THE PUBLISHING HOUSE
OF THE ROMANIAN ACADEMY

PROCEEDINGS OF THE ROMANIAN ACADEMY, Series A,
Volume 18, Special Issue 2017, pp. 333-350

NEW RELATED-KEY ATTACKS AND PROPERTIES OF SKINNY-64-128 CIPHER

Raluca POSTEUCA*, Gabriel NEGARA*

* University of Bucharest, Romania
E-mail: raluca.e.posteuca@gmail.com

Abstract. The SKINNY family of lightweight block ciphers was introduced at CRYPTO'16 by Beierle et al. [1]. In order to encourage the analysis of the SKINNY ciphers, the authors introduced the SKINNY Cryptanalysis Competition [2]. In this paper we introduce five rectangle attacks on 16-round, 18-round and 20-round reduced SKINNY-64-128, three of them having practical complexity. We also found new related-key impossible distinguishers on 13 rounds of SKINNY-64-128. In the last part of the paper, we present the results obtained by a fixed-point analysis on round-reduced SKINNY-64-128.

***Key words*:** lightweight cryptography, SKINNY family of block ciphers, SKINNY-64-128, related-key attacks, rectangle attacks, impossible differential attacks, fixed-point analysis.

1. INTRODUCTION

The domain of lightweight cryptography has become a very active research area due to the necessity of designing and implementing cryptographic primitives for secure constrained devices. Due to the fact that the existing cryptographic primitives were not able to fulfill the requirements of devices with limited resources, during the last years, new primitives were created, with better properties regarding the implementation costs, the performance and efficiency.

In the area of lightweight block ciphers, two families of algorithms, SIMON and SPECK [3], were proposed by NSA in 2013, both being more efficient than other ciphers. With the openly stated purpose to compete with SIMON and SPECK families of ciphers, both in terms of software and hardware performance, Beierle et al. introduced, at CRYPTO'16, the SKINNY family of lightweight tweakable block ciphers. SKINNY has a Substitution-Permutation-Network structure and supports two block sizes (64 and 128 bits), each of them supporting three tweak/key sizes (the block size, twice the block size or three times larger than the block size). The notation introduced by the authors of the cipher, and used further in this paper is SKINNY-*n*-*t*, where *n* represents the block length and *t* represents the tweak/key length.

In order to enhance the cryptanalysis efforts and to encourage the design of practical attacks, the authors organized the SKINNY cryptanalysis competition [2]. Five cryptanalysis categories were proposed, differentiated by the number of rounds required for the attacks; the versions of SKINNY targeted in the competition are the two versions that use 128-bit keys (SKINNY-64-128 and SKINNY-128-128). The first phase of the competition has been held from the autumn of 2016, until March 1st, 2017; two research teams were designated as winners [4], [5]. The second edition of the competition is in progress, with deadline on February 1st, 2018. The rules of the second edition differ from the ones in the first edition only by an increased numbers of rounds required for the attacks against SKINNY-64-128.

Our contribution. We have focused our research on studying and implementing various types of attacks against round-reduced SKINNY-64-128. We introduce new related-key rectangle attacks on 16, 18 and 20 round SKINNY-64-128, the ones on 16 and 18 rounds having practical time and data complexity. Also, the correctness of our practical attacks was confirmed by implementations, while the 20-round attacks were validated by various experimental results. These five attacks, detailed below, were submitted to the first edition of the SKINNY cryptanalysis competition. The distinguishers used in our attacks are derived from a 16-round distinguisher with probability 2^{-17}.

We also studied existing related-key impossible attacks against SKINNY-64-128 [4] and found new distinguishers on 13 rounds of SKINNY-64-128.

Our research also includes a study of some fixed-point properties on round-reduced SKINNY-64-128.

Structure of the paper. The paper is organized as follows: the SKINNY family of lightweight tweakable block ciphers is briefly described in Section 2; in Section 3 we synthetize the related work regarding existing attacks on round-reduced SKINNY-64-128; Section 4 presents the details of our related-key rectangle attacks and the results we obtained; the related-key impossible distinguishers are described in Section 5; in Section 6 we present a comparison between our attacks and the ones previously introduced; details regarding our work on the fixed-point analysis are presented in Section 7; the last part, Section 8, is dedicated to conclusions and future work.

2. RELATED WORK

With the rapid development of the design of new lightweight block ciphers, the cryptographic community concentrated their efforts in testing the security of the newly published ciphers. This section specifies papers published in the field of SKINNY family cryptanalysis.

The best known attacks on round-reduced SKINNY-64-128 are designed in the related-key scenario and won the first edition of the SKINNY Cryptanalysis Competition on 22-round attacks [4]. In this paper, Liu et al. designed impossible-differential attacks and rectangle attacks for 19 rounds of SKINNY-n-n, 23 rounds of SKINNY-n-2n and 27 rounds of SKINNY-n-3n.

The second paper awarded in the first edition of the competition, for the sections dedicated to 18 and 20-round attacks against SKINNY-64-128, was [5]. The authors introduced two related-key impossible-differential attacks on 21 and 22 round SKINNY-64-128.

In [6], Tolba et al. introduced impossible-differential attacks for 18 rounds of SKINNY-n-n, 20 rounds of SKINNY-n-2n and 22 rounds of SKINNY-n-3n. Moreover, Sadeghi et al. [7] focused their research on impossible differential and zero-correlation linear characteristics of different SKINNY versions. They found characteristics for SKINNY-n-n up to 12 rounds.

3. DESCRIPTION OF SKINNY

SKINNY [1] is a family of lightweight block ciphers introduced at CRYPTO 2016 Conference. Following the TWEAKEY framework [8], the ciphers take as input a tweakey, instead of a key or a tweak/key pair. When using a cipher from the SKINNY family, one has the freedom to choose which part of the tweakey will represent the key material (the secret key) and which part will represent the tweak (considered public). The length of the tweakey (denoted by *t*) is dependent on the plaintext's length (denoted by *n*), a cipher from the SKINNY family having three main tweakey sizes: $t = n$, $t = 2n$ and $t = 3n$. The size of a block used as input of the SKINNY ciphers may have 64 or 128 bit length.

The tweakey is handled as a collection of *t/n* 4×4 matrices of nibbles (if $n = 64$) or bytes (if $n = 128$). If $t = n$, the matrix is denoted by TK1. If $t = 2n$, these matrices are denoted by TK1 and TK2. In the last case, when $t = 3n$, the notations are TK1, TK2 and TK3. The plaintext is also viewed as a 4×4 matrix.

The number of rounds of SKINNY-*n*-*t* depends on the block and tweakey sizes, as depicted in the table below:

Table 1

The number of rounds of SKINNY-*n*-*t*

n \ t	n	$2n$	$3n$
64	32	36	40
128	40	48	56

3.1. The round function

The round function of all SKINNY versions contains five different operations: SubCells, AddConstants, AddRoundTweakey, ShiftRows and MixColumn.

Further on, we will describe the 64-128 version of the SKINNY cipher, our research being conducted mainly on this version.

This version operates on 64-bit blocks ($n = 64$, the elements of the internal state are nibbles) and uses 128-bit tweakey ($t = 128$, represented by TK1 and TK2, two 4×4 matrices of nibbles – each of them called "tweakey matrices" for the rest of the paper).

Note: for simplicity, a matrix position $[x, y]$ is denoted by $4x + y$.

The **SubCells** (SC) operation represents the S-box application on each nibble of the internal state:

Table 2

The S-box used in SKINNY-64-128

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
S(x)	c	6	9	0	1	a	2	b	3	8	5	d	4	e	7	f

In the **AddConstants** (AC) operation, the first three values from the first column of the internal state are XORed with the last 4 bits of the round constant C_i (position $[0,0]$), the first 2 bits of C_i (position $[1,0]$) and with the value 2 (position $[2,0]$); C_i represents the constant used in the i[th] round.

The round constants are obtained using a 6-bit LFSR, with the initial state 000001 and the following update function [1]:

$$(\mathrm{rc}_5 || \mathrm{rc}_4 || \mathrm{rc}_3 || \mathrm{rc}_2 || \mathrm{rc}_1 || \mathrm{rc}_0) \rightarrow (\mathrm{rc}_4 || \mathrm{rc}_3 || \mathrm{rc}_2 || \mathrm{rc}_1 || \mathrm{rc}_0 || \mathrm{rc}_5 \oplus \mathrm{rc}_4 \oplus 1)$$

The **AddRoundTweakey** (ART) operation consists of two phases. The first one is performed by XORing the first two rows of the internal state with the corresponding nibbles of the two tweakey matrices. In the second phase the tweakey is prepared for the next round – this phase actually represents the key schedule of SKINNY. The values of the two tweakeys are mixed based on the permutation from Table 3. At this point, the nibbles of the first two rows of the second tweakey matrix are modified based on the following LFSR update function [1]:

$$(\mathrm{x}_3 || \mathrm{x}_2 || \mathrm{x}_1 || \mathrm{x}_0) \rightarrow (\mathrm{x}_2 || \mathrm{x}_1 || \mathrm{x}_0 || \mathrm{x}_3 \oplus \mathrm{x}_2)$$

Table 3

The tweakey permutation used in SKINNY-64-128

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

The **ShiftRows** (SR) operation is similar to the one used in the AES cipher; the difference consists in the direction in which the rows' values are circularly shifted – to the right side in the case of ciphers from the SKINNY family. As a consequence, the ShiftRows operation is actually the following permutation:

P = [0, 1, 2, 3, 7, 4, 5, 6, 10, 11, 8, 9, 13, 14, 15, 12]

The **MixColumn** (MC) operation represents the multiplication of the internal state with the following matrix:

$$M = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

4. BOOMERANG AND RECTANGLE RELATED-KEY ATTACKS

The boomerang attack is an adaptive chosen plaintext attack designed by David A. Wagner in 1999 [9]. The attack is based on differential cryptanalysis, more specifically, on the existence of two highly probable differentials, one for the encryption function (called the forward direction) and one for the decryption function (called the backward direction).

We assume that a block cipher E may be described as a composition of two block ciphers with less rounds: $E = E_1 \circ E_0$. The forward differential is a trail $\Delta \to \Delta'$ on E_0 with probability p and the backward differential is a trail $\nabla \to \nabla'$ on E_1^{-1} with probability q.

After finding the differential trails, the attack consists of two phases:

I. In the first phase, a number of *quartets* (P_1, P_2, Q_1, Q_2) are generated following the steps below, with the probability of finding a good quartet equal to p^2q^2 (E^{-1} denotes the decryption function and K denotes the secret key):
   1. generate a random plaintext P_1 and compute $P_2 = P_1 \oplus \Delta$
   2. compute $C_1 = E(P_1, K)$ and $C_2 = E(P_2, K)$
   3. compute $D_1 = C_1 \oplus \nabla$ and $D_2 = C_2 \oplus \nabla$
   4. compute $Q_1 = E^{-1}(D_1, K)$ and $Q_2 = E^{-1}(D_2, K)$
   5. if $Q_1 \oplus Q_2 = \Delta$, store the quartet (P_1, P_2, Q_1, Q_2).

II. The second phase of the attack is the same as the one performed in a classical differential attack, offering the attacker the ability of computing some parts of the key, considered secret in the first place.

The rectangle attack [10] is based on the boomerang attack, the main difference between them being the fact that, in the case of a rectangle attack, multiple differentials are used in both directions. As a consequence, the probability of finding a good quartet is increased from p^2q^2 to $\hat{p}^2\hat{q}^2$, where:

$$\hat{p} = \sqrt{\sum_{\Delta'} \Pr^2[\Delta \to \Delta']} \tag{1}$$

$$\hat{q} = \sqrt{\sum_{\nabla'} \Pr^2[\nabla \to \nabla']} \tag{2}$$

The related-key attacks, introduced in 1994 by Biham et al. [11], represent a type of differential cryptanalysis where the attacker knows or chooses a relation between two or more keys and studies the influence of these relations on the resulting ciphertexts. The relation between the keys can be an arbitrary bijective function; the simplest and most used such function is the XOR sum with a constant, an attack constructed in this way being very similar to classical differential cryptanalysis.

In this paper we present five rectangle attacks in the related-key scenario: each encryption/decryption is applied using a different tweakey, related to each other, based on some well-chosen differences. The design of the two differential trails is influenced by the chosen tweakey differences, denoted Δ_F (the tweakey difference in the forward direction) and Δ_B (the tweakey difference on the backward direction):

1. Generate a random plaintext P_1 and compute $P_2 = P_1 \oplus \Delta$
2. Compute $C_1 = E(P_1, K_1)$ and $C_2 = E(P_2, K_2)$, where $K_2 = K_1 \oplus \Delta_F$
3. Compute $D_1 = C_1 \oplus \nabla$ and $D_2 = C_2 \oplus \nabla$
4. Compute $Q_1 = E^{-1}(D_1, K_3)$ and $Q_2 = E^{-1}(D_2, K_4)$, where $K_3 = K_1 \oplus \Delta_B$ and $K_4 = K_2 \oplus \Delta_B$
5. If $Q_1 \oplus Q_2 = \Delta$, store the quartet (P_1, P_2, Q_1, Q_2).

4.1. The 16-round SKINNY-64-128 attack

In order to find good forward and backward differentials we used a technique based on a customized form of the SKINNY-64-128 cipher: we removed the SubCells operation and the LFSR application from the

AddRoundTweakey operation; as a consequence, the differences between two internal states are modified in a linear manner, as can be seen below:

$$Enc_{custom}(\delta, \Delta) = Enc_{custom}(P, T) \oplus Enc_{custom}(P \oplus \delta, T \oplus \Delta)$$

Due to this fact, the input that we used for our customised version of SKINNY-64-128 was actually the difference matrix between the internal states (instead of an internal state/plaintext) and the difference matrix between all four tweakeys matrices; moreover, the values of the input matrices were either 1 (having the significance that the initial plaintexts/tweakeys matrices were different) or 0 (the initial plaintexts/tweakeys matrices had the same value).

In this way, we were able to exploit some properties of the MixColumn operation that influence the propagation of differences (the case in which the differences of a column of the internal state are equal, fact that leads, in some cases, to differences' cancellation); also, this approach allowed us to perform exhaustive searches on inputs (2^{16} possible differences of the plaintexts and 2^{16} possible differences between the XOR sum of each tweakey matrices).

In the exhaustive search step, in order to decide if a trail is a feasible one, we computed its maximum probability (the maximum probability of the S-box appliance, 2^{-2}, multiplied by the number of ones in the beginning of each round).

The last step that we performed, after obtaining some feasible trails, was to find differences (with values between 1 and 15) based on the S-box's XOR profile, that keep the probability as high as possible. We used the same approach for the encryption function and for the decryption function, obtaining, for different number of rounds, appropriate trails defined as in the classical approach.

The differential trails that we used in our attack are presented below, in Figure 1 and Figure 2. For simplicity, we have denoted with uppercase the hexadecimal values of the differences, and with lowercase (except the letters a, b, c, d, e, f, in order to avoid the confusion) all the possible values of the differences on a particular position. Also, if two differences from a round are denoted with the same lowercase letter, they must be equal (this property is used in the MixColumn step, to reduce the number of differences).

The blue squares denotes the positions in which the plaintexts are different, but the value of the difference between them is not important for the attack.

In the backward trail, we denoted the inverse of the round operations by "inv" plus the name of the operation.

Our 16 round rectangle attack is based on a set of distinguishers similar to the ones described in Figure 1 and Figure 2 (the distinguishers follow the same pattern, the positions with no difference will remain the same, Δ and ∇ are fixed, while Δ' and ∇' take all possible values).

Note that the tweakey differences are obtained by XORing the 4 tweakeys matrices (2 matrices from each tweakey). In both forward and backward direction, the tweakey differences are the one effectively used in the current round - in the backward case, the differences showed in the right column are obtained after applying the inverse of the LFSR and the inverse of the tweakey permutation (from the AddRoundTweakey function).

All the tweakey differences are uniquely determined by the initial differences on both tweakeys' matrices. While the difference on the first tweakey matrix remains the same (in the AddRoundTweakey operation, on the first tweakey matrix is performed only a permutation), the difference on the second tweakey matrix is modified by the LFSR application.

In our distinguishers, for the forward trail we used the difference 7 on the first tweakey matrix, and 5 on the second tweakey matrix, while for the backward trail, we used the difference 7 on the first tweakey matrix and the difference 15 on the second tweakey matrix.

Due to the fact that the first application of the SubCells operation is not dependent on the round tweakey, we were able to generate plaintext pairs for which, in the first round of the forward differential trail, the probability of difference propagation is 1 - we generated an auxiliary plaintext, we computed the corresponding auxiliary plaintext by XORing the difference from the forward differential in Figure 1, after the first Sbox application. In order to obtain the corresponding initial plaintexts, we applied the inverse of the SubCells layer on these auxiliary plaintexts.

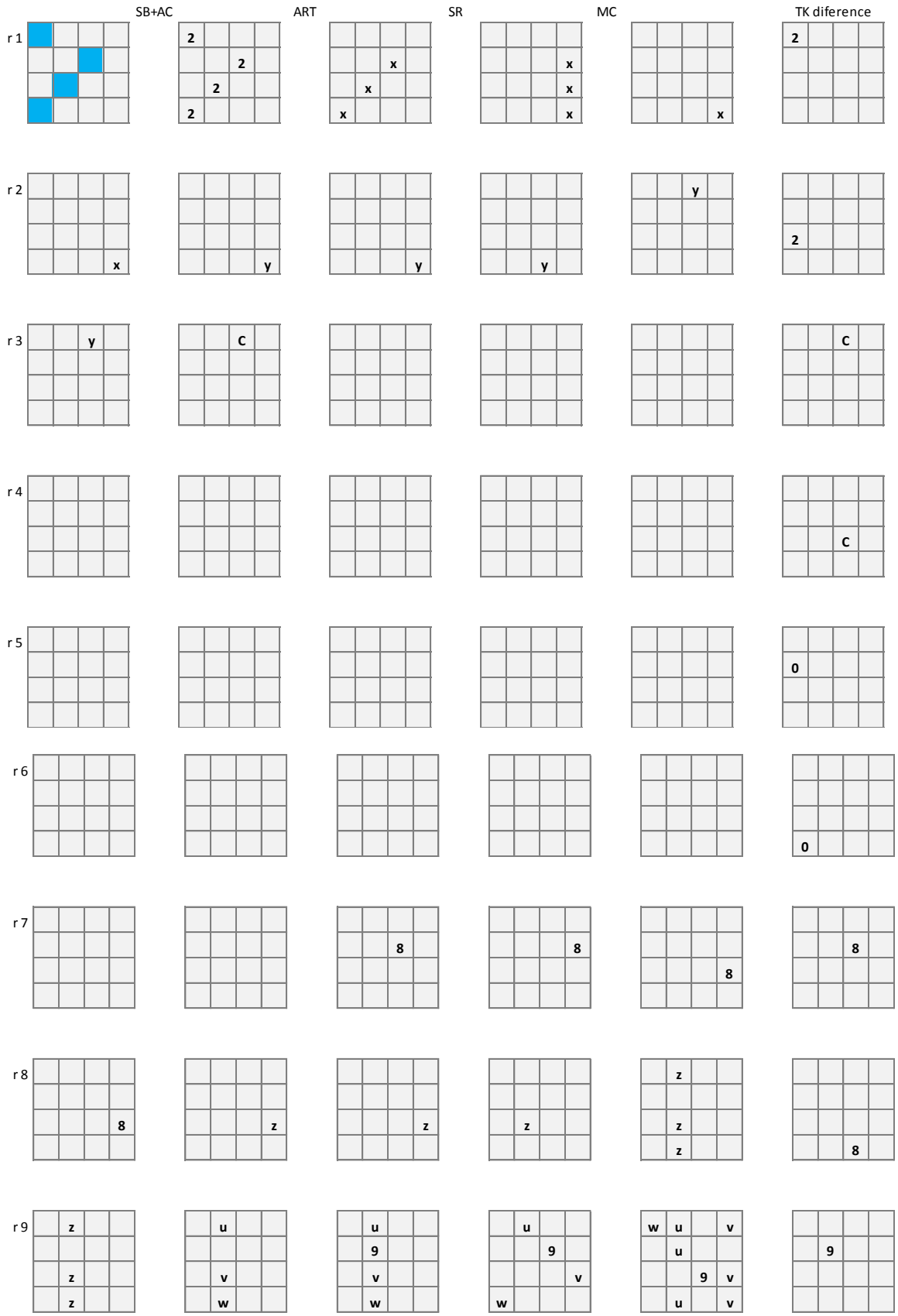

Fig. 1 – Example of forward differential trail used for the 16-round SKINNY-64-128 rectangle attack.

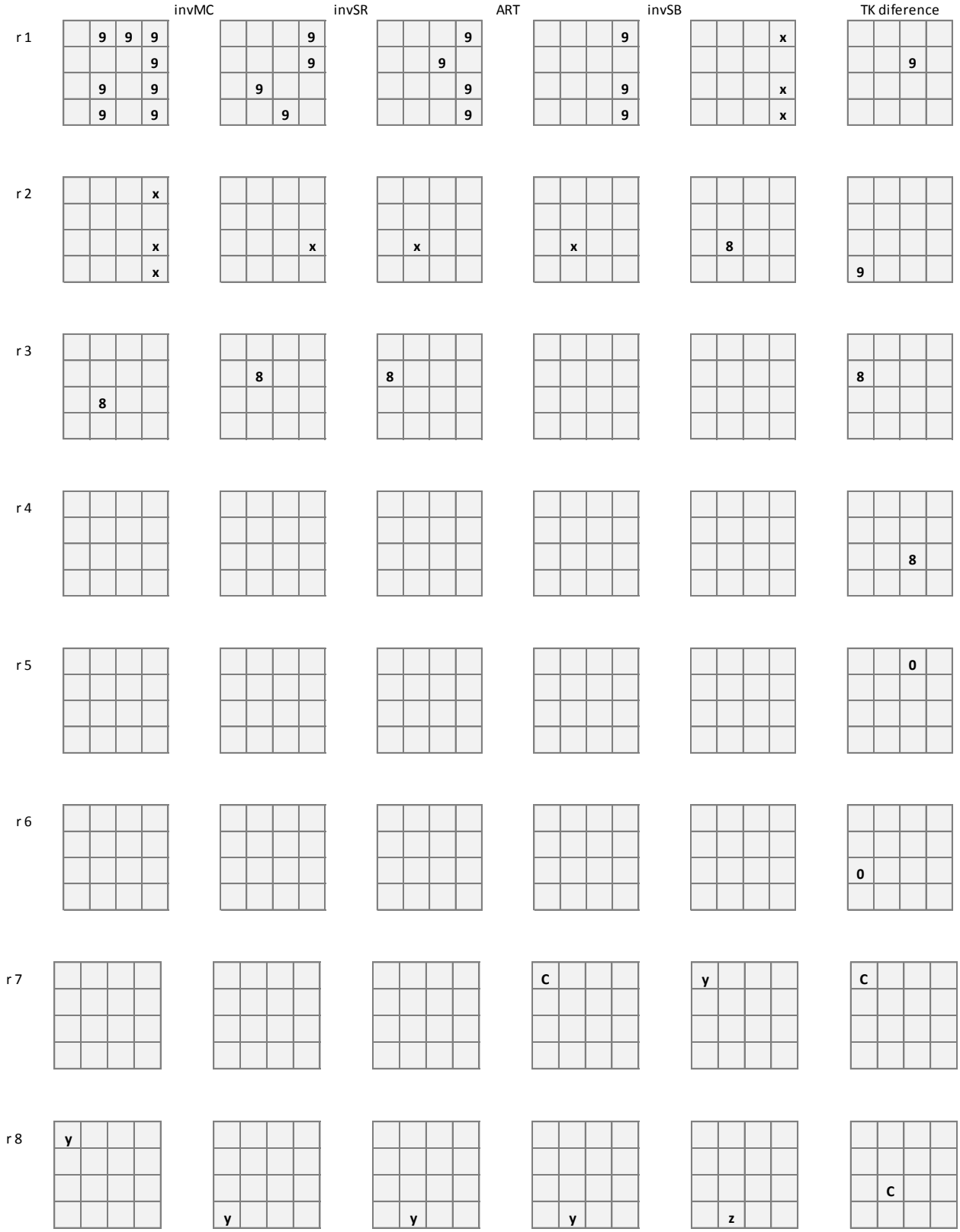

Fig. 2 – Example of backward differential trail used for the 16-round SKINNY-64-128 rectangle attack.

Applying this steps and equations (1) and (2), the probability of finding a good quartet, in our 16-round attack, is 2^{-17}. The pseudocode of the 16-round related-key rectangle attack on SKINNY-64-128:

Phase I

1. generate $2^{17} \times n$ plaintext pairs $(P_1, P_2 = P_1 \oplus \Delta)$, with the differences from round 1 in the forward trail, after the SubCells application
2. foreach pair (P_1, P_2):
   i. compute $C_1 = E(SB^{-1}(P_1), K_1)$ and $C_2 = E(SB^{-1}(P_2), K_2)$ using 16-round SKINNY-64-128, where $K_2 = K_1 \oplus \Delta F$ and ΔF is the tweakey difference from the first round of the forward trail
   ii. compute $D_1 = C_1 \oplus \nabla$ and $D_2 = C_2 \oplus \nabla$
   iii. compute $Q_1 = E^{-1}(D_1, K_3)$ and $Q_2 = E^{-1}(D_2, K_4)$, where $K_3 = K_1 \oplus \Delta_B$, $K_4 = K_2 \oplus \Delta_B$ and Δ_B is the tweakey difference from the first round of the backward trail
   iv. if $SB(Q_1) \oplus SB(Q_2) = P_1 \oplus P_2$ then add the quartet (P_1, P_2, Q_1, Q_2) to the quartet list

Phase II

foreach good *quartet* (P_1, P_2, Q_1, Q_2) stored in the *quartet_list*

1. compute the corresponding ciphertexts quartet (C_1, C_2, D_1, D_2)
2. foreach possible combination of values for the 4 key nibbles $TK1[2] \oplus TK2[2]$, $TK1[3] \oplus TK2[3]$, $TK1[7] \oplus TK2[7]$, $TK1[8] \oplus TK2[8]$:
   i. partially encipher (P_1, P_2) and (Q_1, Q_2) with 2 full rounds and 1 incomplete round (only SB and AC operations) – the remaining tweakey XOR sums may be 0
   ii. if the difference of the nibble in position 2 of the partially enciphered pairs is equal to 12, increment a counter corresponding to the current combination of values of the 4 key nibbles
   iii. the correct values of the key nibbles to be recovered are the ones with the highest frequency

Fig. 3 – The related-key rectangle attack pseudocode for 16-round SKINNY-64-128.

Regarding the attack's implementation, the following observations are useful:

- the number of good quartets needed in order to launch a successful attack is $n = 8$;
- we never obtained a single combination of values with the maximum frequency. In fact, in the key recovery phase we determined unique values for 3 key nibbles (the correct ones) and 2 possible values for the remaining nibble (the correct one is always between them);
- the time complexity of our attack is approximately 2^{22} encryptions/decryptions, computed as follows:
  - phase I: 2^{17} (from the probability of finding a good quartet) × 4 (the number of encryptions/decryptions per quartet) × 8 (the number of good quartets used in our attack) +
  - phase II: 8 (the number of good quartets used in our attack) × 2^{16} (the exhaustive search of the 4 tweakey nibbles)
- the data complexity of our attack is $2^{17} \times 2 \times 8$ plaintext/ciphertext pairs, while the stored data is of n quartets (n × 32 bytes).

4.2. The 18-round SKINNY-64-128 attacks

We designed two 18-round attacks on SKINNY-64-128, in two different scenarios (with known or unknown tweakey nibbles). The forward and backward differential trails (used in both the attacks) are based on the ones described in the previous section, each of them being extended with one round in the beginning – in consequence, the forward differentials have 10 rounds and the backward differentials have 8 rounds. Figure 4 depicts the first 2 rounds of the forward trails, while Figure 5 contains the last 2 rounds of the backward trails.

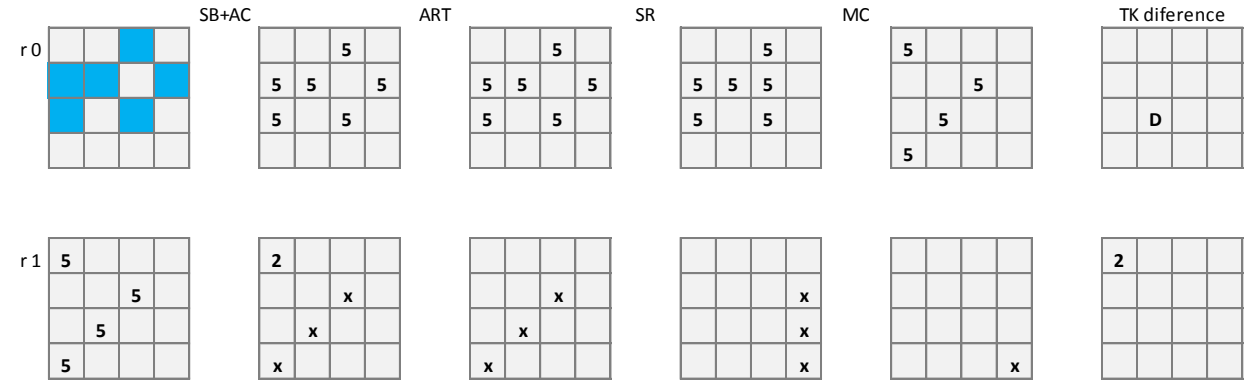

Fig. 4 – The first two rounds of the forward differentials for 18-round attacks.

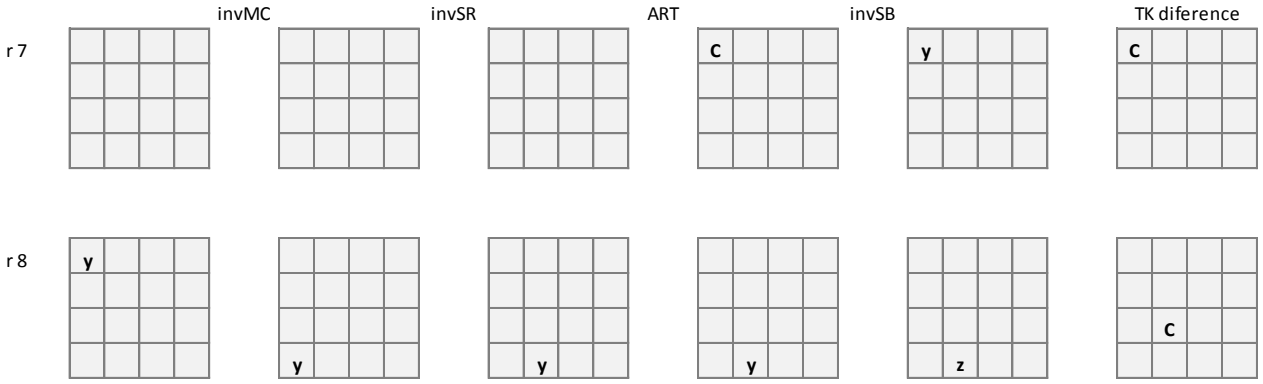

Fig. 5 – The last two rounds of the backward differentials for 18-round attacks.

4.2.1. The attack on 18-round Skinny-64-128 with unknown tweakey

In our distinguishers, for the forward trail we used the difference 7 on the first tweakey matrix, and the difference 10 on the second tweakey matrix, while for the backward trail, we used the difference 7 on the first tweakey matrix, and the difference 14 on the second tweakey matrix.

In this case, the probability of finding a good quartet is approximatively 2^{-45}. The pseudocode of our attack is described below:

The time complexity of our attack is approximately 2^{51} encryptions/decryptions, computed as follows:

- Phase I: 2^{45} (the probability of finding a good quartet) × 4 (the number of encryptions/decryptions per quartet) × n (the number of good quartets used in our attack) +
- Phase II: n (the number of good quartets used in our attack) × 2^4 (the exhaustive search of the one nibble) × (1+1) (the exhaustive search is performed independently for each of the two tweakey sum nibble found)

The data complexity of our attack is $2^{45} \times 2 \times n$ plaintext/ciphertext pairs, while the stored data is of n quartets ($n \times 32$ bytes). In the case of this attack, we consider $n = 16$ sufficient for the efficiency of the key recovery phase.

4.2.2 The attack on 18-round Skinny-64-128 with known tweakey nibbles

This attack differs from the previous one by the hypothesis that the attacker knows the values of $TK1[0] \oplus TK2[0]$, $TK1[2] \oplus TK2[2]$, $TK1[4] \oplus TK2[4]$ (12 known bits) or the values of the following nibbles: $TK1[0]$, $TK2[0]$, $TK1[2]$, $TK2[2]$, $TK1[4]$ and $TK2[4]$ (24 known bits).

Based on this new hypothesis, the probability of finding a good quartet is 2^{-29}, significantly higher than the one from the previous attack. We randomly generated p_1 and computed $p_2 = p_1 \oplus \Delta'$, where Δ' is the difference from the round 2 of the forward trail (we used $x = 2$), after the SubCells application. We

constructed $P_1 = f(p_1)$ and $P_2 = f(p_2)$, where $f(x) = SB^{-1}(ART(AC(SR^{-1}(MC^{-1}(SB^{-1}(x))))))$; the ART layer is applied using the zero value on the positions with unknown nibble sum. Due to this approach in the phase I of the attack, the probability of finding a good quartet increases from 2^{-45} to 2^{-29}.

Phase I

i. generate $2^{45} \times$ n plaintext pairs $(P_1, P_2 = P_1 \oplus \Delta)$, with the differences corresponding to the first round, after the SubCells application
ii. foreach pair (P_1, P_2):
iii. compute $C_1 = E(SB^{-1}(P_1), K_1)$ and $C_2 = E(SB^{-1}(P_2), K_2)$ using 18-round SKINNY-64-128, where $K_2 = K_1 \oplus \Delta F$ and ΔF is the tweakey difference from the first round of the forward trail
iv. compute $D_1 = C_1 \oplus \nabla$ and $D_2 = C_2 \oplus \nabla$
v. compute $Q_1 = E^{-1}(D_1, K_3)$ and $Q_2 = E^{-1}(D_2, K_4)$, where $K_3 = K_1 \oplus \Delta_B$, $K_4 = K_2 \oplus \Delta_B$ and Δ_B is the tweakey difference from the first round of the backward trail
vi. if $SB(Q_1) \oplus SB(Q_2) = P_1 \oplus P_2$ then add the quartet (P_1, P_2, Q_1, Q_2) to the *quartet_list*

Phase II

foreach good quartet (P_1, P_2, Q_1, Q_2) from the *quartet_list*

i. compute the corresponding ciphertexts quartet (C_1, C_2, D_1, D_2)
ii. foreach possible value of $TK1^{18}[3] \oplus TK2^{18}[3]$ where $TKi^{18}[j]$ is the j^{th} nibble of the tweakey *i*, from the round 18:
    1. partially decipher (C_1, C_2) and (D_1, D_2) with 1 incomplete round (only the inverse of MixColumn, the inverse of ShiftRows and AddConstants)
    2. if the differences on the nibbles from positions 3, 11 and 15 of the partially decrypted pairs are equal (after applying the XOR with the current value of the key nibble and the inverse of the SubCells operation), increment a counter corresponding to the current value of the key nibble
iii. foreach possible value of $TK1^{18}[7] \oplus TK2^{18}[7]$:
    1. compute $C_1' = MC^{-1}(SR^{-1}(C_1))$, $C_2' = MC^{-1}(SR^{-1}(C_2))$
    2. compute $D_1' = MC^{-1}(SR^{-1}(D_1)), D_2' = MC^{-1}(SR^{-1}(D_2))$
    3. compute $c_i = S^{-1}(S^{-1}(C_i'[7] \oplus TK1^{18}[7] \oplus TK2^{18}[7]) \oplus S^{-1}(C_i'[15]))$
    4. compute $d_i = S^{-1}(S^{-1}(D_i'[7] \oplus TK1^{18}[7] \oplus TK2^{18}[7]) \oplus S^{-1}(D_i'[15]))$
    5. *if* $c_1 \oplus c_2 = 8$ and $d_1 \oplus d_2 = 8$, increment a counter corresponding to the current value of the key nibble
iv. the correct key nibbles are between the ones with the highest frequencies.

Fig. 6 – Related-key rectangle attack pseudocode for 18-round SKINNY-64-128, with unknown tweakey nibbles.

The pseudocode of the attack is described in Figure 7.

The time, data and memory complexity is computed as is the previous attack. In our implementation, we used $n = 4$ quartets: the overall time complexity is approximatively 2^{33} encryptions, the data complexity is 2^{32} plaintexts/ciphertexts pairs and the needed memory is of 4 quartets (128 bytes).

4.3 The 20-round SKINNY-64-128 attacks

The previous related-key rectangle attacks for 18-round SKINNY-64-128 cipher can be extended to 20-round attacks by adding one round in the beginning of the forward differentials and one round to the final of the backward differentials, as showed in Figure 8 and Figure 9.

We have designed two 20-round attacks, one in which we consider the tweakey unknown and one in which some nibbles of the tweakey are considered known.

Phase I

1. generate $2^{29} \times$ n pairs $(p_1, p_2 = p_1 \oplus \Delta')$, with the differences from the second round of the forward trail, after the SubCells application
2. foreach pair (p_1, p_2):
   i. compute $P_1 = f(p_1)$ and $P_2 = f(p_2)$ as described above
   ii. compute $C_1 = E(P_1, K_1)$ and $C_2 = E(P_2, K_2)$ using 18-round SKINNY-64-128, where $K_2 = K_1 \oplus \Delta_F$ and Δ_F is the key difference from the first round of the forward trail
   iii. compute $D_1 = C_1 \oplus \nabla$ and $D_2 = C_2 \oplus \nabla$
   iv. compute $Q_1 = E^{-1}(D_1, K_3)$ and $Q_2 = E^{-1}(D_2, K_4)$, where $K_3 = K_1 \oplus \Delta_B$, $K_4 = K_2 \oplus \Delta_B$ and Δ_B is the key difference from the first round of the backward trail
   v. if $f^{-1}(Q_1) \oplus f^{-1}(Q_2) = p_1 \oplus p_2$ then add the quartet (P_1, P_2, Q_1, Q_2) to the *quartet_list*

Phase II

foreach good quartet (P_1, P_2, Q_1, Q_2) stored in the *quartet_list*

1. compute the corresponding ciphertexts quartet (C_1, C_2, D_1, D_2)
   i. foreach possible value of $TK1^{18}[3] \oplus TK2^{18}[3]$, where $TKi^{18}[j]$ is the j^{th} nibble of the tweakey i, from the round 18:
      - partially decipher $(\boldsymbol{C_1}, \boldsymbol{C_2})$ and $(\boldsymbol{D_1}, \boldsymbol{D_2})$ with 1 incomplete round (only the inverse of MixColumn, the inverse of ShiftRows and AddConstants)
      - if the differences on the nibbles from positions 3, 11 and 15 of the partially decrypted pairs are equal (after XORing the nibbles from position 3 with the current value of the key nibble and applying the inverse of the SubCells operation, on the nibbles from positions 3, 11 and 15), increment a counter corresponding to the current value of the key nibble
   ii. foreach possible value of $TK1^{18}[7] \oplus TK2^{18}[7]$:
      - partially decipher (C_1, C_2) and (D_1, D_2) with 2 full rounds (only the nibbles involved in order to compute the nibble in position 9)
      - compute $C_1' = MC^{-1}(SR^{-1}(C_1))$, $C_2' = MC^{-1}(SR^{-1}(C_2))$
      - compute $D_1' = MC^{-1}(SR^{-1}(D_1))$, $D_2' = MC^{-1}(SR^{-1}(D_2))$
      - compute $c_i = S^{-1}(S^{-1}(C_i'[7] \oplus TK1^{18}[7] \oplus TK2^{18}[7]) \oplus S^{-1}(C_i'[15]))$
      - compute $d_i = S^{-1}(S^{-1}(D_i'[7] \oplus TK1^{18}[7] \oplus TK2^{18}[7]) \oplus S^{-1}(D_i'[15]))$
      - if $c_1 \oplus c_2 = 8$ and $d_1 \oplus d_2 = 8$, increment a counter corresponding to the current value of the key nibble
   iii. the correct key nibbles are between the ones with the highest frequencies

Fig. 7 – Related-key rectangle attack pseudocode for 18-round SKINNY-64-128, with known tweakey nibbles.

In our distinguishers, for the forward trail we used the difference 7 on the first tweakey matrix, and the difference 10 on the second tweakey matrix, while for the backward trail, we used the difference 7 on the first tweakey matrix, and the difference 14 on the second tweakey matrix.

4.3.1. The attack on 20-round Skinny-64-128 with unknown tweakey

The pseudocode of this attack is similar to the one used for the 18-round attack (described in Figure 6), with the difference that the encryption and the decryption functions will be applied for 20 rounds of SKINNY-64-128 and the difference between the plaintexts is the one described in Figure 8. Also, in the key recovery phase, the exhaustive search is performed on the tweakey nibbles $TK1^{20}[13] \oplus TK2^{20}[13]$ and $TK1^{20}[11] \oplus TK2^{20}[11]$.

The probability of finding a good quartet for this attack is approximately 2^{-85}.

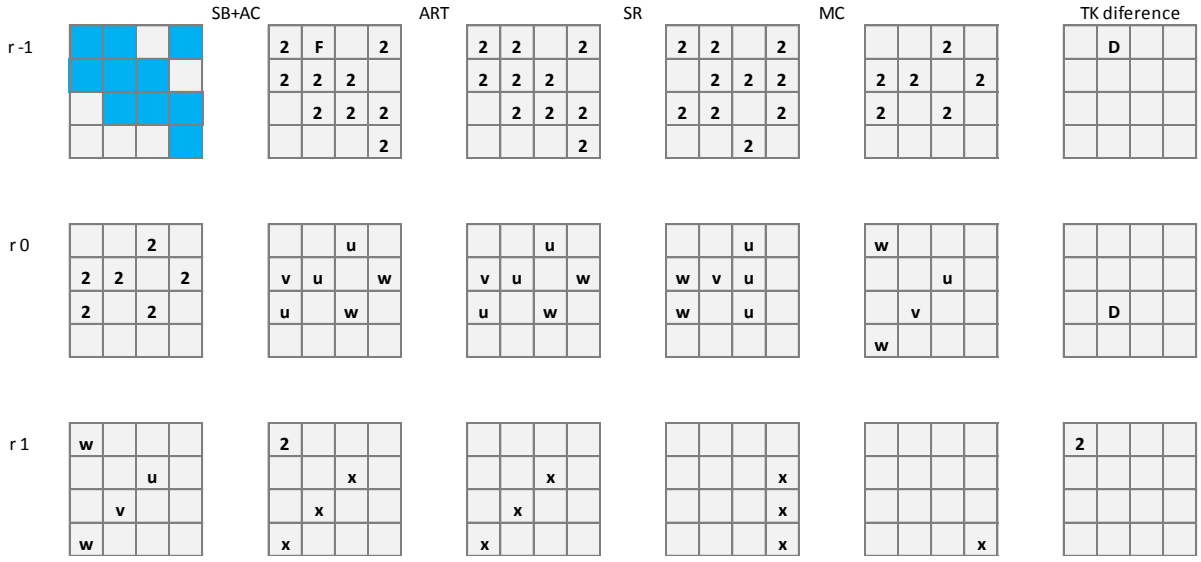

Fig. 8 – The first three rounds of the forward differentials for the 20-round attacks.

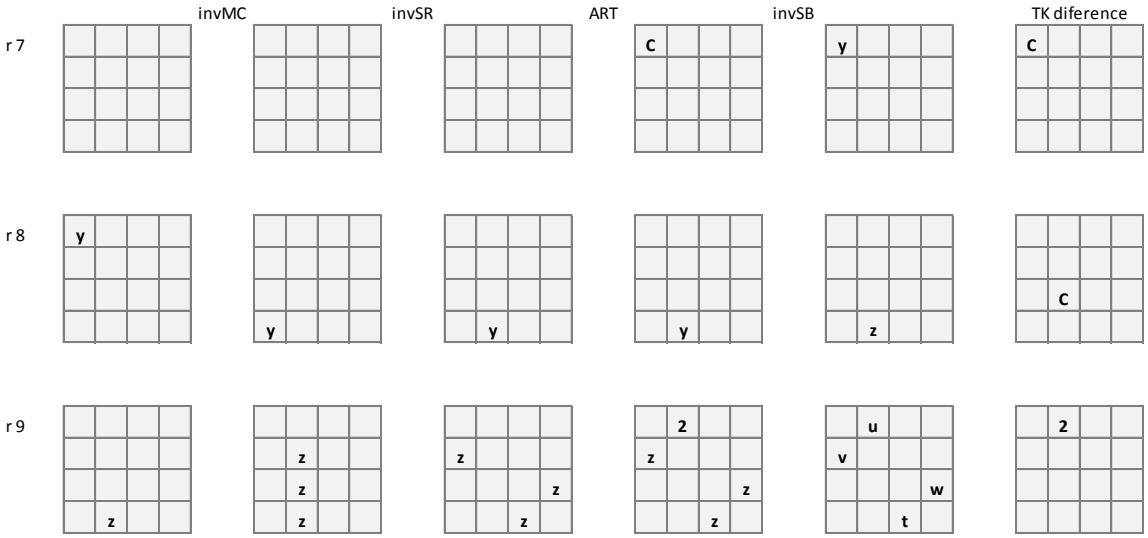

Fig. 9 – The last three rounds of the backward differentials for the 20-round attacks.

The time complexity of this attack is approximately 2^{91} encryptions/decryptions, computed as a sum of the following two quantities:

- Phase I: 2^{85} (the probability of finding a good quartet) × 4 (the number of encryptions/decryptions per quartet) × *n* (the number of good quartets used in our attack)
- Phase II: *n* (the number of good quartets used in our attack) × 2^4 (the exhaustive search for one nibble) × (1+1) (the exhaustive search is performed independently for each of the two tweakey sum nibbles).

The data complexity of our attack is 2^{85} × 2 × n plaintext/ciphertext pairs, while the stored data is of n quartets (n × 32 bytes). In the case of this attack, we consider n = 16 sufficient for the efficiency of the key recovery phase.

4.3.2. Attack on 20-round Skinny-64-128 with known tweakey

This attack is also similar to the 18-round attack that uses the hypothesis that some of the tweakey nibbles are known. In the case of the 20-round attack, the hypothesis used is that the attacker knows the XOR sum between the two matrices of the initial tweakey, on the positions 0, 1, 2, 3, 5 and 7 (24 known bits) or

the actual values of the two matrices on the mentioned positions (48 known bits). This new hypothesis allows the attacker to generate good quartets, in the same manner as the one described in the section 4.2.2, with an improved probability of approximatively 2^{-61}.

The time complexity of this attack, computed in the hypothesis that $n = 16$ is a sufficient number of good quartets, is approximately of 2^{67} encryptions, the data complexity is approximately 2^{65} pairs of plaintexts/ciphertexts, while the memory needed is the same, 512 bytes, used for good quartets storage.

4.4. Versatility of the related-key rectangle attacks

All the related-key rectangle attacks described above, using exactly the same differentials and pseudocode, can be also applied for the Skinny-64-192 version, using zero differences in the third tweakey matrix. By using the same trails structures with other difference values (and, implicitly, other probabilities), similar attacks could be applied for up to 20-round Skinny-128-256 and 20-round Skinny-128-384.

5. IMPOSSIBLE RELATED-KEY DISTINGUISHERS

The impossible differential cryptanalysis was introduced independently by Knudsen [11] and Biham et al. [12]. Opposite to the classical differential approach, where the main idea is to find a differential trail with high probability, in the case of impossible differential cryptanalysis, a differential trail with probability equal to zero is searched. If an impossible input-output difference is obtained when using some values of the key, these values must be wrong.

Using this observation, a big part of the possible keys may be filtered, obtaining a small set of potentially correct keys. If the key filtering step does not provide a unique key, the right key values may be found by performing an exhaustive search on the remaining keys.

In our research we focused on finding impossible differential trails for SKINNY-64-128, in the related-key scenario. We managed to construct 7 impossible differential trails on 11-round SKINNY-64-128 that are related to the one used by Ankele et al. in [5]. Actually, we proved the fact that, if we use only two nibbles of the tweakey difference, positioned on the row 3 or 4 of the tweakey matrices (the difference on the two tweakey matrices is on the same position), then we can construct an 11-round impossible differential trail.

Similar to the research work from [5], in the first phase of our research, in order to design the impossible differential trails, we only looked for the positions with zero difference.

We also studied the possibility of constructing impossible differential trails that take into account the non-zero values of the differences, the contradiction being obtained in the case that two non-zero differences are not equal. In this context, we analysed some custom versions of the SKINNY-64-128 Sbox's XOR profile.

More specifically, we computed the probability that an output difference is obtained after two or three consecutive applications of the S-box function. As a consequence, we have shown that, after 3 consecutive Sbox applications, every output value (except 0) may be generated using any of the possible input values.

We focused on the exploitation of the input-output pairs of differences that have 0 probability of appearance after two consecutive applications of the Sbox; we found impossible distinguishers similar to the one described in [5].

In the same context, we looked for impossible differential trails up to 15 rounds. Our search used the following hypotheses: the plaintexts and ciphertexts used are equal, while the tweakey takes all the possible differences on one nibble of TK1 and the corresponding nibble of TK2.

The maximum length of such impossible distinguishers found by our implementation was 13 rounds, following two possible patterns: one of them is similar to the one used by Ankele et al. in [5] (the 11-round distinguisher is extended, in the beginning, with two rounds in which the tweakey and plaintext differences are zero), while in the other pattern is extends the 11-round differential with two rounds at the end, using the same zero difference property on the tweakey and ciphertexts.

In our implementation, when searching for an n rounds impossible differential, we looked for all possible combinations of forward-backward trails. One example of another 11-round distinguisher, constructed by our approach, is shown in the Figure 10.

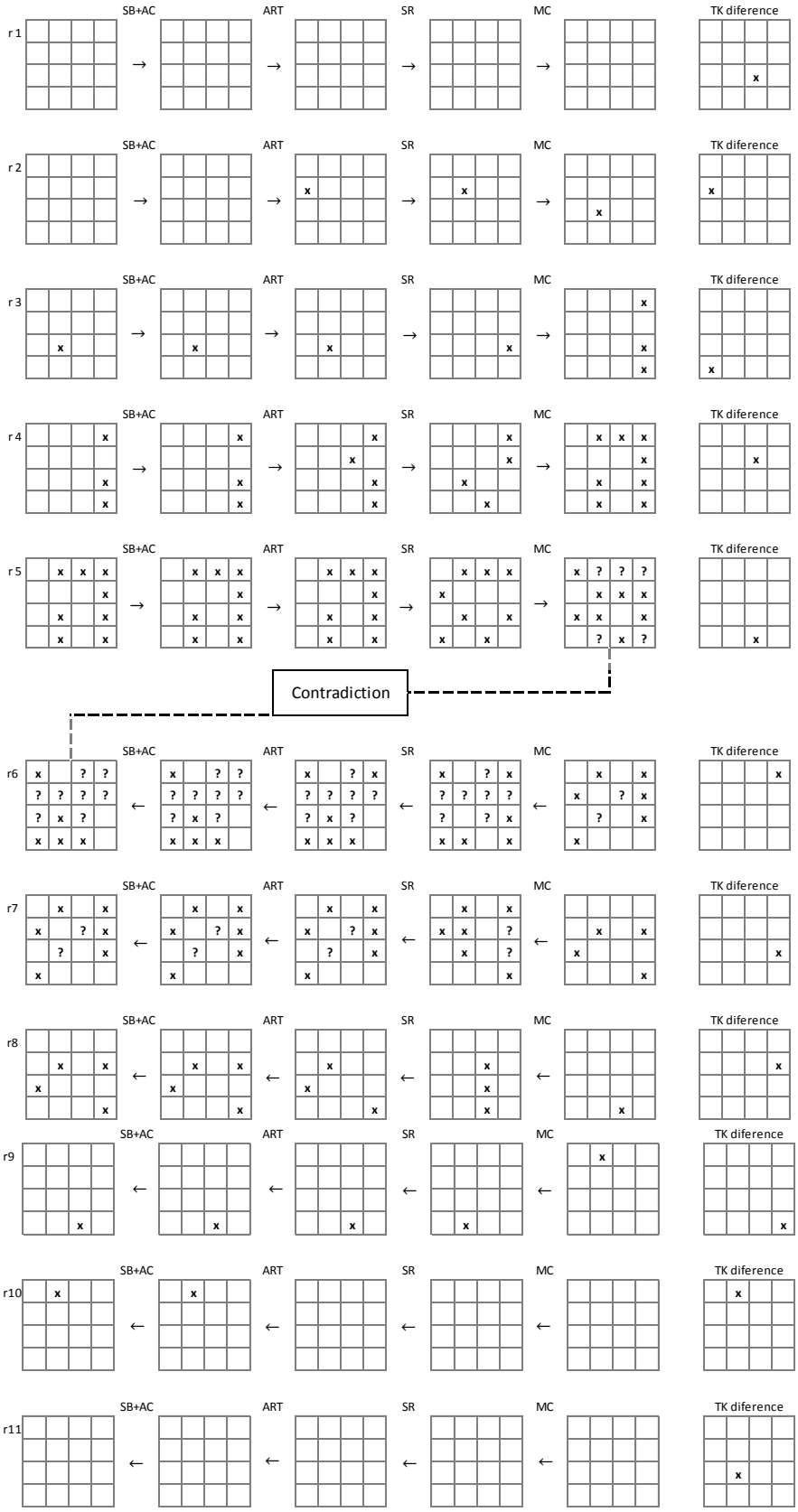

Fig. 10 − Impossible differential of SKINNY-64-128 in the related-key scenario.

6. SUMMARY OF RESULTS

Our results and the ones obtained in previous attacks on round-reduced SKINNY-64-128, are summarized in Table 4. As far as we know, our paper introduces the first practical attacks against 16-round and 18-round SKINNY-64-128. We do not know if the other mentioned attacks, when reduced to 16 or 18 rounds, become practical as well.

Table 4

Summary of attacks on round-reduced SKINNY-64-128

Reference	Attack type		Rounds number	Tweak size (bits)	Complexity		
					Time	Data	Memory
[5]	Related-Key	Impossible Differential	21	0	$2^{74,6}$	2^{72}	
[5]			22	48	2^{71}	2^{71}	
[4]		Impossible Truncated Differential	23	0	$2^{124,2}$	$2^{62.47}$	2^{124}
[4]		Rectangle	21	0	$2^{87,9}$	2^{54}	2^{54}
[4]			22	0	$2^{109,9}$	2^{63}	2^{63}
Section 4.1			16	0	2^{22}	2^{22}	2^{8}
Section 4.2.1			18	0	2^{51}	2^{51}	2^{9}
Section 4.2.2			18	24	2^{33}	2^{33}	2^{7}
Section 4.3.1			20	0	2^{91}	2^{91}	2^{9}
Section 4.3.2			20	48	2^{67}	2^{67}	2^{9}
[6]	Single-Key	Impossible Differential	20	0	$2^{121.08}$	$2^{47,69}$	$2^{74,69}$

7. FIXED POINTS ANALYSIS

In this paper, we denote by fixed point a (plaintext, tweakey) pair for which the ciphertext, obtained after applying a round-reduced version of SKINNY-64-128, is equal to the original plaintext.

The initial approach was a heuristic one, based on genetic algorithms, and led to the discovery of fixed points for 2-round SKINNY-64-128. The genetic algorithms paradigm [14] has been successfully applied for various optimization problems. In the context of the genetic algorithms, the solutions are usually called individuals; in our case, the individuals are represented by tweakeys. The goal of our approach was to minimize the number of differences between the plaintext and the corresponding ciphertext, when using these tweakeys.

The function that we wanted to optimise (called the objective function) was the number of differences between the (randomly) generated, but fixed plaintext and the ciphertext, obtained using individuals – a population of tweakeys. The purpose of the algorithm was finding a global minimum of the objective function, and, in consequence, a fixed point. By increasing the number of rounds, the genetic algorithms found only local solutions (pairs with a small number of differences). Examples of results obtained using this approach are described in the following table.

Table 5

Examples of the heuristic approach results regarding the fixed-point analysis

Rounds number	Local minimum	Plaintext	Tweakey
2	0	0x42d6cd7f101d1bc5	0x1a7acca08aac77def7f03dcbe7591832
3	2	0x00e1770b00dd0000	0xdd6c0000547300af00ff000000001000
4	5	0x9100000000009b00	0x00000054000014000000000dd0000

A detailed description of the genetic algorithm scheme and the specific settings we used in our research is out of the goal of this paper. More information regarding the applicability of genetic algorithms in cryptanalysis can be found in [15, 16].

The second approach, inspired by the results obtained using the heuristic approach, is an algebraic one, based on the system of equations corresponding to the round-reduced versions of the cipher. This approach allowed us to find fixed-points for 2, 4, 6 and 8 round-reduced SKINNY-64-128.

Let us denote the 1-round encryption function by $Enc(p, t_i)$, where by p we denote the internal state/plaintext and by t_i we denote the tweakey used for the round i. For example, a 2-round SKINNY-64-128 will be represented by $Enc(Enc(p, t_1), t_2)$. $Enc^n(p, T)$ represents the n-round encryption of the plaintext p, using the initial tweakey T. Also, let us denote the 1-round decryption function by $Dec(c, t_i)$.

In order to reduce the complexity of the equations, we used a meet-in-the-middle based technique. For the 2-rounds analysis, instead of writing the equations corresponding to $Enc(Enc(p, t_1), t_2) = c$, we wrote the equations corresponding to $Enc(p, t_1) = Dec(c, t_2)$. We were able to prove that, for any randomly generated plaintext, there exists 2^{64} tweakeys such that the pair (plaintext, tweakey) is a fixed point. Also, for any randomly generated plaintext, we were able to generate all such tweakeys.

In order to find 4, 6 and 8-round fixed-points, we searched for the (plaintext, tweakey) pairs that have one more property:

$$Enc^{2n}(p, T) = Enc^n(p, T) = p \quad (3)$$

where $2n = 4, 6$ or 8.

This new property allowed us to reuse the equations corresponding to the n-round encryption, so the complexity of these equations is again decreased.

For the 4-round encryption, we were able to prove that, for any possible plaintext p, there exists a unique tweakey T such that the pair (p, T) is a fixed-point for the 2- and 4- round reduced SKINNY-64-128; we were also able to generate, for any randomly generated p, the tweakey that has the property previously mentioned.

In the case of 6-round and 8-round SKINNY-64-128 encryption, for any randomly chosen plaintext, we were able to generate, if any, all the tweakeys for which the equation (3) holds. For both the 6-round and 8-round encryption, we found plaintexts for which there exists up to 5 tweakeys such as the (plaintext, tweakey) pair is a solution for equation (3). We also found plaintexts for which there are no tweakeys with the previously described property. Some examples of the results that we obtained are listed in Table 6.

Table 6

Examples of 6-rounds and 8-rounds fixed points

Rounds number	Plaintext	Tweakey
6	0x0000000000000011	0xa7377562c9b432d12cfa189995daf1d9
		0x1cd2dca2bfa801c83639e634624cdf6b
		0xc47681e35dacba9096a6052b754164e3
		0xb763abbf21be19d6c1940438a64a981d
		0x6d8e18dd135fb163caf3a0ea5dd2cf83
8	0x401a4911c9645c20	0x0f44a9a7c0ae176d1a47208279b9ec95
		0x3760f977058522e1e32bd643cffa16f0
		0x402a43bce6ac433105865444570fcf1a
		0xf45d3a65d94ca3984313bc38d05d3de1
		0xa503657717d8f618ceb701fd322d65fa

8. CONCLUSION. FUTURE WORK

In this paper we propose some new related-key attacks for round-reduced SKINNY-64-128 lightweight block cipher. We successfully designed five attacks on 16, 18 and 20 rounds of SKINNY-64-128 (also suitable for SKINNY-64-192), in both the hypothesis that the tweakey has or has not a tweak. To the best of our knowledge, these are the first practical attacks introduced against any cipher of the SKINNY family. We have also focused our research on finding new properties of round-reduced SKINNY-64-128, introducing some fixed point analysis on up to 8 rounds and some new impossible related-key distinguishers.

Our future work will include, but not limit to, some new approaches that will allow us to reduce the complexity of our attacks and to attack a higher number of rounds. We also intend to continue our research

regarding the fixed-points analysis and its possible applicability in different scenarios of attack, targeting a higher number of rounds of SKINNY-64-128 and also other ciphers from the SKINNY family.

REFERENCES

1. C. BEIERLE, J. JEAN, S. KŐLBL, G. LEANDER, A. MORADI, T. PEYRIN, Y. SASAKI, P. SADRICH, S. M. SIM, *The SKINNY Family of Block Ciphers and its Low-Latency Variant MANTIS*, in Cryptology ePrint Archive – Report 2016/660, 2016.
2. *SKINNY Cipher Competition*, https://sites.google.com/site/skinnycipher/cryptanalysis-competition, 2016-2017.
3. R. BEAULIEU, D. SHORS, J. SMITH, S. TREATMAN-CLARK, B. WEEKS, L. WINGERS, *The Simon and Speck Families Of Lightweight Block Ciphers*, in Cryptology ePrint Archive – Report 2013/404, 2013.
4. G. LIU, M. GHOSH, L. SONG, *Security Analysis of SKINNY under Related-Tweakey Settings*, in Cryptology ePrint Archive: Report 2016/1108, 2016
5. R. ANKELE, S. BANIK, A. SHAKRABORTI, E. LIST, F. MENDEL, S. M. SIM, G. WANG, *Related-Key Impossible-Differential Attack on Reduced-Round SKINNY*, in Cryptology ePrint Archive – Report 2016/1127, 2016.
6. M. TOLBA, A. ABDELKHALEK, A. M. YOUSSEF, *Impossible Differential Cryptanalysis of Reduced-Round SKINNY*, in Cryptology ePrint Archive – Report 2016/1115, 2016.
7. S. SADEGHI, T. MOHAMMADI, N. BAGHERI, *Cryptanalysis of Reduced round SKINNY Block Cipher*, in Cryptology ePrint Archive – Report 2016/1120, 2016.
8. J. JEAN, I. NIKOLIC, T. PEYRIN, *Tweaks and keys for block ciphers: the TWEAKEY framework*, in ASIACRYPT 2014 Proceedings, Part II, **8874** of LNCS, Springer, pp. 274-288
9. D. WAGNER, *The boomerang attack*, 6th International Workshop on Fast Software Encryption (FSE'99), Spinger-Verlag pp. 156-170.
10. E. BIHAM, O. DUNKELMAN, N. KELLER, *The Rectangle Attack - Rectangling the Serpent*, Advances in Cryptology (EUROCRYPT 2001), pp. 340-357.
11. E. BIHAM, *New types of cryptanalytic attacks using related keys*, in Journal of Cryptology, 7.4, 1994, pp. 229-246.
12. L. KNUDSEN, *DEAL – A 128-bit Block Cipher*, in NIST AES Proposal, 1998.
13. E. BIHAM, A. BIRYUKOV, A. SHAMIR, *Cryptanalysis of SKIPJACK Reduced to 31 Rounds Using Impossible Differentials*, in EUROCRYPT. Lecture Notes in Computer Science, **1592**, pp.12-23, Springer, 1999.
14. J.H. HOLLAND, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.
15. G. NEGARA, *An Evolutionary Approach for the Playfair Cipher Cryptanalysis*, Proceedings of the 2012 International Conference on Security & Management (SAM 2012), USA, 2012, pp. 8-14.
16. G. NEGARA, *Nature-inspired Computation. Applications in Bioinformatics and Cryptography*, PhD Thesis, "Al. I. Cuza" University of Iasi, Romania, Faculty of Computer Science, 2013.