IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2025, No. 3, pp. 729–754. DOI:10.46586/tosc.v2025.i3.729-754

# Cryptanalysis: Theory Versus Practice

## Correcting Cryptanalysis Results on `Ascon`, `ChaCha`, and `Serpent` Using GPUs

Cihangir Tezcan[1,2], Gregor Leander[3] and Hosein Hadipour[3]

[1] Department of Cyber Security, Graduate School of Informatics, Middle East Technical University, 06800 Ankara, Turkey
cihangir@metu.edu.tr
[2] Department of Computer Science and Engineering, Sabanci University, 34956 Istanbul, Turkey
[3] Ruhr Bochum University, Bochum, Germany
gregor.leander@rub.de, Hossein.Hadipour@ruhr-uni-bochum.de

**Abstract.** Most modern cryptanalysis results are obtained through theoretical analysis, often relying on simplifications and idealized assumptions. In this work, we use the parallel computational power of GPUs to experimentally verify a small portion of the cryptanalysis results that have been published in recent years. Our focus is on the ciphers `Ascon`, `ChaCha`, and `Serpent`. In none of the attacks we considered did the theoretical estimates fully match the actual practical values. More precisely, we show that the 4.5-round truncated differential with probability one, the 6-round differential-linear (DL), and the 6-round impossible differential distinguishers on `Ascon`, as well as the best known 7- and 7.5-round DL distinguisher on `ChaCha`, do not actually work in practice. Moreover, we demonstrate that the best known 10, 11, and 12-round DL attacks on `Serpent` perform better in practice than previously estimated. Additionally, we provide a new experimentally obtained 9-round DL distinguisher on `Serpent`, which can be used in 10 and 11-round attacks with reduced data complexity. In a broader sense, we recommend that cryptanalysts experimentally verify reduced versions of their theoretically obtained analysis results whenever possible. In order to simplify this process, we make our optimized code for the ciphers treated here available for future use.

**Keywords:** cryptanalysis · GPU · ASCON · ChaCha · SERPENT

## 1 Introduction

The cryptography community's advancements in designing secure symmetric encryption algorithms have made it difficult for cryptanalysts to achieve practical attacks. As a result, most attacks are now theoretical, targeting a reduced number of rounds with time, data, or memory complexities that exceed the limits of current technology.

Since many theoretically obtained distinguishers or attacks have not been verified through experiments, their results might differ in practice due to overlooked properties of the cipher. We even claim that many theoretically obtained cryptanalysis results do not work in practice, while some perform better than expected. Specifically, the time, data, and memory complexities of an attack might sometimes be better in practice than theoretically estimated. This can generally occur due to the following three reasons:

1. Human errors: Implementation errors, misreporting of obtained results, or misinterpretation of experimental findings can mislead the researcher and lead to incorrect conclusions. In particular, since many attack steps are derived theoretically, important details can be overlooked. For example, in [DIK08], the authors add one


Received: 2024-11-22 Revised: 2025-06-01 Accepted: 2025-08-01 Published: 2025-09-25

round to the bottom and two rounds to the top of a 9-round differential-linear (DL) distinguisher to attack 12 rounds of `Serpent`. They guess 112 out of 128 bits of the first-round key because only 12 out of 16 S-boxes are active in their attack. However, the remaining 16 bits of the round key must also be guessed to compute the next-round values, which are needed to carry out the attack. As a result, the time complexity of their attack is underestimated by a factor of $2^{16}$. Once corrected, the time complexity exceeds that of a brute-force key search. Thus, the 12-round attack proposed in [DIK08] and its subsequent improvements in [Tez15] and [Lu15] do not work in practice.

2. Differences between theoretically calculated probabilities and actual probabilities: Theoretically obtained distinguishers generally focus on a single characteristic (path), but the actual probabilities can differ significantly due to the clustering effect and fixed-key versus average-key behavior [BR22]. Additionally, whether in the fixed-key or average-key case, it is often computationally infeasible to examine every possible path and to compute the correct probabilities theoretically. Moreover, when two different cryptanalysis techniques are combined, e.g., in boomerang and DL attacks, their interaction may have a significantly positive or negative impact [Mur11, HNE22, HDE24]. For instance, it was observed in [DEM15] that the 4-round DL distinguisher for `Ascon`, which has a theoretical bias of $2^{-20}$, actually exhibits a bias of $2^{-2}$ in practice. As another example, Beyne and Neyt [BN24] proved that the main differential characteristic used in the full-round attack on `SPEEDY-7-192` at EUROCRYPT 2023 [BDBN23] does not hold for any key in practice.

3. Unknown properties: Many assumptions that are generally valid may fail in specific cases due to the inner workings of a cipher. For instance, in a differential attack on a block cipher, partial round keys are guessed in the added rounds, and the counter for a guessed key is incremented when the input and output differences of the distinguisher are observed. Here, it is assumed that the wrong keys behave like random permutations. This is known as the *Wrong Key Randomization Hypothesis*, and it holds true for most attacks. However, this hypothesis becomes invalid when a special property of the cipher prevents it.

   For example, if an S-box has a differential factor [TÖ14] $\lambda$ for the input difference $\alpha$ and output difference $\beta$, then for any round key bits $k$ just before the S-box operation that lead to these $\alpha$ and $\beta$ differences, $k \oplus \lambda$ will also satisfy the same differences. As a result, the attacker cannot distinguish $k$ from $k \oplus \lambda$. For a single differential factor, half of the round key bits behave identically to the other half in differential attacks. Thus, regardless of the number of pairs used, at least one wrong key will always have the same counter as the correct key, preventing the attacker from distinguishing the correct key.

It should be noted that many of these three reasons for the difference between theory and practice could be addressed by performing experiments on reduced versions of the theoretically obtained distinguisher or attack.

## Our Contribution

In this work, we use the parallel computing power of GPUs to experimentally verify cryptanalysis results that were obtained through theoretical analysis on the `Ascon`, `ChaCha`, and `Serpent` ciphers.

We optimized the implementation of `Ascon`, `ChaCha`, and `Serpent` ciphers using the CUDA programming language. Our GPU implementations allowed us to perform $2^{35.10}$ `Ascon` initializations, $2^{34.92}$ `ChaCha` encryptions, or $2^{34.70}$ `Serpent` encryptions per second

on a single NVIDIA RTX 4090 GPU. Note that the fastest implementations of symmetric-key encryption algorithms typically achieve between $2^{33}$ and $2^{37}$ encryptions per second on an RTX 4090 [TL25]. We made all of our source codes publicly available to support verification and future research. We used these optimized implementations to experimentally verify results obtained through theoretical cryptanalysis. Below, we summarize our key observations:

1. **Observations for `Ascon`.** The first claimed 6-round DL distinguisher for `Ascon` [PZWD24a], presented at CRYPTO 2024, with a claimed bias of $2^{-22.43}$, does not work in practice. We ran our experiments using $2^{54}$ data, which is enough to detect a bias of $2^{-22.43}$, but we observed no deviation from random behavior. We contacted the authors of [PZWD24a], and they confirmed that they also noticed some potential issues with their distinguisher. This observation suggests that the second method for calculating DL biases in [PZWD24a] (Section 4.3) lacks accuracy, and that the actual biases can be significantly smaller in practice. Additionally, the first claimed 6-round impossible differential distinguisher for `Ascon` and the 4.5-round truncated differential with probability one, both presented in [BJKK24], should not be considered to hold with probability zero and probability one, respectively. Because these distinguishers obtain valid input pairs by performing partial round operations using the fact that the `Ascon` permutation does not have a key addition layer. However, having no key addition layer allows us to obtain arbitrarily long distinguishers for permutations. Thus, we suggest these kinds of distinguishers to be treated as a different set of distinguishers and should not be compared with the traditional ones.

2. **Observations for `Serpent`.** We examined the improved DL distinguishers presented for up to 9 rounds of `Serpent` at CRYPTO 2024 by [PZWD24a]. While the authors experimentally verified their results for up to 4 rounds, as well as one of their 5-round distinguishers, they did not check the correctness of their longer distinguishers. We observed that their 6, 7, 8, and 9-round, and 6 out of 7 of their 5-round DL distinguishers for `Serpent`, contain errors in the reported output masks and do not work in practice. We contacted the authors, and they confirmed the issue and replied that the errors occurred during the write-up. They provided us with the corrected versions of these distinguishers, claiming the same reported biases as in [PZWD24a]. We include some of these corrected distinguishers in this paper and report our experimental results. For example, we observed that the corrected 6-round DL distinguisher with a reported bias of $2^{-19.61}$ does work in practice, but with a measured bias of $2^{-21.55}$. Re-contacting the authors allowed them to re-correct their 6-round DL distinguisher which we measured its bias as $2^{-19.62}$.

   We examined the improved DL distinguishers for `Serpent` by Hadipour et al. [HDE24] presented at CRYPTO 2024. The authors of [HDE24] proposed 3 to 9-round DL distinguishers using a boomerang-like technique. They experimentally verified the 3, 4, and 5-round distinguishers. However, due to limited CPU resources, they could not verify the 6-round distinguisher, which has a theoretical bias of $2^{-21.58}$. We verified the 6-round distinguisher experimentally and confirmed that it works in practice. The observed bias was slightly different from the theoretical value, being off by $2^{-0.25}$. This difference is small and marginally affects the data and time complexities in practice, increasing them by a factor of $2^{0.5}$.

   We experimentally verified the DL distinguishers of [BDK03] and [DIK08] on `Serpent`. In [DIK08], the authors could only run experiments on 4 rounds of their 9-round DL distinguisher due to limited CPU resources. They claimed that the distinguisher achieves a bias better than the theoretical value by a factor of $2^{1.25}$. We repeated the experiments on 5 and 6 rounds and found that the actual bias is better by a factor of $2^{1.39}$ instead of $2^{1.25}$.

By rotating the input difference of the DL distinguisher proposed in [BDK03], we obtained a stronger 9-round distinguisher. It achieves an experimental bias of $2^{-24.33}$ compared to the theoretical bias of $2^{-27}$ for its first six rounds. This corresponds to a gain by a factor of $2^{2.67}$, which exceeds the factor of $2^{1.39}$ that we obtained for [DIK08]. Our improved distinguisher can be used to attack 10 and 11 rounds of `Serpent` with better data complexity than previous attacks.

We show that the 12-round DL attack on `Serpent` proposed in [DIK08], as well as its subsequent improvements such as [Lu15] and [Tez15], actually require a time complexity that is no better than exhaustive search. Moreover, we show that the 12-round DL attacks proposed in [BCD+22] and [LLL21] offer better performance in terms of data, time, and memory complexities by a factor of at least $2^{0.28}$ for [BCD+22] and $2^{0.252}$ for [LLL21].

3. **Observations for `ChaCha`.** The best 7-round DL distinguisher on `ChaCha`, proposed in [WGM24], is not correct. In [WGM24], the authors claim to have experimentally verified their distinguisher using CPU computations. However, what they actually observe is consistent with random behavior. We further tested 128 additional variants of this 7-round DL distinguisher, each having a single-bit input difference, and did not observe any non-random behavior in any of these cases. Therefore, the distinguisher proposed in [WGM24], along with the attacks based on it, does not work in practice. The same error persists in the 7 and 7.5-round DL distinguisher of [OWGM25], which is the full version of [WGM24].

We provide two tables to summarize our findings. Table 1 summarizes the distinguishers and attacks that we invalidate in this work while Table 2 lists the DL distinguishers that we experimentally showed to behave differently in practice than predicted by theory.

Table 1: The list of distinguishers and attacks that we invalidated in this work.

| Cipher | Type | Rounds | Reference | Reason |
|---|---|---|---|---|
| `Ascon` | Truncated Differential Distinguisher | 4.5 | [BJKK24] | Wrong assumptions |
| `Ascon` | DL Distinguisher | 6 | [PZWD24a] | Wrong assumptions |
| `Ascon` | Impossible Differential Distinguisher | 6 | [BJKK24] | Wrong assumptions |
| `ChaCha` | DL Distinguisher | 7 | [WGM24] | Human error |
| `ChaCha` | DL Distinguisher | 7, 7.5 | [OWGM25] | Human error |
| `Serpent` | DL Distinguisher | 6, 7, 8, 9 | [PZWD24a] | Human error |
| `Serpent` | DL Attack | 12 | [Lu15] | Wrong assumptions |
| `Serpent` | DL Attack | 12 | [Tez15] | Wrong assumptions |

In all our experiments for DL distinguishers, we used approximately $c \times q^{-2}$ plaintext–ciphertext pairs for a target bias $q$, with $c$ ranging from $2^3$ to $2^{20}$ depending on our computational limits. We repeated our experiments for permutations 100 times with randomly generated data and for the cases when a secret key is required, we repeated the experiments with 100 to 1000 random keys to ensure that our observations are not due to some specific weak keys.

Finally, and more generally, we recommend that cryptanalysts experimentally verify reduced versions of their theoretically obtained results whenever possible, to ensure that the theoretical analysis holds in practice, and to publicly share their code. To facilitate this process for `Ascon`[1], `ChaCha`[2], and `Serpent`[3], we have made our optimized GPU implementations publicly available for future use.

[1] https://github.com/cihangirtezcan/CUDA_ASCON_DL
[2] https://github.com/cihangirtezcan/CUDA_CHACHA
[3] https://github.com/cihangirtezcan/CUDA_SERPENT

Table 2: The list of DL distinguishers which we experimentally showed in this work that work differently in practice.

| Cipher | Rounds | Reference | Data | Reported | Experimental | Gain |
|---|---|---|---|---|---|---|
| Ascon | 5 | [PZWD24a] | $2^{40}$ | $2^{-10.10}$ | $2^{-9.94}$ | $2^{0.16}$ |
| Serpent | 4 | [DIK08] | $2^{50}$ | $2^{-15}$ | $2^{-13.73}$ | $2^{1.27}$ |
| Serpent | 4 | Sec. 3.3.1 | $2^{50}$ | $2^{-15}$ | $2^{-12.33}$ | $2^{2.67}$ |
| Serpent | 5 | [DIK08] | $2^{50}$ | $2^{-19}$ | $2^{-17.63}$ | $2^{1.37}$ |
| Serpent | 5 | Sec. 3.3.1 | $2^{50}$ | $2^{-19}$ | $2^{-16.33}$ | $2^{2.67}$ |
| Serpent | 6 | [DIK08] | $2^{50}$ | $2^{-27}$ | $2^{-25.61}$ | $2^{1.39}$ |
| Serpent | 6 | Sec. 3.3.1 | $2^{50}$ | $2^{-27}$ | $2^{-24.33}$ | $2^{2.67}$ |
| Serpent | 6 | [HDE24] | $2^{50}$ | $2^{-21.58}$ | $2^{-21.83}$ | $2^{-0.25}$ |
| Serpent | 6 | [PZWD24a] | $2^{54}$ | $2^{-19.61}$ | $2^{-19.62}$ | $2^{-0.01}$ |

## 2 Preliminaries

Modern GPUs operate based on the single instruction, multiple threads (SIMT) execution model, which allows them to run many threads in parallel. They typically contain thousands of lightweight cores designed for high-throughput parallel computations. However, these cores are not as powerful as CPU cores, and due to architectural constraints, one must carefully optimize the implementation to fully utilize the GPU.

When a GPU kernel is launched, threads are organized into blocks, and the total number of threads often exceeds the number of physical cores by a large margin. As a result, the kernel launch configuration, namely the number of blocks and threads per block, has a significant effect on performance. This SIMT-level parallelism is particularly suitable for verifying distinguishers in symmetric-key cryptanalysis, since each thread can independently perform encryption for a fixed number of rounds with different inputs.

In our implementations, we applied several optimization techniques based on the internal structure of the cipher and the hardware characteristics of the GPU. Our aim was to maximize resource utilization and achieve the highest possible throughput in our experimental cryptanalysis.

In this work, we used an NVIDIA RTX 4090 GPU, which features 16,384 cores and a maximum boost clock speed of 2,550 MHz. These specifications may vary slightly across different RTX 4090 models, depending on the manufacturer and version. This GPU is based on the *Ada Lovelace* architecture and supports compute capability 8.9.

The compute capability of a CUDA device determines the range of hardware and software features it supports. CUDA devices are backward compatible with respect to compute capability, allowing code written for earlier versions to run on newer hardware.

We also tested all of our GPU implementations on a variety of desktop and mobile GPUs with compute capabilities ranging from 5.0 to 8.9. Our results showed that the optimizations remained effective across all tested architectures and were not tailored to a specific GPU. However, for simplicity and consistency, we report performance numbers only for the fastest GPU used in this study, the RTX 4090.

Finally, when we provide distinguishers in this work, we represent the input difference with $I$, output difference or mask of the substitution layer with $S_i$, and the output difference or mask of the permutation layer with $P_i$ for the $i$-th round.

### 2.1 Ascon Family Of Authenticated Encryption And Hashing Algorithms

Ascon is a family of authenticated encryption and hashing algorithms designed by the Ascon team [DEMS21]. It was the primary choice in the lightweight applications category

of the CAESAR competition [Ber13]. On 23 February 2023, `Ascon` was also selected as the winner of NIST's Lightweight Cryptography competition. Although the final version of NIST's standardization document is not yet published, an initial public draft has recently been released [TMC+24].

`Ascon` authenticated encryption with associated data scheme (a.k.a. `Ascon`-AEAD) is based on a sponge construction and uses a permutation referred to as `Ascon`-$p$ to update its internal state. Figure 1 shows an overview of the `Ascon`-AEAD. As seen in Figure 1, the `Ascon`-AEAD operates in four main steps: initialization, processing of associated data, processing of plaintext, and finalization. `Ascon`-AEAD has two instances, named `Ascon-128` and `Ascon-128a`, which differ in their data block sizes and the number of rounds. All versions of `Ascon`-AEAD use a 320-bit internal state, represented by five 64-bit words $x_0, x_1, \ldots, x_4$. During initialization, a 64-bit initialization vector (IV), a 128-bit secret key, and a 128-bit nonce are combined to form the 320-bit state. The number of rounds applied in the permutation varies depending on the version of `Ascon`-AEAD and the step of the algorithm. The permutation is applied $b = 6$ or $b = 8$ times during the associated data and plaintext processing steps, and $a = 12$ times during the initialization and finalization steps (see Figure 3a). The nonce in `Ascon`-AEAD must be unique for each encryption, because the mode of operation relies on the MonkeyDuplex structure [BDPVA12].

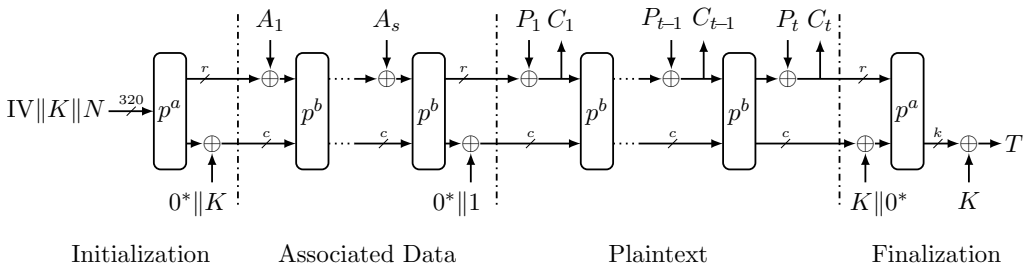
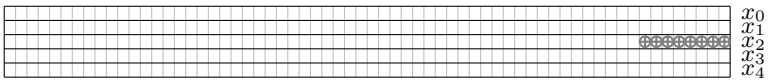
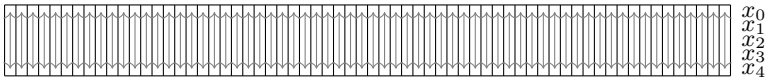



Figure 1: The overview of `Ascon`-AEAD

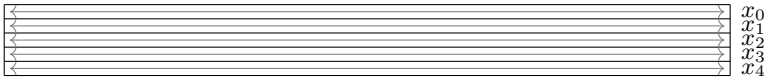
The `Ascon`-$p$ follows the substitution-permutation network (SPN) design paradigm. Each SPN-based round transformation $p$ consists of three steps: $p = p_L \circ p_S \circ p_C$. The 320-bit state $S$ of `Ascon` is divided into five 64-bit register words $x_i$, such that $S = x_0\|x_1\|x_2\|x_3\|x_4$ (see Figure 2).




(a) Round constant addition $p_C$




(b) Substitution layer $p_S$ with 5-bit S-box $\mathcal{S}(x)$

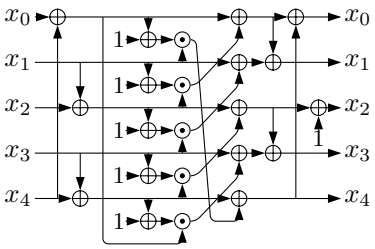



(c) Linear layer $p_L$ with 64-bit diffusion functions $\Sigma_i(x_i)$

Figure 2: The register words of the 320-bit state $S$ and operations $p_L \circ p_S \circ p_C$.

The substitution layer $p_S$ updates the state $S$ with 64 parallel applications of the 5-bit S-box $\mathcal{S}(x)$ defined in Figure 3a. It applies the S-box to each bit-slice across the

five registers $x_0, \ldots, x_4$. The $5 \times 5$ S-box of `Ascon` can be implemented using only 22 logical operations, as illustrated in Figure 3a. The linear diffusion layer $p_L$ applies a linear function $\Sigma_i(x_i)$ defined in Figure 3b to each word $x_i$.




(a) 5-bit S-box $\mathcal{S}(x)$

$$x_0 \leftarrow \Sigma_0(x_0) = x_0 \oplus (x_0 \ggg 19) \oplus (x_0 \ggg 28)$$
$$x_1 \leftarrow \Sigma_1(x_1) = x_1 \oplus (x_1 \ggg 61) \oplus (x_1 \ggg 39)$$
$$x_2 \leftarrow \Sigma_2(x_2) = x_2 \oplus (x_2 \ggg 1) \oplus (x_2 \ggg 6)$$
$$x_3 \leftarrow \Sigma_3(x_3) = x_3 \oplus (x_3 \ggg 10) \oplus (x_3 \ggg 17)$$
$$x_4 \leftarrow \Sigma_4(x_4) = x_4 \oplus (x_4 \ggg 7) \oplus (x_4 \ggg 41)$$

(b) Linear layer with 64-bit functions $\Sigma_i(x_i)$

Figure 3: `Ascon`'s substitution layer and linear diffusion layer.

Note that the recently published initial public draft by NIST [TMC+24] specifies the `Ascon-Aead128` algorithm, which is based on `Ascon-128a`. `Ascon` has been studied extensively since 2014, and a summary of existing analyses is available on `Ascon`'s official website[4].

In our GPU optimizations, we used the bitsliced implementation of `Ascon`'s S-box, as shown in Figure 3a. This approach allowed us to perform 64 S-box operations using only 22 logical operations, such as AND, NOT, and XOR, with 5 temporary variables. We observed that the best performance was obtained when the GPU kernel was launched with 256 blocks of 1024 threads, resulting in a total of $2^{18}$ threads. Each thread performed more than one initialization when the requested number of initializations exceeded $2^{18}$. Using this setup, we achieved $2^{35.10}$ `Ascon` initializations per second on an RTX 4090.

Although there are some differences between `Ascon-128`, `Ascon-128a`, and `Ascon-Aead128`, our optimized implementation applies to all variants. This is because we focus on optimizing the implementation of the core common component, namely `Ascon-`$p$.

## 2.2 `ChaCha` Stream Cipher

`ChaCha` [Ber08] is a stream cipher that supports 128 to 256-bit secret keys. It is widely used in various security applications and cryptographic protocols, including TLS 1.3 [LHT18], SSH [Dev14], and VPN software such as WireGuard [Don17]. `ChaCha` is included in TLS 1.3 as an alternative to `AES` and is responsible for encrypting a significant portion of internet traffic.

The 512-bit internal state of `ChaCha` is represented by 16 words $x_0, x_1, \ldots, x_{15}$, each consisting of 32 bits. Initially, these words are filled with four constants, eight key words, one counter word, and three nonce words. These 16 words at the $m^{\text{th}}$ round are represented as a $4 \times 4$ matrix as follows:

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

`ChaCha` consists of 20 rounds. Each round applies four quarter-round functions. In odd-numbered rounds, these functions operate on the columns of the $4 \times 4$ state matrix, while in even-numbered rounds, they are applied to the diagonals. Figure 4 illustrates the operations performed in an odd-numbered round of `ChaCha`.

[4] https://ascon.iaik.tugraz.at/publications.html

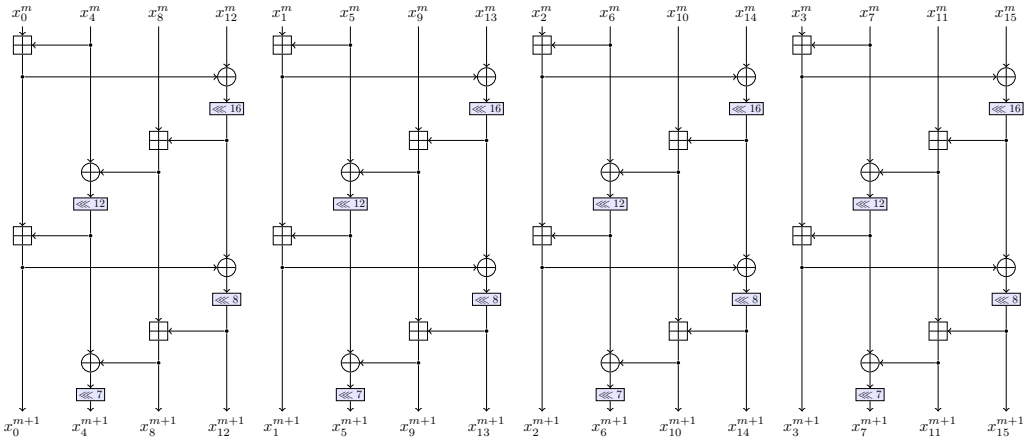



Figure 4: $m^{th}$ round of `ChaCha` where $m$ is odd.

In our GPU optimizations, we observed that the best performance is obtained when the GPU kernel is run with 256 blocks of 512 threads, resulting in $2^{17}$ threads. Each thread performs more than one encryption when the requested number of encryptions exceeds $2^{17}$. Using this configuration, we achieved $2^{34.92}$ 20-round `ChaCha` encryptions per second on an RTX 4090.

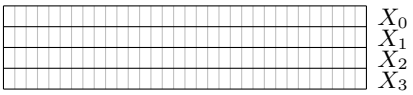
## 2.3 `Serpent` Block Cipher

`Serpent` [BAK98] was designed in 1998 by Anderson, Biham, and Knudsen, and it was selected as one of the five finalists of the Advanced Encryption Standard (AES) competition organized by NIST. `Serpent` is a block cipher with a 128-bit block size and supports key sizes of 128, 192, or 256 bits. We denote the version of `Serpent` with a $k$-bit key as `Serpent`-$k$. `Serpent` was designed with a strong emphasis on security, using a conservative approach that prioritizes resistance to cryptanalysis. Although it was not selected as the AES winner, `Serpent` remains widely studied in the academic community and is included in several cryptographic libraries and protocols as an alternative to `AES`, especially in applications where formal security margins are preferred.

Like many attacks on `Serpent` in the literature, we focus on recovering the round key bits rather than the master key. Therefore, we omit the description of the key schedule in this work. Each round function consists of a round key addition, followed by a bitsliced S-box layer applied across words, and a linear transformation that mixes all four words. Note that the linear layer is omitted in the last round. The S-box layer alternates between different S-boxes $S_0, \ldots, S_7$ in consecutive rounds. Overall, the round function at round $i$ is defined by the instructions in Figure 5 and the diffusion layer is shown in Figure 6. Here, $Y_i$ denotes the output state of round $i-1$, $K_i$ is the round key for round $i$, and $\lll$ and $\ll$ represent left rotation and left shift, respectively.

In our GPU optimizations, we used a bitsliced implementation of `Serpent`'s eight S-boxes. This approach allowed us to perform 32 S-box operations using only 15, 14, 16, 16, 15, 16, 15, and 16 logical operations for $S_0, S_1, S_2, S_3, S_4, S_5, S_6$, and $S_7$, respectively. The logical operations consist of AND, OR, NOT, and XOR, and the implementation uses at most 10 temporary variables.

We observed that the best performance is obtained when the GPU kernel is executed with 512 blocks of 1024 threads, resulting in a total of $2^{19}$ threads. Each thread performs more than one encryption when the requested number of encryptions exceeds $2^{19}$. Using this configuration, we achieved $2^{34.70}$ `Serpent` encryptions per second on an RTX 4090,




(a) State layout

| $x$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $S_0(x)$ | 3 | 8 | f | 1 | a | 6 | 5 | b | e | d | 4 | 2 | 7 | 0 | 9 | c |
| $S_1(x)$ | f | c | 2 | 7 | 9 | 0 | 5 | a | 1 | b | e | 8 | 6 | d | 3 | 4 |
| $S_2(x)$ | 8 | 6 | 7 | 9 | 3 | c | a | f | d | 1 | e | 4 | 0 | b | 5 | 2 |
| $S_3(x)$ | 0 | f | b | 8 | c | 9 | 6 | 3 | d | 1 | 2 | 4 | a | 7 | 5 | e |
| $S_4(x)$ | 1 | f | 8 | 3 | c | 0 | b | 6 | 2 | 5 | 4 | a | 9 | e | 7 | d |
| $S_5(x)$ | f | 5 | 2 | b | 4 | a | 9 | c | 0 | 3 | e | 8 | d | 6 | 7 | 1 |
| $S_6(x)$ | 7 | 2 | c | 5 | 8 | 4 | 6 | b | e | 9 | 1 | f | d | 3 | a | 0 |
| $S_7(x)$ | 1 | d | f | 0 | e | 8 | 2 | b | 7 | 4 | c | a | 9 | 3 | 5 | 6 |

(b) S-boxes of `Serpent`

$$
\begin{aligned}
X_0, X_1, X_2, X_3 &\leftarrow \mathcal{S}_{i \bmod 8}(Y_i \oplus K_i) \\
X_0 &\leftarrow X_0 \lll 13 \\
X_2 &\leftarrow X_2 \lll 3 \\
X_1 &\leftarrow X_1 \oplus X_0 \oplus X_2 \\
X_3 &\leftarrow X_3 \oplus X_2 \oplus (X_0 \ll 3) \\
X_1 &\leftarrow X_1 \lll 1 \\
X_3 &\leftarrow X_3 \lll 7 \\
X_0 &\leftarrow X_0 \oplus X_1 \oplus X_3 \\
X_2 &\leftarrow X_2 \oplus X_3 \oplus (X_1 \ll 7) \\
X_0 &\leftarrow X_0 \lll 5 \\
X_2 &\leftarrow X_2 \lll 22 \\
Y_{i+1} &\leftarrow X_0, X_1, X_2, X_3
\end{aligned}
$$

(c) Instructions for round $i$

Figure 5: Round function of `Serpent`.

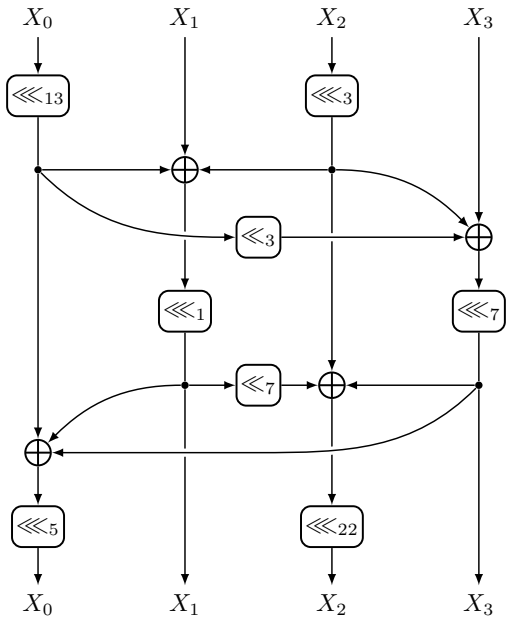



Figure 6: Diffusion layer of `Serpent`

including the key schedule in the computation.

## 3 Theory vs. Practice

Here, we elaborate on the experiments we conducted on the three symmetric-key cryptographic primitives listed above and discuss the deviations between practical and analytical results in more detail. Additionally, we propose improved attacks based on the practical

observations.

## 3.1 Ascon

We start with our observations on Ascon. One of the first gaps between theory and practice for Ascon was observed by the Ascon team [DEM15], where they proposed a 4-round DL distinguisher. Its bias was expected to be $2^{-20}$ based on a basic analytical estimation, whereas the actual observed bias in practice was $2^{-2}$. This large gap between the theoretical estimation and the experimental result stemmed from overlooking the interaction between the two underlying distinguishers within the combined DL distinguisher.

It turned out that once we consider the dependencies and interactions between the two underlying distinguishers of a combined distinguisher, the gap between theory and practice can be significantly reduced. A common approach for capturing these dependencies in combined distinguishers (such as boomerang and DL distinguishers) is the sandwich framework [DKS10, DKS14].

For example, in EUROCRYPT 2019, Bar-On et al. [BDKW19] proposed the Differential-Linear Connectivity Table (DLCT) to capture the interaction between the underlying differential and linear distinguishers within a combined DL distinguisher. Using the DLCT approach, they significantly reduced the gap between the analytical estimation and the practical bias by proposing a new bound of $2^{-5}$ for the 4-round DL distinguisher of Ascon. However, this bound of $2^{-5}$ is still notably lower than the practical bias of $2^{-2}$.

This gap was mainly due to the fact that DLCT could capture the dependency between the two trails only up to a single S-box layer, whereas the dependency between the two trails often spans multiple rounds. At CRYPTO 2024, Hadipour et al. generalized the DLCT framework to model the dependency across multiple rounds of DL distinguishers, and they successfully addressed the gap between analytical and experimental biases for strongly aligned SPN ciphers. They also proposed an automatic method to search for good DL distinguishers. In another work, Peng et al. [PZWD24a] introduced a different method to provide improved theoretical estimations for the bias of DL distinguishers in SPN ciphers.

However, for a given input difference and output mask, it can sometimes be computationally hard to provide an accurate analytical estimation using the state-of-the-art methods [LLL21, PZWD24a, HDE24, NSLL22]. On the other hand, it is often easier to calculate the bias experimentally by testing the distinguisher with a large amount of data, as long as the bias is large enough to be detected by the amount of data used in the experiment. For example, Civek and Tezcan [CT22] used GPUs to search for the best 5-round DL distinguishers for Ascon in both the single-key and related-key settings and reported competitive results. They also used GPU power to search for 6-round DL distinguishers with an input difference that activates only a single S-box. Using $2^{48}$ data, they did not find any such distinguishers. This result suggests that if such a 6-round distinguisher exists, its bias must be lower than $2^{-24}$.

However, despite the above observation, a 6-round DL distinguisher for Ascon with a reported bias of $2^{-22.43}$ was recently introduced at CRYPTO 2024 [PZWD24a]. This result appears counterintuitive, given that a thorough search in [CT22] failed to find any such distinguisher even with $2^{48}$ data. The analytical estimation of the bias for the first-ever 6-round (ordinary) DL distinguisher of Ascon was provided using the second method described in Section 4.3 of [PZWD24a]. This method accounts for the dependency between the differential and linear trails within only a single round, trading accuracy for higher computational efficiency in estimating the analytical bias.

The 6-round DL distinguisher of [PZWD24a] starts by activating a single S-box with an input difference of 0x8000000000000000 at words $x_3$ and $x_4$ (with zero difference in the other words), and uses an output mask of 0x9324496da496ddb4 at $x_0$ (with zero mask in the other words). Since the propagation of differences and linear masks through the final

round's linear layer is deterministic and holds with probability one, we can equivalently test a 5.5-round DL distinguisher with the output mask `0x0200000000000000` at $x_0$, which enables more efficient experimentation. We provide this distinguisher in Table 3.

Table 3: The 6-round DL of [PZWD24a] for `Ascon` with a claimed bias of $2^{-22.43}$. Intermediate round differences or masks are not shown since they were not explicitly provided in [PZWD24a]. We confirmed these differences and masks via personal communication with the authors.

| | **6-Round Differential-Linear** |
|---|---|
| Input Difference | 0000000000000000000000000000000000000000000000000000000000000000<br>0000000000000000000000000000000000000000000000000000000000000000<br>0000000000000000000000000000000000000000000000000000000000000000<br>1000000000000000000000000000000000000000000000000000000000000000<br>1000000000000000000000000000000000000000000000000000000000000000 |
| $S_6$ Mask | 0000001000000000000000000000000000000000000000000000000000000000<br>0000000000000000000000000000000000000000000000000000000000000000<br>0000000000000000000000000000000000000000000000000000000000000000<br>0000000000000000000000000000000000000000000000000000000000000000<br>0000000000000000000000000000000000000000000000000000000000000000 |
| $P_6$ Mask | 1001001100100100010010011011011010010010011011011101100110100<br>0000000000000000000000000000000000000000000000000000000000000000<br>0000000000000000000000000000000000000000000000000000000000000000<br>0000000000000000000000000000000000000000000000000000000000000000<br>0000000000000000000000000000000000000000000000000000000000000000 |

By using our optimized GPU implementations for `Ascon`, we conducted extensive experiments with $2^{54}$ data and observed that the claimed distinguisher does not exhibit the reported bias of $2^{-22.43}$. Specifically, the measured bias was consistently below $2^{-28}$ in our experiments, which aligns with what one would expect from a random permutation when tested with $2^{54}$ data. We contacted the authors of [PZWD24a], and they confirmed that they had also identified potential issues with their distinguisher.

Running such an experiment with $2^{54}$ random input pairs takes approximately six days on a single NVIDIA RTX 4090 GPU. Our CUDA code can be used to reproduce this experiment with the same or larger data sizes. The runtime can be significantly reduced by employing multiple GPUs in parallel.

Our experimental results for the 6-round DL distinguisher of [PZWD24a], along with the results for the 4 and 5-round distinguishers, are summarized in Table 4.

Table 4: Our Experimental results for the 4, 5, and 6-round DL distinguishers of [PZWD24a] on `Ascon`.

| **Rounds** | **Used Data** | **Theoretical Bias** | **Experimental Bias** |
|---|---|---|---|
| 4 | $2^{40.00}$ | $2^{-2.00}$ | $2^{-2.00}$ |
| 5 | $2^{40.00}$ | $2^{-10.10}$ | $2^{-9.94}$ |
| 6 | $2^{55.58}$ | $2^{-22.43}$ | $\ll 2^{-32.57}$ |

According to Table 4, the bias for the 5-round distinguisher is slightly better in practice than theoretically predicted. This suggests that the 4 and 5-round DL distinguishers on `Ascon` proposed in [PZWD24a] not only work in practice but even perform slightly better than expected.

However, by performing extensive experiments on the 6-round DL distinguisher of `Ascon` proposed in [PZWD24a], we confirmed that it does not work in practice, and its actual bias must be significantly smaller than the claimed $2^{-22.43}$. In our first experiment, we used $2^{55.58}$ randomly chosen plaintext pairs and measured a bias of $2^{-32.57}$. This result was already disappointing, and it merely implies that the true bias must be lower than $2^{-32.57}$, as $2^{55.58}$ data is insufficient to reliably detect such a low bias.

When we repeated the experiment four times with $2^{54}$ data, the sign of the bias was positive in two trials and negative in the other two. This observation indicates that the claimed 6-round distinguisher does not work in practice, and its bias cannot be distinguished from that of a random permutation using this amount of data ($2^{56}$ randomly selected plaintext pairs). Therefore, this experiment shows that the actual bias must be even lower than $2^{-28}$. We reported our observations to the authors of [PZWD24a], and they also admitted that this distinguisher does not hold with the claimed bias. But, the actual bias of this distinguisher remains an open problem.

One might wonder why the 4- and 5-round DL distinguishers of `Ascon` in [PZWD24a] work well in practice, whereas the 6-round DL distinguisher fails completely. We observed that [PZWD24a] proposes two different methods for analytically estimating the bias. The first method considers multiple rounds when handling the dependencies between the differential and linear trails, while the second method only takes a single round into account. The second method is more efficient, as reducing the number of rounds lowers the computational complexity of the analytical estimation. However, our results show that this efficiency comes at the cost of a significant loss in accuracy. In line with the rule-of-thumb stated in [HNE22], we believe that the number of rounds used to handle dependencies between the two parts of a combined distinguisher (such as DL distinguishers) should match the cipher's full diffusion length, which is 3.5 for `Ascon`. Therefore, we argue that the actual bias of the claimed 6-round DL distinguisher of `Ascon` must be much smaller than the reported $2^{-22.43}$, since the authors of [PZWD24a] consider only one round for modeling the dependency, far fewer than what is needed for accurate estimation.

As other recent results for `Ascon` we inspected the claimed 4.5-round deterministic truncated differential in [BJKK24] which we show in Table 5.

This attracted our attention since the best know deterministic truncated differential trails of `Ascon` could reach up to 3.5 [Tez16] and 4 rounds [HDE24] (a 4-round deterministic DL distinguisher for `Ascon`). This new 4.5-round truncated differential with probability one of [BJKK24] is a significant improvement because the best probability one truncated differential for `Ascon` was 3.5 rounds [Tez16] and in the same work a 4.5-round truncated differential was provided that has a probability of $2^{-108}$.

It should be noted that the authors of [BJKK24] do not claim that the input difference that is shown in Table 5 leads to the provided output difference with two non-active S-boxes. Instead, the distinguisher is obtained by starting at $P_1$ and the pairs are chosen in a way that the difference of `0x03` at a single S-box (represented as the first column of $P_1$ in bit notation) leads to the output difference of `0x10` with probability one. The rest of the differential in the forward direction is trivial from the round function and undisturbed bits [Tez14], which are probability one truncated differentials for S-boxes. Then from $P_1$, 1-round is added to the top of the distinguisher and the input difference of the distinguisher is obtained as shown in Table 5 which has 15 inactive S-boxes. Thus, conditions for an input pair to follow this differential with probability one is pre-determined and only pairs satisfying these conditions are allowed to be used for the distinguisher. This requires DDT table look-ups to determine if a pair is valid or not. However, a distinguisher obtained by starting from the middle of a permutation and using its properties should not be compared with other distinguishers as if they are in the same category. Because one can create a distinguisher for an arbitrary round of a permutation by looking at some input output pairs. For instance, we can find 10 input pairs that do not have an output difference in a specific bit after 12-round `Ascon` permutation and use it to distinguish 12-round `Ascon` permutation because it is impossible for these pairs to have a difference in that bit (because we chose them in this way). However, the expected value is $10 \times \frac{1}{2} = 5$ for a random permutation.

Thus, the 4.5-round truncated differential with probability one and 6-round impossible differential distinguishers provided in [BJKK24] should not be compared with the state of

Table 5: The 4.5-round probability one truncated differential of [BJKK24] for `Ascon`.

| | **4.5-Round Truncated Differential** |
|---|---|
| I | ********0*****00***0**************0***00***0*******00*0*00*0**0* |
| | ********0*****00***0**************0***00***0*******00*0*00*0**0* |
| | ********0*****00***0**************0***00***0*******00*0*00*0**0* |
| | ********0*****00***0**************0***00***0*******00*0*00*0**0* |
| | ********0*****00***0**************0***00***0*******00*0*00*0**0* |
| $S_1$ | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | 01101011010010000110011110111101110111001010100111000100001001010101 |
| | 11111100011101001000111011001111000110001100111110100101000011010101 |
| $P_1$ | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | 10000000000000000000000000000000000000000000000000000000000000000000 |
| | 10000000000000000000000000000000000000000000000000000000000000000000 |
| $S_2$ | 10000000000000000000000000000000000000000000000000000000000000000000 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| $P_2$ | 10000000000000000010000000010000000000000000000000000000000000000000 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| $S_3$ | *0000000000000000000*00000000*0000000000000000000000000000000000000 |
| | 10000000000000000010000000010000000000000000000000000000000000000000 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | *0000000000000000000*00000000*0000000000000000000000000000000000000 |
| | *0000000000000000000*00000000*0000000000000000000000000000000000000 |
| $P_3$ | *0000000000000000000*00000000*000000000*00000000*00000000*0000000 |
| | *00*000000000000*00*00000*00*0000000000*0000000000000000000*00*00 |
| | 00000000000000000000000000000000000000000000000000000000000000000000 |
| | *000000000*000000*0*00000000**000000*0*000000*000000000000000000 |
| | *0000*0*00000000000*000000*0*000000*00000*000000000000000000*000 |
| $S_4$ | *00*0*0*00*00000**0*00000**0**00000**0**0*000*0*00000000*0*0**00 |
| | *00*0*0*00*00000**0*00000**0**00000**0**0*000*0*00000000*0*0**00 |
| | *00*0*0*00*00000**0*00000**0**00000**0**0*000*0*00000000*0*0**00 |
| | *00*0*0*00*00000**0*00000**0**00000**0**0*000*0*00000000*0*0**00 |
| | *00*0*0*00*00000**0*00000**0**00000**0**0*000*0*00000000*0*0**00 |
| $P_4$ | *0**0*0*0***0*0***0**0*0***0**0*0*0**0**0*00**0**0000******0**0* |
| | ******0*00**0**0**0**0**0***0**0**0**0**0****0*000*0****0**00 |
| | ***********0*00*****0**0********0********0*****00*0*00*******0 |
| | *0******0***0*****0**0*0******000****0**0***0*****0***0****0***0 |
| | ********00*0********00*****0**00**0*****0********00**0*0***0**0* |
| $S_5$ | ********************0****************************0************ |
| | ********************0****************************0************ |
| | ********************0****************************0************ |
| | ********************0****************************0************ |
| | ********************0****************************0************ |

the art results in this category [Tez16].

## 3.2 Experiments on `ChaCha`

This subsection describes our observations regarding the results on the `ChaCha` stream cipher. There are many works in the literature on differential, linear, and differential-linear cryptanalysis of `ChaCha`. Some of these results were later found to be incorrect. For example, [CN20] introduced new linear approximations that had lower correlation than previous results but allowed more probabilistic neutral bits. However, these results were

later invalidated by [DDSM22].

Unlike many other cryptanalysis results in the literature, most of the differential-linear results for ChaCha have been experimentally verified using GPUs. For example, one of the currently best DL distinguishers and attacks on ChaCha was proposed in [BGG+23], where the authors verified reduced versions of their results using 32 GPUs. Their 7-round attack requires $2^{110.8}$ data and $2^{206.8}$ time complexity. More recently, a 7-round DL attack on ChaCha with $2^{127.7}$ data and $2^{148.2}$ time complexity was introduced in [FT25].

However, the best 7-round DL distinguisher for ChaCha was reported in [WGM24], with data and time complexities of both $2^{120.9}$. Generally in the literature, DL distinguishers for ChaCha consists of an experimentally obtained DL and a linear approximation appended to it. It was shown in [WGM24] that the 3-round DL of [CN21] has an experimental bias of $2^{-12.02}$. The main improvement in [WGM24] comes from a 4-round DL with a claimed bias of $2^{-16.6}$, which was obtained experimentally using CPU resources. Later on, they improved this DL to 4.25 rounds with an experimental bias of $2^{-19.93}$ in [OWGM25]. We report these distinguishers and claimed biases in Table 6.

Table 6: The beginning parts of DL distinguishers for ChaCha starting from the first round and experimental biases obtained by [WGM24] and [OWGM25].

| Work | #Rounds | Input Difference | Output Mask | Claimed Bias |
|---|---|---|---|---|
| [CN21] | 3 | $x_{14} = \texttt{0x00000040}$ | $x_3 = \texttt{0x00000001}$, $x_4 = \texttt{0x00000001}$ | $2^{-12.02}$ |
| [WGM24] | 4 | $x_{12} = \texttt{0x00000001}$ | $x_3 = \texttt{0x00000001}$ | $2^{-16.60}$ |
| [OWGM25] | 4.25 | $x_{12} = \texttt{0x00000001}$ | $x_3 = \texttt{0x00000001}$ | $2^{-19.93}$ |

To estimate the bias, the authors of [WGM24] performed $2^{10}$ independent experiments over $2^{10}$ randomly selected keys, each using $2^{30}$ different IVs. They then reported the median bias from these experiments. According to their claim, $2^{32.2}$ IV samples are sufficient to estimate the bias, and with their experiment having a total complexity of $2^{40}$, they argue that the experiment supports their result.

However, $2^{30}$ samples per key fall short of the claimed threshold of $2^{32.2}$, making the verification insufficient. Using our optimized CUDA implementation, we repeated this experiment with $2^{49}$ IV samples and observed no detectable bias—consistent with the behavior of a random function. This suggests that if the 4-round differential-linear distinguisher exists, its actual bias must be lower than $2^{-24.5}$, which is significantly smaller than the claimed bias of $2^{-16.6}$.

The 4-round differential-linear distinguisher in [WGM24] was constructed by introducing a one-bit input difference in the zeroth bit of $x_{12}$ and expecting a one-bit output difference in the zeroth bit of $x_3$ where other bits can have any difference. This is equivalent to an output mask only at the zeroth bit of $x_3$. As the authors noted, this input difference can be applied to any bit of the words $x_{12}$, $x_{13}$, $x_{14}$, or $x_{15}$, giving a total of 128 possible input positions. We repeated the experiment for each of these 128 positions using $2^{49}$ samples (i.e. one-bit input difference in one of the bits of the words $x_{12}$, $x_{13}$, $x_{14}$, or $x_{15}$ and one-bit output mask in the zeroth bit of $x_3$). All experiments produced results consistent with random behavior, with no detectable bias in any case. Therefore, we conclude that the distinguisher—and consequently the attacks—presented in [WGM24] are not valid.

In [OWGM25], which is the full version of [WGM24], the authors improved their 7-round differential-linear distinguisher by reducing its data and time complexities to $2^{68.18}$. Furthermore, they introduced a 7.5-round DL distinguisher with data and time complexities of $2^{132.18}$. However, both improvements rely on the previously misestimated 4-round differential bias and its extension to 4.25 rounds with a claimed bias of $2^{-19.93}$. This bias was again obtained using $2^{10}$ keys and $2^{30}$ IVs per key, a method that lacks sufficient statistical power to accurately estimate such a small bias. Consequently, the

reported complexities for the 7- and 7.5-round DL distinguishers in [OWGM25] are not valid. At the time of writing, the best-known DL distinguishers and attacks on `ChaCha` remain those presented in [BGG+23] and [FT25].

We verified our CUDA implementation of `ChaCha` using the test vectors provided in [LHT18]. We also experimentally observed the bias of $2^{-12.02}$ for the 3-round DL of [CN21] and obtained the other claimed biases of DLs and linear approximations provided in [CN21] which were within our computational bounds. Thus, both our and [WGM24]'s experiments provide exactly the same bias for the 3-round DL of [CN21]. However, our experiments do not provide the claimed biases[5] of $2^{-16.6}$ and $2^{-19.93}$ for the DLs of [WGM24] and [OWGM25] provided in Table 6. These different experimental results might be due to the fact that compared to our GPU experiments, the CPU experiments of [WGM24] and [OWGM25] use insufficient data. It might also be due to a bad RNG used during the experiments. Without publicly shared source codes, it is not possible for us to pinpoint the main cause.

All our experiments were performed on a single RTX 4090 GPU. Our optimized CUDA code can be used to replicate these experiments with the same or even larger amounts of data, and is included as supplementary material.

## 3.3 Experiments on `Serpent`

Here, we present our observations regarding the results on the `Serpent` block cipher.

### 3.3.1 Experiments on Distinguishers

One of the best-known attacks on `Serpent` is the DL attack proposed by Biham, Dunkelman, and Keller [BDK03], which breaks 10 rounds of `Serpent`, and extends to 11 rounds for `Serpent`-192 and `Serpent`-256. This attack was later improved by Dunkelman, Indesteege, and Keller [DIK08], who extended the results to 12 rounds of `Serpent`-256. These attacks commonly rely on a 9-round DL distinguisher that combines a 3-round differential characteristic with probability $2^{-5}$ and a 6-round linear approximation with bias $2^{-27}$.

In [DIK08], by using $2^{36}$ samples under 100 different random keys, it was experimentally observed that when the 3-round differential is combined with the first round of the linear approximation, the experimental bias of the resulting 4-round DL distinguisher becomes $2^{-13.75}$, compared to the theoretical bias of $2^{-15}$. This indicates that the distinguisher actually performs better in practice. Assuming that the piling-up lemma holds for the remaining five rounds of the linear approximation, the authors estimated a $2^{1.25}$ improvement in the overall bias of their 9-round DL distinguisher. They adopted this improved bias in their attack complexities that rely on this distinguisher.

By using our GPU power, we repeated the same experiment on the first 4 rounds of the 9-round distinguisher, but this time with $2^{50}$ samples under 100 random keys, and observed that the experimental bias is $2^{-13.73}$, which is marginally better than the reported value in [DIK08]. Note that the same bias was approximated as $2^{-13.736}$ in [LLL21] using the Differential Algebraic Transitional Form (DATF) method. Moreover, we extended the experiment to the first 5 and 6 rounds of the distinguisher and observed that the biases are $2^{-17.63}$ instead of $2^{-19}$ and $2^{-25.61}$ instead of $2^{-27}$, respectively. This improvement is likely due to the presence of multiple high-probability differential and linear paths. As a result, the advantage of $2^{1.25}$ reported in [DIK08] is actually $2^{1.39}$, which translates to approximately $2^{0.28}$ better time and data complexity in the corresponding attacks.

[5] In this work, a *bias* refers to $\epsilon$ when the probability of a distinguisher is $\frac{1}{2} + \epsilon$. In some works such a probability is represented as $\frac{1}{2}(1 + \epsilon_d)$ where $\epsilon_d$ is called the *correlation* and $\epsilon_d = 2 \times \epsilon$. Note that [WGM24] and [OWGM25] refer to $\epsilon_d$ as the differential bias and this is why in those works the biases were reported as twice the numbers we provided here, namely $2^{-15.6}$ and $2^{-18.93}$, instead of $2^{-16.6}$ and $2^{-19.93}$, respectively.

In [TÖ14], the input difference used in [BDK03] was rotated to examine whether the new differential aligns better with the linear approximation. Using this approach, the authors obtained five additional 9-round differential-linear distinguishers with observed biases of $2^{-13.49}$, $2^{-13.43}$, $2^{-13.56}$, $2^{-13.43}$, and $2^{-14.65}$. The best of these distinguishers achieves a $2^{0.30}$ stronger bias compared to the original distinguisher used in [BDK03, DIK08]. However, all of these new distinguishers activate one more S-box when one round is added to the top, which increases the time complexity by roughly a factor of $2^4$.

The approach of [TÖ14] was to first obtain the distinguisher theoretically and then evaluate its bias experimentally. However, as observed in [CT22] for `Ascon`, some distinguishers that cannot be found using existing analytical methods can still be discovered through experiments. Motivated by this, we fixed the 6-round linear approximation and rotated the input difference to all possible positions to search for better experimental bias. We conducted our experiments using 100 random keys and $2^{50}$ samples for each configuration. Our results show that rotating the input difference 14 bits to the left yields a bias of $2^{-12.33}$ for the first four rounds. The details of this distinguisher are provided in Table 10. Moreover, the gain of $2^{2.67}$ in the bias is preserved when extending the distinguisher to cover the fifth and sixth rounds. The results of our experiments on the distinguisher from [DIK08] and our experimentally obtained distinguisher are provided in Table 7.

Table 7: Experiment results on the original distinguisher and our 14-bit left shifted one.

| | | Distinguisher of [DIK08] | | Our Distinguisher | |
|---|---|---|---|---|---|
| r | Theoretical Bias | Experimental Bias | Gain | Experimental Bias | Gain |
| 4 | $2^{-15}$ | $2^{-13.73}$ | $2^{1.27}$ | $2^{-12.33}$ | $2^{2.67}$ |
| 5 | $2^{-19}$ | $2^{-17.63}$ | $2^{1.37}$ | $2^{-16.33}$ | $2^{2.67}$ |
| 6 | $2^{-27}$ | $2^{-25.61}$ | $2^{1.39}$ | $2^{-24.33}$ | $2^{2.67}$ |

Recently, in CRYPTO 2024, Hadipour et al. used the structural similarities between DL and boomerang distinguishers to propose several new DL distinguishers for 3 to 9 rounds of `Serpent` [HDE24]. These distinguishers have reported biases of $2^{-1.68}$, $2^{-6.54}$, $2^{-12.10}$, $2^{-21.58}$, $2^{-29.45}$, $2^{-40.18}$, and $2^{-55.43}$ for rounds 3 through 9, respectively. They verified their 3-, 4-, and 5-round distinguishers experimentally, but their CPU resources were not sufficient to verify the 6-, 7-, 8-, or 9-round distinguishers.

We verified their 5- and 6-round distinguishers using $2^{46}$ plaintext pairs, repeated for 100 randomly chosen keys, and observed that both distinguishers work in practice. However, for the 6-round distinguisher which is provided in Table 8, we observed a small discrepancy: the practical bias is $2^{-21.83}$ instead of the reported $2^{-21.58}$. This reduction by a factor of $2^{-0.25}$ in bias increases the data and time complexities by a factor of $2^{0.5}$. An experiment on 6 rounds with $2^{46}$ plaintext pairs takes around one day on a single RTX 4090 GPU using our codes.

Figure 7 illustrates the distribution of the observed bias (correlation) for 100 randomly selected keys. As seen in Figure 7, the bias depends on the underlying key, and the shape of the distribution clearly deviates from a normal distribution. This confirms that the assumption that the bias follows a normal distribution does not accurately reflect the practical observations. In particular, the distribution is not symmetric, meaning that the average bias is not a perfect representative of the typical behavior. While for some keys the bias is higher than expected, for others it can be lower. Nevertheless, the standard deviation remains small, i.e., $\sigma = 2^{-24.54}$ and for most of the keys that we tried the bias remains within the range of $[2^{-22.25}, 2^{-21.45}]$.

The distinguishers presented in [HDE24] were obtained using an automatic tool, similar to those developed for identifying boomerang distinguishers. The bias (correlation) of

Table 8: The 6-round DL of [HDE24] for `Serpent` which has a claimed bias of $2^{-21.58}$. Our experiments showed that in practice the bias is $2^{-21.83}$.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Input Difference | $X_0$: | 0000 | 0000 | 1000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_1$: | 0000 | 0100 | 1000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_2$: | 0000 | 0000 | 1000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_3$: | 0000 | 0100 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| $P_1$ Difference | $X_0$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_1$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_2$: | 0000 | 0000 | 0000 | 0001 | 0000 | 0000 | 0000 | 0000 |
| | $X_3$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| $P_4$ Mask | $X_0$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_1$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_2$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_3$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0001 | 0000 | 0000 |
| $P_6$ Mask | $X_0$: | 1000 | 0100 | 0000 | 0000 | 0001 | 0000 | 1000 | 0000 |
| | $X_1$: | 0000 | 1100 | 0010 | 1000 | 0100 | 0000 | 1000 | 0100 |
| | $X_2$: | 0000 | 0000 | 0000 | 1001 | 0000 | 1000 | 0000 | 0000 |
| | $X_3$: | 0010 | 0000 | 0000 | 0000 | 0000 | 0000 | 1000 | 0100 |

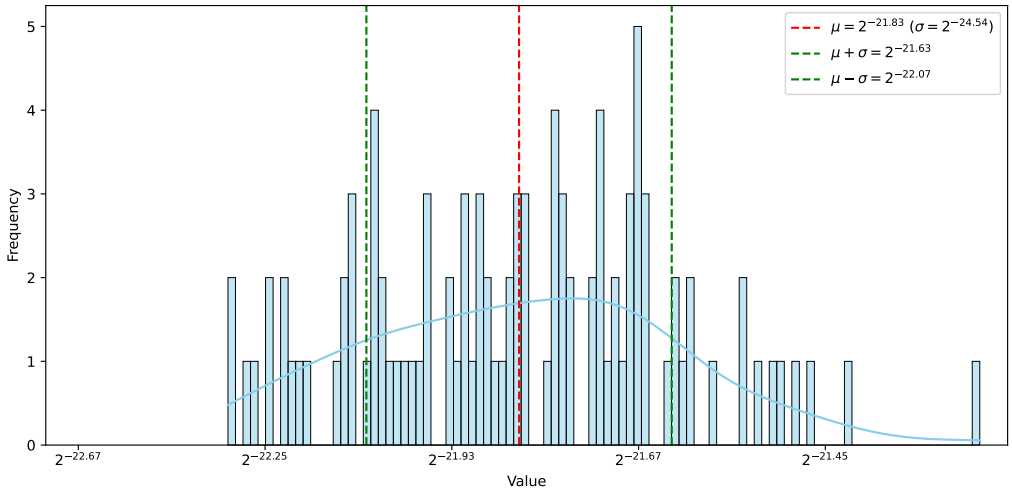



Figure 7: Distribution of the bias for 6-round `Serpent` distinguisher in [HDE24].

these DL distinguishers was estimated based on the sandwich framework, and considering multiple rounds to capture the dependencies between the differential and linear components. Our experimental results confirm the effectiveness of this technique. However, we also observe that the theoretically estimated biases may slightly differ from those observed in practice. This is mainly because the average bias is not always a reliable representative of the fixed-key biases, especially when the bias distribution is not symmetric.

Therefore, we anticipate that the reported biases for the 7-, 8-, and 9-round DL distinguishers in [HDE24] might be slightly lower or higher in practice depending on the key. Note that we could not verify these distinguishers experimentally due to our limited computational resources. To determine the practical bias of the 7-round DL distinguisher from [HDE24], at least $2^{60}$ 7-round `Serpent` encryptions over plaintext pairs are required. With our optimized implementation, this would take approximately seven months on a single RTX 4090 GPU. However, this task could be completed in a more reasonable time

using multiple GPUs. Such an experiment may be left for future work, particularly when faster next-generation GPUs become available.

As another recent result on `Serpent`, we examined the DL distinguishers for 3 to 9 rounds presented at CRYPTO 2024 by Peng et al. [PZWD24a]. The authors experimentally verified their results for 3, 4, and 5 rounds. Although the explicit distinguishers are not listed in the main paper, they are included in Table 12 of the Appendix in the full version of the paper published on ePrint [PZWD24b]. In our investigation, we found that the reported output masks for the 6-, 7-, 8-, and 9-round distinguishers, as well as for 6 out of the 7 provided 5-round distinguishers, are incorrect and do not lead to valid distinguishers. Since the authors provided some intermediate values of their distinguishers in [PZWD24b], we could perform experiments on different parts of the distinguishers and pinpoint the problematic part which was the final linear approximation. We contacted the authors, provided our results, and they have confirmed the presence of these errors.

Moreover, the authors discovered that the issues originated when writing down the output masks and confirmed that the reported biases were correct. They subsequently provided us with the corrected versions of these distinguishers, maintaining the same bias values as reported in [PZWD24a], and updated the ePrint version of their paper accordingly. We conducted experiments on these corrected distinguishers. For instance, using $2^{51}$ plaintext pairs and 100 randomly selected keys, we observed that the corrected 6-round DL distinguisher, originally reported with a bias of $2^{-19.61}$, indeed works in practice, but the observed bias was $2^{-21.55}$. After communicating these results with the authors, they identified additional remaining issues and provided us with re-corrected versions of the distinguishers. They also updated the ePrint version of their paper with these fixes (see version 20250427:093844). We then repeated the 6-round experiment using $2^{47}$ data and 100 random keys, and obtained a bias of $2^{-19.62}$, which is almost identical to the theoretical bias of $2^{-19.61}$. This distinguisher is provided in detail in Table 9. The details of the other corrected distinguishers can be found in the updated version of [PZWD24b].

Figure 8 illustrates the distribution of bias that we measured for 100 random keys. As can be seen, the bias is key-dependent and the fixed-key bias may deviate from the average-key bias. However, the standard deviation is small enough, i.e., $\sigma = 2^{-24.64}$ and for most of the keys the bias lies within the range $[2^{-19.73}, 2^{-19.50}]$.

Additionally, we performed experiments on the 7-round DL distinguishers of [PZWD24a]. These distinguishers have a reported bias of $2^{-29.45}$ and one needs to use more than $2^{60}$ data to verify it. Instead of the whole 7-round distinguisher, we experimentally checked the last 6 rounds of it using $2^{55.4}$ data. The last 6 rounds of this distinguisher has a reported bias of $2^{-26.45}$ but our experiments showed a bias of $2^{-29.10}$ which is no different than a random behavior for the used amount of data. Contacting the authors with these observations allowed them to discover the problem and they provided us the corrected version of their distinguisher. By eliminating the first round, we performed our experiments on the remaining 3, 4, and 5 rounds to obtain biases of $2^{-8.453}$, $2^{-16.451}$, and $2^{-20.449}$, respectively. We observed that, for the "corrected results", our experimentally measured biases match the theoretically estimated biases reported in [PZWD24a], with only slight deviations. Corrected distinguishers are provided in version 20250427:093844 of [PZWD24b]. These slight differences stem from the variation between fixed-key and average-key bias behavior, as well as the fact that the distribution of bias does not precisely follow a normal distribution.

### 3.3.2 New Distinguishers and Attacks

Note that the DL attacks on `Serpent` typically aim to recover the key bits involved in the first two rounds, which are applied before the S-boxes $S_0$ and $S_1$. Before explaining the key recovery attacks on `Serpent`, we define the concept of a *differential factor* as follows.

Table 9: The 6-round DL of [PZWD24a] for `Serpent` which has a claimed bias of $2^{-19.61}$. We experimentally observed that this distinguisher does not work in practice. Through personal communication with the authors, they provided a correction to the linear part which resulted in an experimental bias of $2^{-21.55}$. Further communication resulted in a correct 6-round DL distinguisher which we observed to have an experimental bias of $2^{-19.62}$.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Input Difference | $X_0$: | 0010 | 0000 | 0000 | 0001 | 0000 | 0000 | 0000 | 0000 |
| | $X_1$: | 0000 | 0000 | 0000 | 0001 | 0000 | 0000 | 0000 | 0000 |
| | $X_2$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_3$: | 0010 | 0000 | 0000 | 0001 | 0000 | 0000 | 0000 | 0000 |
| $P_1$ Difference | $X_0$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0100 |
| | $X_1$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_2$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_3$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| Original (invalid) | $X_0$: | 0000 | 0000 | 0000 | 0001 | 0000 | 0000 | 0000 | 0000 |
| | $X_1$: | 0000 | 0000 | 0000 | 0000 | 0000 | 1000 | 0000 | 0000 |
| $P_4$ Mask | $X_2$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_3$: | 0000 | 0000 | 0000 | 0000 | 0000 | 1000 | 0000 | 0000 |
| Corrected | $X_0$: | 0000 | 0000 | 0001 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_1$: | 0000 | 0000 | 0000 | 0000 | 1000 | 0000 | 0000 | 0000 |
| $P_4$ Mask | $X_2$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_3$: | 0000 | 0000 | 0000 | 0000 | 1000 | 0000 | 0000 | 0000 |
| Original (invalid) | $X_0$: | 1000 | 0100 | 1000 | 0000 | 1000 | 0000 | 0000 | 0110 |
| | $X_1$: | 0100 | 1100 | 0000 | 0100 | 0000 | 0011 | 0011 | 0110 |
| $P_6$ Mask | $X_2$: | 0000 | 0000 | 0000 | 0000 | 0010 | 0000 | 0000 | 1000 |
| | $X_3$: | 1000 | 0111 | 0010 | 0100 | 0000 | 0010 | 0001 | 1110 |
| Corrected | $X_0$: | 0010 | 0000 | 0001 | 0000 | 1000 | 0100 | 1000 | 0000 |
| | $X_1$: | 0000 | 0011 | 0000 | 0010 | 1000 | 0100 | 0010 | 1100 |
| $P_6$ Mask | $X_2$: | 0000 | 0001 | 0000 | 0000 | 0100 | 0000 | 0000 | 0000 |
| | $X_3$: | 0000 | 0000 | 0000 | 0000 | 1000 | 0100 | 0010 | 0000 |

**Definition 1** ([TÖ14])**.** Let $S$ be a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$. For all $x, y \in \mathbb{F}_2^n$ that satisfy $S(x) \oplus S(y) = \mu$, if we also have $S(x \oplus \lambda) \oplus S(y \oplus \lambda) = \mu$, then we say that the S-box has a *differential factor* $\lambda$ for the output difference $\mu$. (i.e. $\mu$ remains invariant for $\lambda$).

Note that, the existence of differential factors causes wrong keys to behave similarly to the correct key in differential-based key recovery attack, thereby violating the *Wrong Key Randomization Hypothesis (WKRH).*

**Theorem 1** ([TÖ14])**.** *Let $S$ be an S-box used in a block cipher, and suppose that $S$ contains a* differential factor *$\lambda$ for an output difference $\mu$. Assume that a partial round key $k$ is XORed with the input to $S$. If an input pair leads to the output difference $\mu$ under a partial subkey $k'$, then the same output difference also appears under the partial subkey $k' \oplus \lambda$. As a consequence, in a differential attack that involves guessing a partial subkey corresponding to the output difference $\mu$, the cryptanalyst's advantage is reduced by 1 bit. Hence, the time complexity of this key-guessing (guess-and-filter) step is halved.*

One of the best known 11-round DL attacks on `Serpent` was presented in [DIK08]. This attack requires $2^{121.8}$ chosen plaintexts and $2^{135.7}$ memory accesses to recover 48 bits of round key material. However, it was later shown in [Tez15] that the attack cannot fully recover these 48 key bits due to the presence of two differential factors. As a result, the effective key recovery is limited to 46 bits, and the time complexity of the attack is reduced to $2^{133.7}$ memory accesses.

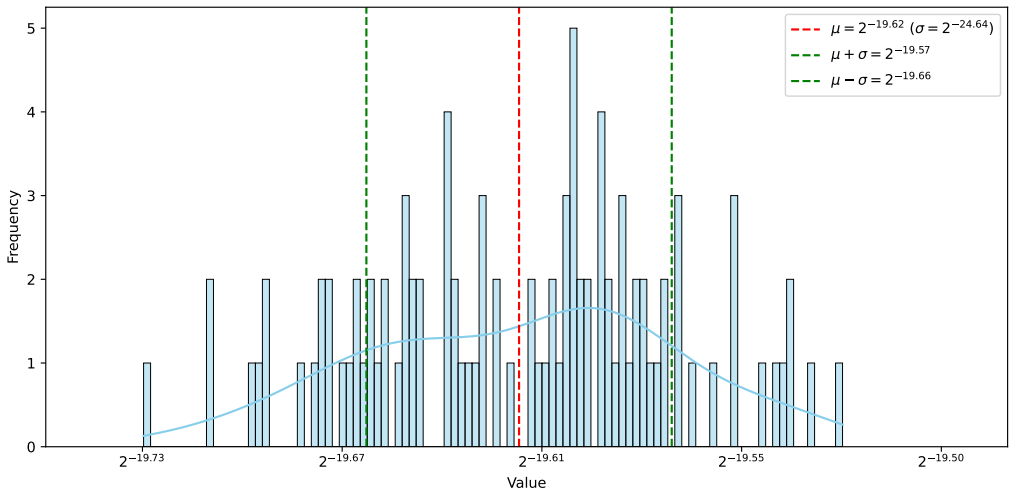



Figure 8: Distribution of the bias for the corrected version of the 6-round `Serpent` distinguisher in [PZWD24a]. We identified a flaw in the original 6-round distinguisher presented in [PZWD24a], and the authors corrected it based on our experimental verification and feedback through several response-feedback cycles.

As explained in Sec. 3.3.1, we obtained a DL distinguisher with a better bias by shifting the input difference 14 bits to the left. As in the original attack, we can add one round to the top and one round to the bottom of our experimentally obtained new 9-round DL distinguisher and attack 11 rounds of `Serpent`. The propagation of differences through the key recovery part and the involved sub-keys are presented in Table 10. It can be observed that our distinguisher also involves two differential factors for $S_1$.

We used the Blondeau-Gérard-Tillich algorithm [BGT11] to compute the required number of pairs and observed that our 11-round attack requires $2^{117.08}$ chosen plaintexts to achieve an 84% success probability, which is the same target as in previous DL attacks. The increased bias of our distinguisher results in an attack with a data complexity of $2^{118.08}$ chosen plaintexts, which is $2^{3.72}$ less than the data complexity of the 11-round attack in [DIK08]. Although this also reduces the time complexity by the same factor, our distinguisher activates one more S-box in the first round. As a result, our attack has a time complexity that is $2^{0.28}$ higher than the attack in [Tez15], but it achieves four more bits of advantage.

### 3.3.3 Wrong Attacks

The 11-round DL attack of [DIK08] is also extended to 12 rounds in the same work by adding one more round to the top. Only four S-boxes in the first round are not activated when the attack is extended to 12 rounds, and thus 112 bits of the first-round key are guessed in this attack, and the 11-round attack is repeated for every guess. Such an attack leads to a time complexity of $2^{249.4}$ encryptions, and it is reduced to $2^{246.4}$ encryptions in [Tez15] due to one more differential factor in this added first round. However, as previously noted in the literature, in order to perform the 12-round attack, one needs to guess all 128 bits of the first-round key (instead of 112 bits as claimed) to obtain the input of the 11-round attack. Thus, this increases the time complexities of the attacks by a factor of $2^{16}$ and these two attacks in practice require $2^{265.4}$ and $2^{262.4}$ encryptions, respectively. Since these numbers exceed the exhaustive search time complexity, they are both considered invalid attacks.

Table 10: Our new DL attack on 11-round `Serpent` actually starts at $P_0$. The top round is added to illustrate that, although three S-boxes have no input difference, the round key bits corresponding to those S-boxes must still be guessed to compute the intermediate values and verify the distinguisher. This effectively extend the attack into 12 rounds but time complexity exceeds brute-force search. Differential factors are shown in bold.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| I | $X_0$: | ???? | ???? | ???? | ???? | ???? | ???? | ??0? | ?00? |
| | $X_1$: | ???? | ???? | ???? | ???? | ???? | ???? | ??0? | ?00? |
| | $X_2$: | ???? | ???? | ?**1**?? | ???? | ???? | ???? | ??0? | ?00? |
| | $X_3$: | ???? | ???? | ???? | ???? | ???? | ???? | ??0? | ?00? |
| $S_0$ | $X_0$: | 0?00 | ?000 | 0000 | ??00 | ???? | 0?00 | ?000 | ?00? |
| | $X_1$: | ??0? | ???? | ?00? | 000? | ?0?? | ???? | ??00 | ?00? |
| | $X_2$: | 001? | 00?0 | ?10? | 00?0 | 00?0 | 0?00 | ??0? | ?00? |
| | $X_3$: | ???? | 0?0? | ?0?? | ?10? | 0??0 | ???0 | ??00 | ?001 |
| $P_0$ | $X_0$: | ?00? | 00?0 | 0000 | 000? | 00?0 | 0000 | 0000 | 000? |
| | $X_1$: | ?00? | 00?0 | 0000 | 000? | 00?0 | 0000 | 0000 | 000? |
| | $X_2$: | ?00? | 00?0 | 0000 | 000? | 00?0 | 0000 | 0000 | 000? |
| | $X_3$: | ?00? | 0010 | 0000 | 000? | 00?0 | 0000 | 0000 | 0001 |
| $S_1$ | $X_0$: | 000**0** | 00**0**0 | 0000 | 0000 | 0000 | 0000 | 0000 | 0001 |
| | $X_1$: | 100**1** | 00**0**0 | 0000 | 0000 | 0010 | 0000 | 0000 | 0000 |
| | $X_2$: | 000**1** | 00**1**0 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_3$: | 100**1** | 00**0**0 | 0000 | 0001 | 0000 | 0000 | 0000 | 0000 |
| $P_1$ | $X_0$: | 0000 | 0000 | 0000 | 0100 | 0000 | 0000 | 0000 | 0000 |
| | $X_1$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | $X_2$: | 0000 | 0000 | 0010 | 0100 | 0000 | 0000 | 0000 | 0000 |
| | $X_3$: | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| 9-Round DL Distinguisher | | | | | | | | | |
| Extra Round | | | | | | | | | |

Moreover, the new differential factor used in [Tez15] is two rounds away from the distinguisher, and it was shown in [KSS+24] that it does not actually prevent distinguishing the round key bits and therefore cannot be used to reduce the time complexity. This is because Theorem 1 does not directly apply to differential factors that are not located in the round immediately before or after the distinguisher. Thus, the correct time complexity should be $2^{263.4}$ encryptions for the 12-round attack of [Tez15].

If we also extend our 11-round attack to a 12-round attack by adding one more round to the top, as shown in Table 10, we obtain a better data complexity of $2^{119.78}$ chosen plaintexts and are able to capture 4 more key bits compared to previous attacks. However, the overall time complexity still exceeds that of exhaustive search, and therefore, the attack will not be valid.

### 3.3.4 Improved Attacks

The 10-round DL attack of [DIK08] is obtained by removing the last round of the 11-round attack, which is based on the 9-round DL distinguisher. If we shift the input difference 14 bits to the right, as we did in the 11-round attack shown in Table 10, the resulting 10-round attack increases the time complexity from $2^{113.2}$ to $2^{113.84}$ encryptions. At the same time, it reduces the data complexity from $2^{101.2}$ to $2^{97.84}$ chosen plaintexts. Moreover, this new attack captures four more round key bits.

The invalid 12-round DL attack of [DIK08] is corrected in [BCD+22] by using the same 9-round distinguisher, but reducing the work required to guess the round key bits by observing certain properties of the S-boxes. Note that they consider the bias of the distinguisher to be $2^{-57.75}$ based on the experimental result of [DIK08]. Since we showed

that the actual bias of the distinguisher is $2^{-57.61}$, all data, memory, and time complexities reported in [BCD+22] can be reduced by at least a factor of $2^{0.28}$. Similarly, the 12-round DL attack of [LLL21] uses a bias of $2^{-57.736}$, and using our experimentally obtained bias of $2^{-57.61}$ results in $2^{0.252}$ improvement in data, time, and memory complexities. It should be noted that the 12-round attack of [LLL21] starts from the fourth round, while the 12-round attack of [BCD+22] starts from the zeroth round.

However, the improvements presented in [LLL21] and [BCD+22] were obtained theoretically, and we do not know whether the authors may have overlooked certain properties such as differential factors, or underestimated any probabilities or related calculations. To verify that the provided data, time, and memory complexities are accurate, it would be preferable if the authors had practically performed the attack by reducing the 9-round DL distinguisher to 4 or 5 rounds. The summary of DL attacks on `Serpent` is provided in Table 11.

Table 11: Summary of the DL attacks on `Serpent`. We report the corrected time complexities of the 12-round attacks, which turn out to be no better than exhaustive search. *En: Encryptions, B: bytes, bl: blocks, MA: Memory Accesses, CP: Chosen Plaintexts, CC: Chosen Ciphertexts, n/s: not specified.*

| #Rounds | Key Size | Data | Time | Memory | Advantage | Success | Reference |
|---|---|---|---|---|---|---|---|
| 10 | All | $2^{105.2}$ CP | $2^{123.2}$ En | $2^{40}$ B | 38 | 72.1% | [BDK03] |
| 10 | All | $2^{101.2}$ CP | $2^{115.2}$ En | $2^{40}$ B | 38 | 84% | [DIK08] |
| 10 | All | $2^{101.2}$ CP | $2^{113.2}$ En | $2^{40}$ B | 38 | 84% | [Tez15] |
| 10 | All | $2^{97.84}$ CP | $2^{113.84}$ En | $2^{40}$ B | 42 | 84% | This paper |
| 11 | 192, 256 | $2^{125.3}$ CP | $2^{172.4}$ En | $2^{30}$ B | 46 | 72.1% | [BDK03] |
| 11 | 192, 256 | $2^{125.3}$ CP | $2^{139.2}$ En | $2^{60}$ B | 46 | 72.1% | [BDK03] |
| 11 | 192, 256 | $2^{121.8}$ CP | $2^{135.7}$ MA | $2^{76}$ B | 46 | 84% | [DIK08] |
| 11 | 192, 256 | $2^{121.8}$ CP | $2^{133.7}$ MA | $2^{76}$ B | 46 | 84% | [Tez15] |
| 11 | 192, 256 | $2^{118.08}$ CP | $2^{133.98}$ MA | $2^{76}$ B | 50 | 84% | This paper |
| 11 | All | $2^{126.6}$ CP | $2^{127.1}$ En | $2^{126.6}$ bl | 40 | 85% | [LLL21] |
| 11 | All | $2^{126.35}$ CP | $2^{126.85}$ En | $2^{126.35}$ bl | 40 | 85% | This paper |
| 12 | 256 | $2^{123.5}$ CP | $2^{265.4}$ En | $2^{128.5}$ B | 158 | 84% | [DIK08] |
| 12 | 256 | $2^{123.5}$ CP | $2^{263.4}$ En | $2^{128.5}$ B | 158 | 84% | [Tez15] |
| 12 | 256 | $2^{119.78}$ CP | $2^{263.68}$ En | $2^{128.5}$ B | 162 | 84% | This paper |
| 12 | 256 | $2^{127}$ CC | $2^{251}$ MA | $2^{127}$ bl | n/s | 77% | [LLL21] |
| 12 | 256 | $2^{126.75}$ CC | $2^{250.75}$ MA | $2^{126.75}$ bl | n/s | 77% | [LLL21] |
| 12 | 256 | $2^{127.92}$ CP | $2^{233.55}$ En | $2^{127.92}$ bl | 158 | 84% | [BCD+22] |
| 12 | 256 | $2^{127.64}$ CP | $2^{233.27}$ En | $2^{127.64}$ bl | 158 | 84% | This paper |
| 12 | 256 | $2^{125.74}$ CP | $2^{236.91}$ En | $2^{125.71}$ bl | 158 | 84% | [BCD+22] |
| 12 | 256 | $2^{125.46}$ CP | $2^{236.63}$ En | $2^{125.46}$ bl | 158 | 84% | This paper |
| 12 | 256 | $2^{118.40}$ CP | $2^{242.93}$ En | $2^{118.40}$ bl | 158 | 84% | [BCD+22] |
| 12 | 256 | $2^{118.12}$ CP | $2^{242.65}$ En | $2^{118.12}$ bl | 158 | 84% | This paper |

## 4 Conclusion and Future Works

Many cryptanalytic results are derived theoretically; however, they may behave differently in practice due to various factors such as incorrect assumptions, discrepancies between theoretical and actual probabilities, or undiscovered structural properties of the cipher under analysis. In this work, we utilized the parallel computational power of GPUs to experimentally evaluate several theoretically proposed cryptanalytic results on the `Ascon`, `ChaCha`, and `Serpent` ciphers. We showed that the first 6-round (ordinary) DL distinguisher of `Ascon`, presented in CRYPTO 2024 [PZWD24a], is not valid, and we identified several issues in the recently proposed DL distinguishers of `Serpent` from the same work [PZWD24a]. All of these issues were confirmed through direct communication

with the authors, who acknowledged the correctness of our findings. We also showed that the 4.5-round truncated differential with probability one and the 6-round impossible differential distinguishers on `Ascon` in [BJKK24] are flawed. Moreover, we verified that the best-known 7- and 7.5-round DL distinguishers on `ChaCha` do not work in practice.

On the other hand, we demonstrated that the best-known 10- and 11-round DL attacks on `Serpent` actually perform better in practice. We also provided a new experimentally obtained 9-round DL distinguisher for `Serpent`, which can be used in 10- and 11-round attacks to reduce data complexity. Furthermore, we showed that the best 12-round DL attacks on `Serpent` can be performed with marginally less data, time, and memory complexity.

These findings suggest that other theoretical results may not hold in practice. We recommend experimentally validating reduced versions of such analyses. Developing theoretical methods to assess DL distinguishers, similar to geometric or quasidifferential techniques for differential analysis [BR22], is also a promising direction for future work.

## Acknowledgments

This work was supported by The Scientific and Technological Research Council of Türkiye (TÜBITAK) and German Academic Exchange Service (DAAD) Bilateral Research Cooperation Project (TÜBİTAK 2531 Project) under the grant number 123N546 and titled "Cryptanalysis of Symmetric Key Encryption Algorithms: Theory vs. Practice". The authors thank TÜBİTAK and DAAD for their support.

This work was also supported by the ERC project 101097056 (SYMTRUST) and the enCRYPTON project. The later has received funding from the European Union's Horizon Europe Research and innovation programme under grant agreement No: 101079319. Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union, European Commission or European Research Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

## References

[BAK98] Eli Biham, Ross J. Anderson, and Lars R. Knudsen. Serpent: A new block cipher proposal. In Serge Vaudenay, editor, *Fast Software Encryption, 5th International Workshop, FSE '98, Paris, France, March 23-25, 1998, Proceedings*, volume 1372 of *LNCS*, pages 222–238. Springer, 1998.

[BCD+22] Marek Broll, Federico Canale, Nicolas David, Antonio Flórez-Gutiérrez, Gregor Leander, María Naya-Plasencia, and Yosuke Todo. New attacks from old distinguishers improved attacks on Serpent. In Steven D. Galbraith, editor, *CT-RSA 2022*, volume 13161 of *LNCS*, pages 484–510. Springer, 2022.

[BDBN23] Christina Boura, Nicolas David, Rachelle Heim Boissier, and María Naya-Plasencia. Better steady than speedy: Full break of SPEEDY-7-192. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023*, volume 14007 of *LNCS*, pages 36–66. Springer, 2023.

[BDK03] Eli Biham, Orr Dunkelman, and Nathan Keller. Differential-linear cryptanalysis of Serpent. In Thomas Johansson, editor, *Fast Software Encryption, 10th International Workshop, FSE 2003, Lund, Sweden, February 24-26, 2003, Revised Papers*, volume 2887 of *LNCS*, pages 9–21. Springer, 2003.

[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, *EUROCRYPT 2019*, volume 11476 of *LNCS*, pages 313–342. Springer, 2019.

[BDPVA12] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Permutation-based encryption, authentication and authenticated encryption. *Directions in Authenticated Ciphers*, pages 159–170, 2012.

[Ber08] 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 *LNCS*, pages 84–97. Springer, 2008.

[Ber13] Daniel J. Bernstein. Caesar: Competition for authenticated encryption: Security, applicability, and robustness. https://competitions.cr.yp.to/caesar.html, 2013. Accessed: 2024-11-18.

[BGG+23] Emanuele Bellini, David Gérault, Juan Grados, Rusydi H. Makarim, and Thomas Peyrin. Boosting differential-linear cryptanalysis of ChaCha7 with MILP. *IACR Trans. Symmetric Cryptol.*, 2023(2):189–223, 2023.

[BGT11] Céline Blondeau, Benoît Gérard, and Jean-Pierre Tillich. Accurate estimates of the data complexity and success probability for various cryptanalyses. *Des. Codes Cryptogr.*, 59(1-3):3–34, 2011.

[BJKK24] Seungjun Baek, Yongjin Jeon, Giyoon Kim, and Jongsung Kim. On impossible and truncated distinguishers for IoT-friendly AEAD algorithms. *IEEE Internet Things J.*, 11(19):31878–31891, 2024.

[BN24] Tim Beyne and Addie Neyt. Note on the cryptanalysis of Speedy. Cryptology ePrint Archive, Paper 2024/262, 2024.

[BR22] Tim Beyne and Vincent Rijmen. Differential cryptanalysis in the fixed-key model. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022*, volume 13509 of *LNCS*, pages 687–716. Springer, 2022.

[CN20] Murilo Coutinho and T. C. Souza Neto. New multi-bit differentials to improve attacks against ChaCha. Cryptology ePrint Archive, Paper 2020/350, 2020.

[CN21] 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, *EUROCRYPT 2021*, volume 12696 of *LNCS*, pages 711–740. Springer, 2021.

[CT22] Aslı Basak Civek and Cihangir Tezcan. Experimentally obtained differential-linear distinguishers for permutations of ASCON and drygascon. In Paolo Mori, Gabriele Lenzini, and Steven Furnell, editors, *ICISSP 2022*, volume 1851 of *Communications in Computer and Information Science*, pages 91–103. Springer, 2022.

[DDSM22] Sabyasachi Dey, Chandan Dey, Santanu Sarkar, and Willi Meier. Revisiting cryptanalysis on ChaCha from Crypto 2020 and Eurocrypt 2021. *IEEE Transactions on Information Theory*, 68(9):6114–6133, 2022.

[DEM15] Christoph Dobraunig, Maria Eichlseder, and Florian Mendel. Heuristic tool for linear cryptanalysis with applications to CAESAR candidates. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015*, volume 9453 of *LNCS*, pages 490–509. Springer, 2015.

[DEMS21] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1.2: Lightweight authenticated encryption and hashing. *J. Cryptol.*, 34(3):33, 2021.

[Dev14] OpenSSH Developers. Openssh release notes: Chacha20-poly1305 authenticated encryption. https://www.openssh.com/txt/release-6.5, 2014.

[DIK08] Orr Dunkelman, Sebastiaan Indesteege, and Nathan Keller. A differential-linear attack on 12-round Serpent. In Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das, editors, *INDOCRYPT 2008*, volume 5365 of *LNCS*, pages 308–321. Springer, 2008.

[DKS10] Orr Dunkelman, Nathan Keller, and Adi Shamir. A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3G telephony. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223, pages 393–410. Springer, 2010.

[DKS14] Orr Dunkelman, Nathan Keller, and Adi Shamir. A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3G telephony. *J. Cryptol.*, 27(4):824–849, 2014.

[Don17] Jason A. Donenfeld. Wireguard: Next generation kernel network tunnel. In *NDSS*, 2017.

[FT25] Antonio Flórez-Gutiérrez and Yosuke Todo. Improved cryptanalysis of ChaCha: Beating PNBs with bit puncturing. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025*, volume 15601 of *LNCS*, pages 427–457. Springer, 2025.

[HDE24] Hosein Hadipour, Patrick Derbez, and Maria Eichlseder. Revisiting differential-linear attacks via a boomerang perspective with application to AES, Ascon, CLEFIA, SKINNY, PRESENT, KNOT, TWINE, WARP, LBlock, Simeck, and SERPENT. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024*, volume 14923 of *LNCS*, pages 38–72. Springer, 2024.

[HNE22] Hosein Hadipour, Marcel Nageler, and Maria Eichlseder. Throwing boomerangs into feistel structures application to CLEFIA, WARP, LBlock, LBlock-s and TWINE. *IACR Trans. Symmetric Cryptol.*, 2022(3):271–302, 2022.

[KSS+24] Seonkyu KIM, Myoungsu SHIN, Hanbeom SHIN, Insung KIM, Sunyeop KIM, Donggeun KWON, Deukjo HONG, Jaechul SUNG, and Seokhie HONG. Differential factors revisited: A sufficient condition for the practical use of differential factors. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, advpub:2024EAP1058, 2024.

[LHT18] Adam Langley, Mike Hamburg, and Sean Turner. ChaCha20 and Poly1305 for IETF Protocols. *RFC*, 8439, 2018.

[LLL21] Meicheng Liu, Xiaojuan Lu, and Dongdai Lin. Differential-linear cryptanalysis from an algebraic perspective. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021*, volume 12827 of *LNCS*, pages 247–277. Springer, 2021.

[Lu15] Jiqiang Lu. A methodology for differential-linear cryptanalysis and its applications. *Des. Codes Cryptogr.*, 77(1):11–48, 2015.

[Mur11] Sean Murphy. The return of the cryptographic boomerang. *IEEE Trans. Inf. Theory*, 57(4):2517–2521, 2011.

[NSLL22] Zhongfeng Niu, Siwei Sun, Yunwen Liu, and Chao Li. Rotational differential-linear distinguishers of ARX ciphers with arbitrary output linear masks. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022*, volume 13507 of *LNCS*, pages 3–32. Springer, 2022.

[OWGM25] Yurie Okada, Ryo Watanabe, Nasratullah Ghafoori, and Atsuko Miyaji. Improved differential-linear cryptanalysis of reduced rounds of ChaCha permutation. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, advpub:2024DMP0008, 2025.

[PZWD24a] Ting Peng, Wentao Zhang, Jingsui Weng, and Tianyou Ding. New approaches for estimating the bias of differential-linear distinguishers. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024*, volume 14923 of *LNCS*, pages 174–205. Springer, 2024.

[PZWD24b] Ting Peng, Wentao Zhang, Jingsui Weng, and Tianyou Ding. New approaches for estimating the bias of differential-linear distinguishers (full version). Cryptology ePrint Archive, Paper 2024/871, 2024.

[Tez14] Cihangir Tezcan. Improbable differential attacks on Present using undisturbed bits. *J. Comput. Appl. Math.*, 259:503–511, 2014.

[Tez15] Cihangir Tezcan. Differential factors revisited: Corrected attacks on PRESENT and SERPENT. In Tim Güneysu, Gregor Leander, and Amir Moradi, editors, *Lightweight Cryptography for Security and Privacy - 4th International Workshop, LightSec 2015, Bochum, Germany, September 10-11, 2015, Revised Selected Papers*, volume 9542 of *LNCS*, pages 21–33. Springer, 2015.

[Tez16] Cihangir Tezcan. Truncated, impossible, and improbable differential analysis of ASCON. In Olivier Camp, Steven Furnell, and Paolo Mori, editors, *ICISSP 2016*, pages 325–332. SciTePress, 2016.

[TL25] Cihangir Tezcan and Gregor Leander. GPU assisted brute force cryptanalysis of GPRS, GSM, RFID, and TETRA. *IACR Trans. Symmetric Cryptol.*, 2025(1):309–327, 2025.

[TMC+24] Meltem Sönmez Turan, Kerry A. McKay, Donghoon Chang, Jinkeon Kang, and John Kelsey. Ascon-based lightweight cryptography standards for constrained devices: Authenticated encryption, hash, and extendable output functions. NIST Special Publication 800 – NIST SP 800-232 ipd. https://csrc.nist.gov/pubs/sp/800/232/ipd, 2024. Accessed: 2024-11-17.

[TÖ14] Cihangir Tezcan and Ferruh Özbudak. Differential factors: Improved attacks on SERPENT. In Thomas Eisenbarth and Erdinç Öztürk, editors, *Lightweight Cryptography for Security and Privacy - Third International Workshop, LightSec 2014, Istanbul, Turkey, September 1-2, 2014, Revised Selected Papers*, volume 8898 of *LNCS*, pages 69–84. Springer, 2014.

[WGM24] Ryo Watanabe, Nasratullah Ghafoori, and Atsuko Miyaji. Improved differential-linear cryptanalysis of reduced rounds of ChaCha. In Howon Kim and Jonghee Youn, editors, *Information Security Applications*, pages 269–281, Singapore, 2024. Springer Nature Singapore.