512 plaintexts, the system with FWBW representation is solved in 978.47 seconds on average, but the MQ representation leads to a solving time of 6303.87 seconds on average. It should be noted that with MQ representation, PolyBoRi library failed to solve the system for two instances, in this case.

With the above pieces of evidence, we have evidences that FWBW is a better representation for Gröbner basis based algebraic cryptanalysis. Considering the results reported in [1], it seems that FWBW representation is the most convenient description of S-boxes, among currently proposed representations, for algebraic cryptanalysis.

5.2 Attacking MIBS

Description of MIBS. MIBS [13] is a 64-bit lightweight block cipher based on Fiestel structure. It was presented in CANS 2009. Its round function consists of an S-box layer, a multiplication by 8-by-8 binary matrix and a permutation layer. This cipher consists of 32-rounds. Table 4 defines the MIBS cipher S-box.

Table 4. MIBS S-Box.

S	4,15,3,8,13,10,12,0,11,5,7,14,2,6,1,9

The binary matrix, represented with M, is given as follows:

$$M = \begin{bmatrix} 0&1&1&1&1&1&1&0 \\ 1&0&1&1&0&1&1&1 \\ 1&1&0&1&1&0&1&1 \\ 1&1&1&0&1&1&0&1 \\ 1&1&0&1&1&1&0&0 \\ 1&1&1&0&0&1&1&0 \\ 0&1&1&1&0&0&1&1 \\ 1&0&1&1&1&0&0&1 \end{bmatrix}.$$

It has a complicated diffusion layer. Some algebraic cryptanalysis of reduced-round MIBS are reported in [1, 27, 29, 30].

The polynomial system for MIBS cipher is generated as following:

$$\begin{aligned} & L_{0,,j} \oplus X_j|_{[0:31]} \\ & L_{1,j} \oplus X_j|_{[32:63]} \\ & SbxPol(L_{i,j} \oplus K_i, P^{-1}(M^{-1}(L_{i,j-1} \oplus L_{i,j+1}))) \qquad (4) \\ & L_{N_r,j} \oplus Y_j|_{[32:63]} \\ & L_{N_r+1,j} \oplus Y_j|_{[0:31]} \end{aligned}$$

We managed to attack 5 and 6 rounds of MIBS cipher with 5 and 12 chosen plaintexts, respectively.

Table 5. Algebraic Attacks on MIBS Using FWBW Description of S-Boxes and Universal Proning.

N_r	*Data*	#vars	#lin	#fw	#bw	#pr	#orph	#eqs	T_U	T_G
Higher-Order Chosen Plaintext Scenario										
5	5 CP	592	294	298	24	16	68	1748	0.09	7.61
6	12 CP	1656	831	842	292	8	16	4720	0.30	18.14

Table 6. PRESENT S-Box.

S	12,5,6,11,9,0,10,13,3,14,15,8,4,7,1,2

Table 5 present our result on MIBS cipher.

Using FWBW representation with the Universal Proning technique, we were not able to improve the number of rounds in comparison with [1], but we achieved better running time for computation of Gröbner basis. Intuitively it seems that adding more linear polynomials to the system should make the solving easier, but MIBS cipher was an exception. We found that adding Backward universal polynomials and Universal polynomials increase the running time for MIBS, significantly. Therefore, we did not add these polynomials to the final system.

5.3 Attacking PRESENT

Description of Present. PRESENT [14], presented in CHES 2007, is a light-weight block cipher based on SPN structure. It has a block size of 64-bit and a key size of 64/80 bits. Its round function consists of application of 16 4-bit S-boxes in parallel, then applying a bit bit-oriented permutation. The cipher consists of 31 rounds. Table 6 defines PRESENT S-box.

It is also received much attention for cryptanalysis in the literature [1, 27, 31, 32].

The polynomial system for PRESENT cipher is generated by the following:

$$\begin{aligned} &L_{0,j} \oplus X_j \\ &SbxPol(L_{0,j} \oplus K_1, L_{1,j}) \\ &SbxPol(P(L_{i-1,j}) \oplus K_i, L_{i,j}) \\ &P(L_{i,Nr}) \oplus K_{Nr+1} \oplus Y_j \end{aligned} \tag{5}$$

In above equation we have $i = 2, \ldots, Nr$. Using FWBW representation together with Universal Proning not only enabled us to break 5 and 6 rounds of the cipher more efficient than our previous results, but also allowed to break 7 rounds of the cipher with 256 chosen plaintexts. To our best knowledge, this is the first algebraic attack on 7 rounds of PRESENT. The average running for attacking 6 rounds of the cipher was reduced from around 2000 seconds to 48.7 seconds on average. 7 round version of the cipher is broken with 256 chosen plaintexts and 59316.72 seconds on average. Table 7 shows the results.

5.4 Attacking SKINNY

Description of SKINNY. SKINNY [15] is a family of lightweight tweakable block ciphers following AES design, but with some modification to minimize hardware implementation costs. The major difference between AES and SKINNY is that SKINNY uses a binary matrix for the MixColumn operation. The cipher has two variants: 64-bit and 128-bit versions. The first version operates on a state matrix with sixteen nibbles but the later version works on state matrix with sixteen bytes. The 64-bit version uses 4-bit S-boxes and the 128-bit version uses 8-bit ones. None of those S-boxes posses strong cryptographic properties, in a deal with a light implementations.

Its round function consists of the following four operations, similar to AES:

(1) **SubCells (SC)**: S-box is applied on nibbles (bytes) in parallel, 4-bit S-boxes in 64-bit version and 8-bit S-boxes in 128-bit version. The 4-bit S-box is represented in following Table 8.
(2) **AddConstants (AC)**: round constants derived using a 6-bit LFSR are added into the state.
(3) **AddRoundTweakey (ART)**: For SKINNY round keys depend on both the master key and the tweak. This operations adds such key material to half of the internal state.
(4) **ShiftRows (SR)**: Similar to AES shift row operation.
(5) **MixColumns (MC)**: Each column is multiplied by a binary matrix M given below.

$$M = \begin{bmatrix} 1\ 0\ 1\ 1 \\ 1\ 0\ 0\ 0 \\ 0\ 1\ 1\ 0 \\ 1\ 0\ 1\ 0 \end{bmatrix}.$$

The polynomial system for SKINNY cipher is gen-

Table 7. Algebraic Attacks on PRESENT Using FWBW Description of S-Boxes And Universal Proning.

N_r	*Data*	#vars	#lin	#fw	#bw	#pr	#orph	#eqs	T_U	T_G
Higher-Order Chosen Plaintext Scenario										
5	6 CP	2020	1466	784	681	2	226	4490	0.14	13.19
6	32 CP	12392	9879	5752	4123	4	715	27387	1.63	47.07
7	256 CP (4)	114796	92141	51496	40461	4	5744	241320	253.27	59316.72

Table 8. SKINNY 4-Bit S-Box.

S	12,6,9,0,1,10,2,11,3,8,5,13,4,14,7,15

erated by following:

$$\begin{aligned}
&L_{0,j} \oplus X_j \\
&SbxPol(L_{0,j}, L_{1,j}) \\
&SbxPol(M(P(L_{i-1,j} \oplus C_{i-1} \oplus K_{i-1})), L_{i,j}) \\
&M(P(L_{Nr,j} \oplus C_{Nr} \oplus K_{Nr})) \oplus Y_j
\end{aligned} \tag{6}$$

In above equation we have $i = 2, \ldots, Nr$. We applied our method to SKINNY with 64-bit block size and key sizes of 64 and 128 bits. Results are presented in Table 9.

We managed to attack 8 and 9 rounds of SKINNY with 16 and 256 chosen plaintexts, respectively. The running time for solving the system of equation for SKINNY-64-64 and SKINNY-64-128, is 1757.58 and 3437.16 seconds on average, respectively. Although the number of different type of equations that found by Universal Proning for SKINNY-64-64 and SKINNY-64-128 are close to each other, due to that both cipher have structure and these equations arise from the properties of the structure, the average running time for SKINNY-64-128 is as twice as the average running time for SKINNY-64-64.

We also report attacks in known-plaintext scenario on 5 and 6 rounds of SKINNY. In a known plaintext scenario, we noticed that encryption and decryption of SKINNY exhibit some unbalanced properties. Considering 5 rounds of SKINNY-64-128, number linear polynomials that recovered from decryption (backward) direction is more than the number recovered polynomials from encryption (forward) direction. So we tried the attack in a known ciphertext scenario which means recovering linear polynomials in backward direction first. It shows a significant improvement in T_G. Considering the observation we have been able to attack 6 rounds if SKINNY-64-128 with 256 known plaintexts.

5.5 A Comparison

In this section, we provide a comparison for our algebraic attacks of the lightweight block ciphers LBlock, MIBS, PRESENT and SKINNY. Table 10 summarizes other previous attacks on the above mentioned ciphers. As the nature of attacks under considerations are different, a comparison of them with our algebraic cryptanalysis is not straightforward. For example, differential attacks are probabilistic and their efficiency can be relatively easily extrapolated, while algebraic attacks are deterministic and their success depends on solving a system of (nonlinear) relations. So far, a generic algorithm that would solve (efficiently) any system of nonlinear relations is not known, particularly the system of nonlinear relations arising from block ciphers for a large number of rounds. Due to the behaviour of solving algorithms, it is difficult to derive efficiency measurement, and tight bounds on data, time and memory requirements of the algorithm are not known. It worth noting that algebraic cryptanalysis is still evolving and many of its aspects are yet to be discovered, and many results in this area are reported based only on experiments. So, we compare the attacks only based on the number of required plaintexts to be encrypted and the time needed for cryptanalysis in the experiments.

Table 10 details previous differential, integral and cube attacks for the ciphers.

Let's consider the differential cryptanalysis of LBlock. The best differential characteristics for 11-round LBlock implies at least 22 active S-boxes [12]. If this characteristic is used in a 12-round key recovery attack, the data complexity of the attack would be of order $O(2^{44})$. Our algebraic attack requires a much smaller number of plaintexts, *i.e.*, only 256 plaintexts. The work in [28] reports an integral attack on 22-round LBlock with data and time complexity of 2^{61} and $O(2^{70})$, respectively, where the attack is based on a 15-round integral distinguisher.

Z'aba et al. [33] present a bit-pattern-based integral attack on 6 rounds of PRESENT-80. The attack takes advantage of a 4.5 round integral distinguisher. The data and time complexity of the attack is $2^{22.4}$ and

Table 9. Algebraic Attacks on SKINNY Using FWBW Description of S-Boxes and Universal Proning.

N_r	*Data*	#vars	#lin	#fw	#bw	#pr	#orph	#eqs	T_U	T_G
SKINNY-64-64 Higher-Order Chosen Plaintext Scenario										
8	16 CP	8256	6819	4377	2431	9	700	18108	0.63	4.01
9	256 CP	147520	134637	85612	48973	46	9188	320484	82.44	1757.58
SKINNY-64-64 Known Plaintext Scenario										
5	32 KP	10304	9807	4725	5034	48	681	23209	1.64	14.11
5	32 KC	10304	9806	4016	5742	48	757	15152	1.93	11.90
SKINNY-64-128 Higher-Order Chosen Plaintext Scenario										
8	16 CP	8320	6814	4377	2437	0	752	18160	0.80	8.08
9	256 CP	147584	134586	85617	48969	0	9137	254946	101.04	3437.16
SKINNY-64-128 Known Plaintext Scenario										
5	40 KP	12928	12315	6087	6228	0	865	29025	2.62	90.80
5	40 KC	12928	12314	5016	7298	0	1129	29289	3.4	27.68
6	256 KC	98432	94519	52171	42348	0	6442	219434	136.13	1501.70

$2^{41.7}$, respectively. Our algebraic cryptanalysis attack requires only 32 chosen plaintexts only and a running time of about 48.7 seconds on average. We have been also able to attack 7-round PRESENT with just 256 chosen plaintexts. In [34], an integral attack for 7-round PRESENT is presented. The attack has data and time complexity of $2^{8.3}$ and 2^{60}, respectively. The best attack for PRESENT-80 reported in [36] which covers 28 rounds of the cipher. This attack is based on linear cryptanalysis and its data and time complexity is 2^{64} and $2^{77.4}$, respectively.

For MIBS cipher, the best 4-round differential characteristics has the probability of $O(2^{-15})$ [13]. If this characteristic is used in a hypothetical key recovery attack on 6-round MIBS, it would require a data complexity of at least $O(2^{15})$. In this paper, however, the 6-round MIBS cipher is broken with only 12 chosen plaintexts in an algebraic cryptanalysis attack. In [29], a differential cryptanalysis attack is proposed on 13 rounds of MIBS based on a 12-round differential characteristic, with data and time complexity of 2^{61} and 2^{56}, respectively .

Let's consider the differential cryptanalysis of SKINNY. The best differential characteristics for 7-round SKINNY-64 implies 28 active S-boxes [15]. This would lead to an attack on 8-round SKINNY-64 with a data complexity of 2^{56}. While our attack on 8-round SKINNY-64 needs only 16 chosen plaintexts. Attacking 9-round SKINNY-64 requires only 256 chosen plaintexts. In [26], an integral distinguisher for 10 rounds of SKINNY is also reported. A related-tweaky impossible differential attack on 23-round SKINNY-64-128 is reported by Sadeghi et. al in [37]. The data and time complexity of the attack is 2^{127} and $2^{62.95}$, respectively.

6 Conclusions and Discussion

In this paper, we proposed a new method to launch a more efficient Algebraic Cryptanalysis. We employed an effective FWBW representation of S-boxes for algebraic description of ciphers. Then, we showed that combining this representation with carefully selected plaintexts and Universal Proning in the solving stage improves the running time for solving the system to find the key.

In this work, we have also done experiments on a limited number of light-weight block ciphers with 4-bit S-boxes. These ciphers are designed based on different strategies and we were able to report a successful first algebraic attack on 12-round LBlock, 7-round PRESENT and 9-round SKINNY. Although, we do not yet have a theoretic result for the effectiveness of our method, *i.e.*, FWBW with Universal Proning, for algebraic cryptanalysis in general, we can expect that the reported results should be extended to other ciphers as well. Consequently, our proposed method could be used as a criterion for the evaluation of resistance of light-weight ciphers against algebraic

Table 10. Some Integral and Differential Cryptanalysis of LBlock, MIBS, PRESENT and SKINNY.

N_r	$RunTime$	$Data$	$note$	$work$
LBlock-80				
11	-	$O(2^{44})$ CP	Differential Characteristics	[12]
22	$O(2^{70})$	2^{61} CP	Integral Cryptanalysis	[28]
PRESENT-80				
6	$2^{41.7}$	$2^{22.4}$ CP	Integral Cryptanalysis	[33]
7	2^{60}	$2^{8.3}$ CP	Integral Cryptanlysis	[34]
9	2^{60}	$2^{20.3}$ CP	Integral Cryptanlysis	[34]
9	-	2^{60} CP	Integral Distinguisher	[26]
16	2^{64}	2^{64} CP	Differential Cryptanalysis	[35]
28	$2^{77.4}$	2^{64} KP	Linear Cryptanalysis	[36]
MIBS-80				
4	-	$O(2^{15})$ CP	Differential Characteristics	[13]
13	2^{56}	2^{61} CP	Differential Cryptanalysis	[29]
SKINNY-64				
7	-	$O(2^{56})$ CP	Differential Characteristics	[15]
10	-	2^{48} CP	Integral Distinguisher	[26]
23	2^{127}	$2^{62.95}$ CP	Related-Tweakey Impossible Differential Cryptanalysis	[37]

cryptanalysis, regarding the NIST competition of lightweight cryptography.

In general, the Universal Proning technique alone helps to find many linear equations. Since these polynomials are universal and satisfied for all keys, they do not contribute to finding the key, but help to simplify the system of equations by removing much of variables from the system [11]. For example, considering the attack on LBlock with 256 correlated plaintexts, we could remove 71205 variables of total 98472 from the system, with Universal Proning. Hence, the resulting system of equations is much simplified. This may question the role of effective representation, for the next step. To answer the question, we also applied our method with MQ representation of S-boxes. As can be seen in Table 3, the average running time of T_G significantly increased in comparison with FWBW representation. Therefore, we can conclude that FWBW is an effective method for the description of S-boxes.

We also found some irregular properties in MIBS and SKINNY ciphers. For MIBS cipher, the running time of T_G would exceptionally increase, when the linear polynomials that found by Universal Backward Proning and Universal Proning are taken into account. This contrasts the common intuition that removing variables or adding linear equations should result in more efficient solving time. For SKINNY, we found that the cipher exhibits more linear equations in Backward (decryption) direction than Forward (encryption) direction, in a known-plaintext scenario. As it is shown in Table 9, this leads to more efficient attacks in both known/chosen-ciphertext scenario. It worth investigating the effect of this unbalanced algebraic property on other types of attacks.

Our tool could not handle a large system of equations that arises from a large number of samples due limitation in employed software and hardware. Therefore, it is worth improving the implementation in order to better investigate the limitations and capabilities of algebraic cryptanalysis with Gröbner basis methods.

References

[1] H. Arabnezhad-Khanoki, B. Sadeghiyan, and J. Pieprzyk. S-boxes representation and efficiency of algebraic attack. *IET Information Se-*

curity, 13(5):448–458, 2019. ISSN 1751-8709. doi:10.1049/iet-ifs.2018.5201.

[2] J. Faugere and L. Perret. Improving the recognition of faces occluded by facial accessories. In *International Conference on Information Security and Cryptology*, pages 266–277. Springer, 2009. ISBN 978-3-642-16341-8. doi:10.1007/978-3-642-16342-5_19.

[3] G. V. Bard, N. T. Courtois, J. Nakahara, P. Sepehrdad, and B. Zhang. Improving the recognition of faces occluded by facial accessories. In *International Conference on Cryptology in India*, pages 176–196. Springer, 2010. ISBN 978-3-642-17400-1. doi:10.1007/978-3-642-17401-8_14.

[4] P. Sušilα, P. Sepehrdad, and S. Vaudenay. On Selection of Samples in Algebraic Attacks and a New Technique to Find Hidden Low Degree Equations. In *Information Security and Privacy*, pages 50–65. Springer, 2014. ISBN 978-3-319-08343-8. doi:10.1007/978-3-319-08344-5.

[5] E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 4(1):3–72, 1991. ISSN 0898-929X. doi:10.1007/BF00630563.

[6] L. Knudsen and D. Wagner. Integral cryptanalysis. In *International Workshop on Fast Software Encryption*, pages 112–127. Springer, 2002. ISBN 978-3-540-44009-3. doi:10.1007/3-540-45661-9_9.

[7] I. Dinur and A. Shamir. Annual International Conference on the Theory and Applications of Cryptographic Techniques. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, pages 278–299. Springer, 1999. ISBN 978-3-642-01000-2. doi:10.1007/978-3-642-01001-9_16.

[8] Y. Todo. Structural evaluation by generalized integral property. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 287–314. Springer, 2015. ISBN 978-3-662-46799-2. doi:10.1007/978-3-662-46800-5_12.

[9] M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of computer and system sciences*, 47(3):549–595, 1993. doi:10.1016/0022-0000(93)90044-W.

[10] S. Abdul-Latip, M. R. Reyhanitabar, W. Susilo, and J. Seberry. Extended cubes: enhancing the cube attack by extracting low-degree non-linear equations. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, page 296–305. Springer, Berlin, Heidelberg, 2011. doi:10.1145/1966913.1966952.

[11] P. SUŠIL. Algebraic Cryptanalysis of Deterministic Symmetric Encryption. URL `http://infoscience.epfl.ch/record/210605/files/EPFL_TH6651.pdf`.

[12] J. Faugere and L. Perret. LBlock: a lightweight block cipher. In *International Conference on Applied Cryptography and Network Security*, pages 327–344. Springer, 2011. ISBN 978-3-642-21553-7. doi:10.1007/978-3-642-21554-4_19.

[13] M. Izadi, B. Sadeghiyan, S. S. Sadeghian, and H. A. Khanooki. Structural evaluation by generalized integral property. In *International Conference on Cryptology and Network Security*, pages 334–348. Springer, 2009. ISBN 978-3-642-10432-9. doi:10.1007/978-3-642-10433-6_22.

[14] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. Robshaw, Y. Seurin, and C. Vikkelsoe. PRESENT: An ultra-lightweight block cipher. In *International workshop on cryptographic hardware and embedded systems*, pages 450–466. Springer, 2007. ISBN 978-3-540-74734-5. doi:10.1007/978-3-540-74735-2_31.

[15] 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. In *Annual International Cryptology Conference*, pages 123–153. Springer, 2016. ISBN 978-3-662-53007-8. doi:10.1007/978-3-662-53008-5_5.

[16] X. Lai. Higher order derivatives and differential cryptanalysis. In *Communications and cryptography*, pages 227–233. Springer, 1994.

[17] J. Daemen, L. Knudsen, and V. Rijmen. The block cipher Square. In *International Workshop on Fast Software Encryption*, pages 149–165. Springer, 2009. ISBN 978-3-540-63247-4. doi:10.1007/BFb0052343.

[18] J. Faugere and L. Perret. The cipher SHARK. In *International Workshop on Fast Software Encryption*, pages 99–111. Springer, 1996. ISBN 978-3-540-60865-3. doi:10.1007/3-540-60865-6_47.

[19] J. L. Massey. SAFER K-64: A byte-oriented block-ciphering algorithm. In *International Workshop on Fast Software Encryption*, pages 1–17. Springer, 2009. ISBN 978-3-540-58108-6. doi:10.1007/3-540-58108-1_1.

[20] N. T. Courtois and J. Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 267–287. Springer, 2002. ISBN 978-3-540-00171-3. doi:10.1007/3-540-36178-2_17.

[21] J. N. Jr, P. Sepehrdad, B. Zhang, and M. Wang. Linear (hull) and algebraic cryptanalysis of the block cipher PRESENT. In *International Conference on Cryptology and Network Security*, pages 58–75. Springer, 2009. ISBN 978-3-642-10432-9. doi:10.1007/978-3-642-10433-6_5.

[22] A. Biryukov and C. D. Cannière. Block Ciphers and Systems of Quadratic Equations. In *International Workshop on Fast Software Encryption*, pages 274–289. Springer, 2003. ISBN 978-3-540-20449-7. doi:10.1007/978-3-540-39887-5_21.

[23] S. Islam, M. Afzal, and A. Rashdi. On the security of LBlock against the cube attack and side channel cube attack. In *International Conference on Availability, Reliability, and Security*, pages 105–121. Springer, 2013. ISBN 978-3-642-40587-7. doi:10.1007/978-3-642-40588-4_8.

[24] M. Brickenstein and A. Dreyer. Polybori: A framework for gröbner-basis computations with boolean polynomials. *Journal of Symbolic Computation*, 44(9):1326–1345, 2009. doi:10.1016/j.jsc.2008.02.017.

[25] M. Albrecht and G. Bard. *The M4RI Library – Version 20140914*. The M4RI Team, 2014. URL `http://m4ri.sagemath.org`.

[26] Z. Eskandari, A. B. Kidmose, S. Kölbl, and T. Tiessen. Finding Integral Distinguishers with Ease. In *International Conference on Information Security and Cryptology*, pages 115–138. Springer, 2018. ISBN 978-3-030-10969-1. doi:10.1007/978-3-030-10970-7_6.

[27] N. T. Courtois, P. Sepehrdad, P. Sušil, and S. Vaudenay. ElimLin algorithm revisited. In *International Workshop on Fast Software Encryption*, pages 306–325. Springer, 2012. ISBN 978-3-642-34046-8. doi:10.1007/978-3-642-34047-5_18.

[28] Y. Sasaki and L. Wang. Comprehensive study of integral analysis on 22-round LBlock. In *International Conference on Information Security and Cryptology*, pages 156–169. Springer, 2012. ISBN 978-3-642-37681-8. doi:10.1007/978-3-642-37682-5_12.

[29] A. Bay, J. Nakahara, and S. Vaudenay. Cryptanalysis of reduced-round MIBS block cipher. In *International Conference on Cryptology and Network Security*, pages 1–19. Springer, 2010. ISBN 978-3-642-17618-0. doi:10.1007/978-3-642-17619-7_1.

[30] S. Wu and M. Wang. Automatic search of truncated impossible differentials for word-oriented block ciphers. In *International Conference on Cryptology in India*, pages 283–302. Springer, 2012. ISBN 978-3-642-34930-0. doi:10.1007/978-3-642-34931-7_17.

[31] M. Albrecht and C. Cid. Algebraic techniques in differential cryptanalysis. In *International Workshop on Fast Software Encryption*, pages 193–208. Springer, 2009. ISBN 978-3-642-03316-2. doi:10.1007/978-3-642-03317-9_12.

[32] B. Collard and F. X. Standaert. A statistical saturation attack against the block cipher PRESENT. In *Cryptographers' Track at the RSA Conference*, pages 195–210. Springer, 2009. ISBN 978-3-642-00861-0. doi:10.1007/978-3-642-00862-7_13.

[33] M. R. Z'aba, H. Raddum, M. Henricksen, and E. Dawson. Bit-pattern based integral attack. In *International Workshop on Fast Software Encryption*, pages 363–381. Springer, 2008. ISBN 978-3-540-71038-7. doi:10.1007/978-3-540-71039-4_23.

[34] S. Wu and M. Wang. Integral attacks on reduced-round PRESENT. In *International Conference on Information and Communications Security*, pages 331–345. Springer, 2013. ISBN 978-3-319-02725-8. doi:10.1007/978-3-319-02726-5_24.

[35] M. Wang. Structural evaluation by generalized integral property. In *International Conference on Cryptology in Africa*, pages 40–49. Springer, 2008. ISBN 978-3-540-68159-5. doi:10.1007/978-3-540-68164-9_4.

[36] A. Flórez-Gutiérrez and M. Naya-Plasencia. Improving Key-Recovery in Linear Attacks: Application to 28-Round PRESENT. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 221–249. Springer, 2020. ISBN 978-3-030-45720-4. doi:10.1007/978-3-030-45721-1_9.

[37] S. Sadeghi, T. Mohammadi, and N. Bagheri. Cryptanalysis of reduced round SKINNY block cipher. *IACR Transactions on Symmetric Cryptology*, 2018(3):124–162, 2018. doi:10.13154/tosc.v2018.i3.124-162.

Hossein Arabnezhad received his B.S. degree in Computer Engineering from Ferdowsi University, Mashhad, Iran, in 2007. He received his M.S. degree in Information Technology Engineering from Amirkabir University Technology, Tehran, Iran, in 2010. Currently, he is a Ph.D. candidate in Computer Engineering at Amirkabir University of Technology, Tehran, Iran. His research interest is the cryptanalysis of block ciphers.

Babak Sadeghiyan received his B.Sc. in 1985 in Electrical (Electronics) Engineering from Isfahan University of Technology, and his M.Sc. in 1989 in Electronics Engineering from Amirkabir University of Technology, Tehran, Iran. He received his Ph.D. in 1993, in Computer Science from University College, University of New South Wales, Australia, on the design of secure hash functions. Then he joined the Department of Computer Engineering, Amirkabir University of Technology in 1993, where he is still continuing his academic activities, and is currently an associate professor of the department. His research area of interest includes all aspects of Cryptology and Information Security, more specifically, he has contributed on the design and analysis of cryptographic algorithms and cryptographic protocols.