PAPER Special Section on Information and Communication System Security

PNB Based Differential Cryptanalysis of Salsa20 and ChaCha*

Nasratullah GHAFOORI[†a)], ***Nonmember***, **Atsuko MIYAJI**[†b)], ***Member***, **Ryoma ITO**[††c)], ***and*** **Shotaro MIYASHITA**[†d)], ***Nonmembers***

SUMMARY This paper introduces significant improvements over the existing cryptanalysis approaches on Salsa20 and ChaCha stream ciphers. For the first time, we reduced the attack complexity on Salsa20/8 to the lowest possible margin. We introduced an attack on ChaCha7.25. It is the first attack of its type on ChaCha7.25/20. In our approach, we studied differential cryptanalysis of the Salsa20 and ChaCha stream ciphers based on a comprehensive analysis of probabilistic neutral bits (PNBs). The existing differential cryptanalysis approaches on Salsa20 and ChaCha stream ciphers first study the differential bias at specific input and output differential positions and then search for *probabilistic neutral bits*. However, the differential bias and the set of PNBs obtained in this method are not always the ideal combination to conduct the attack against the ciphers. The researchers have not focused on the comprehensive analysis of the probabilistic neutrality measure of all key bits concerning all possible output difference positions at all possible internal rounds of Salsa20 and ChaCha stream ciphers. Moreover, the relationship between the neutrality measure and the number of inverse quarter rounds has not been scrutinized yet. To address these study gaps, we study the *differential cryptanalysis based on the comprehensive analysis of probabilistic neutral bits* on the reduced-round Salsa20 and ChaCha. At first, we comprehensively analyze the neutrality measure of 256 key bits positions. Afterward, we select the output difference bit position with the best average neutrality measure and look for the corresponding input differential with the best differential bias. Considering all aspects, we present an attack on Salsa20/8 with a time complexity of $2^{241.62}$ and data complexity of $2^{31.5}$, which is the best-known single bit differential attack on Salsa20/8 and then, we introduced an attack on ChaCha7.25 rounds with a time complexity of $2^{254.011}$ and data complexity of $2^{51.81}$.

key words: *stream cipher, Salsa20, ChaCha, differential cryptanalysis, PNBs*

1. Introduction

This paper is the extended version of the author's submission to the 17*th* International Conference on Information Security Practice and Experience (ISPEC 2022). We summarize the extension below.

- We extended the proposed attack to ChaCha stream cipher and introduced the first and best-known attack on ChaCha7.25.
- In this study, we introduced the differential attack on Salsa20/8 with a time complexity of $2^{241.62}$ and data complexity of $2^{31.5}$. We improved the time complexity of the attack on Salsa20/8 by a factor of $2^{2.08}$.
- In our previous study, we used only the 5*th* internal round to attack Salsa20/8. As the higher internal rounds increase the number of neutral bits and consequently reduce the attack complexity. In this study, we examined all feasible internal rounds and introduced an optimal internal round to attack Salsa20/8. We studied $4r, 4.25r, 4.5r, 4.75r, 5r, 5.25r, 5.5r$, and $5.75r$ internal rounds and presented the 4.75*th* as an optimal internal round to attack Salsa20/8.
- We introduced the upper bound for the maximum number of forward and inverse rounds to attack Salsa20/8 and ChaCha stream ciphers.

1.1 Background

Salsa20 and ChaCha stream ciphers were designed by Daniel J. Bernstein in April 2005 [1] and January 2008 [2] with a 256-bit security level against key-recovery attacks and 20 rounds. Both ciphers provide a 128 key bits version as well. Salsa20 and ChaCha ciphers are constructed based on the *Addition, Rotation, and exclusive-OR* [ARX] structure. The ARX structure security leans on modular addition, which generates non-linearity. The rotations help the ARX structure to provide faster diffusion and mix the bits on the left and right sides of a word. Both ciphers deployed widely in protocols, networks, operating Systems, software [**]. Considering its wide adaption, it's important to study the security of mentioned ciphers. The designer submitted the 20-round Salsa20 stream cipher to the ECRYPT Stream Cipher Project, eSTREAM [3], as a candidate for stream ciphers for software applications with high throughput requirements and hardware applications with restricted resources. The eSTREAM portfolio was completed in September 2008. Eventually, the 12-round Salsa20, Salsa20/12, was selected as one of the finalists for the eSTREAM software portfolio. ChaCha also has a variant of 12 rounds. However, JP-Aumasson [4] proposed ChaCha's 8-rounds instead of 20 with no security risk. It would yield a 2.5× speedup. Following the release of Salsa20

Manuscript received November 9, 2022.
Manuscript revised April 2, 2023.
Manuscript publicized July 13, 2023.

[†]The authors are with the Graduate School of Engineering, Osaka University, Suita-shi, 565–0871 Japan.
[††]The author is with National Institute of Information and Communications Technology, Koganei-shi, 184-8795 Japan.

*This is the extended version of our submission to ISPEC 2022.

a) E-mail: ghafoori@cy2sec.comm.eng.osaka-u.ac.jp
b) E-mail: miyaji@comm.eng.osaka-u.ac.jp
c) E-mail: itorym@nict.go.jp
d) E-mail: miyashita@cy2sec.comm.eng.osaka-u.ac.jp

DOI: 10.1587/transinf.2022ICP0015

[**]https://ianix.com/pub/salsa20-deployment.html
https://ianix.com/pub/chacha-deployment.html

and ChaCha, numerous studies have been conducted on the security evaluations of Salsa20 and ChaCha [5]–[24]. The most consequential of these existing studies is the differential attack based on a concept called *probabilistic neutral bits* PNBs, proposed by Aumasson et al. at FSE 2008 [5]. PNB is a concept that divides secret key bits into two subsets of significant key bits m and non-significant key bits n. The *neutral measure* is used as a threshold to contradistinguish these two subsets. The number of key bits in subset m and n significantly affect the complexity of the attack. Considering the stated fact, we have to study the PNBs in such a way as to understand the conditions and circumstances which affect the neutrality measure of key bits and decrease the elements in significant key bits subset m. Thus, it is a crucial task to analyze PNBs in detail for the differential attacks on Salsa20. The author in [5] first searched for the input/output differential pair with the best differential bias; then, based on the obtained input/output differential pair, they divided secret key bits into two subsets by applying the concept of PNBs; finally, they performed a differential attack on the 8-round version of Salsa20, and 7-round of ChaCha. They introduced attack on Salsa8 with a time complexity of 2^{251} and data complexity of 2^{31}. Similarly, they present an attack on ChCha7 with a time complexity of 2^{248} and data complexity of 2^{27}. Thenceforth, several studies have been reported on the improvements of their proposed attack [8], [9], [11], [13], [14], [23], [24]. Allegedly, the best single-bit differential attack on Salsa20/8 with a time complexity of $2^{243.67}$ was proposed by Dey and Sarkar [9]. We are the first to put forward an attack on ChaCha7.25 using $r = 3.5r$ internal round. As mentioned, the existing differential attacks on Salsa20 and ChaCha have focused on searching for the input/output differential pair with the best differential bias. However, the differential bias and PNBs obtained in the existing attacks are not always the best combination. As the *Probabilistic Neutral Bits* and differential bias affect the time complexity and data complexity of an attack. Thus, we study the conditions that increase the number of (PNBs) and improve the differential bias.

1.2 Our Contributions

In this study, for the first time, we apply the *differential cryptanalysis based on the comprehensive analysis of PNBs* on the Salsa20/8 and ChaCha7.25. This study first deeply investigates the *neutrality measure* of all key bits positions considering different internal rounds and all possible 512 output difference bits $\mathcal{OD}$. Then it looks for differential bias. To further explore, the attack can be applied on the reduced-round Salsa20 by (1) comprehensively analyzing the output difference $\mathcal{OD}$ bit position with the highest average neutral measures considering 256 key bits positions and (2) looking for the input difference $\mathcal{ID}$ bit position with the best differential bias in the obtained output difference $\mathcal{OD}$ bit position. This research aims to find the combination of $\mathcal{ID}, \mathcal{OD}$ pair with the best differential bias and the subset of *PNBs* through an extensive analysis of the neutrality measure of all key bits positions. The contributions of this study are summarized below:

- We study the relationship between the modular addition and *probabilistic neutral bits* in different internal rounds of Salsa20/8. We show that the Salsa20 inverse quarter-round function impact in $4.25r - 4.75r$ and $5.5r - 5.75r$ are the same. It is a weakness of inverse quarter round. It allows the adversary to attack higher rounds of Salsa20.
- Through an extensive analysis of *PNBs*, for the first time, we illustrate the distribution of neutral measures in different internal rounds of Salsa20 and ChaCha stream ciphers. Moreover, we experimentally and theoretically demonstrate that the neutrality measures vary greatly depending on output differential $\mathcal{OD}$ bit position, not on input differential $\mathcal{ID}$ bit position.
- We introduce the best internal round and $\mathcal{OD}$ position to attack Salsa20/8 and ChaCha7.25. To be precise, we found that the $27th$ bit of the $8th$ word in the $4.75r$ internal round of Salsa20/8 is the best $\mathcal{OD}$ position to attack Salsa20/8. We used the mentioned $\mathcal{OD}$ position and internal round in our proposed attack. Furthermore, We used the $3.5r$ internal round and $0th, 1st, 2nd, 3rd$ $\mathcal{OD}$ positions to attack ChaCha7.25.
- We searched for the best differential bias at all possible $\mathcal{ID}$ positions (i.e., 128 bits) given the selected $\mathcal{OD}$ position. By analysis of the differential bias at the obtained output differential bit position in detail, we found that the $6th$ bit of the $13th$ word gives the best differential bias. For ChaCha, we used the $12th, 13th, 14th, 15th$ $\mathcal{ID}$ positions to attack ChaCha7.25.
- Based on the combination of the obtained differential bias at a specific $\mathcal{ID}, \mathcal{OD}$ position, and the subset of PNBs, we present a differential attack on the Salsa20/8 with a time complexity of $2^{241.62}$ and data complexity of $2^{31.5}$, which is the best differential attack reported. In addition, We could attack ChaCha7.25 with a time complexity of $2^{254.011}$ and data complexity of $2^{51.81}$
- One can use our proposed cryptanalysis method to attack Salsa20/7 and ChaCha7 or lower rounds. However, in this paper, we focused only on Salsa20/8 and ChaCha7.25.

Table 1 Summary of the proposed and existing attacks on Salsa20 with 256-key bits security

Target	Time	Data	Reference
Salsa20/7	2^{151}	2^{26}	[5]
	2^{148}	2^{24}	[14]
	2^{137}	2^{61}	[8]
Salsa20/8	2^{251}	2^{31}	[5]
	2^{250}	2^{27}	[14]
	$2^{245.5}$	$2^{22.4}$	[13]
	$2^{244.9}$	2^{96}	[8]
	$2^{243.6}$	$2^{30.4}$	[9]
	$\mathbf{2^{241.62}}$	$\mathbf{2^{31.5}}$	This work

Table 2 Summary of the proposed and existing attacks on ChaCha with 256-key bits security

Target	Time	Data	Reference
ChaCha20/6	2^{139}	2^{30}	[5]
	2^{136}	2^{28}	[14]
	$2^{127.5}$	$2^{37.5}$	[8]
	$2^{77.4}$	2^{58}	[18]
	2^{51}	2^{51}	[20]
ChaCha20/7	2^{248}	2^{27}	[5]
	$2^{246.5}$	2^{27}	[14]
	$2^{238.9}$	$2^{23.8}$	[13]
	$2^{237.7}$	2^{96}	[8]
	$2^{231.9}$	2^{50}	[24]
	$2^{230.86}$	$2^{48.8}$	[18]
	$2^{221.95}$	$2^{90.20}$	[23]
ChaCha20/7.25	$2^{255.62}$	$2^{48.36}$	[7]
	$\mathbf{2^{254.011}}$	$\mathbf{2^{51.81}}$	This Work

1.3 Organization of the Paper

The remainder of this paper is structured as follows. In Sect. 2, we briefly describe the specification of the Salsa20 and ChaCha stream ciphers. In Sect. 3, we review generic techniques of differential cryptanalysis and the concept of probabilistic neutral bits. In Sect. 4, we introduce the differential cryptanalysis based on the comprehensive analysis of *PNBs*. In Sect. 5, we present the result of our cryptanalysis approach on Salsa20 stream cipher. In addition, we show the theoretical and experimental results associated with the extensive study of *PNBs* and then discuss the structure of the Salsa20 stream cipher and how it impacts the *PNBs*. In Sect. 6 we present the attack impact on ChaCha stream cipher and the distribution of neutrality measures in different target and internal rounds of ChaCha. In Sect. 7 we discuss and differentiate our proposed approach and the existing attack on Salsa20 and ChaCha and summarize our results. Finally, Sect. 8 concludes this research work.

2. Specification of Ciphers and Preliminaries

2.1 Specification of Salsa20

Salsa20 stream cipher consists of the following three steps to generate a keystream block of 16 words, where each word size is 32 bits:

Step 1. The initial state matrix $X^{(0)}$ of order 4×4 is initialized from a 256-bit secret key $k = (k_0, k_1, \ldots, k_7)$, a 64-bit nonce $v = (v_0, v_1)$, a 64-bit block counter $t = (t_0, t_1)$, and four 32-bit constants $c = (c_0, c_1, c_2, c_3)$, such as c_0 = 0x61707865, c_1 = 0x3320646e, c_2 = 0x79622d32, and c_3 = 0x6b206574. It is worthwhile to mention that a 128-bit key length could be used (not recommended) to form the initial state matrix of Salsa20. That being so, the constant words change to c_0 = 0x61707865, c_1 = 0x3120646e, c_2 = 0x79622d36, and c_3 = 0x6b206574. For more detail about constant words considering the key size please refer to Sect. 4 [1]. After the initialization, we obtain the following initial state matrix $X^{(0)}$:

$$X^{(0)} = \begin{pmatrix} x_0^{(0)} & x_1^{(0)} & x_2^{(0)} & x_3^{(0)} \\ x_4^{(0)} & x_5^{(0)} & x_6^{(0)} & x_7^{(0)} \\ x_8^{(0)} & x_9^{(0)} & x_{10}^{(0)} & x_{11}^{(0)} \\ x_{12}^{(0)} & x_{13}^{(0)} & x_{14}^{(0)} & x_{15}^{(0)} \end{pmatrix} = \begin{pmatrix} c_0 & k_0 & k_1 & k_2 \\ k_3 & c_1 & v_0 & v_1 \\ t_0 & t_1 & c_2 & k_4 \\ k_5 & k_6 & k_7 & c_3 \end{pmatrix}.$$

Step 2. The round function of Salsa20 consists of four computations of the so-called quarter-round function. According to the procedure of the quarter-round function, a vector $(x_a^{(r)}, x_b^{(r)}, x_c^{(r)}, x_d^{(r)})$ in the internal state matrix $X^{(r)}$ is updated by sequentially computing

$$\begin{cases} x_b^{(r+1)} = ((x_a^{(r)} + x_d^{(r)}) \lll 7) \oplus x_b^{(r)}, \\ x_c^{(r+1)} = ((x_b^{(r+1)} + x_a^{(r)}) \lll 9) \oplus x_c^{(r)}, \\ x_d^{(r+1)} = ((x_c^{(r+1)} + x_b^{(r+1)}) \lll 13) \oplus x_d^{(r)}, \\ x_a^{(r+1)} = ((x_d^{(r+1)} + x_c^{(r+1)}) \lll 18) \oplus x_a^{(r)}, \end{cases} \quad (1)$$

where the symbols '+', '$\lll$', and '$\oplus$' represent word-wise modular addition, bit-wise left rotation, and bit-wise XOR, respectively. In odd number rounds, which are called column-rounds, the quarter-round function is applied to the following four column vectors:$(x_0^{(r)}, x_4^{(r)}, x_8^{(r)}, x_{12}^{(r)})$, $(x_5^{(r)}, x_9^{(r)}, x_{13}^{(r)}, x_1^{(r)})$, $(x_{10}^{(r)}, x_{14}^{(r)}, x_2^{(r)}, x_6^{(r)})$, and $(x_{15}^{(r)}, x_3^{(r)}, x_7^{(r)}, x_{11}^{(r)})$. In even number rounds, which are called row-rounds, the quarter-round function is applied to the following four row vectors: $(x_0^{(r)}, x_1^{(r)}, x_2^{(r)}, x_3^{(r)})$, $(x_5^{(r)}, x_6^{(r)}, x_7^{(r)}, x_4^{(r)})$, $(x_{10}^{(r)}, x_{11}^{(r)}, x_8^{(r)}, x_9^{(r)})$, and $(x_{15}^{(r)}, x_{12}^{(r)}, x_{13}^{(r)}, x_{14}^{(r)})$.

Step 3. A 512-bit keystream block is generated as $Z = X^{(0)} + X^{(R)}$, where R is the final round. The original version of Salsa20 stream cipher, called Salsa20, is $R = 20$ rounds, however, the accepted version as one of the finalists for the eSTREAM software portfolio [3] is Salsa20/12, where $R = 12$.

Table 3 Notations

Notation	Description
X	The Salsa20 matrix of 4×4 with 16 words of 32 bit each
$X^{(0)}$	The initial state matrix of Salsa20
$X'^{(0)}$	The associate matrix with a single bit difference at $x_{i,j}$ position.
$X^{(R)}$	The matrix after Salsa20 R rounds
$X^{(r)}$	The matrix after Salsa20 r rounds where $R > r$ (internal round)
$x_i^{(R)}$	The i^{th} word of state matrix $X^{(R)}$
$x_{i,j}^{(R)}$	The j^{th} bit of i^{th} word of matrix $X^{(R)}$
$x + y$	The word-wise addition of word x and y *modulo* 2^{32}
$x - y$	The word-wise subtraction of word x and y *modulo* 2^{32}
$x \bigoplus y$	Bit-wise XOR operation of the word x and y
$x \lll n$	The left rotation of word x by n bits
Δx	The XOR difference of word x and x' defined as $\Delta x = x \bigoplus x'$
$\varepsilon_e, \varepsilon_a$	The forward and backward differential bias of Salsa20.

The round function of Salsa20 is reversible., i.e., a vector $(x_a^{(r+1)}, x_b^{(r+1)}, x_c^{(r+1)}, x_d^{(r+1)})$ in the internal state matrix $X^{(r+1)}$ is reversed by sequentially computing:

$$\begin{cases} x_a^{(r)} = ((x_d^{(r+1)} + x_c^{(r+1)}) \lll 18) \oplus x_a^{(r+1)}, \\ x_d^{(r)} = ((x_c^{(r+1)} + x_b^{(r+1)}) \lll 13) \oplus x_d^{(r+1)}, \\ x_c^{(r)} = ((x_b^{(r+1)} + x_a^{(r)}) \lll 9) \oplus x_c^{(r+1)}, \\ x_b^{(r)} = ((x_a^{(r)} + x_d^{(r)}) \lll 7) \oplus x_b^{(r+1)}. \end{cases} \quad (2)$$

Proceedings all other in time, we studied the differential cryptanalysis and distribution of neutrality measure in all possible internal rounds of Salsa20. We worked on the neutrality measure in the Salsa20 full round, half round, quarter round, and three-quarter rounds defined as the following sequentially.

$$\left\{ x_b^{(r+1)} = ((x_a^{(r)} + x_d^{(r)}) \lll 7) \oplus x_b^{(r)} \right. \tag{3}$$

$$\begin{cases} x_b^{(r+1)} = ((x_a^{(r)} + x_d^{(r)}) \lll 7) \oplus x_b^{(r)} \\ x_c^{(r+1)} = ((x_b^{(r+1)} + x_a^{(r)}) \lll 9) \oplus x_c^{(r)} \end{cases} \tag{4}$$

$$\begin{cases} x_b^{(r+1)} = ((x_a^{(r)} + x_d^{(r)}) \lll 7) \oplus x_b^{(r)} \\ x_c^{(r+1)} = ((x_b^{(r+1)} + x_a^{(r)}) \lll 9) \oplus x_c^{(r)} \\ x_d^{(r+1)} = ((x_c^{(r+1)} + x_b^{(r+1)}) \lll 13) \oplus x_d^{(r)} \end{cases} \tag{5}$$

In addition, the inverse quarter round, half round, and three-quarters round are defined as the following sequentially.

$$\left\{ x_a^{(r)} = ((x_d^{(r+1)} + x_c^{(r+1)}) \lll 18) \oplus x_a^{(r+1)} \right. \tag{6}$$

$$\begin{cases} x_a^{(r)} = ((x_d^{(r+1)} + x_c^{(r+1)}) \lll 18) \oplus x_a^{(r+1)}, \\ x_d^{(r)} = ((x_c^{(r+1)} + x_b^{(r+1)}) \lll 13) \oplus x_d^{(r+1)}, \end{cases} \tag{7}$$

$$\begin{cases} x_a^{(r)} = ((x_d^{(r+1)} + x_c^{(r+1)}) \lll 18) \oplus x_a^{(r+1)}, \\ x_d^{(r)} = ((x_c^{(r+1)} + x_b^{(r+1)}) \lll 13) \oplus x_d^{(r+1)}, \\ x_c^{(r)} = ((x_b^{(r+1)} + x_a^{(r)}) \lll 9) \oplus x_c^{(r+1)}, \end{cases} \tag{8}$$

2.2 Specification of ChaCha

ChaCha stream cipher consists of the following three steps to generate a keystream block of 16 words, where each word size is 32 bits:

Step 1. To generate 512 bits key stream, ChaCha initial state matrix $X^{(0)}$ of order 4×4 is initialized from a 256-bit secret key $k = (k_0, k_1, \ldots, k_7)$, a 96-bit nonce $v = (v_0, v_1, v_2)$, a 32-bit block counter t_0, and four 32-bit constants $c = (c_0, c_1, c_2, c_3)$, such as $c_0 =$ 0x61707865, $c_1 =$ 0x3320646e, $c_2 =$ 0x79622d32, and $c_3 =$ 0x6b206574. After initialization, we obtain the following initial state matrix:

$$X^{(0)} = \begin{pmatrix} x_0^{(0)} & x_1^{(0)} & x_2^{(0)} & x_3^{(0)} \\ x_4^{(0)} & x_5^{(0)} & x_6^{(0)} & x_7^{(0)} \\ x_8^{(0)} & x_9^{(0)} & x_{10}^{(0)} & x_{11}^{(0)} \\ x_{12}^{(0)} & x_{13}^{(0)} & x_{14}^{(0)} & x_{15}^{(0)} \end{pmatrix} = \begin{pmatrix} c_0 & c_1 & c_2 & c_3 \\ k_0 & k_1 & k_2 & k_3 \\ k_4 & k_5 & k_6 & k_7 \\ t_0 & v_0 & v_1 & v_2 \end{pmatrix}.$$

Step 2. The round function of ChaCha comprises four simultaneous computations of the quarterround function. According to the procedure, a vector $(x_a^{(r)}, x_b^{(r)}, x_c^{(r)}, x_d^{(r)})$ in the internal state matrix $X^{(r)}$ is updated by sequentially computing the following:

$$\begin{cases} x_{a'}^{(r)} = x_a^{(r)} + x_b^{(r)} & x_a^{(r+1)} = x_{a'}^{(r)} + x_{b''}^{(r)} \\ x_{d'}^{(r)} = x_d^{(r)} \oplus x_{a'}^{(r)} & x_{d'''}^{(r)} = x_{d''}^{(r)} \oplus x_a^{(r+1)} \\ x_{d''}^{(r)} = x_{d'}^{(r)} \lll 16 & x_d^{(r+1)} = x_{d'''}^{(r)} \lll 8 \\ x_{c'}^{(r)} = x_c^{(r)} + x_{d''}^{(r)} & x_c^{(r+1)} = x_{c'}^{(r)} + x_d^{(r+1)} \\ x_{b'}^{(r)} = x_b^{(r)} \oplus x_{c'}^{(r)} & x_{b'''}^{(r)} = x_{b''}^{(r)} \oplus x_c^{(r+1)} \\ x_{b''}^{(r)} = x_{b'}^{(r)} \lll 12 & x_b^{(r+1)} = x_{b'''}^{(r)} \lll 7 \end{cases} \tag{9}$$

The symbols "+", "$\oplus$", and "$\lll$" represent word-wise modular addition, bitwise XOR, and bitwise left rotation, respectively. For odd-numbered rounds, which are called columnrounds, the quarterround function is applied to the following four column vectors: $(x_0^{(r)}, x_4^{(r)}, x_8^{(r)}, x_{12}^{(r)})$, $(x_1^{(r)}, x_5^{(r)}, x_9^{(r)}, x_{13}^{(r)})$, $(x_2^{(r)}, x_6^{(r)}, x_{10}^{(r)}, x_{14}^{(r)})$, and $(x_3^{(r)}, x_7^{(r)}, x_{11}^{(r)}, x_{15}^{(r)})$. For even-numbered rounds, which are called diagonalrounds, the quarterround function is applied to the following four diagonal vectors: $(x_0^{(r)}, x_5^{(r)}, x_{10}^{(r)}, x_{15}^{(r)})$, $(x_1^{(r)}, x_6^{(r)}, x_{11}^{(r)}, x_{12}^{(r)})$, $(x_2^{(r)}, x_7^{(r)}, x_8^{(r)}, x_{13}^{(r)})$, and $(x_3^{(r)}, x_4^{(r)}, x_9^{(r)}, x_{14}^{(r)})$.

Step 3. A 512-bit keystream block is computed as $Z = X^{(0)} + X^{(R)}$, where R is the final round. The original version of ChaCha has $R = 20$ rounds, and the ChaCha20/R denotes the reduced-round version of ChaCha.

The round function of ChaCha is reversible. In other words, an input vector $(x_a^{(r+1)}, x_b^{(r+1)}, x_c^{(r+1)}, x_d^{(r+1)})$ in the internal state matrix $X^{(r+1)}$ is backdated by sequentially computing the following:

$$\begin{cases} x_{b'''}^{(r)} = x_b^{(r+1)} \lll 25, \quad x_{b''}^{(r)} = x_{b'''}^{(r)} \oplus x_c^{(r+1)}, \quad x_{c'}^{(r)} = x_c^{(r+1)} - x_d^{(r+1)}, \\ x_{d'''}^{(r)} = x_d^{(r+1)} \lll 24, \quad x_{d''}^{(r)} = x_{d'''}^{(r)} \oplus x_a^{(r+1)}, \quad x_{a'}^{(r)} = x_a^{(r+1)} - x_{b''}^{(r)}, \\ x_{b'}^{(r)} = x_{b''}^{(r)} \lll 20, \quad x_b^{(r)} = x_{b'}^{(r)} \oplus x_{c'}^{(r)}, \quad x_c^{(r)} = x_{c'}^{(r)} - x_{d''}^{(r)}, \\ x_{d'}^{(r)} = x_{d''}^{(r)} \lll 16, \quad x_d^{(r)} = x_{d'}^{(r)} \oplus x_{a'}^{(r)}, \quad x_a^{(r)} = x_{a'}^{(r)} - x_b^{(r)} \end{cases} \tag{10}$$

We studied the differential cryptanalysis and distribution of neutrality measures in all possible internal rounds (i.e., $3r$ and $3.5r$) of ChaCha stream cipher. We worked on the distribution of neutrality measures in the ChaCha full round, half round, quarter round, and three-quarter round.

3. Differential Cryptanalysis

Differential cryptanalysis is a common method of attack on symmetric key cryptography. Biham, E. and A. Shamir [25] introduced differential cryptanalysis. In the beginning, it broke the DES cipher. Since then, it has changed to one of the main cryptanalysis models and is utilized to attack and evaluate the security of different encryption schemes. It is mainly a chosen plain text attack that aims to study the propagation of an *input difference* through several rounds of an encryption scheme. The cryptanalysts can take advantage of the non-randomness to fully or partially recover the secret key. The *XOR* operation computes the difference. The cryptanalysts are interested in searching for *input* and *output* differences denoted by Δ_x and Δ_z or by α and β, respectively. The XOR differential probability of addition xdp^+ and the additive differential probability of XOR $adp^{\oplus}$ were studied by [30]. The differential probability (DP) of addition modulo 2^n is the probability at which the input difference propagates to the output difference.

$$DP^{+}(\delta) = DP^{+}(\alpha, \beta \mapsto \delta) := P_{x,y}[(x+y)\oplus((x\oplus\alpha)+(y\oplus\beta)) = \delta] \tag{11}$$

The x and y are the inputs of size n. In this research, we study the differential cryptanalysis of the Salsa20 stream cipher where we send two initial state matrices of Salsa20 X and a copy of the initial state matrix with a single bit difference X' to the target round of the Salsa20 function. The Salsa20 function returns the interrelated states of X^R and X'^R. Moreover, we also study the intermediate rounds of Salsa20 denoted by X^r and X'^r where $R > r$.

3.1 Differential Cryptanalysis of Salsa20 Stream Cipher

The majority of attacks introduced by cryptanalysts on Salsa20 are differential attacks. Among proposals, Aumasson et al. at FSE 2008 [5] introduced the illustrious attack. The author proposed a differential attack based on *probabilistic neutral bits* (PNBs), which applied on the reduced round of Salsa20, Rumba, and ChaCha. In this section, we clarify generic techniques of differential attack based on PNB. The attack consists of two phases: pre-computation and online phases. In the pre-computation phase, the attacker examines single-bit differential biases and searches for the subset of PNBs. Subsequently, the adversary executes a probabilistic backward computation (PBC). Finally, the online phase recovers the unknown set of key bits.

3.1.1 Pre-computation Phase

At first, we initialize two-state matrices X and X'. Both X and X' matrices consist of the same keywords $(k_1, k_2...k_8)$ and constants. However, the X' matrix consists of a single bit difference in nonce v or counter t. To throw light on, let $x_i^{(0)}[j]$ be the j-th bit of the i-th word of initial state matrix $X^{(0)}$ for $0 \le i \le 15$ and $0 \le j \le 31$, and let $x_i'^{(0)}[j]$ be an associated word with a single bit difference at j^{th} bit of i^{th} word as $\Delta_i^{(0)}[j] = x_i^{(0)}[j] \oplus x_i'^{(0)}[j]$ be the difference. Given a difference $\Delta_i^{(0)}[j] = 1$ at the j^{th} bit of i^{th} word of state matrix $X^{(0)}$, which is called *input difference* or $\mathcal{ID}$, we obtain the corresponding initial state matrix X' as $v' = v \bigoplus \Delta v$ or $t' = t \bigoplus \Delta t$ where v' and t' denotes the single bit difference at nonce or counter. Next, we run the Salsa20 or ChaCha round function with the initial state matrices $X^{(0)}$, $X'^{(0)}$ as inputs, and obtain single bit output difference $\Delta_p^{(r)}[q] = x_p^{(r)}[q] \bigoplus x_p'^{(r)}[q]$ from the r-round internal state matrices $X^{(r)}$, $X'^{(r)}$, which is called *output difference* or $\mathcal{OD}$ where $1 \le r < R$ and q denotes the q-th bit of the p-th word of internal state matrix $X^{(r)}$ for $0 \le p \le 15$ and $0 \le q \le 31$ after r rounds of Salsa20 or ChaCha. For a fixed key and random nonces and block counters, the bias ε_d is defined as

$$\Pr(\Delta_p^{(r)}[q] = 1 \mid \Delta_i^{(0)}[j] = 1) = \frac{1}{2}(1 + \varepsilon_d), \tag{12}$$

The ε_d denotes the bias of the $\mathcal{OD}$. If the key bits are random, we compute the ε_d^* as a median value of ε_d [5]. To differentiate between the $\mathcal{OD}$ obtained from a true random number generator and the $\mathcal{OD}$ obtained from the r-round internal state matrices in Salsa20 or ChaCha, we use the following theorem proved by Mantin and Shamir at FSE 2001 [26].

Theorem 1 ([26]) *Let $\mathcal{X}$ and $\mathcal{Y}$ be two distributions, and suppose that the target event occurs in $\mathcal{X}$ with a probability p and $\mathcal{Y}$ with a probability $p \cdot (1 + q)$. Then, for small p and q, $O(\frac{1}{p \cdot q^2})$ samples suffice to distinguish $\mathcal{X}$ from $\mathcal{Y}$ with a constant probability of success. Let $\mathcal{X}$ and $\mathcal{Y}$ be two distributions. The event E in $\mathcal{X}$ happens with probability $\frac{1}{2}$, (i.e, the result of a true random number generator) and the event E' in $\mathcal{Y}$ happens with the probability $\frac{1}{2} \cdot (1 + \epsilon_d)$, (i.e, the $\mathcal{OD}$ obtained from the r-round internal matrices of Salsa20 or ChaCha stream ciphers). According to Theorem 1 and Eq. (12), the number of samples to distinguish $\mathcal{X}$ and $\mathcal{Y}$ is $O(\frac{2}{\epsilon_d^2})$ since p and q are equal to $\frac{1}{2}$ and ε_d, respectively.*

3.1.2 Probabilistic Neutral Bits

The probabilistic neutral bits concept allows us to divide the key bits set into two subsets. Throughout this paper, we call it *significant key bits* subset m and *non-significant key bits* subset n where $m = 256 - n$. To distinguish between the two aforementioned subsets, the PNB concept focuses on the amount of effect each key bit has on the output of the Salsa20 or ChaCha function called $\mathcal{OD}$ here. The effect of key bits is called *neutral measure*.

Definition 1 ([5]) The neutral measure of the key bit position γ_i with respect to the $\mathcal{OD}$ is defined as γ_κ, where $\frac{1}{2}(1 + \gamma_\kappa)$ is the probability that complementing the key bit κ at γ_i position does not change the $\mathcal{OD}$.

According to [5], the following singular cases of the neutral measure exist:

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

Algorithm 1 computes the neutrality measure of key bits in the Salsa20 or ChaCha stream ciphers.

Algorithm 1 [5]The key bits neutrality measure estimation

Require: Key bit position γ_i random(IV, Z, Z')
Ensure: The estimated neutrality measure γ_k of each key bit position γ_i

1. Compute the $(X^{(R)}, X'^{(R)})$ with $\Delta_i^{(0)}[j] = 1$.
2. Derive $Z = X^{(0)} + X^{(R)}$ and $Z' = X'^{(0)} + X'^{(R)}$.
3. Prepare $(\overline{X}^{(0)}, \overline{X'}^{(0)})$ with the key bit position γ_i flipped in $(X^{(0)}, X'^{(0)})$.
4. Compute $(Y^{(r)}, Y'^{(r)})$ with $Z - \overline{X}^{(0)}$ and $Z' - \overline{X'}^{(0)}$ as inputs to the inversed round function of Salsa20 or ChaCha.
5. Compute $\Gamma_p^{(r)}[q] = y_p^{(r)}[q] \oplus y_p'^{(r)}[q]$.
6. Repeatedly perform Steps 1-5 by using different initial state matrices with the same $\Delta_i^{(0)}[j] = 1$; compute the neutral measure as $\Pr(\Delta_p^{(r)}[q] = \Gamma_p^{(r)}[q] \mid \Delta_i^{(0)}[j] = 1) = \frac{1}{2}(1 + \gamma_k)$.
7. Set a threshold γ, and put all key bits with $\gamma_\kappa \ge \gamma$ into a set of n-bit non-significant key bits.

3.1.3 Probabilistic Backwards Computation (PBC)

Based on our discussion in Sect. 3.1.1, we calculate the forward differential bias ε_d by Eq. (12). Furthermore, we compute the r internal round differential bias from backward by probabilistic backward computation. Given Z, Z', X, X', one can reverse the Salsa20 or ChaCha keystream $Z = X + Round_R(X)$ and $Z' = X' + Round_R(X')$ to obtain the r rounds single bit differential bias from backward. To compute the PBC, we need the $Z - X$, and $Z' - X'$ matrices as inputs for the reverse round of Salsa20 or ChaCha explained in Eq. (10). Algorithm 2 explains the PBC steps. The bias $\varepsilon \approx \varepsilon_e \cdot \varepsilon_a$ under the independence assumption [5], [13].

Algorithm 2 [5]Probabilistic backward computation

Require: Random(X, X',Z, Z')
Ensure: The r internal round bias

1. Compute the R-round internal state matrix pair $(X^{(R)}, X'^{(R)})$ with the $\mathcal{ID}$ $\Delta_i^{(0)}[j] = 1$.
2. Derive the keystream matrices $Z = X^{(0)} + X^{(R)}$ and $Z' = X'^{(0)} + X'^{(R)}$.
3. Prepare $(\hat{X}^{(0)}, \hat{X'}^{(0)})$ initial states with only PNB bits reset to a fixed value from $(X^{(0)}, X'^{(0)})$.
4. Compute r-round internal state matrix pair $(\hat{Y}^{(r)}, \hat{Y}'^{(r)})$ with $Z - \hat{X}^{(0)}$ and $Z' - \hat{X'}^{(0)}$ as inputs to the inversed round function of Salsa20 or ChaCha.
5. Compute $\hat{\Gamma}_p^{(r)}[q] = \hat{y}_p^{(r)}[q] \oplus \hat{y}_p'^{(r)}[q]$ for all possible choices of p and q, where $\hat{y}_p^{(r)}[q]$ and $\hat{y}_p'^{(r)}[q]$ are the q-th bit of the p-th word of $\hat{Y}^{(r)}$ and $\hat{Y}'^{(r)}$, respectively.
6. Repeatedly perform Steps 1-5 by using different initial state matrices with the same $\Delta_i^{(0)}[j] = 1$; compute the r-round backward bias ε_a as $\Pr(\Delta_p^{(r)}[q] = \hat{\Gamma}_p^{(r)}[q] \mid \Delta_i^{(0)}[j] = 1) = \frac{1}{2}(1 + \varepsilon_a)$.

3.1.4 Online Phase

According to [5], for the online phase, we run Algorithm 3. We need the following parameters as an input to the algorithm: $\mathcal{OD}$ word p $\mathcal{OD}$ bit q, $\mathcal{ID}$ in nonce Δ_V, the subset of significant key bits m concerning some threshed γ and N of keystream block to recover the secret key with some probability. For a detailed understanding, one can also refer to Sect. 2.5 of [27].

3.1.5 Complexity Estimation

Once we found an optimal $\mathcal{ID}, \mathcal{OD}$ pair, and defined $(\Delta_p^{(r)}[q] = 1 \mid \Delta_i^{(0)}[j] = 1)$. The set of key bits is divided into significant key bits m and non-significant key bits n and computed the median bias ε^*. To calculate the time complexity of the attack, step 2 is repeated for all possible 2^m subkey candidates. Step 2-1 and 2-2 execution would require N keystream. Step 2-4 is executed with the probability $P_{fa} = 2^{-\alpha}$, which adds a cost of 2^n considering the obtained bias. As result, the complexity of the attack from steps 2-3 to steps 2-5 is $2^n P_{fa}$ [27]. The total complexity calculation is as below.

Algorithm 3 [5] [27]Effective Attack

Require: p, q, r ΔV, m threshed γ and N
Ensure: Recovered Key

1. For an unknown key, collect N pair of keystream, where each pair is generated by random nonce and satisfy the $\mathcal{ID}$.
2. For each of m-bit significant key bit, run:
   - **2-1.** Set the significant key bits m of a key k to k_s and the non-significant key bits to a random value.
   - **2-2.** Obtain the r-round single-bit differential biases from backward with the subkey k_s for the N keystream block pairs.
   - **2-3.** If the bias validates the subkeys candidate k_s as a (possibly) correct one:
   - **2-4.** Run an additional exhaustive search over the n-bit PNB bits to confirm the correctness of the filtered subkey k_s and to find the n-bit non-significant key bits.
   - **2-5.** Stop if the correct key appears.

$$2^m(N + 2^n P_{fa}) = 2^m N + 2^{256-\alpha} \quad N \approx \left(\frac{\sqrt{\alpha \log 4} + 3\sqrt{1-\varepsilon^2}}{\varepsilon}\right)^2 \tag{13}$$

Practically, we choose α (and hence N) in a way to minimize the time complexity of the attack. In Sect. 5.5, we used the median bias ε^* to calculate N and subsequently the complexity of our proposed attack.

3.2 Related Works

In this subsection, we present the major differential attacks against the Salsa20 and ChaCha stream ciphers. In 2005, **Crowley** [6] presented the first differential attack on Salsa20. The author found a 3-round differential and presented an attack on Salsa20/5 with a complexity of 2^{165}. Later in 2006, The **Fischer et al** [10] reported a 4-round differential and presented a key-recovery attack on Salsa20/6 with 2^{177} trials using 2^{16} pairs of keystream. Fischer et al used the χ^2 test to measure the statistical weaknesses of Salsa20. In 2007, **Tsunoo et al** [28] reported the bias in 4-round Salsa20 and used the obtained bias to attack the $5-6-7-8$ rounds of Salsa20. Tsunoo et al reported an attack on Salsa20/7 with 2^{190} trials. In 2008, **Jean-Philippe Aumasson** [5] introduced the most important cryptanalysis attack on reduced rounds of Salsa20 and ChaCha. Jean-Philippe Aumasson presented an attack on Salsa20/8 and ChaCha7 with time and data complexity of 2^{255}, 2^{31}, 2^{248}, and 2^{27} respectively. **Zhenqing Shi** [14] introduced the concept of column chaining distinguisher (CCD) and probabilistic neutral vector (PNV). The author attack Salsa20/8 with a time complexity of 2^{250} and ChaCha7 with a time complexity of $2^{246.5}$ and data complexity of 2^{27}. In 2015, **Maitra** [29] revisited the idea of (PNBs) and explained certain parameters to reduce the complexity of the existing attack. The paper achieved the key search with the complexity of $2^{247.2}$. In 2016, **Maitra** [13] introduced the chosen IV attack on Salsa20 and Chacha stream ciphers. The author reduced the complexity of the attack on Salsa20/8 to $2^{245.5}$ from $2^{247.2}$ and attacked ChaCha7 with time complexity of $2^{238.94}$ and data complexity of $2^{23.89}$. In 2016 **Choudhuri** [8], intro-

duced the multi-bit differential cryptanalysis on Salsa20 and ChaCha, with their proposed adversary model, they introduced an attack on Salsa20/8 with time complexity of $2^{244.85}$ and an attack on ChaCha7 with time complexity of $2^{237.65}$ and data complexity $2^{31.6}$. In 2017, **Sabyasachi Dey** [9] improved the attack on Salsa20 by adding more PNBs. The author attack Salsa20/8 with time complexity $2^{243.6}$. In 2021, **Christof** [18] proposed a differential linear adversary framework for ARX ciphers and introduced an attack on ChaCha7 with a time complexity $2^{230.86}$ and data complexity of $2^{48.83}$. In 2021, **Miyashita and Ito** [7] applied the PNB-focused differential cryptanalysis on the ChaCha stream cipher. The proposed approach introduced the attack on ChaCha 7.25 – *rounds* with time complexity of $2^{255.62}$ and data complexity $2^{48.36}$. In 2022 **Dey** [23] improved the differential linear cryptanalysis and proposed an attack on ChaCha7 with a time complexity of 2^{221} and data complexity of 2^{90}. Considering the importance of the Salsa20 and ChaCha stream ciphers, their wide adoption, and the structure. We further studied the adversary model proposed by Miyashita and Ito [7] on Salsa20 and ChaCha. We applied the differential cryptanalysis based on extensive analysis of PNBs on $Salsa20/8$ and ChaCha stream ciphers to evaluate the resistance of Salsa20 and ChaCha against the mentioned cryptanalysis approach. Sect. 4 studies the mentioned cryptanalysis approach on Salsa20/8.

4. An Extensive Study of Probabilistic Neutral Bits

The existing differential cryptanalysis of the Salsa20 stream cipher studies the $\mathcal{ID}$-$\mathcal{OD}$ pair with the highest forward differential bias ε_d among all possible pairs (i.e., 128 $\mathcal{ID}$ bits and 512 $\mathcal{OD}$ bits). At first, the $\mathcal{ID}$ is determined then the corresponding $\mathcal{OD}$ with the highest forward bias is selected. That is to say, the existing studies focused on the differential bias at specific $\mathcal{ID}, \mathcal{OD}$ pairs and then tried to find the subset of PNBs to attack the target round of Salsa20. Considering our observation in Sect. 5.1 and the previous research results [5], the neutrality measures of all 256 key bits in Salsa20 mainly depend on $\mathcal{OD}$ bits, therefore, when a key-recovery attack is effectuated in mentioned flow, it cannot be shown whether the bias ε_d at specific $\mathcal{ID}, \mathcal{OD}$ pair and the subset of PNBs are truly optimal or not. In this section, we focus on an extensive study of the neutrality measure of all 256 key bits with the respect to all possible 512 $\mathcal{OD}$ bits. Furthermore, we analyze the conditions and circumstances that possibly instigate the high neutral measures as the size of PNBs affects the time complexity of the attack, as shown in Eq. (13). Presumably, no study on analyzing the PNBs in detail has been reported. If the conditions that induce high neutral measure γ_κ of key bits can be clarified, we can claim that the existing attacks may still have room for improvement. We used Algorithm 4 to compute the neutrality measure of 256 key bits with the respect to 512 $\mathcal{OD}$ bits. For this purpose, we generated a set of 2^{10} random keys and 2^{24} random initialization vectors (IVs) through a random generation process. Then, we calculated the probabilistic neutrality measure of the key bits positions using these randomly generated keys and IVs. This process was repeated for each iteration of the randomly generated keys and IVs. As a result, we calculated the neutrality measure of 256-bit key positions, for each of the 512 $\mathcal{OD}$ positions. Throughout our experiments, we used the Theorem 1 to decide the number of random keys and IVs. To select an optimal $\mathcal{ID}$ position, we tried all possible 128 $\mathcal{ID}$ positions. It should be pointed out that we could not observe a significant impact of the $\mathcal{ID}$ on the neutrality measure of key bits. Therefore, we decided to select a random $\mathcal{ID}$ at this phase of experiment[†].

Algorithm 4 Computing $\mathcal{OD}$ bit with best neutral measure

Require: Random(X, X', Z, Z')
Ensure: The $\mathcal{OD}$ bit with best neutral measure

1. Generate random keywords $k = (k_0, \ldots, k_7)$.
2. Decide the single $\mathcal{ID}$ $\Delta_i^{(0)}[j]$ position, and generate random $v = (v_0, v_1)$ and $t = (t_0, t_1)$, initiate $X^{(0)}$ and $X'^{(0)} = X^{(0)} \oplus \Delta_i^{(0)}[j]$.
3. From $(X^{(0)}, X'^{(0)})$, compute $(X^{(r)}, X'^{(r)})$ and $(X^{(R)}, X'^{(R)})$.
4. From $(X^{(r)}, X'^{(r)})$ compute $\mathcal{OD}$ $\Delta_p^{(r)}[q] = X_p^{(r)}[q] \oplus X_p'^{(r)}[q]$ for all possible p and q.
5. From $(X^{(R)}, X'^{(R)})$ obtain the keystream $Z = X^{(0)} + X^{(R)}$ and $Z' = X'^{(0)} + X'^{(R)}$.
6. Complement a key bit κ ($\kappa \in \{0, \ldots, 255\}$) and compute $\overline{X}^{(0)}$ and $\overline{X'}^{(0)}$ from initial states $(X^{(0)}, X'^{(0)})$.
7. Compute $(Y^{(r)}, Y'^{(r)})$ with $Z - \overline{X}^{(0)}$ and $Z' - \overline{X'}^{(0)}$ as inputs to the inversed round function of Salsa20.
8. Derive $\Gamma_p^{(r)}[q] = Y_p^{(r)}[q] \oplus Y_p'^{(r)}[q]$ for all possible choices of p and q.
9. Increment the sum for each p, q, and κ only if $\Delta_p^{(r)}[q] = \Gamma_p^{(r)}[q]$.
10. Divide the sum of $\Delta_p^{(r)}[q] = \Gamma_p^{(r)}[q]$ by key trail and $\mathcal{ID}$ samples to get the probability.

5. The Impact on Salsa20

5.1 Experimental Results

In this section, we demonstrate the experimental results of the comprehensive analysis of PNBs. To analyze the neutrality measure of key bits concerning all possible $\mathcal{OD}s$, we have conducted experiments with the complexity of 2^{30} (i.e., 2^6 key trials and 2^{24} IV sample). The probability was obtained over the key trial, nonce, and counter. According to Theorem 1, let $\mathcal{X}$ be a distribution of $\Delta_p^{(r)}[q] = \Gamma_p^{(r)}[q]$ obtained from the r-round internal state matrices in true random number generator, and let $\mathcal{Y}$ be a distribution of $\Delta_p^{(r)}[q] = \Gamma_p^{(r)}[q]$ obtained from the r-round internal state matrices of Salsa20. The target event occurs in $\mathcal{X}$ with a probability of $\frac{1}{2}$ and $\mathcal{Y}$ with a probability of γ_κ; thus, the number of samples to distinguish $\mathcal{X}$ and $\mathcal{Y}$ is $O(\frac{2}{\gamma_\kappa^2})$. Our experimental results are reliable when the derived neutral measures γ_κ are greater than $2^{-14.5}$ as we have experimented with total 2^{30} numbers of IV samples and key trials.

Figures 1 and 2 show the experimental result of a comprehensive analysis of key bits neutrality measures with the respect to all possible $\mathcal{OD}$ bit positions on Salsa20. In

[†]In our experiment (Algorithm 4), we used the $\mathcal{ID}$ (7,31) reported in [5] and used in [13] and [9]

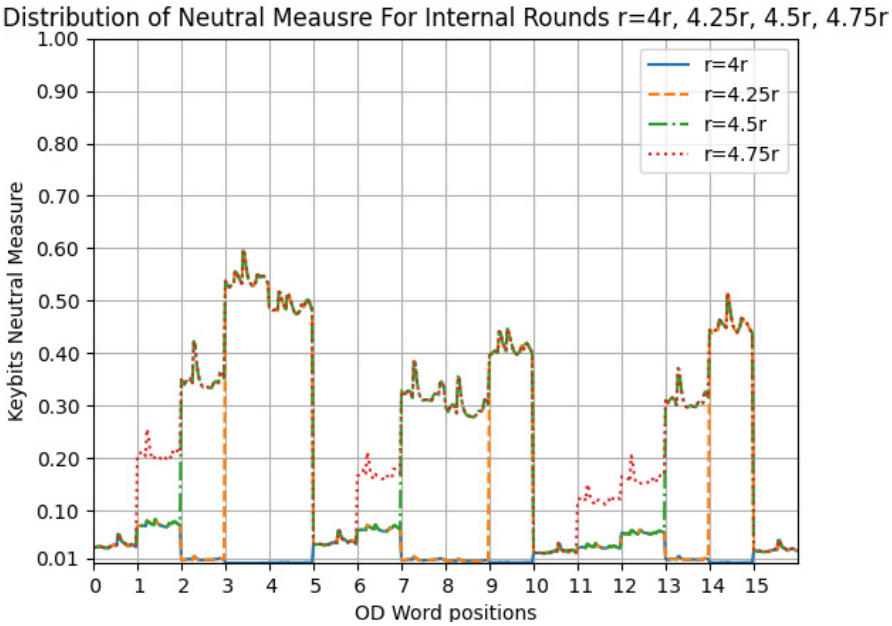


Fig. 1 Distributions of neutral measures γ_κ in each of $r = 4$, $r = 4.25$, $r = 4.5$, and $r = 4.75$ internal rounds.

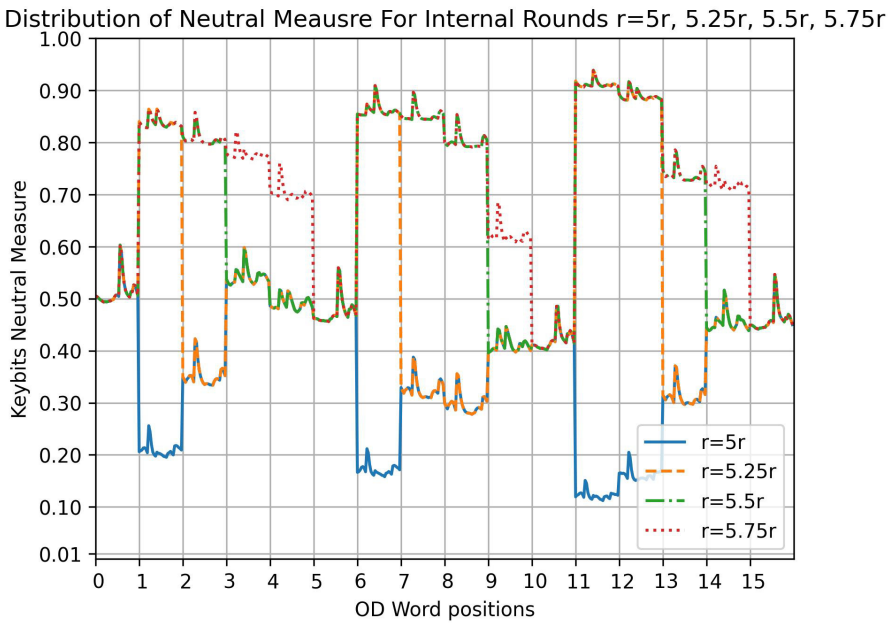


Fig. 2 Distributions of neutral measures γ_κ in each of $r = 5$, $r = 5.25$, $r = 5.5$, and $r = 5.75$ internal rounds.

these figures, the vertical axis represents the average neutral measures of key bits at each $\mathcal{OD}$ position, the horizontal axis represents the $\mathcal{OD}$ bit positions. The blue, orange, Green, red lines show the distribution of neutral measures in the full internal round, quarter internal round, half internal round, and three-quarters internal rounds respectively. We used all the possible internal rounds within the 4th and 5th internal rounds. Figure 1 and Fig. 2 show the $r = 4, r = 4.25, r = 4.5, r = 4.75$ and, $r = 5, r = 5.25, r = 5.5, r = 5.75$, respectively. From Figs. 1 and 2, we obtain the following properties:

- The distribution of neutrality measure of key bits varies for each $\mathcal{OD}$ bit position. As Figs. 1 and 2 support the claim.
- The $(4.25r, 4.5r$, and $4.75r)$ internal rounds generate the same neutrality measure in some $\mathcal{OD}$ positions. To be precise, $(3rd, 4th, 5th, 9th, 10th, 14th, 15th)$ $\mathcal{OD}$ words give the same neutral measure. However, the $\mathcal{OD}$ positions are different for $r = 5, r = 5.25, r =$

Table 4 The cumulative number of modular addition executed in $3, 2.25, 2.5, 2.75$ reverse rounds from target round R=8

R	Input word position		Cumulative number of modular additions			
	Even round	Odd round	2.25 rounds	2.5 rounds	2.75 rounds	3 rounds
Even	$A \rightarrow$	A	24	24	24	24
	B $\rightarrow$	D	7	7	7	76
	C $\rightarrow$	C	11	11	52	52
	D $\rightarrow$	B	16	35	35	35

Table 5 The cumulative number of modular addition executed in $4, 3.25, 3.5, 3.75$ reverse rounds from target round R=8

R	Input word position		Cumulative number of modular additions			
	Even round	Odd round	3.25 rounds	3.5 rounds	3.75 rounds	4 rounds
Even	$A \rightarrow$	A	112	112	112	112
	B $\rightarrow$	D	76	164	164	164
	C $\rightarrow$	C	52	52	241	241
	D $\rightarrow$	B	35	35	35	277

5.5, $r = 5.75$ internal rounds. It will be further explored in Sect. 5.2.

- The neutrality measure of key bits is affected by the inversed-quarter round function of Salsa20. As the number of reverse rounds increases, the neutrality measure of key bits is decreased. It will be further discussed in Sect. 5.2.
- The $\mathcal{OD}$ bit position with a high neutral measure varies in different word positions.
- The distributions of neutral measures concerning the $\mathcal{OD}$ bit position are much similar regardless of the internal round r.

5.2 Relationship Between Neutrality Measures and Inversed Rounds

To study the correlation between the neutrality measure of key bits and inversed quarter-round function of Salsa20, we scrutinize the relationship between the input word position to the inversed quarter-round function and the cumulative number of modular addition executed for each input vector (A, B, C, D) given the specific number of reverse rounds. In Table 4 and 5, we investigated the cumulative number of modular addition executed for different input word positions to the inversed quarter-round function for different internal rounds mentioned above tables.

The R column points to the number of target rounds in our attack (i.e., 8-rounds in this study). The combination of input word positions to the inversed quarter-round function is different depending on whether the target round R is even or odd. To enumerate, the input word positions, such as a vector (A, B, C, D), to the inversed quarter-round function is different in odd or even rounds as the reverse round of even rounds (row round) will be executed in a case when the target round R is even. However, the reverse round of the odd round (column round) is executed in case the target round R is odd. Please refer to section 2 for further detail. To clarify, we consider the case when the number of target rounds R is even and the input word position to quarter-round round is B. When the number of the inversed rounds is 3, the word position moves B (even number round) $\rightarrow D$ (odd number round) by executing the inversed quarter-round function. In other words, the same element in the Salsa20 matrix is picked in a different order by inverse quarter-round in odd and even reverse rounds. For instance, element X_3 is represented by D in vector (A, B, C, D) when the target round is even while the same element X_3 is represented by B when the target round is odd. Similarly, when the number of the inversed rounds is three, the word position transitions B (even number round) $\rightarrow D$ (odd number round) $\rightarrow B$ (even number round). The cumulative number of modular addition column shows the cumulative number of modular additions executed by the inversed quarter-round function for each transition of the word positions for different inversed rounds. At this point, we focus only on the execution of the cumulative number of modular addition because it plays an important role in ensuring the security of the ARX ciphers. As per Table 4 and Table 5, the execution of modular addition differs depending on the input word position to the inversed round function and the number of revere rounds. For example, we consider the case when the number of target round R is even, the transition of the word position is A (even number round) $\rightarrow A$ (odd number round), and the number of inversed rounds are $2.25r, 2.5r, 2.75r, 3r$ respectively. In this case, the cumulative number of modular addition is 24 for each round. Similarly, when the number of inversed round functions is 3, the maximum and minimum values of the cumulative modular addition are 76 and 24. Thus, as the number of the inversed-round is increased, the difference between the maximum and minimum values of the cumulative number modular addition is increased as well. Equally important, the cumulative number of modular addition executed for 2.25 reverse rounds is equal to the 3 reverse rounds for input word position A. Likewise, the cumulative number of modular addition for word positions such as the transition of word position $B \rightarrow D$ and the transition of word position $D \rightarrow B$ do not change for some internal rounds such as 2.25, 2.5 and 2.75. This analysis also applies to Salsa20 forward quarter round function. Considering the stated fact, we can target higher rounds of Salsa20 such as

Table 6 Experimental result to find $\mathcal{OD}$ bit with best average neutral measure in different internal rounds for target round R=8

Internal Round	$\mathcal{OD}$ Word	$\mathcal{OD}$ Bit	Average Neutral Measure
$4r$	1	13	0.083327305
$4.25r$	3	13	0.598177289
$4.5r$	3	13	0.598177289
$4.75r$	3	13	0.598177289
$5r$	0	18	0.602847941
$5.25r$	11	13	0.938618043
$5.5r$	11	13	0.938618043
$5.75r$	11	13	0.938618043

$R = 8.75$ which would be equivalent to $R = 8.25$ for some specific words. This shows a weakness in Salsa20 quarter-round function design. Table 6 shows the result obtained from Algorithm 4. As we can see the $\mathcal{OD}$ word with the best neutral measure in internal rounds r $4.25r, 4.5r.4.75r$ with corresponding reverse rounds $3.25, 3.5, 3.75$ appeared in X_3 which is the input word position $D \rightarrow B$ in Table 5 and have the same number of cumulative number of modular addition (i.e., 35) and generated the same average neutral measure $\gamma_k = 0.598177289$. The same analysis applies for the internal rounds $4r, 5r, 5.25r, 5.5r, 5.75r$. The experimental results are in Fig. 1 and 2 plot our findings of Table 4, Table 5 and Table 6. The input word position that induces the lower neutral measures in $4r$ internal round D corresponds to X_3 in the Salsa20 matrix which corresponds to the input word position $D \rightarrow B$ when $R - r = 4$ and has 277 cumulative number of modular addition. The $D \rightarrow B$ word positions move through the high cumulative number of modular addition for internal round $4r$. Furthermore, X_4, X_9, X_{14} generate the lowest average neutral measure when $r = 4$ which is the transition of words from $D \rightarrow B$. The same analysis applies to all possible internal rounds. In summary, we can see that the neutral measure depends on the input word position to the inversed round function, and it is influenced by the cumulative number of modular addition. The conditions that induce high neutral measure depend on the $\mathcal{OD}$ bit position to inverse quarter-round function. This was also claimed in Sect. 3.5 of [5].

To discuss the upper bounds of the inversed round function for our attack, we analyze neutral measures for each inversed round function in detail. Tables 7 and 8 show the maximum, minimum, average, and median values of neutral measures γ_κ for each target round R when $r = 4$ and 5, respectively. The results were obtained by a comprehensive analysis of the experimental results described in Sect. 4 Algorithm 4. As illustrated in the tables, the maximum neutral measures in a certain inversed round function (i. e, $R - r = 3$ in this case) never exceed the minimum neutral measure value in a smaller reverse round (i. e, $R - r = 2$ in this case) in the same internal round r. For instance, when $R = 7$ and the inversed round function $R - r = 3$ (see Table 7), the maximum neutral measure value is $\gamma_\kappa = 2^{-0.8143}$, whereas the minimum neutral measure value is $\gamma_\kappa = 2^{-0.5451}$ for $R = 6$ and $R - r = 2$. It is clear that the minimum neutral measure value for $R - r = 2$ is higher than the maximum neutral measure in $R - r = 3$, and vice versa. It is influenced by the cumulative number of modular addition, as discussed earlier. Our experimental results are reliable when the derived neutral measures γ_κ are greater than $2^{-14.5}$ as we have used 2^{30} $\mathcal{ID}$ samples. From Table 7, all the neutral measure values are reliable when $R = 6, 7, 8$, and $R - r = 2, 3$, respectively. In Sect. 5.5 Table 13, we calculated the complexity of attack for $R - r = 4$. The complexity of the attack is less than Salsa20 security. Hence, we conclude that the maximum number of reverse rounds in our proposed attack is $4r$. Similarly, from Table 8, all the neutral measure values when $R = 7$, 8 and $R - r = 2$, 3 and 4 are reliable.

Table 7 Maximum, minimum, average, and median values of neutral measures γ_κ for each target round R when $r = 4$, where p and q are word and bit positions of the $\mathcal{OD}$, respectively, i.e., $\Delta_p^{(r)}[q]$.

R	Maximum			Minimum			Average	Median
	γ_κ	p	q	γ_κ	p	q		
6	$2^{-0.00144}$	2	1	$2^{-0.5451}$	9	7	$2^{-0.1907}$	$2^{-0.1473}$
7	$2^{-0.8143}$	0	18	$2^{-2.4426}$	14	7	$2^{-1.2708}$	$2^{-1.1496}$
8	$2^{-3.5850}$	1	13	$2^{-10.4926}$	2	9	$2^{-4.9755}$	$2^{-5.3066}$

Table 8 Maximum, minimum, average, and median values of neutral measures γ_κ for each target round R when $r = 5$, where p and q are word and bit positions of the $\mathcal{OD}$, respectively, i.e., $\Delta_p^{(r)}[q]$.

R	Maximum			Minimum			Average	Median
	γ_κ	p	q	γ_κ	p	q		
7	$2^{-0.1076}$	0	18	$2^{-0.4300}$	12	7	$2^{-0.2206}$	$2^{-0.1808}$
8	$2^{-0.7301}$	0	18	$2^{-2.7284}$	11	7	$2^{-1.2703}$	$2^{-1.2019}$
9	$2^{-3.6889}$	4	13	$2^{-10.39}$	12	28	$2^{-5.37}$	$2^{-5.9}$

5.3 PNB-Based Differential Cryptanalysis of Salsa20

In this section, we first explain the PNB-based differential cryptanalysis on the reduced round of Salsa20 stream cipher. At first, we study the neutrality measure of all key bits positions with the respect to all possible $\mathcal{OD}$ bit positions. For this purpose, we use Algorithm 4 in Sect. 4. The Algorithm 4 helps us to find the $\mathcal{OD}$ bit with the best average neutral measure. Once we selected the $\mathcal{OD}$ bit with the best average neutral measure. Then we search all possible $\mathcal{ID}$ positions with the best differential bias ε_d at the predefined $\mathcal{OD}$ (i.e., $\mathcal{OD}$ bit with best average neutral measure) position[†]. Following this, we used the $\mathcal{ID}$, $\mathcal{OD}$ pair to search for the PNBs subset. Afterward, we computed the reverse bias ε_a for each threshold γ. Subsequently, we estimated the complexity of the attack on the reduced round of Salsa20. We presented a differential attack based on the comprehensive analysis of PNBs on Salsa20/8 with a time complexity of $2^{241.62}$ and data complexity of $2^{31.5}$. It's worth mentioning, that the obtained neutral measure in Sect. 4 is used only to select the $\mathcal{OD}$ bit with the best neutral measure, we do not use that neutral measure for the attack complexity estimation. The following subsections describe the proposed cryptanalysis method in detail.

[†]We searched in 128 possible $\mathcal{ID}$ bit positions for a given $\mathcal{OD}$ position with best average neutral measure

Table 9 $\mathcal{ID}$ bit positions with the best median bias ε_d^* at given $\mathcal{OD}$ positions in different internal rounds.

Internal Rounds	$\mathcal{ID}$	$\mathcal{OD}$	Bias ε_d^*
$4r$	7,0	1,13	0.154433
$4.25r$	8,11	3,13	0.000031
$4.5r$	8,11	3,13	0.000031
$4.75r$	8,11	3,13	0.000031
$5r$	7,22	0,18	0.000024
$5.25r$	6,3	0,18	0.000026
$5.5r$	9,28	11,13	0.000033
$5.75r$	9,28	11,13	0.000033

5.4 Analysis of Single-bit Differential

In Sect. 4, we comprehensively analyzed the neutrality of 256-key bits positions with the respect to all possible 512 $\mathcal{OD}$ bit positions and selected the $\mathcal{OD}$ bit with the best average neutral measures[††]. According to the information presented in Table 6, it can be observed that the OD position located at (11, 13) within the 5.75 internal round results best average neutral measure. We selected the $\mathcal{OD}$ bit positions in Table 6 as target $\mathcal{OD}$ positions to attack Salsa8. To get the corresponding $\mathcal{ID}$ bit positions with best differential bias at predefined $\mathcal{OD}$ positions in Table 6, we analyzed all possible 128 $\mathcal{ID}$ bit positions for each $\mathcal{OD}$ position listed in Table 6 and selected the $\mathcal{ID}, \mathcal{OD}$ position with best differential bias. In this subsection, we present the result of our search for $\mathcal{ID}$ position with the best median bias ε_d^* at all given $\mathcal{OD}$ positions in Table 6.

As we can see in Table 9, the differential bias ε_d significantly drops after the 4^{th} round. It is directly affected by the dramatic increase in the cumulative number of modular addition. Furthermore, the cumulative number of modular addition for specific words in the Salsa20 state matrix is the same in the listed internal rounds, which caused the same differential bias. Except $4r$ differential bias $\varepsilon_d^* = 0.154433$, all the obtained differential bias in Table 9 could not be verified with a constant probability of success. According to Theorem 1 the obtained differential bias could be trusted when it is greater than 2^{-14} as we have used 2^{30} samples of IVs. The obtained biases are not greater than 2^{-14} and hence could not be trusted for the further attack process. To find the $\mathcal{ID}, \mathcal{OD}$ pair with the reliable median bias ε_d^*, we decided to search for differential bias given the $\mathcal{OD}$ positions for different internal rounds as described in the Table 10.

To get the $\mathcal{ID}$ position with best median bias ε_d^*, for each of 2^5 key trails we tested 2^{30} IVs. The result is summarized in the Table 11. For the first time, we are reporting new pairs of $\mathcal{ID}, \mathcal{OD}$ for single-bit differential cryptanalysis of Salsa20 stream cipher.

To obtain the number of probabilistic neutral bits (PNBs), we have used the Algorithm 1 to get the neutrality measure γ_k of all key bits positions and a threshold

[††]We considered the average neutral value because we computed the neutrality measure of all 256 key bits with the respect to each $\mathcal{OD}$ bit position.

Table 10 New $\mathcal{OD}$ positions to search for reliable bias in different internal rounds

Internal round	$\mathcal{OD}$ Word	$\mathcal{OD}$ bit
$4.5r, 5.25r, 5.75r$	10	13
	11	13
	12	13
	13	13
	14	13
	15	13

Table 11 New $\mathcal{ID}$ bit positions with the best median bias ε_d^* at given $\mathcal{OD}$ positions

Internal Rounds	$\mathcal{ID}$	$\mathcal{OD}$	ε_d^*
$4.75r$	8,27	11,13	0.0651375
$4.75r$	8,20	11,13	0.009411
$4.75r$	8,8	11,13	0.006427
$4.75r$	9,9	11,13	0.0046905
$4.75r$	9,18	12,13	0.0264355
$4.75r$	8,5	12,13	0.000984
$4.75r$	6,2	12,13	0.0004605
$4.75r$	8, 24	12,13	0.000403

Table 12 The number of PNBs obtained for internal rounds $4.75r$ for the target round R=8

Threshold γ	$4.75r$
$\gamma = 0.1$	67
$\gamma = 0.2$	46
$\gamma = 0.25$	43
$\gamma = 0.27$	40
$\gamma = 0.3$	37

$0.1 \leq \gamma \leq 0.3$ to divide the set of key bits into two subsets of significant bits m and non-significant bits n. As the $\mathcal{ID}$ $(8, 27)$ and $\mathcal{OD}$ (11,13) have the highest median bias, we selected the mentioned pair to look for PNBs. We summarize the results obtained from Algorithm 1 in Table 12. We focus on the $4.75r$ internal rounds Salsa20/8, this is due to the following reasons:

- As the cumulative number of modular addition and average neutral measure are the same for the quarter, half, and three-quarter internal rounds, we would compute the complexity of one internal round from quarter, half, and three-quarter internal rounds[†].
- It is difficult to efficiently perform our attack on Salsa20/9 as we could not observe a high average neutral measure for any $\mathcal{OD}$ bit in Sect. 4. It is because of the high number of modular-addition executed both in forwarding and backward rounds that dramatically drop the forward and backward bias.
- Comparing the result of Table 6, we can see that $\mathcal{OD}$ bits provide a better neutral measure using Algorithm 4 when $r = 5.75$ and $R - r = 2.25$. As we could not find valid bias in mentioned internal round thus, we selected the $\mathcal{OD}$ position $(11, 13)$ and $r = 4.75$ and $R - r = 3.25$ from Table 11 to attack Salsa20/8.
- Furthermore, we could not find reliable median bias for $5r, 5.25r, 5.5r$, and $5.75r$, we decided to select the $4.75r$ internal round to attack Salsa8 as the higher num-

[†]We decided to compute the complexity of $4.75r$ internal rounds out of the quarter, half, and three-quarter internal rounds.

Table 13 Summary of attack on Salsa20/8 for threshold γ, PNB bits n, differential bias ε_d, and reverse bias ε_a for $4r$ internal round.

γ	n	$\lvert\varepsilon_d^*\rvert$	$\lvert\varepsilon_a^*\rvert$	α	Time	Data
0.1	40	0.154433	0.0004109	11	$2^{248.9}$	$2^{32.8}$
0.2	32	0.154433	0.00744507	12	$2^{248.6}$	$2^{24.54}$
0.3	26	0.154433	0.04336255	11	$2^{249.46}$	$2^{19.4}$

Table 14 Summary of attack on Salsa20/8 for each threshold γ, PNB bits n, median bias ε_d^*, and reverse bias ε_a for $4.75r$ internal round.

γ	n	$\lvert\varepsilon_d^*\rvert$	$\lvert\varepsilon_a^*\rvert$	α	Data	Time
0.3	37	0.0651375	0.01462765	16	$2^{25.9}$	$2^{244.36}$
0.2	**46**	**0.0651375**	**0.00173888**	**19**	$\mathbf{2^{31.5}}$	$\mathbf{2^{241.62}}$

ber of internal rounds increase the number of PNBs due to low number inverse cumulative number of modular addition which results better neutral measure for key bits positions.

Next section, we estimate the complexity of the attack on Salsa20/8 considering $4.75r$ internal rounds.

5.5 Attack Complexity Estimation on Salsa20

To accurately estimate time and data complexities for our proposed attack on Salsa20/8, the remaining steps should be performed as follows:

Step 1. We recalculate neutral measures corresponding to the determined $\mathcal{ID}$-$\mathcal{OD}$ pair $(\Delta_i^0[j], \Delta_p^r[q])$, and divided secret key bits into two subsets: m-bit subset of significant key bits and n-bit subset of non-significant key bits. For step 1, we used Algorithm 1.

Step 2. In the second step, we executed the probabilistic backward computation Algorithm 2 to obtain the r-round differential biases ε_a from the backward for each threshold $0.1 < \gamma \leq 0.3$ from the obtained keystream, and approximate the overall median bias $|\varepsilon^*| \approx |\varepsilon d^*| \cdot |\varepsilon a^*|^\dagger$ for our attack on Salsa20/8.

Step 3. We run the online phase Algorithm 3 and estimate the time and data complexities to recover an unknown key, as described in Eq. (13).

To perform the above steps, for each of 2^8 key trials, we run 2^{25} $\mathcal{ID}$ samples to compute the neutrality measure of key bits and obtained the subset of significant and non-significant key bits. The attack on Salsa20/8 with $r = 4.75$ is reported in Table 14. The PNBs are $0, 13, 14, 15, 16, 17, 31, 43, 44, 45, 70, 71, 76, 96, 97, 101, 113, 114, 115, 116, 117, 123, 124, 125, 126, 127, 135, 153, 154, 155, 156, 157, 158, 159, 175, 171, 172, 190, 229, 230, 231, 232, 233, 234, 247, 248$

5.5.1 PNBs Verification

Once we select the subset of PNBs (Sect. 5.5 Step 1), we need to verify the authenticity of PNB bits. For this purpose, we used Algorithm 5.5.1.

Algorithm 5 [5] PNB bits verification

Require: Random(X, X', Z, Z')
Ensure: The absolute value of $\hat{\varepsilon}$

1. Compute $(X^{(R)}, X'^{(R)})$ with $\Delta_i^{(0)}[j] = 1$; and derive $Z = X^{(0)} + X^{(R)}$ and $Z' = X'^{(0)} + X'^{(R)}$.
3. Prepare $(\hat{X}^{(0)}, \hat{X}'^{(0)})$ with all key bits set to a random binary value from $(X^{(0)}, X'^{(0)})$.
4. Compute $(\hat{Y}^{(r)}, \hat{Y}'^{(r)})$ with $Z - \hat{X}^{(0)}$ and $Z' - \hat{X}'^{(0)}$ as inputs to the inversed round function of Salsa20.
5. Obtain $\hat{\Gamma}_p^{(r)}[q] = \hat{y}_p^{(r)}[q] \oplus \hat{y}_p'^{(r)}[q]$
6. Compute the $\hat{\varepsilon}$ as $\Pr(\hat{\Gamma}_p^{(r)}[q] \mid \Delta_i^{(0)}[j] = 1) = \frac{1}{2}(1 + \hat{\varepsilon})$.

According to [13], if we find $\hat{\varepsilon}$ with low bias (close to a random event), it implies that the PNBs are selected properly. We can attack Salsa20/8 with a time complexity of $2^{241.62}$ and data complexity of $2^{31.5}$ with threshold $\gamma = 0.2$. Now, we focus on $\varepsilon_a^* = 0.00173888$ $(= 2^{-9.16})$ when $\gamma = 0.2$. According to Theorem 1, $\frac{2}{\varepsilon_a^2} = 2^{19.33}$ $\mathcal{ID}$ samples are sufficient to distinguish the differential bias with a constant probability of success; thus, our experimental results are reliable when $\gamma = 0.2$ since we have used 2^{25} $\mathcal{ID}$s samples for each of 2^8 key trials. For $\gamma = 0.1$, we found 67 PNBs with $\varepsilon_a^* = 0.00001713635$ $(= 2^{-15.832})$ and it could not be constantly verified by our experiment and hence the result are not reliable for $\gamma = 0.1$. To summarize our findings, we have presented that it is feasible to perform the differential attack on Salsa20/8 based on the comprehensive analysis of probabilistic neutral bits with time complexity of $2^{241.62}$ and data complexity of $2^{31.5}$. As shown in Table 2, the existing best key-recovery attack on Salsa20 is the differential attack on Salsa20/8 with time complexity of $2^{243.7}$, proposed by Dey and Sarkar [9]; and we improved the differential attack on Salsa20/8.

6. The Impact on ChaCha

In this section, we present the result of our approach to the ChaCha stream cipher. We analyzed the neutrality measure of key bits of ChaCha20/7, ChaCha20/7.25, and ChaCha20/7.5. However, for the complexity of the attack, we focused only on ChaCha20/7.25.

6.1 ChaCha20/7

We used the Algorithm 4 to evaluate the average neutrality measure of all key bits concerning all possible $\mathcal{OD}$ bit positions for $r = 3, r = 3.5$, and $r = 4$ internal rounds. Table 15 summarizes the result of our experiment.

Considering the Table 15 the $\mathcal{OD}$ bit 0 gives the best neutral measure regardless of internal rounds. It is affected by the cumulative number of modular subtraction, the input word position to the inverse quarter round of ChaCha, and the structure of ChaCha quarter round function. We

[†]According to [5] Under some reasonable independency assumptions, the equality $\varepsilon = \varepsilon_d * \varepsilon_a$ holds.

plotted the distribution of neutrality measure of ChaCha20/7 considering the $r = 3, r = 3.5$, and $r = 4$ internal rounds in Fig. 3. The Y-axis shows the neutrality measure of 256 key bits and X-axis represents the $\mathcal{OD}$ bit position. The results in Table 15 and Fig. 3 suggest that the higher number of internal rounds affects the neutrality measure. The neutral measure for the internal round $r = 3$ is lower than $r = 3.5$. This is directly impacted by the cumulative number of modular subtraction executed for different internal rounds. We decided to attack the higher number of ChaCha rounds with the internal round $r = 3.5$. We could also select the $r = 4$ internal round to attack $R = 7, 7.25$ or $R = 7.75$, however, the forward bias ε_d would significantly drop for $r = 4$ internal round. As the researchers have not focused on the security of evaluation of ChaCha7.25 rounds, we decided the use $r = 3.5$ to attack ChaCha7.25 and reduce the attack complexity reported by [7].

6.2 ChaCha7.25, ChaCha7.5, ChaCha7.75

In this section, we study the neutrality measure of *ChaCha*7.25, *ChaCha*7.5, *ChaCha*7.75 considering the internal round $r = 3.5$. We have used the Algorithm 4 to estimate the neutrality measures of different target rounds. We summarized the result in Table 16 and Fig. 4. The X-axis and Y-axis in Fig. 4 represent the same parameters as Fig. 3.

All things considered, we decided to attack ChaCha7.25 with an internal round $r = 3.5$.

6.3 Analysis of Singe-bit Differential

To attack ChaCha7.25 round, we selected the $\mathcal{OD}$ position$\Delta^{(3.5)}_{7,0}$ from Table 16 and searched for all possible 128 bits $\mathcal{ID}$ positions with the best differential bias ε_d. To identify the $\mathcal{ID}$ position with best differential bias ε_d at predefined $\mathcal{OD}$ position, we experimented with the total complexity of 2^{34} (i.e., 2^6 key trials and 2^{28} IV samples). According to Theorem 1, the obtained bias ε_d is reliable if it's greater than $2^{-15.5}$. We analyzed the result and found the highest forward bias ε_d is in $\Delta^{(0)}_{12[18]}|\Delta^{(3.5)}_{7[0]} = 0.000019$. As the bias, $0.000019 = 2^{-15.68}$ could not be verified by the number of samples we used. Hence, we decided to look for differential bias ε_d at $\Delta^{(3.5)}_{0[0]}, \Delta^{(3.5)}_{1[0]}, \Delta^{(3.5)}_{2[0]}, \Delta^{(3.5)}_{3[0]}$ $\mathcal{OD}$ positions. We summarize the result in Table 17.

All the obtained biases in Table 17 are authentic as they could be verified by the number of samples used during the experiment thus, we used the obtained pairs with best differential bias ε_d in Table 17 to find the subsets of significant key bits m and non-significant key bits n.

6.4 Complexity Estimation of Attack on ChaCha7.25

To approximate the attack complexity on ChaCha7.25 rounds, we repeat the steps mentioned in Sect. 5.5 with

Table 15 Experimental result to find $\mathcal{OD}$ bit with the best neutral measure in different internal rounds for target round R=7

Internal Round	$\mathcal{OD}$ Word	$\mathcal{OD}$ Bit	Average Neutral Measure
$3r$	11	0	0.1606
$3.5r$	9	0	0.3820
$4r$	4	0	0.6485

Table 16 Experimental result to find $\mathcal{OD}$ bit with best neutral measure in internal rounds $r = 3$ for target round R=7.25, 7.5, 7.75

Target Round	$\mathcal{OD}$ Word	$\mathcal{OD}$ Bit	Average Neutral Measure
$R = 7.25$	7	0	0.2826
$R = 7.5$	4	0	0.1509
$R = 7.75$	11	0	0.0747

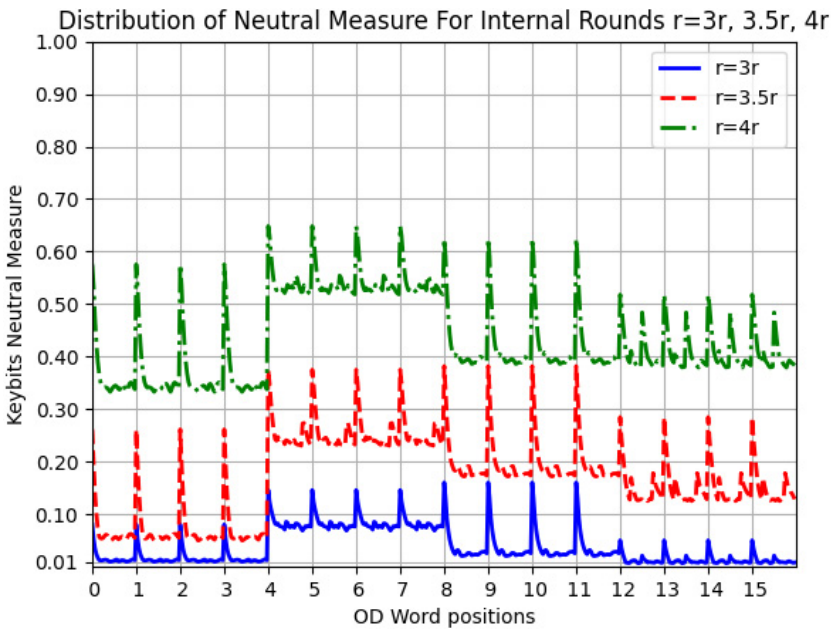


Fig. 3 Distributions of neutral measures γ_κ in each of $r = 3, r = 3.5, r = 3.75$ internal rounds for the target round R=7

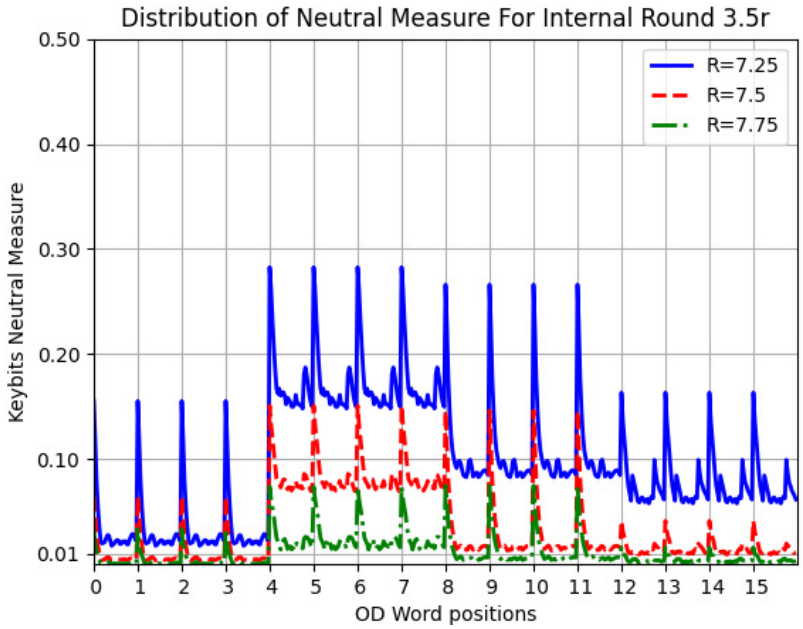


Fig. 4 Distributions of neutral measures γ_κ in each of target rounds $R = 7.5, R = 7.5, R = 7.75$ for internal rounds $r = 3$

Table 17 $\mathcal{ID}$ bit positions with best differential bias ε_d at given $\mathcal{OD}$ positions in $r = 3.5$ internal rounds.

$\mathcal{ID}$ Position	$\mathcal{OD}$ Position	Bias ε_d
$\Delta^{(0)}_{15[6]}$	$\Delta^{(3.5)}_{0[0]}$	0.000463
$\Delta^{(0)}_{12[6]}$	$\Delta^{(3.5)}_{1[0]}$	0.000414
$\Delta^{(0)}_{13[6]}$	$\Delta^{(3.5)}_{2[0]}$	0.000474
$\Delta^{(0)}_{14[6]}$	$\Delta^{(3.5)}_{3[0]}$	0.000472

Table 18 The subset of PNBs n considering different thresholds γ for $r = 3.5$ internal rounds

Threshold γ	$\mathcal{ID}, \mathcal{OD}$ Pair	Number of PNBs
$\gamma = 0.25$	$\Delta^{(0)}_{15[6]}\|\Delta^{(3.5)}_{0[0]}$	54
$\gamma = 0.25$	$\Delta^{(0)}_{12[6]}\|\Delta^{(3.5)}_{1[0]}$	54
$\gamma = 0.25$	$\Delta^{(0)}_{13[6]}\|\Delta^{(3.5)}_{2[0]}$	54
$\gamma = 0.27$	$\Delta^{(0)}_{14[6]}\|\Delta^{(3.5)}_{3[0]}$	50

the $\mathcal{ID}, \mathcal{OD}$ positions from Table 17 and considering the ChaCha stream cipher structure explained in Sect. 2.2. To find the subsets of m and n, we conducted an experiment with total complexity of 2^{36}. As per the value of threshold γ, we have different numbers of the element in both subsets of m and n summarized in Table 18. The list key bits with the respect to $\gamma = 0.25$ and $\mathcal{ID}, \mathcal{OD}$ position $\Delta^{(0)}_{13[6]}|\Delta^{(3.5)}_{2[0]}$ identified as *probabilistic neutral bits* are [66, 67, 74, 77, 78, 83, 84, 90, 91, 95, 104, 108, 109, 110, 111, 115, 123, 124, 125, 126, 127, 135, 155, 156, 157, 158, 159, 160, 168, 169, 191, 192, 193, 194, 199, 200, 207, 208, 211, 212, 219, 220, 221, 222, 223, 224, 225, 226, 227, 244, 245, 246, 247, 255]. Table 19 sum up our attack on ChaCha7.25/20 for the internal round $r = 3.5$, optimal $\mathcal{ID}, \mathcal{OD}$ pair $\Delta^{(0)}_{13[6]}|\Delta^{3.5}_{3[0]}$, the threshold $\gamma = 0.27$ and $n = 54$. The time and data complexity of the attack was estimated $2^{254.01}$ and $2^{51.81}$. Although, the time complexity of the attack is not reduced compared to the proposed attacks on ChaCha7/20. However, it is the first attack on ChaCha7.25/20, and using the differential-linear adversary model, the attack could be further improved which remains an open problem.

Table 19 The attack complexity on ChaCha7.25 for $r = 3.5$ internal rounds

γ	$\mathcal{ID}, \mathcal{OD}$	n	ε_d	ε_a	α	Time	Data
0.25	$\Delta^{(0)}_{15[6]}\|\Delta^{(3.5)}_{0[0]}$	54	0.000463	0.000151	5	$2^{254.19}$	$2^{52.026}$
0.25	$\Delta^{(0)}_{12[6]}\|\Delta^{(3.5)}_{1[0]}$	54	0.000414	0.000156	5	$2^{254.38}$	$2^{52.24}$
0.25	$\mathbf{\Delta^{(0)}_{13[6]}\|\Delta^{(3.5)}_{2[0]}}$	**54**	**0.000474**	**0.000158**	**5**	$\mathbf{2^{254.011}}$	$\mathbf{2^{51.81}}$
0.27	$\Delta^{(0)}_{14[6]}\|\Delta^{(3.5)}_{3[0]}$	50	0.000472	0.000413	4	$2^{255.12}$	$2^{48.95}$

7. Discussion

In this subsection, we discuss the differences between the existing attacks discussed in Sect. 3.2 and our proposed attack. The existing attacks work on the reduced-round Salsa20 by (1) searching for the $\mathcal{ID}$ and $\mathcal{OD}$ differential pair with the best differential bias and (2) analyzing PNB based on obtained input/output differential pair. In addition, some attacks utilize the multi-bit differential bias. For instance, the [8] has used the $\mathcal{ID}$ is $\Delta^{(0)}_7[0]$, the $\mathcal{OD}$ is $\Delta^{(5)}_9[0] \oplus \Delta^{(5)}_{13}[0] \oplus \Delta^{(5)}_1[13]$, the author could reduce the number of significant key bits to 214, and they reported $\varepsilon_a = 0.000752$, $\varepsilon_d = -0.233198$, and subsequently $\varepsilon = -0.000178$. Our proposed attack works on the reduced-round Salsa20 by (1) comprehensively analyzing the $\mathcal{OD}$ bit position with the best average neutral measures and (2) searching for the $\mathcal{ID}$ bit position with the best differential bias in the obtained $\mathcal{OD}$ bit position. Our proposed attack utilizes the single-bit differential bias, such that the $\mathcal{ID}$ is $\Delta^{(0)}_9[12]$, the $\mathcal{OD}$ is $\Delta^{(5)}_{11}[13]$. For ChaCha stream cipher, we operated the $\Delta^{(0)}_{13}[6]$, the $\mathcal{OD}$ is $\Delta^{(3.5)}_2[2]$. Similarly, the ε approximated as $|\varepsilon^*| \approx |\varepsilon^*_d| \cdot |\varepsilon^*_a|$. We summarize the differences between the existing attacks and our attack as follows:

- To the best of our knowledge, for the first time, we used the single bit differential bias in 4.75*th* internal round bias to attack the 8*th* round Salsa20. Previous research mainly used 4*th* internal round bias to attack 8*th* round Salsa20. For instance, the [5] used 4*th* round bias with $\mathcal{ID}, \mathcal{OD}$ pair $\Delta_7^{(0)}[31]$ and $\Delta_1^{(4)}[14]$ with $|\varepsilon_d^*| = 0.131$ and $\varepsilon_a^* = 0.0011$. However, we used the Salsa20/8 4.75*th* and ChaCha7.25 $r = 3.5$ internal round bias to attack Salsa20/8 and ChaCha7.25.
- We scrutinize the conditions and structure of the Salsa20 quarter-round function to increase the number of PNBs. The same analysis is applied to ChaCha quarter-round function.
- We reported 46 probabilistic neutral bits in 256-bit Salsa20/8. For the ChaCha7.25, we reported 54 PNBs. Miyashita and Ito [7] reported 49 PNBs. We added 5 more PNBs which helped to reduce the complexity of the attack.
- We introduce the new pair of $\mathcal{ID}$ and $\mathcal{OD}$ to attack Salsa20/8 and ChaCha7.25 listed in Table 9 and Table 19.
- Our introduced cryptanalysis approach is the first application on Salsa20/8 and ChaCha7.25 to date.
- We introduced new $\mathcal{ID}, \mathcal{OD}$ pairs with high bias. The introduced bias at specific $\mathcal{ID}, \mathcal{OD}$ positions could be extended to higher internal rounds with Differential-Linear adversary to reduce the complexity of the attack.

It is the first application of its type on the reduced rounds of Salsa20/8 and ChaCha7.25. For ChaCha7.25 we improved the attack complexity by a factor of $2^{1.609}$ from the attack introduced by Miyashita and Ito [7]. These are notable improvements in the field of cryptanalysis. In brief, the less significant key bits, the less time complexity an adversary needs to recover an unknown secret key; thus, we demonstrate its superiority as an effective differential attack on Salsa20/8 stream cipher by focusing on the comprehensive analysis of PNBs with the respect to all possible 512 bits output difference.

8. Conclusion

In this study, we analyzed the distribution of neutrality measures of 256 key bits positions of Salsa20/8 and ChaCha7, ChaCha7.25, ChaCha7.5, and ChaCha7.75 rounds. Our approach focused on the comprehensive analysis of PNB rather than looking for $\mathcal{ID}$ and $\mathcal{OD}$ pairs. As a result, the approach allows us to perform the best differential attack on Salsa20/8 with a time complexity of $2^{241.62}$ and data complexity of $2^{31.5}$ and on ChaCha7.25 with a time complexity of $2^{254.011}$ and data complexity of $2^{51.81}$. The proposed cryptanalysis approach may also contribute to the improvement of the existing differential attacks on different stream ciphers and differential-linear cryptanalysis approach, which remains an open problem.

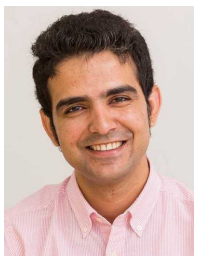
Acknowledgements

This work is partially supported by JSPS KAKENHI Grant Number JP21H03443 and SECOM Science and Technology Foundation.

References

[1] D.J. Bernstein, "The Salsa20 family of stream ciphers," In: M. Robshaw, O. Billet, (eds) New Stream Cipher Designs, Lect. Notes Comput. Sci., vol.4986, Springer, Berlin, Heidelberg, 2008. https://doi.org/10.1007/978-3-540-68351-3_8
[2] D.J. Bernstein, "ChaCha, a variant of Salsa20," Workshop record of SASC, vol.8, no.1, pp.3–5, 2008.
[3] The eStream Project, "eSTREAM: the ECRYPT stream cipher project," https://www.ecrypt.eu.org/stream/, accessed Aug. 9. 2022.
[4] J.-P. Aumasson, "Too much crypto," Cryptology ePrint Archive, 2019.
[5] J.-P. Aumasson, S. Fischer, S. Khazaei, W. Meier, and C. Rechberger, "New features of Latin dances: analysis of Salsa, ChaCha, and Rumba," International Workshop on Fast Software Encryption, pp.470–488, Springer, Berlin, Heidelberg, 2008. https://doi.org/10.1007/978-3-540-71039-4_30
[6] P. Crowley, "Truncated differential cryptanalysis of five rounds of Salsa20," Cryptology ePrint Archive, 2005.
[7] S. Miyashita, R. Ito, and A. Miyaji, "PNB-focused differential cryptanalysis of ChaCha stream cipher," Cryptology ePrint Archive 2021.
[8] A. R.Choudhuri and S. Maitra, "Significantly improved multi-bit differentials for reduced round Salsa and ChaCha". IACR Transactions on Symmetric Cryptology, vol.2016, no.2, pp.261–287, 2017, https://doi.org/10.13154/tosc.v2016.i2.261-287
[9] S. Dey and S. Sarkar, "Improved analysis for reduced round Salsa and Chacha," Discrete Applied Mathematics, vol.227, pp.58–69, Aug. 2017. https://doi.org/10.1016/j.dam.2017.04.034
[10] S. Fischer, W. Meier, C. Berbain, J.F. Biasse, and M.J.B. Robshaw, "Non-randomness in eSTREAM candidates Salsa20 and TSC-4," International Conference on Cryptology in India, pp.2–16, 2006. https://doi.org/10.1007/11941378_2
[11] T. Ishiguro, S. Kiyomoto, and Y. Miyake, "Latin dances revisited: new analytic results of Salsa20 and ChaCha," International Conference on Information and Communications Security, pp.255–266, 2011. https://doi.org/10.1007/978-3-642-25243-3_21
[12] R. Ito, "Rotational cryptanalysis of Salsa core function," International Conference on Information Security, pp.129–145, Springer, Cham, 2020.https://doi.org/10.1007/978-3-030-62974-8_8
[13] S. Maitra, "Chosen IV cryptanalysis on reduced round ChaCha and Salsa," Discrete Applied Mathematics, vol.208, pp.88–97, July 2016.https://doi.org/10.1016/j.dam.2016.02.020
[14] Z. Shi, B. Zhang, D. Feng, and W. Wu, "Improved key recovery attacks on reduced-round Salsa20 and ChaCha," International Conference on Information Security and Cryptology, pp.337–351, Springer, Berlin, Heidelberg, 2012.https://doi.org/10.1007/978-3-642-37682-5_24
[15] L. Ding "Improved related-cipher attack on Salsa20 stream cipher," IEEE Access, vol.7, 30197–30202, 2019.
[16] B. Mazumdar, S. Ali, and O. Sinanoglu, "Power analysis attacks on ARX: an application to Salsa20," 2015 IEEE 21st International On-Line Testing Symposium (IOLTS) (pp.40–43), IEEE, 2015.
[17] K.C.D. Kakumani, K. Singh, and S.K. Karthika, "Improved related-cipher attack on Salsa and ChaCha: revisited," Int. J. Inf. Technol., vol.14, no.3, pp.1535–1542, March 2022.
[18] C. Beierle, G. Leander, and Y. Todo, "Improved differential-linear attacks with applications to ARX ciphers," Annual International Cryptology Conference, pp.329–358, Springer, Cham, 2020.

[19] S. Stachowiak, M. Kurkowski, and A. Soboń, "SAT-based cryptanalysis of Salsa20 cipher," Progress in Image Processing, Pattern Recognit. and Communication Systems, pp.252–266, Springer, Cham, 2021.
[20] M. Coutinho and T.C.S. Neto, "Improved linear approximations to ARX ciphers and attacks against ChaCha," Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp.711–740, Springer, Cham, 2021.
[21] K.K.C. Deepthi and K. Singh, "Cryptanalysis of Salsa and ChaCha: revisited," International Conference on Mobile Networks and Management, pp.324–338. Springer, Cham, 2017.
[22] S. Dey and S. Sarkar, "A theoretical investigation on the distinguishers of Salsa and ChaCha," Discrete Applied Mathematics, vol.302, pp.147–162, Oct. 2021.
[23] S. Dey, H.K. Garai, S. Sarkar, and N. KSharma, "Revamped differential-linear cryptanalysis on reduced round ChaCha," In: O. Dunkelman, S. Dziembowski, (eds) Advances in Cryptology – EUROCRYPT 2022. EUROCRYPT 2022. Lect. Notes Comput. Sci., vol.13277, Springer, Cham. https://doi.org/10.1007/978-3-031-07082-2_4
[24] M. Coutinho and T.C.S.Neto, "New multi-bit differentials to improve attacks against chacha," IACR Cryptol. ePrint Arch. 2020, 350, 2020.
[25] E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," J. Cryptology, vol.4, pp.3–72, 1991. https://doi.org/10.1007/BF00630563
[26] I. Mantin and A. Shamir, "A practical attack on broadcast RC4," In: M. Matsui, (eds) Fast Software Encryption, FSE 2001, Lect. Notes Comput. Sci., vol.2355, Springer, Berlin, Heidelberg, 2002. https://doi.org/10.1007/3-540-45473-X_13
[27] K. Shahram, Neutrality-Based Symmetric Cryptanalysis (Doctoral dissertation, Swiss Federal Institute of Technology Lausanne), Info-Science EPFL Scientific Publication, 2010. https://doi.org/10.5075/epfl-thesis-4755
[28] Y. Tsunoo, T. Saito, H. Kubo, T. Suzaki, and H. Nakashima, "Differential cryptanalysis of Salsa20/8," Workshop Record of SASC, vol.28, 2007.
[29] S. Maitra, G. Paul, and W. Meier, "Salsa20 cryptanalysis: New moves and revisiting old styles," Cryptology ePrint Archive, 2015.
[30] H. Lipmaa and S. Moriai, "Efficient algorithms for computing differential properties of addition," In: M. Matsui, (eds) Fast Software Encryption, FSE 2001, Lect. Notes Comput. Sci., vol.2355, Springer, Berlin, Heidelberg 2002. https://doi.org/10.1007/3-540-45473-X_28



Nasratullah Ghafoori received the B.S. and M.S. degrees in Computer Science and Information Systems respectively. He is currently pursuing his Ph.D. at the Graduate School of Engineering, Osaka University. His research activities are mostly focused on applied cryptography.

Atsuko Miyaji received the B.Sc., the M..Sc., and the Dr. Sci. degrees in mathematics from Osaka University, in 1988, 1990, and 1997 respectively. She is an IPSJ fellow. She joined Panasonic Co., LTD from 1990 to 1998 and engaged in research and development for secure communication. She was an associate professor at the Japan Advanced Institute of Science and Technology (JAIST) in 1998. She joined the computer science department of the University of California, Davis from 2002 to 2003. She has been a professor at Japan Advanced Institute of Science and Technology (JAIST) since 2007. She has been a professor at Graduate School of Engineering, Osaka University since 2015. Her research interests include the application of number theory into cryptography and information security. She received Young Paper Award of SCIS'93 in 1993, Notable Invention Award of the Science and Technology Agency in 1997, the IPSJ Sakai Special Researcher Award in 2002, the Standardization Contribution Award in 2003, the AWARD for the contribution to CULTURE of SECURITY in 2007, the Director-General of Industrial Science and Technology Policy and EnvironmentBureau Award in 2007, DoCoMo Mobile Science Awards in 2008, Advanced Data Mining and Applications (ADMA 2010) Best Paper Award, Prizes for Science and Technology, the Commendation for Science and Technology by the Minister of Education, Culture, Sports, Science and Technology, International Conference on Applications and Technologies in Information Security (ATIS 2016) Best Paper Award, the 16th IEEE Trustocm 2017 Best Paper Award, IEICE milestone certification in 2017, the 14th Asia Joint Conference on Information Security (AsiaJCIS 2019) Best Paper Award, Information Security Applications - 20th International Conference (WISA 2020) Best Paper Gold Award, and Distinguished Educational Practitioners Award in 2020. She is a member of the International Association for Cryptologic Research, the Institute of Electrical and Electronics Engineers, the Institute of Electronics, Information and Communication Engineers, the Information Processing Society of Japan, and the Mathematical Society of Japan.

Ryoma Ito received the B.E. degree from National Defense Academy of Japan in 2009, the M.S. degree from Japan Advanced Institute of Science and Technology in 2015, and the Ph.D. degree from Osaka University in 2019. From 2009 to 2020, he worked at the Japan Air Self-Defense Force, Ministry of Defense, Japan. Since 2020, he has been a senior researcher at National Institute of Information and Communications Technology, Japan. His current research interests include information security and cryptography. He received the SCIS Paper Award from IEICE in 2015, the SCIS Innovation Paper Award from IEICE in 2021, and the IPSJ Yamashita SIG Research Award in 2023. He is a member of IPSJ and IACR.

Shotaro Miyashita received the B. Eng. degrees in mathematics from the National Defense Academy and the M. Eng. degrees in Graduate School of Engineering, Osaka University, in 2015 and in 2021 respectively. He belongs to JASDF (Japan Air Self-Defense Force)