cryptography

MDPI

Article

Partly-Pseudo-Linear Cryptanalysis of Reduced-Round SPECK

Sarah A. Alzakari * and Poorvi L. Vora

Department of Computer Science, The George Washington University, 800 22nd St. NW, Washington, DC 20052, USA; poorvi@gwu.edu
* Correspondence: salzakari@gwu.edu

Abstract: We apply McKay's pseudo-linear approximation of addition modular 2^n to lightweight ARX block ciphers with large words, specifically the SPECK family. We demonstrate that a pseudo-linear approximation can be combined with a linear approximation using the meet-in-the-middle attack technique to recover several key bits. Thus we illustrate improvements to SPECK linear distinguishers based solely on Cho–Pieprzyk approximations by combining them with pseudo-linear approximations, and propose key recovery attacks.

Keywords: SPECK; pseudo-linear cryptanalysis; linear cryptanalysis; partly-pseudo-linear attack

1. Introduction

ARX block ciphers—which rely on Addition-Rotation-XOR operations performed a number of times—provide a common approach to lightweight cipher design. In June 2013, a group of inventors from the US's National Security Agency (NSA) proposed two families of lightweight block ciphers, SIMON and SPECK—each of which comes in a variety of widths and key sizes. The SPECK cipher, as an ARX cipher, provides efficient software implementations, while SIMON provides efficient hardware implementations. Moreover, both families perform well in both hardware and software and offer the flexibility across different platforms that will be required by future applications [1,2]. In this paper, we focus on the SPECK family as an example lightweight ARX block cipher to illustrate our attack.

Pseudo-linear cryptanalysis [3–5] is a method of analyzing and measuring the security of an ARX block cipher. The main idea of the pseudo-linear approximation is to examine a window (group of contiguous bits) of size w, for some $w < n$, and approximate addition modulo 2^n by addition modulo 2^w. If the carry into the window is estimated correctly, the approximation will be perfect. The probability of correctness for a random guess of the value of the window is $\frac{1}{2^w}$, but the accuracy of the pseudo-linear approximation can be much larger.

This paper presents a new approximation and corresponding key recovery attack, Partly-Pseudo-Linear attack, combining pseudo-linear approximation with linear cryptanalysis of addition modulo 2^n using Cho and Pieprzyk's property of modular addition [6,7]. This combination of linear and pseudo-linear attack is original to the best of our knowledge. We illustrate, on SPECK, improvements due to this approximation over the Cho–Pieprzyk approximation for all rounds. We further use our approximation to describe key recovery attacks. Additionally, for SPECK 32/64, we are able to provide experimental results of a few implemented six-round attacks verifying our proposal. We have demonstrated a similar approach to cryptanalysing the SPARX cipher in a later paper [8].

We compare our attack to [9,10] which present linear distinguishers using the Cho–Pieprzyk property. Our key recovery attacks are able to either cover more rounds with similar or better bias, or, when we cover same rounds, our bias is better. We are able to attack nine rounds for SPECK 32/64, 11 rounds for SPECK 48/96, 14 rounds for SPECK 64/128, 12 rounds for SPECK 96/144 and 14 rounds for SPECK 128/256 (see Section 3 for more detailed comparisons). Note that our approximation is itself a key recovery attack

Citation: Alzakari, S.; Vora, P. Partly-Pseudo-Linear Cryptanalysis of Reduced-Round SPECK. *Cryptography* **2021**, *5*, 1. https://doi.org/10.3390/cryptography5010001

Received: 2 July 2020
Accepted: 2 December 2020
Published: 30 December 2020

for more than a single bit of key, because its use requires nonlinear operations with some key bits. Though they do not discuss this, the linear distinguishers of [9,10] could possibly be extended to key recovery attacks, bias permitting, by appending rounds of encryption and/or decryption. In this instance, our attack covers more rounds for the variants SPECK 64/96 and 64/128, and covers same rounds for SPECK 48/72, determining a few more key bits. For SPECK 96/96 and 128/128, the data complexity of the attacks of [9,10] is large enough that they cannot add encryption/decryption rounds and hence can perform no key recovery, while we are able to determine 46 (of 96) and 50 (of 128) key bits respectively. While [9,10] do not report key recovery, they would need to reduce their covered rounds in order to determine key bits. In other instances too, when we are not able to cover same rounds, we often find many more key bits.

We find that our proposed Partly-Pseudo-Linear attack, a combination of Cho–Pieprzyk linear approximations with pseudo-linear approximations, is not necessarily more powerful than attacks that use other approaches to linear trails, such as Wallen's approach [11–13]. This seems reasonable, as the correct comparison with these would be a combination of their linear trails with pseudo-linear approximations, which is out of the scope of this paper.

This paper is organized as follows. In Section 2, we present a brief description of SPECK and the notation used in this paper. In Section 3 we focus on the most relevant related work. In Section 4, we present our first contribution by applying the pseudo-linear attack on SPECK—specifically, SPECK 32. In Section 5, we present our proposed Partly-Pseudo-Linear attack on SPECK and the results of the implementation. We conclude in Section 6.

2. Preliminaries

This section presents our notation and briefly describes the SPECK cipher.

2.1. Notation

The following describes the notation used in this paper.

- $\boxplus_n$: Addition modulo 2^n
- $\boxminus_n$: Subtraction modulo 2^n
- $\oplus$: The bitwise exclusive-or
- $\ggg_r$: r-bit right rotation on an n-bit word
- $\lll_r$: r-bit left rotation on an n-bit word
- $PL(CL)$: Left word of the Plaintext (Ciphertext)
- $PR(CR)$: Right word of the Plaintext (Ciphertext)
- $xl^j(xr^j)$: Left (right) word at round j
- $xl_t^j(xr_t^j)$: tth window of state xl (xr) at round j
- $xl_t^j(i, i+w)$ ($xr_t^j(i, i+w)$): window t with size w of the left (right) word x, where the msb is at i and the lsb is at $i+w-1$, for $0 \leq i < \frac{n}{2}$ and $1 \leq w \leq \frac{n}{2}$.
- $xl_t^j(i)$ ($xr_t^j(i)$): Bit at index i of the window where $0 \leq i < w$ the left (right) word x.

2.2. The SPECK Cipher

The SPECK cipher is a family of lightweight block ciphers, proposed by inventors from the National Security Agency (NSA) in June 2013 [1,2]. A member of the family is denoted by SPECK $2n/mn$, where the block size is $2n$ and the key size is mn for some $m \in \{2, 3, 4\}$. Each round function in SPECK has three main operations:

- Addition modulo 2^n, denoted $\boxplus_n$
- Rotation: right rotation by α, denoted $\ggg \alpha$ and left rotation by β, denoted $\lll \beta$
- bitwise XOR, denoted $\oplus$

In this construction the block of the plaintext is split into two words, PL and PR, which are then added, XORed and rotated by the round function. Figure 1 shows one round from SPECK; where xl^j (xr^j) denotes to the left (right) input words of round j and k^j denotes to the key of round j.

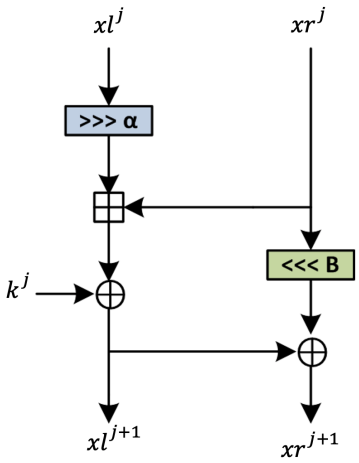

Figure 1. SPECK round function (j denotes to the number of round).

The output of the round function for the left word of SPECK is:

$$xl^{j+1} = ((xl^j \ggg \alpha) \boxplus_n xr^j) \oplus k^j. \quad (1)$$

The output of the round function for the right word is:

$$xr_R^{j+1} = (xr^j \lll \beta) \oplus xl^{j+1}. \quad (2)$$

The parameters specifying the SPECK versions are listed in Table 1.

Table 1. The SPECK cipher family.

Block Size, 2n	Key Size, mn	Word Size, n	α	β	Rounds
32	64	16	7	2	22
48	72	24	8	3	22
	96				23
64	96	32	8	3	26
	128				27
96	96	48	8	3	28
	144				29
128	128	64	8	3	32
	192				33
	256				34

3. Related Works

In this section, we review previous works that are relevant to our contributions. We first review linear cryptanalysis and pseudo-linear cryptanalysis, as we will combine

the two approaches for our attack. We then describe cryptanalysis of SPECK, as we will illustrate our attack on the SPECK family.

3.1. Linear Cryptanalysis

Linear cryptanalysis [14] is one of the most powerful and widely used attacks on block ciphers. It was introduced by Matsui in 1998, and is a known plaintext attack where the attacker has access to both the plaintext and its encrypted version ciphertext [14,15]. Using linear cryptanalysis, an adversary is able to find a linear expression that approximates a non-linear function that connects plaintext, ciphertext, and key bits with high probability.

The quality of the linear approximation is measured by the bias ϵ which is defined as $\epsilon = p - \frac{1}{2}$; a higher bias in absolute value, $|\epsilon|$, implies a better approximation and a more efficient attack. The number of required known plaintext and ciphertext pairs (data complexity, pairs) depends on the success probability desired and is roughly proportional to ϵ^{-2}. For example, pairs $= 2^4 \times \epsilon^{-2}$ corresponds to a 99.80% success rate. Table 2 shows different small multiple of ϵ^{-2} with their success rate of the linear approximation [14,15].

Table 2. The success rate.

Pairs	$\mathbf{2 \times \epsilon^{-2}}$	$\mathbf{2^2 \times \epsilon^{-2}}$	$\mathbf{2^3 \times \epsilon^{-2}}$	$\mathbf{2^4 \times \epsilon^{-2}}$
Rate	48.6%	78.5%	96.7%	99.9%

The Piling Up Lemma [14] provides an expression for the bias of an approximations that results from the xor of s approximations, each with bias ϵ_i:

$$\epsilon = 2^{s-1} \prod_{i=1}^{s} \epsilon_i \tag{3}$$

Linear Cryptanalysis of Modular Addition

The modular addition operation is nonlinear as an operation in $\mathbb{Z}_2$. The result of modular addition in a certain position is the exclusive-or (addition in $\mathbb{Z}_2$) of the two bits in that position and the carry into the position. The carry, in turn, depends on a non-linear operation (the and operation, multiplication over $\mathbb{Z}_2$) of previous bits.

Cho and Pieprzyk [6] describe in their paper the behavior of neighboring bits in modular additions. Consider $c = a \boxplus b$ where $a, b \in \{0,1\}^{32}$ and $\boxplus$ corresponds to addition modulo 2^{32}. Let $a = (a_{n-1}, \ldots, a_0)$, $b = (b_{n-1}, \ldots, b_0)$ and $c = (c_{n-1}, \ldots, c_0)$.

Lemma 1. *(Practically verbatim from Cho and Pieprzyk [6]) Let c_i be the ith output bit of the modular addition. Then, $c_0 = a_0 \oplus b_0$, $c_1 = a_1 \oplus b_1 \oplus a_0 \times b_0$ and for $2 \leq i \leq n-1$:*

$$c_i = a_i \oplus b_i \oplus a_{i-1} \times b_{i-1} \oplus \sum_{t=0}^{i-2} a_t \times b_t \times \prod_{r=t+1}^{i-1} (a_r \oplus b_r)'' \tag{4}$$

According to Cho and Pieperzyk [7], if $Cr(a,b)$ denotes the carry of modular addition, from Lemma 2:

$$Cr_i(a,b) = a_i \times b_i \oplus \sum_{t=0}^{i-1} a_t \times b_t \times \prod_{r=t+1}^{i} (a_r \oplus b_r), \quad i = 0, \ldots, n-2 \tag{5}$$

Then, obviously, $c_i = a_i \oplus b_i \oplus Cr_{i-1}(a,b)$ for $i = 1, \ldots, 31$. Due to Equation (5), the carry $Cr_i(a,b)$ has the following recursive relation [7].

$$Cr_i(a,b) = a_i \times b_i \oplus (a_i \oplus b_i) \times Cr_{i-1}(a,b) \tag{6}$$

All these equations with a_i, b_i and c_i represent one bit each. In another paper, Cho and Pieprzyk [6] describe a property of modular addition that removes the carry chain from

Equation (4) and this property uses consecutive bits. Two consecutive bits can be approximated as:

$$c_i \oplus c_{i+1} = a_i \oplus b_i \oplus a_{i+1} \oplus b_{i+1}, \quad with\ probability\ \ P = \frac{3}{4}. \tag{7}$$

This means:

$$Pr[Cr_i(a,b) \oplus Cr_{i-1}(a,b) = 0] = \frac{3}{4} \tag{8}$$

Removing the carry chain from Equation (4), using a mask λ to mask the bits that we want to throw away and keep the bits that we are interested in, we can write:

$$P[\lambda \times (a \boxplus b) = \lambda \times (a \oplus b)] = \frac{3}{4} \tag{9}$$

The mask λ contains exactly the two consecutive bits we are interested in and we can replace $\lambda \times (a \boxplus b)$ by $\lambda \times (a \oplus b)$ to obtain a linear expression. This approximation holds with a probability equal to $\frac{3}{4}$. Consequently, the bias is equal to $\frac{1}{4}$. In fact, a prerequisite for Equation (9) is that the following two cases are avoided, because these two cases do not adhere to the Cho and Pieprzyk framework [6]:

1. Bitwise rotation breaks the two consecutive bits. After the rotation, one of these two bits will be in the most significant bit position (msb) and the other will be in the least significant bit position (lsb).
   Example: $00011000 \ggg 4 = 10000001$
2. Bitwise exclusive-or breaks the two consecutive bits. These two bits will be not consecutive any more.
   Example: $00011000 \oplus 00110000 = 00101000$

3.2. Pseudo-Linear Cryptanalysis

McKay and Vora present the idea of pseudo-linear cryptanalysis [3–5] which aims to overcome the limitations of traditional linear cryptanalysis by approximating addition modulo 2^n for large values of n with addition modulo 2^w, for a small window size w, $0 < w \leq n$. In other words, the pseudo-linear approximations use addition modulo 2^w and exclusive-or over a w-bit strings of contiguous bits (windows) instead of using the entire n-bit strings. In this section we provide detail about the approach, which was first developed to analyze Threefish for the SHA-3 competition [5].

McKay and Vora [5] illustrate why this is an improvement over traditional linear cryptanalysis. Consider the following example:

In Figure 2, there are two n-bit words added modulo 2^n, only the value of the dark square, labeled z, is needed and it is of size w in bits. Denote by x and y the operand windows in the same position as z. Thus, z can be approximated as $x \boxplus_w y$. The correctness of this approximation is dependent on the value of the carry into the window z.

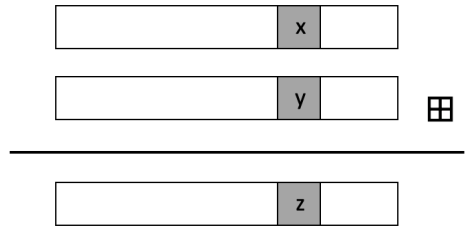

Figure 2. Addition of two words.

Let $part(x, s, e)$ represent bits of the word x in positions $[s, e)$, where s represents the index of the word that the window starts with and e is the size of this window, $0 \leq s < n$, $0 \leq e < n$, and the least significant bit (lsb) is at index s. We have two scenarios, illustrated by examples for $n = 12$. We have two strings added modulo 2^{12}, $z = x \boxplus y$. The adversary

wants to approximate only a window of 4 bits of z. Thus, $part(z,4,4) \approx part(x,4,4) \boxplus_2 part(y,4,4)$. Note that the approximation implicitly assumes that the carry into bit s is zero.

1. Suppose $x = 001001000100$ and $y = 100010101010$. In this case the approximation is correct because the carry into the window is correctly assumed to be zero (See Figure 3).

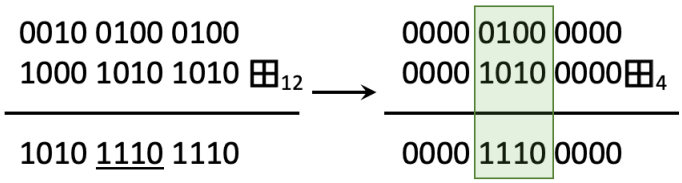

Figure 3. Correct approximation of a window of $z = x \boxplus y$.

2. Suppose $x = 00100100110$ and $y = 100010101010$. In this case the approximation is incorrect because the carry into the window is incorrectly assumed to be zero (See Figure 4).

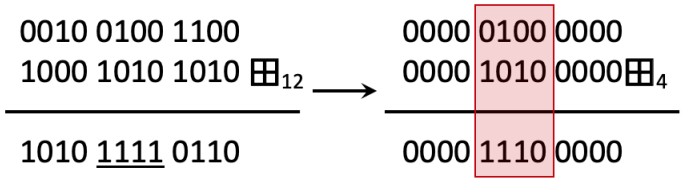

Figure 4. Incorrect approximation of a window of $z = x \boxplus y$.

The probability that the carry is 0 is exactly the probability that the approximation is correct when it is applied for the first time and both summand windows are correct (and not yet the result of approximations). This probability is equal to $\frac{1}{2} + 2^{-(s+1)}$ where s is the lsb of the window [4]. Note that the probability of correctly estimating an entire window is slightly larger than $\frac{1}{2}$. How does one measure the efficacy of this approximation?

Consider the approximation of a single bit, whether by linear approximation or any other technique. A guess made at random with no information would be correct with probability $\frac{1}{2}$. The bias of the approximation is defined as the deviation from $\frac{1}{2}$.

If $w > 1$, the pseudo-linear approach provides an approximation for multiple bits, and we define an error measure for the approximation as the difference between the probability of correctly approximating the (entire) window and $\frac{1}{2^w}$. Thus the pseudo-linear approximation is more advantageous if the size of the window is larger.

Note that the pseudo-linear approximation capture the influence of intermediate carries, which are not typically captured by linear approaches. This is expected to improve the result, even when the aim is to approximate the parity of the final window (see, for example, Section 5.1).

Additionally, intuitively, for a large window, a non-zero carry will not always affect the higher-order bits. Thus, if one is measuring the number of bits that are well-approximated by the pseudo-linear expression (in the previous paragraph, we considered only whether the entire w-bit window was correctly evaluated or not), the higher order bits are more likely to be correct.

Finally, because addition modulo 2^w and exclusive-or do not distribute, the composition of the pseudo-linear approximation and the key injection includes key bits combined in a non-linear manner. For this reason, the use of the pseudo-linear approximation for key recovery requires guessing multiple key bits. In spite of this, we are able to obtain attacks more efficiently than the brute force attack because pseudo-linear approximations enable the reduction of the number of key bits from those required by the cipher [5].

3.2.1. Some Observations Regarding the Addition Window

McKay and Vora [3] provide some properties of the approximation over addition windows. Consider two n-bit words, x and y, selected uniformly at random, and a window size $w < n$. The following notation is used in the lemmas: (quoted verbatim from [4])

- $\oplus$ Bitwise exclusive-or
- $\boxplus$ Addition modulo 2^n
- $\boxplus_w$ Addition modulo 2^w
- $\boxminus$ Subtraction modulo 2^n
- $\boxminus_w$ Subtraction modulo 2^w

Lemma 2. *"Let* $0 \leq s < s + w < n$. *Then* $Pr[\, part\,(x \boxplus y, s, s+w) = part\,(x, s, s+w) \boxplus_w part\,(y, s, s+w)] > \frac{1}{2}$*".*

In the proof of this lemma, McKay and Vora demonstrate that

$$Pr[\, part\,(x \boxplus y, s, s+w) = part\,(x, s, s+w) \boxplus_w part\,(y, s, s+w)] = \frac{1}{2} + 2^{-(s+1)}.$$

Note that this is the probability of the entire window being correctly approximated. The probability of bit parities being correctly approximated will typically be larger.

Lemma 3. $Pr[\, part\,(x \boxplus y, 0, w) = part\,(x, 0, w) \boxplus_w part\,(y, 0, w)] = 1.$

Lemma 4. $Pr[\, part(x \oplus y, s, (s+w) \bmod n) = part(x, s, (s+w) \bmod n) \oplus part(y, s, (s+w) \bmod n)] = 1.$

Corollary 1. *Let* $0 \leq s < n$ *and* $(s+w) \bmod n < s$. $Pr[\, part(x \boxplus y, s, (s+w) \bmod n) = part(x, s, (s+w) \bmod n) \boxplus_w part(y, s, (s+w) \bmod n)] > \frac{1}{2}$.

Corollary 2. $Pr[\, part(x \boxminus y, s, (s+w) \bmod n) = part\,(x, s, (s+w) \bmod n) \boxminus_w part\,(y, s, {}^*(s+w) \bmod n)] = \frac{1}{2}$.

Corollary 1 is for the case when the window wraps around from the higher end of the n-bit word to the lower end. If the window does not wrap around in the word, the corresponding result is presented in Lemma 3.

The use of these equations will lead us to approximate windows derived from a single addition. However, the ARX block cipher is an iterated cipher. Thus, after the first approximated addition, the input of all further subsequent additions changes. In particular, the input for the further additions is dependent on the input of the operand bits that precede this addition over all rounds approximated [5].

3.2.2. Pseudo-Linear Approximations of ARX Round Functions

1. Base Approximation
   The base approximation is a simple approximation that follows the windows until the target window. All exclusive-or operations and addition modulo 2^n operations are preserved, assuming that the carry into all windows is 0 [3].

2. Carry Patterns
   A carry pattern is a series of carry values, $c_i \in \{0, 1\}$ where i denotes to the approximated addition window that may have a carry into it.

   Multiple carry patterns, indexed by j, $C^j = (c_0, \ldots, c_i, \ldots, c_{m-1})$ can be constructed for each base approximation; here j denotes a specific carry pattern for the approximation, i the approximated window, and m the total number of windows approximated. If c_i

= 1, then the carry going into the ith approximated addition window is 1. Thus, the base approximation is overlaid by the m carry patterns, $C^j + base$ to result in m estimates of the target window. [5].

3. Computing Bias
If cp carry patterns are used, the bias may be experimentally computed to be the difference between the probability of the approximation being correct and the probability of correctly guessing a carry pattern at random, with cp tries. The carry patterns will be correct with probability $\frac{cp}{2^w}$ instead of $\frac{1}{2^w}$ since each pattern represents a different approximation. According to McKay [3] the bias is computed using Equation (10)

$$bias = \frac{times\ correct\ - \frac{\#patterns}{2^w} \times pairs}{pairs}. \tag{10}$$

3.3. Comparison between Pseudo-Linear Cryptanalysis and Linear Cryptanalysis

The pseudo-linear attack is clearly inspired by linear cryptanalysis, and there are several differences that should be noted. Table 3, shows these differences [3].

Table 3. The main differences between linear and pseudo-linear cryptanalysis.

Linear Cryptanalysis	Pseudo-Linear Cryptanalysis
The effect of several approximations can be easily concatenated and simplified because there is only one operation (exclusive-or).	The effect of several approximations cannot be concatenated and simplified because the two operations (exclusive-or and addition modulo 2^w) do not commute.
Combining key bits across rounds into a single function of the key, independent of plaintext bits, is possible.	Cannot combine key bits across rounds into a single function of the key independent of plaintext.
The approximation may be used for a distinguisher as well as for key recovery.	The approximation includes a non-linear function of key and plaintext bits, and cannot be used as a distinguisher but can be used for key recovery.
Approximation of a single modular addition for large window sizes has low bias.	Approximation of a single modular addition can result in high accuracy prediction of large windows.

3.4. Cryptanalysis of SPECK

Since the publication of SPECK in 2013 [1,2], there have been several analyses of the cipher, most focused on differential and linear cryptanalysis. Beaulieu et al. summarise the cryptanalysis and implementation results [16]. Section 3.4.1, reviews different methods of cryptanalysis on SPECK. Section 3.4.2, reviews some key results on linear cryptanalysis, as the focus of this paper is to combine linear and pseudo-linear cryptanalysis.

3.4.1. Different Methods of Cryptanalysis on SPECK

There are two previous works that have the best results of the differential cryptanalysis on SPECK. Ling et al. (2016) [17] present differential cryptanalysis of ARX block ciphers. They develop a framework for finding differential characteristics. Lee et al. (2018) [18] present a method of approximating the differentials probability using a SAT solver. In addition, Yunwen et al. (2017) [19] presents a rotational-XOR cryptanalysis on SPECK.

Table 4 summarizes the result of Differential and rotational cryptanalysis on the SPECK family.

Table 4. Summary: differential and rotational cryptanalysis on the SPECK family.

N/K	Ref.	Type of Attack	Number of Rounds	Data Complexity	Time Complexity
32/64	[19]	Rotational	12	NA	NA
	[20]	Differential	12	2^{31}	2^{63}
	[17]	Differential	14	$2^{30.47}$	$2^{62.47}$
	[18]	Differential	15	$2^{31.39}$	$2^{63.39}$
48/72	[17]	Differential	15	$2^{45.31}$	$2^{69.31}$
	[18]	Differential	16	$2^{47.8}$	$2^{95.8}$
48/96	[20]	Differential	13	2^{48}	2^{96}
	[19]	Rotational	15	NA	NA
	[17]	Differential	16	$2^{45.31}$	$2^{93.31}$
	[18]	Differential	17	$2^{47.8}$	$2^{71.8}$
64/96	[17]	Differential	19	$2^{61.56}$	$2^{93.56}$
64/128	[19]	Rotational	13	NA	NA
	[20]	Differential	15	2^{64}	2^{128}
	[17]	Differential	20	$2^{61.56}$	$2^{125.56}$
96/96	[17]	Differential	20	$2^{95.94}$	$2^{95.94}$
96/144	[19]	Rotational	13	NA	NA
	[20]	Differential	13	2^{93}	2^{141}
	[17]	Differential	21	$2^{95.94}$	$2^{143.94}$
128/128	[17]	Differential	23	$2^{125.35}$	$2^{125.35}$
128/128	[17]	Differential	24	$2^{125.35}$	$2^{189.35}$
128/128	[19]	Rotational	13	NA	NA
	[20]	Differential	15	2^{126}	2^{254}
	[17]	Differential	25	$2^{125.35}$	$2^{253.35}$

3.4.2. Linear Cryptanalysis of SPECK

Ashur and Bodden (2016) [10] find a linear approximation of reduced round SPECK using Cho and Pieprzyk's property of modular addition. Bodden (2018) [9] improves on [10] by using the Wallén algorithm to increase the number of attacked rounds by one. These two papers do not have the best results on linear cryptanalysis of SPECK but they present different techniques and both of them are focused on discovering a distinguisher and do not attempt to recover key bits.

Yao et al. (2015) [11] were the first to implement Wallén's enumeration algorithm for the purpose of obtaining linear distinguishers and key recovery attacks on the SPECK family. For SPECK 32/64, SPECK 48/72, SPECK 48/96, SPECK 64/96, SPECK 64/128, and SPECK 96/96 they have the distinction of having attacked the largest number of rounds.

Liu et al. (2016) [13] have the largest number of attacked rounds on the two largest size members of the SPECK family. They show in their paper that they are able to attack more rounds on the large SPECK with large key size especially for SPECK 96/144, SPECK 128/192, and SPECK 128/256. The number of the attacked rounds is larger than [11]. Moreover, they present a new search method for linear approximations of the SPECK family by using the partial linear mask table (pLMT).

Fu et al. [12] present differential and linear trails (hull) for an ARX cipher and implement their approach on SPECK. For the linear trails (hull), they use the Wallén algorithm and the Mixed Integer Linear Programming model (MILP). Table 5 summarises the results of these previous works.

This paper presents a novel attack: the combination of linear and pseudo-linear attacks. It illustrates improvements to SPECK attacks based solely on Cho–Pieprzyk approximations by combining them with pseudo-linear approximations.

Table 5. Summary: linear cryptanalysis on the SPECK family.

N	Ref.	Number of Rounds	Guessed Key Bit/K	Bias	Data Complexity	Time Complexity
32	[10]	7	LT	2^{-14}	2^{28}	2^{28}
	[9]	8	LT	2^{-15}	2^{30}	2^{30}
	This work	9	36/64	$2^{-13.348}$	$2^{26.68}$	$2^{62.68}$
	[12]	9	LT	2^{-14}	NA	NA
	[13]	9	LT	2^{-14}	NA	NA
	[11]	12	29/64	2^{-14}	$2^{30.87}$	$2^{60.21}$
48	[10]	8	LT	2^{-22}	2^{44}	2^{44}
	[9]	9	LT	2^{-23}	2^{46}	2^{46}
	This work	10	27/72	$2^{-22.436}$	$2^{44.872}$	$2^{71.872}$
	[12]	10	LT	2^{-22}	NA	NA
	[13]	10	LT	2^{-22}	NA	NA
	[11]	11	24/72	2^{-20}	$2^{43.72}$	$2^{67.93}$
	This work	11	45/96	2^{-24}	2^{48}	2^{93}
	[11]	12	48/96	2^{-20}	$2^{43.72}$	$2^{91.93}$
64	[9]	11	LT	2^{-31}	2^{62}	2^{62}
	[10]	11	LT	2^{-32}	2^{64}	2^{64}
	[13]	12	LT	2^{-30}	NA	NA
	This work	13	28/96	$2^{-29.88}$	$2^{59.76}$	$2^{87.76}$
	[11]	13	31/96	2^{-25}	$2^{54.63}$	$2^{85.74}$
	[12]	13	LT	2^{-30}	NA	NA
	[11]	14	31/96	2^{-31}	$2^{62.73}$	$2^{94.87}$
	[11]	14	63/128	2^{-25}	$2^{54.80}$	$2^{117.7}$
	This work	14	49/128	$2^{-31.58}$	$2^{63.16}$	$2^{112.16}$
	[11]	15	63/128	2^{-31}	$2^{62.73}$	$2^{126.9}$
96	[11]	8	47/96	2^{-11}	$2^{27.65}$	$2^{74.7}$
	[11]	9	95/144	2^{-11}	$2^{27.65}$	$2^{122.7}$
	[10]	10	LT	2^{-47}	2^{92}	2^{92}
	This work	10	46/96	$2^{-21.86}$	$2^{43.72}$	$2^{89.72}$
	[9]	12	LT	2^{-48}	2^{96}	2^{96}
	This work	12	76/144	$2^{-29.238}$	$2^{58.476}$	$2^{134.476}$
	[12]	15	LT	2^{-45}	NA	NA
	[13]	17	NA/144	2^{-45}	2^{92}	2^{96}
128	[11]	7	191/256	2^{-11}	$2^{28.30}$	$2^{220.7}$
	[11]	8	63/128	2^{-11}	$2^{28.30}$	$2^{92.69}$
	[11]	9	127/192	2^{-11}	$2^{28.30}$	$2^{156.7}$
	[10]	11	LT	2^{-63}	2^{144}	2^{144}
	This work	11	50/128	$2^{-28.179}$	$2^{56.358}$	$2^{106.358}$
	[9]	13	LT	2^{-58}	2^{116}	2^{116}
	This work	13	122/192	$2^{-31.299}$	$2^{62.598}$	$2^{184.598}$
	This work	14	173/256	$2^{-33.415}$	$2^{66.83}$	$2^{239.83}$
	[12]	16	LT	2^{-58}	NA	NA
	[13]	18	NA/192	2^{-61}	2^{124}	2^{128}
	[13]	19	NA/256	2^{-61}	2^{124}	2^{192}

N is the block size and K is the key size. LT refers to a Linear Trail used as a distinguisher. NA refers to Not Available (not reported in the paper).

4. Pseudo-Linear Cryptanalysis Attacks on Reduced-Round SPECK 32/64

In this section, we derive pseudo-linear approximations for 4 and 6 round attacks on SPECK 32/64. That is, we approximate the addition mod 2^n by addition mod 2^w, for $w = 2, 3, 4$, using some carry patterns for each approximated addition window unless its right end is at the least significant bit (lsb) of the word. In later sections, we combine these approximations with linear approximations.

4.1. Four-Round Attack

We begin our work by implementing the pseudo-linear cryptanalysis on four rounds of SPECK 32/64, as a meet-in-the-middle attack with a four-bit window approximated by two rounds in the forward direction and two backward. The approximation requires 12 key bits. The first addition operation is before the key round injection, thus it can be performed for the full word without windows or carry patterns, and is denoted $NewPL$ in Table 6, which shows the approximation for four rounds meeting at $xl_1^2(0,1)$. Note that $x(i, i+w-1)$ is a window of size w beginning at msb i and ending at lsb $i+w-1$.

Table 6. The pseudo-linear attack approximation for 4 rounds meeting at $xl_1^2(0,1)$, $w = 2$.

Round	Encryption	Decryption
1	$NewPL = (PL \ggg 7) \boxplus_{16} PR$ $xl_1^1 = NewPL(0,1) \oplus k_1^1(0,1)$ $xl_2^1 = NewPL(7,8) \oplus k_2^1(7,8)$ $xr_1^1 = (PR(14,15)) \lll 2) \oplus xl_1^1$	
2	$xl_1^2(0,1) = ((xl_2^1 \ggg 7) \boxplus_2 xr_1^1) \oplus k_1^2(0,1)$	
	$xl_1^2(0,1) \equiv xl_1^3(0,1)$	
3		$xl_1^3(0,1) \approx ((xl_1^4(9,10) \oplus k_1^3(9,10)) \boxminus_2 xr_1^3(9,10))) \lll 7$ $xr_1^3(9,10) \approx ((xl_2^4(11,12) \oplus xr_3^4(11,12))) \ggg 2$
4		$xl_2^4(11,12) \approx ((CL(4,5) \oplus k_1^4(4,5)) \boxminus_2 xr_2^4(4,5)) \lll 7$ $xl_1^4(9,10) \approx ((CL(2,3) \oplus k_1^4(2,3)) \boxminus_2 xr_1^4(2,3)) \lll 7$ $xr_1^4(2,3) = (CR(4,5) \oplus CL(4,5)) \ggg 2$ $xr_2^4(4,5) = (CR(6,7) \oplus CL(6,7)) \ggg 2$ $xr_3^4(11,12) = (CR(13,14) \oplus CL(13,14)) \ggg 2$

Figure 5 shows how we drive our target window through two rounds of SPECK 32/64 and how a meet-in-the-middle attack works for four rounds. Approximations for windows of sizes $w = 3$ and $w = 4$ are available in the Appendix A.

There are two approximations of interest, $xl_1^2(0,1)$ (Table 6, Round 2) and $xl_1^3(0,1)$ (Table 6, Round 3), each of size $w = 2$. $xl_1^2(0,1)$ is the first window (windows are denoted in subscript) of the second round (rounds are denoted in superscript) in the left half. It consists of bits 0 through 1. Similarly, $xl_1^3(0,1)$ is the first window in the third round, consisting of the two least significant bits of the left half state. Each window represents a pseudo-linear approximation from a particular direction (forward or backward), and the approximation meets in the middle, at the target window, $xl_1^2(0,1) \equiv xl_1^3(0,1)$. Note that for window $xl_1^2(0,1)$, the approximation is exact when the key is correct because the summands are exact and the window begins at the least significant bit and the incoming carry is always zero. The approximation for window $xl_1^3(0,1)$ needs an approximation for $xl_1^4(9,10)$ and $xr_1^3(9,10)$, which, in turn, needs an approximation for window $xl_2^4(11,12)$ (window $xr_3^4(11,12)$, $xr_2^4(6,7)$, and $xr_1^4(2,3)$ are computed exactly).

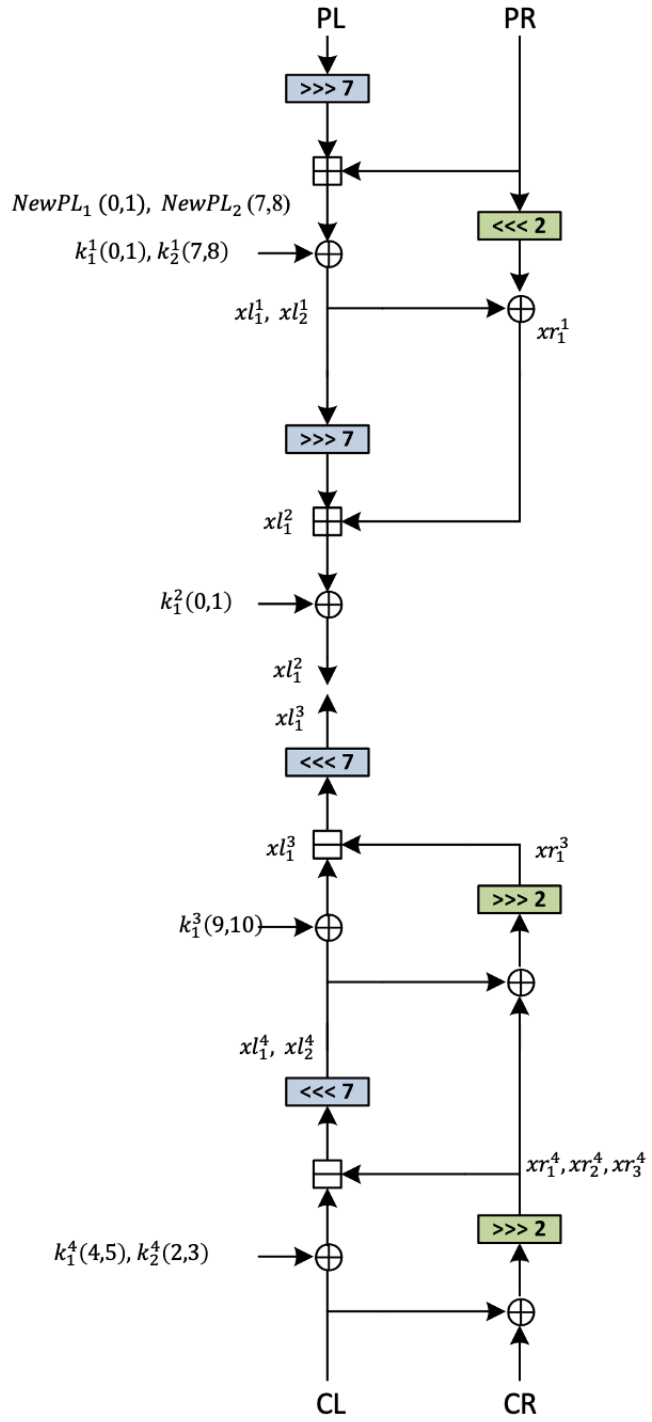

Figure 5. Four-round pseudo-linear attack on SPECK 32/64.

First, we begin with approximating windows $xl_1^4(9,10)$, $xl_2^4(11,12)$, which use the correct value of summand windows, and the approximation error is only due to an error in carry.

McKay shows [4] that the bias of an incoming carry into a window with lsb at position s, assuming a uniform distribution of the bits that have lower significance, is $2^{-(s+1)}$. If $carry_s$ denotes the carry coming into a window with the least significant bit s, and e_1, e_2 are the error in the first and second bit in a window.

$$Pr[e_1 e_0 = 00 \mid carry_s = 0] = 1$$

In addition, the probability with which the intermediate carry is correctly computed (by the pseudo-linear summation which assumes the incoming carry is zero) when the incoming carry is actually one is $\frac{1}{2}$, which is also the probability with which the msb is correctly computed when the incoming carry is one (of the four possibilities for the pairs of lsbs of the summand window, when both lsbs are 0, the approximated intermediate carry is 0 as is the true one. Similarly, when both lsbs are 1, both the true and the approximated intermediate carries are 1. When one of the two lsbs is 0 and the other is 1, the approximated carry is zero, but the true carry is one.). Hence:

$$Pr[e_1e_0 = 01 \mid carry_s = 1] = Pr[e_1e_0 = 11 \mid carry_s = 1] = \frac{1}{2}$$

An approximation which uses the correct values for summand windows can never have an error in the msb if the lsb is correct (that is, the carry was correctly estimated), hence:

$$Pr[e_1e_0 = 10 \mid carry_s = 1] = 0$$

We start from round 4 to approximate $xl_1^4(9,10)$ and $xl_2^4(11,12)$ (note: xl_1^4, the first two-bit window of the fourth round, is located at $(2,3)$ before rotation and, similarly, xl_2^4 is located at $(4,5)$). Thus, we need to calculate the probabilities of these two windows as follows:

For window xl_1^4 at $(2,3)$:

- $Pr[e_1e_0 = 00] = Pr[carry_s = 0] = \frac{1}{2} + 2^{-(3)} = 0.625$
- $Pr[e_1e_0 = 01] = Pr[carry_s = 1]Pr[carry + s + 1 = 0 \mid carry_s = 1] = \frac{1}{2}(1 - Pr[carry_s = 0]) = \frac{1}{2}(1 - 2^{-(3)}) = 0.1875$
- $Pr[e_1e_0 = 10] = 0$
- $Pr[e_1e_0 = 11] = \frac{1}{2}(1 - 2^{-(3)}) = 0.1875$

For window xl_2^4 at $(4,5)$:

- $Pr[e_1e_0 = 00] = Pr[carry_s = 0] = \frac{1}{2} + 2^{-(5)} = 0.53125$
- $Pr[e_1e_0 = 01] = Pr[carry_s = 1]Pr[carry + s + 1 = 0 \mid carry_s = 1] = \frac{1}{2}(1 - Pr[carry_s = 0]) = \frac{1}{2}(1 - 2^{-(5)}) = 0.2343$
- $Pr[e_1e_0 = 10] = 0$
- $Pr[e_1e_0 = 11] = \frac{1}{2}(1 - 2^{-(5)}) = 0.2343$

For window $xr_1^3(9,10)$:

- $xr_1^3(9,10) \approx xl_2^4(11,12) \oplus xr_3^4(11,12)$; the error probabilities are those of xl_2^4, as xr_3^4 is approximated with zero error.

Finally, window $xl_1^3(0,1)$:

- is obtained by adding $xr_1^3(9,10)$ and $xl_1^4(9,10)$. This is the target window and we are trying to compute the entire window correctly, so we compute $Pr[e_1e_0 = 00]$. If the incoming carry is $carry_s = 0$, we have 16 possibilities for two bit errors in each summand window, 6 of these, with an incoming carry of zero (the possibilities are: both summands have error 00 or 10; with probability half, when both have error patterns 01 or 11; with probability half when the summands are 01 and 11 (two possibilities).), and 8 with an incoming carry of one (with probability half, each of the following pairs of summand errors will result in an error of 00 in the approximated window when the true value of the incoming carry is one; each pair occurs twice: 00 and 01, 00 and 11, 01 and 10, 10 and 11) , give $e_1e_0 = 00$. The total probability is obtained using the probabilities of errors in windows $xl_1^4(9,10)$ and $xl_2^4(11,12)$ computed above to obtain:

- $Pr[e_1e_0 = 00] = 0.4198(Pr[carry_s = 0]) + 0.1992(Pr[carry_s = 1]) = 0.3097.$

To experimentally verify our probability, we carried out 200 experiments for the pseudo-linear approximation each with a key chosen at random. We used 2^{10} P/C pairs for each experiment. The average empirically determined probability for the $xl_1^2(0,1) \equiv xl_1^3(0,1)$ was 0.3476

The bias for this approximation is $2^{-3.35}$. Table 7 shows the results of these approximations. The bias for the pseudo-linear approximation above is experimentally computed as described in Section 3 and verified theoretically by computing the probability of each window in this approximation.

Table 7. Results of the pseudo-linear approximations: four rounds of SPECK 32/64.

Approximation	Window Size	Guessed Key Bits	Bias	Data Complexity	Time Complexity
Table 6	2 bits	12	$2^{-3.35}$	2^{10}	2^{22}
Appendix A	3 bits	17	$2^{-2.01}$	2^{10}	2^{27}
Appendix A	4 bits	22	$2^{-1.65}$	2^{10}	2^{32}

4.2. Six-Round Attack

The maximum number of rounds we can analyze for key recovery in SPECK 32/64 using the pseudo-linear approximation is 6. We are limited by the fact that there are several key bits involved in the approximation and the pseudo-linear cryptanalysis requires the adversary to try all possibility of the key bits that are involved in the approximation. Using this approximation 44 key bits may be recovered with data complexity 2^{10} and time complexity 2^{54}.

5. Partly-Pseudo-Linear Cryptanalysis with Illustration on SPECK

In this paper, we present a new attack for the ARX block cipher which we term the Partly-Pseudo-Linear attack: a meet-in-the-middle combination of pseudo-linear and linear attacks. We show that linear cryptanalysis relying on Cho–Pieprzyk approximations of modular addition is improved by replacing some rounds of linear approximation with pseudo-linear approximations. Using the approach of Bodden and Ashur [9,10], we find the longest linear trails to approximate a window of two consecutive bits in each direction (forward and backward). Of these, we choose the trail(s) that would combine with a lower-error pseudo-linear attack.

The pseudo-linear attack itself first uses pseudo-linear approximations for each addition operation. The approximations require the use of key bits, but because the approximation is limited to a window, fewer key bits are used than in the entire round. Every bit of the window is computed with considerable accuracy as a function of a few key bits. The larger the window size the more key bits are required; similarly, the more rounds one covers (the more additions one approximates) the more key bits are required. This typically limits the window size, and we focus on window sizes of two bits. Thus, our pseudo-linear approximation computes each bit of a window of size two bits in one direction, as a function of some key bits. We use the xor of this window and compare it to the xor computed using linear cryptanalysis in the other direction as described above.

We have done this analysis in the forward direction and backward direction since we will use one of these directions by combining it with the pseudo-linear attack. Figure 6 shows an approximated round of SPECK using Cho–Pieperzyk approximations in each direction. Note that the constraint of requiring two consecutive bits in the window to be approximated restricts the windows that can be approximated.

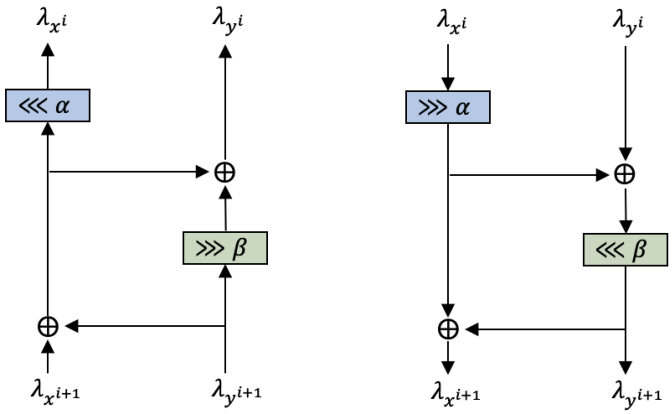

Figure 6. Transformation of a round function of SPECK: forward (**right**) and backward (**left**).

The final linear approximation approximates the xor of the two bits of the window. The bias of the Partly-Pseudo-Linear cryptanalysis approximation hence consists of two parts.

1. The first part is the bias of xor of the bits of the window when the window is computed using the pseudo-linear approximation.
2. The second part is the bias for the linear approximation, computed using traditional linear approaches. The combination of these two biases using the piling up lemma allows us to determine the number of plaintext and ciphertext pairs that we should use in our experiments.

5.1. Implementation of the Partly-Pseudo-Linear Attack on SPECK *32/64*

We illustrate the Partly-Pseudo-Linear attack (including the analytical approach to determining the bias of a pseudo-linear attack) on 6 and 9 rounds of SPECK 32/64.

5.1.1. Six-Round Partly-Pseudo-Linear Attack

We find the longest linear trail arising from a two-consecutive-bit target window, discovering one that covers four rounds in the backward direction and combines it with two rounds approximated using pseudo-linear cryptanalysis in the forward direction. Table 8 shows the derivation of the mask that is used in the linear part of the Partly-Pseudo-Linear attack. Note that we do not cover more rounds than four rounds because rotation breaks the requirement for two consecutive ones.

Table 8. Linear trail of SPECK 32/64 for four rounds—six-rounds Partly-Pseudo-Linear attack (Starting with `0x30000000` forward).

Round	Cost	λ_{x^i}	λ_{y^i}	$\lambda_{x^{i+1}}$	$\lambda_{y^{i+1}}$	Reason to Stop
1	1	`0x0006`	`0x0000`	`0x7800`	`0x6000`	
2	2	`0x7800`	`0x6000`	`0x8331`	`0x83c1`	
3	3	`0x8331`	`0x83c1`	`0xe019`	`0x831f`	
4	3	`0xe019`	`0x831f`	`0xf0be`	`0xc37e`	
5		`0xf0be`	`0xc37e`			Broke requirement of consecutive ones for `0xf0be` $\ggg 7$.

The window size of the pseudo-linear approximation is two, $w = 2$, and 6 key bits are required for the approximation. In the first round, the addition operation is performed before the key round injection; thus, it can be performed exactly for the full word without any need for an approximation. The second round involves a single modular addition

that is approximated by an addition over $2^w = 2^2$ with zero carry. Table 9 shows the Partly-Pseudo-Linear approximation for six rounds meeting at $xl_1^2(1,2)$, which denotes the first (and only) window in round 2. The window is in the left word. (Recall that xl_t^j represents the tth window of the left word in the jth round.)

Table 9. The approximation for the Partly-Pseudo-Linear attack: six rounds, meeting at $xl_1^2(1,2)$.

Round	Encryption	Decryption
1	$NewPL = (PL \ggg 7) \boxplus_{16} PR$ $xl_1^1 = NewPL(1,2) \oplus k_1^1(1,2)$ $xl_2^1 = NewPL(8,9) \oplus k_2^1(8,9)$ $xr_1^1 = (PR(15,16\,mod\,16) \lll 2) \oplus xl_1^1$	
2	$xl_1^2(1,2) \approx ((xl_2^1 \ggg 7) \boxplus_2 xr_1^1) \oplus k_1^2(1,2)$	
	$xl_1^2(1) \oplus xl_1^2(2) \equiv linearT$	
3 to 6		$linearT \approx \lambda_{x^4}.CL \oplus \lambda_{y^4}.CR$

Figure 7 shows how the target window travels through two rounds of SPECK 32/64.

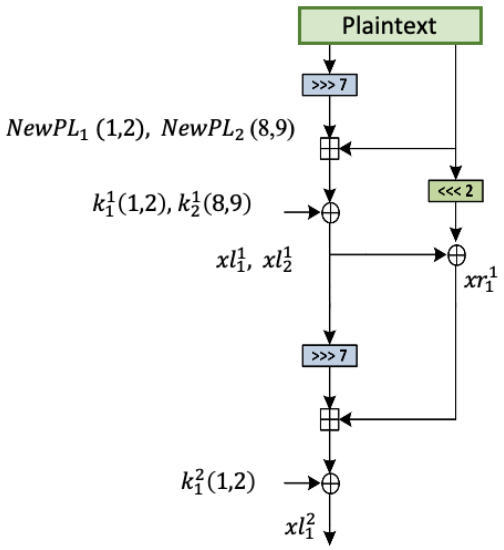

Figure 7. Pseudo-linear approximation of first two rounds of SPECK 32/64.

Linear cryptanalysis, and the techniques for computing bias are well-established. On the other hand, pseudo-linear cryptanalysis is new, and we describe here an approach to computing the bias of the xor of a 2-bit window approximated using one instance of the pseudo-linear approximation, as in this case.

Consider the 2-bit target window of interest, $xl_1^2(1,2)$ (Table 9, Round 2), where the pseudo-linear approximation meets the linear approximation. The pseudo-linear part of the attack approximates the xor of the two bits of the window, $xl_1^2(1) \oplus xl_1^2(2)$, by approximating the window through multiple rounds, and then, finally, xoring the two bits. The linear part of the attack follows the Cho–Pieprzyk property of modular addition through multiple rounds.

Let the pseudo-linear approximation of the xor be denoted ζ. Because this is the first instance of pseudo-linear approximation, the values of the component windows being added to obtain the target window are correct. That is, no approximations have been used while obtaining $xl_2^1 \ggg 7$ and xr_1^1. Thus, if the incoming carry is zero, the entire target window, $xl_1^2(1,2)$, is estimated correctly and, hence, so is ζ. McKay shows [4] that the bias of an incoming carry into a window with lsb s, assuming a uniform distribution of the bits

that have lower significance, is $2^{-(s+1)}$. If $carry_s$ denotes the carry coming into a window with the least significant bit s,

$$Pr[\zeta \text{ is correct} \mid carry_s = 0] = 1$$

Now consider the case when the incoming carry is 1. The lsb in the approximated window of the sum will be incorrect with probability 1. However, there will be instances when the msb is also approximated incorrectly, in which case the xor will be correct. Because the component windows being added are correct, the msb will be correct if and only if the intermediate carry, going from the lsb to the msb, is incorrect. Of the four possible combinations of the lsbs of the two windows that are being summed, the pseudo-linear approximation approximates the intermediate carry correctly when:

(a) Both bits are zero (and the intermediate carry is zero, its value does not depend on the value of the incoming carry).

(b) Both bits are one (the intermediate carry is one, independent of the incoming carry).

Thus the probability with which the intermediate carry is correctly computed when the incoming carry is one is $\frac{1}{2}$, which is also the probability with which the msb is correctly computed when the incoming carry is one. Hence:

$$Pr[\zeta \text{ is correct} \mid carry_s = 1] = \frac{1}{2}$$

Hence:

$$\begin{aligned} Pr[\zeta \text{ is correct}] &= \Pr[\zeta \text{ is correct} \mid carry_s = 0]Pr[carry_s = 0] + \Pr[\zeta \text{ is correct} \mid carry_s = 1]Pr[carry_s = 1] \\ &= 1 \times (\frac{1}{2} + 2^{-(s+1)}) + \frac{1}{2} \times (\frac{1}{2} - 2^{-(s+1)}) \\ &= \frac{1}{2} + \frac{1}{4} + 2^{-(s+2)} \end{aligned}$$

For the pseudo-linear approximation of Table 9, we observe that $s = 1$, hence:

$$Pr[\zeta \ = \ correct] = \frac{1}{2} + \frac{3}{8} \approx \frac{1}{2} + 2^{-1.415}$$

Our bias for the first approximation is larger than the bias of a first-round Cho–Pieprzyk approximation.

To experimentally verify our bias prediction, we carried out 150 experiments for the pseudo-linear approximation each with a key chosen at random. We used 2^{10} P/C pairs for each experiment. The average empirically determined bias for the xor of the target window was $2^{-1.41}$.

Thus, the attack of the approximation of Table 9, using the masks of Table 8, has the following characteristics.

- Bias: The bias for this approximation is a combination of the experimentally-verified bias of the pseudo-linear approximation of the exclusive-or of the two-bit window ($2^{-1.415}$)and the bias of the linear approximation (2^{-10}) using the piling-up lemma: $2 \times 2^{-1.415} \times 2^{-10} = 2^{-10.415}$
- Data complexity: We use the square of the inverse of the bias of the linear approximation: $2^{20.83}$
- Time complexity: Data complexity multiplied by the complexity of trying all possibilities for the number of key bits in the pseudo-linear approximation: $2^{20.83} \times 2^6 = 2^{26.83}$

The summary of attack properties is presented in Table 10. We were able to determine all six key bits correctly for each of the randomly-chosen keys in a list of three best keys.

Table 10. Partly-Pseudo-Linear approximation of six rounds of SPECK 32/64. Window size is 2 bits.

Approximation	Window	No. of Key Bits Correctly Determined	Bias	Data Complexity	Time Complexity
Table 9	2 bits	6	$2^{-10.415}$	$2^{20.83}$	$2^{26.83}$

5.1.2. Nine-Round Partly-Pseudo-Linear Attack

We describe a nine-round key recovery attack. Here in this nine-round attack, we use a different mask that covers four rounds and can be combined with our pseudo-linear approximation. Table 11 shows the mask that is used in this attack.

Table 11. Linear trail of SPECK 32/64 for four rounds—nine rounds Partly-Pseudo-Linear attack (Starting with `0x0c000000` forward).

Round	Cost	λ_{x^i}	λ_{y^i}	$\lambda_{x^{i+1}}$	$\lambda_{y^{i+1}}$	Reason to Stop
1	1	`0x0c00`	`0x0000`	`0x0078`	`0x0060`	
2	2	`0x0078`	`0x0060`	`0x3183`	`0xc183`	
3	3	`0x3183`	`0xc183`	`0x19e0`	`0x1f83`	
4	3	`0x19e0`	`0x1f83`	`0xbef0`	`0x7ec3`	
5		`0xbef0`	`0x7ec3`			Broke requirement for consecutive ones for `0xfbc2` $\ggg 7$.

We use five rounds forward of pseudo-linear approximation (the maximum given the time complexity constraints of the non-linear approximation) and four rounds backward using a linear approximation. The window size of the pseudo-linear approximation is two, $w = 2$. Table 12 shows the approximation for nine rounds meeting at $xl_1^5(10, 11)$; note that 36 key bits are required. Figure 8 shows how we derive the pseudo-linear approximation of the target window through five rounds of SPECK 32/64.

Table 12. The Partly-Pseudo-Linear approximation for nine rounds meeting at $xl_1^5(10, 11)$.

Round	Encryption	Decryption
1	$NewPL = (PL \ggg 7) \boxplus_{16} PR$	
	$xl_1^1 = NewPL(15, 16\, mod\, 2^4) \oplus k_1^1(15, 16\, mod\, 2^4)$	
	$xl_2^1 = NewPL(1, 2) \oplus k_2^1(1, 2)$	
	$xl_3^1 = NewPL(4, 5) \oplus k_3^1(4, 5)$	
	$xl_4^1 = NewPL(6, 7) \oplus k_4^1(6, 7)$	
	$xl_5^1 = NewPL(8, 9) \oplus k_5^1(8, 9)$	
	$xl_6^1 = NewPL(10, 11) \oplus k_6^1(10, 11)$	
	$xl_7^1 = NewPL(13, 14) \oplus k_7^1(13, 14)$	
	$xr_1^1 = (PR(13, 14) \lll 2) \oplus xl_1^1$	
	$xr_2^1 = (PR(15, 16\, mod\, 2^4) \lll 2) \oplus xl_2^1$	
	$xr_3^1 = (PR(2, 3) \lll 2) \oplus xl_3^1$	
	$xr_4^1 = (PR(4, 5) \lll 2) \oplus xl_4^1$	
	$xr_5^1 = (PR(6, 7) \lll 2) \oplus xl_5^1$	
	$xr_6^1 = (PR(8, 9) \lll 2) \oplus xl_6^1$	
	$xr_7^1 = (PR(11, 12) \lll 2) \oplus xl_7^1$	
2	$xl_1^2 \approx ((xl_4^1 \ggg 7) \boxplus_2 xr_1^1) \oplus k_1^2(15, 16\, mod\, 2^4)$	
	$xl_2^2 \approx ((xl_5^1 \ggg 7) \boxplus_2 xr_2^1) \oplus k_2^2(1, 2)$	
	$xl_3^2 \approx ((xl_7^1 \ggg 7) \boxplus_2 xr_4^1) \oplus k_3^2(6, 7)$	
	$xl_4^2 \approx ((xl_1^1 \ggg 7) \boxplus_2 xr_5^1) \oplus k_4^2(8, 9)$	
	$xl_5^2 \approx ((xl_2^1 \ggg 7) \boxplus_2 xr_6^1) \oplus k_5^2(10, 11)$	
	$xr_1^2 \approx (xr_7^1 \lll 2) \oplus xl_1^2$	
	$xr_2^2 \approx (xr_1^1 \lll 2) \oplus xl_2^2$	
	$xr_3^2 \approx (xr_3^1 \lll 2) \oplus xl_3^2$	
	$xr_4^2 \approx (xr_4^1 \lll 2) \oplus xl_4^2$	
	$xr_5^2 \approx (xr_5^1 \lll 2) \oplus xl_5^2$	

Table 12. *Cont.*

Round	Encryption	Decryption
3	$xl_1^3 \approx ((xl_4^2 \ggg 7) \boxplus_2 xr_2^2) \oplus k_1^3(1,2)$ $xl_2^3 \approx ((xl_1^2 \ggg 7) \boxplus_2 xr_4^2) \oplus k_2^3(8,9)$ $xl_3^3 \approx ((xl_2^2 \ggg 7) \boxplus_2 xr_5^2) \oplus k_3^3(10,11)$ $xr_1^3 \approx (xr_1^2 \lll 2) \oplus xl_1^3$ $xr_2^3 \approx (xr_3^2 \lll 2) \oplus xl_2^3$ $xr_3^3 \approx (xr_4^2 \lll 2) \oplus xl_3^3$	
4	$xl_1^4 \approx ((xl_2^3 \ggg 7) \boxplus_2 xr_1^3) \oplus k_1^4(1,2)$ $xl_2^4 \approx ((xl_1^3 \ggg 7) \boxplus_2 xr_3^3) \oplus k_2^4(10,11)$ $xr_1^4 \approx (xr_2^3 \lll 2) \oplus xl_2^4$	
5	$xl_1^5(10,11) \approx ((xl_1^4 \ggg 7) \boxplus_2 xr_1^4) \oplus k_1^5(10,11)$	
	$xl_1^5(10) \oplus xl_1^5(11) \equiv linearT$	
6 to 9		$linearT \approx \lambda_{x^4}.CL \oplus \lambda_{y^4}.CR$

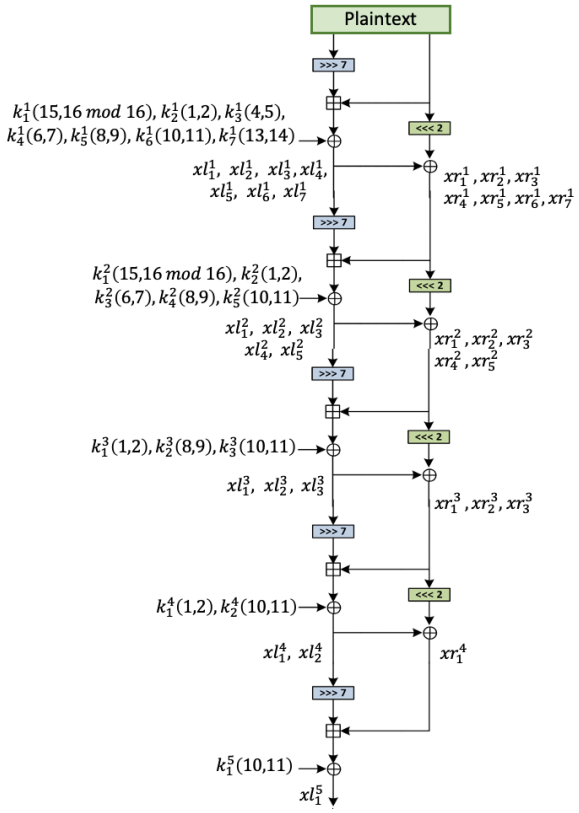

Figure 8. The pseudo-linear approximation of the first 5 rounds of SPECK 32/64 for a nine-round Partly-Pseudo-Linear attack.

For SPECK 32/64, the maximum number of rounds we can reach is nine rounds with a recovery of 36 key bits (See Table 13).

Table 13. Summary of the Partly-Pseudo-Linear approximation: nine rounds of SPECK 32/64, with window size $w = 2$.

Approximation	Window Size	Key	Bias	Data Complexity	Time Complexity
Table 12	2 bits	36	$2^{-13.348}$	$2^{26.68}$	$2^{62.68}$

5.2. The Partly-Pseudo-Linear Attack on the Large Variants of SPECK

The larger variants of SPECK correspond to a larger block with two or three different key sizes. This gives us two features. First, with a larger key size, we have a larger brute-force attack to compare with, so a pseudo-linear attack can cover more rounds in spite of requiring key bits in the approximation. Second, with a larger block size, it is harder to break the mask λ through bitwise rotation.

Table 14 summarizes the results of the Partly-Pseudo-Linear attack on the SPECK family. Details of the individual attacks are in Appendices B–E. For the attacks on larger rounds, our bias predictions are limited by the ability to experimentally determines the error of pseudo-linear approximation.

Table 14. Summary: the Partly-Pseudo-Linear attack on larger variants of SPECK.

Block Size	Key Size	Rounds	Guessed Key Bits	Bias	Data Complexity	Time Complexity	Approximation
32	64	9	36	$2^{-13.348}$	$2^{26.68}$	$2^{62.68}$	Table 12
48	72	10	27	$2^{-22.436}$	$2^{44.872}$	$2^{71.872}$	Appendix B
	96	11	45	2^{-24}	2^{48}	2^{93}	
64	96	13	28	$2^{-29.88}$	$2^{59.76}$	$2^{87.76}$	Appendix C
	128	14	49	$2^{-31.58}$	$2^{63.16}$	$2^{112.16}$	
96	96	10	46	$2^{-21.86}$	$2^{43.72}$	$2^{89.72}$	Appendix D
	144	12	76	$2^{-29.238}$	$2^{58.476}$	$2^{134.476}$	
128	128	11	50	$2^{-28.179}$	$2^{56.358}$	$2^{106.358}$	Appendix E
	192	13	122	$2^{-31.299}$	$2^{62.598}$	$2^{184.598}$	
	256	14	173	$2^{-33.415}$	$2^{66.83}$	$2^{239.83}$	

6. Conclusions

This paper presents a new cryptanalysis of the ARX block cipher Partly-Pseudo-Linear attack: combining linear and the pseudo-linear cryptanalysis. We illustrate this attack by combining linear approximations using Cho–Pieprzyk and pseudo-linear approximations on the SPECK family. We are able to extend distinguishers using Cho–Pieprzyk to key recovery attacks.

We are able to recover 36 encryption key bits for 9 rounds of SPECK 32/64, 45 key bits for 11 rounds of SPECK 48/96, 49 key bits for 14 rounds of SPECK 64/128, 76 key bits for 12 rounds of SPECK 96/144 and 173 key bits for 14 rounds of SPECK 128/256. We propose to apply our Partly-Pseudo-Linear attack to other ARX block ciphers with a design similar to SPECK. Moreover, we are exploring the combination of the pseudo-linear cryptanalysis attack with a linear cryptanalysis attack that uses Wallen's algorithm to improve our Partly-Pseudo-Linear attack.

Author Contributions: This work contributes to the doctoral dissertation of S.A.A., who did 80% of the work. All authors, S.A.A. and P.L.V. have read and approved the final version of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank Kerry McKay for suggesting we look at the SPECK cipher. PV was supported in part by NSF Award No. 2015253.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Four-Rounds Attack of Pseudo-Linear Cryptanalysis

Pseudo-linear cryptanalysis of four-round SPECK 32/64 with different window sizes:

1. Window size $w = 3$. The following Table A1 shows the approximation for four rounds meeting at $xl_1^2(0,2)$.

Table A1. The pseudo-linear attack approximation for 4 rounds meeting at $xl_1^2(0,2)$, $w = 3$.

Round	Encryption	Decryption
1	$NewPL = (PL \ggg 7) \boxplus_{16} PR$ $xl_1^1 = NewPL(0,2) \oplus k_1^1(0,2)$ $xl_2^1 = NewPL(7,9) \oplus k_2^1(7,9)$ $xr_1^1 = (PR(14, 16 \bmod 2^4) \lll 2) \oplus xl_1^1$	
2	$xl_1^2 = ((xl_2^1 \ggg 7) \boxplus_3 xr_1^1) \oplus k_1^2(0,2)$	
	$xl_1^2 \equiv xl_1^3$	
3		$xl_1^3 \approx ((xl_1^4 \oplus k_1^3(9,11)) \boxminus_3 xr_1^3)) \lll 7$ $xr_1^3 \approx ((xl_2^4 \oplus xr_3^4)) \ggg 2$
4		$xl_2^4 \approx ((CL(4,6) \oplus k_1^4(4,6)) \boxminus_3 xr_2^4) \lll 7$ $xl_1^4 \approx ((CL(2,4) \oplus k_1^4(2,4)) \boxminus_3 xr_1^4) \lll 7$ $xr_1^4 = (CR(4,6) \oplus CL(4,6)) \ggg 2$ $xr_2^4 = (CR(6,8) \oplus CL(6,8)) \ggg 2$ $xr_3^4 = (CR(13,15) \oplus CL(13,15)) \ggg 2$

2. Window size $w = 4$. The following Table A2 shows the approximation for four rounds meeting $xl_1^2(0,3)$.

Table A2. Four-round pseudo-linear attack, meeting at $xl_1^2(0,3)$.

Round	Encryption	Decryption
1	$NewPL = (PL \ggg 7) \boxplus_{16} PR$ $xl_1^1 = NewPL(0,3) \oplus k_1^1(0,3)$ $xl_2^1 = NewPL(7,10) \oplus k_2^1(7,10)$ $xr_1^1 = (PR(14, 17 \bmod 2^4)) \lll 2) \oplus xl_1^1$	
2	$xl_1^2 = ((xl_2^1 \ggg 7) \boxplus_2 xr_1^1) \oplus k_1^2(0,3)$	
	$xl_1^2 \equiv xl_1^3$	
3		$xl_1^3 \approx ((xl_1^4 \oplus k_1^3(9,12)) \boxminus_2 xr_1^3)) \lll 7$ $xr_1^3 \approx ((xl_2^4 \oplus xr_3^4)) \ggg 2$
4		$xl_2^4 \approx ((CL(4,7) \oplus k_2^4(4,7)) \boxminus_2 xr_2^4) \lll 7$ $xl_1^4 \approx ((CL(2,5) \oplus k_1^4(2,5)) \boxminus_2 xr_1^4) \lll 7$ $xr_1^4 = (NewCR(4,7)) \ggg 2$ $xr_2^4 = (NewCR(6,9)) \ggg 2$ $xr_3^4 = (NewCR(13, 16 \bmod 2^4)) \ggg 2$ $NewCR = CL \oplus CR$

Appendix B. The Partly-Pseudo-Linear Attack on SPECK 48

In SPECK 48 there are two key sizes: 72 bits and 96 bits. With SPECK 48/72, we are able to attack 10 rounds (four rounds using a pseudo-linear approximation and six rounds using a linear approximation). Using this approach, we are able to recover 27 key bits. With SPECK 48/96, we increase the pseudo-linear approximation by one more round and are able to recover 45 key bits.

In the previous attacks in Section 5, we show that the longest linear trail covers four rounds of SPECK 32/64 in the backward direction and Tables 8 and 11 show that the reason to stop after four rounds was that rotation broke the requirement for two consecutive ones.

Here in this attack, we use a mask that covers six rounds and the reason to stop is that exclusive-or breaks the requirement for two consecutive ones. Table A3 shows the mask that is used in this attack. As with SPECK 32, we can drive our target window $xl_1^7(1,0)$ backward from the ciphertext to build the pseudo-linear approximation.

Table A3. Linear trail of SPECK 48 for six rounds. (Starting with 0x000003000000 backward).

Round	$\lambda_{x^{i+1}}$	$\lambda_{y^{i+1}}$	λ_{x^i}	λ_{y^i}	Cost
1	0xe3036d	0xe03f62	0x3c0f03	0x5f3be3	5
2	0x0f0063	0x03000f	0xe3036d	0xe03f62	6
3	0x000363	0x0f0063	0x03000f	0x6ee30c	3
4	0x030300	0x600303	0x000363	0x0f0063	3
5	0x000300	0x000003	0x030300	0x600303	2
6	0x000003	0x000000	0x000300	0x000003	1

Appendix C. The Partly-Pseudo-Linear Attack on SPECK 64

In SPECK 64 there are two key sizes: 96 bits and 128 bits. With SPECK 64/96, we are able to attack 13 rounds (four rounds using a pseudo-linear approximation and nine rounds using linear approximations). Thus, we are able to recover 28 key bits. With SPECK 64/128, we increase the pseudo-linear approximation by one more rounds and are able to recover 49 key bits.

Table A4 shows the mask that is used in this attack. As with SPECK 32, we can drive our target window $xl_1^{10}(0,1)$ backward from the ciphertext to build the pseudo-linear approximation.

Table A4. Linear trail of SPECK 64 for 9 rounds. (Starting with 0x0000000300000000 backward).

Round	$\lambda_{x^{i+1}}$	$\lambda_{y^{i+1}}$	λ_{x^i}	λ_{y^i}	Cost
1	0x30c03030	0x30f6e836	0x36d80600	0xc6280500	5
2	0x36018000	0x06314030	0x30c03030	0x30f6e836	4
3	0x300c0c00	0x303a0d80	0x36018000	0x06314030	3
4	0x8060006d	0x80500c61	0x300c0c00	0x303a0d80	3
5	0x0303030f	0x6e83630f	0x8060006d	0x80500c61	4
6	0x03000360	0x0c030063	0x0303030f	0x6e83630f	5
7	0x00030300	0x60000303	0x03000360	0x0c030063	3
8	0x00000300	0x00000003	0x00030300	0x60000303	2
9	0x00000003	0x00000000	0x00000300	0x00000003	1

Appendix D. The Partly-Pseudo-Linear Attack on SPECK 96

In SPECK 96 there are two key sizes: 96 bits and 144 bits. With SPECK 96/96, we are able to attack 10 rounds (five rounds using a pseudo-linear approximation and five rounds using linear approximations). Thus, we are able to recover 46 key bits. With SPECK 96/144, we increase the pseudo-linear approximation by one round and increase the linear approximation by one round too. Thus, the total is 12 rounds and we are able to recover 76 key bits.

Table A5 shows the mask that is used in this attack. As with SPECK 32 and SPECK 48, in SPECK 96/96, we can drive our target window $xl_1^5(1,0)$ backward from the ciphertext to build the pseudo-linear approximation. On the other hand, for SPECK 96/144, we increase the attacked rounds by two rounds and the target window is $xl_1^6(0,1)$.

Table A5. Linear trail of SPECK 96 for six rounds. (Starting with 0x000000000003000000000000 backward).

Round	$\lambda_{x^{i+1}}$	$\lambda_{y^{i+1}}$	λ_{x^i}	λ_{y^i}	Cost
1	0x83000060036d	0x803300600c62	0x3300000f0f03	0x5335600c0e83	7
2	0x00030303030c	0x6d800303630f	0x83000060036d	0x803300600c62	6
3	0x000003000360	0x0c0000030063	0x00030303030c	0x6d800303630f	5
4	0x000000030300	0x600000000303	0x000003000360	0x0c0000030063	3
5	0x000000000300	0x000000000003	0x000000030300	0x600000000303	2
6	0x000000000003	0x000000000000	0x000000000300	0x000000000003	1

Appendix E. The Partly-Pseudo-Linear Attack on SPECK 128

In SPECK 128 there are three key sizes: 128 bits, 192 bits, and 256 bits. With SPECK 128/128, we are able to attack 11 rounds (five rounds using a pseudo-linear approximation and 6 rounds using a linear approximations). Thus, we are able to recover 50 key bits. With SPECK 128/192, we increase the pseudo-linear approximation by two more rounds (Total rounds is 13) and are able to recover 122 key bits. With SPECK 128/256, we increase the pseudo-linear approximation by one round (Total rounds is 14) and are able to recover 173 key bits.

Table A6 shows the mask that is used in this attack. As with SPECK 32, we can drive our target window $xl_1^7(0,1)$ backward from the ciphertext to build the pseudo-linear approximation.

Table A6. Linear trail of SPECK 128 for six rounds. (Starting with 0x000000000000000 30000000000000000 backward).

R	$\lambda_{x^{i+1}}$	$\lambda_{y^{i+1}}$	λ_{x^i}	λ_{y^i}	Cost
1	0x8000030000060036d	0x8030000300600c62	0x300303000000f0f00	0x50360303600c0e83	7
2	0x0000000303030300c	0x6d8000000303630f	0x8000030000060036d	0x8030000300600c62	6
3	0x0000000003000360	0x0c00000000030063	0x0000000303030300c	0x6d8000000303630f	5
4	0x0000000000030300	0x6000000000000303	0x0000000003000360	0x0c00000000030063	3
5	0x0000000000000300	0x0000000000000003	0x0000000000030300	0x6000000000000303	2
6	0x0000000000000003	0x0000000000000000	0x0000000000000300	0x0000000000000003	1

References

1. Beaulieu, R.; Shors, D.; Smith, J.; Treatman-Clark, S.; Weeks, B.; Wingers, L. The SIMON and SPECK Families of Lightweight Block Ciphers. *IACR Cryptol. EPrint Arch.* **2013**, *2013*, 404.
2. Beaulieu, R.; Shors, D.; Smith, J.; Treatman-Clark, S.; Weeks, B.; Wingers, L. The SIMON and SPECK lightweight block ciphers. In Proceedings of the 52nd Annual Design Automation Conference, San Francisco, CA, USA, 7–11 June 2015; pp. 175:1–175:6. [CrossRef]
3. McKay, K.A.; Vora, P.L. Analysis of ARX Functions: Pseudo-linear Methods for Approximation, Differentials, and Evaluating Diffusion. *IACR Cryptol. EPrint Arch.* **2014**, *2014*, 895.
4. McKay, K.A. Analysis of ARX Round Functions in Secure Hash Functions. Ph.D. Thesis, The George Washington University, Washington, DC, USA, 2014.
5. McKay, K.A.; Vora, P.L. Pseudo-Linear Approximations for ARX Ciphers: With Application to Threefish. In Proceedings of the Second SHA-3 Candidate Conference, Santa Barbara, CA, USA, 23–24 August 2010; p. 282.
6. Cho, J.Y.; Pieprzyk, J. Multiple Modular Additions and Crossword Puzzle Attack on NLSv2. In Proceedings of the 10th International Conference on Information Security (ISC 2007), Valparaiso, Chile, 9–12 October 2007; Lecture Notes in Computer Science; Garay, J.A., Lenstra, A.K., Mambo, M., Peralta, R., Eds.; Springer: Berlin/Heidelberg, Germany, 2007; Volume 4779, pp. 230–248. [CrossRef]
7. Cho, J.Y.; Pieprzyk, J. Algebraic Attacks on SOBER-t32 and SOBER-t16 without Stuttering. In *Fast Software Encryption, Proceedings of the 11th International Workshop, (FSE 2004), Delhi, India, 5–7 February 2004*; Revised Papers; Lecture Notes in Computer Science; Roy, B.K., Meier, W., Eds.; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3017, pp. 49–64. [CrossRef]
8. Alzakari, S.; Vora, P. Linear and Partly-Pseudo-Linear Cryptanalysis of Reduced-Round SPARX Cipher. *IACR Cryptol. EPrint Arch.* **2020**, *2020*, 978.

9. Bodden, D. Linear Cryptanalysis of Reduced-Round Speck with a Heuristic Approach: Automatic Search for Linear Trails. In Proceedings of the Information Security–21st International Conference (ISC 2018), Guildford, UK, 9–12 September 2018; Lecture Notes in Computer Science; Chen, L., Manulis, M., Schneider, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2018; Volume 11060, pp. 132–150. [CrossRef]
10. Ashur, T.; Bodden, D. Linear Cryptanalysis of Reduced-Round Speck. In Proceedings of the 37th Symposium on Information Theory in the Benelux 2016, Louvain-la-Neuve, Belgium, 19–20 May 2016.
11. Yao, Y.; Zhang, B.; Wu, W. Automatic Search for Linear Trails of the SPECK Family. In Proceedings of the Information Security—18th International Conference (ISC 2015), Trondheim, Norway, 9–11 September 2015; Lecture Notes in Computer Science; Lopez, J., Mitchell, C.J., Eds.; Springer: Berlin/Heidelberg, Germany, 2015; Volume 9290, pp. 158–176. [CrossRef]
12. Fu, K.; Wang, M.; Guo, Y.; Sun, S.; Hu, L. MILP-Based Automatic Search Algorithms for Differential and Linear Trails for Speck. In Proceedings of the Fast Software Encryption—23rd International Conference (FSE 2016), Bochum, Germany, 20–23 March 2016; Revised Selected Papers; Lecture Notes in Computer Science; Peyrin, T., Ed.; Springer: Berlin/Heidelberg, Germany, 2016; Volume 9783, pp. 268–288. [CrossRef]
13. Liu, Y.; Fu, K.; Wang, W.; Sun, L.; Wang, M. Linear cryptanalysis of reduced-round SPECK. *Inf. Process. Lett.* **2016**, *116*, 259–266. [CrossRef]
14. Matsui, M. Linear Cryptanalysis Method for DES Cipher. In *Advances in Cryptology—EUROCRYPT '93, Proceedings of the Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, 23–27 May 1993*; Lecture Notes in Computer Science; Helleseth, T., Ed.; Springer: Berlin/Heidelberg, Germany, 1993; Volume 765, pp. 386–397. [CrossRef]
15. Heys, H.M. A Tutorial on Linear and Differential Cryptanalysis. *Cryptologia* **2002**, *26*, 189–221. [CrossRef]
16. Beaulieu, R.; Shors, D.; Smith, J.; Treatman-Clark, S.; Weeks, B.; Wingers, L. SIMON and SPECK: Block Ciphers for the Internet of Things. *IACR Cryptol. EPrint Arch.* **2015**, *2015*, 585.
17. Song, L.; Huang, Z.; Yang, Q. Automatic Differential Analysis of ARX Block Ciphers with Application to SPECK and LEA. In Proceedings of the Information Security and Privacy—21st Australasian Conference (ACISP 2016), Melbourne, VIC, Australia, 4–6 July 2016; Part II; Lecture Notes in Computer Science; Liu, J.K., Steinfeld, R., Eds.; Springer: Berlin/Heidelberg, Germany, 2016; Volume 9723, pp. 379–394. [CrossRef]
18. HoChang, L.; Seojin, K.; HyungChul, K.; Deukjo, H.; Jaechul, S.; Hong, S. Calculating the Approximate Probability of Differentials for ARX-Based Cipher Using SAT Solver. *J. Korea Inst. Inf. Secur. Cryptol.* **2018**, *28*, 15–24. [CrossRef]
19. Liu, Y.; Witte, G.D.; Ranea, A.; Ashur, T. Rotational-XOR Cryptanalysis of Reduced-round SPECK. *IACR Trans. Symmetric Cryptol.* **2017**, *2017*, 24–36. [CrossRef]
20. Dwivedi, A.D.; Morawiecki, P.; Srivastava, G. Differential Cryptanalysis of Round-Reduced SPECK Suitable for Internet of Things Devices. *IEEE Access* **2019**, *7*, 16476–16486. [CrossRef]