Rotational-XOR Cryptanalysis of Reduced-round SPECK

Yunwen Liu[1,2], Glenn De Witte[1*], Adrián Ranea[1] and Tomer Ashur[1]

[1] imec-COSIC KU Leuven, Leuven, Belgium
[2] College of Science, National University of Defense Technology, Changsha, China
glenn.dewitte-@-skynet.be
[yunwen.liu,tomer.ashur,adrian.ranea]-@-esat.kuleuven.be

Abstract. In this paper we formulate a SAT/SMT model for Rotational-XOR (RX) cryptanalysis in ARX primitives for the first time. The model is successfully applied to the block cipher family Speck, and distinguishers covering more rounds than previously are found, as well as RX-characteristics requiring less data to detect. In particular, we present distinguishers for 10, 11 and 12 rounds for Speck32/64 which have better probabilities than the previously known 9-round differential characteristic, for a certain weak key class. For versions of Speck48, we present several distinguishers, among which the longest one covering 15 rounds, while the previously best differential characteristic only covered 11.

Keywords: Rotational cryptanalysis · ARX · RX-difference · Weak keys · SAT/SMT

1 Introduction

Simon and Speck are two families of lightweight block ciphers designed by the United States National Security Agency (US NSA) and published in 2013 [BSS+15]. The Speck family was designed using the ARX structure, meaning that the only operations used are modular addition, cyclic rotation, and exclusive or (XOR). The family includes 10 members, differing in their block and key sizes. Indeed, due to their claimed efficiency, the two ciphers were the subject of extensive research, and are promoted as candidates into various standards.

Rotational cryptanalysis is a related-key chosen plaintext cryptanalytic technique suggested by Khovratovich *et al.* in [KN10, KNP+15]. In essence, when using rotational cryptanalysis, the adversary asks for the encryption of a pair of plaintexts, where one plaintext is obtained through a cyclic rotation of the other. This is done under two related keys which are also a rotational pair. Khovratovich *et al.* showed that the rotational relation between the two inputs is preserved with some probability through the ARX operations. A countermeasure proposed against rotational cryptanalysis is to XOR round dependent constants, which skews the propagation probability. Some works [BDPVA13, ANWOW13, FLS+10] overcame this by employing ad-hoc approaches that avoid the round constants [BDPVA13] or using an internal pattern within the constants [FLS+10].

*Corresponding author

Related work. Since its publication in 2013, SPECK has received a number of cryptanalyses, most of which focus on statistical analyses such as differential and linear cryptanalysis. In order to find good distinguishers, a study line, leading to a series of new methods and ideas, is the automated search of differential and linear characteristics in ARX ciphers. The core idea is to find a shortest path in a weighted directed acyclic graph. The approaches to solve the problem can be classified into

- Programs with advanced searching strategy, cf. [BV14, BVLC16, YZW15, AB16];
- Mixed integer linear programming, cf. [FWG+16];
- Constraint programming, including SAT (Boolean Satisfiability Problem) and SMT (Satisfiability Modulo Theories), cf. [KLT15, LWR16, DWAL17].

As a cryptanalytic method with wide applications on ARX primitives, rotational cryptanalysis was not evaluated on SPECK until a new method to deal with the constants was proposed in FSE 2017 [AL16]. Ashur and Liu presented a general method for integrating the XOR of round constants into the analysis by combining rotational with differential cryptanalyses. They used SPECK32/64 to exemplify their approach, but did not aim to extend existing attacks. Since the round constants in SPECK are injected through the key schedule, finding an RX characteristic for the key schedule suggests the existence of a weak-key class following the proposed RX-characteristic. Once a key from this class is chosen, a set of RX-differences is injected into the state, which can be used to trace the evolution of an RX-characteristic through the cipher. To test their theory, they presented a limited application of the technique by constructing a distinguisher for a small number of rounds in SPECK32/64.

Our contributions. This paper extends [AL16] by using an automated tool to systematically search for good RX characteristics in SPECK. We present extended characteristics for SPECK 32/64 and SPECK48/96 which are, to the best of our knowledge, the longest characteristics for these versions of SPECK. For SPECK96/144 we present a characteristic matching the length of the already published differential characteristic, but with a much lower data complexity. In some of the cases, the size of the weak-key class may seem small at first. However, we stress that the search strategy we employed favored reducing the data complexity over increasing the size of the weak-key class and therefore, other trade-offs between the data complexity and the weak-key class are possible.

Organization. The rest of the paper is organized as follows: We recall notations and the theory of Rotational-XOR cryptanalysis in section 2. In section 3, the automatic search of RX-characteristics is formulated, and the strategy of the search for optimal solutions is discussed. We present the characteristics found for different versions of SPECK in section 4. section 5 concludes the paper.

2 Preliminaries

2.1 Notations

We denote an n-bit vector by $x = (x_{n-1}, \ldots, x_1, x_0) \in \mathbb{F}_{2^n}$, and the Hamming weight of x is denoted by $|x|$. The bits $(x_{n-1}, \ldots, x_1)$ of x are denoted by $L(x)$. A left (resp. right) circular rotation by the amount γ is $x \lll \gamma$ (resp. $x \ggg \gamma$). A left shift by 1 is denoted by SHL, and $(I \oplus SHL)(x) = x \oplus SHL(x)$. $1_{x \preceq y}$ is the characteristic function which evaluates to 1 when $\forall i : x_i \leq y_i, 0 \leq i < n$, otherwise to 0.

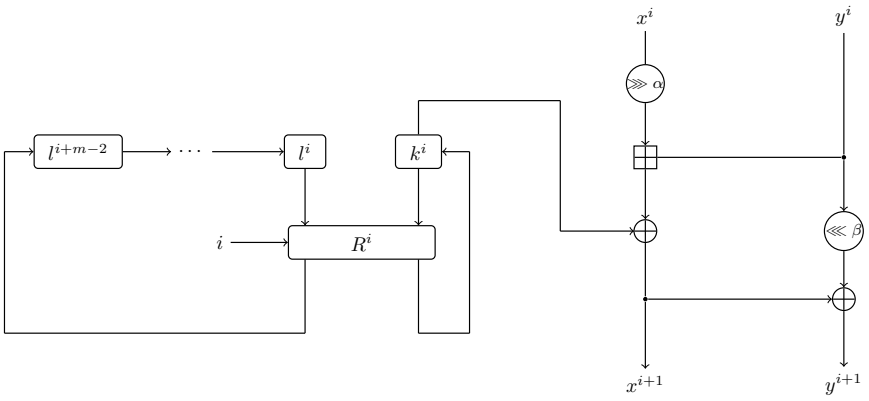

Figure 1: One round of SPECK

2.2 A Brief Description of SPECK[1]

SPECK is a family of lightweight block ciphers designed by the NSA in 2013 [BSS+15]. A member of the family is denoted by SPECK$2n/mn$, where the block size is $2n$ for $n \in \{16, 24, 32, 48, 64\}$, and the key size is mn for $m \in \{2, 3, 4\}$, depending on the desired security.

The round function of SPECK receives two words x^i and y^i, and a round key k^i, all of size n, and outputs two words of size n, x^{i+1} and y^{i+1}, such that

$$(x^{i+1}, y^{i+1}) = F_{k^i}(x^i, y^i) = (f_{k^i}(x^i, y^i), f_{k^i}(x^i, y^i) \oplus (y^i \lll \beta)),$$

where $f_{k^i}(\cdot, \cdot)$ is

$$f_{k^i}(x^i, y^i) = ((x^i \ggg \alpha) \boxplus y^i) \oplus k^i.$$

The SPECK key schedules algorithm uses the same round function to generate the round keys. Let $K = (l^{m-2}, ..., l^0, k^0)$ be a master key for SPECK$2n$, where $l^i, k^0 \in \mathbb{F}_{2^n}$. The sequence of round keys k^i is generated as

$$k^{i+1} = f_{ct}(l^i, k^i) \oplus (k^i \lll \beta)$$

for

$$l^{i+m-1} = f_{ct}(l^i, k^i),$$

with $ct = i$ the round number starting from 0.

The rotation offset (α, β) is $(7, 2)$ for SPECK32, and $(8, 3)$ for the larger versions. A single round of SPECK with $m = 4$ is depicted in Figure 1. For more details, we refer the interested reader to the original design [BSS+15] and to the recently published design rationale [BSS+17].

In SAC 2014, Dinur [Din14] proposed attacks on all versions of SPECK, where dedicated key recovery techniques were combined with the best differential characteristics known by that time. Later, the attacks on SPECK with block size larger than 32 were further improved with the discovery of new differential distinguishers [BVLC16, FWG+16].

2.3 Rotational-XOR cryptanalysis

In [AL16], the notion of Rotational-XOR difference is proposed for Rotational-XOR cryptanalysis. It defines the relation between a pair of bit-strings $x_1 = (x \lll \gamma) \oplus$

[1]The description of SPECK is lifted from [AL16] as is allowed by the license under which ToSC is published.

a_1 and $x_2 = x \oplus a_2$. We use a slightly different notation in the sequel with x and $x' = ((x \oplus a_1) \ggg \gamma) \oplus a_2$.

Definition 1. A Rotational-XOR difference (or RX-difference in short) with rotational offset γ of two bit-strings x and x' is defined as

$$\Delta_\gamma(x, x') = x \oplus (x' \lll \gamma).$$

Since the rotation and XOR are linear operations, the propagation of an RX-difference is similar to that of an XOR-difference through the linear operations of an ARX primitive. For the modular addition, the propagation of RX-differences is non-deterministic and characterised into the following proposition.

Proposition 1 ([AL16])**.** *Suppose that $x, y \in \mathbb{F}_{2^n}$ are independent uniform random variables, $z = x \boxplus y$. Let $\Gamma_x = \Delta_1(x, x')$, $\Gamma_y = \Delta_1(y, y')$ and $\Gamma_z = \Delta_1(z, z')$ be constants in $\mathbb{F}_{2^n}$, which are the RX-differences. Then,*

$$\begin{aligned}\Pr[((x \oplus \Gamma_x) \ggg 1) \boxplus ((y \oplus \Gamma_y) \ggg 1) = (z \oplus \Gamma_z) \ggg 1] \\ = 1_{(I \oplus SHL)(\delta_x \oplus \delta_y \oplus \delta_z) \oplus 1 \preceq SHL((\delta_x \oplus \delta_z)|(\delta_y \oplus \delta_z))} \cdot 2^{-|SHL((\delta_x \oplus \delta_z)|(\delta_y \oplus \delta_z))|} \cdot 2^{-3} \\ + 1_{(I \oplus SHL)(\delta_x \oplus \delta_y \oplus \delta_z) \preceq SHL((\delta_x \oplus \delta_z)|(\delta_y \oplus \delta_z))} \cdot 2^{-|SHL((\delta_x \oplus \delta_z)|(\delta_y \oplus \delta_z))|} \cdot 2^{-1.415}, \quad (1)\end{aligned}$$

where

$$\delta_x = L(\Gamma_x), \delta_y = L(\Gamma_y), \delta_z = L(\Gamma_z).$$

In words: the probability that the input RX-differences Γ_x and Γ_y propagate to the output RX-difference Γ_z through modular addition is given by Proposition 1. In the rest of this paper we only consider RX-differences with $\gamma = 1$. Note that when the constants $\Gamma_X = \Gamma_y = \Gamma_z = 0$, Proposition 1 predicts the case for normal rotational cryptanalysis with rotation amount 1, *i.e.*, $Pr[(x \ggg 1) \boxplus (y \ggg 1) = z \ggg 1] = 2^{-2.145}$.

2.4 The Boolean Satisfiability Problem

A *boolean formula* is an expression consisting of boolean variables taking the values TRUE or FALSE, and the logic operators AND, OR and NOT. A boolean formula is *satisfiable* if there exists an assignment of the variables that makes the formula TRUE. For example the boolean formula a AND (NOT b) is satisfiable since the assignment $(a, b) =$ (TRUE, FALSE) evaluates the entire formula to TRUE.

The boolean satisfiability (SAT) problem is the problem of determining whether a boolean formula is satisfiable. In general, the SAT problem is NP-complete [Coo71], which implies that no known algorithm solves SAT in polynomial time (with respect to the number of variables). In practice, SAT solvers can handle instances with thousands (and sometimes even millions) of variables [ZM02].

A generalization of the SAT problem is the satisfiability modulo theories (SMT) problem. Basically, SMT formulas can be expressed with richer languages (theories) than boolean formulas. In particular, a formula in the bit-vector theory can contain bit-vectors (a vector of boolean variables) and the usual operations of bit-vectors such as bitwise operations (XOR, OR, AND, etc.) arithmetic operations (addition, multiplication, etc.), cyclic operations and so on. A common approach in SMT solvers [GD07, BB09] is to translate the SMT instance into a SAT instance and solve it using a SAT solver.

In addition to richer langauges, SMT solvers also support an *objective function*. This function is an additional constraint forcing a variable to satisfy certain conditions. For example, through an objective function, an adversary can ask the solver for solutions not exceeding some probability for the RX-characteristics.

2.5 Attack Models

As an extension of rotational cryptanalysis, RX-cryptanalysis works in the related-key chosen-plaintext model. In this model an adversary can obtain data encrypted under two different keys with a known relation, for plaintexts selected by the adversary.

Speck presentes a unique challenge to RX-cryptanalysis due to its non-linear key schedule. Whereas in a linear key-schedule, the propogation of RX-characteristics can be predicted with probability 1, only probabalistic predictions can be made for a non-linear key-schedule. When we model the key schedule in section 3, Proposition 1 is used to predict the propogation probability, which may lead to non-integer values for the size of the weak-key class.

In addition, some of the distinguishers presented in Table 1 require more data than what is allowed by the weak-key class (An attack using a weak-key class of size $|K|$ cannot have time complexity larger than $|K|$). These results are marked with †† in the table and can only be used in the open-key model, *i.e.*, in addition to being in the weak-key class and knowing the relation between the two related-keys, the adversary also knows the key values.

This constraint is not required for entries in the table where the number of required plaintext pairs is smaller than the number of weak-keys, and such attacks can be executed in the closed-key model.

3 Automated Search for RX-characteristics

Previous work concerning Speck modeled differential and linear cryptanalysis as *SAT/SMT* or *MILP* problems. We continue this line of research by writing the problem of finding good RX-characteristics using the SMTLIB [BFT16] language, then converting it into a SAT problem using STP [GD07] and solving it using the same tool.

We now explain our model using the notation of Figure 2.

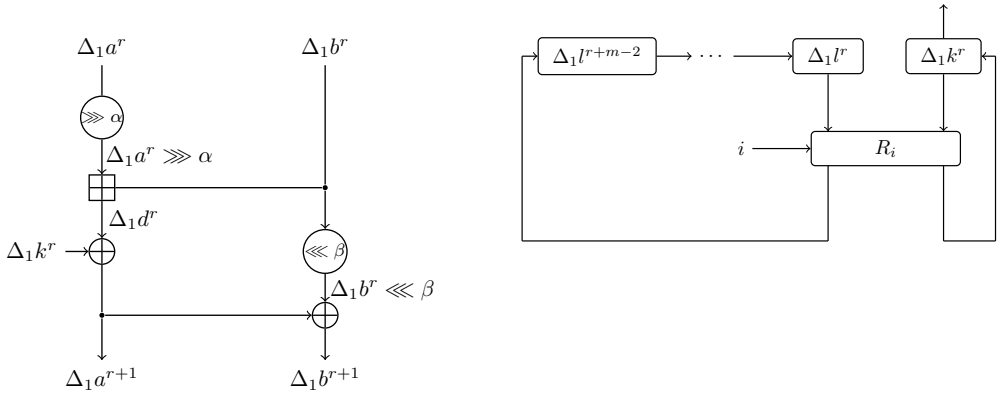

Figure 2: Notation of the RX-differences in Speck. Left: Round function. Right: Key schedule

Since the key schedule of SPECK reuses the same round function as the cipher itself, it is sufficient to only model the round function. The most difficult part of the model is the modular addition which is non-linear. We use two mutually exclusive constraints:

$$(I \oplus SHL)((\Delta_1 a^r \ggg \alpha) \oplus \Delta_1 b^r \oplus \Delta_1 d^r) \oplus 1 \preceq SHL(((\Delta_1 a^r \ggg \alpha) \oplus \Delta_1 d^r)|(\Delta_1 b^r \oplus \Delta_1 d^r)) \quad (2)$$

or

$$(I \oplus SHL)((\Delta_1 a^r \ggg \alpha) \oplus \Delta_1 b^r \oplus \Delta_1 d^r) \quad \preceq SHL(((\Delta_1 a^r \ggg \alpha) \oplus \Delta_1 d^r)|(\Delta_1 b^r \oplus \Delta_1 d^r)) \quad (3)$$

The cost w_r is calculated as

$$w_r = \begin{cases} |SHL(((\Delta_1 a^r \ggg \alpha) \oplus \Delta_1 d^r)|(\Delta_1 b^r \oplus \Delta_1 d^r))| + 3, \text{when Constraint (2) holds} \\ |SHL(((\Delta_1 a^r \ggg \alpha) \oplus \Delta_1 d^r)|(\Delta_1 b^r \oplus \Delta_1 d^r))| + 1.415, \text{when Constraint (3) holds.} \end{cases}$$

Then, the linear operations are modeled as follows:

$$\Delta_1 a^{r+1} = \Delta_1 d^r \oplus \Delta_1 k^r,$$
$$\Delta_1 b^{r+1} = (\Delta_1 b^r \lll \beta) \oplus \Delta_1 a^{r+1}.$$

Our objective function is defined as

$$\sum_r w^r \leq p.$$

Starting from Figure 2 each operation is replaced with the appropriate constraint(s). This is repeated for each round of the round-reduced cipher, where the output constraints of a round are treated as the input constraints of the next one. A target value is set for the objective function and the program is given as input to the STP tool [GD07] which searches for a solution satisfying all constraints. When the STP tool finishes, the target value is replaced with a new one according to the search strategy described in subsection 3.1, and the STP tool is called again until the search is complete.

3.1 Search Strategy

We now describe our search method. For each version of SPECK, we model the propagation of RX-differences through both the round function and the key schedule. Since SPECK uses a non-linear key schedule, an RX-characteristic over the key schedule is akin to a weak-key class. The RX-difference of each subkey is injected into the state and affects the round's input RX-difference.

Our program works in two phases:

3.1.1 Phase 1 — finding a good RX-characteristic over the data part.

The program starts by searching for an RX-characteristic covering the data part of the cipher (*i.e.*, the left side of Figure 2) with probability larger than $2^{-n/2}$, and the key schedule part with probability at least 2^{-mn} for mn the length of the key (*i.e.*, ensuring that at least one weak-key exists). If a solution adhering to these constraints is found, the objective function for the data part is updated and an RX-characteristic with probability larger than $2^{-n/4}$ is sought.

If the program cannot find a solution with probability at least $2^{-n/2}$, the objective function for the data part is relaxed and the program searches for an RX-characteristic with probability at least $2^{-1.5n/2}$. This binary search (over the exponent for the data part) is repeated until no further improvements are possible.

Algorithm 1 Find an optimal RX-characteristic of r rounds for SPECK32/64.

Input: T_d^+,T_d^-,T_k^+,T_k^-.
Output: The probability of an optimal RX-characteristic of r rounds.

$T_d^+ \leftarrow 32$, $T_d^- \leftarrow 0$, $T_k^+ \leftarrow 64$, $T_k^- \leftarrow 0$
$T_d^- \leq W_d \leq T_d^+$, $T_k^- \leq W_k \leq T_k^+$
while $T_d^+ \neq T_d^-$ **do**
    if The problem is *satisfiable* **then**
        $T_d^+ \leftarrow T_d^+/2$
    else
        $T_d^- \leftarrow T_d^+/2$
    end if
    $T_d^- \leq W_d \leq T_d^+$
end while
while $T_k^+ \neq T_k^-$ **do**
    if The problem is *satisfiable* **then**
        $T_k^+ \leftarrow T_k^+/2$
    else
        $T_k^- \leftarrow T_k^+/2$
    end if
    $T_k^- \leq W_k \leq T_k^+$
end while
return $2^{-W_d}, 2^{-W_k}$

3.1.2 Phase 2 — increasing the size of the weak-key class.

After the best RX-characteristic (in terms of its probability) is found, the program sets to increase the size of the weak-key class. Suppose ζ_0 is the probability for the RX-characteristic found in Phase 1, the objective functions in Phase 2 are set such that the program finds RX-characteristics with probability at least ζ_0 for the data part, and probability at least $2^{-mn/2}$ for the key schedule (*i.e.*, the right part of Figure 2). In a binary search similar to that of Phase 1, the best RX-characteristic for the key schedule is improved, under the constraint that this RX-characteristic can support an RX-characteristic for the data part with probability at least ζ_0.

When the program can no longer improve the probability for the key's RX-characteristic, it outputs both RX-characteristics. Using this algorithm it is guaranteed that the data RX-characteristic have optimal probability, and that the corresponding key RX-characteristic allows for a non-empty weak key class. The Algorithm is more formally described in Algorithm 1.

3.2 Additional Search Strategies

Note that, for purposes of obtaining a large number of rounds, the above search strategy prefers RX-characteristics with high probability in the data part over large weak-key classes. Some readers may prefer different tradeoffs, which can be obtained by minor modifications to the code we provide in [Wit17].

In particular, the reviewers of this paper asked for examples where the size of the weak-key class is larger than the required data complexity. We have therefore ran several experiments with the additional constraint that $\zeta_0 \cdot \zeta_1 < 2^{-2n}$ where ζ_0 is as before, ζ_1 is the probability for finding a weak-key, and $2n$ is the block size.

Table 1: Comparison of RX-characteristics with $\gamma = 1$ and previous differentials for different versions of SPECK. Entries marked with † were found through the adjusted search strategy. Entries marked with †† can only be used in the open-key model.

Version	Rounds	Data Prob.	Key Class Size	Ref.
32/64	9	2^{-30}	2^{64}	[Din14]
32/64	10	$2^{-19.15}$	$2^{28.10}$	This paper
32/64	11††	$2^{-22.15}$	$2^{18.68}$	This paper
32/64	12††	$2^{-25.57}$	$2^{4.92}$	This paper
48/96	10	2^{-40}	2^{96}	[Din14]
48/96	11	2^{-45}	2^{96}	[FWG+16]
48/96	11	$2^{-23.15}$	$2^{14.93}$	This paper
48/96	11†	$2^{-24.15}$	$2^{25.68}$	This paper
48/96	12	$2^{-26.57}$	$2^{27.5}$	This paper
48/96	12†	$2^{-26.57}$	$2^{43.51}$	This paper
48/96	13††	$2^{-31.98}$	$2^{24.51}$	This paper
48/96	14††	$2^{-37.40}$	$2^{0.34}$	This paper
48/96	15††	$2^{-43.81}$	$2^{1.09}$	This paper
64/128	14	2^{-60}	2^{128}	[Din14]
64/128	15	2^{-62}	2^{128}	[FWG+16]
64/128	13††	$2^{-37.98}$	$2^{21.92}$	This paper
96/144	13	2^{-84}	2^{144}	[Din14]
96/144	16	2^{-87}	2^{144}	[FWG+16]
96/144	13††	$2^{-37.98}$	$2^{37.92}$	This paper
128/256	14	2^{-112}	2^{256}	[Din14]
128/256	19	2^{-119}	2^{256}	[FWG+16]
128/256	13	$2^{-31.98}$	$2^{182.51}$	This paper

4 RX-characteristics found in SPECK

With the model discussed in section 3 and the search strategy described in subsection 3.1, we present an overview of the distinguishers found in Table 1.

4.1 RX-characteristics of SPECK32/64

Table 2 shows the RX-characteristic covering 11 and 12 rounds found by our program. The best published characteristic so far covered 9 rounds of SPECK with probability 2^{-30}. Our 10-round characteristic has a much better probability of $2^{-19.15}$ for a weak-key class of size $2^{28.10}$. The table also shows that even our 12-round characteristic has probability of $2^{-25.57}$ which is still higher than the previously known 9-round differential characteristic, although ours works for a weak-key class of about 30 keys.

We extended our search to 13-round characteristics and found that none exist, suggesting that a 12-round RX-characteristic is the longest possible one.

4.2 RX-characteristics of SPECK48/96

We found RX-characteristics covering up to 15 rounds for SPECK48/96, some of the characteristics are shown in Table 3 and Table 4. The distinguishers extend the previously best differential characteristic which covers 11 rounds with probability 2^{-45}. Note that the sizes of the weak key class for the 14- and 15-round characteristics are marginal. However, due to resources constraint we killed the program before it completed its search. Hence, the characteristics presented in this subsection are not guaranteed to be optimal in length

Table 2: A 11-round (left) and 12-round (right) RX-characteristic in Speck32/64.

Round	RX-difference in Key	RX-difference in Input
0	0000	(0000||0000)
1	0000	(0000||0000)
2	0000	(0000||0000)
3	0001	(0000||0000)
4	0000	(0000||0000)
5	0003	(0000||0000)
6	0200	(0000||0000)
7	0205	(0200||0200)
8	0801	(0000||0800)
9	2001	(0000||2000)
10	AA0B	(0000||8000)
11		(2A0B||2A09)
Prob.	$2^{-45.32}$	$2^{-22.15}$

Round	RX-difference in Key	RX-difference in Input
0	0000	(0050||2000)
1	0100	(8000||0000)
2	0001	(0000||0000)
3	0000	(0000||0000)
4	0001	(0000||0000)
5	0000	(0000||0000)
6	0001	(0000||0000)
7	0200	(0000||0000)
8	0206	(0200||0200)
9	0800	(0000||0800)
10	2001	(0000||2000)
11	A40E	(0000||8000)
12		(240E||240C)
Prob.	$2^{-59.08}$	$2^{-25.57}$

Table 3: 12-round (left) and 13-round (right) RX-characteristics in Speck48/96.

Round	RX-difference in Key	RX-difference in Input
0	000008	(000000||000008)
1	000240	(000000||000040)
2	000000	(000200||000000)
3	000000	(000000||000000)
4	000000	(000000||000000)
5	000000	(000000||000000)
6	000001	(000000||000000)
7	000001	(000000||000000)
8	000001	(000000||000000)
9	010010	(000001||000001)
10	100089	(000010||000018)
11	8904de	(000080||000040)
12		(09049e||09069e)
Prob.	$2^{-52.49}$	$2^{-26.57}$

Round	RX-difference in Key	RX-difference in Input
0	000008	(000000||000008)
1	000240	(000000||000040)
2	000000	(000200||000000)
3	000000	(000000||000000)
4	000000	(000000||000000)
5	000000	(000000||000000)
6	000001	(000000||000000)
7	000001	(000000||000000)
8	000000	(000000||000000)
9	010018	(000001||000001)
10	1000f1	(000018||000010)
11	880801	(080080||080000)
12	c04911	(000000||400000)
13		(004911||004913)
Prob.	$2^{-71.49}$	$2^{-31.98}$

(*i.e.*, 16-round RX-characteristics may exist) nor in probability (*i.e.*, RX-characteristics with higher probabilities or a larger weak-key class may exist for the same number of rounds). In addition, the probabilities of the round function part in the 14- and 15-round characteristics are relatively high, which suggests that distinguishers with larger weak key classes can be found for different trade-offs.

4.3 RX-characteristics of SPECK96/144

A 13-round RX-characteristic is found for Speck96/144 as shown in Table 5.

Table 4: 14-round (left) and 15-round (right) RX-characteristics in Speck48/96.

Round	RX-difference in Key	RX-difference in Input
0	000008	(000000||000008)
1	000240	(000000||000040)
2	000000	(000200||000000)
3	000000	(000000||000000)
4	000000	(000000||000000)
5	000000	(000000||000000)
6	000001	(000000||000000)
7	000001	(000000||000000)
8	000000	(000000||000000)
9	010018	(000000||000000)
10	1000e0	(010019||010019)
11	680021	(0801e8||000120)
12	000009	(000900||000000)
13	202844	(000000||000000)
14		(202844||202844)
Prob.	$2^{-95.66}$	$2^{-37.40}$

Round	RX-difference in Key	RX-difference in Input
0	000008	(000000||000008)
1	000240	(000000||000040)
2	000000	(000200||000000)
3	000000	(000000||000000)
4	000000	(000000||000000)
5	000000	(000000||000000)
6	000001	(000000||000000)
7	000001	(000000||000000)
8	000001	(000000||000000)
9	010011	(000001||000001)
10	100080	(000010||000018)
11	990391	(000089||000049)
12	480103	(000248||000000)
13	000301	(000100||000100)
14	91101d	(000000||000800)
15		(91181d||91581d)
Prob.	$2^{-94.91}$	$2^{-43.81}$

Table 5: A 13-round RX characteristic for Speck96/144.

Round	RX-difference in Key	RX-difference in Input
0	000000020801	(000002080000||000000000001)
1	000000000008	(000000000000||000000000008)
2	000000000240	(000000000000||000000000040)
3	000000000000	(000000000200||000000000000)
4	000000000000	(000000000000||000000000000)
5	000000000000	(000000000000||000000000000)
6	000000000001	(000000000000||000000000000)
7	000000000010	(000000000000||000000000000)
8	07000000001E	(000000000003||000000000003)
9	390000000001	(000000000018||000000000000)
10	090100000010	(010000000000||010000000000)
11	100800000091	(080000000010||000000000010)
12	767707000425	(000000000080||000000000000)
13		(F67707000425||F67707000425)
Prob.	$2^{-106.08}$	$2^{-37.98}$

4.4 Experimental Verification

The characteristics above were partially verified empirically. For 10-round and 11-round characteristics we generated keys uniformly and their respective RX-related-keys. For each of those, we then executed the key expansion algorithm and tested whether the key characteristic is followed. Once a weak key was found, we encrypted 2^{32} plaintexts, and measured the probability that the RX characteristic is satisfied. For the larger versions, we injected key differences artificially and only tested the probability of the RX characteristics over the cipher part. The results matched the theoretical predictions.

5 Conclusion

In this paper we presented for the first time a SAT/SMT model for RX-cryptanalysis of ARX primitives. We tested the model on various versions of Speck and obtained longer distinguishers than previously published. For Speck32/64 we presented distinguishers for 10, 11 and 12 rounds with respective probabilities of $2^{-19.15}$, $2^{-22.15}$, and $2^{-25.57}$ working for weak-key classes of size $2^{28.10}$, $2^{18.68}$, and $2^{4.92}$, respectively. For versions of Speck48, we presented several distinguishers, the longest of which works for 15 rounds with probability $2^{-43.81}$ and it works for weak key class of size $2^{1.09}$.

Further work may search for longer distinguishers on all versions of Speck except Speck32/64. In addition, different tradeoffs can still be found for all versions by setting differently the objective functions for the data complexity and the size of the weak-key class. The SAT/SMT model we developed can readily be used for other ARX constructions, possibly with a linear key schedule which eliminates the need to consider weak-key classes.

Acknowledgements

The authors would like to extend their thanks to Vincent Rijmen and to the anonymous reviewers for their useful comments.

This work was partially supported by the Research Council KU Leuven, OT/13/071. Yunwen Liu is partially supported by China Scholarship Council (CSC 201403170380) and National Natural Science Foundation (No. 61672530). Adrián Ranea was supported by the EU's Erasmus students exchange program. Tomer Ashur is partially supported by BELSPO under contract BR/132/A4/BCC.

References

[AB16] Tomer Ashur and Daniël Bodden. Linear cryptanalysis of reduced-round SPECK. In *Proceedings of the 37th Symposium on Information Theory in the Benelux*, 2016.

[AL16] Tomer Ashur and Yunwen Liu. Rotational cryptanalysis in the presence of constants. *IACR Transactions on Symmetric Cryptology*, 2016(1):57–70, 2016.

[ANWOW13] Jean-Philippe Aumasson, Samuel Neves, Zooko Wilcox-O'Hearn, and Christian Winnerlein. Blake2: simpler, smaller, fast as MD5. In *International Conference on Applied Cryptography and Network Security - ACNS 2013*, pages 119–135. Springer, 2013.

[BB09] Robert Brummayer and Armin Biere. Boolector: An Efficient SMT Solver for Bit-Vectors and Arrays. In *Tools and Algorithms for the Construction and Analysis of Systems*. Springer, Berlin, Heidelberg, 2009.

[BDPVA13] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Keccak. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques - EUROCRYPT 2013*, pages 313–314. Springer, 2013.

[BFT16] Clark Barrett, Pascal Fontaine, and Cesare Tinelli. The Satisfiability Modulo Theories Library SMT-LIB. `www.SMT-LIB.org`, 2016.

[BSS+15] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. The SIMON and SPECK lightweight block ciphers. In *Proceedings of the 52nd Annual Design Automation Conference - DAC 2015*, pages 175:1–175:6. ACM, 2015.

[BSS+17] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. Notes on the design and analysis of SIMON and SPECK. *IACR Cryptology ePrint Archive*, 2017:560, 2017.

[BV14] Alex Biryukov and Vesselin Velichkov. Automatic search for differential trails in ARX ciphers. In *Topics in Cryptology - CT-RSA 2014*, pages 227–250. Springer, 2014.

[BVLC16] Alex Biryukov, Vesselin Velichkov, and Yann Le Corre. Automatic search for the best trails in ARX: Application to block cipher SPECK. In *International Conference on Fast Software Encryption - FSE 2016*, pages 289–310. Springer, 2016.

[Coo71] Stephen A. Cook. The Complexity of Theorem-proving Procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, New York, NY, USA, 1971. ACM.

[Din14] Itai Dinur. Improved differential cryptanalysis of round-reduced SPECK. In *International Workshop on Selected Areas in Cryptography - SAC 2014*, pages 147–164. Springer, 2014.

[DWAL17] Glenn De Witte, Tomer Ashur, and Yunwen Liu. An automated tool for Rotational-XOR cryptanalysis of ARX-based primitives. In *38th Symposium on Information Theory in the Benelux*, 2017.

[FLS+10] Niels Ferguson, Stefan Lucks, Bruce Schneier, Doug Whiting, Mihir Bellare, Tadayoshi Kohno, Jon Callas, and Jesse Walker. The skein hash function family. *Submission to NIST (round 3)*, 7(7.5):3, 2010.

[FWG+16] Kai Fu, Meiqin Wang, Yinghua Guo, Siwei Sun, and Lei Hu. MILP-based automatic search algorithms for differential and linear trails for SPECK. In *International Conference on Fast Software Encryption - FSE 2016*, pages 268–288. Springer, 2016.

[GD07] Vijay Ganesh and David L. Dill. A decision procedure for bit-vectors and arrays. In Werner Damm and Holger Hermanns, editors, *Computer Aided Verification, 19th International Conference, CAV 2007, Berlin, Germany, July 3-7, 2007, Proceedings*, volume 4590 of *Lecture Notes in Computer Science*, pages 519–531. Springer, 2007.

[KLT15] Stefan Kölbl, Gregor Leander, and Tyge Tiessen. Observations on the SIMON block cipher family. In *Annual Cryptology Conference - CRYPTO 2015*, pages 161–185. Springer, 2015.

[KN10] Dmitry Khovratovich and Ivica Nikolić. Rotational cryptanalysis of ARX. In *International Conference on Fast Software Encryption - FSE 2010*, pages 333–346. Springer, 2010.

[KNP+15] Dmitry Khovratovich, Ivica Nikolić, Josef Pieprzyk, Przemysław Sokołowski, and Ron Steinfeld. Rotational cryptanalysis of ARX revisited. In *International Conference on Fast Software Encryption - FSE 2015*, pages 519–536. Springer, 2015.

[LWR16] Yunwen Liu, Qingju Wang, and Vincent Rijmen. Automatic search of linear trails in ARX with applications to SPECK and chaskey. In *International Conference on Applied Cryptography and Network Security - ACNS 2016*, pages 485–499. Springer, 2016.

[Wit17] Glenn De Witte. Auxiliary package for this paper. http://homes.esat.kuleuven.be/~tashur/FSE2018/Rotational-XOR_Crpytanalysis_Speck.html, 2017. [Online; accessed 11-August-2017].

[YZW15] Yuan Yao, Bin Zhang, and Wenling Wu. Automatic search for linear trails of the SPECK family. In *International Information Security Conference - ISC 2015*, pages 158–176. Springer, 2015.

[ZM02] Lintao Zhang and Sharad Malik. The Quest for Efficient Boolean Satisfiability Solvers. In *Automated Deduction—CADE-18*. Springer, Berlin, Heidelberg, 2002.