Distinguishing and Key Recovery Attacks on the Reduced-Round SNOW-V and SNOW-Vi[*]

Jin Hoki[1], Takanori Isobe[1,2,3], Ryoma Ito[2], Fukang Liu[1], Kosei Sakamoto[1]

[1] University of Hyogo, Japan.
`takanori.isobe@ai.u-hyogo.ac.jp`
[2] National Institute of Information and Communications Technology, Japan.
`itorym@nict.go.jp`
[3] PRESTO, Japan Science and Technology Agency, Japan.

Abstract. This paper presents distinguishing and key recovery attacks on the reduced-round SNOW-V and SNOW-Vi, which are stream ciphers proposed for standard encryption schemes for the 5G mobile communication system. First, we construct a MILP model to search for integral characteristics using the division property, and find the best integral distinguisher in the 3-, 4-, 5-round SNOW-V, and 5-round SNOW-Vi with time complexities of 2^8, 2^{16}, 2^{48}, and 2^{16}, respectively. Next, we construct a bit-level MILP model to efficiently search for differential characteristics, and find the best differential characteristics in the 3- and 4-round versions. These characteristics lead to the 3-round differential distinguishers for SNOW-V and SNOW-Vi with time complexities of 2^{17} and 2^{12} and the 4-round differential distinguishers for SNOW-V and SNOW-Vi with time complexities of 2^{97} and 2^{39}, respectively. Then, we consider single-bit and dual-bit differential cryptanalysis, which is inspired by the existing study on Salsa and ChaCha. By carefully choosing the IV values and differences, we can construct practical bit-wise differential distinguishers for the 4-round SNOW-V, 4-, and 5-round SNOW-Vi with time complexities of $2^{4.466}$, $2^{1.000}$, and $2^{14.670}$, respectively. Finally, we improve the existing differential attack based on probabilistic neutral bits, which is also inspired by the existing study on Salsa and ChaCha. As a result, we present the best key recovery attack on the 4-round SNOW-V and SNOW-Vi with time complexities of $2^{153.97}$ and $2^{233.99}$ and data complexities of $2^{26.96}$ and $2^{19.19}$, respectively. Consequently, we significantly improve the existing best attacks in the initialization phase by the designers.

Keywords: SNOW · Stream cipher · 5G · Integral attack · Differential attack · Probabilistic Neutral Bits (PNB)

---

1 Introduction

1.1 Background

SNOW-V, which is a new variant of a family of SNOW stream ciphers, was proposed for a standard encryption scheme for the 5G mobile communication system in 2019 by Ekdahl et al. [5]. To achieve the strong security requirements by the 3GPP standardization organization for the 5G system, SNOW-V provides a 256-bit security level against key recovery attacks with a 256-bit key and 128-bit IV, while the claimed security of distinguishing attacks is only 2^{64}, i.e., the length of keystreams is limited to at most 2^{64} and also for a fixed key, the number of different keystreams should be less than 2^{64}.

SNOW-V consists of a Linear Feedback Shift Register (LFSR) and Finite State Machine (FSM). The overall structure of SNOW-V follows the design strategy of SNOW 2.0 and SNOW-3G. It takes advantage of AES-NI and some SIMD operations for efficient implementation in high-end software environments. Each round has two AES-round operations to update the states of the FSM. As a result, SNOW-V achieves very impressive performance in software, e.g., 58 Gbps for a long message, which is almost six times faster than that of SNOW-3G.

A slightly modified version of SNOW-V stream cipher, called SNOW-Vi, was proposed by the same designers in 2021 [6]. The structural differences between SNOW-V and SNOW-Vi are the LFSR update function and the location of the tap T_2. The purpose of this change is to better accommodate a fast software implementation on lower grade CPUs which only supports 128-bit wide SIMD registers. As a result, the increase in software performance is approximately 50% in average, up to 92 Gbps for a long message.

The designers evaluated the security of division-property-based cube, time-memory tradeoff, linear/correlation distinguishing, algebraic, and guess-and-determine attacks on both ciphers [5, 6]. Among them, they found a key recovery attack on the 3-round SNOW-V and SNOW-Vi by division-property-based cube attacks, and concluded that more than four rounds provides sufficient security against these attacks as the division-property-based distinguisher reaches only four rounds of AES [17]. As a third-party evaluation, Jiao et al. proposed a byte-based guess-and-determine attack on the full-round SNOW-V with a time complexity of 2^{406} [11]. After Jiao et al.'s study [11] was reported, the designers further improved the Jiao et al.'s attack, and proposed two guess-and-determine attack on the full-round SNOW-V with time complexities of 2^{384} and 2^{378} [19]. These results improved the original evaluation by the designers [5], but its cost is still much larger than the exhaustive 256-bit key search. Thus, to the best of our knowledge, the best attack on SNOW-V and SNOW-Vi is the 3-round division-property-based cube attack by the designers, respectively.

1.2 Our Contribution

In this study, we present the security analysis of SNOW-V and SNOW-Vi with three attack vectors: integral, differential, and bit-wise differential attacks. These

attacks are well-known attacks for stream ciphers. Nevertheless, the designers did not perform the security evaluations for these important attacks. To fill this gap, we evaluate thorough security against these attacks with state-of-the-art search tools and techniques, and we show that these attacks sufficiently improve the previous best attacks with respect to the attacked number of rounds and attack complexity, as shown in Table 1. The details of our attacks are given as follows.

Integral Attack By using a MILP-aided search method for the division property, we show practical integral distinguishers for the 3-, 4-round SNOW-V, and 5-round SNOW-Vi with time complexities of 2^8, 2^{16}, and 2^{16}, respectively. Furthermore, we find a 5-round integral distinguisher with a time complexity of 2^{48} for the initialization of SNOW-V.

Differential Attack We perform a MILP-aided search for the differential characteristics in the chosen-IV setting where differences are inserted in the IV domain. Specifically, we build a bit-level model for each operation, such as the modular addition, S-box, and linear operations. Besides, we modify the objective function in our study [9] to obtain a more accurate differential probability. As a result of SNOW-V, we improve the time complexities of the 3- and 4-round differential distinguishers from 2^{48} and 2^{103} to 2^{17} and 2^{97}, respectively. For SNOW-Vi, we find the 3- and 4-round differential distinguishers with the time complexities of 2^{12} and 2^{39}, respectively. Although the distinguishing attack on the 4-round SNOW-V exceeds the data limitation of 2^{64}, it is important to improve the understanding of the security of SNOW-V.

Bit-wise Differential Attack We conduct a single-bit and dual-bit differential attack based on the existing study on the reduced-round Salsa and ChaCha as reported by Choudhuri and Maitra [3]. In addition, we analyze the source code of the LFSR update algorithm, and suggest that choosing IVs by limiting the domain should suppress the propagation of differences throughout the internal state of the target ciphers. As a result, we find practical bit-wise differential distinguishers for the 4-round SNOW-V, 4-, and 5-round SNOW-Vi with time complexities of $2^{4.466}$, $2^{1.000}$, and $2^{14.670}$, respectively. No study has been reported on applying the bit-wise differential attack to LFSR-based stream ciphers; thus, we have demonstrated the effectiveness of the bit-wise differential attack on LFSR-based stream ciphers.

Key Recovery Attack We apply the bit-wise differential attack based on probabilistic neutral bits (PNB), which was proposed by Aumasson et al. [2], to a key recovery attack on the target ciphers. To apply an existing attack, it is necessary to perform the backwards computation in the target ciphers, but it is difficult to perform this in LFSR-based stream ciphers. To solve this problem, we replace all the backwards computations in the existing attack procedure with forwards computations. As a result, we present a key recovery attack on the 4-round SNOW-V and SNOW-Vi with time complexities of $2^{153.97}$ and $2^{233.99}$ and

3

Table 1. Summary of our results.

Cipher	Attack type	Rounds	Data	Time	Reference
SNOW-V	Integral/Distinguisher	3	$2^{8.00}$	$2^{8.00}$	Section 3
	Integral/Distinguisher	4	$2^{16.00}$	$2^{16.00}$	Section 3
	Integral/Distinguisher	5	$2^{48.00}$	$2^{48.00}$	Section 3
	Differential/Distinguisher	3	$2^{17.00}$	$2^{17.00}$	Section 4
	Differential/Distinguisher	4	$2^{97.00}$	$2^{97.00}$	Section 4
	Differential Bias/Distinguisher	4	$2^{4.47}$	$2^{4.47}$	Section 5
	Differential Bias/Distinguisher	5	$2^{36.87}$	$2^{36.87}$	Section 5
SNOW-Vi	Integral/Distinguisher	5	$2^{16.00}$	$2^{16.00}$	Section 3
	Differential/Distinguisher	3	$2^{12.00}$	$2^{12.00}$	Section 4
	Differential/Distinguisher	4	$2^{39.00}$	$2^{39.00}$	Section 4
	Differential Bias/Distinguisher	4	$2^{1.00}$	$2^{1.00}$	Section 5
	Differential Bias/Distinguisher	5	$2^{14.67}$	$2^{14.67}$	Section 5
	Differential Bias/Distinguisher	6	$2^{38.19}$	$2^{38.19}$	Section 5
SNOW-V	Cube/Key Recovery	3	$2^{15.00}$	$2^{255.00}$	[5]
	Differential Bias/Key Recovery	4	$2^{26.96}$	$2^{153.97}$	Section 6
SNOW-Vi	Cube/Key Recovery	3	$2^{4.00}$	$2^{255.00}$	[6]
	Differential Bias/Key Recovery	4	$2^{7.94}$	$2^{233.99}$	Section 6

data complexities of $2^{26.96}$ and $2^{19.19}$, respectively. To the best of our knowledge, our attack is the best key recovery attack on the reduced-round SNOW-V and SNOW-Vi.

1.3 Organization of the Paper

The rest of the paper is organized as follows. In Section 2, we briefly describe the specifications of SNOW-V and SNOW-Vi. In Section 3, we show the MILP model for searching integral characteristics and provide integral distinguishers for the 3-, 4-, 5- round SNOW-V, and 5-round SNOW-Vi. In Section 4, we show the MILP model for searching differential characteristics and provide differential distinguishers for the 3- and 4-round versions. In Section 5, we introduce the existing cryptanalysis method for bit-wise differential cryptanalysis and present the efficient chosen-IV technique. We then provide bit-wise differential distinguishers for the 4-, 5-round SNOW-V, 4-, 5-, and 6-round SNOW-Vi. In Section 6, we describe our improvements to the existing differential attack and present the best key recovery attack on the 4-round versions. Finally, Section 7 concludes the paper.

2 Specifications of SNOW-V and SNOW-Vi

2.1 Structure of SNOW-V

The overall structure of SNOW-V is shown in Figure 1. It consists of a Linear Feedback Shift Register (LFSR) part and Finite State Machine (FSM) part.

4
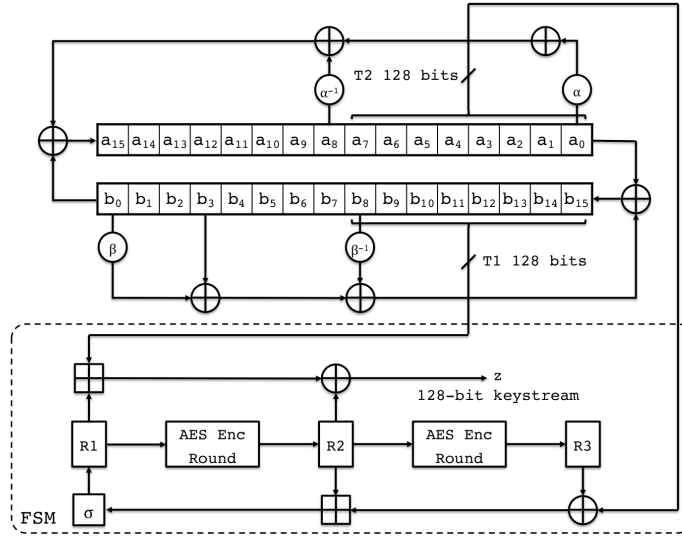
Fig. 1. Overall structure of SNOW-V.

The LFSR part takes a circular construction consisting of two shift registers called LFSR-A and LFSR-B, both involving 16 cells with each cell size of 16 bits denoted by $a_{15}, \ldots, a_0$ and $b_{15}, \ldots, b_0$, respectively. Each cell represents an element in $\mathbb{F}_2^{16}$, and the elements of LFSR-A and LFSR-B are generated by the following polynomials in $\mathbb{F}_2[x]$:

$$g^A(x) = x^{16} + x^{15} + x^{12} + x^{11} + x^8 + x^3 + x^2 + x + 1, \tag{1}$$

$$g^B(x) = x^{16} + x^{15} + x^{14} + x^{11} + x^8 + x^6 + x^5 + x + 1. \tag{2}$$

Let $\alpha \in \mathbb{F}_{2^{16}}^A$ be a root of $g^A(x)$ and $\beta \in \mathbb{F}_{2^{16}}^B$ be a root of $g^B(x)$. At time $t \geq 0$, the LFSRs update sequences $(a_{15}^{(t)}, \ldots, a_0^{(t)})$ and $(b_{15}^{(t)}, \ldots, b_0^{(t)})$ using the following expressions:

$$a_{15}^{(t+1)} = b_0^{(t)} + \alpha a_0^{(t)} + a_1^{(t)} + \alpha^{-1} a_8^{(t)} \mod g^A(\alpha), \tag{3}$$

$$a_i^{(t+1)} = a_{i+1}^{(t)}, \tag{4}$$

$$b_{15}^{(t+1)} = a_0^{(t)} + \beta b_0^{(t)} + a_3^{(t)} + \beta^{-1} b_8^{(t)} \mod g^B(\beta), \tag{5}$$

$$b_i^{(t+1)} = b_{i+1}^{(t)}, \tag{6}$$

for $i = 0, \ldots, 14$. The LFSRs update the internal state eight times in a single step, i.e., 16 cells of the total 32 cells in the LFSR part can be updated in a single step, and the two taps $T1$ and $T2$ will have the following new values:

$$T1^{(t)} = (b_{15}^{(8t)}, \ldots, b_8^{(8t)}), \tag{7}$$

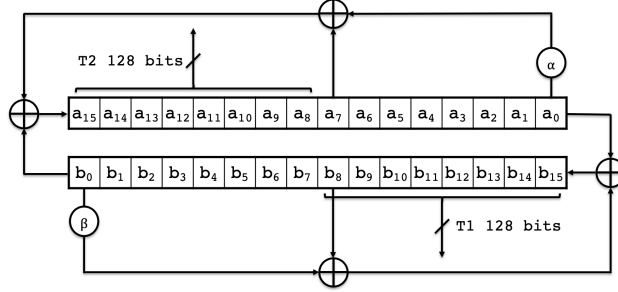$$T2^{(t)} = (a_7^{(8t)}, \ldots, a_0^{(8t)}). \tag{8}$$

Fig. 2. LFSR structure of SNOW-Vi.

The FSM part takes the two taps, $T1$ and $T2$, from the LFSR part as the inputs and generates a 128-bit keystream block $z^{(t)}$ at time $t \geq 0$ as the output. It consists of three 128-bit registers $R1$, $R2$, and $R3$. The symbol $\oplus$ denotes a bit-wise XOR operation, and the symbol $\boxplus_{32}$ denotes parallel application of four additions modulo 2^{32}. The four 32-bit parts of the 128-bit words are added with carry, but the carry does not propagate from a lower 32-bit word to a higher one. At time $t \geq 0$, the FSM first outputs the keystream block, $z^{(t)}$, using the following expression:

$$z^{(t)} = (R1^{(t)} \boxplus_{32} T1^{(t)}) \oplus R2^{(t)}. \tag{9}$$

Then, registers $R2$ and $R3$ are updated throughout a full AES encryption round function as `SubBytes`, `ShiftRows`, `MixColumns`, and `AddRoundKey`, which are denoted by $\texttt{AES}^R(IN, KEY)$ with a 128-bit input block IN and a roundkey KEY. The three registers are updated by the following expressions:

$$R1^{(t+1)} = \sigma(R2^{(t)} \boxplus_{32} (R3^{(t)} \oplus T2^{(t)})), \tag{10}$$
$$R2^{(t)} = \texttt{AES}^R(R1^{(t)}, 0), \tag{11}$$
$$R3^{(t)} = \texttt{AES}^R(R2^{(t)}, 0), \tag{12}$$

where σ is a byte-oriented permutation given by

$$\sigma = [0, 4, 8, 12, 1, 5, 9, 13, 2, 6, 10, 14, 3, 7, 11, 15]. \tag{13}$$

2.2 Structure of SNOW-Vi

The overall structure of SNOW-Vi is almost the same as that of SNOW-V, with the only difference in the LFSR update function and the tap T_2 moved to the upper half of LFSR-A. The LFSR structure of SNOW-Vi is shown in Figure 2. The elements of LFSR-A and LFSR-B are generated by the following polynomials in $\mathbb{F}_2[x]$:

$$g^A(x) = x^{16} + x^{14} + x^{11} + x^9 + x^6 + x^5 + x^3 + x^2 + 1, \tag{14}$$

$$g^B(x) = x^{16} + x^{15} + x^{14} + x^{11} + x^{10} + x^7 + x^2 + x + 1. \tag{15}$$

At time $t \geq 0$, the LFSRs update sequences $(a_{15}^{(t)}, \ldots, a_0^{(t)})$ and $(b_{15}^{(t)}, \ldots, b_0^{(t)})$ using the following expressions:

$$a_{15}^{(t+1)} = b_0^{(t)} + \alpha a_0^{(t)} + a_7^{(t)} \mod g^A(\alpha), \tag{16}$$

$$a_i^{(t+1)} = a_{i+1}^{(t)}, \tag{17}$$

$$b_{15}^{(t+1)} = a_0^{(t)} + \beta b_0^{(t)} + b_8^{(t)} \mod g^B(\beta), \tag{18}$$

$$b_i^{(t+1)} = b_{i+1}^{(t)}, \tag{19}$$

for $i = 0, \ldots, 14$. After updating the LFSRs, the tap T_2 will have the new value, such as $T2^{(t)} = (a_{15}^{(8t)}, \ldots, a_8^{(8t)})$.

2.3 Initialization

Let $K = (k_{15}, \ldots, k_0)$ denote a 256-bit key and $IV = (iv_7, \ldots, iv_0)$ denote a 128-bit initialization vector (IV), where each k_i and iv_j are 16-bit vectors for $0 \leq i \leq 15$ and $0 \leq j \leq 7$, respectively. The initialization begins with loading the key and IV into the LFSRs and setting zero into the three registers using the following expressions:

$$(a_{15}, \ldots, a_0) = (k_7, \ldots, k_0, iv_7, \ldots, iv_0), \tag{20}$$

$$(b_{15}, \ldots, b_0) = (k_{15}, \ldots, k_8, 0, \ldots, 0), \tag{21}$$

$$R1 = 0, \ R2 = 0, \ R3 = 0. \tag{22}$$

The initialization consists of r steps ($r = 16$ in the original version), where the structure is updated in the same way as in the keystream generation, with the exception that the 128-bit keystream block z is not an output but is XORed into the LFSR-A to positions $(a_{15}, \ldots, a_8)$ in every step. Additionally, at the two last steps of the initialization, the 256-bit key is loaded into the register $R1$ using the following expressions:

$$R1^{(r-2)} = R1^{(r-2)} \oplus (k_7, \ldots, k_0), \tag{23}$$

$$R1^{(r-1)} = R1^{(r-1)} \oplus (k_{15}, \ldots, k_8), \tag{24}$$

where time $t = r - 1$ denotes the last step of the initialization.

The designers limited the length of the keystream to a maximum of 2^{64} bits for a single key-IV pair and the number of different IVs to a maximum of 2^{64} for each key.

3 MILP-aided Integral Distinguisher

In this section, we explore the security of SNOW-V and SNOW-Vi against integral attacks. To efficiently search for integral distinguishers in the initialization phase of SNOW-V and SNOW-Vi, we exploit the division property proposed by Todo [16]. Specifically, we utilize the MILP-based method [18] to evaluate the propagation of the bit-based division property [17].

3.1 The MILP Model

In this part, we describe how to construct the linear inequalities to model the propagation of the division property for SNOW-V and SNOW-Vi. First, we will show the constraints for the propagation of the bit-based division property through COPY, XOR, and AND operations based on the work by Xiang et al. [18]. Then, we elaborate the MILP model for SNOW-V and SNOW-Vi based on these constraints. Since SNOW-V and SNOW-Vi are basically the same structure without the LFSR update function and the place of tap T_2, we mainly describe how to construct an MILP model for SNOW-V.

To find an integral distinguisher with the division property with MILP, we do not need to optimize the objective function. Instead, we only need to confirm whether the constructed MILP model is feasible or not, because we search the properties such that the output is balanced or not by bit-wise. If it is infeasible, an integral distinguisher can be obtained.

Xiang et al. first proposed the modeling method [18] for the propagation of the bit-based division property through COPY, XOR, and AND operations. Then, Sun et al. generalized these models [14] as specified below, which will be the components in our MILP model for SNOW-V and SNOW-Vi.

$$\text{MILP Model of COPY [14]} : \begin{cases} \mathcal{M}._{var} \leftarrow a, b_1, \ldots, b_m \text{ as binary.} \\ \mathcal{M}._{con} \leftarrow a + b_1 + \cdots + b_m = 0. \end{cases}$$

$$\text{MILP Model of XOR [14]} : \begin{cases} \mathcal{M}._{var} \leftarrow a_1, \ldots, a_m, b \text{ as binary.} \\ \mathcal{M}._{con} \leftarrow a_1 + \cdots + a_m + b = 0. \end{cases}$$

$$\text{MILP Model of AND [18]} : \begin{cases} \mathcal{M}._{var} \leftarrow a_1, a_2, b \text{ as binary.} \\ \mathcal{M}._{con} \leftarrow b - a_1 \geq 0, \\ \mathcal{M}._{con} \leftarrow b - a_2 \geq 0, \\ \mathcal{M}._{con} \leftarrow b - a_1 - a_2 \leq 0. \end{cases}$$

The pseudo code of our MILP model for SNOW-V is displayed in Algorithm 1, where R denotes the number of rounds in the initialization phase and the explanations for load, funcADD, funcAES, sigma, and funcLFSR are given below.

load. K and IV are loaded into internal states.

funcADD. This function is a model for the 32-bit modular addition. We use the modeling method proposed by Sun et al. [15] with COPY, XOR, and AND.

funcAES. This function consists of SubBytes, ShiftRow, MixColumns, and AddRoundKey of AES. For the modeling of the S-box, we use the modeling method proposed in [18]. Logic Friday [4] is utilized to generate the constraints for the S-box. Thus, we obtain 241 linear inequalities to model the S-box of AES. For the modeling of MixColumns, we use the modeling method proposed in [14]. Specifically, the 4×4 MDS matrix over the filed $\mathbb{F}_2^8$ is converted to a 32×32 binary matrix over the field, $\mathbb{F}_2$ [13]. Then, we construct the model for MixColumn with COPY and XOR. Thus, 64 linear inequalities can be used to model the MDS matrix used in AES.

Algorithm 1 MILP model of division property for SNOW-V

1: **procedure** SNOWVcore(round R)
2:   Prepare an empty MILP model $\mathcal{M}$
3:   $\mathcal{M}.var \leftarrow S_j^0$ for $j \in \{0, \ldots, 511\}$ and $R1_j^0, R2_j^0, R3_j^0$ for $j \in \{0, \ldots, 127\}$
4:   $(\mathcal{M}, \boldsymbol{S}^0, \boldsymbol{R2}^0, \boldsymbol{R3}^0, \boldsymbol{R1}^0) = \texttt{load}(\mathcal{M}, \boldsymbol{K}, \boldsymbol{IV})$
5:   **for** $r = 0$ to R **do**
6:     $(\mathcal{M}, \boldsymbol{T2}^r, S_{128,\ldots,255}^r) = \texttt{COPY}(\mathcal{M}, S_{128,\ldots,255}^r)$
7:     $(\mathcal{M}, \boldsymbol{T1}^r, S_{256,\ldots,383}^r) = \texttt{COPY}(\mathcal{M}, S_{256,\ldots,383}^r)$
8:     $(\mathcal{M}, \boldsymbol{X}_{R1}^r, \boldsymbol{Y}_{R1}^r) = \texttt{COPY}(\mathcal{M}, \boldsymbol{R1}^r)$
9:     $(\mathcal{M}, \boldsymbol{X}_{R2}^r, \boldsymbol{Y}_{R2}^r, \boldsymbol{W}_{R2}^r) = \texttt{COPY}(\mathcal{M}, \boldsymbol{R2}^r)$
10:     $(\mathcal{M}, \boldsymbol{U}^r) = \texttt{funcADD}(\mathcal{M}, \boldsymbol{T1}^r, \boldsymbol{X}_{R1}^r)$
11:     $(\mathcal{M}, \boldsymbol{Z}^r) = \texttt{XOR}(\mathcal{M}, \boldsymbol{U}^r, \boldsymbol{X}_{R2}^r)$
12:     $(\mathcal{M}, \boldsymbol{V}^r) = \texttt{XOR}(\mathcal{M}, \boldsymbol{T2}^r, \boldsymbol{R3}^r)$
13:     $(\mathcal{M}, \boldsymbol{tmp}^r) = \texttt{funcADD}(\mathcal{M}, \boldsymbol{V}^r, \boldsymbol{W}_{R2}^r)$
14:     $(\mathcal{M}, \boldsymbol{R3}^{r+1}) = \texttt{funcAES}(\mathcal{M}, \boldsymbol{Y}_{R2}^r)$
15:     $(\mathcal{M}, \boldsymbol{R2}^{r+1}) = \texttt{funcAES}(\mathcal{M}, \boldsymbol{Y}_{R1}^r)$
16:     $(\mathcal{M}, \boldsymbol{R1}^{r+1}) = \texttt{sigma}(\mathcal{M}, \boldsymbol{tmp}^r)$
17:     **for** $i = 0$ to 7 **do**
18:       $(\mathcal{M}, \boldsymbol{S}^{r,i+1}) = \texttt{funcLFSR}(\mathcal{M}, \boldsymbol{S}^{r,i})$
19:     **if** $r \neq R$ **then**
20:       $(\mathcal{M}, \boldsymbol{S}_{0,\ldots,127}^{r+1}) = \texttt{XOR}(\mathcal{M}, \boldsymbol{S}_{0,\ldots,127}^{r,8}, \boldsymbol{Z}^r)$
21:   **for** $j = 0$ to 512 **do**
22:     $\mathcal{M}.con \leftarrow S_j^{R+1} = 0$
23:   **for** $j = 0$ to 127 **do**
24:     $\mathcal{M}.con \leftarrow R1_j^{R+1} = R2_j^{R+1} = R3_j^{R+1} = 0$
25:   $\mathcal{M}.con \leftarrow \sum_{j=0}^{127} Z_j^R = 1$
---

sigma. This function is used to permute the state in a byte-wise way as described in Section 2.

funcLFSR. There are the operations of α, α^{-1}, β, β^{-1}, and XOR. It is a linear transformation; thus, the division property of the input and the output are constant. Hence, we can use the method from Sun et al. [14], and α, α^{-1}, β, and β^{-1} are each represented with a 16×16 matrix over field $\mathbb{F}_2$, and we obtain 64 linear inequalities. The funcLFSR in SNOW-Vi is constructed by the same method as that of SNOW-V.

Our MILP model for SNOW-Vi is almost the same as Algorithm 1. To construct an MILP model for SNOW-Vi, we need to change line 6 in Algorithm 1 into "$(\mathcal{M}, \boldsymbol{T2}^r, S_{0,\ldots,127}^r) = \texttt{COPY}(\mathcal{M}, S_{128,\ldots,255}^r)$" and funcLFSR into that of SNOW-Vi.

3.2   Our Search and Results

Since there are a total of 2^{128} patterns for IV on both SNOW-V and SNOW-Vi, it is computationally infeasible to take all of them into account when searching for integral distinguishers. Thus, we use a 3-step approach to efficiently find the

Table 2. 3-Round Integral Distinguisher of SNOW-V

iv_7	cccccccc cccccccc
iv_6	cccccccc cccccccc
iv_5	cccccccc cccccccc
iv_4	cccccccc cccccccc
iv_3	aaaaaaaa cccccccc
iv_2	cccccccc cccccccc
iv_1	cccccccc cccccccc
iv_0	cccccccc cccccccc
z	uuuuuuuu uuuuuuuu uuuuuuuu bbbbbbbb
	uuuuuuuu uuuuuuuu uuuuuuuu bbbbbbbb
	uuuuuuuu uuuuuuuu bbbbbbbb bbbbbbbb
	uuuuuuuu uuuuuuuu bbbbbbbb bbbbbbbb

integral distinguisher for SNOW-V and SNOW-Vi. As an explanation of our method, a, c, b, and u represent an active bit, a constant bit, a balanced bit, and an unknown bit, respectively. In addition, $\mathcal{A}$, $\mathcal{C}$, $\mathcal{B}$, and $\mathcal{U}$ denote an active byte, a constant byte, a balanced byte, and an unknown byte, respectively. Our search used Gurobi optimization 9.0 [10] as the solver with a 48-core Intel(R) Xeon(R) Platinum 8260 CPU @ 2.40GHz for our experiments.

Step 1. We try to find the longest integral distinguisher by setting the 128-bit IV as all $\mathcal{A}$.

Step 2. To reduce the data complexity, we consider the case where there is at least one byte in IV assigned to $\mathcal{C}$ and at least one byte assigned to $\mathcal{A}$. When 16-byte input is all $\mathcal{A}$, it is the same as Step 1. Also, when 16-byte input is all $\mathcal{C}$, the outputs becomes constants. Thus, these two patterns can be omitted. As a result, there are $2^{16} - 2$ such patterns in total.

Step 3. We utilize the method [8] to reduce the data complexity. In [8], a is only assigned to the MSB of each byte. First, we consider the case when there is only one active bit and the total number of such patterns is $\binom{16}{1}$. Then, we increase the number of a if we can find an integral distinguisher, i.e., consider the case when there are $2, 3, 4, \ldots, 16$ active bits because IV is a 16-byte value. Thus, a total of $2^{16} - 1$ patterns is taken into account in our search.

Results for SNOW-V Our search found integral distinguishers in 3- and 4-round distinguishers with time complexities of 2^8 and 2^{16}, as shown in Tables 2 and 3. Moreover, we can find a 5-round integral distinguisher for the initialization phase of SNOW-V, as shown in Table 4. Specifically, when iv_7, iv_6, iv_4 and iv_0 is constant, the least significant byte of iv_2 and iv_1 is constant, and the remaining bytes of IV take all the possible 2^{48} values, we can compute the sum of the keystreams, z, generated by these 2^{48} different IV; thus, the sum in each of the least two significant bits of z is always zero.

Table 3. 4-Round Integral Distinguisher of SNOW-V

iv_7	aaaaaaaa cccccccc
iv_6	cccccccc cccccccc
iv_5	aaaaaaaa cccccccc
iv_4	cccccccc cccccccc
iv_3	cccccccc cccccccc
iv_2	cccccccc cccccccc
iv_1	cccccccc cccccccc
iv_0	cccccccc cccccccc
z	uuuuuuuu uuuuuuuu uuuuuuuu uuuuuuuu
	uuuuuuuu uuuuuuuu uuuuuuuu uuuuuuuu
	uuuuuuuu uuuuuuuu uuuuuuuu bbbbbbbb
	uuuuuuuu uuuuuuuu uuuuuubb bbbbbbbb

Table 4. 5-Round Integral Distinguisher of SNOW-V

iv_7	cccccccc cccccccc
iv_6	cccccccc cccccccc
iv_5	aaaaaaaa aaaaaaaa
iv_4	cccccccc cccccccc
iv_3	aaaaaaaa aaaaaaaa
iv_2	aaaaaaaa cccccccc
iv_1	aaaaaaaa cccccccc
iv_0	cccccccc cccccccc
z	uuuuuuuu uuuuuuuu uuuuuuuu uuuuuuuu
	uuuuuuuu uuuuuuuu uuuuuuuu uuuuuuuu
	uuuuuuuu uuuuuuuu uuuuuuuu uuuuuuuu
	uuuuuuuu uuuuuuuu uuuuuuuu uuuuuubb

Results for SNOW-Vi Our search found integral distinguishers in 5-round with the time complexity of 2^{16} as shown in Table 5. This is the same result as that of SNOW-V in terms of the number of rounds, however, it should be mentioned that the time complexity is reduced from 2^{48} to 2^{16} compared to SNOW-V.

4 MILP-aided Differential Distinguisher

In this section, we describe our investigation of the resistance of SNOW-V and SNOW-Vi against differential attacks.Specifically, we focus on the initialization phase and our aim is to find differential characteristics with a probability higher than 2^{-128} using a MILP-based method [1, 7] as the IV size where differences of 128 bits can be inserted.

According to the specification of SNOW-V and SNOW-Vi, it can be observed that there are 32 AES S-boxes and 8 modular additions (modulo 2^{32}) used for

Table 5. 5-Round Integral Distinguisher of SNOW-Vi

iv_7	cccccccc cccccccc
iv_6	cccccccc cccccccc
iv_5	cccccccc cccccccc
iv_4	cccccccc cccccccc
iv_3	cccccccc cccccccc
iv_2	cccccccc cccccccc
iv_1	aaaaaaaa aaaaaaaa
iv_0	cccccccc cccccccc
z	uuuuuuuu uuuuuuuu uuuuuuuu uuuuuuuu
	uuuuuuuu uuuuuuuu uuuuuuuu uuuuuuuu
	uuuuuuuu uuuuuuuu uuuuuuuu uuuuuuuu
	uuuuuuuu uuuuuuuu uuuuuuuu uuuuuuub

the 1-round state update, which are the only components where the difference transitions are probabilistic.

4.1 The MILP Model

Here, we explain the details of our MILP modeling for searching differential characteristics of SNOW-V and SNOW-Vi. Similarly to the case of the integral attack, we mainly describe how to construct an MILP model for SNOW-V.

Throughout this paper, $\mathcal{M}.var$, $\mathcal{M}.con$, and $\mathcal{M}.obj$ represent the variables, the constraints and the objective function in the MILP model, respectively.

Because the operations used in SNOW-V and SNOW-Vi include XOR, Sub-Bytes, ShiftRows, MixColumns, modular addition, α, α^{-1}, β, β^{-1}, and sigma (the byte-wise permutation), to construct an accurate model to describe the bit-wise difference in propagation using these components, it is necessary to construct the corresponding linear inequalities for each of them.

XOR. The following linear inequalities can be used to model $x_2 = x_0 \oplus x_1$

$$\text{MILP Model of XOR} : \begin{cases} x_0, x_1, x_2 \text{ as binary.} \\ -x_0 - x_1 - x_2 \geq -2, \\ -x_0 + x_1 + x_2 \geq 0, \\ x_0 - x_1 + x_2 \geq 0, \\ x_0 + x_1 - x_2 \geq 0. \end{cases}$$

load. K and IV are loaded into internal states.

funcAES. This function consists of the SubBytes, ShiftRows, MixColumns and AddRoundKey of AES. As proposed in ref. [1], we utilize Logic Friday [4] to automatically generate the linear inequalities for the AES S-box. There are a total of 8302 linear inequalities needed to describe the difference distribution

---

Algorithm 2 MILP model of differential characteristics for SNOW-V

1: **procedure** SNOWVcore(round R)
2:     Prepare an empty MILP model $\mathcal{M}$
3:     $\mathcal{M}.var \leftarrow S_j^0$ for $j \in \{0, \ldots, 511\}$ and $R1_j^0, R2_j^0, R3_j^0$ for $j \in \{0, \ldots, 127\}$
4:     $(\mathcal{M}, \boldsymbol{S}^0, \boldsymbol{R2}^0, \boldsymbol{R3}^0, \boldsymbol{R1}^0) = \texttt{load}(\mathcal{M}, \boldsymbol{K}, \boldsymbol{IV})$
5:     **for** $r = 0$ to $R - 1$ **do**
6:         $(\mathcal{M}, \boldsymbol{U}^r) = \texttt{funcADD}(\mathcal{M}, S_{256,\ldots,383}^r, \boldsymbol{R1}^r)$
7:         $(\mathcal{M}, \boldsymbol{Z}^r) = \texttt{XOR}(\mathcal{M}, \boldsymbol{U}^r, \boldsymbol{R2}^r)$
8:         $(\mathcal{M}, \boldsymbol{V}^r) = \texttt{XOR}(\mathcal{M}, S_{128,\ldots,255}^r, \boldsymbol{R3}^r)$
9:         $(\mathcal{M}, \boldsymbol{tmp}^r) = \texttt{funcADD}(\mathcal{M}, \boldsymbol{V}^r, \boldsymbol{R2}^r)$
10:        $(\mathcal{M}, \boldsymbol{R2}^{r+1}) = \texttt{funcAES}(\mathcal{M}, \boldsymbol{R1}^r)$
11:        $(\mathcal{M}, \boldsymbol{R3}^{r+1}) = \texttt{funcAES}(\mathcal{M}, \boldsymbol{R2}^r)$
12:        $(\mathcal{M}, \boldsymbol{R1}^{r+1}) = \texttt{sigma}(\mathcal{M}, \boldsymbol{tmp}^r)$
13:        **for** $i = 0$ to 7 **do**
14:            $(\mathcal{M}, \boldsymbol{S}^{r,i+1}) = \texttt{funcLFSR}(\mathcal{M}, \boldsymbol{S}^{r,i})$
15:        $(\mathcal{M}, \boldsymbol{S}_{0,\ldots,127}^{r+1}) = \texttt{XOR}(\mathcal{M}, \boldsymbol{S}_{0,\ldots,127}^{r,8}, \boldsymbol{Z}^r)$
16:    $(\mathcal{M}, \boldsymbol{U}^R) = \texttt{funcADD}(\mathcal{M}, S_{256,\ldots,383}^R, \boldsymbol{R1}^R)$
17:    $(\mathcal{M}, \boldsymbol{Z}^R) = \texttt{XOR}(\mathcal{M}, \boldsymbol{U}^R, \boldsymbol{R2}^R)$
18:    $\mathcal{M}.obj \leftarrow Minimize\ (DCP)$

---

table of the AES S-box. Because it is a linear transform, we could write the $(4{\times}4)$ MDS matrix as a $(32{\times}32)$ binary matrix. Using this method, modeling MixColumns is equivalent to modeling several $\oplus$ operations.

funcADD. As proposed in ref. [7], we obtain 407 linear inequalities to model the 32-bit modular addition.

sigma. This function is used to permute the state in a byte-wise way, as described in Section 2.

funcLFSR. This function consists of α, α^{-1}, β, β^{-1}, and XOR. Because α, α^{-1}, β, and β^{-1} are all linear transformations as well, we can derive the equivalent (16×16) binary matrix for all of them, which can be simply modeled by considering the linear inequalities for the $\oplus$ operation. The funcLFSR in SNOW-Vi is constructed by the same method as that of SNOW-V.

To search for the best differential characteristic, we minimize the objective function, as follows:

$$\sum_{r=0}^{R-1} \left(7 \sum_{m=0}^{31} A_m^r + \sum_{m=0}^{7} \sum_{n=1}^{31} M_m^r[n] \right) + \sum_{m=0}^{4} \sum_{n=1}^{31} M_m^R[n].$$

However, we do not take probabilities of $T1^{R-2} \boxplus R1^{R-2}$, $T1^{R-1} \boxplus R1^{R-1}$, and $AES(R2^{R-1})$ into account. It is because that these non-linear functions do not affect to the keystream.

A_m^r denotes the variable of S-box input, and $M_m^r[n] = \neg eq(\alpha_m^r[n], \beta_m^r[n], \gamma_m^r[n])$, i.e., $eq(\alpha_m^r[n], \beta_m^r[n], \gamma_m^r[n]) = 1$ is the same as $\alpha_m^r[n] = \beta_m^r[n] = \gamma_m^r[n]$. $\alpha_m^r[n]$ and $\beta_m^r[n]$ denote the variables of modular addition inputs, and $\gamma_m^r[n]$ denotes the variables of modular addition output, and, $\alpha_m^r[0], \beta_m^r[0], \gamma_m^r[0]$ are the most

Table 6. Differential characteristic probability (DCP) for the reduced initialization round of SNOW-V.

Rounds	1	2	3	4
Optimal DCP (whole IV space)	1	2^{-8}	2^{-17}	–
Optimal DCP (hamming weight of IV difference is one)	1	2^{-8}	2^{-17}	2^{-97}

Table 7. The differential characteristic for 3-initialization rounds of SNOW-V.

Input: IV	00000000000000000000000000000020	
$a_{15}^0,\ldots,a_0^0$	0000000000000000000000000000000000000000000000000000000000000020	
$b_{15}^0,\ldots,b_0^0$	0000000000000000000000000000000000000000000000000000000000000000	
$R1^0$	00000000000000000000000000000000	1
$R2^0$	00000000000000000000000000000000	
$R3^0$	00000000000000000000000000000000	
$a_{15}^1,\ldots,a_0^1$	0000000000000000000000000000000400000000000000000000000000000000	
$b_{15}^1,\ldots,b_0^1$	0000000000000000000000000000000200000000000000000000000000000000	
$R1^1$	00000000000000000000000000000020	2^{-1}
$R2^1$	00000000000000000000000000000000	
$R3^1$	00000000000000000000000000000000	
$a_{15}^2,\ldots,a_0^2$	0040000000000000000000000000000600000000000000000000000000000040	
$b_{15}^2,\ldots,b_0^2$	0000000000200000000000000000000100000000000000000000000000000020	
$R1^2$	00000000000000000000000000000000	2^{-7}
$R2^2$	00000000000000000000000060202040	
$R3^2$	00000000000000000000000000000000	
$a_{15}^3,\ldots,a_0^3$	004000000020000000000000602020c0004000000000000000000000000060	
$b_{15}^3,\ldots,b_0^3$	0000000000000000000000000000008000000000020000000000000000000010	
$R1^3$	00000000000000000000000000000000	2^{-5}
$R2^3$	00000000000000000000000000000000	
$R3^3$	959531a460c0a06080c0404030101020	
Output: z^3	00000020000000200000002000000008	2^{-4}

significant bit, respectively. We consider the differential probability of the modular addition according to [7], and we consider that the differential probability of AES S-box with 2^{-7} because we consider the worst case in the S-box. Algorithm 2 shows the MILP model of differential characteristics for SNOW-V.

Similarly to the case of the integral attack, we can construct an MILP model for SNOW-Vi based on that of SNOW-V with small changes. To construct an MILP model for SNOW-Vi, we need to change line 8 in Algorithm 2 into "$(\mathcal{M}, \boldsymbol{V}) = \texttt{XOR}(\mathcal{M}, S_{0,\ldots,127}^r, \boldsymbol{R3})$" and `funcLFSR` into that of SNOW-Vi. As the same reason of the case of SNOW-V, we do not take probabilities of $T1^{R-1} \boxplus R1^{R-1}$ and $AES(R2^{R-1})$ into account in the objective function.

4.2 Our Search and Results

We search that assuming the differential characteristics are independent of each round. In our search, a difference will only be inserted in IV, i.e., we do not consider related-key differential characteristics. We conduct this search on a

Table 8. The differential characteristic for 4-initialization rounds of SNOW-V.

Input : IV	00000000000000000000000000000020	
$a_{15}^0, \dots, a_0^0$	00000000000000000000000000000000000000000000000000000000000000020	
$b_{15}^0, \dots, b_0^0$	00000000000000000000000000000000000000000000000000000000000000000	
$R1^0$	00000000000000000000000000000000	1
$R2^0$	00000000000000000000000000000000	
$R3^0$	00000000000000000000000000000000	
$a_{15}^1, \dots, a_0^1$	00000000000000000000000000000040000000000000000000000000000000000	
$b_{15}^1, \dots, b_0^1$	00000000000000000000000000000020000000000000000000000000000000000	
$R1^1$	00000000000000000000000000000020	2^{-1}
$R2^1$	00000000000000000000000000000000	
$R3^1$	00000000000000000000000000000000	
$a_{15}^2, \dots, a_0^2$	00400000000000000000000000000002000000000000000000000000000000040	
$b_{15}^2, \dots, b_0^2$	00000000020000000000000000000001000000000000000000000000000000020	
$R1^2$	00000000000000000000000000000000	2^{-8}
$R2^2$	00000000000000000000000060202040	
$R3^2$	00000000000000000000000000000000	
$a_{15}^3, \dots, a_0^3$	0000000000600000000000007$fdfdf$00004000000000000000000000000000020	
$b_{15}^3, \dots, b_0^3$	00000000000000000000000000000080000000000200000000000000000000010	
$R1^3$	000000a0000000400000004000000000	2^{-33}
$R2^3$	00000000000000000000000000000000	
$R3^3$	404080c04080c04080c0404030101020	
$a_{15}^4, \dots, a_0^4$	df8000e00010002000000020f36870280000000006000000000000007$fdfdf$00	
$b_{15}^4, \dots, b_0^4$	004000000048000000000002000000000400000000000000000000000000000008	
$R1^4$	c0408010008040108040401040404000	2^{-38}
$R2^4$	c6424284c0404080c040408000000000	
$R3^4$	00000000000000000000000000000000	
Output : z^4	0642c294c0880090400000b040404004	2^{-17}

computer equipped with a 48-core Intel(R) Xeon(R) Platinum 8260 CPU @ 2.40GHz with Gurobi optimization 9.0 [10].

Results for SNOW-V The search results of SNOW-V are displayed in Table 6. For 3-rounds, the best differential probability of a single trail is estimated as 2^{-17} as shown in Table 7. It implies that a distinguishing attacks on 3-rounds is feasible with 2^{17} chosen IVs. Since we search the whole space of IV, these differential probabilities are optimal for 1- to 3-rounds.

To search for more rounds, we constrain that the hamming weight of the IV difference is one because the above optimal characteristic of 1- to 3-rounds are started from the IV difference whose hamming weight one. In this way, we search for differential characteristics up to 4-initialization rounds. As a result, we found a differential characteristic with probability of 2^{-97} as shown in Table 8. To mount the attack using this characteristic, it requires 2^{97} chosen IVs. So, it exceeds the data limitations for a fixed key of 2^{64}. However, we believe that it is meaningful for deeply understanding the security of SNOW-V, e.g., it might be feasible in the weak-key setting.

Table 9. Differential characteristic probability (DCP) for the reduced initialization round of SNOW-Vi.

Rounds	1	2	3	4
Optimal DCP (whole IV space)	1	2^{-2}	2^{-12}	2^{-39}
Optimal DCP (hamming weight of IV difference is one)	–	–	–	–

Table 10. The differential characteristic for 4-initialization rounds of SNOW-Vi.

Input : IV	00000000000000000000000000000008	
$a^0_{15},\ldots,a^0_0$	00000000000000000000000000000000000000000000000000000000000000008	
$b^0_{15},\ldots,b^0_0$	00000000000000000000000000000000000000000000000000000000000000000	
$R1^0$	00000000000000000000000000000000	1
$R2^0$	00000000000000000000000000000000	
$R3^0$	00000000000000000000000000000000	
$a^1_{15},\ldots,a^1_0$	00000000000000000000000000000010000000000000000000000000000000000	
$b^1_{15},\ldots,b^1_0$	00000000000000000000000000000008000000000000000000000000000000000	
$R1^1$	00000000000000000000000000000000	1
$R2^1$	00000000000000000000000000000000	
$R3^1$	00000000000000000000000000000000	
$a^2_{15},\ldots,a^2_0$	00000000000000000000000001000080000000000000000000000000000000010	
$b^2_{15},\ldots,b^2_0$	00000000000000000000000000000080000000000000000000000000000000008	
$R1^2$	00000000000000000000000000000010	2^{-2}
$R2^2$	00000000000000000000000000000000	
$R3^2$	00000000000000000000000000000000	
$a^3_{15},\ldots,a^3_0$	00000000000000000000010000800100000000000000000000000000000100008	
$b^3_{15},\ldots,b^3_0$	00000000000000000000000000000080000000000000000000000000000000008	
$R1^3$	00000000000000010000000000000008	2^{-12}
$R2^3$	00000000000000000000000018080810	
$R3^3$	00000000000000000000000000000000	
$a^4_{15},\ldots,a^4_0$	00000000000001000100008183808080000000000000000000000000001000080010	
$b^4_{15},\ldots,b^4_0$	00000000000000000000000001000100000000000000000000000000000000008	
$R1^4$	00000008000000000000000800001000	2^{-20}
$R2^4$	00000000$f4a5a551$000000006c242448	
$R3^4$	2424486c3060503080c0404003010102	
Output : z	0x00000008$f4a5a551$000000086c343458	2^{-5}

Results for SNOW-Vi The search results of SNOW-Vi are displayed in Table 9. For SNOW-Vi, we can evaluate the best differential probability of a single trail in the whole IV space over 4-round. As a result, we found the differential characteristic with probability of 2^{-39}. The detailed differential characteristic is shown in Table 10. It implies that a distinguishing attack on 4-round is feasible with 2^{39} chosen IVs. It should be mentioned that this characteristic is the optimal for 4-round of SNOW-Vi.

5 Bit-wise Differential Distinguisher

In this section, we first introduce single-bit and dual-bit differential cryptanalysis based on the study by Choudhuri and Maitra [3]. Then, we present an effective chosen-IV technique for our cryptanalysis of the 4-round SNOW-V and the 5-round SNOW-Vi. Finally, we provide the experimental results for bit-wise differential biases using the chosen-IV technique.

5.1 Single-bit and Dual-bit Differential Cryptanalysis

To search for bit-wise differential biases of the reduced-round SNOW-V and SNOW-Vi, we utilize single-bit and dual-bit differential cryptanalysis based on the study on the reduced-round Salsa and ChaCha, as reported by Choudhuri and Maitra [3].

Let $iv_i[j]$ be the j-th bit of the i-th element in IV for $0 \leq i \leq 7$ and $0 \leq j \leq 15$ and let $iv'_i[j]$ be an associated bit with the input difference $\Delta^{(0)}_{i,j} = iv_i[j] \oplus iv'_i[j]$, which is described as $\mathcal{ID}$. Let $z_p[q]$ be the q-th bit of the p-th word in the first output keystream block z for $0 \leq p \leq 15$ and $0 \leq q \leq 7$ and let $z'_p[q]$ be an associated bit with the r-round output difference $\Delta^{(r)}_{p,q} = z_p[q] \oplus z'_p[q]$, which is described as $\mathcal{OD}$. Note that $iv_0[0]$ and $iv_7[15]$ are the least significant bit (LSB) and most significant bit (MSB) of IV, and $z_0[0]$ and $z_{15}[7]$ are the LSB and MSB of z, respectively. For a fixed key and all possible choices of IVs, single-bit and dual-bit differential probabilities are defined by

$$\Pr\big(\Delta^{(r)}_{p,q} = 1 \mid \Delta^{(0)}_{i,j} = 1\big) = \frac{1}{2}(1 + \epsilon_d), \tag{25}$$

$$\Pr\big(\Delta^{(r)}_{p_0,q_0} \oplus \Delta^{(r)}_{p_1,q_1} = 1 \mid \Delta^{(0)}_{i,j} = 1\big) = \frac{1}{2}(1 + \epsilon_d), \tag{26}$$

where ϵ_d denotes the bias of the $\mathcal{OD}$.

To distinguish the first keystream block z generated by the reduced-round SNOW-V from true random number sequences, we utilize the following theorem proved by Mantin and Shamir [12].

Theorem 1 ([12, Theorem 2]). *Let $\mathcal{X}$ and $\mathcal{Y}$ be two distributions, and suppose that the event e occurs in $\mathcal{X}$ with a probability p and $\mathcal{Y}$ with a probability $p \cdot (1+q)$. Then, for small p and q, $\mathcal{O}(\frac{1}{p \cdot q^2})$ samples suffice to distinguish $\mathcal{X}$ from $\mathcal{Y}$ with a constant probability of success.*

Let $\mathcal{X}$ be a distribution of the $\mathcal{OD}$ of true random number sequences, and $\mathcal{Y}$ be a distribution of the $\mathcal{OD}$ of the first keystream block z generated by the reduced-round SNOW-V. Based on single-bit and dual-bit differential probabilities, the number of samples to distinguish $\mathcal{X}$ and $\mathcal{Y}$ is $\mathcal{O}(\frac{2}{\epsilon_d^2})$ since p and q are equal to $\frac{1}{2}$ and ϵ_d, respectively.

5.2 Chosen-IV Technique

We analyze the source code of the LFSR_update algorithm in SNOW-V (refer to Listing 1 for details) and notice the following two properties.

Listing 1. LFSR_update algorithm in SNOW-V

```
 1:    typedef uint16_t u16;
 2:    u16 A[16], B[16]; // The 32 cells of the two LFSRs
 3:
 4:    void lfsr_update ( void ){
 5:        for ( int i=0; i<8; i++ ){
 6:            u16 u = mul_x ( A[0], 0x990f ) ^ A[1] ^ mul_x_inv ( A[8], 0xcc87 ) ^ B[0];
 7:            u16 v = mul_x ( B[0], 0xc963 ) ^ B[3] ^ mul_x_inv ( B[8], 0xe4b1 ) ^ A[0];
 8:
 9:            for ( int j=0; j<15; j++ ){
10:                A[j] = A[j+1];
11:                B[j] = B[j+1];
12:            }
13:
14:            A[15] = u;
15:            B[15] = v;
16:        }
17:    }
```

Listing 2. mul_x function

```
1:    typedef uint16_t u16;
2:
3:    u16 mul_x ( u16 v, u16 c ){
4:        if ( v & 0x8000 ){
5:            return ( v << 1 ) ^ c;
6:        } else {
7:            return ( v << 1 );
8:        }
9:    }
```

Listing 3. mul_x_inv function

```
1:    typedef uint16_t u16;
2:
3:    u16 mul_x_inv ( u16 v, u16 d ){
4:        if ( v & 0x0001 ){
5:            return ( v >> 1 ) ^ d;
6:        } else {
7:            return ( v >> 1 );
8:        }
9:    }
```

Listing 4. LFSR_update algorithm in SNOW-Vi

```
 1:    typedef uint16_t u16;
 2:    u16 A[16], B[16]; // The 32 cells of the two LFSRs
 3:
 4:    void lfsr_update ( void ){
 5:      for ( int i=0; i<8; i++ ){
 6:          u16 u = mul_x ( A[0], 0x4a6d ) ^ A[7] ^ B[0];
 7:          u16 v = mul_x ( B[0], 0xcc87 ) ^ B[8] ^ A[0];
 8:
 9:          for ( int j=0; j<15; j++ ){
10:              A[j] = A[j+1];
11:              B[j] = B[j+1];
12:          }
13:
14:          A[15] = u;
15:          B[15] = v;
16:      }
17:    }
```

Property 1. The mul_x function is executed 16 times in the LFSR_update algorithm, and the output varies with the value of the MSB.

Property 2. The `mul_x_inv` function is executed 16 times in the `LFSR_update` algorithm, and the output varies with the value of the LSB.

When the MSB of the input `v` to the `mul_x` function is 0, the output bits are not properly mixed because the input `v` is only shifted one bit to the left (see step 7 in Listing 2). On the contrary, when the MSB of the input `v` to the `mul_x` function is 1, the output bits are sufficiently mixed since the input `v` shifted one bit to the left is XORed with another input `c` (see step 5 in Listing 2). These lead to Property 1 that the MSB of the input `v` to the `mul_x` function affects whether the output bits are mixed or not. Since the `mul_x_inv` function is calculated in the similar manner as the `mul_x` function, Property 2 implies that the LSB of the input `v` to the `mul_x_inv` function affects whether the output bits are mixed or not. Furthermore, these properties may be considered to affect whether the propagation of differences is diffused or not.

Based on the two properties of the `LFSR_update` algorithm in SNOW-V, we present an effective chosen-IV technique for our cryptanalysis of the reduced-round SNOW-V. In the SNOW-V initialization, IV is loaded into the eight cells in the LFSR-A by assigning $(a_7, a_6, \ldots, a_0) = (iv_7, iv_6, \ldots, iv_0)$. In addition, the adversaries can choose arbitrary IVs as the $\mathcal{ID}$. Therefore, choosing IVs whose MSBs and LSBs are 0 should suppress the propagation of differences throughout the internal state of SNOW-V during the initialization phase.

We define the following eight domains for single-bit and dual-bit differential cryptanalysis of the reduced-round SNOW-V:

$$\mathcal{V}_0 = \{\texttt{xxxxxxxxxxxxxxxx}_{(2)} \mid \texttt{x} \in \{\texttt{0}, \texttt{1}\}\},$$
$$\mathcal{V}_1 = \{\texttt{0xxxxxxxxxxxxxx0}_{(2)} \mid \texttt{x} \in \{\texttt{0}, \texttt{1}\}\},$$
$$\mathcal{V}_2 = \{\texttt{00xxxxxxxxxxxx00}_{(2)} \mid \texttt{x} \in \{\texttt{0}, \texttt{1}\}\},$$
$$\mathcal{V}_3 = \{\texttt{000xxxxxxxxx000}_{(2)} \mid \texttt{x} \in \{\texttt{0}, \texttt{1}\}\},$$
$$\mathcal{V}_4 = \{\texttt{0000xxxxxxxx0000}_{(2)} \mid \texttt{x} \in \{\texttt{0}, \texttt{1}\}\},$$
$$\mathcal{V}_5 = \{\texttt{00000xxxxxx00000}_{(2)} \mid \texttt{x} \in \{\texttt{0}, \texttt{1}\}\},$$
$$\mathcal{V}_6 = \{\texttt{000000xxxx000000}_{(2)} \mid \texttt{x} \in \{\texttt{0}, \texttt{1}\}\},$$
$$\mathcal{V}_7 = \{\texttt{0000000xx0000000}_{(2)} \mid \texttt{x} \in \{\texttt{0}, \texttt{1}\}\}.$$

On the other hand, the `LFSR_update` algorithm in SNOW-Vi uses only the `mul_x` function, as shown in Listing 4; thus, we should consider only Property 1 and choosing IVs whose MSBs are 0 should suppress the propagation of differences throughout the internal state of SNOW-Vi during the initialization phase.

We define the following three domains for single-bit and dual-bit differential cryptanalysis of the reduced-round SNOW-Vi:

$$\mathcal{V}'_0 = \{\texttt{xxxxxxxxxxxxxxxx}_{(2)} \mid \texttt{x} \in \{\texttt{0}, \texttt{1}\}\},$$
$$\mathcal{V}'_1 = \{\texttt{00000000xxxxxxxx}_{(2)} \mid \texttt{x} \in \{\texttt{0}, \texttt{1}\}\},$$
$$\mathcal{V}'_2 = \{\texttt{000000000000xxxx}_{(2)} \mid \texttt{x} \in \{\texttt{0}, \texttt{1}\}\}.$$

Table 11. Best single-bit and dual-bit differential biases ($\log_2$) for 4-round SNOW-V.

Domain	Single-bit			Dual-bit		
	$\mathcal{ID}$	$\mathcal{OD}$	$|\epsilon_d|$	$\mathcal{ID}$	$\mathcal{OD}$	$|\epsilon_d|$
$\mathcal{V}_0$	$\Delta_{10,7}^{(0)}$	$\Delta_{0,0}^{(4)}$	-10.299	$\Delta_{4,1}^{(0)}$	$\Delta_{0,1}^{(4)} \oplus \Delta_{1,1}^{(4)}$	-9.432
$\mathcal{V}_1$	$\Delta_{3,1}^{(0)}$	$\Delta_{0,0}^{(4)}$	-10.114	$\Delta_{4,1}^{(0)}$	$\Delta_{0,1}^{(4)} \oplus \Delta_{1,1}^{(4)}$	-9.243
$\mathcal{V}_2$	$\Delta_{4,2}^{(0)}$	$\Delta_{0,0}^{(4)}$	-9.804	$\Delta_{4,2}^{(0)}$	$\Delta_{0,0}^{(4)} \oplus \Delta_{1,0}^{(4)}$	-9.069
$\mathcal{V}_3$	$\Delta_{0,5}^{(0)}$	$\Delta_{2,4}^{(4)}$	-9.121	$\Delta_{4,1}^{(0)}$	$\Delta_{0,1}^{(4)} \oplus \Delta_{1,1}^{(4)}$	-8.825
$\mathcal{V}_4$	$\Delta_{6,6}^{(0)}$	$\Delta_{8,2}^{(4)}$	-8.975	$\Delta_{14,7}^{(0)}$	$\Delta_{0,1}^{(4)} \oplus \Delta_{1,1}^{(4)}$	-7.343
$\mathcal{V}_5$	$\Delta_{13,4}^{(0)}$	$\Delta_{7,3}^{(4)}$	-7.904	$\Delta_{6,7}^{(0)}$	$\Delta_{2,2}^{(4)} \oplus \Delta_{3,2}^{(4)}$	-5.675
$\mathcal{V}_6$	$\Delta_{13,1}^{(0)}$	$\Delta_{5,4}^{(4)}$	-6.197	$\Delta_{0,6}^{(0)}$	$\Delta_{0,0}^{(4)} \oplus \Delta_{1,7}^{(4)}$	-3.725
$\mathcal{V}_7$	$\Delta_{14,1}^{(0)}$	$\Delta_{12,3}^{(4)}$	-4.268	$\Delta_{9,0}^{(0)}$	$\Delta_{0,1}^{(4)} \oplus \Delta_{3,2}^{(4)}$	-1.733

In the next subsection, we show our experimental observations of the bit-wise differential biases of the reduced-round SNOW-V and SNOW-Vi for each domain.

5.3 Experimental Results

We have conducted experiments to search for the bit-wise differential biases of the reduced-round SNOW-V and SNOW-Vi. The following is our experimental environment: five Linux machines with 40-core Intel(R) Xeon(R) CPU E5-2660 v3 (2.60GHz), 128.0 GB of main memory, a gcc 7.2.0 compiler, and the C programming language.

Bit-wise Differential Biases of SNOW-V To search for single-bit (or dual-bit) differential biases of the reduced-round SNOW-V, our experiments have been conducted with 2^8 (or 2^6) trials using 2^{24} $\mathcal{ID}$s for each key, excluding domain $\mathcal{V}_7$. Since domain $\mathcal{V}_7$ contains only 2^{16} elements, we have conducted experiments with 2^{16} (or 2^{14}) trials using 2^{16} $\mathcal{ID}$s for each key to search for the single-bit (or dual-bit) differential biases.

Tables 11 and 12 show the best single-bit and dual-bit differential biases for the 4- and 5-round SNOW-V. As shown in Table 11, we obtain higher biases when the domain is restricted using the chosen-IV technique. For example, we obtain the best single-bit (or dual-bit) differential bias of $|\epsilon_d| = 2^{-4.268}$ (or $2^{-1.733}$) for domain $\mathcal{V}_7$, whereas we find $|\epsilon_d| = 2^{-10.299}$ (or $2^{-9.432}$) for domain $\mathcal{V}_0$. However, as shown in Table 12, all of the best single-bit and dual-bit differential biases are almost constant regardless of the domain in the 5-round SNOW-V. These results demonstrate that the chosen-IV technique is valid for the 4-round SNOW-V, but not for the 5-round SNOW-V.

For the 4-round SNOW-V, the best dual-bit differential bias in domain $\mathcal{V}_7$, i.e., $|\epsilon_d| = 2^{-1.733}$, provides a practical bit-wise differential distinguisher. According to Theorem 1, $2^{4.466}$ samples suffice to distinguish the 4-round SNOW-V

Table 12. Best single-bit and dual-bit differential biases ($\log_2$) for 5-round SNOW-V.

Domain	Single-bit			Dual-bit		
	$\mathcal{ID}$	$\mathcal{OD}$	$|\epsilon_d|$	$\mathcal{ID}$	$\mathcal{OD}$	$|\epsilon_d|$
$\mathcal{V}_0$	$\Delta_{12,6}^{(0)}$	$\Delta_{12,3}^{(5)}$	-13.943	$\Delta_{7,7}^{(0)}$	$\Delta_{4,2}^{(5)} \oplus \Delta_{15,5}^{(5)}$	-12.771
$\mathcal{V}_1$	$\Delta_{0,5}^{(0)}$	$\Delta_{10,6}^{(5)}$	-13.971	$\Delta_{0,1}^{(0)}$	$\Delta_{3,3}^{(5)} \oplus \Delta_{13,2}^{(5)}$	-12.819
$\mathcal{V}_2$	$\Delta_{14,7}^{(0)}$	$\Delta_{2,3}^{(5)}$	-14.055	$\Delta_{4,4}^{(0)}$	$\Delta_{4,0}^{(5)} \oplus \Delta_{14,6}^{(5)}$	-12.622
$\mathcal{V}_3$	$\Delta_{4,0}^{(0)}$	$\Delta_{15,1}^{(5)}$	-14.021	$\Delta_{11,0}^{(0)}$	$\Delta_{9,3}^{(5)} \oplus \Delta_{11,7}^{(5)}$	-12.671
$\mathcal{V}_4$	$\Delta_{1,4}^{(0)}$	$\Delta_{5,2}^{(5)}$	-14.147	$\Delta_{9,1}^{(0)}$	$\Delta_{8,3}^{(5)} \oplus \Delta_{14,3}^{(5)}$	-12.713
$\mathcal{V}_5$	$\Delta_{15,4}^{(0)}$	$\Delta_{2,0}^{(5)}$	-14.047	$\Delta_{6,7}^{(0)}$	$\Delta_{7,5}^{(5)} \oplus \Delta_{15,4}^{(5)}$	-12.669
$\mathcal{V}_6$	$\Delta_{2,5}^{(0)}$	$\Delta_{15,6}^{(5)}$	-14.081	$\Delta_{11,1}^{(0)}$	$\Delta_{4,0}^{(5)} \oplus \Delta_{15,2}^{(5)}$	-12.820
$\mathcal{V}_7$	$\Delta_{6,7}^{(0)}$	$\Delta_{6,7}^{(5)}$	-13.589	$\Delta_{0,7}^{(0)}$	$\Delta_{1,2}^{(5)} \oplus \Delta_{6,1}^{(5)}$	-12.408

from a true random number generator with a constant probability of success. Similarly, for the 5-round SNOW-V, the best dual-bit differential bias in domain $\mathcal{V}_2$, i.e., $|\epsilon_d| = 2^{-12.622}$, provides the best bit-wise differential distinguisher. Although the best dual-bit differential bias in domain $\mathcal{V}_7$ is higher than that in $\mathcal{V}_2$, i.e., $|\epsilon_d| = 2^{-12.408}$, that in domain $\mathcal{V}_7$ cannot provide the best differential distinguisher because domain $\mathcal{V}_7$ contains only 2^{16} elements. Thus, $2^{26.244}$ samples suffice to distinguish the 5-round SNOW-V from a true random number generator; however, the accuracy of the experimental results may be insufficient because we have conducted experiments with only 2^{24} $\mathcal{ID}$s to observe the differential biases. To search for more precise dual-bit differential biases for the 5-round SNOW-V, we have focused on the best $\mathcal{ID}$-$\mathcal{OD}$ pair in each domain (excluding domain $\mathcal{V}_7$) listed in Table 12, and have conducted additional experiments with 2^8 trials using 2^{32} $\mathcal{ID}$s for each key. Consequently, we obtain the best dual-bit differential biases in domain $\mathcal{V}_4$, such that $\mathcal{ID}$ is $\Delta_{9,1}^{(0)}$, $\mathcal{OD}$ is $\Delta_{8,3}^{(5)} \oplus \Delta_{14,3}^{(5)}$, and $|\epsilon_d|$ is approximately $2^{-17.934}$; thus, our experiments have revealed that at least $2^{36.868}$ samples suffice to distinguish the 5-round SNOW-V from a true random number generator.

Bit-wise Differential Biases of SNOW-Vi To search for single-bit or dual-bit differential biases of the reduced-round SNOW-Vi, our experiments have been conducted with 2^8 trials using 2^{24} $\mathcal{ID}$s for each key.

Table 13 shows the best single- and dual-bit differential biases of $|\epsilon_d| = 1$ regardless of the domain in the 4-round SNOW-Vi; thus, these biases provide practical bit-wise differential distinguishers. According to Theorem 1, $2^{1.000}$ samples suffice to distinguish the 4-round SNOW-Vi from a true random number generator with a constant probability of success. Incidentally, we obtain a total of 2749 biases of $|\epsilon_d| = 1$ for the 4-round SNOW-Vi, but only some of the results are shown in the table due to space constraints.

Table 13. Best single-bit and dual-bit differential biases ($\log_2$) for 4-round SNOW-Vi.

Domain	Single-bit			Dual-bit		
	$\mathcal{ID}$	$\mathcal{OD}$	$\lvert\epsilon_d\rvert$	$\mathcal{ID}$	$\mathcal{OD}$	$\lvert\epsilon_d\rvert$
$\mathcal{V}'_0$	$\Delta^{(0)}_{1,0}$	$\Delta^{(4)}_{0,0}$	0	$\Delta^{(0)}_{1,0}$	$\Delta^{(4)}_{0,0} \oplus \Delta^{(4)}_{0,1}$	0
$\mathcal{V}'_1$	$\Delta^{(0)}_{1,0}$	$\Delta^{(4)}_{0,0}$	0	$\Delta^{(0)}_{1,0}$	$\Delta^{(4)}_{0,0} \oplus \Delta^{(4)}_{0,1}$	0
$\mathcal{V}'_2$	$\Delta^{(0)}_{1,0}$	$\Delta^{(4)}_{0,0}$	0	$\Delta^{(0)}_{1,0}$	$\Delta^{(4)}_{0,0} \oplus \Delta^{(4)}_{0,1}$	0

Table 14. Best single-bit and dual-bit differential biases ($\log_2$) for 5-round SNOW-Vi.

Domain	Single-bit			Dual-bit		
	$\mathcal{ID}$	$\mathcal{OD}$	$\lvert\epsilon_d\rvert$	$\mathcal{ID}$	$\mathcal{OD}$	$\lvert\epsilon_d\rvert$
$\mathcal{V}'_0$	$\Delta^{(0)}_{7,0}$	$\Delta^{(5)}_{0,0}$	-10.910	$\Delta^{(0)}_{12,0}$	$\Delta^{(5)}_{0,0} \oplus \Delta^{(5)}_{0,1}$	-11.508
$\mathcal{V}'_1$	$\Delta^{(0)}_{1,0}$	$\Delta^{(5)}_{0,0}$	-9.510	$\Delta^{(0)}_{9,1}$	$\Delta^{(5)}_{0,0} \oplus \Delta^{(5)}_{0,5}$	-9.759
$\mathcal{V}'_2$	$\Delta^{(0)}_{4,0}$	$\Delta^{(5)}_{0,1}$	-6.835	$\Delta^{(0)}_{9,2}$	$\Delta^{(5)}_{0,0} \oplus \Delta^{(5)}_{0,1}$	-7.216

Table 14 shows the best single- and dual-bit differential biases for the 5-round SNOW-Vi, and we obtain higher biases when the domain is restricted using the chosen-IV technique, e.g., the best single-bit (or dual-bit) differential bias of $\lvert\epsilon_d\rvert = 2^{-6.835}$ (or $2^{-7.216}$) for domain $\mathcal{V}'_2$, whereas $\lvert\epsilon_d\rvert = 2^{-10.910}$ (or $2^{-11.508}$) for domain $\mathcal{V}'_0$. For the 5-round SNOW-Vi, the best single-bit differential bias in domain $\mathcal{V}'_2$, i.e., $\lvert\epsilon_d\rvert = 2^{-6.835}$, also provides a practical bit-wise differential distinguisher. Thus, $2^{14.670}$ samples suffice to distinguish the 4-round SNOW-V from a true random number generator with a constant probability of success.

Table 15 shows the best single- and dual-bit differential biases for the 6-round SNOW-Vi, and all of the best single-bit and dual-bit differential biases are almost constant regardless of the domain in the 6-round SNOW-Vi; thus, these results demonstrate that the chosen-IV technique is valid for the 5-round SNOW-Vi, but not for the 6-round SNOW-Vi. For the 6-round SNOW-Vi, the best dual-bit differential bias in domain $\mathcal{V}'_1$, i.e., $\lvert\epsilon_d\rvert = 2^{-13.546}$, provides the best bit-wise differential distinguisher with $2^{28.092}$ samples. However, the accuracy of the experimental results may be insufficient because we have conducted experiments with only 2^{24} $\mathcal{ID}$s to observe the differential biases. To search for more precise single- and dual-bit differential biases for the 6-round SNOW-Vi, we have focused on the best $\mathcal{ID}$-$\mathcal{OD}$ pair in each domain listed in Table 15, and have conducted additional experiments with 2^8 trials using 2^{32} $\mathcal{ID}$s for each key. Consequently, we obtain the best single-bit differential biases in domain $\mathcal{V}'_1$, such that $\mathcal{ID}$ is $\Delta^{(0)}_{13,5}$, $\mathcal{OD}$ is $\Delta^{(6)}_{2,6}$, and $\lvert\epsilon_d\rvert$ is approximately $2^{-18.597}$; thus, our experiments have revealed that at least $2^{38.194}$ samples suffice to distinguish the 6-round SNOW-Vi from a true random number generator.

Table 15. Best single-bit and dual-bit differential biases ($\log_2$) for 6-round SNOW-Vi.

Domain	Single-bit			Dual-bit		
	$\mathcal{ID}$	$\mathcal{OD}$	$|\epsilon_d|$	$\mathcal{ID}$	$\mathcal{OD}$	$|\epsilon_d|$
$\mathcal{V}'_0$	$\Delta^{(0)}_{2,4}$	$\Delta^{(6)}_{13,5}$	-14.045	$\Delta^{(0)}_{15,1}$	$\Delta^{(6)}_{4,0} \oplus \Delta^{(6)}_{7,3}$	-13.647
$\mathcal{V}'_1$	$\Delta^{(0)}_{13,5}$	$\Delta^{(6)}_{2,6}$	-13.951	$\Delta^{(0)}_{10,5}$	$\Delta^{(6)}_{1,2} \oplus \Delta^{(6)}_{14,1}$	-13.546
$\mathcal{V}'_2$	$\Delta^{(0)}_{1,1}$	$\Delta^{(6)}_{14,4}$	-14.027	$\Delta^{(0)}_{10,5}$	$\Delta^{(6)}_{2,2} \oplus \Delta^{(6)}_{8,2}$	-14.726

6 Key Recovery Attack

In this section, we describe a key recovery attack on the 4-round SNOW-V and SNOW-Vi. To the best of our knowledge, our attack is the best key recovery attack on the reduced-round SNOW-V and SNOW-Vi since the cube attack on the 3-round SNOW-V and SNOW-Vi proposed by Ekdahl et al. [5, 6], which was the best to date. Our proposed attack is an improvement on the differential attack based on a technique called *probabilistic neutral bits* (PNB) proposed by Aumasson et al. [2].

6.1 Differential Attack Based on Probabilistic Neutral Bits (PNB)

Aummason et al. proposed a differential attack based on PNB and applied it to Salsa and ChaCha [2]. In this subsection, we introduce their attack to clarify the difference from our proposed attack, which is described in Section 6.2. Their attack consists of two phases: precomputation and online phases. The precomputation phase is further divided into three phases: differential characteristic search (as described in Section 5.1), PNB identification, and probabilistic backwards computation phases.

PNB Identification Phase. PNB is a concept which divides the secret key bits into two sets: m-bit significant key bits and n-bit non-significant key bits. To identify these two sets, Aumasson et al. focused on the amount of influence which each secret key bit has on the output difference $\mathcal{OD}$, and defined that amount as *neutral measure*.

Definition 1 ([2, Definition 1]). *The neutral measure of the key bit κ_i with respect to the output difference $\mathcal{OD}$ is defined as γ_i, where $\mathrm{Pr} = \frac{1}{2}(1 + \gamma_i)$ is the probability that complementing the key bit κ_i does not change the $\mathcal{OD}$.*

For example, according to Definition 1, we have the following singular cases of the neutral measure:

- $\gamma_i = 1$: $\mathcal{OD}$ does not depend on the i-th key bit, i.e., it is non-significant.
- $\gamma_i = 0$: $\mathcal{OD}$ is statistically independent of the i-th key bit, i.e., it is significant.

To identify the PNB by using the concept of the neutral measure, we perform the following procedure after the differential characteristic search phase:

Step 1. Compute the keystream pair Z, Z' corresponding to the input pair $X^{(0)}, X'^{(0)}$ with the input difference $\Delta_{i,j}^{(0)}$. Note that the keystream Z is derived by $X^{(0)} + X^{(R)}$ in the case of Salsa and ChaCha.

Step 2. Prepare a new input pair $\overline{X}^{(0)}, \overline{X'}^{(0)}$ with the key bit position i of the original input pair $X^{(0)}, X'^{(0)}$ flipped by one bit.

Step 3. Compute the internal state pair $Y^{(r)}, Y'^{(r)}$ with $Z - \overline{X}^{(0)}$, $Z' - \overline{X'}^{(0)}$ for $r < R$, as inputs to the inverse function of the initialization in the case of Salsa and ChaCha.

Step 4. Compute $\Gamma_{p,q}^{(r)} = y_p[q] \oplus y'_p[q]$, where $y_p[q]$ and $y'_p[q]$ are the q-th bit of the p-th word of $Y^{(r)}$ and $Y'^{(r)}$, respectively.

Step 5. Repeatedly perform Steps 1-4 by using different input pairs with the same $\Delta_{i,j}^{(0)}$; compute the neutral measure as $\Pr(\Delta_{p,q}^{(r)} = \Gamma_{p,q}^{(r)} \mid \Delta_{i,j}^{(0)} = 1) = \frac{1}{2}(1 + \gamma_i)$, where $\Delta_{p,q}^{(r)}$ is the output difference derived during the differential characteristic search (as described in Section 5.1).

Step 6. Set a threshold γ and put all key bits with $\gamma_i < \gamma$ into a set of significant key bits (of size m) and those with $\gamma_i \geq \gamma$ into a set of non-significant key bits (of size n).

Probabilistic Backwards Computation Phase. In the differential characteristic search phase, we derive the r-th round differential biases from input pairs with the chosen input difference, i.e., this implies that we perform the forwards computation in the target cipher. However, in the case of Salsa and ChaCha, we can also derive the r-th round differential biases from the obtained keystream by performing the backwards computation, which is called the *probabilistic backwards computation*.

In the probabilistic backwards computation phase, we perform the following procedure after the PNB identification phase:

Step 1. Compute the keystream pair Z, Z' corresponding to the input pair $X^{(0)}, X'^{(0)}$ with the input difference $\Delta_{i,j}^{(0)}$.

Step 2. Prepare a new input pair $\hat{X}^{(0)}, \hat{X'}^{(0)}$ with only non-significant key bits reset to a fixed value (e.g., all zero) from the original input pair $X^{(0)}, X'^{(0)}$.

Step 3. Compute the internal state pair $\hat{Y}^{(r)}, \hat{Y'}^{(r)}$ with $Z - \hat{X}^{(0)}$, $Z' - \hat{X'}^{(0)}$ for $r < R$, as inputs to the inverse function of the initialization in the case of Salsa and ChaCha.

Step 4. Compute $\hat{\Gamma}_{p,q}^{(r)} = \hat{y}_p[q] \oplus \hat{y}'_p[q]$, where $\hat{y}_p[q]$ and $\hat{y}'_p[q]$ are the q-th bit of the p-th word of $\hat{Y}^{(r)}$ and $\hat{Y'}^{(r)}$, respectively.

Step 5. Repeatedly perform Steps 1-4 by using different input pairs with the same $\Delta_{i,j}^{(0)}$; compute the r-round bias ϵ_a as $\Pr(\Delta_{p,q}^{(r)} = \hat{\Gamma}_{p,q}^{(r)} \mid \Delta_{i,j}^{(0)} = 1) = \frac{1}{2}(1 + \epsilon_a)$, where $\Delta_{p,q}^{(r)}$ is the output difference derived during the differential characteristic search (as described in Section 5.1).

According to [2], the bias ϵ is approximated as $\epsilon_d \cdot \epsilon_a$ and considered to compute the overall complexity of the attack on the R-round target cipher.

Online Phase. According to [2], we perform the following procedure after the precomputation phase:

Step 1. For an unknown key, we collect N keystream pairs where each pair is generated by a random input pair (satisfying the relevant input difference).

Step 2. For each choice of the subkey (i.e., the m-bit significant key bits) do:

    Step 2-1. Derive the r-th round differential biases from the N keystream pairs by performing the backwards computation.

    Step 2-2. If the optimal distinguisher legitimates the subkeys candidate as a (possibly) correct one, we perform an additional exhaustive search over the n non-significant key bits in order to check the correctness of this filtered subkey and to find the non-significant key bits.

    Step 2-3. Stop if the correct key is found, and output the recovered key.

Complexity Estimation. According to [2, 3], given samples N and probability of false alarm is $P_{fa} = 2^{-\alpha}$, the time complexity of the attack is given by

$$2^m(N + 2^n P_{fa}) = 2^m N + 2^{256-\alpha}, \text{ where } N \approx \left(\frac{\sqrt{\alpha \log 4} + 3\sqrt{1 - \epsilon^2}}{\epsilon} \right)^2,$$

for probability of non-detection $P_{nd} = 1.3 \times 10^{-3}$. In practice, α (and hence N) is chosen such that it minimizes the time complexity of the attack.

6.2 Application to SNOW-V and SNOW-Vi

In this subsection, we present how to apply the differential attack based on PNB, as described in Section 6.1, to the reduced-round SNOW-V and SNOW-Vi. However, its application to SNOW-V and SNOW-Vi, unlike the existing attacks on Salsa and ChaCha, is difficult to compute the difference biases from the obtained keystreams by performing the backwards computation, i.e., it is difficult to perform in the same procedure as Step 3 in the PNB identification phase and Step 3 in the probabilistic backwards computation phase, as described in Section 6.1.

To solve this problem, in our proposed attack, we replace the backwards computations in these steps with the forwards computations. Our attack consists of three precomputation phases: differential characteristic search (as described in Section 5.1), PNB identification, and probabilistic *forwards* computation phases. The online phase is a similar procedure to that described in Section 6.1, i.e., we simply replace the backwards computation with the forwards computation in Step 2-1 of the online phase.

PNB Identification Phase. In the PNB identification phase, we replace Step 3 in the existing phase with a step to perform the forwards computation. Additionally, it is not necessary to perform Step 1 in the existing phase because no backwards computation is performed. In summary, for the application to SNOW-V and SNOW-Vi, we perform the following procedure after the differential characteristic search phase:

Step 1. Prepare a new input pair $\overline{X}^{(0)}, \overline{X'}^{(0)}$ with the key bit position i of the original input pair $X^{(0)}, X'^{(0)}$ flipped by one bit. Note that, according to Section 2.3, an input $X^{(0)}$ of SNOW-V and SNOW-Vi is initialized from a secret key and an initialization vector.

Step 2. Compute the keystream pair z, z' with $\overline{X}^{(0)}, \overline{X'}^{(0)}$ as inputs to the r-round initialization of SNOW-V and SNOW-Vi.

Step 3. Compute $\Gamma_{p,q}^{(r)} = z_p[q] \oplus z_p'[q]$, where $z_p[q]$ and $z_p'[q]$ are the q-th bit of the p-th word of z and z', respectively.

Step 4. Repeatedly perform Steps 1-4 by using different input pairs with the same $\Delta_{i,j}^{(0)}$; compute the neutral measure as $\Pr(\Delta_{p,q}^{(r)} = \Gamma_{p,q}^{(r)} \mid \Delta_{i,j}^{(0)} = 1) = \frac{1}{2}(1 + \gamma_i)$, where $\Delta_{p,q}^{(r)}$ is the output difference derived during the differential characteristic search (as described in Section 5.1).

Step 5. Set a threshold γ, put all key bits with $\gamma_i < \gamma$ into a set of significant key bits (of size m) and those with $\gamma_i \geq \gamma$ into a set of non-significant key bits (of size n).

Probabilistic Forwards Computation Phase. Similar to the proposed PNB identification phase, we improve the existing probabilistic backwards computation phase. In summary, for the application to SNOW-V and SNOW-Vi, we perform the following procedure after the PNB identification phase:

Step 1. Prepare a new input pair $\hat{X}^{(0)}, \hat{X'}^{(0)}$ with only non-significant key bits reset to a fixed value (e.g., all zero) from the original input pair $X^{(0)}, X'^{(0)}$.

Step 2. Compute the keystream pair $\hat{z}, \hat{z}'$ with $\hat{X}^{(0)}, \hat{X'}^{(0)}$ as inputs to the r-round initialization of SNOW-V and SNOW-Vi.

Step 3. Compute $\hat{\Gamma}_{p,q}^{(r)} = \hat{z}_p[q] \oplus \hat{z}_p'[q]$, where $\hat{z}_p[q]$ and $\hat{z}_p'[q]$ are the q-th bit of the p-th word of $\hat{z}^{(r)}$ and $\hat{z}'^{(r)}$, respectively.

Step 4. Repeatedly perform Steps 1-4 by using different input pairs with the same $\Delta_{i,j}^{(0)}$; compute the r-round bias ϵ_a as $\Pr(\Delta_{p,q}^{(r)} = \hat{\Gamma}_{p,q}^{(r)} \mid \Delta_{i,j}^{(0)} = 1) = \frac{1}{2}(1 + \epsilon_a)$, where $\Delta_{p,q}^{(r)}$ is the output difference derived during the differential characteristic search (as described in Section 5.1).

Complexity Estimation. In our proposed attack, we can construct the following two independent distinguishers:

- A distinguisher based on the differential bias ϵ_d.
- A distinguisher based on the bias ϵ_a.

This is because these biases are derived from the (secret) internal states in the existing attacks, whereas they are derived from the keystreams, which are obtained by an adversary under the known plaintext attack scenario, in the application to SNOW-V and SNOW-Vi. Thus, the number of samples N for our attack is given by

$$N \approx \max\left(\left(\frac{\sqrt{\alpha \log 4} + 3\sqrt{1 - \epsilon_d^2}}{\epsilon_d} \right)^2, \left(\frac{\sqrt{\alpha \log 4} + 3\sqrt{1 - \epsilon_a^2}}{\epsilon_a} \right)^2 \right).$$

Table 16. The best parameters for our attack in domain $\mathcal{V}_0$ for the 4-round SNOW-V for each threshold γ, where m is the size of significant key bits.

γ	$\mathcal{ID}$	$\mathcal{OD}$	m	ϵ_a	ϵ_d	α	Data	Time	Probability
1.00	$\Delta_{4,3}^{(0)}$	$\Delta_{0,0}^{(4)} \oplus \Delta_{1,0}^{(4)}$	127	1.000	$2^{-9.548}$	109	$2^{26.96}$	$2^{153.97}$	1.000
0.90	$\Delta_{4,0}^{(0)}$	$\Delta_{0,0}^{(4)} \oplus \Delta_{1,0}^{(4)}$	82	$2^{-0.183}$	$2^{-9.546}$	154	$2^{27.36}$	$2^{109.37}$	0.958
0.80	$\Delta_{4,0}^{(0)}$	$\Delta_{0,0}^{(4)} \oplus \Delta_{1,0}^{(4)}$	72	$2^{-0.538}$	$2^{-9.546}$	164	$2^{27.44}$	$2^{99.45}$	0.729
0.70	$\Delta_{4,0}^{(0)}$	$\Delta_{0,0}^{(4)} \oplus \Delta_{1,0}^{(4)}$	67	$2^{-0.714}$	$2^{-9.546}$	169	$2^{27.48}$	$2^{94.48}$	0.650
0.60	$\Delta_{4,0}^{(0)}$	$\Delta_{0,0}^{(4)} \oplus \Delta_{1,0}^{(4)}$	66	$2^{-0.830}$	$2^{-9.546}$	170	$2^{27.48}$	$2^{93.49}$	0.542
0.50	$\Delta_{4,3}^{(0)}$	$\Delta_{0,0}^{(4)} \oplus \Delta_{1,0}^{(4)}$	61	$2^{-1.388}$	$2^{-9.548}$	175	$2^{27.52}$	$2^{88.53}$	0.334

Additionally, the time complexity of our attack is given in the same way as that of the existing attacks [2, 3], as described in Section 6.1.

6.3 Experimental Results

We have conducted experiments to find the best parameters for our attack on the reduced-round SNOW-V and SNOW-Vi. The following is our experimental environment: five Linux machines with 40-core Intel(R) Xeon(R) CPU E5-2660 v3 (2.60GHz), 128.0 GB of main memory, a gcc 7.2.0 compiler, and the C programming language.

Key Recovery Attack on SNOW-V To find the best parameters for our attack on the reduced-round SNOW-V, our experiments have been conducted with 2^8 trials using 2^{24} $\mathcal{ID}$s for each key excluding domain $\mathcal{V}_7$. Since domain $\mathcal{V}_7$ contains only 2^{16} elements, we have conducted experiments with 2^{16} trials using 2^{16} $\mathcal{ID}$s for each key. In addition, we need to consider the possibility that our attack has no validity because the application to SNOW-V, unlike the existing attacks on Salsa and ChaCha, only perform the forwards computation throughout all phases. To calculate the success probability of our attack, our experiments have been conducted with 1000 trials by using the best parameters obtained from the experiments. In our experiments, we consider the attack to be failed if we can guess a subkey candidate with a higher bias ϵ_a^* than the bias ϵ_a obtained from the correctly guessed subkey.

Tables 16 and 17 show the best parameters for our attack in domains $\mathcal{V}_0$ and $\mathcal{V}_7$ on the 4-round SNOW-V for each threshold γ. Based on these tables, we appear to be able to perform our attack on the 4-round SNOW-V with the least time complexity of $2^{60.35}$ by using the parameter for the threshold $\gamma = 0.50$ in domain $\mathcal{V}_7$, but it has no validity because its success probability is zero. However, as shown in these tables, we can perform our attack with a success probability of one by using the parameter for the threshold $\gamma = 1.00$ in both domains $\mathcal{V}_0$ and $\mathcal{V}_7$. This is because all key bits with a threshold $\gamma_i \geq \gamma = 1.00$ are put into the set of non-significant key bits, and these have no influence on the output difference, i.e., this implies that we can always guess all the m-bits subkeys in

Table 17. The best parameters for our attack in domain $\mathcal{V}_7$ for the 4-round SNOW-V for each threshold γ, where m is the size of significant key bits.

γ	$\mathcal{ID}$	$\mathcal{OD}$	m	ϵ_a	ϵ_d	α	Data	Time	Probability
1.00	$\Delta^{(0)}_{1,6}$	$\Delta^{(4)}_{0,0} \oplus \Delta^{(4)}_{1,1}$	149	1.000	$2^{-1.878}$	108	$2^{11.59}$	$2^{154.60}$	1.000
0.90	$\Delta^{(0)}_{10,7}$	$\Delta^{(4)}_{0,0} \oplus \Delta^{(4)}_{2,7}$	84	$2^{-0.123}$	$2^{-1.747}$	167	$2^{11.84}$	$2^{95.86}$	0.858
0.80	$\Delta^{(0)}_{10,7}$	$\Delta^{(4)}_{0,0} \oplus \Delta^{(4)}_{2,7}$	74	$2^{-0.570}$	$2^{-1.747}$	177	$2^{11.91}$	$2^{85.93}$	0.253
0.70	$\Delta^{(0)}_{10,7}$	$\Delta^{(4)}_{0,0} \oplus \Delta^{(4)}_{2,7}$	71	$2^{-0.646}$	$2^{-1.747}$	181	$2^{11.94}$	$2^{81.95}$	0.150
0.60	$\Delta^{(0)}_{10,7}$	$\Delta^{(4)}_{0,0} \oplus \Delta^{(4)}_{2,7}$	64	$2^{-1.037}$	$2^{-1.747}$	187	$2^{11.98}$	$2^{75.99}$	0.012
0.50	$\Delta^{(0)}_{1,6}$	$\Delta^{(4)}_{0,0} \oplus \Delta^{(4)}_{1,1}$	47	$2^{-1.742}$	$2^{-1.878}$	203	$2^{12.35}$	$2^{60.35}$	0.000

Table 18. The best parameters for our attack on the 4- and 5-round SNOW-Vi for the threshold $\gamma = 1.00$, where m is the size of significant key bits.

Round	Domain	$\mathcal{ID}$	$\mathcal{OD}$	m	ϵ_a	ϵ_d	α	Data	Time
4	$\mathcal{V}'_0$	$\Delta^{(0)}_{9,5}$	$\Delta^{(4)}_{0,7}$	226	1,000	$2^{-0.862}$	27	$2^{7.94}$	$2^{233.99}$
5	$\mathcal{V}'_2$	$\Delta^{(0)}_{1,0}$	$\Delta^{(5)}_{0,0}$	239	1,000	$2^{-7.531}$	1	$2^{19.19}$	$2^{258.34}$

the online phase. As a result, we can perform our attack on the 4-round SNOW-V with a time complexity of $2^{153.97}$ and data complexity of $2^{26.96}$ by using the parameter for the threshold $\gamma = 1.00$ in domain $\mathcal{V}_0$; this is the best key recovery attack on the reduced-round SNOW-V.

Key Recovery Attack on SNOW-Vi To find the best parameters for our attack on the reduced-round SNOW-Vi, our experiments have been conducted with 2^8 trials using 2^{24} $\mathcal{ID}$s for each key. As discussed at the previous paragraph, we can perform our attack on the 4-round SNOW-V with a success probability of one by using the parameter for the threshold $\gamma = 1.00$; thus, we should perform our attack on the reduced-round SNOW-Vi by focusing solely on the threshold $\gamma = 1.00$.

As described in Section 5, we obtain the best single- and dual-bit differential biases of $|\epsilon_d| = 1$ for the 4-round SNOW-Vi. However, $\mathcal{ID}$-$\mathcal{OD}$ pairs with these biases cannot be used in our attack because they do not work properly in the PNB identification phase (i.e., all key bits can be included in the set of non-significant key bits). Due to this, we randomly selected 40 $\mathcal{ID}$-$\mathcal{OD}$ pairs with biases of $|\epsilon_d| < 1$ to properly perform our attack on the 4-round SNOW-Vi.

Table 18 shows our findings of the best parameters for our attack on the 4- and 5-round SNOW-Vi, respectively. We can perform our attack on the 4-round SNOW-Vi with a time complexity of $2^{233.99}$ and a data complexity of $2^{7.94}$ by using the best parameter, such that domain is $\mathcal{V}'_0$, $\mathcal{ID}$ is $\Delta^{(0)}_{9,5}$, $\mathcal{OD}$ is $\Delta^{(4)}_{0,7}$, and $\alpha = 27$. Similarly, we can perform our attack on the 5-round SNOW-Vi with a time complexity of $2^{258.34}$ and a data complexity of $2^{19.19}$ by using the best parameter, such that domain is $\mathcal{V}'_2$, $\mathcal{ID}$ is $\Delta^{(0)}_{1,0}$, $\mathcal{OD}$ is $\Delta^{(5)}_{0,0}$, and $\alpha = 1$; however, this is beyond the security level of SNOW-Vi. Consequently, we have presented

the best key recovery attack on the 4-round SNOW-Vi. It should be noted here that our attack still has room for improvement since we have tried only 40 $\mathcal{ID}$-$\mathcal{OD}$ pairs with biases of $|\epsilon_d| < 1$ to properly perform our attack on the 4-round SNOW-Vi.

6.4 Discussion

The distinguishing attack on the reduced-round SNOW-Vi can be a little stronger than that on the reduced-round SNOW-V. This is because we can construct the 5-round practical distinguisher for SNOW-Vi, even though we can only construct the 4-round practical distinguisher for SNOW-V. Conversely, the key recovery attack on the reduced-round SNOW-Vi may be slightly weaker than that on the reduced-round SNOW-V, as described in this section. Here, we discuss the factors that induce the difference between the distinguishing and key recovery attacks on the reduced-round SNOW-V and SNOW-Vi.

The structural differences between SNOW-V and SNOW-Vi are the LFSR update function and the location of the tap T_2. First, we focus on the difference in the LFSR update function. According to [6, Section 3.2], the new LFSR in SNOW-Vi has a maximum cycle length of 2^{512-1}; thus, it has the same property as the LFSR in SNOW-V. Next, we focus on the difference in the location of the tap T_2, whose values are loaded into the register $R1$ in the FSM part. The tap T_2 in SNOW-V is the location where the 128-bit IV is loaded during the initilization phase, while the tap T_2 in SNOW-Vi is the location where a half of the 256-bit key is loaded during the initilization phase. Given that our distinguishing attacks use the IV difference, it should be more resistant to our distinguishing attacks if the IV difference is easily propagated to the FSM part immediately after the initialization phase; thus, this factor leads to strong resistance to our distinguishing attacks on SNOW-V, due to the location of the tap T_2. On the other hand, we use the key difference in our key recovery attack since non-significant key bits are reset to a fixed value (e.g., all zero) during the probabilistic forwards computation phase (see Section 6.2). It should be more resistant to our key recovery attack if the key difference is easily propagated to the FSM part immediately after the initialization phase; thus, this factor leads to strong resistance to our key recovery attack on SNOW-Vi, due to the location of the tap T_2.

7 Conclusion

In this study, we have analyzed the security of SNOW-V and SNOW-Vi with three attacks: the MILP-aided integral attack, the MILP-aided differential attack, and the bit-wise differential attack. These attacks allow us to construct practical distinguishers of up to four rounds for SNOW-V and five rounds for SNOW-Vi. Furthermore, the differential biases obtained by the bit-wise differential attack can be integrated into our improved key recovery attack based on probabilistic neutral bits, which is inspired by the existing study on Salsa and

ChaCha [2, 3]. As a result, we have presented the best key recovery attack on the 4-round SNOW-V and SNOW-Vi with time complexities of $2^{153.97}$ and $2^{233.99}$ and data complexities of $2^{26.96}$ and $2^{19.19}$, respectively. Consequently, we have improved the best existing attack, which was evaluated by the designers, in the initialization phase of the reduced-round SNOW-V and SNOW-Vi.

References

[1] Ahmed Abdelkhalek, Yu Sasaki, Yosuke Todo, Mohamed Tolba, and Amr M. Youssef. MILP modeling for (large) s-boxes to optimize probability of differential characteristics. *IACR Trans. Symmetric Cryptol.*, 2017(4):99–129, 2017.

[2] Jean-Philippe Aumasson, Simon Fischer, Shahram Khazaei, Willi Meier, and Christian Rechberger. New features of latin dances: Analysis of Salsa, ChaCha, and Rumba. In Kaisa Nyberg, editor, *FSE 2008*, volume 5086 of *LNCS*, pages 470–488. Springer, Heidelberg, Germany, 2008.

[3] Arka Rai Choudhuri and Subhamoy Maitra. Significantly improved multi-bit differentials for reduced round Salsa and ChaCha. *IACR Trans. Symm. Cryptol.*, 2016(2):261–287, 2016.

[4] CNET. Logic friday. `https://download.cnet.com/Logic-Friday/3000-20415_4-75848245.html/`.

[5] Patrik Ekdahl, Thomas Johansson, Alexander Maximov, and Jing Yang. A new SNOW stream cipher called SNOW-V. *IACR Trans. Symmetric Cryptol.*, 2019(3):1–42, 2019.

[6] Patrik Ekdahl, Thomas Johansson, Alexander Maximov, and Jing Yang. SNOW-Vi: an extreme performance variant of SNOW-V for low-end CPUs. Cryptology ePrint Archive, Report 2021/236, 2021. `https://eprint.iacr.org/2021/236`.

[7] Kai Fu, Meiqin Wang, Yinghua Guo, Siwei Sun, and Lei Hu. MILP-Based Automatic Search Algorithms for Differential and Linear Trails for Speck. In Thomas Peyrin, editor, *Fast Software Encryption - 23rd International Conference, FSE 2016*, volume 9783 of *Lecture Notes in Computer Science*, pages 268–288. Springer, 2016.

[8] Yuki Funabiki, Yosuke Todo, Takanori Isobe, and Masakatu Morii. Several MILP-Aided Attacks Against SNOW 2.0. In Jan Camenisch and Panos Papadimitratos, editors, *Cryptology and Network Security - 17th International Conference, CANS 2018*, volume 11124 of *Lecture Notes in Computer Science*, pages 394–413. Springer, 2018.

[9] Jin Hoki, Takanori Isobe, Ryoma Ito, Fukang Liu, and Kosei Sakamoto. Distinguishing and Key Recovery Attacks on the Reduced-Round SNOW-V. Cryptology ePrint Archive, Report 2021/546 (Version: 20210427:061034), 2021. `https://eprint.iacr.org/2021/546`.

[10] Gurobi Optimization Inc. Gurobi optimizer 9.0, 2019. `http://www.gurobi.com/`.

[11] Lin Jiao, Yongqiang Li, and Yonglin Hao. A Guess-And-Determine Attack On SNOW-V Stream Cipher. *The Computer Journal*, 2020.

[12] Itsik Mantin and Adi Shamir. A practical attack on broadcast RC4. In Mitsuru Matsui, editor, *FSE 2001*, volume 2355 of *LNCS*, pages 152–164. Springer, Heidelberg, Germany, 2002.

[13] Bing Sun, Zhiqiang Liu, Vincent Rijmen, Ruilin Li, Lei Cheng, Qingju Wang, Hoda AlKhzaimi, and Chao Li. Links Among Impossible Differential, Integral and Zero Correlation Linear Cryptanalysis. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference*, volume 9215 of *Lecture Notes in Computer Science*, pages 95–115. Springer, 2015.

[14] Ling Sun, Wei Wang, and Meiqin Wang. MILP-Aided Bit-Based Division Property for Primitives with Non-Bit-Permutation Linear Layers. *IACR Cryptol. ePrint Arch.*, 2016:811, 2016.

[15] Ling Sun, Wei Wang, and Meiqin Wang. Automatic Search of Bit-Based Division Property for ARX Ciphers and Word-Based Division Property. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on teh Theory and Applications of Cryptology and Information Security*, volume 10624 of *Lecture Notes in Computer Science*, pages 128–157. Springer, 2017.

[16] Yosuke Todo. Structural Evaluation by Generalized Integral Property. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on teh Theory and Applications of Cryptographic Techniques*, volume 9056 of *Lecture Notes in Computer Science*, pages 287–314. Springer, 2015.

[17] Yosuke Todo and Masakatu Morii. Bit-Based Division Property and Application to Simon Family. In Thomas Peyrin, editor, *Fast Software Encryption - 23rd International Conference, FSE 2016*, volume 9783 of *Lecture Notes in Computer Science*, pages 357–377. Springer, 2016.

[18] Zejun Xiang, Wentao Zhang, Zhenzhen Bao, and Dongdai Lin. Applying MILP Method to Searching Integral Distinguishers Based on Division Property for 6 Lightweight Block Ciphers. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on teh Theory and Application of Cryptology and Information Security*, volume 10031 of *Lecture Notes in Computer Science*, pages 648–678, 2016.

[19] Jing Yang, Thomas Johansson, and Alexander Maximov. New improved attacks on SNOW-V. Cryptology ePrint Archive, Report 2021/544, 2021. `https://eprint.iacr.org/2021/544`.