Revisiting cryptanalysis on ChaCha from Crypto 2020 and Eurocrypt 2021

Sabyasachi Dey[1], Chandan Dey[2], Santanu Sarkar[2] and Willi Meier[3]

Abstract

ChaCha has been one of the prominent ARX designs of the last few years because of its use in several systems. The cryptanalysis of ChaCha involves a differential attack which exploits the idea of Probabilistic Neutral Bits (PNBs). For a long period, the single-bit distinguisher in this differential attack was found up to 3 rounds. At Crypto 2020, Beierle et. al. introduced for the first time single bit distinguishers for 3.5 rounds, which contributed significantly in regaining the flow of research work in this direction. This discovery became the primary factor behind the huge improvement in the key recovery attack complexity in that work. This was followed by another work at Eurocrypt 2021, where a single bit distinguisher of 3.5-th round helped to produce a 7-round distinguisher of ChaCha and a further improvement in key recovery.

In the first part of this paper, we provide the theoretical framework for the distinguisher given by Beierle et. al. We mathematically derive the observed differential correlation for the particular position where the output difference is observed at 3.5 rounds. Also, Beierle et. al. mentioned the issue of the availability of proper IVs to produce such distinguishers, and pointed out that not all keys have such IVs available. Here we provide a theoretical insight of this issue.

Next we revisit the work of Coutinho et. al. (Eurocrypt 2021). Using Differential-Linear attacks against ChaCha, they claimed distinguisher and key recovery with complexities 2^{218} and $2^{228.51}$ respectively. We show that the differential correlation for 3.5 rounds is much smaller than the claim of Coutinho et. al. This makes the attack complexities much higher than their claim.

Index Terms

Stream Cipher, ChaCha, Correlation, Theoretical interpretation

I. Introduction

Symmetric key ciphers play an important role in secure communication. The ARX (Addition, Rotation and XOR) based symmetric key ciphers are very popular because of their security and efficiency in software. Also, the algebraic degree of round functions of ARX ciphers increases highly after some rounds. The Salsa20 is an ARX based stream cipher designed by Bernstein in 2005 and Salsa20/12 was selected in eStream project as a finalist in software portfolio [5]. In this paper, we will discuss the ARX-based stream cipher ChaCha, which is a variant of Salsa20. ChaCha was also designed by Bernstein in early 2008 for high performance in software implementation. Google adopted ChaCha for symmetric encryption in TLS [13]. ChaCha is also used as a pseudo-random number generator in different systems, for example, in any operating system running Linux kernel 4.8 or newer [17], [20]. Additionally, ChaCha is used in several applications, like WireGuard (VPN), Keepass (password manager), and Veracrypt (disk encryption).

So it is necessary to do third-party analysis for a cipher with such extensive use, to understand its security level. There are many crypatanalysis results available on reduced version of ChaCha till now. Security analysis has been done mainly using differential cryptanalysis. Differential attack is a very popular statistical attack, the main idea of which is as follows: In the initial state some difference is given in the input and we suppose after r rounds we get some biases in the output. Then we say that this cipher is distinguishable up to r rounds and this bias is called forward bias.

Initially some attacks on this cipher were proposed in [9], [12], [21]. One of the most effective attack was proposed in [1] by introducing the Probabilistic Neutral Bits (PNBs) against the reduced round versions of Salsa20 and ChaCha with time complexities 2^{251} and 2^{248} respectively. In 2012, Shi et. al. [19] gave the concept of Column Chaining Distinguisher(CCD) to further enhance the attack complexity of both Salsa20 and ChaCha. Later Maitra et. al. [14] did some improvement on the existing attack by the idea of chosen IV. Dey and Sarkar in [10] showed how to choose PNB for further enhancement of previous works. At FSE 2017, Choudhuri and Maitra [2] gave a technique to construct multiple bit distinguishers in the next rounds and perform a differential-linear attack.

In [1], a distinguisher was observed in the 3rd round of ChaCha. Using this distinguisher a key recovery attack was provided for 7-round ChaCha using a meet-in-the-middle approach. After that, several works have added ideas in this approach and reductions have been achieved in the complexity. But for a long period of more than ten years, no significant improvement was made in finding a distinguisher in the next rounds. In this meet-in-the-middle attack approach, the attack can be improved up to the next round in two possible ways, either by finding a distinguisher in the fourth round, or by improving the technique

[1]Department of Mathematics, Birla Institute of Technology and Science Pilani, Hyderabad, Jawahar Nagar, Hyderabad 500078, India
[2]Department of Mathematics, Indian Institute of Technology Madras, Sardar Patel Road, Chennai TN 600036, INDIA;
[3]University of Applied Sciences and Arts Northwestern Switzerland (FHNW), Windisch, Switzerland
E-mail: sabya.ndp@gmail.com, c.dey.math95@gmail.com, sarkar.santanu.bir1@gmail.com, willimeier48@gmail.com

of coming back from the final round. The idea of Choudhuri and Maitra, provided a significant progress in creating multi-bit distinguishers in the next rounds, but the source of this multi-bit distinguisher was a single bit distinguisher in third round only. So, though this idea significantly improved the key recovery attack complexity for fewer rounds, for 7-round ChaCha the improvement was not very significant. Up to 2020, there was no single bit distinguisher known for more than 3 rounds of ChaCha. This is the reason why no major breakthrough was achieved in the key recovery. Finally, at Crypto 2020, a vital contribution in this direction has been given by Beierle et. al. [4]. In their work they have found a single bit distinguisher in the 3.5 rounds. The correlation observed for this distinguisher is very low compared to the known distinguishers in the third round. But this extension by half a round resulted in a significant jump in the complexity reduction for 7 round ChaCha. According to [4] the new time complexity has come down to $2^{230.86}$ from $2^{235.22}$. For the 6-round version of ChaCha, this work provided for the first time a partial key recovery attack with very low complexity $2^{77.4}$. As an impact of the discovery of the 3.5 rounds distinguisher, an immediate further improvement in this direction came from Coutinho et. al., which will appear at Eurocrypt 2021 [8]. They used the linear correlation of a single bit of 3.5 rounds to the XOR of 2 bits of the 4-th rounds, thus found a 4-rounds distinguisher of 2 bits only. Using this they produced a key recovery attack of time complexity $2^{228.51}$ for 7 rounds of ChaCha.

A full mathematical explanation of the observed correlation is important for a theoretical foundation of the result. In this paper, we have taken care of the theoretical side of the distinguisher. We derived mathematically the observed result for the input-output pairs. Also, for the 3.5 rounds distinguisher, the authors of [4] mentioned about the minimization of the Hamming weight of the difference matrix in the first round. But it has been also mentioned that such minimization of the Hamming weight is not possible for any random key. Approximately 70% of the keys have been considered as weak keys for which such IVs are available which minimizes the Hamming weight. The remaining keys they have considered as strong keys. Here, we have also given a theoretical insight into this and figured out the reason behind this. Also, we reviewed the 3.5 rounds single bit distinguisher of ChaCha from Eurocrypt 2021 paper by Coutinho et. al. [8]. We provided theoretical illustration of the correlations at position $(0, 0)$ and $(1, 0)$ of 3.5 rounds ChaCha from [8]. Then we identified incompatibility between our result and their experimental result. After that, we experimented with all the single bit distinguishers of 3.5 rounds ChaCha with a sufficient number of random samples and found a mismatch between their and our results. If we use our 3.5 distinguisher at position $(5, 0)$ to compute the key recovery attack complexity, it increases the time complexity over that of the Crypto 2020 paper [4]. Therefore the key recovery attack at Eurocrypt 2021 paper by Coutinho et. al. [8] has not improved the existing work. Finally, the novelty of our work is that we have provided theoretical interpretation of experimentally obtained 3.5 rounds distinguisher of ChaCha from Crypto 2020 paper [4] and also, we have shown that the recently improved distinguisher and key recovery attack complexities of Eurocrypt 2021 paper [8] are not accurate.

Roadmap: We organize the paper in the following manner: section II describes the structure of the cipher. In section III we discuss some results related to correlation in mathematical operations, mainly addition. After that, in section IV we provide theoretical results related to the first round of the cipher. This involves the bit positions containing the differences after the first round (corresponding to an input difference) and the issue of availability of IVs to produce the 3.5 round distinguisher. After this, we move to the theoretical part of explaining the distinguisher. Here we observe that the distinguisher at 3.5 rounds depends on the correlations of two positions $(2, 0)$ and $(7, 0)$ at the third round. So, our entire analysis of this distinguisher has been divided into two parts. section V discusses about the generation of correlation at $(2, 0)$ step by step and section VI analyzes the same for $(7, 0)$. Then in section VII, we combine the two results to find the correlation value for the final distinguisher and discuss the generation of multi-bit distinguishers in the 5-th round. In section VIII we discuss the correctness of the 3.5 rounds single bit distinguisher of ChaCha from the Eurocrypt 2021 paper [8]. In this section, we have pointed out that there is a huge gap between their obtained time complexity and the actual time complexity.

Notations: Here we give some notations which we have used throughout this paper.

- A general state of ChaCha consists of 32-bit numbers called 'word'. A n-bit word is basically a concatenation of n bits. In our paper, for convenience we use both tuple form and concatenation form to express a word. A n-bit word x is also treated as an n-tuple vector from $\mathbb{F}_2^n$.
- $\oplus$ denotes the bit-wise XOR and $+$ denotes Addition modulo 2^n and $a \lll b$ is used for Left cyclic shift of word a by b bits.
- We treat a n bit word as a n tuple vector and for a Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ we define correlation as: $\mathbf{Cor}[f(x)] = \frac{1}{2^n}\sum_{x\in\mathbb{F}_2^n}(-1)^{f(x)}$.
- $[i]$ denotes the unit vector of $\mathbb{F}_2^n$ whose i-th bit is 1, i.e.,

$$[i] = (0, 0, 0, \cdots, \underset{\substack{\uparrow \\ i\text{–th bit}}}{1}, \cdots, 0).$$

- The sum of n unit vectors is denoted as $[i_1, i_2, \cdots, i_n] = \oplus_{k=1}^{n}[i_k]$
- $x[i]$: i-th bit of word/vector x i.e, $x = x[n-1]x[n-2]\cdots x[i]\cdots x[0]$ or $x = (x[n-1], x[n-2], \cdots, x[i], \cdots, x[0])$.

- $Car[i]$: The addition operation involves a carry at each bit. In an addition operation of two n-bit numbers x, y modulo 2^n, considering all the n carry bits together as vector $Car[i]$ denotes the i-th carry bit. Similarly, $\widetilde{Car}[i]$ is the same for two n-bit numbers $\tilde{x}, \tilde{y}$.
- Equality function: For two n-bit vectors/words u and v, the equality function is defined as

$$eq(u,v) \quad = \quad \begin{cases} 1, & \text{if } u = v, \\ 0, & \text{if } u \neq v \end{cases}$$

- $x[i_2, i_1]$: For any word/vector x, this denotes the word/vector formed by the bits starting from $x[i_1]$ to $x[i_2]$, i.e, if $x = x[n-1]x[n-2]\cdots x[i_2]\cdots x[i_1]\cdots x[0]$ where $i_2 > i_1$. then $x[i_2, i_1] = x[i_2]x[i_2 - 1]\cdots x[i_1]$.
- $x(i_2, i_1)$: For an n-bit vector/word x, $x(i_2, i_1)$ denotes the n bits vector/word whose bits from i_1 to i_2 are same as that of $x[i_2, i_1]$ and all other bits are zero. This means, for $x = x[n-1]x[n-2]\cdots x[i_2]\cdots x[i_1]\cdots x[0]$, $x(i_2, i_1) = 00\cdots 0x[i_2]x[i_2 - 1]\cdots x[i_1]0\cdots 0$.
- $\big|A\big|$ denotes the cardinality of a set A.
- **Pr** denotes the probability. Suppose A is an event defined on $x, y \in \mathbb{F}_2^n$, then $\mathbf{Pr}(A) = \frac{\big|\text{set of values } (x,y) \text{ satisfying } A\big|}{\big|\text{set of all possible values of } (x,y)\big|}$. For a Boolean function f, if we consider the event $f(x) = 0$, then probability and correlation is related as follows: $\mathbf{Pr}(f(x) = 0) = \frac{1}{2}[1 + \mathbf{Cor}[f]]$
- For two bits $x[i]$ and $\tilde{x}[i]$ the XOR-difference is denoted by $\Delta x[i]$ i.e, $\Delta x[i] = x[i] \oplus \tilde{x}[i]$. In a differential attack, for a chosen input difference, any initial state x generates an another initial state $\tilde{x}$, where $\Delta x = x \oplus \tilde{x}$. Running the operations of the cipher on both $x, \tilde{x}$, if $y, \tilde{y}$ is produced and for any chosen position i, $\Delta y[i]$ is computed, then $x \to y[i]$ is a function. For this function we can compute $\mathbf{Cor}[\Delta y[i]]$ and we use $\theta_{y[i]}$ to denote it, i.e, $\theta_{y[i]} = \mathbf{Cor}[\Delta y[i]]$.
- In a state matrix of ChaCha, position (p, q) means q-th bit of p-th word.
- $X^{(r)}, X', X''$: Usually, the state matrix at r-th round is denoted by $X^{(r)}$. Here we have used this notation very occasionally, in the cipher description and in linear correlation between 3.5 round and the 5-th round. In our proofs we avoided this notation to avoid unnecessary cluttering. Rather, in each round we have gone for X, X', X'' to denote initial state, middle state (half round) and final state. However, in the statements of the results in section V, section VI, section VII we have used only the X notation since the stage is obviously clear to the reader.

II. Description of the structure of ChaCha

A general state of ChaCha is of size 64 bytes or 512 bits, which is divided into 16 words, each of which is of 32 bits. These words are framed in the form of a 4×4 matrix. In the initial matrix, which we denote by $X^{(0)}$, the 1st row consists of 4 constant words $c_0 = 0x61707865, c_1 = 0x3320646e, c_2 = 0x79622d32, c_3 = 0x6b206574$.

The second and third row consist of 8 key words $k_0, k_1, \cdots, k_7$ and the fourth row consists of the two 32 bits nonce v_0, v_1 and two 32 bits counter t_0, t_1. We have considered the nonce and the counter as IVs.

$$X^{(0)} = \begin{pmatrix} X_0^{(0)} & X_1^{(0)} & X_2^{(0)} & X_3^{(0)} \\ X_4^{(0)} & X_5^{(0)} & X_6^{(0)} & X_7^{(0)} \\ X_8^{(0)} & X_9^{(0)} & X_{10}^{(0)} & X_{11}^{(0)} \\ X_{12}^{(0)} & X_{13}^{(0)} & X_{14}^{(0)} & X_{15}^{(0)} \end{pmatrix} = \begin{pmatrix} c_0 & c_1 & c_2 & c_3 \\ k_0 & k_1 & k_2 & k_3 \\ k_4 & k_5 & k_6 & k_7 \\ t_0 & t_1 & v_0 & v_1 \end{pmatrix}.$$

The rows and columns are updated by an operation called quarter-round($\mathcal{QR}$) which operates on a 4 tuple (a, b, c, d) and updates it as follows.

$$\begin{aligned} a' &= a + b, \\ d' &= ((d \oplus a') \lll 16), \\ c' &= c + d', \\ b' &= ((b \oplus c') \lll 12), \\ a'' &= a' + b', \\ d'' &= ((d' \oplus a'') \lll 8), \\ c'' &= c' + d'', \\ b'' &= ((b' \oplus c'') \lll 7). \end{aligned}$$

i.e,

$$\mathcal{QR}(a, b, c, d) \to (a'', b'', c'', d'')$$

This operation is applied on the 4 words of each column in the odd rounds and each diagonal in the even rounds. These rounds are therefore called column rounds and diagonal rounds respectively. The order of choosing (a, b, c, d) for each column and diagonal is from top to bottom i.e, more specifically for the column round or odd round it operates on the input state matrix as follows

$$\{\mathcal{QR}(X_0, X_4, X_8, X_{12}), \mathcal{QR}(X_1, X_5, X_9, X_{13}), \mathcal{QR}(X_2, X_6, X_{10}, X_{14}), \mathcal{QR}(X_3, X_7, X_{11}, X_{15})\}$$

and after that the state matrix is updated and for the diagonal round or even round it operates on the input sate matrix as follows

$$\{\mathcal{QR}(X_0, X_5, X_{10}, X_{15}), \mathcal{QR}(X_1, X_6, X_{11}, X_{12}), \mathcal{QR}(X_2, X_7, X_8, X_{13}), \mathcal{QR}(X_3, X_4, X_9, X_{14})\}$$

and then the state matrix is updated. Also, later on in the paper we have used the term first half of the round and second half of the round which means in the $\mathcal{QR}$ operation update of (a, b, c, d) to (a', b', c', d') and update of (a', b', c', d') to (a'', b'', c'', d'') respectively. Suppose after r rounds the updated state matrix is denoted by $X^{(r)}$.

$$X^{(r)} = \begin{pmatrix} X_0^{(r)} & X_1^{(r)} & X_2^{(r)} & X_3^{(r)} \\ X_4^{(r)} & X_5^{(r)} & X_6^{(r)} & X_7^{(r)} \\ X_8^{(r)} & X_9^{(r)} & X_{10}^{(r)} & X_{11}^{(r)} \\ X_{12}^{(r)} & X_{13}^{(r)} & X_{14}^{(r)} & X_{15}^{(r)} \end{pmatrix}$$

The output key-stream block Z is executed as $Z = X^{(0)} + X^{(R)}$ for ChaCha20/R, where $X^{(R)}$ denotes the state after R rounds. The quarter round($\mathcal{QR}$) operation is reversible which means if we know the values of (a'', b'', c'', d''), we can compute the values of (a, b, c, d). Hence the rounds of ChaCha are also reversible. For more details description of ChaCha we refer to [6].

In this paper, in every round, we have used the notation X, X', X'' to denote initial, middle and final state. This means, X'' in r-th state becomes X in $(r + 1)$-th state.

III. Some results on the differential and linear properties of mathematical operations

We start by stating the well-known ***Piling-up Lemma*** [15] in our notation which we use afterwards in several proofs.

Lemma 1
(Piling-up Lemma): Let $Y_i (1 \leq n)$ be n independent random variables such that $\mathbf{Cor}[Y_i] = \epsilon_i$. Then $\mathbf{Cor}[Y_1 \oplus Y_2 \oplus \cdots \oplus Y_n]$ is $\prod_{i=1}^{n} \epsilon_i$.

Next we discuss some properties on the correlation of the bits when two terms are added modulo 2^n. In this regard, we introduce a n-tuple Car, which represents the carries of n positions in the addition operation. In the addition $z = x + y$, $Car[i]$ denotes the i-th carry bit which is added to $x[i], y[i]$ to generate $z[i]$. So, $z[i] = x[i] \oplus y[i] \oplus Car[i]$. By $\widetilde{Car}$ we denote the carry vector for $\tilde{z} = x + \tilde{y}$.

For $x, y \in \mathbb{F}_2^n$, we define $\tilde{y}[i] = y[i]\ \forall\ i \in [1, n-1] \cap \mathbb{Z}$ and $\tilde{y}[0] = y[0] \oplus 1$. Now, considering all these vectors as n-bit numbers, suppose $z = x + y \pmod{2^n}$ and $\tilde{z} = x + \tilde{y} \pmod{2^n}$. We define for any $k \in [0, n-1] \cap \mathbb{Z}$, $f_k = eq(\Delta z(k, 0), [k, k-1, k-2, \cdots, 0])$. Therefore f_k is a Boolean function from $\mathbb{F}_2^n \times \mathbb{F}_2^n$ to $\mathbb{F}_2$. For any such f_k we have the following theorem:

Theorem 1
For any $f_k : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2$ as defined above:

$$\mathbf{Cor}[f_k] = 1 - \frac{1}{2^{k-1}}.$$

Proof. We have 2^n possible x and 2^n possible y. So, (x, y) has 2^{2n} possible values. In the 0-th bit, clearly, $z[0] \neq \tilde{z}[0]$. Therefore for all values of (x, y) this is true. So, $eq(\Delta z(0, 0), [0]) = 1$ for any (x, y). Therefore, for $k = 0$,

$$\mathbf{Cor}[eq(\Delta z(0, 0), [0])] = \frac{1}{2^{2n}} \sum_{(x,y) \in F^{2n}} (-1)^1 = -1 = 1 - \frac{1}{2^{k-1}}.$$

So, we prove it for $k = 0$. Now, $z[1] \neq \tilde{z}[1]$ if and only if the carry generated at the 0-th bit of z and $\tilde{z}$ is different. Now, this is possible iff $x[0] = 1$, which is true for half of the possible values. In general, all the bits from 0-th to the k-th bit

of z and $\tilde{z}$ are different iff in all the bits the generated carry is different for z and $\tilde{z}$. This is possible if for all of the bits $x[0], x[1], x[2], \cdots, x[k-1]$, $x[i] \neq y[i]$, i.e., $x[i] = y[i] \oplus 1$. This is true for a fraction $\frac{1}{2^k}$ of all possible values of (x, y). So,

$$\begin{aligned}\mathbf{Cor}\big[eq(\Delta z(k,0),[k,k-1,\cdots,0])\big] &= \frac{1}{2^{2n}}\big[2^{2n-k}.(-1)^1 + (2^{2n} - 2^{2n-k})(-1)^0\big] \\ &= \frac{1}{2^{2n}}\big[-2^{2n-k} + (2^{2n} - 2^{2n-k})\big] \\ &= \frac{1}{2^{2n}}\big[2^{2n} - 2^{2n+1-k}\big] \\ &= 1 - \frac{1}{2^{k-1}}.\end{aligned}$$

□

Next we generalise this idea further. Consider x, y same as above. But construct $\tilde{y}$ as: for some integer $r \in [0, n-1] \cap \mathbb{Z}$, $\tilde{y}[i] = y[i]\ \forall\ i \in [0, n-1] \cap \mathbb{Z}$ except $i = r$ and $\tilde{y}[r] = y[r] \oplus 1$. Then, for $z, \tilde{z}$ same as before we define f'_k as $f'_k = eq(\Delta z(r+k, r), [r+k, r+k-1, r+k-2, \cdots, r])$. Then we have the following theorem.

Theorem 2
For any f'_k as defined above where $k \geq 0$ and $r + k \leq n-1$, $\mathbf{Cor}[f'_k] = 1 - \frac{1}{2^{k-1}}$.

Proof. Now, if $Car[r] = \widetilde{Car}[r] = 0$, then this situation is analogous to Theorem 1, except for the fact that here the difference is at the r-th bit. So, in similar manner as in Theorem 1, we have the result.
Now, if $Car[r] = \widetilde{Car}[r] = 1$, then the carry difference would propagate to the next bit iff $x[r] = 0$. In general, iff $x[r] = x[r+1] = \cdots = x[r+k-1] = 0$, then the carry difference would propagate to the k-th bit. This is true for a fraction $\frac{1}{2^k}$ of all possible values of (x, y). So, again in similar manner as Theorem 1, we have the same result. □

Corollary 1
Let $x, y, \tilde{y}$ be n-bit numbers such that $\tilde{y}[i] = y[i]\ \forall i \in [0, n-1] \cap \mathbb{Z}$ except $\tilde{y}[i] = y[i] \oplus 1\ for\ i \in [0, r-1] \cap \mathbb{Z}$. Then between $z = x + y$ and $\tilde{z} = x + \tilde{y}$, $\mathbf{Cor}\big[eq(\Delta z(r+k, r), [r+k, r+k-1, r+k-2, \cdots, r])\big] = 1 - \frac{1}{2^{k-1}}$.

Proof. This is a special case of Theorem 2. □

Now, in the quarter round function of ChaCha we see that an addition is followed by a XOR-cum-rotation and then another addition operation occurs. Now, in the last two theorems we have got the idea that if the initial difference is at a single position then what happens to the correlation of the bits of the sum (i.e., $(z, \tilde{z})$) after the addition. Now, the correlation values of the sum moves to another word and then that updated word is used in the next addition operation. So, the scenario no is as follows: we have addition between $x + y$ and $x + \tilde{y}$ where $(y, \tilde{y})$ is selected from a distribution which already follows the property that $\mathbf{Cor}\big[eq(\Delta z(r+k, r), [r+k, r+k-1, r+k-2, \cdots, r])\big] = 1 - \frac{1}{2^{k-1}}$. For a formal representation, we construct a set of values for $(y, \tilde{y})$ which would follow this correlation.

Let us fixed some r, where $r \in [0, n-1] \cap \mathbb{Z}$. Then for any $y \in \mathbb{F}_2^n$ we define an n bit vector $\tilde{y}^k$, where $k \in [0, n-r-1] \cap \mathbb{Z}$ such that

$$\tilde{y}^k[i] = \begin{cases} 1 \oplus y[i] & \text{if } i \in [r, r+k] \cap \mathbb{Z} \\ y[i] & \text{otherwise.} \end{cases}$$

Now, let us construct a set $S_k = \{(y, \tilde{y}^k) : y \in \mathbb{F}_2^n\} \subset \mathbb{F}_2^n \times \mathbb{F}_2^n$ such that $|S_k| = \frac{1}{2^k}|S_0|$ for $k \in [0, n-r-2] \cap \mathbb{Z}$ and $|S_{n-r-2}| = |S_{n-r-1}|$. Let $S = \bigcup_{k=0}^{n-r-1} S_k$, where $S_0, S_1, \cdots S_{n-r-1}$ are disjoint and so $|S| = 2|S_0|$. Then one can easily verify that in the set S

$$\mathbf{Cor}\big[eq(\Delta z(r+k, r), [r+k, r+k-1, \cdots, r])\big] = 1 - \frac{1}{2^{k-1}}$$

holds.

Theorem 3
For S as defined above, let us define the function $g_k : \mathbb{F}_2^n \times S \to \mathbb{F}_2^n$ as

$$g_k = \Delta z[k], \qquad (k \geq 1)$$

where $z = x + y$, $\tilde{z} = x + \tilde{y}$ and $x \in \mathbb{F}_2^n$; $y, \tilde{y} \in S$. Then

$$\mathbf{Cor}[g_k] = 1 - \frac{k+1}{2^k}$$

Proof. For convenience we assume that $r = 0$. Let us take set $S' = \mathbb{F}_2^n \times S$ and similarly, $S'_i = \mathbb{F}_2^n \times S_i$, where S and S_i defined as above. Also, $|S'| = 2^n \times 2|S_0| = 2^{n+1}|S_0|$ and $|S'_i| = 2^n \times |S_i| = 2^{n-i}|S_0|$ for $i \in [0, n-2] \cap \mathbb{Z}$. Now, consider

$$\mathbf{Cor}\big[\Delta z[k]\big] = \frac{1}{|S'|}\Sigma_{(x,y,\tilde{y})\in S'}(-i)^{\Delta z[k]}$$

and for S'_i,

$$\mathbf{Cor}\big[\Delta z[k]\big] = \frac{1}{|S'_i|}\Sigma_{(x,y,\tilde{y}^k)\in S'_i}(-i)^{\Delta z[k]}.$$

Now in the proof of Theorem 1 we see $\Delta\widetilde{Car}[1] = 1$ for exactly half of the cases. Now if $\Delta Car[1] = 1$, without loss of generality let us assume $Car[1] = 0$ and $\widetilde{Car}[1] = 1$. Then $\Delta Car[2] = 1$ iff $x[1] = 0$ (as $\Delta y[1] = 1$). Also, if $\Delta Car[1] = 0$ then $\Delta Car[2] = 1$ iff $x[1] = 1$. So, both of these occur for exactly half of the cases. Therefore, $\Delta Car[2] = 1$ for exactly half of the cases. Thus we can proceed up to the k-th bit and so $\Delta Car[i] = 1$ for exactly half of the cases. Now, from $(i+1)$-bit onward the scenario becomes same as Theorem 2. So, in S'_i we have $\mathbf{Cor}\big[\Delta z[k]\big] = 1 - \frac{1}{2^{k-i-1}}$. Therefore, in S' we get

$$\mathbf{Cor}\big[\Delta z[k]\big] = \frac{1}{|S'|}\big[\Sigma_{i=0}^{k-1}(1 - \frac{1}{2^{k-i-1}}) \cdot |S'_i| + |S'_k \cup S'_{k+1} \cup \cdots \cup S'_{n-1}|\big]$$

Now for the set $S'_k \cup S'_{k+1} \cup \cdots \cup S'_{n-1}$, all the pairs $(y, \tilde{y})$ have a difference up to the k-th bit. Therefore, in this set, $\Delta Car[k] = 0$ for exactly half of the samples and $\Delta z[k] = 0$ for exactly half of the samples. Therefore, in this set, $\mathbf{Cor}\big[\Delta z[k]\big] = 0$. So,

$$\begin{aligned}
\mathbf{Cor}\big[\Delta z[k]\big] &= \frac{1}{|S'|}\big[\Sigma_{i=0}^{k-1}(1 - \frac{1}{2^{k-i-1}}) \cdot |S'_i|\big] \\
&= \frac{1}{2^{n+!}|S_0|}\big[\Sigma_{i=0}^{k-1}(1 - \frac{1}{2^{k-i-1}}) \cdot 2^{n-i}|S_0|\big] \\
&= \frac{1}{2}\big[\Sigma_{i=0}^{k-1}(\frac{1}{2^i} - \frac{1}{2^{k-1}})\big] \\
&= 1 - \frac{k+1}{2^k}
\end{aligned}$$

□

Next, for our convenience we introduce a notation θ. For any vector pair $x, \tilde{x}$; $\theta_{x[i]}$ represents $\mathbf{Cor}[\Delta x[i]]$. Suppose all the notations are same as in the last theorem. Then we have the following two theorems.

Theorem 4
Consider $x, y, \tilde{x}, \tilde{y}, z, \tilde{z}$ such that $z = x + y, \tilde{z} = \tilde{x} + \tilde{y}$. If $\theta_{x[i]}, \theta_{y[i]}, \theta_{Car[i]}$ is known for some i, we can compute $\theta_{Car[i+1]}$ as follows:

$$\theta_{Car[i+1]} = \frac{1}{4}\big[\theta_{x[i]} + \theta_{y[i]} + \theta_{Car[i]}(1 + \theta_{x[i]}\theta_{y[i]})\big].$$

Proof. We consider two different cases, i.e., $\Delta Car[i] = 0$ and $\Delta Car[i] = 1$.

$\Delta Car[i] = 0$: We focus on the situation when $Car[i] = \widetilde{Car}[i] = 0$. We divide the event of $Car[i+1] \neq \widetilde{Car}[i+1]$ in two sub-cases, $\Delta y[i] = 1$ and $\Delta y[i] = 0$. If $\Delta y[i] = 1$, without loss of generality we assume that $y[i] = 1, \tilde{y}[i] = 0$. Then, the carry difference would propagate to the next bit if and only if $x[i] = 1$. So, for this case the probability is

$$\begin{aligned}
\mathbf{Pr}(\Delta Car[i+1] = 1, \Delta y[i] = 1 | \Delta Car[i] = 0) &= \mathbf{Pr}(x[i] = 1) \cdot \mathbf{Pr}(\Delta y[i] = 1) \\
&= \frac{1}{2} \cdot \frac{1}{2}(1 - \theta_{y[i]}) = \frac{1}{4}(1 - \theta_{y[i]}).
\end{aligned}$$

If $\Delta y[i] = 0$, then the carry difference would occur if and only if $y[i] = \tilde{y}[i] = 1$ and $\Delta x[i] = 1$, whose probability is

$$\begin{aligned}
\mathbf{Pr}(\Delta Car[i+1] = 1, \Delta y[i] = 0 | \Delta Car[i] = 0) &= \mathbf{Pr}(y[i] = \tilde{y}[i] = 1).\mathbf{Pr}(\Delta x[i] = 1) \\
&= \frac{1}{8}(1 + \theta_{y[i]})(1 - \theta_{x[i]}).
\end{aligned}$$

We can obtain the same values for the case $Car[i] = \tilde{Car}[i] = 1$.
Therefore, for the case $\Delta Car[i] = 0$,

$$\begin{aligned}
\mathbf{Pr}(\Delta Car[i+1] = 1 | \Delta Car[i] = 0) &= \frac{1}{4}(1 - \theta_{y[i]}) + \cdot\frac{1}{8}(1 + \theta_{y[i]})(1 - \theta_{x[i]}) \\
&= \frac{1}{8}(3 - \theta_{y[i]} - \theta_{x[i]} - \theta_{y[i]}\theta_{x[i]}).
\end{aligned}$$

$\Delta Car[i] = 1$: Without loss of generality we assume that $Car[i] = 1, \widetilde{Car}[i] = 0$. Now, if $x[i] = 1, \tilde{x}[i] = 0$, then whatever value $y[i]$ and $\tilde{y}[i]$ takes, $Car[i+1] \neq \tilde{Car}[i+1]$. The probability of this is $\mathbf{Pr}(x[i] = 1, \tilde{x}[i] = 0) = \frac{1}{4}\big[1 - \theta_{x[i]}\big]$. Secondly, if $x[i] = 0, \tilde{x}[i] = 1$, carry difference would occur if and only if $\Delta y[i] = 1$. Probability of this event is

$$\begin{aligned}\mathbf{Pr}(\Delta Car[i+1] = 1) &= \mathbf{Pr}(x[i] = 0, \tilde{x}[i] = 1) \cdot \mathbf{Pr}(\Delta y[i] = 1) \\ &= \frac{1}{4}(1 - \theta_{x[i]}) \cdot \frac{1}{2}(1 - \theta_{y[i]}).\end{aligned}$$

For $x[i] = \tilde{x}[i] = 0$, the carry difference occurs if $y[i] = 1$. Its probability is $\frac{1}{8}(1 + \theta_{x[i]})$. Similarly, if $x[i] = \tilde{x}[i] = 1$, the carry difference occurs if $\tilde{y}[i] = 0$. Probability is again $\frac{1}{8}(1 + \theta_{x[i]})$.
So, in this case, the total probability is the sum of these four cases, which is
$\mathbf{Pr}(\Delta Car[i+1] = 1 | \Delta Car[i] = 1) = \frac{1}{8}(5 - \theta_{x[i]} - \theta_{y[i]} + \theta_{x[i]}\theta_{y[i]})$.
Now, multiplying both the probabilities with the probability of their corresponding condition and then adding, we get:
$\mathbf{Pr}(\Delta Car[i+1] = 1)$
$= \frac{1}{2}(1 + \theta_{Car[i]}) \cdot \frac{1}{8}(3 - \theta_{y[i]} - \theta_{x[i]} - \theta_{y[i]}\theta_{x[i]}) + \frac{1}{2}(1 - \theta_{Car[i]}) \cdot \frac{1}{8}(5 - \theta_{x[i]} - \theta_{y[i]} + \theta_{x[i]}\theta_{y[i]})$
$= \frac{1}{16}\big[8 - 2\theta_{x[i]} - 2\theta_{y[i]} + \theta_{Car[i]}(-2 - 2\theta_{x[i]}\theta_{y[i]})\big]$
$= \frac{1}{8}\big[4 - \theta_{x[i]} - \theta_{y[i]} - \theta_{Car[i]}(1 + \theta_{x[i]}\theta_{y[i]})\big]$
$= \frac{1}{2} - \frac{1}{8}\big[\theta_{x[i]} + \theta_{y[i]} + \theta_{Car[i]}(1 + \theta_{x[i]}\theta_{y[i]})\big]$.
So, $\mathbf{Pr}(\Delta Car[i+1] = 0) = \frac{1}{2} + \frac{1}{8}\big[\theta_{x[i]} + \theta_{y[i]} + \theta_{Car[i]}(1 + \theta_{x[i]}\theta_{y[i]})\big]$. Therefore

$$\theta_{Car[i+1]} = \frac{1}{4}\big[\theta_{x[i]} + \theta_{y[i]} + \theta_{Car[i]}(1 + \theta_{x[i]}\theta_{y[i]})\big].$$

□

Theorem 5
Considering all the notations same as above, $\theta_{z[i]}$ can be computed as:

$$\theta_{z[i]} = \theta_{Car[i]} \cdot \theta_{x[i]} \cdot \theta_{y[i]}.$$

Proof. We know that $z[i] = Car[i] \oplus x[i] \oplus y[i]$.
Suppose $z_1[i] = x[i] \oplus y[i]$ and $\tilde{z}_1[i] = \tilde{x}[i] \oplus \tilde{y}[i]$.
Now $\Delta z_1[i] = 0$ iff $\Delta x[i] = \Delta y[i] = 0$ or 1.
So,

$$\begin{aligned}\theta_{z_1[i]} =& 2\big[\mathbf{Pr}(\Delta z_1[i] = 0) - \frac{1}{2}\big] \\ =& 2\big[\mathbf{Pr}(\Delta x[i] = \Delta y[i] = 0) + \mathbf{Pr}(\Delta x[i] = \Delta y[i]) = 1) - \frac{1}{2}\big] \\ =& 2\big[\frac{1}{4}(1 + \theta_{x[i]})(1 + \theta_{y[i]}) + \frac{1}{4}(1 - \theta_{x[i]})(1 - \theta_{y[i]}) - \frac{1}{2}\big] \\ =& \frac{1}{2}(2 + 2\theta_{x[i]}\theta_{y[i]}) - 1 \\ =& \theta_{x[i]}\theta_{y[i]}.\end{aligned}$$

Now, $\Delta z[i] = \Delta z_1[i] \oplus \Delta Car[i]$.
Hence, in a similar manner;
$\theta_{z[i]} = \theta_{Car[i]} \cdot \theta_{z_1[i]} = \theta_{Car[i]} \cdot \theta_{x[i]} \cdot \theta_{y[i]}$. □

Next we state a lemma on the linear properties of subtraction modulo 2^n.

Lemma 2
Let $z = x - y$ modulo 2^n where all are n-bit numbers and x, y are randomly chosen. Then for any $i \in [1, n-1] \cap \mathbb{Z}$,

1) $\mathbf{Cor}[z[i] \oplus x[i] \oplus y[i]] \oplus x[i-1]] = -2^{-1}$,
2) $\mathbf{Cor}[z[i] \oplus x[i] \oplus y[i]] \oplus y[i-1]] = 2^{-1}$.

Proof. For this we use Lemma 2 of [4], According to that, if $x[i-1] \neq y[i-1]$, then $z[i] = 1 \oplus x[i] \oplus y[i] \oplus x[i-1]$. Therefore, for $\frac{1}{2}$ possible pairs, $\mathbf{Cor}[z[i] \oplus x[i] \oplus y[i]] \oplus x[i-1]] = -1$. For remaining $\frac{1}{2}$ possible pairs, due to randomness $\mathbf{Cor}[z[i] \oplus x[i] \oplus y[i]] \oplus x[i-1]] = 0$. Therefore the final value would be the average of -1 and 0, i.e. -2^{-1}.
In a similar manner we can prove the other one. □

IV. Analysis of the difference propagation in the first round

In the attack of [4] the authors finally give a 5-rounds multi-bit distinguisher. These 5 rounds of the cipher they have decomposed into three parts:

$$E = E_2 \circ E_m \circ E_1$$

where E_1 consists of 1-st round, and E_m consists of next 2.5 rounds and E_2 consists of the remaining part. In the key recovery attack, the distinguisher is observed for 3.5 rounds $(E_m \circ E_1)$. This distinguisher is exploited in two different ways to attack 6-rounds and 7 rounds, respectively. In 6 rounds ChaCha, in the E_m part the distinguisher is extended to 1.5 more rounds by the help of linear cryptanalysis. Therefore, $E_2 \circ E_m \circ E_1$ constructs a 5 rounds distinguisher. Then a key recovery F is done in the 6th round. In the 7-rounds ChaCha the E_2 part is replaced by PNBs approach, where the 3.5 rounds distinguisher $(E_m \circ E_1)$ is used to construct a meet-in-the-middle attack.
Therefore our work focuses on the $E_m \circ E_1$ part, where the initial input difference is $(\Delta X^{(0)} = X^{(0)} \oplus \tilde{X}^{(0)})$ provided in the 6-th bit of one of the IV words. Since the structure of ChaCha is symmetric with respect to all the columns, we take the case for the IV word X_{13}.
In the first round E_1, the IV is chosen in such a way that the Hamming weight of $\Delta X^{(1)} (= X^{(1)} \oplus \tilde{X}^{(1)}$, which is the difference matrix after the first round) is minimum. In this case the differences are at positions $\{(1,2),(5,5),(5,29),(5,17),(5,9),(9,30),(9,22),(9,10),(13,30),(13,10)\}$ after the 1-st round, as already mentioned in [4].

A. Analysis of the availability of IVs for minimum Hamming weight

In the paper [4] it has been mentioned that for approximately 70% of the keys there exist IVs which give minimum number of differences (Hamming weight: 10) after the first round. Here we try to provide a theoretical explanation of why it is not possible to avail IVs for all keys.
There are 4 addition operations in a round. To minimize the number of differences, it is necessary that in each of these operations, the difference would not propagate to the next bit. This means, in the operation $x + y$, if the initial difference is at i-th bit of y, then in the sum also the difference would be in the i-th bit only and would not propagate to the $(i+1)$-th bit. However, in the second $(X_0'' = X_0' + X_4')$ operation, there is an initial difference at $X_4[2]$. Now, this difference would not propagate to the next bit only if the carry difference is 0. This depends on the values of X_0' and X_4' after the first half.

Theorem 6
If the last three bits of X_4 i.e, $X_4[2,0]$ are 111, there is no IV which can produce the minimum number of differences.

Proof. Since the last three bits of X_0 are 101, if the last three bits of X_4 are 111, after the addition the last three digits of X_0' become 100. Now, in the second half, whatever be the value of X_4, since the last two bits of X_0 are 0, no carry would be generated from these bits. Now, since $X_0'[2] = 1$ and $X_4'[2]$ contains the difference, therefore in the sum there must be a carry difference generated at this position which would propagate to the next bit. Therefore a minimum difference can not be produced.

□

This theorem has been explained in Figure 1. The square boxes are the bit positions which contain a difference. Since $X_0'[2] = 1$, the difference at $X_4'[2]$ propagates to $Car[3]$.

Theorem 7
If the last three bits of X_4 i.e, $X_4[2,0]$ are 011, there always exists an IV which can produce the minimum number of differences.

Proof. If the last three bits of X_4 are 011, after the first addition the last three bits of X_0' become 000. Since the last 3 bits are 0, in the second half whatever be the value of X_4, there is no carry produced in any of these bits. So the carry differences for X_0' and $\tilde{X}'_0$ are zero at the second bit. So IV would be available. □

This theorem has been explained in Figure 2. Since $X_0'[2] = 0$, the difference at $X_4'[2]$ is unable to propagate to $Car[3]$. So $Car[3]$ does not contain a difference.

For all the remaining possible values for four bits of X_4 we can not say for sure whether there exists an IV generating minimum difference, because in these cases the carry difference propagation also depends on the corresponding values of X_4'. X_4' is generated at the fourth step of the first half. So, it is difficult to put conditions on the initial matrix to control X_4'.

In Table I, we provide the percentage of weak keys available for each possible value for the last four bits of X_4.

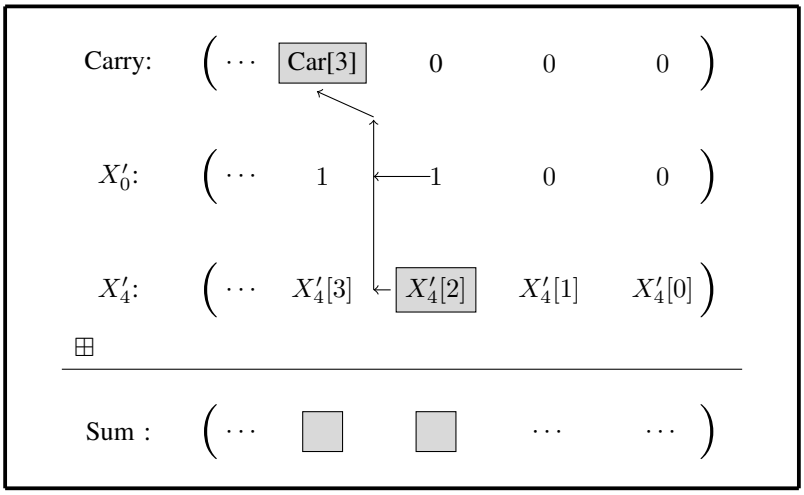

Fig. 1: Difference Propagation in the $X_0'' = X_0' + X_4'$ operation in the second half for the case of Theorem 6 $\Big($ $\square$ denotes the bits which contain a difference$\Big)$

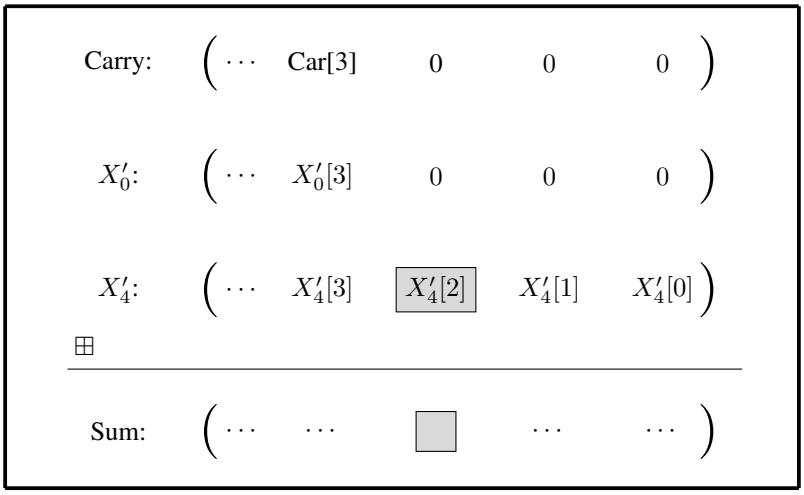

Fig. 2: Difference Propagation in the $X_0'' = X_0' + X_4'$ operation in the second half for the case of Theorem 7 $\Big($ $\square$ denotes the bits which contains difference$\Big)$

V. Analyzing the path for the correlation at $(2, 0)$

In this section we analyze the propagation of difference which results in the correlation at $(2, 0)$ after the third round. This section contains the results up to the first half of the third round. To derive our results, we use the theorems of Section III. The initial conditions of the results do not accurately fit into the criteria to apply the theorems. But the difference do make any impact on the result. In such scenario we use the relevant theorem, ignoring the negligible difference. For example, Theorem 2 requires difference at exactly 1 bit of y. But in our results we arrive at scenarios where there are differences in more than 1 bit, but the positions are far from each other, therefore have negligible influence on each other. In such scenario we apply the theorem.

TABLE I: Percentage of weak keys for different possible values of the last 3 bits of X_4

Values	0	1	2	3	4	5	6	7
Percentage of weak keys	64%	77%	86%	100%	87%	75%	63%	0%

A. Second round

Result 1

In the cell X_6 we have the correlation values for the following 4 bit positions.

I. $\mathbf{Cor}[\Delta X_6[3]] = \frac{1}{4}$,
II. $\mathbf{Cor}[\Delta X_6[4]] = \frac{1}{2}$,
III. $\mathbf{Cor}[\Delta X_6[0]] \approx 0.94$,
IV. $\mathbf{Cor}[\Delta X_6[20]] \approx 0.56$.

Proof. 1) In the first half of the second round, in the diagonal $(X_1, X_6, X_{11}, X_{12})$, the first operation is $X'_1 = X_1 + X_6$. Since initially there is a difference in $X_1[2]$, we can apply Theorem 2 here. Therefore, putting $z = X_1$ and $r = 2$ in Theorem 2, we have after the first half, $\mathbf{Cor}\big[eq(\Delta X'_1, [2+k, 1+k, k, \cdots, 2])\big] = 1 - \frac{1}{2^{k-1}}$.
Then, in the next operation $X'_{12} = (X_{12} \oplus X'_1) \lll 16$, these Cor values go to X_{12} and move 16 bits rightward. In the second half, in operation $X''_{12} = (X'_{12} \oplus X''_1) \lll 8$, these move to the right further by 8 bits. Therefore by total $16 + 8 = 24$ bits rotation, we have in the second round, $\mathbf{Cor}[eq(\Delta X''_{12}, [26+k, 25+k, k+24, \cdots, 26])] = 1 - \frac{1}{2^{k-1}}$.
Now, in the next step, the same addition $X''_{11} = X'_{11} + X''_{12}$ occurs. Here, since initially there is no influence of difference yet at the corresponding bits of X'_{11}, we can apply Theorem 3 to find the $\mathbf{Cor}[\Delta X''_{11}[26+k]]$. Applying Theorem 3 for $k = 2, 3$ respectively, we get $\mathbf{Cor}[\Delta X''_{11}[28]] = \frac{1}{4}$, $\mathbf{Cor}[\Delta X''_{11}[29]] = \frac{1}{2}$. Therefore, in the next XOR with X_6 and rotation by 7 bits, we get the results.

2) As mentioned in the first part, after the operation $X'_{12} = (X_{12} \oplus X'_1) \lll 16$ we have $\mathbf{Cor}[eq(\Delta X'_{12}, [18+k, 17+k, 16+k, \cdots, 18])] = 1 - \frac{1}{2^{k-1}}$. In the immediate next addition $X'_{11} = X_{11} + X'_{12}$, we can apply Theorem 3. So, we get in X'_{11}, $\mathbf{Cor}[\Delta X'_{11}[18+k]] = 1 - \frac{k+1}{2^k}$.
We are particularly interested in $k = 7$, which gives us the value 0.94. In the second half, X'_{11} is again updated to $X''_{11} = X'_{11} + X''_{12}$. However, by this time the respective bits of X'_{12} gets rotated during the update to X''_{12} and the present bits of X''_{12} at those positions does not have difference, which results no change in the correlation values of $X''_{11}[25]$ from $X'_{11}[25]$. This X''_{11} is XOR-ed with X'_6 and rotated by 7 bits. So, X'_6 receives the same value of correlation with rotation by 7 bits. So, $X''_6[25+7] = X''_6[0]$ (since we compute in modulo 2^{32}), which has $\mathbf{Cor}[\Delta X''_6[0])] = 0.94$.

3) In the addition $X'_1 = X_1 + X_6$, the difference of $(1, 2)$ propagates to the next bits. By Corollary 1 we have $\mathbf{Cor}[\Delta X'_1[4])] = \frac{1}{2}$ and $\mathbf{Cor}[\Delta X'_1[5])] = \frac{3}{4}$. In the second half, in the XOR with X'_{12} and rotation, we get $\mathbf{Cor}[\Delta X''_{12}[13])] = 0.75$ and $\mathbf{Cor}[\Delta X''_{12}[12])] = 0.5$. In the next operation $X''_{11} = X'_{11} + X''_{12}$, by Theorem 4 we find out that in this addition, $\theta_{Car[13]} = \frac{1}{4}\big[(1 + 0.5) + 1 \times (1 + 0.5)\big] = 0.75$. Therefore, by Theorem 5, in the updated X''_{11}, $\mathbf{Cor}[\Delta X''_{11}[13])] = 0.75 \times 1 \times 0.75 \approx 0.56$. This value comes to $X''_6[20]$ in the next XOR and rotation.

□

Result 2

At the end of the second round, we observe the following result in X_{14} : $\mathbf{Cor}[\Delta X_{14}[4]] \approx 0.91$.

Proof. At the end of the first round, there is a difference at $(9, 10)$. In the second round first half, in the operation $X'_9 = X_9 + X_{14}$ this difference propagates to the left side by Theorem 2. Therefore $\mathbf{Cor}[\Delta X'_9[10+k]] = 1 - \frac{1}{2^{k-1}}$
After that in the next operation $X'_4 = (X_4 \oplus X'_9) \lll 12$, this values comes to $X'_4[22+k]$'s. In particular, we look for $k = 4, 5, 6$, which gives: $\mathbf{Cor}[\Delta X'_4[26]] = 1 - \frac{1}{2^3} \approx 0.88$, $\mathbf{Cor}[\Delta X'_4[27]] = 1 - \frac{1}{2^4} \approx 0.94$ and $\mathbf{Cor}[\Delta X'_4[28]] = 1 - \frac{1}{2^5} \approx 0.97$.

In the second half of the second round the following operation takes place. $X''_3 = X'_3 + X'_4$. X'_3 does not have any difference at positions 27 and 26 up to 1.5 round, i.e $\mathbf{Cor}[\Delta X'_3[26]] = \mathbf{Cor}[\Delta X'_3[27]] = 1$. For X'_4 we use the results just obtained. So, using Theorem 4, at first we find

$$\begin{aligned}\theta_{car[27]} =& \frac{1}{4}\big[\theta_{X'_4[26]} + \theta_{X'_3[26]} + \theta_{Car[26]}[1 + \theta_{X'_3[26]}\theta_{X'_4[26]}]\big] \\ =& \frac{1}{4}\big[0.88 + 1 + 1 \times [1 + 0.88 \times 1]\big] \\ =& 0.94.\end{aligned}$$

Then,

$$\begin{aligned}\theta_{Car[28]} =& \frac{1}{4}[\theta_{X'_4[27]} + \theta_{X'_3[27]} + \theta_{Car[27]}[1 + \theta_{X'_3[27]}\theta_{X'_4[27]}]] \\ =& \frac{1}{4}[0.94 + 1 + 0.94 \times [1 + 0.94 \times 1]] \\ \approx& 0.94.\end{aligned}$$

After that, using Theorem 5, we have $\mathbf{Cor}[\Delta X''_3[28]] = 1 \times 0.97 \times 0.94 \approx 0.91$. In the next XOR with X'_{14}, this value comes to $X''_{14}[4]$ by rotation. □

The framework of this proof has been shown in Figure 3.

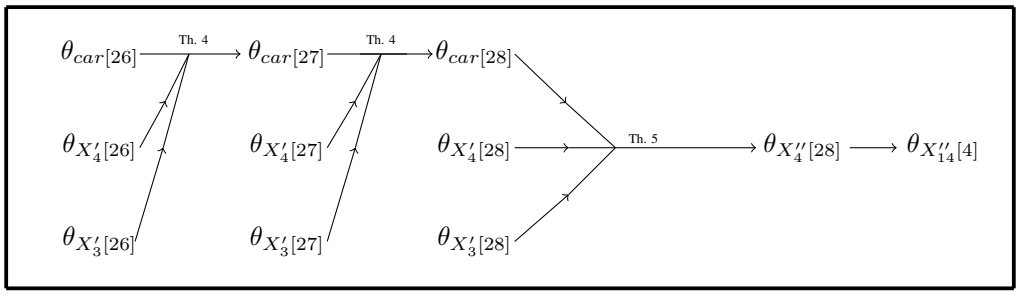

Fig. 3: Framework for the proof of Result 2

Result 3
We observe the following result in X_{10} : $\mathbf{Cor}[\Delta X_{10}[20]] \approx 0.47$.

Proof. In the first round $(5, 9)$ has a difference. In the first step of the second round $X'_0 = X_0 + X_5$, this difference comes to X'_0 and propagates by Theorem 2. In the next operation $X'_{15} = (X_{15} \oplus X'_0) \lll 16$, these values come to X'_{15}, being rotated by 16 bits. So, $\mathbf{Cor}[eq(\Delta X'_{15}, [25 + k, 24 + k, 23 + k, \cdots, 25])] = 1 - \frac{1}{2^{k-1}}$. In the next addition with X'_{10}, these come to X'_{10} using Theorem 3. In particular, for $k = 6$, we get $\mathbf{Cor}[\Delta X'_{10}[31]] = 1 - \frac{1+6}{2^6} \approx 0.9$. This comes to $X'_5[11]$ in the next XOR and rotation.
On the other hand, $\mathbf{Cor}[\Delta X'_0[16]] = 1 - \frac{1}{2^6} \approx 0.984$. This comes to $\mathbf{Cor}[\Delta X'_5[12]]$.
Again, $\mathbf{Cor}[\Delta X'_0[12]] = 0.75$ and $\mathbf{Cor}[\Delta X'_0[11]] = 0.5$ by the same reason. From these two, we get by applying Theorem 4 and Theorem 5 that after the second round, $\mathbf{Cor}[\Delta X''_0[12]] = 0.68$ and $\mathbf{Cor}[\Delta X''_0[11]] = 0.46$. Therefore, in the next XOR with X'_{15} and rotation by 8 bits, we get $\mathbf{Cor}[\Delta X''_{15}[20]] = 0.68, \mathbf{Cor}[\Delta X''_{15}[19]] = 0.46$.
On the other hand, $\mathbf{Cor}[\Delta X'_{10}[20]] = 0.97, \mathbf{Cor}[\Delta X'_{10}[19]] = 0.94$. After that the addition $X''_{10} = X'_{10} + X'_{15}$ occurs. Using Theorem 4, we can find for the propagation difference at the 20-th bit

$$\begin{aligned}\theta_{Car[20]} =& \frac{1}{4}[\theta_{X'_{10}[19]} + \theta_{X'_{15}[19]} + \theta_{Car[19]}(1 + \theta_{X'_{10}[19]}\theta_{X_{15}[19]})] \\ =& \frac{1}{4}[0.94 + 0.46 + (1 + 0.94 \times 0.46)] \\ =& \frac{2.83}{4} \approx 0.71.\end{aligned}$$

Then, applying Theorem 5, we get $\mathbf{Cor}[\Delta X''_{10}[20]] = 0.71 \times 0.68 \times 0.97 \approx 0.47$. □

B. First half of the third round

Result 4
After the first half of the third round, $\mathbf{Cor}[\Delta X_6[0]] \approx 0.08$.

Proof. In the first step of the third round, X_2 is updated by the operation $X'_2 = X_2 + X_6$. We have observed in 1 that $\mathbf{Cor}[\Delta X_6[3]] = \frac{1}{4}$ and $\mathbf{Cor}[\Delta X_6[4]] = \frac{1}{2}$. Since the corresponding bits of X_2 are not yet influenced by the difference, $\mathbf{Cor}[\Delta X_2[3]] = \mathbf{Cor}[\Delta X_2[4]] = 1$.
We apply Theorem 4 here and use $\theta_{Car[3]} = 1$. Therefore,

$$\begin{aligned}\theta_{Car[4]} =& \frac{1}{4}[\theta_{X_6[3]} + \theta_{X_2[3]} + \theta_{Car[3]}(1 + \theta_{X_6[3]}\theta_{X_2[3]})] \\ =& \frac{1}{4}[\frac{1}{4} + 1 + [1 + \frac{1}{4}.1]] \\ =& \frac{1}{4}[\frac{5}{4} + \frac{5}{4}] \\ =& \frac{5}{8}.\end{aligned}$$

Therefore, using Theorem 5 we have,

$$\begin{aligned}\mathbf{Cor}[\Delta X'_2[4]] =&\theta_{Car[4]} \cdot \theta_{X_6[4]} \cdot \theta_{X_2[4]}\\ =&\frac{5}{8} \times \frac{1}{2} \times 1\\ =&\frac{5}{16} \approx 0.31.\end{aligned}$$

Then a XOR operation between X'_2 and X_{14} and a rotation by 16 bits take place. So, $X'_2[4]$ and $X_{14}[4]$ generates $X'_{14}[20]$. Applying Theorem 5 with $\theta_{X'_2[4]} = 0.31$ and $\theta_{X_{14}[4]} = 0.91$, we have $\mathbf{Cor}[\Delta X'_{14}[20]] = 0.29$.
In the Addition operation between X_{10} and X'_{14}, applying Theorem 5 we get $\mathbf{Cor}[\Delta X'_{10}[20]] = 0.29 \times 0.47 \approx 0.14$. X'_{10} is XOR-ed with X_6 and rotated by 12 bits. So, by Theorem 5, $\mathbf{Cor}[\Delta X'_6[0]] = \theta_{X_6[20]} \cdot \theta_{X'_{10}[20]} = 0.56 \times 0.14 = 0.08$.

□

VI. Analyzing the path for the correlation at $(7, 0)$

Same as the last section, here we discuss the results up to 2.5 rounds which generates the correlation at $(7, 0)$ at the end of the third round.

A. Second round

Result 5
After the second round,

I. $\mathbf{Cor}[\Delta X_{11}[13]] \approx 0.5$,
II. $\mathbf{Cor}[\Delta X_{11}[25]] \approx 0.94$,
III. $\mathbf{Cor}[\Delta X_{11}[24]] \approx 0.88$.

Proof. After the first round $(1, 2)$ has a difference. Now, in the addition $X'_1 = X_1 + X_6$ in the second round, this difference propagates by 1. Therefore, for $r = 2$ in Corollary 1, $\mathbf{Cor}[eq(\Delta X'_1, [5, 4, 3, \cdots, 2])] = 1 - \frac{1}{2^{k-1}}$. Next, in the operation $X'_{12} = (X_{12} \oplus X_1) \lll 8$, this value comes to ΔX_{12} and gets rotated by 8 bits. Therefore, $\mathbf{Cor}[eq(\Delta X'_{12}, [10 + k, 9 + k, \cdots, 10])] = 1 - \frac{1}{2^{k-1}}$. In the next addition, $X'_{11} = X_{11} + X'_{12}$, using Theorem 3, we get that for $k = 3, r = 10$

$$\mathbf{Cor}[\Delta X_{11}[13]] = 1 - \frac{1+3}{2^3} = \frac{1}{2}.$$

For the second and third, we know that initially $(1, 2)$ contains a difference. In the first addition it propagates toward right by Theorem 2. In the update of X_{12}, all these get rotated by 16 bits. Then in the addition $X'_{11} = X_{11} + X'_{12}$, these come to X'_{11}. Here, by Theorem 4 we get the results for the 24-th and 25-th bit. The values remain same in the addition of the second half since the other component has correlation 1. □

Result 6
We have the following results for X_{15} at the end of the second round:

I. $\mathbf{Cor}[\Delta X_{15}[1]] \approx 0.77$,
II. $\mathbf{Cor}[\Delta X_{15}[29]] \approx 0.86$,
III. $\mathbf{Cor}[\Delta X_{15}[21]] \approx 0.86$,
IV. $\mathbf{Cor}[\Delta X_{15}[20]] \approx 0.74$.

Proof. $(5, 9)$ has a difference after the first round. In the second half it comes to $(5, 21)$ after rotation by 12 bits. Another difference at $(5, 29)$ comes to $(0, 29)$ in the first addition, then to $(15, 13)$ in the operation $X'_{15} = (X_{15} \oplus X'_0) \lll 16$. In the next addition and then XOR, it comes to $X'_5[25]$. So, after 1.5 rounds, both $\Delta X'_5[25]$ and $\Delta X'_5[21]$ are 0. In the next addition $X''_0 = X'_5 + X'_0$, these differences come to the same bits of X''_0. The difference at $X''_0[21]$ propagates to 25-th bit, for which we get using Corollary 1, $\mathbf{Cor}[\Delta X''_0[25]] \approx \frac{7}{8}$, which becomes negative because of the difference already at the 25-th bit.
Similarly, in the first half addition $X'_0 = X_5 + X_0$, the initial difference at $(5, 5)$ and $(5, 9)$, together results in $\mathbf{Cor}[\Delta X'_0[9]] \approx -\frac{3}{4}$, in the first half. Next XOR with X'_{15} and rotation by 16 bits result in $\mathbf{Cor}[\Delta X'_{15}[25]] \approx -\frac{7}{8}$. After this, in the operation $X''_{15} = (X'_{15} \oplus X''_0) \lll 8$, using Theorem 5 we get

$$\mathbf{Cor}[\Delta X''_{15}[1]] = \mathbf{Cor}[\Delta X'_{15}[25]] \cdot \mathbf{Cor}[\Delta X''_0[25]] \approx \frac{49}{64} \approx 0.77.$$

In the first addition of the second round, i.e, $X'_0 = X_0 + X_5$, the initial difference at $(5, 17)$ propagates by Corollary 1. Therefore, here for $r = 17$ $\mathbf{Cor}[\Delta X'_0, (17 + k, 16 + k, \cdots, 17)] = 1 - \frac{1}{2^{k-1}}$. In the second half, in the operation $X''_{15} =$

$(X'_{15} \oplus X''_0) \lll 8$, these values come to X'', getting rotated by 8 bits. So, $\mathbf{Cor}[\Delta X''_{15}, (25+k, 24+k, \cdots, 25)] = 1 - \frac{1}{2^{k-1}}$. For $k = 4$, we achieve the result.
In exactly similar way, the difference at $(5, 9)$ results in the same value for $\mathbf{Cor}[\Delta X''_{15}[21]]$ and 0.74 for $\mathbf{Cor}[\Delta X''_{15}[20]]$. Note here that the difference between $(15, 29)$ and $(15, 21)$ is 8 bits, which is same as the difference between $(5, 17)$ and $(5, 9)$. □

Result 7
For X_3, the following values are observed after the second round:

I. $\mathbf{Cor}[\Delta X_3[29]] \approx 0.98$,
II. $\mathbf{Cor}[\Delta X_3[21]] \approx 0.97$,
III. $\mathbf{Cor}[\Delta X_3[20]] \approx 0.94$.

Proof. $(9, 10)$ has a difference after the first round. In the operation $X'_9 = (X_9 + X'_{14})$ this difference stays at the same position. Then, in the operation $X'_4 = (X_4 \oplus X'_9) \lll 12$, this difference comes to $X'_4[22]$. Then in the addition $X''_3 = X'_3 + X'_4$ it comes to $X''_3[22]$ and propagates. By Corollary 1, for $r = 22, k = 7$, we have $\mathbf{Cor}[\Delta X''_3[29]] = 1 - \frac{1}{2^{7-1}} \approx 0.98$. In the second one, a similar reason is valid. □

Result 8
In the word X_7, we have the following values:

I. $\mathbf{Cor}[\Delta X_7[17]] \approx 0.83$,
II. $\mathbf{Cor}[\Delta X_7[16]] \approx 0.67$,
III. $\mathbf{Cor}[\Delta X_7[13]] \approx -0.56$,
IV. $\mathbf{Cor}[\Delta X_7[29]] \approx -0.92$.

Proof. The initial difference at $(13, 10)$ and $(13, 30)$ after the first round shifts to $(13, 14)$ and $(13, 26)$ respectively when X_{13} is updated in the first half of the second round. Then it gets added with X_8. Therefore, the difference at $(13, 14)$ comes to $X'_8[14]$ and propagates by Corollary 1. Then it is XOR-ed with X_7 and rotated by 12 bits. So, the differences at $X'_8[14+k]$ shift to $X'_7[26+k]$. Then, it the second half it is added with X'_2, which gives the values of $\mathbf{Cor}[\Delta X''_2[26+k]]$ by Theorem 2. Then, in the update of X'_{13}, these values come to X''_{13} by getting rotated by 8 bits. Therefore, $\mathbf{Cor}[\Delta X''_{13}[2+k]]$ can be given by Theorem 3. However, the difference at $(13, 2)$ is neutralised by the initial difference at $(13, 26)$ that we already mentioned. For the rest, the values follow the theorem. We find $\mathbf{Cor}[\Delta X''_{13}[8]], \mathbf{Cor}[\Delta X''_{13}[9]], \mathbf{Cor}[\Delta X''_{13}[10]]$ using this. Then we use them to find $\mathbf{Cor}[\Delta X''_8[9]]$ and $\mathbf{Cor}[\Delta X''_8[10]]$ as 0.9 and 0.94 respectively, which is the next addition operation $X''_8 = X'_8 + X''_{13}$.
On the other hand, the same difference at $(13, 26)$ in the first half, when added with X_8, produces a difference in $X'_8[26]$ which propagates by Corollary 1. These values of $\mathbf{Cor}[\Delta X'_8[26+k]]$ come to $X'_7[6+k]$ in the next operation $X'_7 = (X_7 \oplus X'_8) \lll 12$. In particular, we use the values $\mathbf{Cor}[\Delta X'_7[9]] \approx 0.88$ and $\mathbf{Cor}[\Delta X'_7[10]] \approx 0.74$ along with the previously found values $\mathbf{Cor}[\Delta X''_8[9]]$ and $\mathbf{Cor}[\Delta X''_8[10]]$ to get the results by Theorem 5. □

B. First half of the third round

Here, we obtain the correlation values for some particular bits of three words from the column containing X_7, i.e., X_{11}, X_7 and X_{15}.

Result 9
After 2.5 rounds, we have the following result for X_{15} :

I. $\mathbf{Cor}[\Delta X_{15}[17]] \approx 0.67$,
II. $\mathbf{Cor}[\Delta X_{15}[25]] \approx 0.9$.

Proof. In the third round we have the operation $X'_3 = X_3 + X_7$. At the end of the second round, $\mathbf{Cor}[\Delta X_7[1]] = \frac{7}{8}$, which comes to X'_3 in this addition. So, $\mathbf{Cor}[\Delta X'_3[1]] = \frac{7}{8}$. In the next XOR, using Theorem 5 we get

$$\mathbf{Cor}[\Delta X'_{15}[17]] = \mathbf{Cor}[\Delta X_{15}[1]] \cdot \mathbf{Cor}[\Delta X'_3[1]] = 0.77 \times \frac{7}{8} \approx 0.67.$$

Similarly we can prove the other one also. □

Result 10
For X_{11} we have the correlation values for the two following bits as follow:

I. $\mathbf{Cor}[\Delta X_{11}[5]] \approx 0.71$,

II. $\mathbf{Cor}[\Delta X_{11}[25]] \approx 0.66$.

Proof. In the first half of the third round the following XOR and rotation take place.

$$X'_{15} = (X_{15} \oplus X'_3) \lll 16$$

Here, from Result 7 and 6, using Theorem 5 we can compute that

$$\begin{aligned}\mathbf{Cor}[\Delta X'_{15}[5]] =& \mathbf{Cor}[\Delta X'_3[21]] \cdot \mathbf{Cor}[\Delta X_{15}[21]] \\ =& 0.97 \times 0.86 \approx 0.83.\end{aligned}$$

and similarly $\mathbf{Cor}[\Delta X'_{15}[4]] = 0.94 \times 0.74 \approx 0.70$. After that, an addition between X_{11} and X'_{15} takes place. Since the corresponding bits of X_{11} have negligible influence of difference yet, so using these results, by Theorem 4 we get

$$\begin{aligned}\theta_{car[5]} =& \frac{1}{4}[1 + 0.70 + 1 \times (1 + 1 \times 0.70)] \\ =& \frac{1}{4}[1.70 + 1.70] = \frac{1}{4} \times 3.40 \approx 0.85.\end{aligned}$$

Therefore, using Theorem 5 we get: $\mathbf{Cor}[\Delta X'_{11}[5]] = 0.85 \times 1 \times 0.83 \approx 0.71$.

Then, in the same operation $X'_{11} = X_{11} + X'_{15}$. We use the fact that after the second round, $\mathbf{Cor}[\Delta X_{11}[25]] \approx 0.94$, $\mathbf{Cor}[\Delta X_{11}[24]] \approx 0.88$ (Result 5) and $\mathbf{Cor}[\Delta X_{15}[25]] \approx 0.88$ (Result 9) after 2.5 rounds. Therefore, we apply first Theorem 4 on the 24-th bit and then Theorem 5 to get the result. □

Result 11
After 2.5 rounds, $\mathbf{Cor}[\Delta X_7[25]] \approx 0.21$.

Proof. After the second round, $\mathbf{Cor}[\Delta X_3[29]] \approx 0.98$ (Result 7). Now, after its addition with X_7, by Theorem 5 we have $\mathbf{Cor}[\Delta X'_3[29]] = 0.98 \times (-0.92) = -0.9$ (Result 8). Then, in the XOR with X_{15} and rotation by 16 bits, we get $\mathbf{Cor}[\Delta X'_{15}[13]] = 0.82 \times (-0.9) \approx -0.74$ (using Lemma 1). Now, we know from Result 5 that $\mathbf{Cor}[\Delta X_{11}[13]] = 0.5$. So, when X'_{15} is added with X_{11}, we have $\mathbf{Cor}[\Delta X'_{11}[13]] = 0.5 \times (-0.74) = -0.37$. After this, a XOR between X'_{11} and X_7 takes place and it is rotated by 12 bits. From Result 8 we know that $\mathbf{Cor}[\Delta X_7[13]] = -0.56$. Therefore, in X'_7, we have $\mathbf{Cor}[\Delta X'_7[25]] = (-0.56) \times (-0.37) \approx 0.21$. □

VII. Analysis of the 3-rd and 3.5-th round

In the 2.5 round, in the sum $X''_3 = X'_3 + X'_7$, $\mathbf{Cor}[\Delta X'_3[17]]$ can be found to be 0.64 using Theorem 4 and Theorem 5. Similarly it can be found that after 2.5 round $\mathbf{Cor}[\Delta X'_7[17]] = 0.5$. Therefore, in the sum $X''_3 = X'_3 + X'_7$ at the second half of the third round, by Theorem 5 we get that $\mathbf{Cor}[\Delta X''_3[17]] = 0.64 \times 0.5 = 0.32$. Then, it is XOR-ed with X'_{15}, which gives that

$$\mathbf{Cor}[\Delta X''_{15}[25]] = \mathbf{Cor}[\Delta X''_3[17]] \cdot \mathbf{Cor}[\Delta X'_{15}[25]] = 0.32 \times 0.67 \approx 0.21 \text{ (Result 9)}.$$

Then, in the addition with X'_{11}, we have

$$\mathbf{Cor}[\Delta X'_{11}[25]] = \mathbf{Cor}[\Delta X''_{15}[25]] \cdot \mathbf{Cor}[\Delta X'_{11}[25]] = 0.66 \times 0.21 = 0.14.$$

In the next XOR with X'_7 and rotation by 7 bits, from Result 11 we get by Theorem 5, $\mathbf{Cor}[\Delta X''_7[0]] = 0.2 \times 0.14 = 0.03$.

Similarly, in the second half of 3rd round, X_2 is updated by addition with X_6. We know from 4, $\mathbf{Cor}[\Delta X_6[0]] = 0.08$. This value comes to $X_2[0]$ after this addition since $X_2[0]$ was negligibly influenced by the difference before this. Now, in the fourth round first half, we have $X'_2 = X_2 + X_7$. Since we are interested in the 0-th bit, we don't worry about carry difference. So,

$$\mathbf{Cor}[\Delta X'_2[0]] = \mathbf{Cor}[\Delta X_2[0]] \cdot \mathbf{Cor}[\Delta X_7[0]] = 0.08 \times 0.03 \approx 0.0024 \approx 2^{-8.7}.$$

According to [4], the experimentally observed value is $2^{-8.3}$. In other words, among all possible states, for approximately $\frac{1}{2}[1 + 2^{-8.3}] = 0.5016$ fraction of states gives equal value at position $(2, 0)$ for X and $\tilde{X}$ after 3.5 rounds. According to our theoretical analysis this fraction is 0.5012, which matches up to 3 decimal places with the experiment.

A. Extension to multi-bit distinguisher of the 5-th round

In [4], a linear trail between the single bit distinguisher of 3.5 rounds and two different multi-bit distinguishers of the 5-th round has been used. This linear relation has correlation 2^{-1}. Here we explain this linear trail in 3 steps.

1) 3.5 round to 4-th round

$X_2[0]$ is updated as $X_2''[0] = X_2'[0] \oplus X_7'[0]$ (since least significant bit).
Again, from $X_7'' = (X_7' \oplus X_8'') \lll 7$ we have

$$X_7''[7] = X_7'[0] \oplus X_8''[0]$$
$$i.e, X_7'[0] = X_8''[0] \oplus X_7''[7].$$

Therefore, $X_2'[0] = X_2''[0] \oplus X_8''[0] \oplus X_7''[7]$.

2) 4 round to 4.5-th round

In this column-round each of the three bits $X_2[0]$, $X_8[0]$ and $X_7[7]$ of 4-th round in terms of bits of 4.5 round we now express in terms of bits of 4.5 round.
From, $X_2' = X_2 + X_7$ we get in similar manner as before:

$$\begin{aligned} X_2[0] &= X_6[0] \oplus X_2'[0] \\ &= X_6'[12] \oplus X_{10}'[0] \oplus X_2'[0] \text{ (since } X_6' = (X_6 \oplus X_{10}) \lll 12) \end{aligned}$$

From, $X_7' = (X_7 \oplus X_{11}') \lll 12$ we get $X_7[7] = X_7'[19] \oplus X_{11}'[7]$.
From, $X_8' = X_8 + X_{12}'$ we get $X_8[0] = X_8'[0] \oplus X_{12}'[0]$.
Therefore, we have

$$X_2[0] \oplus X_8[0] \oplus X_7[7] = X_2'[0] \oplus X_6'[12] \oplus X_{10}'[0] \oplus X_8'[0] \oplus X_{12}'[0] \oplus X_7'[19] \oplus X_{11}'[7].$$

3) 4.5 round to 5-th round

In a similar manner as before,

$$\begin{aligned} X_2'[0] &= X_2''[0] \oplus X_6'[0] \\ &= X_2''[0] \oplus X_6''[7] \oplus X_{10}''[0]. \end{aligned}$$

From, $X_6'' = (X_6' \oplus X_{10}'') \lll 7$ we get $X_6'[12] = X_6'[19] \oplus X_{10}'[12]$.
From, $X_{10}'' = X_{10}' + X_{14}''$ we get $X_{10}'[0] = X_{10}''[0] \oplus X_{14}''[0]$.
From, $X_7'' = X_7' + X_{11}''$ we get $X_7'[19] = X_7''[26] \oplus X_{11}'[19]$.
From, $X_8'' = X_8' + X_{12}''$ we get $X_8'[0] = X_8''[0] \oplus X_{12}''[0]$.
From, $X_{12} = (X_{12} \oplus X_0) \lll 8$ we get $X_{12}[0] = X_{12}[8] \oplus X_0[0]$.
Now only for the expression $X_{11}'[7]$, we can't directly write it as linear sum of 5-th round elements since $X_{11}'' = X_{15}'' + X_{11}'$ involves carry from the previous term.
Here we use Lemma 2. We have, $X_{11}' = X_{11}'' - X_{15}''$. From Lemma 2 we have

$$\mathbf{Cor}(X_{11}'[7] \oplus X_{11}''[7] \oplus X_{15}''[7] \oplus X_{11}''[6]) = -2^{-1}$$
$$\mathbf{Cor}(X_{11}'[7], X_{11}''[7] \oplus X_{15}''[7] \oplus X_{15}''[6]) = 2^{-1}$$

We denote ϕ, ϕ', ϕ'' as follows:

- $\phi' = X_2'[0] \oplus X_6'[12] \oplus X_{10}'[0] \oplus X_8'[0] \oplus X_{12}'[0] \oplus X_7'[19] \oplus X_{11}'[7]$,
- $\phi_1'' = X_2''[0] \oplus X_6''[7] \oplus X_{10}''[0] \oplus X_6'[19] \oplus X_{10}'[12] \oplus X_{10}''[0] \oplus X_{14}''[0] \oplus X_7''[26] \oplus X_{11}'[19] \oplus X_8''[0] \oplus X_{12}''[0] \oplus X_{12}[8] \oplus X_0[0] \oplus X_{11}''[7] \oplus X_{15}''[7] \oplus X_{11}''[6]$,
- $\phi_2'' = X_2''[0] \oplus X_6''[7] \oplus X_{10}''[0] \oplus X_6'[19] \oplus X_{10}'[12] \oplus X_{10}''[0] \oplus X_{14}''[0] \oplus X_7''[26] \oplus X_{11}'[19] \oplus X_8''[0] \oplus X_{12}''[0] \oplus X_{12}[8] \oplus X_0[0] \oplus X_{11}''[7] \oplus X_{15}''[7] \oplus X_{15}''[6]$.

Thus

$$\mathbf{Cor}(\phi' \oplus \phi_i'') = \begin{cases} -2^{-1} & \text{if } i = 1 \\ 2^{-1} & \text{if } i = 2. \end{cases}$$

Therefore,

$$\mathbf{Cor}(X_2^{3.5}[0] \oplus \phi_i'') = \begin{cases} -2^{-1} & \text{if } i = 1 \\ 2^{-1} & \text{if } i = 2. \end{cases}$$

This linear relation between single bit of 3.5-th round and multi-bit of 5-th round, combined with the distinguisher of 3.5 round, gives a distinguisher of the 5-th round.

VIII. Revisiting the attack of Eurocrypt 2021 paper

At Eurocrypt 2021 Coutinho et al. [8] improved the key recovery attack for 7 round ChaCha with complexity $2^{228.51}$ and also gave a distinguisher attack for 7 round ChaCha with complexity 2^{218}. For key recovery attack they used the probabilistic neutral bit technique which is an old idea in [1]. They found a 3.5 rounds differential distinguisher at $(5, 0)$ by experimental approach which they have used for both key recovery and distinguisher attack. They extended the 3.5 rounds differential distinguisher to 7 rounds using linear approximation and they unveiled the reason of linear approximation by both theoretical and computational results. In their theoretical proofs of the linear approximations they used two probabilistic equations and Piling up lemma to approximate the nonlinear operations. For more details of their theoretical explanation of linear approximation we refer to [8].

The IV's are chosen in such a way that the Hamming weight of the output difference is minimized. They have chosen the IV in the third column of the state matrix and the input difference was given at $(14, 6)$. This input difference influences the following bit positions $(2, 2), (6, 5), (6, 29), (6, 17), (6, 9), (10, 30), (10, 22), (10, 10), (14, 13), (14, 10)$ on average with probability 2^{-5} after the first round which has been shown in [4].

Distinguishing Attack Procedure: Now we discuss how to compute differential-linear correlation and corresponding distinguisher complexity. Suppose the differential state matrix is denoted by $\Delta X^{(r)} = X^{(r)} \oplus \widetilde{X}^{(r)}$, the differential of a word as $\Delta X_i^{(r)} = X_i^{(r)} \oplus \widetilde{X}_i^{(r)}$ and the differential of j-th bit of i-th word as $\Delta X_i^{(r)}[j] = X_i^{(r)}[j] \oplus \widetilde{X}_i^{(r)}[j]$. Consider the state matrix $X^{(r)}$ and $\widetilde{X}^{(r)}$ after r rounds and suppose the linear combinations of bits of $X^{(r)}$ and $\widetilde{X}^{(r)}$ is denoted by $\sigma = \bigoplus_{i,j} X_i^{(r)}[j]$ and $\sigma' = \bigoplus_{i,j} \widetilde{X}_i^{(r)}[j]$ respectively. Then $\Delta\sigma = \bigoplus_{i,j} \Delta X_i^{(r)}[j]$ denotes the linear combination of differentials. Now

$$\mathbf{Pr}(\Delta\sigma = 0 | \Delta X^{(0)}) = \frac{1}{2}(1 + \epsilon)$$

where ϵ is the differential correlation.

Finally, it is possible to extend the differential distinguisher to few more rounds $(R > r)$ using linear cryptanalysis by finding relations between initial state matrix and state matrix after R rounds. For that we take the linear combinations of bits after $R(> r)$ rounds which are denoted as $\rho = \bigoplus_{i,j} X_i^{(R)}[j]$ and $\rho' = \bigoplus_{i,j} \widetilde{X}_i^{(R)}[j]$. Therefore, $\Delta\rho = \Delta \bigoplus_{i,j} X_i^{(R)}[j]$, which is computed similarly as above. The linear correlation is denoted by ϵ_l and is defined by $\mathbf{Pr}(\sigma = \rho) = \frac{1}{2}(1 + \epsilon_l)$. Our target is to find the differential-linear correlation ϵ^* such that $\mathbf{Pr}(\Delta\rho = 0 | \Delta X^{(0)}) = \frac{1}{2}(1 + \epsilon^*)$. Therefore, we can compute the ϵ^* using the following probabilistic relations (for simplification $\Delta X^{(0)}$ is omitted). Now,

$$\mathbf{Pr}(\Delta\sigma = \Delta\rho) = \mathbf{Pr}(\sigma = \rho) \cdot \mathbf{Pr}(\sigma' = \rho') + \mathbf{Pr}(\sigma = \bar{\rho}) \cdot \mathbf{Pr}(\sigma' = \bar{\rho'}) = \frac{1}{2}(1 + \epsilon_l^2),$$

where $\bar{\rho} = \rho \oplus 1$ and others are similarly follows. Then,

$$\mathbf{Pr}(\Delta\rho = 0) = \mathbf{Pr}(\Delta\sigma = 0) \cdot \mathbf{Pr}(\Delta\sigma = \Delta\rho) + \mathbf{Pr}(\Delta\sigma = 1) \cdot \mathbf{Pr}\left(\Delta\sigma = \overline{\Delta\rho}\right) = \frac{1}{2}(1 + \epsilon\epsilon_l^2).$$

Therefore, the obtained differential-linear correlation is $\epsilon^* = \epsilon\epsilon_l^2$ and the corresponding distinguisher complexity will be $\mathcal{O}(\frac{1}{\epsilon^2\epsilon_l^4})$. Generally we require $\mathcal{O}\left(\frac{1}{pq^2}\right)$ random samples when we like to distinguish between two events, one with probability p and the other with probability $p(1 + q)$, where q is small.

At Eurocrypt 2021 paper [8] the authors have presented several 3.5 rounds differential correlations of ChaCha for single output bit and these are listed in Table 3 of [8]. For this 3.5 rounds differential distinguisher they have used the idea that after one round the Hamming weight of the differential is minimized. They have given the input difference at $(14, 6)$ and experimentally obtained the differential correlation for the output difference at $(5, 0)$ after 3.5 rounds. Then the linear correlation has been found out between the bit $(5, 0)$ after 3.5 rounds and some bits after 6 rounds and 7 rounds respectively to give better differential-linear distingushers of 6 rounds and 7 rounds ChaCha than existing works. They have found out the distinguisher complexity using the above discussed method and the corresponding complexities for 6 rounds is 2^{77} and for 7 rounds is 2^{218}.

Key recovery attack procedure: Till now all the key recovery attacks against reduced round ChaCha have been presented using the probabilistic neutral bits (PNBs) approach which was introduced in [1]. Now we describe a glimpse of PNBs on key recovery attack albeit this is a fully experimental approach. We give an input difference at any desired bit $X_i[j]$ of the initial state matrix $X^{(0)}$ and obtain the new matrix $\widetilde{X}^{(0)}$. Our target is to obtain some correlation of output difference at some particular bit or combination of bits of the output matrix at some r-th round. We can compute $\mathbf{Pr}(\Delta\sigma = 0 | \Delta X_i[j]^{(0)} = 1)$ and suppose this value is $\frac{1}{2}(1 + \epsilon)$, where ϵ is the differential correlation of the output difference. We can also find out the differential correlation in backward direction from final state matrix of ChaCha as the rounds of ChaCha are reversible.

Notion of PNB: The idea of PNB was introduced in [1]. Later on this idea was reviewed by Maitra et al. [14] to come up with an improved attack. At first, we give a brief idea of Probabilistic Neutral Bits or PNB as given in [1], [14].

The main aim of this idea is to reduce the complexity of searching 256 bits of the unknown key. We try to partition the set of key bits into two parts:

1) **Significant Key bits:** key bits which have high influence on the output.
2) **Non-significant Key bits:** key bits which have low influence on the output.

To be more precise, we find a set of key bits such that, if the values of the key bits of this set is changed arbitrarily, the probability that the output will change too, is lower than usual. These key bits are considered to have low influence on the output (Non-significant key bits). If we can find a set of such key bits, we try to find the values of the remaining key bits, i.e, the significant key bits, by guessing randomly and considering a distinguisher to identify the correct set of values. After finding the significant bit values, we can find the values of non-significant bits by similar guessing and identifying. The advantage of this idea is that, since the number of significant key bits is much less than the total size of the key (256), the maximum number of guesses required is significantly less than 2^{256}.

To identify PNBs, we consider a predetermined threshold probability $\frac{1}{2}(1+\gamma)$. So, the whole set of key bits are divided into two sets, PNBs and non-PNBs. We suppose the size of these sets are m and n respectively ($m+n=256$).

Actual attack after PNB construction: Now, in main attack, attacker's aim is to find the values of the non-PNBs, without knowing the correct values of PNBs. Since changing PNBs affect the output with low probability, we take a random value for each PNB and set it to that fixed value. Instead of an exhaustive search over all possible 2^{256} values for the key bits, the concept of PNB helps to reduce the complexity of search. If the size of PNB set is m, then the number of non-PNBs is $n = 256 - m$.

Complexity Estimation: Here we briefly repeat the estimation provided by [1] for the reader's convenience. We have 2^n possible sequences of random values for the n non-PNBs. Out of them, only 1 sequence is correct and remaining $2^n - 1$ sequences are incorrect. In our hypothesis testing, we consider the null hypothesis H_0 as: The chosen sequence is incorrect. So, $2^n - 1$ sequences satisfy the null hypothesis and only 1 sequence satisfies the alternative hypothesis H_1 (chosen sequence is correct).

Two possible errors can occur in this attack:

1) **Error of Non-Detection:** The chosen sequence A is correct, i.e, $A \in H_1$, but it can't be detected. The probability of this error is P_{nd}.
2) **False Alarm Error:** The chosen sequence A is incorrect, i.e, $A \in H_0$, but it gives a significant bias. As a result, wrong sequence is accepted. The probability of this event is P_{fa}.

Now, to achieve a bound on these probabilities, authors [1] used a result given by Neyman- Pearson decision theory. According to this result, the number of samples is

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

where ϵ is the correlation in forward direction and ϵ_a is the correlation in the backward direction, where all PNB bits are fixed to zero and non-PNB bits are fixed to the correct ones. These samples can be used to achieve the bound of $P_{nd} = 1.3 \times 10^{-3}$ and the bound of P_{fa} by $2^{-\alpha}$. Based on these values, the time complexity can be given by

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

At Eurocrypt paper [8] the authors have reported 3.5 rounds differential correlation which they have found out in experimental approach. In the key recovery attack they extended this 3.5 rounds differential distinguisher at $(5, 0)$ to 4 rounds with linear correlation one and the corresponding linear relation is $X_5^{3.5}[0] = X_5^4[7] \oplus X_{10}^4[0]$ which directly follows from the $\mathcal{QR}$ operations. Also, their reported 4 rounds differential-linear correlation is $\epsilon = 0.0000002489$ which is the correlation in the forward direction. Also, using the threshold $\gamma = 0.35$ they have provided 108 PNBs and they have found out the correlation in the backward direction which is $\epsilon_a = 0.000169$. Also, they have written that the parameter α the attacker can choose. Using these values they have computed $N = 2^{75.51}$ and the corresponding time complexity is $2^{223.51}$. As in [4], they have repeated this attack 2^5 times on average. Thus, the final attack has data complexity of $2^{80.51}$ and time complexity $2^{228.51}$.
We have checked using their obtained values of ϵ and ϵ_a that the value of N and time complexity will be optimal when $\alpha = 37.56$. So, the actual data complexity of their attack will be $2^{75.64+5} = 2^{80.64}$. In this case the time complexity will be $2^{223.68}$. So, the final time complexity will be $2^{228.68}$.

A. Theoretical Interpretation and Experimental Results

We were trying to analyse theoretically the differential distinguisher of Eurocrypt 2021 paper [8] for 3.5 rounds ChaCha which they found out by fully experimental approach. At that time we have found out some mismatch between our theoretical results and their experimental results. Theoretical demonstration of the correlation for the bit position $(0, 0)$ has been presented in the following.

Output Difference	$\lvert\epsilon\rvert$	
	Eurocrypt 2021 [8]	Our
$\Delta X_0^{3.5}[0]$	0.000307	0.0000328273
$\Delta X_1^{3.5}[0]$	0.000124	0.0000017375
$\Delta X_{12}^{3.5}[0]$	0.000017	0.0000052242
$\Delta X_{13}^{3.5}[0]$	0.000016	0.0000064889
$\Delta X_5^{3.5}[0]$	0.0000002489	0.0000000568

TABLE II: Comparison of experimentally found 3.5 rounds differential correlation

As the number of rounds increases, the values of the correlation decreases. So, to find or verify a small correlation in higher round, we have to perform the simulation for high margin. Therefore, we also provide a mathematical explanation in support of our claim where we express the 3.5 rounds bit in form of 3 rounds bits. We try to provide a mathematical explanation which would help the reader to understand why the claimed bias at Eurocrypt 2021 paper [8] of the position $(0, 0)$ is not correct. In the update function of $(0, 0)$ we have

$$X_0^{(3.5)}[0] = X_0^{(3)}[0] + X_5^{(3)}[0].$$

Now, in such scenario, we generally use Theorem 5 to find the correlation of $X_0^{(3.5)}$ from the known correlations of $X_0^{(3)}[0]$ and $X_5^{(3)}[0]$. But Theorem 5 is based on the assumption of the independence of the correlations of the two i.e, **Cor**$[\Delta X_0^{(3)}[0]]$ given $\Delta X_5^{(3)}[0] = 0$ is same as **Cor**$[\Delta X_0^{(3)}[0]]$ given $\Delta X_5^{(3)}[0] = 1$. For high correlation values this assumption of independence does not deviate the theoretical result much from the experiment. But here since the correlation is very small, for an accurate computation we don't take the assumption of independence. Rather we deal with the two conditions $\Delta X_5^{(3)}[0] = 1$ and $\Delta X_5^{(3)}[0] = 0$ separately. We have, given $\Delta X_5^{(3)}[0] = 0$, **Cor**$[X_0^{(3)}[0]] = 0.006304$ and given $\Delta X_5^{(3)}[0] = 1$, **Cor**$[X_0^{(3)}[0]] = 0.006262$. We take the probability approach to compute the correlation of $\Delta X_0^{(3.5)}[0]$,

$$\begin{aligned}
\mathbf{Pr}(\Delta X_0^{(3.5)}[0] = 0) &= \mathbf{Pr}(\Delta X_0^{(3)}[0] \oplus \Delta X_5^{(3)}[0] = 0) \\
&= \mathbf{Pr}(\Delta X_0^{(3)}[0] = \Delta X_5^{(3)}[0] = 0) + \mathbf{Pr}(\Delta X_0^{(3)}[0] = \Delta X_5^{(3)}[0] = 1) \\
&= \mathbf{Pr}(\Delta X_5^{(3)}[0] = 0) \cdot \mathbf{Pr}(\Delta_0^{(3)}[0] = 0 \mid \Delta X_5^{(3)}[0] = 0) + \\
&\quad \mathbf{Pr}(\Delta X_5^{(3)}[0] = 1) \cdot \mathbf{Pr}(\Delta_0^{(3)}[0] = 1 \mid \Delta X_5^{(3)}[0] = 1) \\
&= (0.500287 \times 0.503152) + (1 - 0.500287) \times (1 - 0.503131) \\
&\approx 0.5000123.
\end{aligned}$$

Therefore the correlation is approximately 0.0000246. By experiment we find this value to be approximately 0.0000328, (Table II). Though there is a difference between this mathematically obtained result from the third round correlations and the full experimental result, both of them show that the correlation claimed by [8] for this position is incorrect. A similar theoretical approach can be used to verify the other results as well.

Keeping this knowledge in mind we have experimented with all other distinguishers which are listed in Table II with sufficient number of random samples. For this computation we have used the CPU Intel(R) Xeon(R) CPU E5-2670 v3 @ 2.30GHz, the OS is 64-bit Ubuntu-20.04. To achieve the required correlation we have run the programs parallelly using GCC compiler version 9.3.0 and for random number generation we have used $drand48()$ function in the programs. We need $\mathcal{O}\left(\frac{1}{pq^2}\right)$ random samples to distinguish between two events, one with probability p and the other with probability $p(1 + q)$, where q is small. We have computed the correlations at output difference positions $(0, 0)$, $(1, 0)$, $(12, 0)$, $(13, 0)$ for 3.5 rounds ChaCha with random samples 2^{37}. So in these cases we are using $\frac{6476.74}{pq^2}$, $\frac{1056.63}{pq^2}$, $\frac{19.86}{pq^2}$ and $\frac{17.59}{pq^2}$ samples and these should give success probability around $100\%, 100\%, 99\%$ and 98% respectively. But our experimental results do not match with their experimental results. We have listed a comparison of the experimentally obtained differential correlations in the Table II.

We have used $2^{46} \approx \frac{2.18}{pq^2}$ random samples to verify the correlation at position $(5, 0)$ of 3.5 rounds ChaCha but we got a lower correlation value than their claim. In this case, we have seen that when the number of random samples increases the correlation decreases. The major change in the paper [8] compared to the previous existing best result in [4] is that in the 3.5 rounds it uses an output difference position in the second row instead of first row of the state matrix (as in Crypto paper [4]). This approach is definitely interesting because the words in the second row are updated much later than in the first row. Therefore, it has a larger number of PNBs when one comes back from the 7-th round. This is also evident from the fact that they used 108 PNBs in there key recovery attack whereas in [4] the authors used only 74 PNBs. However, the forward 3.5 rounds correlations that they claimed in [8] are not accurate. As a result, we see that their attack does not actually improve the complexity of [4]. Rather according to our calculation, the distinguisher and key recovery time complexities are as follows. To calculate the linear correlation we have used the computational results 6-10 of [8, page 19 & page 21] which are $0.00867 \approx 2^{-6.85}$, $0.0416 \approx 2^{-4.59}$, $0.0278 \approx 2^{-5.19}$, $0.000398 \approx 2^{-11.29}$, $0.000047 \approx 2^{-14.38}$ and the corresponding linear correlation is $\epsilon_l = 2^{(-6.85-4.59-5.19-11.29-14.38)} = 2^{-42.3}$. Then using our obtained differential correlation $\epsilon = 0.0000000568 = 2^{-24.07}$,

we get the differential-linear correlation for 7 rounds ChaCha which is $\epsilon \cdot \epsilon_l^2 = 2^{-108.67}$. This gives us a distinguisher for 7 rounds of ChaCha with complexity $2^{217.34+5} = 2^{222.34}$ $\big($here the procedure has to be repeated 2^5 times on average as in [8]$\big)$. Using 2^{46} random values, we can claim with confidence 77% that the mentioned correlation for 3.5 rounds at position $(5, 0)$ in [8] is not correct and the actual value is much smaller than 0.0000002489. However, more experimental values are required to find the accurate value of ϵ.

Using our obtained forward correlation $\epsilon = 0.0000000568$ and their obtained backward correlation $\epsilon_a = 0.000169$ and 108 PNBs we get the data complexity $N = 2^{79.78}$ for optimal $\alpha = 33.27$ and the time complexity will be $2^{227.83}$. Then for the key recovery attack the final data and time complexities are $2^{84.78}$ and $2^{232.83}$ respectively (since the procedure has to be repeated 2^5 times on average as in [4], [8]). This time complexity is higher than the existing complexity $2^{230.86}$ of Crypto 2020 paper [4].

IX. Conclusion

We analysed the newly found 3.5 rounds single bit distinguisher in [4] which has a great impact on the key recovery attack of ChaCha. Also, we revisited the recently improved key recovery attack of 7 rounds ChaCha from Eurocrypt 2021 [8] and showed that their obtained 3.5 rounds single bit distinguishers are not accurate. The single bit distinguisher at position $(5, 0)$ has huge influence on the 7 rounds distinguisher and key recovery attack time complexities. Finally, we can conclude that the best key recovery attack till now has been given by Beierle et. al. at Crypto 2020 [4] with time complexity $2^{230.86}$. In general, theoretical analysis of such differential-linear attacks against ARX standards are not very frequent. In fact, the researches in these areas are mostly done by treating the cipher like a black box and obtaining results experimentally. But the works in this direction in the last few years have shown the importance of studying the various operations mathematically. Moreover, these studies are not only useful to one cipher, but can be also helpful to cryptanalyse other ARX ciphers such as Sparkle [3] permutation based authenticated cipher and hash function which is a finalist in the NIST lightweight competition [18]. Also, there are other ARX based designs such as the MAC algorithm Chaskey [16], block cipher Sparx [11] etc. Therefore, we believe that more importance should be given on mathematical studies of the ciphers for theoretical justification about security analysis.

References

[1] J. P. Aumasson, S. Fischer, S. Khazaei, W. Meier and C. Rechberger. New Features of Latin Dances: Analysis of Salsa, ChaCha, and Rumba. FSE 2008, LNCS 5086, pp. 470–488, 2008.

[2] A. R. Choudhuri and S. Maitra. Significantly Improved Multi-bit Differentials for Reduced Round Salsa and ChaCha. IACR Trans. Symmetric Cryptol 2016(2) pp. 261–287, 2016. Available at http://eprint.iacr.org/2016/1034.

[3] C. Beierle, A. Biryukov, L. Cardoso dos Santos, J.n Großschädl, L. Perrin, A. Udovenko, V. Velichkov and Q. Wang. Schwaemm and Esch: Lightweight Authenticated Encryption and Hashing using the Sparkle Permutation Family, 2019. https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/round-2/spec-doc-rnd2/sparkle-spec-round2.pdf

[4] C. Beierle, G. Leander, Y. Todo. Improved differential-linear attacks with applications to ARX ciphers. In: Annual International Cryptology Conference. pp. 329-358. Springer (2020).

[5] D. J. Bernstein. Salsa20 specification. eSTREAM Project algorithm description, http://www.ecrypt.eu.org/stream/salsa20pf.html, 2005.

[6] D. J Bernstein. ChaCha, a variant of Salsa20. In Workshop Record of SASC, volume 8, 2008.

[7] IANIX. Chacha usage and deployment. https://ianix.com/pub/chacha-deployment.html, 2020. Accessed: 2020-01-13.

[8] M. Coutinho and T. C. Souza Neto: Improved Linear Approximations to ARXCiphers and Attacks Against ChaCha. 2021. https://eprint.iacr.org/2021/224.pdf

[9] P. Crowley. Truncated differential cryptanalysis of five rounds of Salsa20. IACR 2005. http://eprint.iacr.org/2005/375.

[10] S. Dey and S. Sarkar. Improved analysis for reduced round Salsa and ChaCha. Discrete Applied Mathematics 227(2017) pp. 58-69, 2017.

[11] D. Dinu, L. Perrin, A. Udovenko, V. Velichkov, J. Großschädl and A. Biryukov. Design Strategies for ARX with Provable Bounds: Sparx and LAX. In Jung Hee Cheon and Tsuyoshi Takagi, editors, ASIACRYPT I, volume 10031 of Lecture Notes in Computer Science, pages 484–513, 2016.

[12] S. Fischer, W. Meier, C. Berbain, J. F. Biasse. Non-randomness in eSTREAM Candidates Salsa20 and TSC-4. Indocrypt 2006, LNCS 4329, pp. 2–16, 2006.

[13] A. Langley, W Chang, Nikos Mavrogiannopoulos, Joachim Stromberg-son, and Simon Josefsson. Chacha20-poly1305 cipher suites for transportlayer security (tls).RFC 7905, (10), 2016.

[14] S. Maitra. Chosen IV Cryptanalysis on Reduced Round ChaCha and Salsa. Discrete Applied Mathematics, volume 208, pp. 88–97, 2016.

[15] M. Matsui. Linear cryptanalysis method for DES cipher. In Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings, pages 386–397, 1993.

[16] N. Mouha, B. Mennink, A.V. Herrewege, D. Watanabe, B. Preneel, I. Verbauwhede. Chaskey: An efficient MAC algorithm for 32-bit microcontrollers. In: Joux, A., Youssef, A.M. (eds.) SAC 2014, Revised Selected Papers. LNCS, vol. 8781, pp. 306–323. Springer (2014)

[17] S. Muller. Documentation and analysis of the linux ran-domnumbergenerator-federalofficeforinformationsecu-rity (germany's),2019. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Studies/LinuxRNG/LinuxRNG_EN.pdf;jsessionid=6B0F8D7795B80F5EADA3DB3DB3E4043B.1_cid360?__blob=publicationFile&v=19.

[18] National Institute of Standards and Technology. Lightweight Cryptography (LWC) Standardization project, 2019. https://csrc.nist.gov/projects/lightweight-cryptography.

[19] Z. Shi, B. Zhang, D. Feng, W. Wu. Improved Key Recovery Attacks on Reduced-Round Salsa20 and ChaCha. ICISC 2012, LNCS 7839, pp. 337–351.

[20] L. Torvalds. Linux kernel source tree, 2016. https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=818e607b57c94ade9824dad63a96c2ea6b21baf3.

[21] Y. Tsunoo, T. Saito, H. Kubo, T. Suzaki and H. Nakashima. Differential Cryptanalysis of Salsa20/8. SASC 2007. : http://www.ecrypt.eu.org/stream/papersdir/2007/010.pdf