Cryptanalysis of Salsa and ChaCha: Revisited

Kakumani K. C. Deepthi(✉) and Kunwar Singh

Computer Science and Engineering Department,
National Institute of Technology, Tiruchirappalli, India
{406115002,kunwar}@nitt.edu

Abstract. Stream cipher is one of the basic cryptographic primitives that provide the confidentiality of communication through insecure channel. EU ECRYPT network has organized a project for identifying new stream suitable for widespread adoption where the ciphers can provide a more security levels. Finally the result of the project has identified new stream ciphers referred as eSTREAM. Salsa20 is one of the eSTREAM cipher built on a pseudorandom function. In this paper our contribution is two phases. First phase have two parts. In WCC 2015, Maitra et al. [9] explained characterization of valid states by reversing one round of Salsa20. In first part, we have revisited the Maitra et al. [9] characterization of valid states by reversing one round of Salsa20. We found there is a mistake in one bit change in 8^{th} and 9^{th} word in first round will result in valid initial state. In second part, Maitra et al. [9] as mentioned that it would be an interesting combinatorial problem to characterize all such states. We have characterized nine more values which lead to valid initial states. The combinations (s_4, s_7), (s_2, s_3), (s_{13}, s_{14}), (s_1, s_6), (s_1, s_{11}), (s_1, s_{12}), (s_6, s_{11}), (s_6, s_{12}) and (s_{11}, s_{12}) which characterized as valid states.

In second phase, FSE 2008 Aumasson et al. [1] attacked 128-key bit of Salsa20/7 within 2^{111} time and ChaCha6 in within 2^{107} time. After this with best of our knowledge there does not exist any improvement on this attack. In this paper we have attacked 128-key bit of Salsa20/7 within 2^{107} time and ChaCha6 within 2^{102} time. Maitra [8] improved the attack on Salsa20/8 and ChaCha7 by choosing proper IVs corresponding to the 256-key bit. Applying the same concept we have attacked 128-key bit of Salsa20/7 within time 2^{104} and ChaCha7 within time 2^{101}.

Keywords: Stream cipher · eSTREAM · Salsa · ChaCha
Non-randomness · Quarterround · Reverseround · Valid states
Probabilistic neutral bit (PNB) · ARX cipher

1 Introduction

Stream cipher is one of the basic cryptographic primitives that provide the confidentiality of communication through insecure channel. It produces a long pseudorandom sequence called keystream from short random string called secret key

J. Hu et al. (Eds.): MONAMI 2017, LNICST 235, pp. 324–338, 2018.
https://doi.org/10.1007/978-3-319-90775-8_26

(seed). Here encryption of message is carried out bit by bit which can be achieved XORing the keystream to the message. Receiver regenerates the keystream from shared secret key (seed) then decryption is achieved XORing the key stream to the ciphertext. Single secret key is used to encrypt two different messages which is vulnerable to some kind of attack. So, for each and every message there should be different key which cannot be considered practically.

In the modern time, a pseudo-random generator used in stream cipher which depends on a secret key (seed) and Initialization Vector (IV). Here IV does not need to be kept secret and it must change for every encryption session, which it is used as randomizer. The IV is communicated to the receiver publicly (TRIVIUM), or could be combination of this publicly communicated value and a counter value generated at both ends (SALSA 20/12). This gives the flexibility that same key can be used to different messages. Security model of these stream cipher captures the idea that for distinct values of IV with same key, the output of a pseudo-random generator should appear uniform random to an adversary with polynomial bounded computational power.

EU ECRYPT network has organized a project for identifying new stream suitable for widespread adoption where the ciphers can provide a more security levels. Finally the result of the project has identified new stream ciphers referred as eSTREAM. Salsa20 [3] is the one of eSTREAM which provides much better speed-security profile. Salsa20 offers a very simple, clean, and scalable design developed by Daniel J. Bernstein. It supports 128-bit and 256-bit keys in a very natural way. The simplicity and scalability of the algorithm has given more importance in cryptanalytic area. ChaCha [2] is family of stream ciphers, a variant of Salsa published by Daniel J. Bernstein. ChaCha has better diffusion per round and increasing resistance to cryptanalysis.

Related Work: In SASC 2005, Crowley [5] has reported first attack in Salsa20 and won Bernstein's US$1000 prize amount for "most interesting Salsa20 cryptanalysis". He presented an attack on Salsa20 reduced to five of its twenty rounds and is based on truncated differentials. Truncated differential cryptanalysis is a generalization of differential cryptanalysis, an attack against block ciphers. In INDOCRYPT 2006, Fisher et al. [7] has reported in Salsa20/6 and Salsa20/7 an attack and presented a key recovery attack on six rounds and observe non-randomness after seven rounds. In SASC 2007, Tsunoo et al. [12] reported that there is a significant bias in the differential probability for Salsa20's 4^{th} round internal state. In FSE 2008, Aumasson et al. [1] have reported an attack which makes use of the new concept of probabilistic neutral key bits (PNB). PNB is the process of identifying a large subset of key bits which can be replaced by fixed bits so that detectable bias after approximate backwards computation is still significant. This attack was further improved by Shi et al. [11] using the concept of Column Chaining Distinguisher (CCD). By choosing IVs corresponding to the keys, Maitra [8] improved the attack on Salsa20/8. Further the attacks on Salsa and ChaCha were improved by Choudhuri and Maitra [4] and Sabyasachi and Sarkar [6]. All these attacks are on 256 version of Salsa20/8. In WCC 2015, Maitra et al. [9] provided interpretation based on Fisher's 2006

result. Maitra et al. [9] included the key bits in the PNB set by providing less probability for distinguishing. In this way they have considered and obtained $(36+5=41)$ PNBs by key recovery attack with key search complexity of $2^{247.2}$. Maitra et al. [9] characterization of valid initial state by reversing one round of Salsa20, which helps in obtaining sharper bias value. It is found that change in some bit position after one round can obtain the initial value state by reversing one round back. Maitra et al. [9] given that one bit change in 8^{th} and 9^{th} word in first round will result in valid initial state as follows.

$$\begin{bmatrix} 0\ 0\ 0x80000000\ 0 \\ 0\ 0\ 0x80001000\ 0 \\ 0\ 0\ \ 0\ \ 0 \\ 0\ 0\ 0x???80040\ 0 \end{bmatrix} \Leftarrow \begin{bmatrix} 0 & 0 & 0\ 0 \\ 0 & 0 & 0\ 0 \\ 0x80000000 & 0x80000000 & 0\ 0 \\ 0 & 0 & 0\ 0 \end{bmatrix} \tag{1}$$

Our Contribution: Our contribution is of two phases. First phase have two parts. In first part, we have revisited the Maitra et al. [9] characterization of valid states by reversing one round of Salsa20. We found there is a mistake in Eq. (1) in which one bit change of 8^{th} and 9^{th} word in first round will result in valid initial state. Instead of $0x???80040$, it should be as follows.

$$\begin{bmatrix} 0\ 0\ \ 0x80000000\ \ 0 \\ 0\ 0\ \ 0x80001000\ \ 0 \\ 0\ 0\ \ \ \ 0\ \ \ \ 0 \\ 0\ 0\ \mathbf{0x???800??}\ 0 \end{bmatrix} \Leftarrow \begin{bmatrix} 0 & 0 & 0\ 0 \\ 0 & 0 & 0\ 0 \\ 0x80000000 & 0x80000000 & 0\ 0 \\ 0 & 0 & 0\ 0 \end{bmatrix}$$

In second part, Maitra et al. [9], as mentioned that it would be an interesting combinatorial problem to characterize all such states. We have characterized nine more values which lead to valid initial states. The combinations (s_4, s_7), (s_2, s_3), (s_{13}, s_{14}), (s_1, s_6), (s_1, s_{11}), (s_1, s_{12}), (s_6, s_{11}), (s_6, s_{12}) and (s_{11}, s_{12}) which characterized as valid states.

In second phase, FSE 2008 Aumasson et al. [1] attacked 128-key bit of Salsa20/7 within 2^{111} time and ChaCha6 in within 2^{107} time. After this with best of our knowledge there does not exist any improvement on this attack. In this paper we have attacked 128-key bit of Salsa20/7 within 2^{107} time and ChaCha6 within 2^{102} time. Maitra [8] improved the attack on Salsa20/8 and ChaCha7 by choosing proper IVs corresponding to the 256-key bit. Applying the same concept we have attacked 128-key bit of Salsa20/7 within time 2^{104} and ChaCha7 within time 2^{101}.

Paper Outline: Our paper is organized as follows. In Sect. 2, with preliminaries we describe Salsa20 specification, ChaCha specification, differential analysis, PNBs and estimation of the complexity. Section 3, Maitra et al. [9] work characterization of valid state. In Sect. 4, we describe our work. In Sect. 5, we give conclusion and related open problems.

2 Preliminaries

2.1 Salsa20 Specification

Salsa20 is built on a pseudorandom function based on Add-Rotate-XOR (ARX) operations. It carries 32-bit addition, left rotation and bitwise addition as XOR. Salsa20 stream cipher considers 16 words, each of 32-bit. Basic structure is as follows.

$$S = \begin{bmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{bmatrix} = \begin{bmatrix} c_0 & k_0 & k_1 & k_2 \\ k_3 & c_1 & v_0 & v_1 \\ t_0 & t_1 & c_2 & k_4 \\ k_5 & k_6 & k_7 & c_3 \end{bmatrix}$$

The first matrix with $(s_0, s_1, s_2, \ldots, s_{15})$ represents the words of initial states and in second matrix where c_0, c_1, c_2, c_3 are the predefined constants, $k_0, k_1, \ldots, k_7$ represents key of 256-bit, $IV = (t_0, t_1, v_0, v_1)$ represents t_0, t_1 are the 64-bit counter and v_0, v_1 are the 64-bit nonce. Constant values may be changing based on the key value. If we are considering 256-bit key then we can call it as 256-bit Salsa else if we are considering 128-bit key then it is 128-bit Salsa.

The main operation in Salsa20 is carried out by a nonlinear operation called quarterround function. One quarterround function is the four ARX rounds. One ARX round is the one addition (A) plus one cyclic left rotation (R) plus one XOR (X). Quarterround (w, x, y, z) is defined as follows.

$x = x \oplus ((w + z) <<< 7)$
$y = y \oplus ((x + w) <<< 9)$
$z = z \oplus ((y + x) <<< 13)$
$w = w \oplus ((z + y) <<< 18)$

The addition (A) is carried out by two words and result is divided modulo by 2^{32}, cyclic left rotation (R) is carried out for each bit in the given word that is the leftmost bits move to the rightmost positions and exclusive-or (X) is carried out by sum of the two words with carries suppressed.

The rounds in Salsa20 can be performed based on column and row of the matrix as the initial states are considered. So performing round in row of matrix is called as rowround and column of matrix as columnround. If it is a columnround then one columnround is the four quarterrounds, one on each of the four columns of the initial state matrix. If it is rowround then one rowround is the four quarterrounds, one on each of the four rows of the initial state matrix.

Suppose if S is a 16-word input with values $(s_0, s_1, \ldots, s_{15})$ then the rowround(S) and coulmnround(S) is as follows

Rowround:	Columnround:
quarterround (s_0, s_1, s_2, s_3)	quarterround (s_0, s_4, s_8, s_{12})
quarterround (s_5, s_6, s_7, s_4)	quarterround (s_5, s_9, s_{13}, s_1)
quarterround $(s_{10}, s_{11}, s_8, s_9)$	quarterround $(s_{10}, s_{14}, s_2, s_6)$
quarterround $(s_{15}, s_{12}, s_{13}, s_{14})$	quarterround $(s_{15}, s_3, s_7, s_{11})$

The **one round** [9,10] **of the Salsa20** can also be considered by one columnround and one transpose of the initial state matrix (so 12 rounds can be considered as one columnround and one transpose of 12 times). In this paper we

will be considering Salsa20 as 20 rounds in which one round as one columnround and one transpose.

Let $S^{(0)}$ be the initial state S and $S^{(r)}$ be r rounds applied on the initial state S. So that after R rounds it becomes $S^{(R)}$. Then the keystream for 512 bits is obtained as $Z = S + S^{(R)}$.

Reversing One Round: Maitra et al. [9] given that Salsa20 round can be reversible as the state-transition operations are reversible. If $S^{(r+1)}$, then $S^{(r)} =$ reverseround $(S^{(r+1)})$. Where reverseround is the inverse of the round and consists of first transposing the state and then applying the inverse of quarterround for each column by reverseround. Reverseround (w,x,y,z) is as follows.

$w = w \oplus ((z + y) <<< 18)$
$z = z \oplus ((y + x) <<< 13)$
$y = y \oplus ((x + w) <<< 9)$
$x = x \oplus ((w + z) <<< 7)$
Reverseround works as follows.

1. Consider initial matrix after first round be S_1 and perform transpose be S^T.

$$S_1 = \begin{bmatrix} s'_0 & s'_1 & s'_2 & s'_3 \\ s'_4 & s'_5 & s'_6 & s'_7 \\ s'_8 & s'_9 & s'_{10} & s'_{11} \\ s'_{12} & s'_{13} & s'_{14} & s'_{15} \end{bmatrix} \quad S^T = \begin{bmatrix} s'_0 & s'_4 & s'_8 & s'_{12} \\ s'_1 & s'_5 & s'_9 & s'_{13} \\ s'_2 & s'_6 & s'_{10} & s'_{14} \\ s'_3 & s'_7 & s'_{11} & s'_{15} \end{bmatrix}$$

2. Perform reverseround by column wise for each column individually. Where reverseround (w,x,y,z) is equal to $[s'_0, s'_1, s'_2, s'_3]$, $[s'_5, s'_6, s'_7, s'_4]$, $[s'_{10}, s'_{11}, s'_8, s'_9]$ and $[s'_{15}, s'_{12}, s'_{13}, s'_{14}]$.

2.2 ChaCha Specification

ChaCha is a family of stream cipher, a variant of Salsa. It is similar to Salsa is of 16 words, each 32-bit. Basic structure is as follows

$$S = \begin{bmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{bmatrix} = \begin{bmatrix} c_0 & c_1 & c_2 & c_3 \\ k_0 & k_1 & k_2 & k_3 \\ k_4 & k_5 & k_6 & k_7 \\ t_0 & v_0 & v_1 & v_2 \end{bmatrix}$$

The first matrix with $(s_0, s_1, s_2, \ldots, s_{15})$ represents the words of initial states and in second matrix where c_0, c_1, c_2, c_3 are the predefined constants, $k_0, k_1, \ldots, k_7$ represents key of 256-bit, $IV = (t_0, t_1, v_0, v_1)$ represents t_0, t_1 are the 64-bit counter and v_0, v_1 are the 64-bit nonce. In ChaCha nonlinear operations are slightly different from Salsa and are as follows

$w = w + x; z = z \oplus w; z = z <<< 16;$
$y = y + z; x = x \oplus y; x = x <<< 12;$
$w = w + x; z = z \oplus w; z = z <<< 8;$
$y = y + z; x = x \oplus y; x = x <<< 7;$

As in Salsa columnround and rowround are considered, but here in ChaCha columnround and diagonalround are considered. It is considered as for odd rounds first columnround is applied and for even rounds first diagonalround is applied. Columnround and diagonalrounds are as follows

Columnround:	Diagonalround:
quarterround (s_0, s_4, s_8, s_{12})	quarterround $(s_0, s_5, s_{10}, s_{15})$
quarterround (s_1, s_5, s_9, s_{13})	quarterround $(s_1, s_6, s_{11}, s_{12})$
quarterround $(s_2, s_6, s_{10}, s_{14})$	quarterround (s_2, s_7, s_8, s_{13})
quarterround $(s_3, s_7, s_{11}, s_{15})$	quarterround (s_3, s_4, s_9, s_{14})

Like in Salsa, in ChaCha also reverseround is the inverse of round and defined as follows

$x = x >>> 7; x = x \oplus y; y = y - z;$
$z = z >>> 8; z = z \oplus w; w = w - x;$
$x = x >>> 12; x = x \oplus y; y = y - z;$
$z = z >>> 16; z = z \oplus w; w = w - x;$

2.3 Differential Analysis

The differential attack is that some small differences in input states have a perceptible chance in producing small differences after the first round of computation, the second round of computation, etc. The behavior of the differential is heavily key-dependent. There are many unbalanced bits in the states of Salsa20 after four rounds.

Let s_i is the i^{th} word of the matrix S and $s_{i,j}$ is the j^{th} least significant bit of s_i. Suppose if we are having two states $S^{(r)}$ and $S'^{(r)}$, after r rounds it can be denoted as $\Delta_i^{(r)} = s_i^{(r)} \oplus s_i'^{(r)}$. Thus

$$\Delta^{(r)} = S^{(r)} \oplus S'^{(r)} = \begin{bmatrix} \Delta_0^{(r)} & \Delta_1^{(r)} & \Delta_2^{(r)} & \Delta_3^{(r)} \\ \Delta_4^{(r)} & \Delta_5^{(r)} & \Delta_6^{(r)} & \Delta_7^{(r)} \\ \Delta_8^{(r)} & \Delta_9^{(r)} & \Delta_{10}^{(r)} & \Delta_{11}^{(r)} \\ \Delta_{12}^{(r)} & \Delta_{13}^{(r)} & \Delta_{14}^{(r)} & \Delta_{15}^{(r)} \end{bmatrix}$$

After performing few rounds biases can be obtained. For that take Input Differential (ID) at the initial state and try to obtain some biases value corresponding to combinations of some output bits as Output Differential (OD).

Let $S^{(1)}$ and $S'^{(1)}$ be the two initial states that differ in a few places. That is probability is different at the q^{th} bit of the p^{th} word and they are same at all the other bits of the complete state or differ at the j^{th} bit of the i^{th} word and they are same at all the other bits of the complete state is the amount of bias and is computed as $Pr(\Delta_{p,q}^{(r)} = 1 \mid \Delta_{i,j}^{(0)} = 1) = \frac{1}{2}(1 + \epsilon_d)$, where ϵ_d is a measure of the bias. Here the probability is estimated for fixed key and by all the possible choices of nonces and counters, other than the constraints imposed due to the input differences on the nonces or counters.

2.4 Probabilistic Neutral Bits (PNBs)

In FSE 2008, Aumasson et al. [1] have reported an attack which makes use of the new concept of probabilistic neutral key bits (PNB) for probabilistic detection of a truncated differential. In 2015, Maitra et al. [9] revisited the concept and explained the concept in simple manner. PNB is the process of identifying a large subset of key bits which can be replaced by fixed bits so that detectable bias after approximate backwards computation is still significant.

Setting Up: Consider S and S' are the two initial states change in j-th bit of i-th word for the given input differential $\Delta_{i,j}^{(0)}$, by executing Salsa algorithm for r rounds i.e., $r < R$ observed a high bias ϵ_d in the output differential $\Delta_{p,q}^{(r)}$ by randomly choosing keys, nonces and counters. Bias is estimated by $Pr(\Delta_{p,q}^{(r)} = 1 \mid \Delta_{i,j}^{(0)} = 1) = \frac{1}{2}(1 + \epsilon_d)$, where ϵ_d is a measure of the bias.

PNB Concept: Now execute Salsa algorithm for R rounds then $Z = S + S^{(R)}$ and $Z' = S' + S'^{(R)}$ are two keystream blocks. By complementing particular key bit position k in S and S' yields to the states $\overline{S}$ and $\overline{S'}$. After executing reverse $R - r$ rounds by $Z - \overline{S}$ and $Z' - \overline{S'}$ results $\overline{Y}$ and $\overline{Y'}$. Let the difference is considered as $\Gamma_{p,q} = Y_{p,q} \oplus Y'_{p,q}$. Now compare this difference with the previous calculated after r rounds of the Salsa. The bias $Pr(\Delta_{p,q}^{(r)} = \Gamma_{p,q} \mid \Delta_{i,j}^{(0)} = 1)$ is high then the key bit k is probabilistic neutral bit (PNB). The neutrality measure of the key bit is $Pr(\Delta_{p,q}^{(r)} = \Gamma_{p,q} \mid \Delta_{i,j}^{(0)} = 1) = \frac{1}{2}(1 + \gamma_k)$.

To Obtain PNBs: We have experimented for 2^{24} samples (randomly choosing nonce and counter) corresponding to each key bit. We have repeated this process for all 128-key bit to obtain the PNBs. A threshold propability $\frac{1}{2}(1+\gamma)$ is chosen to filter PNBs i.e., if $\gamma_k \geq \gamma$, then key bit k is included in the set of the PNBs. So, the whole set of key bits are divided into PNBs and non-PNBs set. Let n be the set of PNBs and m be set of non-PNBs, then the size of whole set is $(m + n = 128)$.

Attack Idea: Main idea of attack considers search over the key bits which are non-PNBs without knowing the correct values of PNBs. The correct key values have been assigned to the m non-PNBs key bits and random binary values are assigned to the n PNBs key bits in both S and S' yields to $\hat{S}$ and $\hat{S}'$. Then compute $Z - \hat{S}$ and $Z' - \hat{S}'$ and apply R-r rounds on both of them to result $\hat{Y}$ and $\hat{Y}'$. Forward Salsa by r rounds results S^r and S'^r respectively. The difference is $\hat{\Gamma}_{p,q} = \hat{Y}_{p,q} \oplus \hat{Y}'_{p,q}$. Probability is $Pr(\hat{\Gamma}_{p,q} = 1 \mid \Delta_{p,q}^{(r)} = 1) = \frac{1}{2}(1 + \epsilon_a)$ by which PNBs can be identified. The bias after R rounds is $Pr(\hat{\Gamma}_{p,q} = 0) = \frac{1}{2}(1 + \epsilon)$. Where $\epsilon \approx \epsilon_a \cdot \epsilon_d$. One can make exhaustive search over all possible keys (2^{128}). Most probable key is that key which has high bias ϵ^* value. Where ϵ^* median of the all ϵ's.

2.5 Estimation of Complexity

By revisiting FSE 2008, Aumasson et al. [1] the complexity of the attack is given by

$$2^m(N + 2^n P_{fa}) = 2^m N + 2^{256-\alpha}.$$

Here m and n are the number of non-PNBs and PNBs. N is the probabilities of required number of samples and by Neyman-Pearson decision theory

$$N \approx \left(\frac{\sqrt{\alpha \log 4} + \sqrt[3]{1 - \epsilon^{*2}}}{\epsilon^{*2}} \right)^2$$

where $P_{nd} = 1.3 \times 10^{-3}$ and $P_{fa} = 2^{-\alpha}$.

3 Maitra et al. [9] Characterization of Valid State

Maitra et al. [9] has found that one bit change in 8^{th} and 9^{th} word in first round will result in valid initial state as follows.

$$\begin{bmatrix} 0\ 0\ 0x80000000\ 0 \\ 0\ 0\ 0x80001000\ 0 \\ 0\ 0\ 0\ 0 \\ 0\ 0\ 0x???80040\ 0 \end{bmatrix} \Leftarrow \begin{bmatrix} 0 & 0 & 0\ 0 \\ 0 & 0 & 0\ 0 \\ 0x80000000 & 0x80000000 & 0\ 0 \\ 0 & 0 & 0\ 0 \end{bmatrix}$$

Here we have gone through detail calculation of the 8^{th} and 9^{th} word we observed that there is a mistake in Maitra et al. [9] and is as follows.

$$\begin{bmatrix} 0\ 0\ 0x80000000\ 0 \\ 0\ 0\ 0x80001000\ 0 \\ 0\ 0\ 0\ 0 \\ 0\ 0\ \mathbf{0x???800??}\ 0 \end{bmatrix} \Leftarrow \begin{bmatrix} 0 & 0 & 0\ 0 \\ 0 & 0 & 0\ 0 \\ 0x80000000 & 0x80000000 & 0\ 0 \\ 0 & 0 & 0\ 0 \end{bmatrix}$$

According to Maitra et al. [10] given the value for $\Delta_{14}^{(0)}$ is 0x???80040, but we found mistake and the value for $\Delta_{14}^{(0)}$ is 0x???800??. The calculation of $\Delta_{14}^{(0)}$ is as follows.

$$\Delta_{14}^{(0)} = [s_{14}^{(1)} \oplus ((s_{10}^{(0)} + s_6^{(0)}) <<< 7)] \oplus [s_{14}^{(1)} \oplus (s_{10}^{(0)} + (s_6^{(0)} \oplus 0x80001000)) <<< 7]$$

Here $s_6^{(0)} \oplus 0x80001000$ have differential in MSB and 12^{th} bit position. But due to carry propagation to the left of the bit position 12, $(s_{10}^{(0)} + s_6^{(0)})$ become 0x????1000. After 7 bit left rotation, the differential in 12^{th} bit position moves to 19^{th} bit position, which is 0x???800??. The value of '?' depends on $s_6^{(0)}$. So $\Delta_{14}^{(0)}$ should be 0x???800?? instead of 0x???80040.

4 Our Work

4.1 Characterization of Valid States

We have revisited the Maitra et al. [9] considering two bits differences in two words in first round, from this they obtain valid initial state by reversing one round.

We found mistake in Maitra et al. [9] work while characterizing the valid initial state as explained above (Sect. 3).

Maitra et al. [9], as mentioned that it would be an interesting combinatorial problem to characterize all such states. Based upon that we have characterized nine more values which lead to valid initial states. The combinations (s_4, s_7), (s_2, s_3), (s_{13}, s_{14}), (s_1, s_6), (s_1, s_{11}), (s_1, s_{12}), (s_6, s_{11}), (s_6, s_{12}) and (s_{11}, s_{12}) which characterized as valid states. Detail calculation procedure for (s_4, s_7) is as follows.

1. $(s_4, s_7) = (0x80000000, 0x80000000)$
   Differential after first round and Transpose:

$$\Delta^{(1)} = \begin{bmatrix} 0 & 0\,0 & 0 \\ 0x80000000 & 0\,0 & 0x80000000 \\ 0 & 0\,0 & 0 \\ 0 & 0\,0 & 0 \end{bmatrix} \quad (\Delta^{(1)})^T = \begin{bmatrix} 0 & 0x80000000 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0x80000000 & 0 & 0 \end{bmatrix}$$

Differential in reverseround(s_1, s_5, s_9, s_{13}) works as follows.

$$\Delta_5^{(0)} = [s_5^{(1)} \oplus ((s_1^{(1)} + s_{13}^{(1)}) <<< 18)] \oplus [s_5^{(1)} \oplus ((s_1^{(1)} \oplus 0x80000000) + (s_{13}^{(1)} \oplus 0x80000000)) <<< 18]$$

Here $s_1^{(1)} \oplus 0x80000000$ and $s_{13}^{(1)} \oplus 0x80000000$ have differential in MSB. Now adding $s_1^{(1)} \oplus 0x80000000$ and $s_{13}^{(1)} \oplus 0x80000000$ and modulo 2^{32} becomes $s_1^{(1)} + s_{13}^{(1)}$ will result zero. So $\Delta_5^{(0)}$ is zero. This proves that the state $\Delta^{(0)}$ is a valid initial state.

$$\Delta_1^{(0)} = [s_1^{(1)} \oplus ((s_{13}^{(1)} + s_9^{(1)}) <<< 13)] \oplus [(s_1^{(1)} \oplus 0x80000000) \oplus ((s_{13}^{(1)} \oplus 0x80000000) + s_9^{(1)}) <<< 13]$$

Here $s_{13}^{(1)} \oplus 0x80000000$ and $s_9^{(1)}$ changes in MSB of $s_{13}^{(1)} + s_9^{(1)}$. Now $s_{13}^{(1)} \oplus 0x80000000$ perform left rotation of 13 bits changes the 13^{th} least significant bit of $s_{13}^{(1)} + s_9^{(1)}$ to 0x00001000. Now differential moves to 12^{th} bit position. Then XORed with $s_1^{(1)} \oplus 0x80000000$, results 0x80001000. So $\Delta_1^{(0)}$ is 0x80001000.

$$\Delta_{13}^{(0)} = [s_{13}^{(1)} \oplus ((s_9^{(1)} + s_5^{(0)}) <<< 9)] \oplus [(s_{13}^{(1)} \oplus 0x80000000) \oplus (s_9^{(1)} + s_5^{(0)}) <<< 9]$$

By performing addition for $s_9^{(1)} + s_5^{(0)}$ will result zero. Then XORed with $s_{13}^{(1)} \oplus 0x80000000$ results to 0x80000000. So $\Delta_{13}^{(0)}$ is 0x80000000.

$$\Delta_9^{(0)} = [s_9^{(1)} \oplus ((s_5^{(0)} + s_1^{(0)}) <<< 7)] \oplus [s_9^{(1)} \oplus (s_5^{(0)} + (s_1^{(0)} \oplus 0x80001000)) <<< 7]$$

Here $s_1^{(0)} \oplus 0x80001000$ have differential in MSB and 12^{th} bit position. But due to carry propagation to the left of the bit position 12, $(s_5^{(0)} + s_1^{(0)})$ become 0x????1000. After 7 bit left rotation, the differential in 12^{th} bit position moves to 19^{th} bit position, which is 0x???800??. The value of '?' depends on $s_1^{(0)}$. So $\Delta_9^{(0)}$ is 0x???800??. Resultant matrix:

$$\Delta^{(0)} = \begin{bmatrix} 0 & 0x80001000 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0x???800?? & 0 & 0 \\ 0 & 0x80000000 & 0 & 0 \end{bmatrix}$$

By following same above procedure we have characterized the other valid states as $(s_2, s_3), (s_{13}, s_{14}), (s_1, s_6), (s_6, s_{12}), (s_{11}, s_{12}), (s_1, s_{12}), (s_6, s_{11})$ and (s_1, s_{11}) which results valid constants and are as follows.

2. $(s_2, s_3) = (0x80000000, 0x80000000)$

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0x???800?? & 0 & 0 & 0 \\ 0x80000000 & 0 & 0 & 0 \\ 0x80001000 & 0 & 0 & 0 \end{bmatrix} \Leftarrow \begin{bmatrix} 0 & 0 & 0x80000000 & 0x80000000 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

3. $(s_{13}, s_{14}) = (0x80000000, 0x80000000)$

$$\begin{bmatrix} 0 & 0 & 0 & 0x???800?? \\ 0 & 0 & 0 & 0x80000000 \\ 0 & 0 & 0 & 0x80001000 \\ 0 & 0 & 0 & 0 \end{bmatrix} \Leftarrow \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0x80000000 & 0x80000000 & 0 \end{bmatrix}$$

4. $(s_1, s_6) = (0x80000000, 0x80000000)$

$$\begin{bmatrix} 0 & 0x00001000 & 0 & 0 \\ 0x???80000 & 0 & 0 & 0 \\ 0x00000100 & 0x???80000 & 0 & 0 \\ 0x00001000 & 0x00000100 & 0 & 0 \end{bmatrix} \Leftarrow \begin{bmatrix} 0 & 0x80000000 & 0 & 0 \\ 0 & 0 & 0x80000000 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

5. $(s_1, s_{11}) = (0x80000000, 0x80000000)$

$$\begin{bmatrix} 0 & 0 & 0x00000100 & 0 \\ 0x???80000 & 0 & 0x00001000 & 0 \\ 0x00000100 & 0 & 0 & \\ 0x00001000 & 0 & 0x???80000 & 0 \end{bmatrix} \Leftarrow \begin{bmatrix} 0 & 0x80000000 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0x80000000 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

6. $(s_1, s_{12}) = (0x80000000, 0x80000000)$

$$\begin{bmatrix} 0 & 0 & 0 & 0x???80000 \\ 0x???80000 & 0 & 0 & 0x00000100 \\ 0x00000100 & 0 & 0 & 0x00001000 \\ 0x00001000 & 0 & 0 & 0 \end{bmatrix} \Leftarrow \begin{bmatrix} 0 & 0x80000000 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0x80000000 & 0 & 0 & 0 \end{bmatrix}$$

7. $(s_6, s_{11}) = (0x80000000, 0x80000000)$

$$\begin{bmatrix} 0 & 0x00001000 & 0x00000100 & 0 \\ 0 & 0 & 0x00001000 & 0 \\ 0 & 0x???80000 & 0 & 0 \\ 0 & 0x00000100 & 0x???80000 & 0 \end{bmatrix} \Leftarrow \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0x80000000 & 0 \\ 0 & 0 & 0 & 0x80000000 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

8. $(s_6, s_{12}) = (0x80000000, 0x80000000)$

$$\begin{bmatrix} 0 & 0x00001000 & 0 & 0x???80000 \\ 0 & 0 & 0 & 0x00000100 \\ 0 & 0x???80000 & 0 & 0x00001000 \\ 0 & 0x00000100 & 0 & 0 \end{bmatrix} \Leftarrow \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0x80000000 & 0 \\ 0 & 0 & 0 & 0 \\ 0x80000000 & 0 & 0 & 0 \end{bmatrix}$$

9. $(s_{11}, s_{12}) = (0x80000000, 0x80000000)$

$$\begin{bmatrix} 0 & 0 & 0x00000100 & 0x???80000 \\ 0 & 0 & 0x00001000 & 0x00000100 \\ 0 & 0 & 0 & 0x00001000 \\ 0 & 0 & 0x???80000 & 0 \end{bmatrix} \Leftarrow \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0x80000000 \\ 0x80000000 & 0 & 0 & 0 \end{bmatrix}$$

4.2 Attack on Salsa and ChaCha

In FSE 2008, Aumasson et al. [1] attack for 128-key bit of Salsa20/7 within 2^{111} time and ChaCha6 in within 2^{107} time. After this with best of our knowledge there does not exist any improvement on this attack. In this paper we have attacked 128-key bit of Salsa20/7 within 2^{107} time and ChaCha6 within 2^{102} time. Details of the calculation is as follows

Attack on 128-bit Salsa20/7. In FSE 2008, Aumasson et al. [1] use the differential $({\Delta^{(4)}}_{1,14} \mid {\Delta^{(0)}}_{7,31})$ with $|{\epsilon_d}^*| = 0.130$ and $\gamma = 0.4$. With this obtained number of PNBs n = 38, $|{\epsilon_a}^*| = 0.045$ and $|\epsilon^*| = 0.00592$. For $\alpha = 21$, results in time 2^{111} and data 2^{21}. But the list of the PNBs are no mentioned. By considering same differential and $|{\epsilon_d}^*|$ we find number of PNBs n=35 with $|{\epsilon_a}^*| = 0.040$ and $|\epsilon^*| = 0.0052$. For $\alpha = 21$, results in time 2^{114} and data 2^{21}. The list of 35 PNB's and the corresponding γ_k's are shown in Table 1.

In 128-key bit of Salsa20/7 for $\gamma = 0.3$ results in additional 5 PNBs shown in Table 2. and for $\gamma = 0.2$ results in additional 11 PNBs shown in Table 3.

In Table 4. we have compared the results with different γ values for the attack on 128-key bit of Salsa20/7.

Attack on 128-bit ChaCha6. In FSE 2008, Aumasson et al. [1] use the differential $({\Delta^{(3)}}_{11,0} \mid {\Delta^{(0)}}_{13,13})$ with $|{\epsilon_d}^*| = 0.026$ and threshold $\gamma = 0.5$, obtained number of PNBs $n = 51$, $|{\epsilon_a}^*| = 0.013$ and $|\epsilon^*| = 0.00036$. For $\alpha = 26$, results in time 2^{107} and data 2^{30}. But the list of PNBs are not mentioned. By

Table 1. Key-bits and the corresponding γ_k's for the 35 PNBs of the Salsa 7-round attack.

0	1	18	19	20	21	22	23	24	25	26	27	28	29	30
0.59	0.42	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.98	0.98	0.97	0.97
31	33	34	39	43	44	45	57	58	59	65	71	72	73	96
0.96	0.61	0.41	0.45	0.75	0.62	0.45	0.71	0.59	0.45	0.52	0.73	0.60	0.44	0.59
97	110	111	112	116										
0.40	0.76	0.61	0.41	0.41										

Table 2. Key-bits and the corresponding γ_k's for the additional 5 PNBs of the Salsa 7-round attack.

60	66	78
0.30	0.30	0.39
92	124	
0.30	0.30	

Table 3. Key-bits and the corresponding γ_k's for the additional 11 PNBs of the Salsa 7-round attack.

2	6	12	35	40	46
0.23	0.26	0.20	0.20	0.23	0.26
53	74	79	113	117	
0.25	0.26	0.21	0.20	0.24	

considering same requirements we find the time 2^{106} and data 2^{28}. The list of 51 PNB's and the corresponding γ_k's are shown in Table 5.

In 128-key bit of ChaCha6 for $\gamma = 0.4$ results in additional 6 PNBs shown in Table 6. and for $\gamma = 0.3$ results in additional 2 PNBs shown in Table 7.

In Table 8. we have compared the results with different γ values for the attack on 128-key bit of ChaCha6.

4.3 Choosing Proper IVs on Salsa and ChaCha

Maitra [8] improved the attack on Salsa20/8 and ChaCha7 by choosing proper IVs corresponding to the 256-key bit. Further improved key search complexity on Salsa20/7 and ChaCha6. Applying the same concept we have attacked 128-key bit of Salsa20/7 within time 2^{104} and ChaCha7 within time 2^{101}. Details of the calculation is as follows

Attack on 128-bit Salsa20/7. Considering differential $(\Delta^{(4)}{}_{1,14} \mid \Delta^{(0)}{}_{7,31})$ with $s_3 = s_{11} = 0$ and $s_7 = 0xaaaaaaaa$, that is $k_2 = k_4 = 0$ and

Table 4. Different parameters for our attack on 128-key bit Salsa20/7

γ	n	$\lvert\epsilon_a{}^*\rvert$	$\lvert\epsilon^*\rvert$	α	Time	Data
0.3	40	0.023	0.0030	21	2^{110}	2^{22}
0.2	51	0.011	0.0014	21	2^{107}	2^{24}

Table 5. Key-bits and the corresponding γ_k's for the 51 PNBs of the ChaCha 6-round attack.

2	3	8	9	10	11	12	13	14	15	16	19	20	21	22
0.69	0.50	0.99	0.99	0.99	0.98	0.96	0.93	0.87	0.76	0.56	0.96	0.93	0.88	0.76
23	26	27	28	29	30	31	47	63	72	73	95	96	97	98
0.56	0.94	0.90	0.85	0.76	0.65	0.56	0.59	0.76	0.73	0.53	0.87	0.96	0.93	0.87
99	100	103	104	105	108	109	110	111	112	113	114	115	120	121
0.75	0.55	0.87	0.76	0.56	0.99	0.98	0.97	0.96	0.93	0.87	0.79	0.66	1.0	1.0
122	123	124	125	126	127									
1.0	1.0	1.0	1.0	1.0	1.0									

Table 6. Key-bits and the corresponding γ_k's for the additional 6 PNBs of the ChaCha 6-round attack.

35	51	59
0.44	0.43	0.45
64	88	116
0.42	0.41	0.47

Table 7. Key-bits and the corresponding γ_k's for the additional 2 PNBs of the ChaCha 6-round attack.

39	48
0.33	0.33

$v_1 = 0xaaaaaaaa$. Then $|\epsilon_d{}^*| = 0.13225$. If threshold is $\gamma = 0.2$, we find the number of PNBs $n = 41$, $|\epsilon_a{}^*| = 0.230$ and $|\epsilon^*| = 0.0145$. For $\alpha = 26$, results in time 2^{104} and data 2^{17}. The list of 41 PNB's and the corresponding γ_k's are shown in Table 9.

Attack on 128-bit ChaCha6. Considering differential $(\Delta^{(3)}{}_{11,0} \mid \Delta^{(0)}{}_{13,13})$ with $s_5 = s_9 = 0$ and $s_{13} = 0xaaaaaaaa$, that is $k_1 = k_5 = 0$ and $v_0 = 0xaaaaaaaa$. Then $|\epsilon_d{}^*| = 0.041586$. If threshold is $\gamma = 0.4$, we find the number of PNBs $n = 52$, $|\epsilon_a{}^*| = 0.0110$ and $|\epsilon^*| = 0.00045$. For $\alpha = 26$, results in time 2^{101} and data 2^{28}. The list of 52 PNB's and the corresponding γ_k's are shown in Table 10.

Table 8. Different parameters for our attack on 128-key bit ChaCha6

γ	n	$\lvert\epsilon_a{}^*\rvert$	$\lvert\epsilon^*\rvert$	α	Time	Data
0.4	57	0.011	0.00030	26	2^{102}	2^{29}
0.3	59	0.009	0.00025	26	2^{102}	2^{30}

Table 9. Key-bits and the corresponding γ_k's for the 41 PNBs of the Salsa 7-round attack.

0	1	2	6	18	19	20	21	22	23	24	25	26	27	28
0.59	0.42	0.23	0.25	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.98	0.98
29	30	31	33	34	35	39	40	43	44	45	46	53	57	58
0.98	0.97	0.96	0.61	0.41	0.20	0.46	0.23	0.75	0.62	0.45	0.26	0.25	0.71	0.59
59	60	96	97	110	111	112	113	116	117	124				
0.45	0.30	0.60	0.40	0.76	0.61	0.41	0.20	0.414	0.244	0.30				

Table 10. Key-bits and the corresponding γ_k's for the 52 PNBs of the ChaCha 6-round attack.

2	3	8	9	10	11	12	13	14	15	16	19	20	21	22
0.69	0.50	0.99	0.99	0.99	0.98	0.96	0.93	0.87	0.76	0.56	0.96	0.94	0.88	0.76
23	26	27	28	29	30	31	64	72	73	88	95	96	97	98
0.56	0.94	0.90	0.84	0.76	0.66	0.56	0.42	0.73	0.53	0.41	0.87	0.96	0.93	0.87
99	100	103	104	105	108	109	110	111	112	113	114	115	116	120
0.76	0.55	0.87	0.76	0.56	0.99	0.98	0.97	0.96	0.93	0.87	0.79	0.66	0.47	1.0
121	122	123	124	125	126	127								
1.0	1.0	1.0	1.0	1.0	1.0	1.0								

5 Conclusion

In this paper, we explained reverting some differences from the first round leads to valid differentials in the initial state. We characterized different valid states by reversing one round of Salsa20. This can be helpful for producing the sharper biases value. Here we have characterized valid initial states by considering 256-key bit Salsa20. For future work, it will be interesting to characterize valid initial states by considering 128-key bit Salsa20.

We have successfully improved the attack on 128-key bit of Salsa7 and ChaCha6. However our work and previous cryptanalysis result on 128 version of Salsa do not make any threat against Salsa20/12 and Salsa20/20. Finally we hope that these cryptanalysis will result in better understanding of what makes these stream cipher secure.

References

1. Aumasson, J.-P., Fischer, S., Khazaei, S., Meier, W., Rechberger, C.: New features of Latin dances: analysis of Salsa, ChaCha, and Rumba. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 470–488. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-71039-4_30
2. Bernstein, D.J.: Chacha, a variant of Salsa20. In: Workshop Record of SASC, vol. 8, pp. 3–5 (2008)

3. Bernstein, D.J.: Snuffle 2005: the Salsa20 encryption function (2015)
4. Choudhuri, A.R., Maitra, S.: Significantly improved multi-bit differentials for reduced round Salsa and Chacha. IACR Trans. Symmetric Cryptol. **2016**(2), 261–287 (2017)
5. Crowley, P.: Truncated differential cryptanalysis of five rounds of Salsa20. In: The State of the Art of Stream Ciphers SASC, vol. 2006, pp. 198–202 (2006)
6. Dey, S., Sarkar, S.: Improved analysis for reduced round Salsa and Chacha. Discret. Appl. Math. **227**, 58–69 (2017)
7. Fischer, S., Meier, W., Berbain, C., Biasse, J.-F., Robshaw, M.J.B.: Non-randomness in eSTREAM candidates Salsa20 and TSC-4. In: Barua, R., Lange, T. (eds.) INDOCRYPT 2006. LNCS, vol. 4329, pp. 2–16. Springer, Heidelberg (2006). https://doi.org/10.1007/11941378_2
8. Maitra, S.: Chosen IV cryptanalysis on reduced round Chacha and Salsa. Discret. Appl. Math. **208**, 88–97 (2016)
9. Maitra, S., Paul, G., Meier, W.: Salsa20 cryptanalysis: new moves and revisiting old styles. In: 9th International Workshop on Coding and Cryptography, WCC 2015 (2015)
10. Mouha, N., Preneel, B.: Towards finding optimal differential characteristics for ARX: application to Salsa20. Technical report, Cryptology ePrint Archive, Report 2013/328 (2013)
11. Shi, Z., Zhang, B., Feng, D., Wu, W.: Improved key recovery attacks on reduced-round Salsa20 and ChaCha. In: Kwon, T., Lee, M.-K., Kwon, D. (eds.) ICISC 2012. LNCS, vol. 7839, pp. 337–351. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37682-5_24
12. Tsunoo, Y., Saito, T., Kubo, H., Suzaki, T., Nakashima, H.: Differential cryptanalysis of Salsa20/8. In: Workshop Record of SASC, p. 12 (2007)