IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2023, No. 2, pp. 189–223. DOI:10.46586/tosc.v2023.i2.189-223

Boosting Differential-Linear Cryptanalysis of ChaCha7 with MILP

Emanuele Bellini[1], David Gerault[1], Juan Grados[1], Rusydi H. Makarim[2] and Thomas Peyrin[3]

[1] Cryptography Research Centre, Technology Innovation Institute, Abu Dhabi, UAE
{emanuele.bellini,david.gerault,juan.grados}@tii.ae
[2] Independent Researcher
rusydi.hasan@gmail.com
[3] Nanyang Technological University, Singapore, Singapore
thomas.peyrin@ntu.edu.sg

Abstract. In this paper, we present an improved differential-linear cryptanalysis of the `ChaCha` stream cipher. Our main contributions are new differential-linear distinguishers that we were able to build thanks to the following improvements: a) we considered a larger search space, including 2-bit differences (besides 1-bit differences) for the difference at the beginning of the differential part of the differential-linear trail; b) a better choice of mask between the differential and linear parts; c) a carefully crafted MILP tool that finds linear trails with higher correlation for the linear part. We eventually obtain a new distinguisher for `ChaCha` reduced to 7 rounds that requires $2^{166.89}$ computations, improving the previous record (ASIACRYPT 2022) by a factor of 2^{47}. Also, we obtain a distinguisher for `ChaCha` reduced to 7.5 rounds that requires $2^{251.4}$ computations, being the first time of a distinguisher against `ChaCha` reduced to 7.5 rounds. Using our MILP tool, we also found a 5-round differential-linear distinguisher. When combined with the probabilistic neutral bits (PNB) framework, we obtain a key-recovery attack on `ChaCha` reduced to 7 rounds with a computational complexity of $2^{206.8}$, improving by a factor $2^{14.2}$ upon the recent result published at EUROCRYPT 2022.

Keywords: Cryptanalysis · Differential-Linear Attack · ChaCha20

1 Introduction

Symmetric-key cryptographic primitives are considered the workhorses of cryptography and they represent a crucial element of the global security chain. While guarantees are usually provided by the designers for some specific attacks, the confidence we have in the security of most symmetric-key primitives is largely based on the amount of cryptanalysis conducted by the research community. It is therefore very important to push these attacks to their limits in order to better assess the security margin provided by the primitives, especially the high-profile ones.

Different constructions are possible when building symmetric-key primitives such as block ciphers or stream ciphers. Among these, Addition-Rotation-XOR (ARX) ciphers are generally seen as very efficient functions for software scenarios, where modular addition, word-wise rotation and eXclusive OR (XOR) are naturally available as processor instructions. Moreover, the combined usage of XORs and modular additions leads to interesting cryptographic properties, with a rapid increase of the function degree and a good resistance against differential and linear cryptanalysis. Typical examples of ARX primitives are

Received: 2023-03-01 Accepted: 2023-05-01 Published: 2023-06-16

the SHA-1 and SHA-2 hash functions [NIS15], the TEA [WN94] or SPECK [BSS+13] block ciphers.

The Salsa [Ber08c] or ChaCha [Ber08a] stream ciphers are two other prominent examples of high-profile ARX-based primitives, as they are used in many commercial products and crypto libraries [Ian]. Salsa was submitted as a candidate for eSTREAM (the ECRYPT Stream Cipher Project [ECR08]) in 2005 and was selected two years later for the final software portfolio. Bernstein then proposed ChaCha [Ber08b], a Salsa variant aiming at improving its performance and security (via a faster diffusion). Both Salsa and ChaCha have versions with 12 and 20 rounds and can handle 128 or 256-bit secret keys.

Related Works. Before ChaCha was proposed to the public, some attempts on applying differential cryptanalysis to Salsa, ChaCha's predecessor, were made, for example [Cro05, TSK+07, FMB+06]. Also, two works explored rotational properties of the ChaCha internal permutation [BBM21] and of the full stream cipher [BBB22], showing essentially that this kind of techniques are deemed to fail due to the presence of constants injected into the initial state.

All current best attacks on ChaCha are variations of the 2008 seminal work of Aumasson *et al.* [AFK+08]. As we are going to describe below, in the time span of 14 years, these works reduced the security of the best 7-round attack by approximately 26 bits, for an average of 1.86 bits per year. Furthermore, practical distinguishers exist for up to 6 rounds [CN21b], with a time complexity of 2^{51}.

The main technique proposed in [AFK+08] is inspired by correlation attacks and the notion of neutral bits [BC04]. It is a chosen plaintext attack, and consists in the following steps. First, build a distinguisher by applying some input difference to the initial state and observe a differential correlation after some rounds. Then, select randomly chosen keys to invert the ciphertext for a few round backwards in order to observe the differential correlation. Determine which bits of the key are not affecting the observation of the correlation (we call these bits the Probabilistic Neutral Bits or PNBs). Finally, find the value of the non-PNBs of the key by means of exhaustive search filtered by the observation of the correlation, and eventually the value of the remaining PNBs by exhaustive search. The structure of all main attacks on ChaCha remains similar to the one above, where attackers try to minimize the time complexity by either improving the success probability of the distinguisher or by increasing the number of PNBs.

The first distinguisher proposed in [AFK+08] was a 3-round truncated differential of a 1-bit input difference injected in the nonce and a 1-bit output difference. To attack R rounds, the output difference was observed after inverting $R-3$ rounds from an n-round keystream block. The largest R Aumasson *et al.* were able to attack was 7, with a time complexity of 2^{248} and a data complexity of 2^{27}. For future improvements, the authors suggested that an attacker could use multi-bit differentials and/or combinations of several biased output differences. On the other hand, their exhaustive search showed no bias in 4-round ChaCha, be it with one, two, or three target output bits. The authors also conjectured that *four seems to be the highest number of rounds one can invert from a partial key guess, while still observing a non-negligible bias after inversion, and such that the overall cost improves from exhaustive key search.* Indeed, this remains an open problem as of today. They also foresaw that, in order to break further rounds by statistical cryptanalysis, novel techniques and ideas are required, and 7 rounds is still the largest number of rounds for which ChaCha has been attacked.

In 2012, Shi *et al.* [SZFW12] improved the attacks of [AFK+08] by a couple of bits in time complexity, by considering a collection of distinguishers, called Column Chaining Distinguishers (CCD), depending on sub-keys sharing common bits, and exhibiting new high probability second-order differential trails. Overall, the attack can be seen as concurrent applications of the attack in [AFK+08], using 3 to 4 CCDs. It whould be shown later in

2016, by Maitra [Mai16], that the actual time complexity of this attack is $2^{246.7}$ making the considerable effort of using CCDs not really worth it. Maitra was also able to reach a time complexity of $2^{238.94}$ mostly by carefully picking IV values that yield higher biases in the output difference when combined with the keys. In 2016, around the time of `ChaCha20` standardization, Choudhuri and Maitra [CM16a] proposed the first practical distinguisher on `ChaCha5` with time complexity of 2^{16}. They were able to do so by taking into consideration the structure of the cipher and deriving 2-round correlations that linked the 3-round 1-bit output difference to a 21-bit mask. This was the first explicit example of differential-linear cryptanalysis of `ChaCha`, giving a time complexity of $2^{237.65}$ and a data complexity of 2^{96} on 7-round `ChaCha`. The ideas in [CM16a] were inspired by the work of [Leu16] and the differential-linear cryptanalysis techniques of [LH94]. All subsequent significant attacks on `ChaCha` use differential-linear (DL) distinguishers. Note that differential-linear cryptanalysis allowed to find the first known biases for 4, 4.5, and 5 rounds of `ChaCha` by exploiting masks of more than 3 bits, going beyond what had been tried in [AFK+08].

Building on top of [CM16a], in 2017, Dey and Sarkar [DS17] improved the search of PNBs by finding a combination of key bits which can act as good PNB as a whole set, rather than choosing those key bits which act as good PNB alone.

In 2020, Beierle *et al.* [BLT20] improved the current attacks on many different levels. They divided the DL distinguisher in 3 parts, by adding a middle layer of 2.5 rounds to address the problem of the wrong estimates for the correlation due to the independence assumption between the differential layer and the cipher. In the initial differential part, made of only a single round, they generate only plaintext pairs that satisfy the correlation of the differential-linear approximation with high probability, rather than picking them randomly. This allowed the authors to display the first 3.5-round single bit differential distinguisher (with probability 2^{-5} on average). In the linear part, they considered multiple linear approximations and a partitioning technique for ARX ciphers, inspired by [BC14, Leu16], that splits the space of ciphertexts into subsets in order to increase the correlation of linear approximations. Finally, in the key recovery part, they used the Fast Walsh-Hadamard Transform (FWHT) to recover key bits instead of guessing, due to the linear relation (several key bits actually appear only linearly in the approximations). For 6 rounds, they do not use the PNBs, but their new technique. For 7 rounds, they use standard PNBs technique, and reach a time complexity of $2^{230.86}$ and a data complexity of $2^{48.83}$.

In 2020, Coutinho and Souza [CN20] derived new linear approximations with worse correlation, but yielding more PNBs. Expanding this work, in 2021, Coutinho and Souza [CN21b] claimed to improve the attacks for 6 and 7 rounds of `ChaCha`, by explicitly deriving linear approximations for 3 and 4 rounds. Unfortunately, their work was revisited and invalidated by [DDSM21], who showed that the differential correlation for 3.5 rounds was much smaller than the one claimed. A new revised version was posted on ePrint, correcting the error [CN21a]. In that revised version, a 7-round distinguisher was found with complexity 2^{224}. Unfortunately, they did not improve the key-recovery attack presented by [BLT20].

In [DS20, DS21], Dey *et al.* show a theoretical interpretation of previous DL distinguishers against `ChaCha` and `Salsa`: they develop a probabilistic framework focusing on the non-linear component of the ARX cipher, the modular addition.

In [MIM21], Miyashita *et al.* deeply explores key-recovery attacks using single-bit output masks. Specifically, they search for the best key-recovery attacks using the PNB mentioned above attack and DL distinguishers with a single active bit in their output masks. As a result, they found a key-recovery attacks against `ChaCha` reduced to 7 rounds and 7.25 rounds. For the 7-round attack the time complexity is $2^{231.63}$ of and data complexity is $2^{49.58}$. For the 7.25-round attack the time complexity is $2^{255.62}$ of and data complexity

Table 1: Summary of the best attacks to `ChaCha7` with 256-bit key.

	Dist.	Key Recovery			
Ref.	Rounds	#PNBs	Time	Data	Novelty/Comments
2008, Aumasson *et al.* [AFK+08]	3+0+0	35	2^{248}	2^{27}	Truncated differential + PNB
2012, Shi *et al.* [SZFW12]	3+0+0	(35,34,32,28)	$2^{246.5}$	2^{27}	Column chaining distinguisher (CCD)
2016, Maitra [Mai16]	3+0+0	41	$2^{238.94}$	2^{96}	Chosen IV cryptanalysis
2017, Choudhuri *et al.* [CM16b]	4.5+0+1.5	50	$2^{237.65}$	2^{96}	1st diff-lin using multi-bit distinguisher
2017, Dey *et al.* [DS17]	4.5+0+1.5	53	$2^{235.22}$	-	Global PNBs
2020, Beierle *et al.* [BLT20]	1+2.5+1.5	74	$2^{230.86}$	$2^{48.83}$	1st 3.5-round single-bit differential
2021, Coutinho and Souza 2021, Miyashita *et al.* [MIM21]	3.5+0+0	74	$2^{231.63}$	$2^{49.58}$	Analysis of 1-bit masks on PNB attack
2021, Coutinho *et al.* [CN21b]	1+2.5+1.5	108	$2^{228.51}$	$2^{80.51}$	Invalidated ($2^{236.04}$, $2^{87.99}$)
2022, Dey *et al.* [DGSS22]	1+2.5+0.5	79	$2^{221.95}$	$2^{90.20}$	Better input pairs selection and new PNBs strategy
Our work	1+2+2	160	$\mathbf{2^{206.8}}$	$\mathbf{2^{110.81}}$	Minimizing the Hamming weight of the 5-round diff-lin distinguisher's output

is $2^{48.36}$. Although, these last complexities values are like brute force their distinguishers and analysis are valuable for future improvements.

In 2022, Dey *et al.* [DGSS22] improved the selection of right pairs in the differential part used in [BLT20]. They also provided an original strategy to get a good set of PNBs, going from 74 of [CN21b] to 79. This unlocked the first attack on 6.5 rounds of `ChaCha`. It is worth noting that the complexity calculation formula given by Aumasson *et al.* [AFK+08] was also revisited, in favor of a more accurate complexity.

In 2023, Coutinho *et al.* [CPV+22] improved the DL distinguisher correlation against `ChaCha` reduced to 7 rounds by a factor of 2^{10}. They do this, by observing carefully half rounds of `ChaCha`. Also, they manage to find, for first time ever, a distinguisher for `Salsa` reduced to 8 rounds, and also they propose, for the first time ever, a key-recovery attack against `Salsa` reduced to 8 rounds using the PNB technique. They found these attacks, by taking into advantage the diffusion properties of `Salsa`. Specifically, by exploiting the "hourglass structure" [MMH+14] presented in DL distinguihsers against ARX ciphers presented in the previous works.

Finally, in 2023, Dey *et al.* improved the best 6 rounds attack by a factor of 2^{40}, to $2^{99.48}$. This leap is made possible by combining multiple input and output difference pairs, and considering different sets of PNBs for each of them.

We summarize the progress of the above described attacks in Table 1.

Our Contributions. In this article, we propose a new differential-linear cryptanalysis of `ChaCha`, improving over the state-of-the-art.

First, in order to look for good linear approximations for the linear part of the differential-linear attack, we designed a new MILP model for `ChaCha` based on techniques similar to the ones proposed by Fu *et al.* [FWG+16] for the `SPECK` cipher (see Appendix A). This automation allows us to better cover the search space and find linear trails with higher correlation. As an example, the trail from round 3.5 to round 7 of `ChaCha`, as presented in Lemma 1 and Lemma 2, was found by us, using the lemmas presented in [CN20], with a theoretical probability of 2^{-47}. However, by using our linear trail MILP search model, a new candidate trail starting from the same input mask could be identified with an improved theoretical probability of 2^{-37}. Even better, a similar example occurs with Lemma 3, Lemma 4 and Linear Trail 3, where a trail from round 3 to round 7 was found by using the lemmas presented in [CN20] with a theoretical probability of 2^{-30}. However, by using our MILP search model for linear trails, a new candidate trail starting from the same input mask could be identified with an improved theoretical probability of 2^{-26} (see Subsubsection 3.3.2).

Secondly, we show that better differential-linear trails can be found on `ChaCha` by starting from a 2-bit difference at the beginning of the differential part of the trail (see

Table 2: Time complexity of 7-round and 7.5 distinguishers of `ChaCha` with 256-bit key.

Rounds	Rounds split	Complexity	Reference
7	1+2.5+3.5	2^{224}	Coutinho *et al.* [CN21a]
	3+4	2^{214}	Coutinho *et al.*[CPV+22]
	1+2+4	$2^{166.89}$	This work
7.5	1+2+4.5	$2^{251.54}$	This work

Subsection 3.2). Previously, all attacks only used a single bit flip on the input and this was quite natural in order to minimize the difference spread due to the good diffusion properties of `ChaCha`. Indeed, when searching for differential paths with high probability on ARX ciphers, it is generally a good strategy to minimize as much as possible the number of active bits. However, although those differential trails with a 2-bit difference at the beginning do not improve the differential part, their output differences produce useful 3-round and 3.5-round differential distinguishers when connected with the middle part. Specifically, these distinguishers help to have a better 7-round DL distinguisher than the previous attacks, and to have for the first time ever a DL distinguisher for `ChaCha` reduced to 7.5 rounds.

Thirdly, by using a better choice of masks between the differential and linear parts of the attack (at round 3.5) and by combining this with our previous improvements, we were able to find a new distinguisher for `ChaCha` reduced to 7 rounds (see Subsection 4.2). This distinguisher requires only $2^{166.89}$ computations, an improvement factor of about 2^{47} over the previously best-known distinguisher on the same primitive, published at ASIACRYPT 2022 [CPV+22]. By extending the same distinguinsher one round forward we found a distinguisher for `ChaCha` reduced to 7.5 rounds which requires $2^{251.54}$. To the best of our knowledge it is the first time of a distinguisher reaching 7.5 rounds. The aforementioned masks were found by observing that two active bits at round 3.5 of `ChaCha` can generate only one active bit in round 4 with a probability of 1. This dramatically reduces the number of active bits for the subsequent rounds and thus gives us a better distinguisher. This is explained in detail in Subsection 3.3.

Finally, using some constraints in our MILP tool, we found two new 4-round DL distinguisher. Plugging the first one into the PNB framework [AFK+08], we obtain a key-recovery attack on `ChaCha` reduced to 7 rounds with a computational complexity of $2^{213.14}$, improving by a factor $2^{8.76}$ over the recent key-recovery attack published at EUROCRYPT 2022 [DGSS22]. See Subsection 4.3. Plugging the second one into the PNB framework, we obtain a key-recovery attack on `ChaCha` reduced to 7 rounds with a computational complexity of $2^{217.58}$, improving by a factor $2^{4.31}$ over the recent key-recovery attack published at EUROCRYPT 2022 [DGSS22]. By utilizing the PNB framework in conjunction with the initial 5-round distinguisher from the preceding paragraph, we obtain a key-recovery attack on `ChaCha` reduced to 7 rounds. The computational complexity of this attack is $2^{206.8}$, which represents an improvement over the recently published key-recovery attack at EUROCRYPT 2022 [DGSS22], by a factor of $2^{14.2}$.

Our results and a comparison with the state-of-the-art are given in Table 1 and in Table 2.

We used GPUs in order to find and verify the new DL distinguishers. Furthermore, we implemented our MILP model using MiniZinc. All our code is made public for the community, and it is available at (Github: Link to Repository). It is important to note that our GPU experiments were conducted using the cryptDances library [Cou22], which we optimized to handle a larger number of samples. These optimizations resulted in a sixfold improvement in the speed of cryptDances to check some of our differential-linear distinguishers.

2 Preliminaries

2.1 Notation

In this article, we will use the following notation. The addition modulo 2^{32} (or the addition in $\mathbb{Z}$) of x and y will be denoted $x+y$ or $x \boxplus y$, while $\Theta_i(x,y)$ will represent the i^{th} carry bit of the operation. The bitwise exclusive-OR (XOR) operation of two words x and y of equal size will be denoted $x \oplus y$. Also, we will denote as $|x|$ the number of bits of x.

X^m (respectively X^{-m}) will represent the m^{th} state of `ChaCha` after m rounds (respectively of the inverse `ChaCha` after m rounds). When discussing differential attacks, the XOR-based difference observed on X^m will be denoted Δ^m and the differential starting from Δ_{in} and ending to Δ_{out} is denoted $\Delta_{in} \to \Delta_{out}$. We will denote as $\Delta_{in} \xrightarrow[\mathcal{DL}]{} \Gamma_{out}$ the DL distinguisher starting from Δ_{in} and ending to Γ_{out}. Similarly, we will denote as $\Gamma_{in} \xrightarrow[\mathcal{L}]{} \Gamma_{out}$ the linear distinguisher starting from Γ_{in} and ending to Γ_{out}. $X^m_{a,i}$ (respectively $\Delta^m_{a,i}$) will stand for the i^{th} bit of the word a of the state X^m (respectively the state difference Δ^m). Finally, to improve the readability of long linear approximations such as $x^R_{i,j_0} \oplus x^R_{i,j_1} \oplus \cdots \oplus x^R_{i,j_t}$, we use the shorter notation $x^R_i[j_0, j_1, ..., j_t]$. For some short linear approximations we stick to the explicit notation.

Given a set $\mathcal{S} \subset \mathbb{F}_2^n$ and a Boolean function $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$, we define

$$\mathbf{Cor}_{x\in\mathcal{S}}\left[f(x)\right] \coloneqq \frac{1}{|\mathcal{S}|}\sum_{x\in\mathcal{S}}(-1)^{f(x)}.$$

2.2 Description of the `ChaCha` Stream Cipher

In [Ber08a], Bernstein proposed `ChaCha`, a stream cipher manipulating a 128-bit or a 256-bit key. It was built to improve the diffusion properties of `Salsa` [Ber08c] while using the same amount of components. `ChaCha` works on 32-bit words, and its 512-bit (or 16-word) initial state is filled with four constants words $c_1 = 0x61707865$, $c_2 = 0x3320646e$, $c_3 = 0x79622d32$, $c_4 = 0x6b206574$, one counter word t, three nonce words n_1, n_2, n_3, and eight key words $k_1, \ldots, k_8$ for the 256-bit key version. The distribution of these words are depicted in Figure 2. In addition, we depict the output of the m^{th} state of `ChaCha` in Figure 3.

`ChaCha` is an iterative stream cipher composed of 20 rounds. Each round updates the i^{th} state of `ChaCha` (see Figure 3), using the following operations, called a quarter round (QR) and that takes 4 words (x_a, x_b, x_c, x_d) as input:

$$\begin{array}{lll}
x_a^{m-1} = x_a^{m-1} + x_b^{m-1}; & x_d^{m-1} = x_d^{m-1} \oplus x_a^{m-1}; & x_d^{m-1} = x_d^{m-1} \lll 16;\\
x_c^{m-1} = x_c^{m-1} + x_d^{m-1}; & x_b^{m-1} = x_b^{m-1} \oplus x_c^{m-1}; & x_b^{m-1} = x_b^{m-1} \lll 12;\\
x_a^{m} = x_a^{m-1} + x_b^{m-1}; & x_d^{m-1} = x_d^{m-1} \oplus x_a^{m-1}; & x_d^{m} = x_d^{m-1} \lll 8;\\
x_c^{m} = x_c^{m-1} + x_d^{m-1}; & x_b^{m-1} = x_b^{m-1} \oplus x_c^{m-1}; & x_b^{m} = x_b^{m-1} \lll 7;
\end{array}$$

If the round is an odd round then the QR equations are applied to the columns of the state. Thus, the first column, the second column, the third column, and the fourth column of the state x^m will be respectively

$$\begin{aligned}
(x_0^m, x_4^m, x_8^m, x_{12}^m) &= QR(x_0^{m-1}, x_4^{m-1}, x_8^{m-1}, x_{12}^{m-1}),\\
(x_1^m, x_5^m, x_9^m, x_{13}^m) &= QR(x_1^{m-1}, x_5^{m-1}, x_9^{m-1}, x_{13}^{m-1}),\\
(x_2^m, x_6^m, x_{10}^m, x_{14}^m) &= QR(x_2^{m-1}, x_6^{m-1}, x_{10}^{m-1}, x_{14}^{m-1}),\\
(x_3^m, x_7^m, x_{11}^m, x_{15}^m) &= QR(x_3^{m-1}, x_7^{m-1}, x_{11}^{m-1}, x_{15}^{m-1}).
\end{aligned}$$

If the round is even, the QR equations are applied to the diagonals of the state instead. Thus, the first diagonal, the second diagonal, the third diagonal, and the fourth diagonal of the state x^m will be respectively

$$
\begin{aligned}
(x_0^m, x_5^m, x_{10}^m, x_{15}^m) &= QR(x_0^{m-1}, x_5^{m-1}, x_{10}^{m-1}, x_{15}^{m-1}),\\
(x_1^m, x_6^m, x_{11}^m, x_{12}^m) &= QR(x_1^{m-1}, x_6^{m-1}, x_{11}^{m-1}, x_{12}^{m-1}),\\
(x_2^m, x_7^m, x_8^m, x_{13}^m) &= QR(x_2^{m-1}, x_7^{m-1}, x_8^{m-1}, x_{13}^{m-1}),\\
(x_3^m, x_4^m, x_9^m, x_{14}^m) &= QR(x_3^{m-1}, x_4^{m-1}, x_9^{m-1}, x_{14}^{m-1}).
\end{aligned}
$$

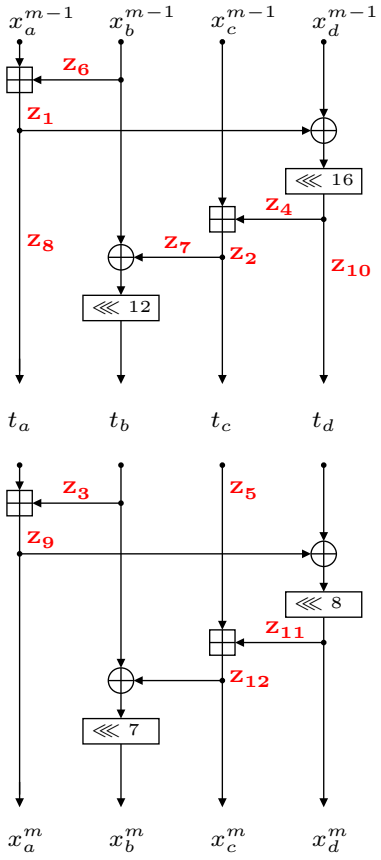

Figure 1: ChaCha quarter round diagram with intermediate variables $\mathbf{Z_1}, \ldots \mathbf{Z_{12}}$ and t_a, t_b, t_c, t_d.

c_1	c_2	c_3	c_4
k_1	k_2	k_3	k_4
k_5	k_6	k_7	k_8
t	n_1	n_2	n_3

Figure 2: Initial state of ChaCha.

x_0^m	x_1^m	x_2^m	x_3^m
x_4^m	x_5^m	x_6^m	x_7^m
x_8^m	x_9^m	x_{10}^m	x_{11}^m
x_{12}^m	x_{13}^m	x_{14}^m	x_{15}^m

Figure 3: X^m state of ChaCha.

In the literature, one can usually find descriptions of ChaCha by splitting the QR into two halves. Based on the description presented in [CN21b], we depict in Figure 1 the half-round of the QR, using (t_a, t_b, t_c, t_d) as intermediate words. In that figure, we can directly read the formulas of Equation 1 in the forward direction for the first half of the

quarter round:

$$\begin{aligned}
t_a &= x_a^{m-1} \boxplus x_b^{m-1} \text{ then } t_{a,i} = x_{a,i}^{m-1} \oplus x_{b,i}^{m-1} \oplus \Theta_i\left(x_a^{m-1}, x_b^{m-1}\right) \\
t_{b,i+12} &= x_{b,i}^{m-1} \oplus t_{c,i} \\
t_c &= x_c^{m-1} \boxplus t_d \text{ then } t_{c,i} = x_{c,i}^{m-1} \oplus t_{d,i} \oplus \Theta_i\left(x_c^{m-1}, t_d\right) \\
t_{d,i+16} &= x_{d,i}^{m-1} \oplus t_{a,i} \\
x_{a,i}^{m} &= t_{a,i} \oplus t_{b,i} \oplus \Theta_i\left(t_a, t_b\right) \\
x_{d,i+8}^{m} &= t_{d,i} \oplus x_{a,i}^{m} \\
x_{c,i}^{m} &= t_{c,i} \oplus x_{d,i}^{m} \oplus \Theta_i\left(t_c, x_d^{m}\right) \\
x_{b,i+7}^{m} &= t_{b,i} \oplus x_{c,i}^{m}
\end{aligned} \tag{1}$$

Also, we can deduce the following formulas in backward direction for the second half of the quarter round:

$$\begin{aligned}
t_{a,i} &= x_{a,i}^{m} \oplus x_{b,i+7}^{m} \oplus x_{c,i}^{m} \oplus \Theta_i\left(t_a, t_b\right) \\
t_{b,i} &= x_{b,i+7}^{m} \oplus x_{c,i}^{m} \\
t_{c,i} &= x_{c,i}^{m} \oplus x_{d,i}^{m} \oplus \Theta_i\left(t_c, x_d^{m}\right) \\
t_{d,i} &= x_{a,i}^{m} \oplus x_{d,i+8}^{m} \\
x_{a,i}^{m-1} &= \mathcal{L}_{a,i}^{m} \oplus \Theta_i\left(t_a, t_b\right) \oplus \Theta_i\left(t_c, x_d^{m}\right) \oplus \Theta_i\left(x_a^{m-1}, x_b^{m-1}\right) \\
x_{b,i}^{m-1} &= \mathcal{L}_{b,i}^{m} \oplus \Theta_i\left(t_c, x_d^{m}\right) \\
x_{c,i}^{m-1} &= \mathcal{L}_{c,i}^{m} \oplus \Theta_i\left(t_c, x_d^{m}\right) \oplus \Theta_i(x_c^{m-1}, t_d) \\
x_{d,i}^{m-1} &= \mathcal{L}_{d,i}^{m} \oplus \Theta_i\left(t_a, t_b\right)
\end{aligned} \tag{2}$$

where

$$\begin{aligned}
\mathcal{L}_{a,i}^{m} &= x_{a,i}^{m} \oplus x_{b,i+7}^{m} \oplus x_{b,i+19}^{m} \oplus x_{c,i+12}^{m} \oplus x_{d,i}^{m} \\
\mathcal{L}_{b,i}^{m} &= x_{b,i+19}^{m} \oplus x_{c,i}^{m} \oplus x_{c,i+12}^{m} \oplus x_{d,i}^{m} \\
\mathcal{L}_{c,i}^{m} &= x_{a,i}^{m} \oplus x_{c,i}^{m} \oplus x_{d,i}^{m} \oplus x_{d,i+8}^{m} \\
\mathcal{L}_{d,i}^{m} &= x_{a,i}^{m} \oplus x_{a,i+16}^{m} \oplus x_{b,i+7}^{m} \oplus x_{c,i}^{m} \oplus x_{d,i+24}^{m}
\end{aligned}$$

A common strategy in the literature is to combine Equation 1 and Equation 2 to minimize the number of terms and thus derive better linear approximations. In particular, we obtain a good linear approximation by combining equations $t_{a,i} = x_{a,i}^{m} \oplus x_{b,i+7}^{m} \oplus x_{c,i}^{m} \oplus \Theta_i\left(t_a, t_b\right)$ and $t_{b,i} = x_{b,i+7}^{m} \oplus x_{c,i}^{m}$. In fact, combining these equations we get $t_{a,i} \oplus t_{b,i} = x_{a,i}^{m} \oplus \Theta_i\left(t_a, t_b\right)$. By setting $i = 0$, we have an expression with only one active term at round m for each QR. This term is $x_{a,0}^{m}$. We will use this and other combinations in Section 3.

2.3 Differential-Linear Attack

The differential-linear attack was introduced by Langford and Hellman in [LH94] (we will refer to this version as the classical differential-linear attack, see the left side of Figure 4). Similarly to the boomerang attack [Wag99], the strategy of this attack consists into dividing a cipher E into two sub ciphers E_1 and E_2, such that $E = E_2 \circ E_1$. Then, one looks for a differential distinguisher and a linear distinguisher for the cipher E_1 and E_2 respectively. In particular, assume that the differential $\Delta_{in} \to \Delta_m$ holds with probability

$$\Pr_{x \in \mathbb{F}_2^n}\left[E_1(x) \oplus E_1(x \oplus \Delta_{in}) = \Delta_m\right] = p.$$

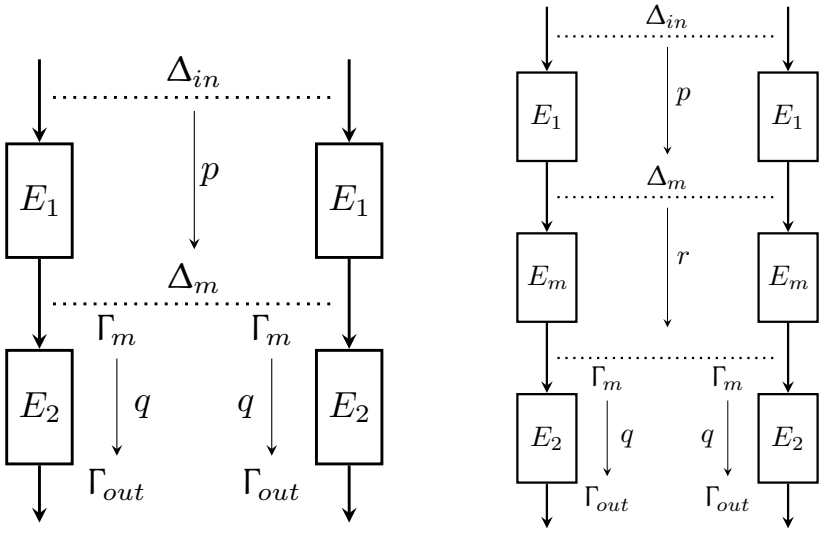

Figure 4: On the left-hand side, the structure of a classical DL distinguisher. On the right-hand side, a DL distinguisher with an improved structure that mitigates the independence assumption between the top and the bottom parts.

Moreover, let a certain linear trail $\Gamma_m \xrightarrow[\mathcal{L}]{E_2} \Gamma_{out}$ to be satisfied with correlation

$$\mathbf{Cor}_{x\in\mathbb{F}_2^n}\left[\langle\Gamma_m, x\rangle \oplus \langle\Gamma_{out}, E_2(x)\rangle\right] = q.$$

By assuming that $E_1(x)$ and $E_2(x)$ are independent random variables, the DL distinguisher exploits the property that

$$\mathbf{Cor}_{x\in\mathbb{F}_2^n}\left[\langle\Gamma_{out}, E(x)\rangle \oplus \langle\Gamma_{out}, E(x\oplus\Delta_{in})\rangle\right] = pq^2. \tag{3}$$

Thus, by preparing $\epsilon p^{-2}q^{-4}$ pairs of chosen plaintexts $(x, \tilde{x})$ for $\tilde{x} = x \oplus \Delta_{in}$, where $\epsilon \in \mathbb{N}$ is a small constant, one can distinguish the cipher from a random permutation.

The aforementioned assumption sometimes overestimates or underestimates the correlation value presented in Equation 3. Therefore, to mitigate this issue, a common strategy (see right-hand side of Figure 4) is to divide the cipher into three parts instead of two $E(x) = E_2 \circ E_m \circ E_1$, effectively adding a middle layer $E_m(x)$. For more details on this strategy, see [BDKW19]. This middle part is generally evaluated experimentally. In particular let

$$r = \mathbf{Cor}_{\mathcal{S}}\left[\langle\Gamma_m, E_m(x)\rangle \oplus \langle\Gamma_m, E_m(x\oplus\Delta_m)\rangle\right],$$

where $\mathcal{S}$ denotes the set of samples over which the correlation is computed. Then, the total correlation can be estimated as prq^2. We will also denote this improved DL distinguisher as:

$$\Delta_{in} \xrightarrow[\mathcal{DL}]{} \Gamma_{out}.$$

2.4 Probabilistic Neutral Bits Attack

Aumasson *et al.* proposed the PNB attack in [AFK+08] and applied it to `ChaCha` and `Salsa`. It is a key-recovery attack that consists essentially of two steps. In the first step, the attacker identifies some key bits to be discarded using a function f that depends on the key bits and a chosen r-round truncated differential. This truncated differential has a

correlation ε_d and a single active bit $\Delta^0_{i,j}$ on the input (others input bits differences are undetermined), and a single active bit $\Delta^r_{y,z}$ on the output (others output bits differences are undetermined). Where i, j, y and z are chosen so that ε_d is higher as possible. Those bits are called probabilistic neutral bits as they have no or little impact on the output of the computation of f. The remaining key bits are called significant bits. In a second step, the attacker constructs a function g similar to f. The only difference between this function g concerning f, is that the former depends on the significant bits instead of the entire key bits. Higher values of the correlation of g mean we can get better distinguishers. This second step is called backward computation. This attack has been improved several times, and the most recent one was presented in [DGSS22]. Due to its simplicity, we opted to use the original version since these last improvements did not alter our results.

2.4.1 Computing the Probabilistic Neutral Bits

Let f be a function with the following input parameters: a key k, a 96-bit nonce v, a counter t, and two states $Z = X + X^R$ and $Z' = X' + (X')^R$ where X is an initial state built using k, v and t, X' is computed by flipping the bit $X_{i,j}$, and R is the number of rounds. The output of f is $\left((Z - X)^{r-R} \oplus (Z' - X')^{r-R}\right)_{y,z}$. In other words,

$$f(k, v, t, Z, Z') = \left((Z - X)^{r-R} \oplus (Z' - X')^{r-R}\right)_{y,z}.$$

The objective of this first step is to statistically estimate which key bits do not influence the output of the computation of f for a certain number of rounds R. Specifically, for every key bit position l, we generate an instance of X randomly and we compute the output of f. After that, we compute the output of f with the same input parameters, but only flipping the key bit at position l. We repeat that process with N random samples of X and check how often the outputs are the same for every X. Let h be the number of times these outputs are the same; then the PNBs are those such that $2(h/N) - 1 < \lambda$ where λ is an input parameter of this first step.

2.4.2 Backward Computation

If we denote a guess of k by $\hat{k}$, then it is expected that $\Pr[f(k, v, t, Z, Z') = 1] = \frac{1}{2}$ for $k \neq \hat{k}$, and $\Pr[f(k, v, t, Z, Z') = 1] = \frac{1}{2} + \varepsilon_d$ for $k \neq \hat{k}$ on average for N random samples. In order to avoid attempting to guess all the key bits (as in a simple brute-force attack), Aumasson *et al.* proposed to use the aforementioned PNBs. Let $\bar{k}$ be a subkey of $m = |k| - n$ bits extracted from k where n is the number of PNBs. Let f be correlated to g with correlation ε_a, that is

$$\Pr[f(k, v, t, Z, Z') = g(\bar{k}, v, t, Z, Z')] = \frac{1}{2}(1 + \varepsilon_a).$$

Denote the correlation of g by ε, then under some reasonable independence assumptions, the equality $\varepsilon = \varepsilon_d \cdot \varepsilon_a$ holds.

Similarly to the previous step, we can compute ε experimentally, and obtaining a larger ε will naturally give a better distinguisher. One can find the details for calculating time and data complexity for this attack in [AFK+08]. Moreover, a correction to these complexities can be found in [DGSS22], so we only work with the corrected formulas in this paper. The time complexity is $2^{|k|-n}N + 2^{|k|-\alpha} + 2^{|k|-m}$, and the data complexity is given by

$$N = \left(\frac{\sqrt{\alpha \log(4)} + 3\sqrt{1 - \varepsilon_a^2 \varepsilon_d^2}}{\varepsilon_a \varepsilon_d}\right)^2,$$

where α is a parameter that the attacker can choose. Note that for ChaCha with a 256-bit key size, the time complexity is $2^{256-n}N + 2^{256-\alpha} + 2^n$.

3 Differential-Linear Attack on ChaCha

In Subsection 3.1, we present useful 1-round differential trails obtained by flipping two bits on the input difference instead of only one as in previous differential-linear attacks on ChaCha. In Subsection 3.2, we present useful 2-round and 2.5-round differential-linear trails obtained by carefully analyzing the ChaCha formulas presented in Subsection 2.2, and by minimizing the number of active bits of their output mask. In Subsection 3.3, we present linear distinguishers found manually and also found by using the MILP model explained in Appendix A. In our search, we minimize the hamming weight of their inputs and/or outputs, due to the diffusion properties of ChaCha.

3.1 1-round differential distinguishers with 2 active input bits

Symmetric-key ciphers such as ChaCha are designed specially to provide very good and fast diffusion of the differences that might appear in the internal state. Thus, cryptanalysts very often try to minimize the number of active bits present in the input and/or output difference, in a hope to keep the success probability as high as possible. Yet, we explored what happens for the total DL distinguisher correlation when we use more active bits on the input difference. Surprisingly, we could find differential paths with 2 input active bits that improve the situation for distinguishers or key-recovery attack on 7-round ChaCha. Even better, to the best of our knowledge, we found a 7.5 rounds distinguisher using these differential path with 2 input active bits.

Table 4 in Appendix B, shows some 1-round differential trails created by flipping two bits on the input in the same column of the ChaCha state. We obtained these trails using a MILP model that constrains the number of active bits in the inputs to two. That MILP model was constructed using the same techniques as in [FWG+16], but used to search differential trails (we do not give its description here as it did not help to improve the probability of the differential part).

This gave us many candidate paths with a probability of 2^{-7}. However, due to the large number of computations we require when connecting with the middle part, we did not explore all 1-round differential trails with 2-bit differences at the beginning. We took a random sample of 18 of them (see Table 4 in Appendix B). Only two of these trails, which starts from $\Delta^0_{in_4} = (\Delta^0_{14,21}, \Delta^0_{14,9})$ and end in $\Delta^1_{out_4} = (\Delta^1_{2,17}, \Delta^1_{2,5}, \Delta^1_{6,24}, \Delta^1_{6,8}, \Delta^1_{10,5}, \Delta^1_{10,1}, \Delta^1_{14,25}, \Delta^1_{14,1})$, and the one starting from $\Delta^0_{in_{17}} = (\Delta^0_{15,29}, \Delta^0_{15,9})$ and ending in $\Delta^1_{out_{17}} = (\Delta^1_{3,25}, \Delta^1_{3,5}, \Delta^1_{7,28}, \Delta^1_{7,12}, \Delta^1_{11,25}, \Delta^1_{11,21}, \Delta^1_{15,21}, \Delta^1_{15,13})$ were used to improve the time complexity of the previous differential-linear attacks against ChaCha.

Additionally, we explored differential trails with three active bits on the input in an effort to improve current DL distinguishers. However, we have not been successful. Again, we believe this is caused by the diffusion properties of ChaCha. In fact, our MILP model found that flipping 3 bits on the input results in a minimum of 12 active output bits, with a probability of 2^{-7}. Although they have the same probability of the differential trails we got by flipping two bits on the input, their minimum number of active output bits is larger by 4. We also explored 2-round differential trails on ChaCha, but as they lead to a much lower probability ($\leq 2^{-37}$), we could not use them to improve previous attacks.

3.2 2.5-round and 2-round differential-linear distinguishers

After an exhaustive search to find a good combination between the top part and the middle part, we found two useful DL distinguishers for the middle part, one with 2.5 rounds and the other one with two rounds. These distinguishers are presented in rows 5 and the final row of Table 5 (see Appendix B). Specifically, they are $\Delta^1_{out_4} \xrightarrow[\mathcal{DL}]{} x^{3.5}_{11,0}$ and $\Delta^1_{out_{17}} \xrightarrow[\mathcal{DL}]{} x^3_2[4,3,0] \oplus x^3_7[20,4,0] \oplus x^3_8[20,19] \oplus x^3_{13}[4]$, and we later utilized them to enhance both the 7-round DL distinguishers and the key recovery attack. Additionally, we used the distinguisher from the final row to achieve a significant milestone: the first-ever discovery of a DL distinguisher with 7.5 rounds. The 2.5-round and 2-round distinguishers were found experimentally using CUDA and the Piling-Up Lemma. Specifically, the first one was entirely found experimentally with CUDA, while for the second one we also used the Piling-Up Lemma. The use of this lemma was necessary as otherwise we would need an intractable amount of computational resources. We present in DL Distinguisher 1 and DL Distinguisher 2 the two DL distinguishers.

Differential-Linear Distinguisher 1. *The following 2.5-round DL distinguisher holds with a correlation of* $2^{-18.75}$ *(verified using* 2^{42} *random samples):*

$$\left(\Delta^1_{out_4}\right) = \left(\Delta^1_{2,17}, \Delta^1_{2,5}, \Delta^1_{6,24}, \Delta^1_{6,8}, \Delta^1_{10,5}, \Delta^1_{10,1}, \Delta^1_{14,25}, \Delta^1_{14,1}\right) \xrightarrow[\mathcal{DL}]{} \left(\Delta^{3.5}_{11,0}\right).$$

Differential-Linear Distinguisher 2. *The following 2-round DL distinguisher holds with a correlation of* $2^{-30.15}$*:*

$$\left(\Delta^1_{out_{17}}\right) = \left(\Delta^1_{3,25}, \Delta^1_{3,5}, \Delta^1_{7,28}, \Delta^1_{7,12}, \Delta^1_{11,25}, \Delta^1_{11,21}, \Delta^1_{15,21}, \Delta^1_{15,13}\right) \xrightarrow[\mathcal{DL}]{} \mathcal{A}',$$

where $\mathcal{A}' = \left(x^3_2[4,3,0] \oplus x^3_7[20,4,0] \oplus x^3_8[20,19] \oplus x^3_{13}[4]\right)$.

Table 3: A comparison of the estimation of the correlations of the groups created by partitioning the mask of DL Distinguisher 2 into four partitions. The column named Experimental shows the estimated correlation using only GPUs, while the column named Piling-Up lemma shows the estimated correlation using GPUs and the Piling-Up lemma.

Group	Partitions	Experimental	Piling-Up
1	Partition 1 and Partition 2	$2^{-9.56}$	$2^{-10.15}$
2	Partition 1 and Partition 3	$2^{-10.3}$	$2^{-12.21}$
3	Partition 1 and Partition 4	$2^{-22.59}$	$2^{-24.6}$
4	Partition 2 and Partition 3	$2^{-7.62}$	$2^{-8.44}$
5	Partition 2 and Partition 4	$2^{-20.48}$	$2^{-20.83}$
6	Partition 3 and Partition 4	$2^{-21.76}$	$2^{-22.89}$
7	Partition 1 and Partition 2 and Partition 3	$2^{-12.75}$	$2^{-15.4}$
8	Partition 2 and Partition 3 and Partition 4	$2^{-24.36}$	$2^{-26.08}$
9	Partition 1 and Partition 3 and Partition 4	-	-
10	Partition 1 and Partition 2 and Partition 4	$2^{-24.9}$	$2^{-27.79}$

To come up with DL Distinguisher 2 we realized that the state at round four coming from the propagation in the forward direction of the mask $x^{3.5}_{2,0} \oplus x^{3.5}_{7,0}$ has only one active bit. So $\mathcal{A}'$ is the linear mask generated from $x^{3.5}_{2,0} \oplus x^{3.5}_{7,0}$ when we expand it in a backward direction. We use Equations (1), to come up with $\mathcal{A}'$. Specifically, we use $x^{3.5}_{a,i} = x^3_{a,i} \oplus x^3_{b,i} \oplus \Theta_i\left(x^3_a, x^3_b\right)$ and $x^{3.5}_{b,i} = x^3_{b,i-12} \oplus x^{3.5}_{c,i-12}$. Setting $i = 0$, we have that

$$\begin{aligned} x^{3.5}_{a,0} \oplus x^{3.5}_{b,0} &= x^3_{a,0} \oplus x^3_{b,0} \oplus x^3_{b,-12} \oplus x^{3.5}_{c,-12} \\ &= x^3_{a,3} \oplus x^3_{c,19} \oplus x^3_{a,0} \oplus x^3_{d,4} \oplus x^3_{a,4} \oplus x^3_{b,0} \oplus x^3_{b,20} \oplus x^3_{c,20} \oplus x^3_{b,4} \end{aligned} \tag{4}$$

holds with probability $6/16 = 0.375$.

Before discussing the computation of DL Distinguisher 2 correlation, let us review some information about the dependencies present in the propagation of an input difference through two rounds of `ChaCha`. Each quarter round of `ChaCha` in round i is dependent on the outputs of the four quarter rounds from round $i-1$. Within a quarter round, the first and fourth words are influenced by the outcomes of three modular additions, while the second and third words are influenced by the outcomes of four modular additions. So, we can say that a word inside a quarter round depends on the output bits of the previous round plus the influence due to some modular additions of its own quarter round.

The Piling-Up Lemma is a commonly used lemma for computing the correlation of linear and DL distinguishers. Consider two independent random variables ϵ_α and ϵ_β, representing the correlation between an input difference and two output bits α and β, respectively. The correlation between the input difference and $\alpha \oplus \beta$ can be approximated by multiplying the individual correlations ϵ_α and ϵ_β, resulting in $\mathbf{Cor}[\alpha \oplus \beta] = \epsilon_\alpha \epsilon_\beta$. If we now want to compute the correlation of $\alpha \oplus \beta \oplus \gamma$, where α and β are dependent and $\alpha \oplus \beta$ and γ are independent, one approach is to first compute the correlation of $\alpha \oplus \beta$ without the use of the Piling-Up Lemma, for example through experimentation. After that, the Piling-Up Lemma can be applied between $\alpha \oplus \beta$ and γ.

After two rounds of `ChaCha` the situation is similar to the one discussed paragraphs above; specifically, after two rounds of `ChaCha`, any three terms α, β, and γ are dependent. Calculating correlations in these situations can be challenging, especially when the correlations are small. To overcome this, we partitioned the mask positions in a way that allows us to compute the correlations of each partition in a feasible time. What would be the best way to create the partitions? It is difficult to answer this question, but intuitively it seems better to create partitions containing stronger dependent terms, to avoid spreading the dependencies and, at the same time, save time in the correlation computation process. This approach was applied when computing the correlation of DL Distinguisher 2 and validated experimentally.

To compute the DL Distinguisher 2 correlation with $a = 2, b = 7, c = 8$, and $d = 13$, we consider the dependencies from two consecutive rounds of `ChaCha` and the dependencies from consecutive modular additions inside a `ChaCha` quarter round. We use CUDA and the Piling-Up Lemma. Initially, we attempted to determine the correlation of DL Distinguisher 2 by observing its convergence using 2^{48}, 2^{49}, and 2^{50} samples, but there was no stable convergence after these trials. Therefore, as a first approach, we decided to compute the correlation using the Piling-Up Lemma. Specifically, we divided the output mask into four partitions: 1) Partition 1: $x_2^3[4, 3, 0]$, 2) Partition 2: $x_7^3[20, 4, 0]$, 3) Partition 3: $x_8^3[20, 19]$, and 4) Partition 4: $x_{13}^3[4]$. For each partition, we compute its correlation experimentally, and after that, we use the Piling-Up Lemma to estimate the correlation of DL Distinguisher 2. As stated in the previous explanation, we divided the output mask into these partitions to prevent spreading the dependencies existing inside the quarter rounds at round 3 to different partitions.

The correlations, we found for these four partitions are $2^{-6.96}$, $2^{-3.19}$, $2^{-5.25}$, and $2^{-17.64}$, respectively, for each partition. Despite the fact that the Piling-Up Lemma requires independent terms, it still proves to be useful in estimating the lower bound of the correlation of DL Distinguisher 2. This can be seen in Table 3, where we can observe that for all the groups created from the partitions and for which we can calculate the correlation in a practical amount of time, the correlations we obtained experimentally are higher than those computed using the Piling-Up Lemma. As a result, to estimate the correlation of DL Distinguisher 2, we created two groups, one conformed by Partition 1, Partition 2, and Partition 4, and another one being Partition 3. In this way, the estimation for the correlation of DL Distinguisher 2 is $2^{-30.15}$. The cells marked with a dash symbol (-) in Table 3 indicate that the number of samples required to compute the correlation

exceeds two weeks.

In order to compute the correlation of each group listed in Table 3, we used cryptDances [Cou22]. We observe the convergences of these correlations through multiple trials. For instance, to determine the correlation of Group 8, we used up to 2^{55} samples and employed 32 GPUs (comprising 8 TITAN RTX, 8 Quadro RTX 6000/8000, and 16 GeForce RTX 2080). It took us approximately 11 days to compute the correlation of this group using this number of samples. So, in order to conduct experiments with larger sample sizes, such as when verifying the correlation of Group 8 or Group 10, we implemented optimizations to cryptDances code to improve its performance when computing DL Distinguisher 2. The optimizations we made were as follows:

- Given that DL Distinguisher 2 comprises only 2 rounds, we have created a macro F that consolidates all the ARX operations for precisely 2 rounds.
- To compute the number of parallel CUDA threads and CUDA blocks, we use the function *cudaOccupancyMaxPotentialBlockSize*, which returns the grid and block size that achieves maximum potential occupancy for a device function [1].
- To verify the DL Distinguisher 2 correlation, the program needs to evaluate a pair of states $(P, P \oplus \Delta P)$ and check the parity of the DL Distinguisher 2 mask on $F(P) \oplus F(P \oplus \Delta P)$. As ΔP only affects a few words of P, some operations in $F(P)$ can be reused in the computation of $F(P \oplus \Delta P)$, such as the modular addition between P_0 and P_5, i.e. $P_0 + P_5$.
- Since the DL Distinguisher 2 mask affects only a few words, some operations of every quarter round of round 3 (DL Distinguisher 2 starts at round 1) can be avoided. For instance, as DL Distinguisher 2 only affects word 2 of the first quarter, two XOR operations, one modular addition, one rotation left by 7, and one rotation left by 8 can be skipped.
- We employ the algorithm outlined in section 5-2 of [War12] to verify the parity of values within the output mask. By doing so, we are able to reduce the number of instructions required to verify the parity from 10 to 9.

After implementing these optimizations, we were able to speed up the verifications by a factor of 6. For instance, when checking the correlation of Group 8 using 2^{55} samples and the same number of GPUs, the time required was approximately 1.83 days, as opposed to the previous duration of 11 days.

We applied recommendations from [Tez21] to accelerate cryptographic operations, specifically rotations by 8 and 16, using the *__byte_perm*() CUDA instruction. However, our attempts did not yield any significant improvement. We also followed advice presented in [WCC21] in an attempt to optimize the timings. Specifically, we use the inline Parallel Thread Execution (PTX) techniques to consolidate the operations of every half quarter or every quarter round. Once again, we did not observe any significant improvement.

We tried to interpret DL Distinguisher 1 and DL Distinguisher 2 using the techniques presented in [NSLL22b], but we found divergences with the correlation we computed experimentally. For example, using the techniques presented in [NSLL22b], we obtained a value of $2^{-8.66}$ for DL Distinguisher 1 while for DL Distinguisher 2, we obtained a value of $\approx 2^{-45}$. In Table 6 of Appendix B, we summarize both the theoretical and practical correlation of every distinguisher presented in this section.

When writing this paper, a tool automating the search for differential-linear distinguishers appears in [BGG+23]. We have implemented the techniques presented in that paper and try to find the theoretical correlations of all our DL distinguishers using these

[1] A device function in CUDA programming is a function executed inside the GPU.

techniques; however, we obtained as output unsatisfiable (so we did not include these results on Table 6 of Appendix B). We believe this happens because the tool relies on Gurobi, which has a default precision of 2^{-32}, whereas the correlation values rapidly decrease over the rounds. As a result, precision violations may occur even after just a few rounds of `ChaCha`. You can find our attempts in (Github: Link to Anonymous Repository).

3.3 New Linear Trails for `ChaCha`

The linear part in the differential-linear attack is one of the most expensive. In fact, if we look at the complexity formula, presented in Subsection 2.3, we can see it has an exponent of two while the others factors do not have any exponent. Motivated by this, we look for useful linear trails starting at round 3, or 3.5, and ending at round 7, or 7.5, for the DL distinguishers. Also, we look for useful linear trails starting at round 3.5, or 2.5 and ending at round 4, or 5 for the key-recovery part. So we first study how to get linear trails manually in Subsubsection 3.3.1. After that, in Subsubsection 3.3.2, we automatize the search for linear trails using the MILP techniques presented in Appendix A. As in the search for good differential parts, we search for those linear trails with less terms than previous works since generally those have better correlation. In Table 6 of Appendix B, we summarize both the theoretical and practical correlation of every distinguisher presented in this section.

3.3.1 Linear Trails Found Manually

This section presents linear trails using the Piling-Up Lemma and other lemmas in [CM16b, CN21b]. The linear trails start at round 3 and at round 3.5, and end at round 7 or round 7.5. To find these linear trails, we tried to minimize the number of active terms across the rounds so we could have a higher or similar correlation than previous works. After extensive exploration, we notice that two linear trails obtained from the expressions $\mathcal{A}'$ (see DL Distinguisher 2) and $x_{c,i}^{3.5}$ have fewer terms at round 4 and 5. In particular, the former linear trail, starting from $\mathcal{A}'$, has only one term at round four. That differs from previous works since, to our knowledge, they have more than one term at round 4. For example, the 4-round DL distinguisher used in [BLT20] for its 6-round key-recovery attack has two terms. Specifically, the output mask is $x_{0,0}^4 \oplus x_{5,0}^4$. The 7-round DL distinguisher from [CN21a] has three terms at round four. Specifically, the output mask is $x_{3,0}^4 \oplus x_{4,7}^4 \oplus x_{9,0}^4$. The 4-round differential-linear used in [CN21a] to mount its key-recovery attack has two terms at round four. Specifically, the output mask is $x_{2,0}^4 \oplus x_{13,8}^4$. The 4-round DL distinguisher used in [DGSS22] to mount its key-recovery attack has three terms at round 4. Specifically, the output mask is $x_{2,0}^4 \oplus x_{7,7}^4 \oplus x_{8,0}^4$. Each of these 4-round distinguisher examples transition to round 5 with a probability lesser than 1, which is another distinction from our trail that starts from $\mathcal{A}'$: the single term at round four of our trail generates an expression at round five, with a probability of only one. To summarize, we obtain a 1.5-round linear trail with a probability of one.

The following lemmas present the steps for obtaining the two aforementioned linear trails. In particular, Lemma 1 and Lemma 2 outline the transitions from rounds 3.5 to 7, starting from $x_{c,i}^{3.5}$. Lemma 3 and Lemma 4 provide the transitions from rounds 3 to 7, starting from $\mathcal{A}'$. All proofs are provided in Appendix C.

Lemma 1. *The following 2.5-round linear trail holds with probability* $\frac{1}{2}(1 + 2^{-7})$*:*

$$x_{11}^{3.5}[0] = \mathcal{A}, \text{ where} \tag{5}$$

$$\begin{aligned}\mathcal{A} = {} & x_0^6[8,24] \oplus x_1^6[0,8,24] \oplus x_2^6[0] \oplus x_3^6[0] \oplus x_4^6[7,19,26] \oplus x_5^6[2,3,14,15,19,22,23] \oplus \\ & x_6^6[30,31] \oplus x_8^6[0] \oplus x_9^6[7,12,19] \oplus x_{10}^6[0,7,8,12,27,28] \oplus x_{11}^6[0,23,24] \oplus \\ & x_{12}^6[0,8,16] \oplus x_{13}^6[0,8] \oplus x_{14}^6[0,6,7] \oplus x_{15}^6[14,16,24].\end{aligned}$$

Lemma 2. *The following 1-round linear trail holds with probability* $\frac{1}{2}(1+2^{-40})$*:*

$$\mathcal{A} = \mathcal{B}, \text{ where} \tag{6}$$

$$\begin{aligned}\mathcal{B} = {} & x_0^7[0] \oplus x_4^7[6,7,10,11,13,22,23,27,30,31] \oplus x_8^7[3,4,6,8,15,16,19,20,26,31] \\ & \oplus x_{12}^7[7,8,18,19,22,25,26] \oplus x_1^7[6,7,11,12,16,18,19] \\ & \oplus x_5^7[1,2,9,11,19,21,22,26,27,30,31] \oplus x_9^7[0,3,4,8,20,22,23,26,27] \\ & \oplus x_{13}^7[2,3,7,8,11,12,18,20,21,22,23,26,27] \oplus x_2^7[0,6,8,11,12,16,22,23,27,28] \\ & \oplus x_6^7[13,14,17,18,19] \oplus x_{10}^7[6,8,10,11,27,28,30,31] \\ & \oplus x_{14}^7[3,4,7,11,12,15,16,19,20,24,27,28] \oplus x_3^7[0,8,14,16,23,30] \\ & \oplus x_7^7[7,19,20,21,22,23,30,31] \oplus x_{11}^7[0,12,13,14,15,16] \oplus x_{15}^7[0,6,16,23,24,31].\end{aligned}$$

Thus, to summarize, assuming independence between the sub-cipher starting at round 3.5 and ending at round 6, and the sub-cipher starting at round 6 and ending at round 7, the linear trail starting from $x_{11,0}^{3.5}$ and ending at $\mathcal{B}$ theoretically holds with a correlation of 2^{-47}.

Linear Trail 1. *We found manually the following 3.5-round linear trail, with probability* $\frac{1}{2}(1+2^{-47})$*:*

$$x_{11}^{(3.5)}[0] = \mathcal{B}.$$

Lemma 3. *The following 3-round linear trail holds with probability* $\frac{1}{2}(1+2^{-6})$*:* $\mathcal{A}' = \mathcal{C}$ *where* $\mathcal{A}' = \left(x_2^3[4,3,0] \oplus x_7^3[20,4,0] \oplus x_8^3[20,19] \oplus x_{13}^3[4]\right)$ *and*

$$\begin{aligned}\mathcal{C} = {} & x_0^6[11,12] \oplus x_2^6[0] \oplus x_3^6[0,16] \oplus x_4^6[7] \oplus x_6^6[6,26] \oplus x_7^6[7,19] \oplus x_8^6[12] \oplus x_9^6[0] \oplus \\ & x_{10}^6[12] \oplus x_{11}^6[7,31,6,18] \oplus x_{12}^6[7,19] \oplus x_{13}^6[0] \oplus x_{14}^6[24] \oplus x_{15}^6[11,12,19,20].\end{aligned}$$

Lemma 4. *The following 1-round linear trail holds with probability* $\frac{1}{2}(1+2^{-24})$*:*

$$\mathcal{C} = \mathcal{D}, \text{ where} \tag{7}$$

$$\begin{aligned}\mathcal{D} = {} & x_0^7[3,7,19,23] \oplus x_4^7[13,14,18,19,25,30,31] \oplus x_8^7[6,12,18,23,24] \\ & \oplus x_{12}^7[6,7,10,19,20,31] \oplus x_1^7[16] \oplus x_5^7[7] \oplus x_{13}^7[0,8,24] \oplus x_2^7[0,8,11,12,24] \\ & \oplus x_6^7[7,13,19,25,30,31] \oplus x_{10}^7[18,23,24,26] \oplus x_{14}^7[0,5,6,11,12,16,19,20,25,26] \\ & \oplus x_3^7[0,3,4,6,7,11,12,16,17,18,19,20,27,28,30,31] \oplus x_7^7[2,3,6,7,18,22,23,27] \\ & \oplus x_{11}^7[6,11,18,19,20,27,28] \oplus x_{15}^7[0,3,4,5,7,11,12,14,16,18,19,25,26,30,31].\end{aligned}$$

Thus, to summarize, assuming independence between the sub-cipher starting at round 3 and ending at round 6, and the sub-cipher starting at round 6 and ending at round 7, the linear trail starting from $\mathcal{A}'$ and ending at $\mathcal{D}$ theoretically holds with a correlation of 2^{-30}.

Linear Trail 2. *We found manually the following 4-round linear trail, with probability* $\frac{1}{2}(1+2^{-30})$*:* $\mathcal{A}' = \mathcal{D}$.

3.3.2 Linear Trails using our MILP Implementation

Looking at the proof for the linear trails presented in Subsubsection 3.3.1, we can see that it is possible to have several ways to group terms across the rounds and thus obtain different linear trails starting from the terms $x_{c,i}^{3.5}$ and $\mathcal{A}'$. So, we implemented the MILP techniques presented in Appendix A to search linear trails with higher correlation than Linear Trail 1 and Linear Trail 2, while starting from the same input masks. In the Linear Trail 3 and Linear Trail 4, we present the aforementioned linear trails that start from $x_{c,i}^{3.5}$ and $\mathcal{A}'$ respectively. In particular, the correlation for Linear Trail 4 increased dramatically from Linear Trail 2, so we could verify its correlation experimentally using CUDA, and we observed a correlation of $\epsilon_L = 2^{-23.11}$ (verified using 2^{51} random samples). We also experimentally verified Linear Trail 3 by dividing the trail into two sub-trails. The first sub-trail starts at round 3.5 and ends at round 6.5, while the second sub-trail start at round 6.5 and ends at round 7. For the first sub-trail, we get a correlation of $2^{-15.389}$, while for the second one, $2^{-19.30}$, assuming independence between these two sub-trails we obtain the value of as an estimation for the correlation of Linear Trail 3 (Readers can verify the sub-trails by accessing the repository that accompanies this paper).

Linear Trail 3. *We found the following 3.5-round linear trail, with probability* $\frac{1}{2}(1+2^{-37})$*, using our MILP model, starting in* $x_{11,0}^{(3.5)}$ *as in Lemma 1:*

$$\begin{aligned} x_{11}^{(3.5)}[0] =& x_0^7[0] \oplus x_1^7[19,18,16,12,11,7] \oplus x_2^7[28,23,16,12,8,6,0] \oplus x_3^7[31,16,8,0] \oplus \\ & x_4^7[31,30,27,26,23,22,13,11,10,7,6] \oplus x_5^7[31,30,27,26,22,19,11,6,2] \oplus \\ & x_6^7[19,18,17,14] \oplus x_7^7[31,30,23,19,7] \oplus x_8^7[31,26,25,23,20,16,15,8,6,4,3] \oplus \\ & x_9^7[31,27,23,20,19,8,7,4,3,0] \oplus x_{10}^7[31,28,10,8,7] \oplus x_{11}^7[16,12,0] \oplus \\ & x_{12}^7[26,19,18,8] \oplus x_{13}^7[27,26,23,20,19,14,12,11,8,3,2] \oplus \\ & x_{14}^7[28,24,20,16,15,12,11,7,4] \oplus x_{15}^7[24,16,7,0]. \end{aligned}$$

Linear Trail 4. *We found the following 4-round linear trail, with probability* $\frac{1}{2}(1+2^{-26})$*, using our MILP model, starting in* $\mathcal{A}'$ *as in Lemma 3:*

$$\mathcal{A}' = \mathcal{B}', \text{ where}$$

$$\begin{aligned} \mathcal{B}' =& x_0^7[23,22,19,7,3,2] \oplus x_1^7[16] \oplus x_2^7[24,23,12,8,0] \oplus \\ & x_3^7[31,28,20,16,12,11,7,6,4,3,0] \oplus x_4^7[31,19,14] \oplus x_5^7[7] \oplus \\ & x_6^7[31,25,19,13,7] \oplus x_7^7[27,26,23,22,7,6,3,2] \oplus x_8^7[24,12] \oplus \\ & x_{10}^7[26,25,24,18] \oplus x_{11}^7[28,27,20] \oplus x_{12}^7[31,30,20,19,11,10,7,6] \oplus \\ & x_{13}^7[24,8,0] \oplus x_{14}^{[10]}[26,20,16,12,11,6,5,0] \oplus x_{15}^7[31,19,18,16,14,12,11,7,4,0]. \end{aligned}$$

Studying the key-recovery attack presented in Subsection 2.4, we realized that, in general, key-recovery attacks with lower complexity could be mounted when the number of active bits at the output of the DL distinguishers is minimized. So, using our MILP model, we minimize the number of active bits at the output of the DL distinguishers by constraining them to one. We chose one since, as we mentioned before, previous works use 2 or 3 active bits at the output.

Something similar happens when we need to connect the middle part and the linear part. Specifically, if we look for a good correlation for the middle part, we need to minimize the input mask of the linear part. Again, using our MILP model, we minimize the number of active bits at input of the linear part of the DL distinguishers. In Linear Trail 5 and Linear Trail 6, we present two 1.5 linear trails obtained by minimizing their input and output masks.

Linear Trail 5. *We found the following 1.5-round linear trail, with probability* $\frac{1}{2}(1+2^{-12})$*, using our the MILP model:*

$$\mathcal{E} = x_0^4[0], \text{ where}$$
$$\begin{aligned}\mathcal{E} = {} & x_0^{2.5}[0,4] \oplus x_1^{2.5}[5,17,21] \oplus x_2^{2.5}[11,12] \oplus x_3^{2.5}[27,28] \oplus x_4^{2.5}[0,4] \oplus \\ & x_5^{2.5}[5,13,17,21,25,29] \oplus x_6^{2.5}[12] \oplus x_7^{2.5}[28] \oplus x_9^{2.5}[13,25,29] \oplus x_{10}^{2.5}[20] \oplus \\ & x_{13}^{2.5}[4,5,16,17,20,21] \oplus x_{14}^{2.5}[12] \oplus x_{15}^{2.5}[28].\end{aligned}$$

We could verify experimentally the correlation of Linear Trail 5 using CUDA and we observed a correlation of $\epsilon_L = 2^{-11.1}$ (verified using 2^{48} random samples).

Differential-Linear Distinguisher 3. *The following 1.5-round DL distinguisher holds with a correlation of* $2^{-17.5}$ *(experimentally verified using* 2^{42} *samples):*

$$\left(\Delta^1_{2,2}, \Delta^1_{6,5}, \Delta^1_{6,9}, \Delta^1_{6,17}, \Delta^1_{6,29}, \Delta^1_{10,10}, \Delta^1_{10,22}, \Delta^1_{10,30}, \Delta^1_{14,10}, \Delta^1_{14,30}\right) \xrightarrow[\mathcal{DL}]{} (\mathcal{E}).$$

Finally, combining DL Distinguisher 3 and Linear Trail 5, and assuming independence between the 1.5-round trail of DL Distinguisher 3 and the 1-round trail of Linear Trail 5, we can construct the 3-round DL Distinguisher 4 that will be used in the next section to mount a key-recovery attack on 7-round reduced `ChaCha`.

Differential-Linear Distinguisher 4. *The following 3-round DL distinguisher holds with a correlation of* $2^{-17.5-22.2} = 2^{-39.8}$*:*

$$\left(\Delta^1_{2,2}, \Delta^1_{6,5}, \Delta^1_{6,9}, \Delta^1_{6,17}, \Delta^1_{6,29}, \Delta^1_{10,10}, \Delta^1_{10,22}, \Delta^1_{10,30}, \Delta^1_{14,10}, \Delta^1_{14,30}\right) \xrightarrow[\mathcal{DL}]{} \left(x^4_{0,0}\right).$$

The input of DL Distinguisher 4 was obtained by iterating overall differentials trails proposed in the page 25 of [BLT20]. Specifically, that one starting in $\Delta^0_{14,6}$ and ending in $\left(\Delta^1_{2,2}, \Delta^1_{6,5}, \Delta^1_{6,9}, \Delta^1_{6,17}, \Delta^1_{6,29}, \Delta^1_{10,10}, \Delta^1_{10,22}, \Delta^1_{10,30}, \Delta^1_{14,10}, \Delta^1_{14,30}\right)$.

Linear Trail 6. *We found the following 1.5-round linear trail, with probability* $\frac{1}{2}(1+2^{-12})$*, using our MILP model:*

$$\mathcal{F} = x_3^4[0], \text{ where}$$
$$\begin{aligned}\mathcal{F} = {} & x_0^{2.5}[5,17,21] \oplus x_1^{2.5}[12] \oplus x_2^{2.5}[28] \oplus x_3^{2.5}[0,4] \oplus x_4^{2.5}[5,13,17,21,25,29] \oplus x_5^{2.5}[11,12] \oplus \\ & x_6^{2.5}[27,28] \oplus x_7^{2.5}[0,4] \oplus x_8^{2.5}[13,25,29] \oplus x_9^{2.5}[20] \oplus \\ & x_{12}^{2.5}[4,5,16,17,20,21] \oplus x_{13}^{2.5}[12] \oplus x_{14}^{2.5}[28].\end{aligned}$$

We could verify the correlation of Linear Trail 6 experimentally using CUDA, and we observed a correlation of $\epsilon_L = 2^{-11.19}$ (verified using 2^{48} random samples).

Linear Trail 7. *We found the following 4.5-round linear trail, with probability* $\frac{1}{2}(1+2^{-48})$*, using our MILP model:*

$$\mathcal{A}' = \mathcal{C}', \text{ where}$$
$$\begin{aligned}\mathcal{C}' = {} & x_0^{7.5}[31,23,16,14,12,11,6,4,3,0] \oplus x_1^{7.5}[31,30,20,19,16,11,10,7,6] \oplus x_2^{7.5}[12] \\ & \oplus x_3^{7.5}[31,28,26,11,7,6,5,4] x_4^{7.5}[28,27,26,24,19,16,12,10,8,7,0] \oplus x_5^{7.5}[31,15,3] \\ & \oplus x_6^{7.5}[31,28,27,25,19,11,5] \oplus x_7^{7.5}[24,23,20,18,15,14,12,7,6,4,3,2] \\ & \oplus x_8^{7.5}[27,26,23,22,8,6,3,2,0] \oplus x_9^{7.5}[30,28,27,20,16,15,14,12,7,4,0] \\ & \oplus x_{10}^{7.5}[26,24,23,19,18,3] \oplus x_{11}^{7.5}[31,28,25,20,19,16,15,13,7] \\ & \oplus x_{12}^{7.5}[28,27,26,23,22,20,19,15,14,4,3] \oplus x_{13}^{7.5}[23,16,12,8] \\ & \oplus x_{14}^{7.5}[28,27,22,21,16,10,4,0] \oplus x_{15}^{7.5}[30,28,27,26,24,20,18,17,16,15,3,2,0].\end{aligned}$$

We conducted an experimental verification of Linear Trail 7 by dividing the 4.5 rounds into two segments. The first trail began at round 3 and ended at round 7, while the second trail began at round 7 and ended at round 7.5. The first trail occurred between $\mathcal{A}'$ and $\mathcal{B}'$, while the second trail took place between $\mathcal{B}'$ and $\mathcal{C}'$. Our calculations, which were performed using CUDA, produced a value of $2^{-23.15}$ for the first trail and a value of $2^{-21.16}$ for the second trail. Assuming that the first and second segments are independent, the experimental correlation for Linear Trail 7 is $2^{-44.31}$.

Differential-Linear Distinguisher 5. *The following 1.5-round DL distinguisher holds with a correlation of* $2^{-15.6}$ *(verified using* 2^{40} *samples):*

$$\left(\Delta_3^1[18], \Delta_7^1[1,13,21,25], \Delta_{11}^1[6,14,26], \Delta_{15}^1[14,26]\right) \xrightarrow[\mathcal{DL}]{} (\mathcal{F}) .$$

Combining DL Distinguisher 5 and Linear Trail 6, we construct the 3-round DL Distinguisher 6 that we use to mount a key-recovery attack on 7-round reduced `ChaCha`.

Differential-Linear Distinguisher 6. *The following 3-round DL distinguisher holds with a correlation of* $2^{-15.6-2\times(11.19)} = 2^{-37.98}$*:*

$$\left(\Delta_3^1[18], \Delta_7^1[1,13,21,25], \Delta_{11}^1[6,14,26], \Delta_{15}^1[14,26]\right) \xrightarrow[\mathcal{DL}]{} \left(x_{3,0}^4\right) .$$

4 New Attacks on Reduced `ChaCha`

In this section, we will use the 2.5-round DL Distinguisher 1 and the 2-round DL Distinguisher 2 to build 7-round and 7.5-round DL distinguishers. In addition, we will use the 2.5-round DL Distinguisher 1, the 2-round DL Distinguisher 2, the 3-round DL Distinguisher 4, the 3-round DL Distinguisher 6 to mount key-recovery attacks on 7 and 7.25 rounds of `ChaCha` using 2^{46} samples to verify ε_a (see Subsubsection 2.4.2). Before, we will explain we use a previously-known technique to extend our DL distinguisher by 1 round for a low computational cost.

4.1 Extending the DL distinguishers by One Round

We apply the technique from [BLT20] that allows to extend an existing DL distinguisher with differential correlation $\epsilon = rq^2$ by prepending a differential path of probability p, obtaining a longer DL distinguisher for which the time/data complexity is $1/(pr^2q^4)$ instead of the expected $1/(p^2r^2q^4)$. The idea is to make sure that enough neutral bits exist for this extra differential part, so as to amortize the computational cost spent to find one valid pair for this part, thus leading to a saving factor $1/p$ on the total attack complexity. Note that this technique can only work if the number e of such bits is $2^e > 1/\epsilon^2$. Another way to look at this technique is to consider that the differential correlation of the extended DL distinguisher is rq^2 instead the expected prq^2 (we call this the reduced differential correlation), but the entire attack would have to be run $1/p$ times.

In our case, we will prepend the 1-round differential trail (with probability $p = 2^{-5}$):

$$\Delta_{14,6}^0 \to \left(\Delta_{2,2}^1, \Delta_{6,5}^1, \Delta_{6,9}^1, \Delta_{6,17}^1, \Delta_{6,29}^1, \Delta_{10,10}^1, \Delta_{10,22}^1, \Delta_{10,30}^1, \Delta_{14,10}^1, \Delta_{14,30}^1\right) ,$$

to DL Distinguisher 4 to obtain a 4-round DL distinguisher with reduced differential correlation of $2^{-39.3}$. This path allows enough neutral bits to fulfill the $2^e > 1/\epsilon^2$ requirement. This aforementioned differential trail was first presented in [BLT20] and since then it has been widely used for differential-linear attacks on `ChaCha`.

Regarding DL Distinguisher 1, we prepend the 1-round differential trail starting from $\Delta_{in_4}^0$ and ending to $\Delta_{out_4}^1$ (see Table 4 in Appendix B) with a probability $p = 2^{-7}$ to

obtain a 3.5-round DL distinguisher with reduced differential correlation of $2^{-18.75}$. Again, enough neutral bits exist for this path to fulfill the requirement.

Regarding DL Distinguisher 2, we prepend the 1-round differential trail starting from $\Delta^0_{in_{17}}$ and ending to $\Delta^1_{out_{17}}$ with a probability $p = 2^{-7}$ to obtain a 3-round DL distinguisher with a reduced differential correlation of $2^{-30.15}$. Again, enough neutral bits exist for this path to fulfill the requirement.

Regarding DL Distinguisher 6, we prepend the following 1-round differential trail (with a probability $p = 2^{-5}$):

$$\Delta^0_{15,22} \rightarrow \left(\Delta^1_3[18], \Delta^1_7[1, 13, 21, 25], \Delta^1_{11}[6, 14, 26], \Delta^1_{15}[14, 26]\right),$$

to obtain a 3.5-round DL distinguisher with reduced differential correlation of $2^{-22.17}$. Again, enough neutral bits exist for this path to fulfill the requirement.

4.2 New Differential-Linear Distinguishers on 7/7.5 rounds of ChaCha

Using Linear Trail 3 with experimental correlation $\varepsilon_L = 2^{-34.6}$ and the reduced differential correlation $\varepsilon_d \approx -2^{-18.75}$ of the 3.5-round extended version of DL Distinguisher 1, we get $\varepsilon_d(\varepsilon_L)^2 \approx 2^{-18.75-69.2} = 2^{-87.95}$ which gives us a distinguisher for 7 rounds of ChaCha with complexity $2^{2\times(87.95+7)} = 2^{189.9}$ taking into consideration the 1-round differential trail of the top part.

Using the experimental correlation $\varepsilon_L = 2^{-23.15}$ of Linear Trail 4, the differential correlation $\varepsilon_d \approx -2^{-30.15}$ of the 3-round extended version of DL Distinguisher 2, we get $\varepsilon_d(\varepsilon_L)^2 \approx 2^{-7-30.15-46.3} = 2^{-83.45}$ which gives us a distinguisher for 7 rounds of ChaCha with complexity $2^{166.89}$ taking into consideration the 1-round differential trail of the top part.

Using Linear Trail 7 with experimental correlation $\varepsilon_L = 2^{-44.31}$, the differential correlation $\varepsilon_d \approx -2^{30.15}$ of the 3-round extended version of DL Distinguisher 2, we get $\varepsilon_d(\varepsilon_L)^2 \approx 2^{-7-30.15-44.31\times 2} = 2^{-125.77}$ which gives us a distinguisher for 7.5 rounds of ChaCha with complexity $2^{251.54}$ taking into consideration the 1-round differential trail of the top part.

4.3 New PNB-based Key-Recovery Attack on 7/7.25 rounds of ChaCha

The 3.5-round extended version of DL Distinguisher 1 can be extended to a 4-round DL distinguisher appending 0.5 round of linear trail at the end. This extension occurs with probability one because there is no carry involved in this extension. In fact, looking at Equation 2, you will observe that this extension starts at the least significant bit (LSB) of the word $\Delta^{3.5}_{11,0}$. The reduced differential correlation for this 4-round DL distinguisher remains $\varepsilon_d = 2^{-18.75}$. We combined this DL distinguisher with the PNB technique found in Subsection 2.4 to yield better result than [BLT20]. Specifically, we use the differential correlation for $x^4_{11,0} \oplus x^4_{12,0}$, and the PNB attack described in Subsection 2.4. Thus, considering $\Delta^1_{out_4} \xrightarrow[\mathcal{DL}]{} x^4_{11,0} \oplus x^4_{12,0}$, to attack 7 rounds, we need to go back 3 rounds to reach $x^4_{11,0} \oplus x^4_{12,0}$. In this case, using $\gamma = 0.24$ we found 126 PNBs (see List of PNBs 1) and we obtained $\varepsilon_a = 2^{-22}$. From that, we get an attack with data complexity of $2^{88.07}$ and time complexity $2^{256-126} \times 2^{88.07} + 2^{256-38} + 2^{126} = 2^{219.03}$ for $\alpha = 38$. We have to repeat this attack 2^7 times on average because of the transition probability from ΔX^0 to $\Delta X^{(1)}$ when using the extension trick from [BLT20]. Thus, the final attack has data complexity of $2^{95.07}$ and time complexity $2^{226.03}$.

The 3-round extended version of DL Distinguisher 2 can be extended to a 5-round DL distinguisher appending a 2-round linear trail at the end. This extension occurs with a correlation of 2^{-2}. In fact, the transition from round 3 to round 3.5 occurs with that correlation, while the transition from round 3.5 to round 5 occurs with probability one. The

reduced differential correlation for this 5-round DL distinguisher is then $\varepsilon_d = 2^{-30.15-2\times 2}$. We combined this DL distinguisher with the PNB technique found in Subsection 2.4 to yield the best results for key-recovery attacks against ChaCha reduced to 7 rounds. Specifically, we use the differential correlation for $x^5_{2,0} \oplus x^5_{6,7} \oplus x^5_{6,19} \oplus x^5_{10,12} \oplus x^5_{14,0}$, and the PNB attack described in Subsection 2.4. Thus, considering $\Delta^1_{out_{17}} \xrightarrow[\mathcal{DL}]{} x^5_{2,0} \oplus x^5_{6,7} \oplus x^5_{6,19} \oplus x^5_{10,12} \oplus x^5_{14,0}$, to attack 7 rounds, we need to go back 2 rounds to reach $x^5_{2,0} \oplus x^5_{6,7} \oplus x^5_{6,19} \oplus x^5_{10,12} \oplus x^5_{14,0}$. In this case, using $\gamma = 0.34$ we found 160 PNBs (see List of PNBs 2) and we obtained $\varepsilon_a = 2^{-14.18}$. From that, we get an attack with data complexity of $2^{103.8}$ and time complexity $2^{256-160} \times 2^{103.8} + 2^{256-57} + 2^{96} = 2^{199.8}$ for $\alpha = 57$. We have to repeat this attack 2^7 times on average because of the transition probability from $\Delta^0_{in_{17}}$ to $\Delta^1_{out_{17}}$ when using the extension trick from [BLT20]. Thus, the final attack has data complexity of $2^{110.8}$ and time complexity $2^{206.8}$. We use the same 5-round differential-linear distintiguisher to mount an attack against ChaCha reduced to 7.25 rounds. In this attack, instead to go back 2 round, we need to go back 2.5 to reach $x^5_{2,0} \oplus x^5_{6,7} \oplus x^5_{6,19} \oplus x^5_{10,12} \oplus x^5_{14,0}$. Using $\gamma = 0.28$ we found 133 PNBs (see List of PNBs 3) and we obtained $\varepsilon_a = 2^{-16.85}$. From that, we get an attack with data complexity of $2^{115.34}$ and time complexity $2^{256-133} \times 2^{115.34} + 2^{256-25} + 2^{123} = 2^{231.34}$ for $\alpha = 25$. We have to repeat this attack 2^7 times on average because of the transition probability from $\Delta^0_{in_{17}}$ to $\Delta^1_{out_{17}}$ when using the extension trick from [BLT20]. Thus, the final attack has data complexity of $2^{122.34}$ and time complexity $2^{238.34}$. In an effort to achieve 7.5 rounds, we incrementally introduced additional components to the existing 7.25 rounds. The first component we added to every 0.25 quarter round was a modular operation, which yielded similar data and time complexities as the 7.25 rounds. However, when we introduced the XOR operation, we could not find any attack more effective than brute force in the key-recovery setting.

We combined DL Distinguisher 4 (on 4 rounds with correlation $\varepsilon_d = 2^{-39.8}$) with the PNB technique found in Subsection 2.4 to yield the best results in the key-recovery attack setting. Thus, considering $\left(\Delta^0_{14,6}\right) \xrightarrow[\mathcal{DL}]{} \left(x^4_{0,0}\right)$, to attack 7 rounds, we need to go back 3 rounds to reach $x^4_{0,0}$. In this case, using $\gamma = 0.27$ we found 166 PNBs (see List of PNBs 4) and we obtained $\varepsilon_a = 2^{-17.47}$. From that, we get an attack with data complexity of $2^{121.58}$ and time complexity $2^{90} \times 2^{121.58} + 2^{211} + 2^{166} = 2^{212.58}$ for $\alpha = 45$. As in [BLT20], we have to repeat this attack 2^5 times on average because of the transition probability from $\Delta^0_{14,6}$ to $\left(\Delta^1_{2,2}, \Delta^1_{6,5}, \Delta^1_{6,9}, \Delta^1_{6,17}, \Delta^1_{6,29}, \Delta^1_{10,10}, \Delta^1_{10,22}, \Delta^1_{10,30}, \Delta^1_{14,10}, \Delta^1_{14,30}\right)$. Thus, the final attack has data complexity of $2^{126.58}$ and time complexity $2^{217.58}$.

We combined DL Distinguisher 6 (on 4 rounds with correlation $\varepsilon_d = 2^{-37.98}$) with the PNB technique found in Subsection 2.4. Thus, considering $\left(\Delta^0_{15,22}\right) \xrightarrow[\mathcal{DL}]{} \left(x^4_{3,0}\right)$, to attack 7 rounds, we need to go back 3 rounds to reach $x^4_{3,0}$. In this case, using $\gamma = 0.265$ we found 170 PNBs (see List of PNBs 5) and we obtained $\varepsilon_a = 2^{-19.4}$. From that, we get an attack with data complexity of $2^{121.15}$ and time complexity $2^{86} \times 2^{121.15} + 2^{207.15} + 2^{170} = 2^{208.15}$ for $\alpha = 49$. As in [6], we have to repeat this attack 2^5 times on average because of the transition probability from $\Delta^0_{15,22}$ to $\left(\Delta^1_3[18], \Delta^1_7[1, 13, 21, 25], \Delta^1_{11}[6, 14, 26], \Delta^1_{15}[14, 26]\right)$. Thus, the final attack has data complexity of $2^{126.15}$ and time complexity $2^{213.15}$.

5 Conclusions and Future Work

In this work, we studied differential-linear cryptanalysis on ChaCha. We improved the best 7-round attacks presented in the literature on this primitive (both distinguisher and key-recovery). Also, to the best of our knowledge, we present the first distinguisher against ChaCha reduced to 7.5 rounds. These results were possible thanks to several new strategies: we explored DL distinguishers with 2 bits flipped at the beginning of the differential part, and we implemented a MILP model to automate the search for more effective linear masks in the linear part. For the middle part, we studied and optimised

the CUDA implementation presented in [Cou22] to verify our results. We believe that the aforementioned ideas can generally help to improve other cryptanalysis of ARX designs.

We expect that our key-recovery attack complexity has the potential to be further improved. Specifically, in future work, we will explore the decomposition of the key space of the input difference column in order to reduce the effort of the p^{-1} iterations discussed in Subsection 4.2. Also, we can explore the three-stage strategy presented in that paper to increase the number of PNBs and thus reduce the key-recovery attack complexities even more.

References

[AFK+08] Jean-Philippe Aumasson, Simon Fischer, Shahram Khazaei, Willi Meier, and Christian Rechberger. New Features of Latin Dances: Analysis of Salsa, ChaCha, and Rumba. In Kaisa Nyberg, editor, *Fast Software Encryption, 15th International Workshop, FSE 2008, Lausanne, Switzerland, February 10-13, 2008, Revised Selected Papers*, volume 5086 of *Lecture Notes in Computer Science*, pages 470–488. Springer, 2008.

[BBB22] Stefano Barbero, Danilo Bazzanella, and Emanuele Bellini. Rotational Cryptanalysis on ChaCha Stream Cipher. *Symmetry*, 14(6):1087, 2022.

[BBM21] Stefano Barbero, Emanuele Bellini, and Rusydi H. Makarim. Rotational analysis of ChaCha permutation. *Advances in Mathematics of Communications*, 0:–, 2021.

[BC04] Eli Biham and Rafi Chen. Near-collisions of SHA-0. In Matthew K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*, pages 290–305. Springer, 2004.

[BC14] Eli Biham and Yaniv Carmeli. An Improvement of Linear Cryptanalysis with Addition Operations with Applications to FEAL-8X. In Antoine Joux and Amr M. Youssef, editors, *Selected Areas in Cryptography - SAC 2014 - 21st International Conference, Montreal, QC, Canada, August 14-15, 2014, Revised Selected Papers*, volume 8781 of *Lecture Notes in Computer Science*, pages 59–76. Springer, 2014.

[BDKW19] Achiya Bar-On, Orr Dunkelman, Nathan Keller, and Ariel Weizman. DLCT: A New Tool for Differential-Linear cryptanalysis. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 313–342. Springer, 2019.

[Ber08a] D. J. Bernstein. ChaCha, a variant of Salsa20. In *Workshop Record of SASC*, volume 8, pages 3–5, 2008.

[Ber08b] Daniel J. Bernstein. ChaCha, a variant of Salsa20. https://cr.yp.to/papers.html#chacha, January 2008. Online; accessed 15 August 2022. Document ID: 4027b5256e17b9796842e6d0f68b0b5e.

[Ber08c] Daniel J. Bernstein. The Salsa20 Family of Stream Ciphers. In Matthew J. B. Robshaw and Olivier Billet, editors, *New Stream Cipher Designs - The*

eSTREAM Finalists, volume 4986 of *Lecture Notes in Computer Science*, pages 84–97. Springer, 2008.

[BGG+23] Emanuele Bellini, David Gerault, Juan Grados, Rusydi Makarim, and Thomas Peyrin. Fully Automated Differential-Linear Attacks against ARX Ciphers. Cryptology ePrint Archive, Paper 2023/181, 2023. https://eprint.iacr.org/2023/181.

[BLT20] Christof Beierle, Gregor Leander, and Yosuke Todo. Improved Differential-Linear Attacks with Applications to ARX Ciphers. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part III*, volume 12172 of *Lecture Notes in Computer Science*, pages 329–358. Springer, 2020.

[BSS+13] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. The SIMON and SPECK families of lightweight block ciphers. *IACR Cryptol. ePrint Arch.*, page 404, 2013.

[CM16a] Arka Rai Choudhuri and Subhamoy Maitra. Significantly Improved Multi-bit Differentials for Reduced Round Salsa and ChaCha. *IACR Trans. Symmetric Cryptol.*, 2016(2):261–287, 2016.

[CM16b] Arka Rai Choudhuri and Subhamoy Maitra. Significantly Improved Multi-bit Differentials for Reduced Round Salsa and ChaCha. *IACR Trans. Symmetric Cryptol.*, 2016(2):261–287, 2016.

[CN20] Murilo Coutinho and Tertuliano C. Souza Neto. New Multi-bit Differentials to Improve Attacks Against ChaCha. *IACR Cryptol. ePrint Arch.*, 2020:350, 2020.

[CN21a] Murilo Coutinho and T. C. Souza Neto. Improved Linear Approximations to ARX Ciphers and Attacks Against ChaCha. *IACR Cryptol. ePrint Arch.*, page 224, 2021. Revised Version: 2021-09-22.

[CN21b] Murilo Coutinho and Tertuliano C. Souza Neto. Improved Linear Approximations to ARX Ciphers and Attacks Against ChaCha. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I*, volume 12696 of *Lecture Notes in Computer Science*, pages 711–740. Springer, 2021.

[Cou22] Murilo Coutinho. Cryptanalysis ChaCha. https://github.com/MurCoutinho/cryptDances, 2022.

[CPV+22] Murilo Coutinho, Iago Passos, Juan C. Grados Vásquez, Fábio Lúcio Lopes de Mendonça, Rafael Timteo de Sousa, and Fábio Borges. Latin Dances Reloaded: Improved Cryptanalysis Against Salsa and Chacha, and the Proposal of Forró. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part I*, volume 13791 of *Lecture Notes in Computer Science*, pages 256–286. Springer, 2022.

[Cro05] Paul Crowley. Truncated differential cryptanalysis of five rounds of Salsa20. *IACR Cryptol. ePrint Arch.*, page 375, 2005.

[DDSM21] Sabyasachi Dey, Chandan Dey, Santanu Sarkar, and Willi Meier. Revisiting cryptanalysis on ChaCha from Crypto 2020 and Eurocrypt 2021. *IACR Cryptol. ePrint Arch.*, page 1059, 2021.

[DGSS22] Sabyasachi Dey, Hirendra Kumar Garai, Santanu Sarkar, and Nitin Kumar Sharma. Revamped Differential-Linear Cryptanalysis on Reduced Round ChaCha. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part III*, volume 13277 of *Lecture Notes in Computer Science*, pages 86–114. Springer, 2022.

[DS17] Sabyasachi Dey and Santanu Sarkar. Improved analysis for reduced round Salsa and Chacha. *Discret. Appl. Math.*, 227:58–69, 2017.

[DS20] Sabyasachi Dey and Santanu Sarkar. Proving the biases of Salsa and ChaCha in differential attack. *Des. Codes Cryptogr.*, 88(9):1827–1856, 2020.

[DS21] Sabyasachi Dey and Santanu Sarkar. A theoretical investigation on the distinguishers of Salsa and ChaCha. *Discret. Appl. Math.*, 302:147–162, 2021.

[ECR08] ECRYPT - European Network of Excellence for Cryptology. eSTREAM: the ECRYPT Stream Cipher Project. https://www.ecrypt.eu.org/stream/index.html, 2008. Online; accessed 15 August 2022.

[FMB+06] Simon Fischer, Willi Meier, Côme Berbain, Jean-François Biasse, and Matthew J. B. Robshaw. Non-randomness in eSTREAM Candidates Salsa20 and TSC-4. In Rana Barua and Tanja Lange, editors, *Progress in Cryptology - INDOCRYPT 2006, 7th International Conference on Cryptology in India, Kolkata, India, December 11-13, 2006, Proceedings*, volume 4329 of *Lecture Notes in Computer Science*, pages 2–16. Springer, 2006.

[FWG+16] Kai Fu, Meiqin Wang, Yinghua Guo, Siwei Sun, and Lei Hu. MILP-Based Automatic Search Algorithms for Differential and Linear Trails for Speck. In Thomas Peyrin, editor, *Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers*, volume 9783 of *Lecture Notes in Computer Science*, pages 268–288. Springer, 2016.

[Ian] Ianix. ChaCha Usage & Deployment . https://ianix.com/pub/chacha-deployment.html. [Online; accessed 1-Aug-2022].

[Leu16] Gaëtan Leurent. Improved Differential-Linear Cryptanalysis of 7-Round Chaskey with Partitioning. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 344–371. Springer, 2016.

[LH94] Susan K. Langford and Martin E. Hellman. Differential-Linear Cryptanalysis. In Yvo Desmedt, editor, *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*, volume 839 of *Lecture Notes in Computer Science*, pages 17–25. Springer, 1994.

[Mai16] Subhamoy Maitra. Chosen IV cryptanalysis on reduced round ChaCha and Salsa. *Discret. Appl. Math.*, 208:88–97, 2016.

[MIM21] Shotaro Miyashita, Ryoma Ito, and Atsuko Miyaji. Pnb-based differential cryptanalysis of chacha stream cipher. *IACR Cryptol. ePrint Arch.*, page 1537, 2021.

[MMH+14] Nicky Mouha, Bart Mennink, Anthony Van Herrewege, Dai Watanabe, Bart Preneel, and Ingrid Verbauwhede. Chaskey: An Efficient MAC Algorithm for 32-bit Microcontrollers. In Antoine Joux and Amr M. Youssef, editors, *Selected Areas in Cryptography - SAC 2014 - 21st International Conference, Montreal, QC, Canada, August 14-15, 2014, Revised Selected Papers*, volume 8781 of *Lecture Notes in Computer Science*, pages 306–323. Springer, 2014.

[MWGP11] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and Linear Cryptanalysis Using Mixed-Integer Linear Programming. In Chuankun Wu, Moti Yung, and Dongdai Lin, editors, *Information Security and Cryptology - 7th International Conference, Inscrypt 2011, Beijing, China, November 30 - December 3, 2011. Revised Selected Papers*, volume 7537 of *Lecture Notes in Computer Science*, pages 57–76. Springer, 2011.

[NIS15] NIST. Secure Hash Standard (SHS). Federal Information Processing Standards Publications 180-4, Information Technology Laboratory National Institute of Standards and Technology, Gaithersburg, MD 20899-8900, August 2015. Available at https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf.

[NSLL22a] Zhongfeng Niu, Siwei Sun, Yunwen Liu, and Chao Li. Practical (Rotational) Differential-Linear Distinguishers of ARX Ciphers with Arbitrary Output Linear Masks, 2022. https://github.com/ZhongfengNiu/rot-differential-linear.

[NSLL22b] Zhongfeng Niu, Siwei Sun, Yunwen Liu, and Chao Li. Rotational Differential-Linear Distinguishers of ARX Ciphers with Arbitrary Output Linear Masks. *IACR Cryptol. ePrint Arch.*, page 765, 2022.

[SZFW12] Zhenqing Shi, Bin Zhang, Dengguo Feng, and Wenling Wu. Improved Key Recovery Attacks on Reduced-Round Salsa20 and ChaCha. In Taekyoung Kwon, Mun-Kyu Lee, and Daesung Kwon, editors, *Information Security and Cryptology - ICISC 2012 - 15th International Conference, Seoul, Korea, November 28-30, 2012, Revised Selected Papers*, volume 7839 of *Lecture Notes in Computer Science*, pages 337–351. Springer, 2012.

[Tez21] Cihangir Tezcan. Optimization of Advanced Encryption Standard on Graphics Processing Units. *IEEE Access*, 9:67315–67326, 2021.

[TSK+07] Yukiyasu Tsunoo, Teruo Saito, Hiroyasu Kubo, Tomoyasu Suzaki, and Hiroki Nakashima. Differential cryptanalysis of Salsa20/8. In *Workshop Record of SASC*, volume 28, 2007.

[Wag99] David A. Wagner. The Boomerang Attack. In Lars R. Knudsen, editor, *Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings*, volume 1636 of *Lecture Notes in Computer Science*, pages 156–170. Springer, 1999.

[War12] Henry S. Warren. *Hacker's Delight*. Addison-Wesley Professional, 2nd edition, 2012.

[WCC21] Ziheng Wang, Heng Chen, and Weiling Cai. A hybrid CPU/GPU Scheme for Optimizing ChaCha20 Stream Cipher. In *2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom), New York City, NY, USA, September 30 - Oct. 3, 2021*, pages 1171–1178. IEEE, 2021.

[WN94] David J. Wheeler and Roger M. Needham. TEA, a Tiny Encryption Algorithm. In Bart Preneel, editor, *Fast Software Encryption: Second International Workshop. Leuven, Belgium, 14-16 December 1994, Proceedings*, volume 1008 of *Lecture Notes in Computer Science*, pages 363–366. Springer, 1994.

A Finding Linear Trails with MILP Solvers

Automated tools for finding differential and linear characteristics have become increasingly important helps for cryptanalysts and designers, as they allow to save a lot of time and reduce the possibility of mistakes when the modeling is not too complex. Different works exist for automatizing the search and among them the use of Mixed Integer Linear Programming (MILP) was the first to emerge [MWGP11]. Since then, many different improvements of MILP models for various ciphers have been proposed, often leading to improved cryptanalysis results. In this section, we describe a MILP model to find linear trails for `ChaCha`. The techniques used to build this model is based on the work from [FWG+16], where the model aimed at searching for linear trails for the `SPECK` cipher.

We modeled the components of the `ChaCha` stream cipher using inequalities and we describe in this section the inequalities used to model the linear behavior of each component.

Constraints of the XOR Operation. For every XOR operation with input masks $\alpha \in \mathbb{F}_2^n$ and $\beta \in \mathbb{F}_2^n$ and output mask $\gamma \in \mathbb{F}_2^n$, the constraints at bit level for j in $\{0, \cdots, n-1\}$ are

$$\alpha_j = \beta_j = \gamma_j.$$

Constraints of the Three-Forked Branch. For every three-forked branch with input masks $\alpha \in \mathbb{F}_2^n$ and $\beta \in \mathbb{F}_2^n$ and output mask $\gamma \in \mathbb{F}_2^n$, we used the following inequalities

$$\begin{aligned} d_{\mathcal{K}_j} &\geq \alpha_j \\ d_{\mathcal{K}_j} &\geq \beta_j \\ d_{\mathcal{K}_j} &\geq \gamma_j \\ \alpha_j + \beta_j + c_j &\geq 2d_{\mathcal{K}_j} \\ \alpha_j + \beta_j + c_j &\leq 2 \end{aligned} \tag{8}$$

where $d_{\mathcal{K}_j}$ is a dummy variable used to verify that there are at least two active terms in $a_j \oplus b_j = c_j$ whenever $a_j \neq 0$, $b_j \neq 0$, or $c_j \neq 0$.

Constraints of the Modular Addition. To model this component, we follow the transition state approach presented in [FWG+16]. Therefore, for every modular addition with input masks $\alpha \in \mathbb{F}_2^n$ and $\beta \in \mathbb{F}_2^n$ and output mask $\gamma \in \mathbb{F}_2^n$, the constraints at bit level for i in $\{0, \cdots, n-1\}$ are

$$\begin{aligned} s_i - \gamma_i - \alpha_i + \beta_i + s_{i+1} &\geq 0 \\ s_i + \gamma_i + \alpha_i - \beta_i - s_{i+1} &\geq 0 \\ s_i + \gamma_i - \alpha_i - \beta_i + s_{i+1} &\geq 0 \\ s_i - \gamma_i + \alpha_i - \beta_i + s_{i+1} &\geq 0 \\ s_i + \gamma_i - \alpha_i + \beta_i - s_{i+1} &\geq 0 \\ s_i - \gamma_i + \alpha_i + \beta_i - s_{i+1} &\geq 0 \\ \gamma_i - d_i + \alpha_i + \beta_i + s_{i+1} &\geq 0 \\ s_i + \gamma_i + \alpha_i + \beta_i + s_{i+1} &\leq 4 \\ s_{n-1} &= 0. \end{aligned} \tag{9}$$

where s_i is the probability weight variable [FWG+16].

Constraints for the Quarter Round. In the quarter round, auxiliary variables are also created for the half quarter round's output. However, because of the three-forked branch, we need new auxiliary variables Z_i for $1 \leq i \leq 12$. In Figure 1, we illustrate these new auxiliary variables and we can observe that the inequalities for the modular addition operations are the inequalities in Equation 9 where

$$(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \in \{(x_a^{m-1}, \mathbf{Z_6}, \mathbf{Z_1}), (x_c^{m-1}, \mathbf{Z_4}, \mathbf{Z_2}), (\mathbf{Z_8}, \mathbf{Z_3}, \mathbf{Z_9}), (\mathbf{Z_5}, \mathbf{Z_{11}}, \mathbf{Z_{12}})\}. \tag{10}$$

and the inequalities for the three-forked branch operations are the inequalities in Equation 8 where

$$\begin{aligned}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \in \{&(x_b^{m-1}, \mathbf{Z_6}, \mathbf{Z_7}), (\mathbf{Z_1}, x_d^{m-1}, \mathbf{Z_8}), (x_d^{m-1} \lll 16, \mathbf{Z_4}, \mathbf{Z_{10}}), (\mathbf{Z_2}, \mathbf{Z_7}, \mathbf{Z_5}),\\ &(\mathbf{Z_7} \lll 12, \mathbf{Z_3}, x_b^m \ggg 7), (\mathbf{Z_9}, \mathbf{Z_{10}}, x_a^m), (\mathbf{Z_{10}} \lll 8, \mathbf{Z_{11}}, x_d^m),\\ &(\mathbf{Z_{12}}, x_b^m \ggg 7, x_c^m)\}.\end{aligned}$$

Constraints for R Rounds. Considering that in each round the quarter round is applied four times, then we need to use the inequalities of quarter round (Equation 10) four times. Thus, for the entire R rounds we need $R \times 4$ quarter rounds. This gives us $16 \times n \times (R+1)$ variables to model the input and output masks of each quarter round; $36 \times n \times R$ variables for modeling d_λ of Equation 8; $13 \times 4 \times n \times R$ variables for modeling the intermediate outputs of the quarter rounds; $16 \times n \times R$ variables for modeling the variable d of Equation 9. Summing up, this gives us a total of $120nR + 16n$ variables. The number of inequalities are distributed as follows: $16 \times 10 \times ((n-1) \times R) + 16 \times R$ inequalities for the modular addition operations; $16 \times 40 \times n \times R$ inequalities for the three-forked branch operations. Summing up this gives us a total $800nR - 144R$ inequalities.

Objective Function. Let R be the number of rounds that we are modeling. Also, let $s_{i,j}^r$ be the variable representing the probability weight variable at round r and bit j of the i^{th} modular addition, then according to [FWG+16], we need to minimize the following expression

$$\sum_{r=1}^{R} \sum_{i=0}^{16} \sum_{j=0}^{n-1} s_{i,j}^r$$

B Some tables

The table below compares the experimental and theoretical correlations of all distinguishers discussed in this paper. The theoretical values for the DL distinguishers were obtained using a tool described in [NSLL22a], which uses the techniques presented in [NSLL22b]. However, we could not compute the theoretical values for DL Distinguisher 3 and DL Distinguisher 5 using this tool as it is limited to masks with only one active bit in the output mask, whereas DL Distinguisher 3 and DL Distinguisher 5 have more than one active bit in their output masks. Therefore, we consider the theoretical analysis of these distinguishers, using the techniques presented for output masks with multiple active bits in [NSLL22b], as a future task. The tool provided an output of 0 for DL Distinguisher 4 and DL Distinguisher 6.

Table 4: Differential trails for the first round of `ChaCha`. All of them have a probability of 2^{-7}. Candidates starting from $\Delta^0_{in_4}$ and from $\Delta^0_{in_{17}}$ are later used to improve 7-round state-of-the-art DL distinguishers on `ChaCha` and their 2^{-7} probabilities have been verified in practice.

Input difference	Output difference
$\Delta^0_{in_0} = (\Delta^0_{15,21}, \Delta^0_{15,9})$	$\Delta^1_{out_0} = (\Delta^1_{3,17}, \Delta^1_{3,5}, \Delta^1_{7,24}, \Delta^1_{7,8}, \Delta^1_{11,5}, \Delta^1_{11,1}, \Delta^1_{15,25}, \Delta^1_{15,1})$
$\Delta^0_{in_1} = (\Delta^0_{14,14}, \Delta^0_{14,2})$	$\Delta^1_{out_1} = (\Delta^1_{2,30}, \Delta^1_{2,10}, \Delta^1_{6,17}, \Delta^1_{6,1}, \Delta^1_{10,30}, \Delta^1_{10,26}, \Delta^1_{14,26}, \Delta^1_{14,18})$
$\Delta^0_{in_2} = (\Delta^0_{14,18}, \Delta^0_{14,6})$	$\Delta^1_{out_2} = (\Delta^1_{2,14}, \Delta^1_{2,2}, \Delta^1_{6,21}, \Delta^1_{6,5}, \Delta^1_{10,30}, \Delta^1_{10,2}, \Delta^1_{14,30}, \Delta^1_{14,22})$
$\Delta^0_{in_3} = (\Delta^0_{14,21}, \Delta^0_{14,1})$	$\Delta^1_{out_3} = (\Delta^1_{2,29}, \Delta^1_{2,17}, \Delta^1_{6,20}, \Delta^1_{6,4}, \Delta^1_{10,17}, \Delta^1_{10,13}, \Delta^1_{14,13}, \Delta^1_{14,5})$
$\Delta^0_{in_4} = (\Delta^0_{14,21}, \Delta^0_{14,9})$	$\Delta^1_{out_4} = (\Delta^1_{2,17}, \Delta^1_{2,5}, \Delta^1_{6,24}, \Delta^1_{6,8}, \Delta^1_{10,5}, \Delta^1_{10,1}, \Delta^1_{14,25}, \Delta^1_{14,1})$
$\Delta^0_{in_5} = (\Delta^0_{14,26}, \Delta^0_{14,14})$	$\Delta^1_{out_5} = (\Delta^1_{2,22}, \Delta^1_{2,10}, \Delta^1_{6,29}, \Delta^1_{6,13}, \Delta^1_{10,10}, \Delta^1_{10,6}, \Delta^1_{14,30}, \Delta^1_{14,6})$
$\Delta^0_{in_6} = (\Delta^0_{13,18}, \Delta^0_{13,6})$	$\Delta^1_{out_6} = (\Delta^1_{1,14}, \Delta^1_{1,2}, \Delta^1_{5,21}, \Delta^1_{5,5}, \Delta^1_{9,30}, \Delta^1_{9,2}, \Delta^1_{13,30}, \Delta^1_{13,22})$
$\Delta^0_{in_7} = (\Delta^0_{13,21}, \Delta^0_{13,1})$	$\Delta^1_{out_7} = (\Delta^1_{1,29}, \Delta^1_{1,17}, \Delta^1_{5,20}, \Delta^1_{5,4}, \Delta^1_{9,17}, \Delta^1_{9,13}, \Delta^1_{13,13}, \Delta^1_{13,5})$
$\Delta^0_{in_8} = (\Delta^0_{13,21}, \Delta^0_{13,9})$	$\Delta^1_{out_8} = (\Delta^1_{1,17}, \Delta^1_{1,5}, \Delta^1_{5,24}, \Delta^1_{5,8}, \Delta^1_{9,5}, \Delta^1_{9,1}, \Delta^1_{13,25}, \Delta^1_{13,1})$
$\Delta^0_{in_9} = (\Delta^0_{13,26}, \Delta^0_{13,14})$	$\Delta^1_{out_9} = (\Delta^1_{1,22}, \Delta^1_{1,10}, \Delta^1_{5,29}, \Delta^1_{5,13}, \Delta^1_{9,10}, \Delta^1_{9,6}, \Delta^1_{13,30}, \Delta^1_{13,6})$
$\Delta^0_{in_{10}} = (\Delta^0_{12,14}, \Delta^0_{12,2})$	$\Delta^1_{out_{10}} = (\Delta^1_{0,30}, \Delta^1_{0,10}, \Delta^1_{4,17}, \Delta^1_{4,1}, \Delta^1_{8,30}, \Delta^1_{8,26}, \Delta^1_{12,26}, \Delta^1_{12,18})$
$\Delta^0_{in_{11}} = (\Delta^0_{12,18}, \Delta^0_{12,6})$	$\Delta^1_{out_{11}} = (\Delta^1_{0,14}, \Delta^1_{0,2}, \Delta^1_{4,21}, \Delta^1_{4,5}, \Delta^1_{8,30}, \Delta^1_{8,2}, \Delta^1_{12,30}, \Delta^1_{12,22})$
$\Delta^0_{in_{12}} = (\Delta^0_{12,21}, \Delta^0_{12,1})$	$\Delta^1_{out_{12}} = (\Delta^1_{0,29}, \Delta^1_{0,17}, \Delta^1_{4,20}, \Delta^1_{4,4}, \Delta^1_{8,17}, \Delta^1_{8,13}, \Delta^1_{12,13}, \Delta^1_{12,5})$
$\Delta^0_{in_{13}} = (\Delta^0_{12,21}, \Delta^0_{12,9})$	$\Delta^1_{out_{13}} = (\Delta^1_{0,17}, \Delta^1_{0,5}, \Delta^1_{4,24}, \Delta^1_{4,8}, \Delta^1_{8,5}, \Delta^1_{8,1}, \Delta^1_{12,25}, \Delta^1_{12,1})$
$\Delta^0_{in_{14}} = (\Delta^0_{13,29}, \Delta^0_{13,9})$	$\Delta^1_{out_{14}} = (\Delta^1_{1,25}, \Delta^1_{1,5}, \Delta^1_{5,28}, \Delta^1_{5,12}, \Delta^1_{9,25}, \Delta^1_{9,21}, \Delta^1_{13,21}, \Delta^1_{13,13})$
$\Delta^0_{in_{15}} = (\Delta^0_{12,29}, \Delta^0_{12,9})$	$\Delta^1_{out_{15}} = (\Delta^1_{0,25}, \Delta^1_{0,5}, \Delta^1_{4,28}, \Delta^1_{4,12}, \Delta^1_{8,25}, \Delta^1_{8,21}, \Delta^1_{12,21}, \Delta^1_{12,13})$
$\Delta^0_{in_{16}} = (\Delta^0_{14,29}, \Delta^0_{14,9})$	$\Delta^1_{out_{16}} = (\Delta^1_{2,25}, \Delta^1_{2,5}, \Delta^1_{6,28}, \Delta^1_{6,12}, \Delta^1_{10,25}, \Delta^1_{10,21}, \Delta^1_{14,21}, \Delta^1_{14,13})$
$\Delta^0_{in_{17}} = (\Delta^0_{15,29}, \Delta^0_{15,9})$	$\Delta^1_{out_{17}} = (\Delta^1_{3,25}, \Delta^1_{3,5}, \Delta^1_{7,28}, \Delta^1_{7,12}, \Delta^1_{11,25}, \Delta^1_{11,21}, \Delta^1_{15,21}, \Delta^1_{15,13})$

Table 5: DL distinguishers for `ChaCha` starting after the first round and ending after round 3 or 3.5. The symbol (*) in this table means we used partitions to estimate the correlation of these distinguishers. In this case, we used up to 2^{34} random samples to verify the correlation of these partitions. Similar to (*), the symbol (†) in this table means we used up to 2^{58} random samples for every partition to construct the distinguisher.

$\mathcal{ID}$	Mask	Correlation (ε_d)	# Samples
$\Delta^1_{out_0}$	$x^{3.5}_{8,0}$	$2^{-18.84}$	2^{46}
$\Delta^1_{out_1}$	$x^{3.5}_{10,0}$	$2^{-20.64}$	2^{46}
$\Delta^1_{out_2}$	$x^{3.5}_{10,0}$	$2^{-20.55}$	2^{46}
$\Delta^1_{out_3}$	$x^{3.5}_{11,0}$	$2^{-20.58}$	2^{46}
$\Delta^1_{out_4}$	$x^{3.5}_{11,0}$	$2^{-18.75}$	2^{46}
$\Delta^1_{out_5}$	$x^{3.5}_{8,0}$	$2^{-20.91}$	2^{46}
$\Delta^1_{out_6}$	$x^{3.5}_{9,0}$	$2^{-20.57}$	2^{46}
$\Delta^1_{out_7}$	$x^{3.5}_{10,0}$	$2^{-20.36}$	2^{46}
$\Delta^1_{out_8}$	$x^{3.5}_{10,0}$	$2^{-18.88}$	2^{46}
$\Delta^1_{out_9}$	$x^{3.5}_{11,0}$	$2^{-20.89}$	2^{46}
$\Delta^1_{out_{10}}$	$x^{3.5}_{8,0}$	$2^{-20.99}$	2^{46}
$\Delta^1_{out_{11}}$	$x^{3.5}_{8,0}$	$2^{-20.22}$	2^{46}
$\Delta^1_{out_{12}}$	$x^{3.5}_{9,0}$	$2^{-20.48}$	2^{46}
$\Delta^1_{out_{13}}$	$x^{3.5}_{9,0}$	$2^{-18.75}$	2^{46}
$\Delta^1_{out_{14}}$	$x^{3.5}_{0,0} \oplus x^{3.5}_{5,0}$	$2^{-74.58}$	2^{34^*}
$\Delta^1_{out_{15}}$	$x^{3.5}_{3,0} \oplus x^{3.5}_{4,0}$	$2^{-74.58}$	2^{34^*}
$\Delta^1_{out_{16}}$	$x^{3.5}_{1,0} \oplus x^{3.5}_{6,0}$	$2^{-74.58}$	2^{34^*}
$\Delta^1_{out_{17}}$	$x^3_2[4,3,0] \oplus x^3_7[20,4,0] \oplus x^3_8[20,19] \oplus x^3_{13}[4]$	$2^{-30.15}$	$2^{58^\dagger}$

Table 6: Comparison of the experimental and theoretical correlations for all the distinguishers presented in this paper.

Distinguisher	Theoretical correlation	Experimental correlation	Samples	Observations
DL Distinguisher 1	$2^{-8.66}$	$2^{-18.75}$	2^{42}	Theoretical correlation computed using [NSLL22b]. Experimental one totally computed on CUDA
DL Distinguisher 2	2^{-45}	$2^{-30.15}$	2^{58}	Verified with the Piling-Up Lemma. Theoretical correlation found using [NSLL22b].
DL Distinguisher 3	-	$2^{-17.5}$	2^{42}	Experimental correlation verified on CUDA
DL Distinguisher 4	0	$2^{-39.8}$	-	Created from DL3 and LT5, and verified by assuming sub-cipher independence.
DL Distinguisher 5	-	$2^{-15.6}$	2^{40}	Experimental correlation verified on CUDA.
DL Distinguisher 6	0	$2^{-37.98}$	-	Created from DL5 and LT6, and verified by assuming sub-cipher independence.
Lemma 1	2^{-7}	$2^{-6.58}$	2^{37}	Experimental correlation verified on CUDA
Lemma 2	2^{-40}	$2^{-32.1}$	2^{40}	Verified using the Piling-Up Lemma
Lemma 3	2^{-6}	$2^{-3.58}$	2^{33}	Experimental correlation verified CUDA
Lemma 4	2^{-24}	$2^{-21.35}$	2^{41}	Verified using the Piling-Up Lemma
Linear Trail 1	2^{-47}	$2^{-38.69}$	-	Created from Lemma 1 and Lemma 2, and verified by assuming sub-cipher independence.
Linear Trail 2	2^{-30}	2^{-25}	-	Created from Lemma 3 and Lemma 4, and verified by assuming sub-cipher independence.
Linear Trail 3	2^{-37}	$2^{-34.6}$	2^{48}	Theoretical correlation verified using MILP. Experimental one computed assuming sub-cipher independence.
Linear Trail 4	2^{-26}	$2^{-23.11}$	2^{51}	Theoretical correlation computed using MILP. Experimental verified on CUDA.
Linear Trail 5	2^{-12}	$2^{-11.1}$	2^{48}	Experimental correlation verified CUDA
Linear Trail 6	2^{-12}	$2^{-11.19}$	2^{48}	Experimental correlation verified CUDA
Linear Trail 7	2^{-48}	$2^{-44.31}$	2^{48}	Experimental correlation computed assuming sub-cipher independence. Theoretical one using MILP

C Proof of some results

Proof of Lemma 1. *The following 2.5-round linear trail holds with probability* $\frac{1}{2}\left(1+\frac{1}{2^7}\right)$

$$\begin{aligned}x_{11}^{3.5}[0] = {} & x_0^6[8,24] \oplus x_1^6[0,8,24] \oplus x_2^6[0] \oplus x_3^6[0] \oplus x_4^6[7,19,26] \oplus x_5^6[2,3,14,15,19,22,23] \oplus \\ & x_6^6[30,31] \oplus x_8^6[0] \oplus x_9^6[6,7,12,19] \oplus x_{10}^6[0,7,8,12,27,28] \oplus x_{11}^6[0,23,24] \oplus \\ & x_{12}^6[0,8,16] \oplus x_{13}^6[0,8] \oplus x_{14}^6[0,7] \oplus x_{15}^6[14,16,24]\end{aligned}$$

Proof. In fact from Equation 1, we have

$$\begin{aligned}x_{11,0}^{3.5} &= x_{11,0}^4 \oplus x_{12,0}^4 \\ &= x_{0,0}^5 \oplus x_{0,16}^5 \oplus x_{3,0}^5 \oplus x_{4,7}^5 \oplus x_{8,0}^5 \oplus x_{11,0}^5 \oplus \\ &\quad x_{12,24}^5 \oplus x_{15,0}^5 \oplus x_{15,8}^5\end{aligned}$$

Using [CM16b] expressions, we have

$$\begin{aligned}x_{0,0}^5 &= x_{0,0}^6 \oplus x_{5,7}^6 \oplus x_{5,19}^6 \oplus x_{10,12}^6 \oplus x_{15,0}^6 \\ x_{3,0}^5 &= x_{3,0}^6 \oplus x_{4,7}^6 \oplus x_{4,19}^6 \oplus x_{9,12}^6 \oplus x_{14,0}^6 \\ x_{8,0}^5 &= x_{2,0}^6 \oplus x_{8,0}^6 \oplus x_{13,0}^6 \oplus x_{13,8}^6 \\ x_{11,0}^5 &= x_{1,0}^6 \oplus x_{11,0}^6 \oplus x_{12,0}^6 \oplus x_{12,8}^6 \\ x_{15,0}^5 &= x_{0,0}^6 \oplus x_{0,16}^6 \oplus x_{5,7}^6 \oplus x_{10,0}^6 \oplus x_{15,24}^6.\end{aligned}$$

And using Lemma 3 of [CN21b], we have

$$\begin{aligned}x_{0,16}^5 = {} & x_{0,0}^6 \oplus x_{0,16}^6 \oplus x_{5,2}^6 \oplus x_{5,3}^6 \oplus x_{5,7}^6 \oplus x_{5,19}^6 \oplus x_{5,22}^6 \oplus x_{5,23}^6 \\ & \oplus x_{10,12}^6 \oplus x_{10,27}^6 \oplus x_{10,28}^6 \oplus x_{15,0}^6 \oplus x_{15,14}^6 \oplus x_{15,16}^6 \quad w.p. \frac{1}{2}\left(1+\frac{1}{2^4}\right) \\ x_{4,7}^5 = {} & x_{4,26}^6 \oplus x_{9,7}^6 \oplus x_{9,19}^6 \oplus x_{14,6}^6 \oplus x_{14,7}^6 \quad w.p. \frac{1}{2}\left(1+\frac{1}{2}\right) \\ x_{12,24}^5 = {} & x_{1,8}^6 \oplus x_{1,24}^6 \oplus x_{6,30}^6 \oplus x_{6,31}^6 \oplus x_{11,23}^6 \oplus x_{11,24}^6 \oplus x_{12,16}^6 \quad w.p. \frac{1}{2}\left(1+\frac{1}{2}\right) \\ x_{15,8}^5 = {} & x_{0,8}^6 \oplus x_{0,24}^6 \oplus x_{5,14}^6 \oplus x_{5,15}^6 \oplus x_{10,7}^6 \oplus x_{10,8}^6 \oplus x_{15,0}^6 \quad w.p. \frac{1}{2}\left(1+\frac{1}{2}\right)\end{aligned}$$

Finally, using the Piling-Up Lemma, we obtain Equation 5. We verify this trail experimentally with 2^{37} samples, and get a $2^{-6.58}$ correlation. □

Proof of Lemma 2. *The following 1-round linear trail*

$$\mathcal{A} = \mathcal{B}$$

where

$$\begin{aligned}\mathcal{A} = {} & x_0^6[8,24] \oplus x_1^6[0,8,24] \oplus x_2^6[0] \oplus x_3^6[0] \oplus x_4^6[7,19,26] \oplus x_5^6[2,3,14,15,19,22,23] \oplus \\ & x_6^6[30,31] \oplus x_8^6[0] \oplus x_9^6[7,12,19] \oplus x_{10}^6[0,7,8,12,27,28] \oplus x_{11}^6[0,23,24] \oplus \\ & x_{12}^6[0,8,16] \oplus x_{13}^6[0,8] \oplus x_{14}^6[0,6,7] \oplus x_{15}^6[14,16,24],\end{aligned}$$

and

$$\begin{aligned}\mathcal{B} = {} & x_0^7[0] \oplus x_4^7[6,7,10,11,13,22,23,27,30,31] \oplus x_8^7[3,4,6,8,15,16,19,20,26,31] \\ & \oplus x_{12}^7[7,8,18,19,22,25,26] \oplus x_1^7[6,7,11,12,16,18,19] \\ & \oplus x_5^7[1,2,9,11,19,21,22,26,27,30,31] \oplus x_9^7[0,3,4,8,20,22,23,26,27] \\ & \oplus x_{13}^7[2,3,7,8,11,12,18,20,21,22,23,26,27] \oplus x_2^7[0,6,8,11,12,16,22,23,27,28] \\ & \oplus x_6^7[13,14,17,18,19] \oplus x_{10}^7[6,8,10,11,27,28,30,31] \\ & \oplus x_{14}^7[3,4,7,11,12,15,16,19,20,24,27,28] \oplus x_3^7[0,8,14,16,23,30] \\ & \oplus x_7^7[7,19,20,21,22,23,30,31] \oplus x_{11}^7[0,12,13,14,15,16] \oplus x_{15}^7[0,6,16,23,24,31]\end{aligned}$$

holds with probability $\frac{1}{2}\left(1+\frac{1}{2^{40}}\right)$.

Proof. Applying Lemma 3 of [CM16b] in $x_{14,0}^6, x_{13,0}^6, x_{8,0}^6, x_{11,0}^6, x_{2,0}^6, x_{3,0}^6, x_{12,0}^6, x_{1,0}^6$, and $x_{10,0}^6$, we obtain an expression A with probability one. Applying Lemma 3 of [CN21b] in

$$x_{0,8}^6, x_{0,24}^6, x_{1,24}^6, x_{4,26}^6, x_{4,7}^6, x_{9,12}^6, x_{12,16}^6, x_{13,8}^6, x_{5,19}^6, x_{10,12}x_{1,8}^6, x_{15,14}^6,$$

$x_{15,16}^6$, $x_{9,19}^6$, $x_{9,7}^6$,$x_{12,8}^6$, $x_{4,19}^6$, and $x_{15,24}^6$ we obtain an expression B with a correlation of 2^{-29}. Applying Lemma 7 of [CN21b] in

$$x_{14,6}^6x_{14,7}^6, x_{11,23}^6, x_{11,24}^6, x_{10,7}^6, x_{10,8}^6, x_{10,27}^6, x_{10,28}^6, x_{5,2}^6, x_{5,3}^6, x_{5,14}^6, x_{5,15}^6, x_{5,22}^6, x_{5,23}^6, x_{6,30}^6, x_{6,31}^6$$

we can obtain an expression C with correlation 2^{-11}. Finally, using the Piling-Up Lemma in A, B, and C, we obtain equation Equation 6. We verify this correlation experimentally. We used 2^{40} samples and divided them into four partitions, each corresponding to a linear trail associated with its respective quarter round. As these are 1-round trails, the partitions are independent of each other. By applying the Piling-Up Lemma, we obtained an experimental correlation of $2^{-32.1}$. □

Proof of Lemma 3. *The following 3-round linear trail*

$$\begin{aligned}\mathcal{A}' = {} & x_0^6[11,12] \oplus x_2^6[0] \oplus x_3^6[0,16] \oplus x_4^6[7] \oplus x_6^6[6,26] \oplus x_7^6[7,19] \oplus x_8^6[12] \oplus \\ & x_9^6[0] \oplus x_{10}^6[12] \oplus x_{11}^6[6,7,18,31] \oplus x_{12}^6[7,19] \oplus x_{13}^6[0] \oplus x_{14}^6[24] \\ & \oplus x_{15}^6[11,12,19,20]\end{aligned} \tag{11}$$

where $\mathcal{A}' = \left(x_2^3[4,3,0] \oplus x_7^3[20,4,0] \oplus x_8^3[20,19] \oplus x_{13}^3[4]\right)$ *holds with probability* $\frac{1}{2}\left(1+\frac{1}{2^6}\right)$.

Proof. Using equations (2) we have

$$x_{2,0}^{3.5} \oplus x_{7,0}^{3.5} = x_{2,0}^4.$$

Using Lemma 3 of [CM16b], we have

$$x_{2,0}^4 = x_{2,0}^5 \oplus x_{6,7}^5 \oplus x_{6,19}^5 \oplus x_{10,12}^5 \oplus x_{14,0}^5.$$

Applying Lemma 3 of [CM16b], we have

$$\begin{aligned}x_{2,0}^5 &= x_{2,0}^6 \oplus x_{7,7}^6 \oplus x_{7,19}^6 \oplus x_{8,12}^6 \oplus x_{13,0}^6 \\ x_{14,0}^5 &= x_{3,0}^6 \oplus x_{3,16}^6 \oplus x_{4,7}^6 \oplus x_{9,0}^6 \oplus x_{14,24}^6.\end{aligned}$$

Applying Lemma 3 [CN21b], we have

$$x^5_{6,7} = x^6_{6,26} \oplus x^6_{11,6} \oplus x^6_{11,7} \oplus x^6_{11,19} \oplus x^6_{12,7} \quad w.p. \frac{1}{2}\left(1+\frac{1}{2}\right)$$
$$x^5_{6,19} = x^6_{6,6} \oplus x^6_{11,18} \oplus x^6_{11,19} \oplus x^6_{11,31} \oplus x^6_{12,19} \quad w.p. \frac{1}{2}\left(1+\frac{1}{2}\right)$$
$$x^5_{10,12} = x^6_{0,11} \oplus x^6_{0,12} \oplus x^6_{10,12} \oplus x^6_{15,11} \oplus x^6_{15,12} \oplus x^6_{15,19} \oplus x^6_{15,20} \quad w.p. \frac{1}{2}\left(1+\frac{1}{2^2}\right).$$

Finally, using Equation 4 and the Piling-Up Lemma, we finalize the proof. □

Proof of Lemma 4. *The following 1-round linear trail*

$$\mathcal{C} = \mathcal{D} \tag{12}$$

where

$$\begin{aligned}\mathcal{C} = {} & x^6_0[11,12] \oplus x^6_2[0] \oplus x^6_3[0,16] \oplus x^6_4[7] \oplus x^6_6[6,26] \oplus \\ & x^6_7[7,19] \oplus x^6_8[12] \oplus x^6_9[0] \oplus x^6_{10}[12] \oplus x^6_{11}[7,31,6,18] \oplus \\ & x^6_{12}[7,19] \oplus x^6_{13}[0] \oplus x^6_{14}[24] \oplus x^6_{15}[11,12,19,20]\end{aligned}$$

and

$$\begin{aligned}\mathcal{D} = {} & x^7_0[3,7,19,23] \oplus x^7_4[13,14,18,19,25,30,31] \oplus x^7_8[6,12,18,23,24] \\ & \oplus x^7_{12}[6,7,10,19,20,31] \oplus x^7_1[16] \oplus x^7_5[7] \oplus x^7_{13}[0,8,24] \oplus x^7_2[0,8,11,12,24] \\ & \oplus x^7_6[7,13,19,25,30,31] \oplus x^7_{10}[18,23,24,26] \oplus x^7_{14}[0,5,6,11,12,16,19,20,25,26] \\ & \oplus x^7_3[0,3,4,6,7,11,12,16,17,18,19,20,27,28,30,31] \oplus x^7_7[2,3,6,7,18,22,23,27] \\ & \oplus x^7_{11}[6,11,18,19,20,27,28] \oplus x^7_{15}[0,3,4,5,7,11,12,14,16,18,19,25,26,30,31]\end{aligned}$$

holds with probability $\frac{1}{2}\left(1+\frac{1}{2^{24}}\right)$.

Proof. Using Lemma 3 of [CM16b] in $x^6_{9,0}, x^6_{13,0}, x^6_{2,0}$ and $x^6_{3,0}$ we obtain a expression A with probability 2^{-1}. Applying Lemma 3 of [CN21b] in $x^6_{11,31}$, $x^6_{14,24}$, $x^6_{12,7}$, $x^6_{11,18}$, $x^6_{12,19}$, $x^6_{3,16}$, $x^6_{4,7}$, $x^6_{6,6}$, $x^6_{7,7}$, $x^6_{7,19}$, $x^6_{8,12}$, and $x^6_{6.26}$, we obtain a expression B with probability 2^{-17}. Applying Lemma 7 of [CN21b] in $x^6_{11,6} \oplus x^6_{11,7}$ and $x^6_{15,19} \oplus x^6_{15,20}$, we can obtain an expression C with probability 2^{-3}. And applying the equation 30 of Lemma 9 of [CN21b] in $x^6_{15,11}$, $x^6_{15,12}$, $x^6_{10,12}$, $x^6_{0,11}$, and $x^6_{0,12}$, we can obtain an expression D with probability of 2^{-3}

Finally, using the Piling-Up Lemma in A, B, C and D we obtain Equation 7. We verify this correlation experimentally. We used 2^{40} samples and divided them into four partitions, each corresponding to the linear trail associated with its respective quarter round. As these are 1-round trails, the partitions are independent of each other. By applying the Piling-Up Lemma, we obtained an experimental correlation of $2^{-21.35}$ □

D PNBs

List of PNBs 1. *3, 4, 7, 15, 16, 17, 18, 23, 27, 31, 34, 35, 36, 38, 43, 47, 48, 49, 51, 54, 55, 59, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 78, 79, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 119, 120, 121, 122, 123, 124, 127, 128, 136, 137, 138, 139, 144, 148, 156, 157, 159, 168, 169, 175, 184, 187, 188, 191, 192, 193, 194, 204, 208, 209, 210, 211, 212, 213, 214, 215, 216, 220, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 240, 241, 242, 243, 248, 249, 250, 251, 252, 253, 254, 255*

List of PNBs 2. *0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 14, 19, 20, 21, 26, 31, 32, 33, 34, 35, 36, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 71, 72, 73, 74, 77, 78, 79, 80, 83, 84, 85, 86, 89, 90, 91, 95, 99, 100, 103, 104, 105, 106, 107, 108, 109, 110, 115, 123, 124, 125, 126, 127, 128, 129, 130, 140, 141, 142, 147, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 198, 199, 200, 204, 205, 206, 207, 210, 211, 212, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 231, 244, 245, 246, 247, 248, 255*

List of PNBs 3. *0, 7, 20, 21, 31, 35, 39, 44, 45, 46, 47, 51, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 71, 72, 73, 74, 77, 80, 83, 84, 85, 86, 89, 90, 91, 95, 99, 100, 104, 108, 109, 110, 115, 123, 124, 125, 126, 127, 128, 129, 130, 135, 140, 141, 142, 147, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 198, 199, 200, 201, 204, 205, 206, 207, 210, 211, 212, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 231, 244, 245, 246, 247, 248, 252, 255*

List of PNBs 4. *0, 1, 2, 3, 4, 7, 8, 9, 10, 13, 14, 15, 16, 19, 20, 21, 22, 25, 26, 27, 31, 35, 36, 39, 40, 41, 42, 43, 44, 45, 46, 47, 51, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 78, 79, 83, 84, 85, 90, 95, 96, 97, 98, 99, 100, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 134, 135, 136, 137, 140, 141, 142, 143, 146, 147, 148, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 167, 180, 181, 182, 183, 184, 188, 191, 192, 193, 194, 199, 204, 205, 206, 211, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255*

List of PNBs 5. *3, 4, 7, 8, 9, 10, 11, 12, 13, 14, 19, 23, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 46, 47, 51, 52, 53, 58, 59, 63, 64, 65, 66, 67, 68, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 103, 104, 105, 106, 109, 110, 111, 112, 115, 116, 117, 118, 119, 121, 122, 123, 127, 128, 129, 130, 131, 135, 148, 149, 150, 151, 152, 156, 159, 160, 161, 162, 167, 172, 173, 174, 179, 180, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 230, 231, 232, 233, 236, 237, 238, 239, 240, 242, 243, 244, 250, 251, 252, 253, 254, 255*