Revisiting Differential-Linear Attacks via a Boomerang Perspective with Application to AES, Ascon, CLEFIA, SKINNY, PRESENT, KNOT, TWINE, WARP, LBlock, Simeck, and SERPENT

Hosein Hadipour, Patrick Derbez, 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

Hosein Hadipour[1](✉), Patrick Derbez[2] and Maria Eichlseder[1]

[1] Graz University of Technology, Graz, Austria
hossein.hadipour@iaik.tugraz.at, hsn.hadipour@gmail.com
[2] Univ Rennes, Inria, CNRS, IRISA, Rennes, France

Abstract. In 1994, Langford and Hellman introduced differential-linear (DL) cryptanalysis, with the idea of decomposing the block cipher E into two parts, E_u and E_ℓ, such that E_u exhibits a high-probability differential trail, while E_ℓ has a high-correlation linear trail. Combining these trails forms a distinguisher for E, assuming independence between E_u and E_ℓ. The dependency between the two parts of DL distinguishers remained unaddressed until EUROCRYPT 2019, where Bar-On et al. [3] introduced the DLCT framework, resolving the issue up to one S-box layer. However, extending the DLCT framework to formalize the dependency between the two parts for multiple rounds remained an open problem. In this paper, we first tackle this problem from the perspective of boomerang analysis. By examining the relationships between DLCT, DDT, and LAT, we introduce a set of new tables facilitating the formulation of dependencies between the two parts of the DL distinguisher across multiple rounds. Then, we introduce a highly versatile and easy-to-use automatic tool for exploring DL distinguishers, inspired by automatic tools for boomerang distinguishers. This tool considers the dependency between differential and linear trails across multiple rounds. We apply our tool to various symmetric-key primitives, and in all applications, we either present the first DL distinguishers or enhance the best-known ones. We achieve successful results against Ascon, AES, SERPENT, PRESENT, SKINNY, TWINE, CLEFIA, WARP, LBlock, Simeck, and KNOT. Furthermore, we demonstrate that, in some cases, DL distinguishers outperform boomerang distinguishers significantly.

Keywords: Differential-linear analysis · DLCT · UDLCT · LDLCT · EDLCT · DDLCT · AES · Ascon · SKINNY · SERPENT · PRESENT · KNOT · WARP · LBlock · Simeck · TWINE

1 Introduction

The security assessment of a symmetric primitive typically involves subjecting it to various cryptanalysis techniques to ascertain that none pose a significant

threat. This analysis also aims to gauge the security margins provided by the primitive, allowing designers to finely tune operational parameters (such as the number of rounds in block ciphers) to strike a balance between efficiency and security. Differential [12] and linear [43] attacks represent two fundamental cryptanalysis techniques. However, given the well-known nature of these cryptanalysis techniques, most new cryptographic primitives are designed to resist them, at least in terms of *basic* applications of the attacks. One way to resist these attacks is to prevent the existence of long, high-probability differential or linear trails.

Nevertheless, in 1994, Langford and Hellman [38] showed that the non-existence of such trails does not necessarily imply the security of the primitive against differential and linear attacks. They introduced the concept of a combined attack known as the *differential-linear* (DL) attack. This attack merges the principles of both differential and linear cryptanalysis, forming an effective distinguisher on more rounds than achievable using either technique alone. They showed that if we can decompose the block cipher E into two parts $E = E_\ell \circ E_u$ such that there is a high-probability differential trail for E_u and a high-correlation linear trail for E_ℓ, then we can combine them to create a distinguisher for E. DL attacks have been successfully applied to many ciphers and led to a full-round attack on COCONUT98 [10] as well as the best known attacks for several ciphers such as SERPENT [26], ICEPOLE [36], and Chaskey [6, 39].

However, the complexity analysis of the DL distinguisher relies on two statistical assumptions: an independence assumption for the two parts E_u and E_ℓ and certain randomness assumptions for the output difference of E_u. Thus, many of the follow-up works were dedicated to formalizing the complexity of the DL distinguisher, relaxing the underlying assumptions. In 2002, Biham et al. [10] showed that some of the assumptions made in [38] may fail in practice, and proposed an enhanced DL attack. In 2017, Blondeau et al. [13] provided an exact expression of the correlation for DL distinguishers in a closed form under the sole assumption that the two parts are independent. While the dependence between the two parts E_u and E_ℓ remained unexplored for a considerable period, it was eventually investigated within the context of another differential-based combined attack, namely, the boomerang attack [55]. Dunkelman et al. [27] proposed the sandwich framework to consider the dependence between the two parts of the boomerang distinguisher. The core idea of the sandwich framework is to divide the block cipher E into three parts E_u, E_m, and E_ℓ, where the middle part E_m takes dependencies into account. A similar framework is also applicable for differential-linear distinguishers. In this case, E_u is covered by an ordinary differential distinguisher and E_ℓ by a linear distinguisher, while E_m is subject to a small combined distinguisher that connects the two parts, as illustrated in Figure 1b. The main role of the middle part E_m is to take the dependency between E_u and E_ℓ into account while computing the correlation of the DL distinguisher. One option is to estimate the correlation of differential-linear approximations over E_m with an experimental approach using a sufficiently large set of random inputs. As a more formal alternative, at EUROCRYPT 2019, Bar-On et al. [3] introduced the Differential-Linear Connectivity Table (DLCT)

to derive the correlation of the middle part E_m when it covers only one S-box layer. However, they left extending the DLCT framework to the case where the middle part includes multiple rounds as an open problem. At CRYPTO 2021, Liu et al. [41] proposed a pure algebraic method to estimate the correlation of DL distinguishers for multiple rounds, but this approach is quite different from the DLCT framework. This situation is different compared to boomerang attacks, where recent papers made significant progress in generalizing the BCT framework to multi-round middle parts for various design paradigms [15, 16, 30, 33]. Despite the similar structure of boomerang distinguishers and differential-linear distinguishers, the duality between these two has so far not been systematically explored, and it remains an open problem to leverage insights gained in one technique for the other.

An even more pressing open problem is that identifying the most effective DL distinguishers against a primitive is a non-trivial task, particularly when taking dependencies between the two parts into account. Bar-On et al. [3] demonstrated the importance of studying the junction between differential and linear trails carefully, as optimizing the two trails independently may not necessarily result in the distinguisher with the highest correlation. While recent years have seen the emergence of several automated tools focused on identifying optimal boomerang/rectangle distinguishers [15,21,30,33] and even the most effective full boomerang/rectangle attacks [25,48] across various classes of primitives, there is a noticeable gap in research dedicated to exploring differential-linear distinguishers and attacks. Only very recently, Bellini et al. [7] introduced an automated tool based on Mixed-Integer Linear Programming (MILP) and Mixed-Integer Quadratic Constraint Programming (MIQCP) for identifying DL distinguishers in ARX ciphers and applied it to Speck-32. However, the authors themselves acknowledge that their Constraint Programming (CP) model is resource-intensive, limiting its application to smaller variants of Speck like Speck-32. Furthermore, the tool's efficiency with respect to SPN ciphers remains an open question. In another recent work from ASIACRYPT 2023, Chen et al. [18] proposed an alternative method for searching for DL distinguishers. However, their approach is also tailored to ARX ciphers and does not leverage general-purpose CP/MILP solvers. Developing an efficient and generic approach to automatically search for good DL distinguishers of different classes of primitives, particularly SPN designs, remains an open problem (see Section A for a detailed comparison).

Our contributions. This work addresses two important research gaps in the context of DL cryptanalysis. First, we address an open problem proposed at EUROCRYPT 2019 [3], which involves generalizing the DLCT framework to formalize the correlation of the middle part in the DL distinguisher when composed of multiple rounds. To achieve this, we explore the relations between DLCT, DDT, and LAT and propose a set of new tables that enable us to formulate the correlation of the middle part of the DL distinguisher across multiple rounds. Our approach is inspired by the generalized BCT framework in boomerang analysis and reveals a certain duality between the two frameworks. The advantage of our approach is that it allows us to leverage the insights gained in one framework for

the other. To demonstrate the utility of our new tables, we apply them to the DL distinguishers of various block ciphers to derive a formula for the correlation of the multiple-round middle part of the DL distinguisher. Then, as another contribution and inspired by advancements in boomerang analysis, we introduce an automated search method for DL distinguishers based on CP/MILP. This method is both generic and user-friendly, designed for searching good DL distinguishers across a broad spectrum of symmetric primitives. Our tool is applied to various primitives, including weakly aligned permutations (Ascon, KNOT), bit-sliced block ciphers (SERPENT, PRESENT), and AndRX designs (Simeck), as well as strongly aligned SPN block ciphers (AES, SKINNY) and Feistel ciphers (CLEFIA, TWINE, LBlock, WARP). In all applications, we are either the first to propose DL distinguishers or improve the best-known ones (see Table 1):

- For AES, we propose single-key DL distinguishers for up to 5 rounds for the first time.
- For Ascon, we propose a deterministic DL distinguisher for 4 rounds of the permutation, though it was considered not to have any deterministic DL distinguisher. We also introduce a new 5-round DL distinguisher with a correlation improved from 2^{-9} to $2^{-4.33}$.
- For SERPENT, we provide a new 9-round DL distinguisher with correlation improved from $2^{-56.50}$ to $2^{-50.95}$.
- For TWINE, CLEFIA, LBlock, and WARP, we propose DL distinguishers for the first time and show that in some cases the DL distinguisher can perform much better than boomerang distinguishers. For instance, we found a 17-round DL distinguisher for TWINE, which is 1 round longer than all distinguishers proposed so far. We also found a 17-round DL distinguisher for LBlock and LBlock-s exceeding its boomerang distinguisher by 1 round.
- For KNOT-256, we provide new DL distinguishers reaching up to 23 rounds of the permutation, whereas the best previous DL distinguisher is a 15-round conditional DL distinguisher [56].
- For SKINNY, we found DL distinguishers in both single-tweakey and related-tweakey settings for the first time. We introduce 15-round (resp. 17-round) related-tweakey distinguishers for SKINNY-64-128 (resp. SKINNY-64-192) that are 1 round longer than the best related-tweakey distinguishers of SKINNY with only 2 related tweakeys. Note that boomerang distinguishers of SKINNY require at least 4 related tweakeys.
- For Simeck, we significantly improve the DL distinguishers of all variants. Notably, we improve the DL distinguishers of Simeck-64, and Simeck-48 by 1 and 2 rounds, respectively.

The tool's source code is available at `https://github.com/hadipourh/DL`. Most of the distinguishers reported in this paper can be obtained within minutes, or even seconds, on a standard laptop.

Outline. In Section 2, we revisit the basics of differential-linear cryptanalysis. In Section 3, we extend the `DLCT` framework and introduce new connectivity tables to formalize the correlation of a DL distinguisher. In Section 4, we first

Table 1: Summary of our DL distinguishers. $\#R$: number of rounds, $\mathbb{C}$: correlation, S/RTK: Single/Related-Tweakey. 💻: Experimental verification.

Cipher	#R	$\mathbb{C}$	💻	Ref.
Ascon-p	4	2^{-1}		[22]
	4	$\mathbf{1}$	✓	D.2
	5	2^{-9}		[22]
	5	$\mathbf{2^{-4.33}}$	✓	D.2
AES	2	$\mathbf{1}$	✓	C.2
	3	$\mathbf{2^{-7.66}}$	✓	C.2
	4	$2^{-31.66}$		C.2
	5	$2^{-55.66}$		C.2
SERPENT	3	$\mathbf{2^{-0.68}}$	✓	E.2
	4	$2^{-12.75}$		[26]
	4	$\mathbf{2^{-5.54}}$	✓	E.2
	5	$2^{-16.75}$		[26]
	5	$\mathbf{2^{-11.10}}$	✓	E.2
	8	$2^{-39.18}$		E.2
	9	$2^{-56.50}$		[26]
	9	$\mathbf{2^{-50.95}}$		E.2
SKINNY (STK)	6	$\mathbf{1}$	✓	F.2
	9	$\mathbf{2^{-10.86}}$	✓	F.2
	10	$2^{-19.72}$		F.2
	11	$2^{-26.36}$		F.2
SKINNY-64-128 (RTK)	8	$\mathbf{1}$	✓	F.2
	14	$2^{-23.03}$		F.2
	15	$2^{-28.72}$		F.2
SKINNY-64-192 (RTK)	10	$\mathbf{1}$	✓	F.2
	16	$2^{-20.57}$		F.2
	17	$2^{-27.59}$		F.2
WARP	11	$\mathbf{1}$	✓	K.2
	14	$\mathbf{2^{-6}}$	✓	K.2
	15	$\mathbf{2^{-8}}$	✓	K.2
	16	$\mathbf{2^{-11}}$	✓	K.2
	17	2^{-15}		K.2
	18	2^{-19}		K.2
	19	2^{-25}		K.2
	20	2^{-31}		K.2
	22	2^{-51}		K.2
CLEFIA	4	$\mathbf{1}$	✓	L.2
	6	$\mathbf{2^{-7.07}}$	✓	L.2
	7	$\mathbf{2^{-11.75}}$	✓	L.2
	8	$2^{-33.43}$		L.2
	9	$2^{-55.29}$		L.2

Cipher	#R	$\mathbb{C}$	💻	Ref.
Simeck-32	7	$\mathbf{1}$	✓	I.2
	14	$2^{-16.63}$		[61]
	14	$\mathbf{2^{-13.92}}$	✓	I.2
Simeck-48	8	$\mathbf{1}$	✓	I.2
	17	$2^{-22.37}$		[61]
	17	$\mathbf{2^{-13.89}}$	✓	I.2
	18	$2^{-24.75}$		[61]
	18	$\mathbf{2^{-15.89}}$		I.2
	19	$\mathbf{2^{-17.89}}$		I.2
	20	$\mathbf{2^{-21.89}}$		I.2
Simeck-64	10	$\mathbf{1}$	✓	I.2
	24	$2^{-38.13}$		[61]
	24	$\mathbf{2^{-25.14}}$		I.2
	25	$2^{-41.04}$		[61]
	25	$\mathbf{2^{-27.14}}$		I.2
	26	$\mathbf{2^{-30.35}}$		I.2
KNOT-256	8	$\mathbf{1}$	✓	H.2
	15	$\mathbf{2^{-17.20}}$		H.2
	23	$2^{-58.88}$		H.2
PRESENT	3	$\mathbf{1}$	✓	G.2
	9	$\mathbf{2^{-9.23}}$	✓	G.2
	12	$2^{-23.77}$		G.2
	13	$2^{-27.01}$		G.2
TWINE	7	$\mathbf{1}$	✓	M.2
	10	$\mathbf{2^{-7.85}}$	✓	M.2
	12	$\mathbf{2^{-11.49}}$	✓	M.2
	16	$2^{-23.64}$		M.2
	17	$2^{-29.62}$		M.2
LBlock-s	7	$\mathbf{1}$	✓	J.2
	11	$\mathbf{2^{-7.42}}$	✓	J.2
	12	$\mathbf{2^{-9.42}}$	✓	J.2
	13	$\mathbf{2^{-12.10}}$	✓	J.2
	16	$2^{-22.80}$		J.2
	17	$2^{-28.80}$		J.2
LBlock	7	$\mathbf{1}$	✓	J.2
	11	$\mathbf{2^{-8}}$	✓	J.2
	12	$\mathbf{2^{-10}}$	✓	J.2
	13	$\mathbf{2^{-11.89}}$	✓	J.2
	16	$2^{-23.13}$		J.2
	17	$2^{-29.15}$		J.2

discuss in more detail the DLCT and switches connecting both trails to give an intuition about the effective parameters when searching for DL distinguishers. Finally, we present our CP/MILP model and its results on several primitives. For detailed results including detailed cipher specifications as well as an introduction to constraint satisfaction and constraint optimization problems (CSP and COP), we refer to the appendix.

2 Background

Here, we recall the basics of differential-linear analysis, and introduce the notations we use in the rest of this paper. For $\lambda, x \in \mathbb{F}_2^n$, we define the dot product of λ, x as $\sum_{i=0}^{n-1} \lambda[i] \cdot x[i]$, where $x[i]$ (resp. $\lambda[i]$) represents the ith bit of x (resp. λ). Given a set $\mathcal{S} \subseteq \mathbb{F}_2^n$ and a Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$, the correlation of f over the sample space $\mathcal{S}$ is defined as

$$\begin{aligned}\mathbb{C}_{\mathcal{S}}(f) \coloneqq \frac{1}{|\mathcal{S}|} \sum_{X \in \mathcal{S}} (-1)^{f(X)} &= \mathbb{P}\left(f(X) = 0 \mid X \in \mathcal{S}\right) - \mathbb{P}\left(f(X) = 1 \mid X \in \mathcal{S}\right) \\ &= 2 \cdot \mathbb{P}\left(f(X) = 0 \mid X \in \mathcal{S}\right) - 1,\end{aligned}$$

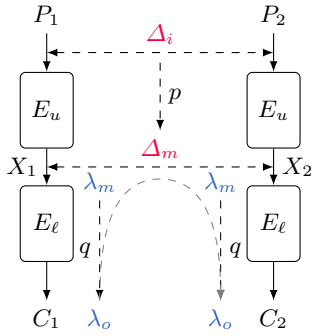
where $\mathbb{P}$ denotes the probability when X is chosen uniformly at random from $\mathcal{S}$.

Differential Cryptanalysis. Differential cryptanalysis, first introduced by Biham and Shamir [12], exploits the propagation of difference between two plaintexts through the encryption algorithm. We use $\Delta_i \xrightarrow{p} \Delta_o$ to denote a differential transition through the block cipher E with probability $p = \mathbb{P}(C_1 \oplus C_2 = \Delta_o \mid P_1 \oplus P_2 = \Delta_i)$, where $C_i = E\left(P_i\right)$ for $i \in \{0, 1\}$. Sometimes, we use the compact notation $\mathbb{P}\left(\Delta_i, \Delta_o\right)$ to denote the probability. Differential cryptanalysis relies on the fact that $\Delta_i \xrightarrow{p} \Delta_o$ may have a non-negligible probability for some Δ_i, Δ_o.

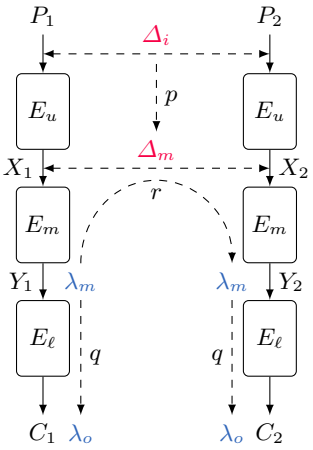
Linear Cryptanalysis. Linear cryptanalysis, first introduced by Matsui [43], exploits biased linear approximations connecting a plaintext with its ciphertext. We use $\lambda_i \xrightarrow{q} \lambda_o$ to denote a linear approximation with correlation $q = \mathbb{C}\left(\lambda_i \cdot P \oplus \lambda_o \cdot C\right)$, where $C = E\left(P\right)$. Sometimes, we use the compact notation $\mathbb{C}\left(\lambda_i, \lambda_o\right)$ to denote the correlation. Linear cryptanalysis relies on the fact that $\lambda_i \xrightarrow{q} \lambda_o$ may have a non-negligible correlation for some λ_i, λ_o.

Differential-Linear Cryptanalysis. In 1994, Langford and Hellman [38] merged concepts from both differential and linear cryptanalysis, giving rise to differential-linear cryptanalysis. The main concept of DL analysis involves merging a high-probability short differential transition with a high-correlation short linear approximation, aiming to create a long distinguisher for the block cipher. Later, in 1999, Wagner proposed the boomerang attack [55] that combines two high-probability short differential transitions to build a longer distinguisher. Combined attacks, like DL and boomerang attacks, highlight that the absence of long, high-probability differentials or high-correlation linear approximations does not

guarantee the cipher's security against differential or linear cryptanalysis. The presence of short, high-probability differentials and highly correlated linear approximations can potentially make the cipher susceptible to DL or boomerang attacks, as exemplified by the full-round DL attack on COCONUT98 in [10].



(a) A DL distinguisher.



(b) A DL distinguisher with a middle part.

Fig. 1: The structure of DL distinguishers.

Assume that we decompose the block cipher E, which has a block size of n bits, into two parts E_u and E_ℓ according to Figure 1a. Additionally, assume that we have a differential transition $\Delta_i \xrightarrow{p} \Delta_m$ through E_u and a linear approximation $\lambda_m \xrightarrow{q} \lambda_o$ through E_ℓ. To distinguish E from a random permutation, we encrypt a pair of plaintexts (P_1, P_2) with difference Δ_i and for each pair, we check if the corresponding ciphertexts (C_1, C_2) satisfy the linear approximation $\lambda_o \cdot C_1 = \lambda_o \cdot C_2$.

As can be seen in Figure 1a, the linear approximation $\lambda_o \cdot C_1 \oplus \lambda_o \cdot C_2$ can be rewritten as a combination of three linear approximations:

$$\lambda_o \cdot C_1 \oplus \lambda_o \cdot C_2 = (\lambda_o \cdot C_1 \oplus \lambda_m \cdot X_1) \oplus (\lambda_m \cdot X_1 \oplus \lambda_m \cdot X_2) \oplus (\lambda_m \cdot X_2 \oplus \lambda_o \cdot C_2)\,.$$

If we assume that the linear approximations are independent, then using Matsui's Piling-up lemma [43], the correlation of the linear approximation $\lambda_o \cdot C_1 \oplus \lambda_o \cdot C_2$ is the multiplication of the correlations of the three linear approximations, i.e., $\mathbb{C}\,(\lambda_m \cdot \Delta X) \cdot q^2$, where $\Delta X = X_1 \oplus X_2$. If we also assume that $\lambda_m \cdot \Delta X = 0$

holds in half of the cases when $\Delta X \neq \Delta_m$, then we have[3]:

$$\mathbb{P}\left(\lambda_m \cdot \Delta X = 0\right) = \mathbb{P}\left(\lambda_m \cdot \Delta X = 0 \mid \Delta X = \Delta_m\right) \cdot p + \mathbb{P}\left(\lambda_m \cdot \Delta X = 0 \mid \Delta X \neq \Delta_m\right) \cdot (1-p) = \frac{1}{2} \pm \frac{p}{2}.$$

Hence, $\mathbb{C}\left(\lambda_m \cdot \Delta X\right) = (-1)^{\lambda_m \cdot \Delta_m} p = \pm p$, implying that the correlation of the linear approximation $\lambda_o \cdot C_1 \oplus \lambda_o \cdot C_2$ is $\pm pq^2$. Consequently, if pq^2 is large enough, we can distinguish E from a random permutation with $\mathcal{O}\left(p^{-2}q^{-4}\right)$ chosen plaintexts. Nevertheless, this computation relies on two key assumptions: (1) The subciphers E_u and E_ℓ are statistically independent, and (2) $\lambda_m \cdot \Delta X = 0$ holds in half of the cases when $\Delta X \neq \Delta_m$.

However, subsequent works revealed that the above assumptions may not hold in practice. For instance, Biham et al. [10] demonstrated that the second assumption can fail in numerous cases and proposed experimental verification as a workaround. Later, Blondeau et al. [13] provided an exact expression of the correlation for DL distinguisher in a closed form without the second assumption. Based on Theorem 2 in [13], under the sole assumption that the two parts are independent, the correlation of the DL distinguisher is (see Figure 1a):

$$\mathbb{C}\left(\lambda_o \cdot C_1 \oplus \lambda_o \cdot C_2\right) = \sum_{\Lambda X \in \mathbb{F}_2^n} \mathbb{C}\left(\Lambda X \cdot \left(E_u(P) \oplus E_u(P \oplus \Delta_i)\right)\right) \cdot \mathbb{C}^2(\Lambda X, \lambda_o), \quad (1)$$

where $\mathbb{C}\left(\Lambda X, \lambda_o\right) = \mathbb{C}\left(\Lambda X \cdot X \oplus \lambda_o \cdot E_\ell(X)\right)$, with X chosen uniformly at random from $\mathbb{F}_2^n$.

While the dependency between the two parts (i.e., E_u and E_ℓ) was not addressed for a long time, it was studied in the context of another differential-based combined attack, i.e., boomerang attack. Dunkelman et al. [27] proposed the idea of the sandwich framework to consider the dependency between the two parts of the boomerang distinguisher. This framework is also applicable to DL attacks. In the sandwich framework, we divide the block cipher E into three parts: E_u, E_m, and E_ℓ as illustrated in Figure 1b. While we handle E_u and E_ℓ as standard differential and linear distinguishers, respectively, we treat E_m as a compact combined distinguisher connecting the two segments (refer to Figure 1b).

Assume that $\mathbb{R}\left(\Delta X, \Lambda Y\right) = \mathbb{C}\left(\Lambda Y \cdot \left(E_m(X) \oplus E_m(X \oplus \Delta X)\right)\right)$ in Figure 1b, where X is chosen uniformly at random from $\mathbb{F}_2^n$. Then, the correlation of the DL distinguisher is given by [3]:

$$\mathbb{C}\left(\lambda_o \cdot C_1 \oplus \lambda_o \cdot C_2\right) = \sum_{\Delta X, \Lambda Y} \mathbb{P}\left(\Delta_i, \Delta X\right) \cdot \mathbb{R}(\Delta X, \Lambda Y) \cdot \mathbb{C}^2(\Lambda Y, \lambda_o), \quad (2)$$

where $\mathbb{P}(\Delta_i, \Delta X) = \mathbb{P}(E_u(P) \oplus E_u(P \oplus \Delta_i) = \Delta X)$, with P chosen uniformly at random from $\mathbb{F}_2^n$, and $\mathbb{C}(\Lambda Y, \lambda_o) = \mathbb{C}(\Lambda Y \cdot E_\ell(Y) \oplus \lambda_o \cdot E_\ell(Y))$, with Y chosen uniformly at random from $\mathbb{F}_2^n$.

The fundamental formula for DL distinguishers in Equation 2 does not rely on the two assumptions we mentioned earlier, but computing $\mathbb{P}, \mathbb{R}$, and $\mathbb{C}$ still relies

[3] For a fixed value of λ_m and Δ_m, $\mathbb{P}\left(\lambda_m \cdot \Delta X = 0 \mid \Delta X = \Delta_m\right)$ is either 1 or 0.

on the round independence assumption within E_u, E_m, and E_ℓ. Although the round independence assumption may not precisely hold in practice, it enables us to provide a good approximation for the probability (resp. correlation) of differential (resp. linear) transitions averaged over a large set of random keys.

Given the input difference Δ_i, and the output mask λ_o for a block cipher E, assume that for a fixed $\Delta X = \Delta_m$ and $\Lambda Y = \lambda_m$, $p = \mathbb{P}(\Delta_i, \Delta_o)$, $r = \mathbb{R}(\Delta_m, \lambda_m)$, and $q = \mathbb{C}(\lambda_m, \lambda_o)$. If we decompose E appropriately into three segments: E_i, E_m, and E_o, and if Δ_m and λ_m are good choices, then prq^2 gives a good estimation for the actual correlation of the DL distinguisher with input difference Δ_i, and output mask λ_o. We elaborate on the appropriate decomposition of E and the good choices for Δ_m and λ_m in Section 4.

While the probability of a differential (resp. correlation of a linear approximation) through E_u (resp. E_ℓ) is computed using the DDT (resp. LAT) framework, calculating the correlation of the combined distinguisher over E_m is not a straightforward task. The middle term in Equation 2, namely $\mathbb{R}(\Delta_m, \lambda_m)$, is typically determined through experimental means, involving the encryption of a sufficiently large set of random plaintexts with E_m. More recently, Bar-On et al. [3] introduced the DLCT framework to formalize the correlation of the small combined distinguisher over E_m, when it is composed of only one S-box layer. However, they left extending the DLCT framework to cover more rounds at the boundary between E_u and E_ℓ as an open problem. In Section 3, we show how to extend it for multiple rounds.

3 Generalizing the DLCT Framework

In this section, we extend the DLCT framework to handle multiple rounds in the middle part E_m. Recent advances in boomerang analysis [21,30] have enabled us to formulate the probability of boomerang distinguishers using a generalized BCT framework, providing some basic rules for modeling the involved S-boxes with different BCT tables. Inspired by boomerang analysis, we introduce the alternative of generalized BCT [19] tables to the DLCT framework. Our main motivation is to apply the same technique for DL distinguishers, especially providing a basic guideline for modeling involved S-boxes with certain tables. To this end, we first review the DDT, LAT, and DLCT definitions. Then, we introduce new tables akin to those in [21,30], and demonstrate their use in extending the DLCT framework. Finally, we provide examples to illustrate how these new tables can effectively formulate the correlation of DL distinguishers across multiple rounds.

Definition 1 (Differential Distribution Table (DDT)). *For a vectorial Boolean function $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$, the DDT is a $2^n \times 2^m$ table whose rows correspond to the input difference Δ_i to S and whose columns correspond to the output difference Δ_o of S. The entry at index (Δ_i, Δ_o) is*

$$DDT(\Delta_i, \Delta_o) \coloneqq |\{x \in \mathbb{F}_2^n : \ S(x) \oplus S(x \oplus \Delta_i) = \Delta_o\}|.$$

Definition 2 (Differential Uniformity [46]). *The differential uniformity of an S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is defined as $\mathcal{DU}(S) \coloneqq \max_{\Delta_i \in \mathbb{F}_2^n \setminus \{0\}, \Delta_o \in \mathbb{F}_2^m} DDT(\Delta_i, \Delta_o)$.*

Definition 3 (Linear Approximation Table (LAT)). *For a vectorial Boolean function $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$, the LAT of S is a $2^n \times 2^m$ table whose rows correspond to the input mask λ_i to S and whose columns correspond to the output mask λ_o of S. The entry at index (λ_i, λ_o) is*

$$\mathit{LAT}(\lambda_i, \lambda_o) \coloneqq |\mathit{LAT}_0(\lambda_i, \lambda_o)| - |\mathit{LAT}_1(\lambda_i, \lambda_o)|,$$

where $\mathit{LAT}_b(\lambda_i, \lambda_o) = \{x \in \mathbb{F}_2^n : \ \lambda_i \cdot x \oplus \lambda_o \cdot S(x) = b\}$.

It can be seen that $\mathbb{C}\left(\lambda_i \cdot x \oplus \lambda_o \cdot S(x)\right) = \mathtt{LAT}(\lambda_i, \lambda_o)/2^n$.

Definition 4 (Linearity [46]). *The linearity of an S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is defined as $\mathcal{L}(S) \coloneqq \max_{\lambda_i \in \mathbb{F}_2^n, \lambda_o \in \mathbb{F}_2^m \setminus \{0\}} |\mathit{LAT}(\lambda_i, \lambda_o)|$.*

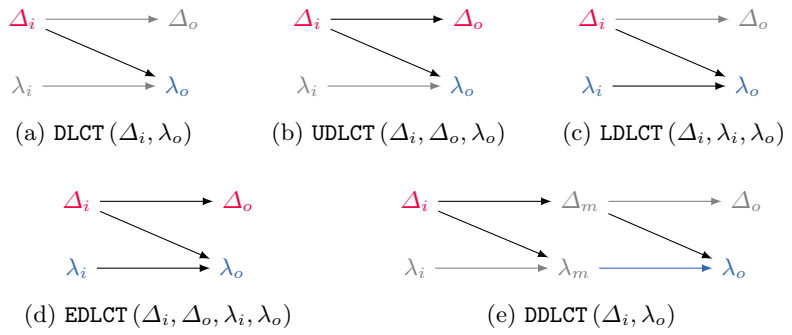
Definition 5 (Differential-Linear Connectivity Table (DLCT) [3]). *For a vectorial Boolean function $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$, the DLCT of S is a $2^n \times 2^m$ table whose rows correspond to the input difference Δ_i to S and whose columns correspond to the output mask λ_o of S. The entry at index (Δ_i, λ_o) is*[4]

$$\mathit{DLCT}(\Delta_i, \lambda_o) = |\mathit{DLCT}_0(\Delta_i, \lambda_o)| - |\mathit{DLCT}_1(\Delta_i, \lambda_o)|,$$

where $\mathit{DLCT}_b(\Delta_i, \lambda_o) = \{x \in \mathbb{F}_2^n : \ \lambda_o \cdot S(x) \oplus \lambda_o \cdot S(x \oplus \Delta_i) = b\}$ (see Figure 2a).

Sometimes we use the *normalized* DLCT, which is defined as $\overline{\mathtt{DLCT}}(\Delta_i, \lambda_o) \coloneqq \mathtt{DLCT}(\Delta_i, \lambda_o)/2^n$, and is equal to $\mathbb{C}(\lambda_o \cdot S(x) \oplus \lambda_o \cdot S(x \oplus \Delta_i))$.

Definition 6 (Differential-Linear Uniformity). *The differential-linear uniformity of an S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is defined as $\mathcal{DLU}(S) \coloneqq \max_{\Delta_i, \lambda_o \neq 0} |\mathit{DLCT}(\Delta_i, \lambda_o)|$.*



(a) $\mathtt{DLCT}\left(\Delta_i, \lambda_o\right)$ (b) $\mathtt{UDLCT}\left(\Delta_i, \Delta_o, \lambda_o\right)$ (c) $\mathtt{LDLCT}\left(\Delta_i, \lambda_i, \lambda_o\right)$

(d) $\mathtt{EDLCT}\left(\Delta_i, \Delta_o, \lambda_i, \lambda_o\right)$ (e) $\mathtt{DDLCT}\left(\Delta_i, \lambda_o\right)$

Fig. 2: Tables of the generalized DLCT framework.

As in boomerang distinguishers, we can extend the definition of the DLCT to the cases where the output difference and/or the input mask are also specified. This leads to the following definitions:

[4] Our definition is slightly different from [3], where DLCT is defined as $|\mathtt{DLCT}_0| - 2^{n-1}$.

Definition 7 (Upper Differential-Linear Connectivity Table (UDLCT)). *For a vectorial Boolean function $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$, the $\mathtt{UDLCT}$ of S is a $2^n \times 2^m \times 2^m$ table. The entry at index $(\Delta_i, \Delta_o, \lambda_o)$ is (see Figure 2b)*

$$\mathtt{UDLCT}(\Delta_i, \Delta_o, \lambda_o) = |\mathtt{UDLCT}_0(\Delta_i, \Delta_o, \lambda_o)| - |\mathtt{UDLCT}_1(\Delta_i, \Delta_o, \lambda_o)|,$$

where $\mathtt{UDLCT}_b(\Delta_i, \Delta_o, \lambda_o) = \{x \in \mathbb{F}_2^n : \ S(x) \oplus S(x \oplus \Delta_i) = \Delta_o \text{ and } \lambda_o \cdot \Delta_o = b\}$.

Definition 8 (Lower Differential-Linear Connectivity Table (LDLCT)). *For a vectorial Boolean function $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$, the $\mathtt{LDLCT}$ of S is a $2^n \times 2^n \times 2^m$ table. The entry at index $(\Delta_i, \lambda_i, \lambda_o)$ is (see Figure 2c)*

$$\mathtt{LDLCT}(\Delta_i, \lambda_i, \lambda_o) = |\mathtt{LDLCT}_0(\Delta_i, \lambda_i, \lambda_o)| - |\mathtt{LDLCT}_1(\Delta_i, \lambda_i, \lambda_o)|,$$

where $\mathtt{LDLCT}_b(\Delta_i, \lambda_i, \lambda_o) = \{x \in \mathbb{F}_2^n : \ \lambda_i \cdot \Delta_i \oplus \lambda_o \cdot S(x) \oplus \lambda_o \cdot S(x \oplus \Delta_i) = b\}$.

Finally, we define the table corresponding to the case where all inputs and outputs are specified.

Definition 9 (Extended Differential-Linear Connectivity Table (EDLCT)). *For a vectorial Boolean function $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$, the $\mathtt{EDLCT}$ of S is a $2^n \times 2^n \times 2^m \times 2^m$ table. The entry at index $(\Delta_i, \Delta_o, \lambda_i, \lambda_o)$ is (see Figure 2d)*

$$\mathtt{EDLCT}(\Delta_i, \Delta_o, \lambda_i, \lambda_o) = |\mathtt{EDLCT}_0(\Delta_i, \Delta_o, \lambda_i, \lambda_o)| - |\mathtt{EDLCT}_1(\Delta_i, \Delta_o, \lambda_i, \lambda_o)|,$$

where $\mathtt{EDLCT}_b(\Delta_i, \Delta_o, \lambda_i, \lambda_o) = \{x \in \mathbb{F}_2^n : \ S(x) \oplus S(x \oplus \Delta_i) = \Delta_o \text{ and } \lambda_i \cdot \Delta_i \oplus \lambda_o \cdot \Delta_o = b\}$.

3.1 Table Properties

Here, we explore the relationship between the introduced tables, both among themselves and with the DDT and LAT tables.

Proposition 1. *The generalized $\mathtt{DLCT}$ tables satisfy the following properties:*

1. $\mathtt{DLCT}(0, \lambda_o) = \mathtt{DLCT}(\Delta_i, 0) = 2^n, \ \forall \Delta_i, \lambda_o$
2. $\mathtt{DLCT}(\Delta_i, \lambda_o) = \sum_{\Delta_o} \mathtt{UDLCT}(\Delta_i, \Delta_o, \lambda_o)$
3. $\mathtt{UDLCT}(\Delta_i, \Delta_o, \lambda_o) = (-1)^{\lambda_o \cdot \Delta_o} \mathtt{DDT}(\Delta_i, \Delta_o)$
4. $\mathtt{LDLCT}(\Delta_i, \lambda_i, \lambda_o) = (-1)^{\lambda_i \cdot \Delta_i} \mathtt{DLCT}(\Delta_i, \lambda_o)$
5. $\mathtt{EDLCT}(\Delta_i, \Delta_o, \lambda_i, \lambda_o) = (-1)^{\lambda_i \cdot \Delta_i \oplus \lambda_o \cdot \Delta_o} \mathtt{DDT}(\Delta_i, \Delta_o)$
6. $\mathtt{LDLCT}(\Delta_i, \lambda_i, \lambda_o) = \sum_{\Delta_o} \mathtt{EDLCT}(\Delta_i, \Delta_o, \lambda_i, \lambda_o)$
7. $\sum_{\Delta_i} \mathtt{LDLCT}(\Delta_i, \lambda_i, \lambda_o) = \mathtt{LAT}^2(\lambda_i, \lambda_o)$

Proof. The last property is the only non-trivial one to prove. Let λ_i and λ_o be two masks, and let A_b^{Δ} be the set $\{x \in \mathbb{F}_2^n : \ \lambda_i \cdot (x \oplus \Delta) \oplus \lambda_o \cdot S(x \oplus \Delta) = b\}$.

We then observe the following equalities:

$$\begin{aligned}
\sum_{\Delta_i} \texttt{LDLCT}(\Delta_i, \lambda_i, \lambda_o) &= \sum_{\Delta_i} |A_0^0 \cap A_0^{\Delta_i}| + |A_1^0 \cap A_1^{\Delta_i}| - |A_0^0 \cap A_1^{\Delta_i}| - |A_1^0 \cap A_0^{\Delta_i}| \\
&= \sum_{\Delta_i} \sum_{x \in A_0^0} |\{x\} \cap A_0^{\Delta_i}| - |\{x\} \cap A_1^{\Delta_i}| \\
&\qquad + \sum_{\Delta_i} \sum_{x \in A_1^0} |\{x\} \cap A_1^{\Delta_i}| - |\{x\} \cap A_0^{\Delta_i}| \\
&= \sum_{x \in A_0^0} \sum_{\Delta_i \oplus x} |\{x\} \cap A_0^{\Delta_i \oplus x}| - |\{x\} \cap A_1^{\Delta_i \oplus x}| \\
&\qquad + \sum_{x \in A_1^0} \sum_{\Delta_i \oplus x} |\{x\} \cap A_1^{\Delta_i \oplus x}| - |\{x\} \cap A_0^{\Delta_i \oplus x}| \\
&= \sum_{x \in A_0^0} \Big(\sum_{\Delta_i \in A_0^0} 1 - \sum_{\Delta_i \in A_1^0} 1 \Big) + \sum_{x \in A_1^0} \Big(\sum_{\Delta_i \in A_1^0} 1 - \sum_{\Delta_i \in A_0^0} 1 \Big) \\
&= |A_0^0|^2 - 2|A_0^0||A_1^0| + |A_1^0| = (|A_0^0| - |A_1^0|)^2 = \texttt{LAT}^2(\lambda_i, \lambda_o)
\end{aligned}$$

As observed, the new DLCT tables differ entirely from their counterparts in the BCT framework for boomerang analysis. In fact, each of them is equivalent, differing only in sign and a scalar factor, to either the DDT or the DLCT.

In many cases, there are two consecutive active S-boxes in the middle part of the DL distinguishers (also referred to as *DL switch*), potentially accompanied by key additions and linear layers in between. To efficiently address this scenario, we introduce the DDLCT table, favoring memory usage over time to effectively reduce the overall time complexity of evaluating DL distinguisher correlations.

Definition 10 (Double Differential-Linear Connectivity Table (DDLCT)). *For a vectorial Boolean function $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$, the DDLCT of S is a $2^n \times 2^m$ table such that the entry at index (Δ_i, λ_o) is (see Figure 2e)*

$$\mathit{DDLCT}(\Delta_i, \lambda_o) = 2^{-n} \sum_{\Delta_m} \sum_{\lambda_m} \mathit{UDLCT}(\Delta_i, \Delta_m, \lambda_m) \cdot \mathit{LDLCT}(\Delta_m, \lambda_m, \lambda_o) \tag{3}$$

Proposition 2. *The DDLCT table satisfies the following properties:*

$$\begin{aligned}
\mathit{DDLCT}(\Delta_i, \lambda_o) &= \sum_{\Delta_m} \mathit{DDT}(\Delta_i, \Delta_m) \cdot \mathit{DLCT}(\Delta_m, \lambda_o) \\
&= 2^{-n} \sum_{\lambda_m} \mathit{DLCT}(\Delta_i, \lambda_m) \cdot \mathit{LAT}^2(\lambda_m, \lambda_o).
\end{aligned}$$

Proof. The goal is to prove the equalities:

$$\texttt{DDLCT}(\Delta_i, \lambda_o) = 2^{-n} \sum_{\lambda_m} \texttt{DLCT}(\Delta_i, \lambda_m) \cdot \texttt{LAT}^2(\lambda_m, \lambda_o) = \sum_{\Delta_m} \texttt{DDT}(\Delta_i, \Delta_m) \cdot \texttt{DLCT}(\Delta_m, \lambda_o).$$

This is done by relying on the 7 equalities given in Proposition 1 and at each step we will indicate which equality is use. We first rewrite $\texttt{LAT}^2$ using the LDLCT (7) and we obtain:

$$\sum_{\lambda_m} \texttt{DLCT}(\Delta_i, \lambda_m) \cdot \texttt{LAT}^2(\lambda_m, \lambda_o) = \sum_{\lambda_m, \Delta'_m} \texttt{DLCT}(\Delta_i, \lambda_m) \cdot \texttt{LDLCT}(\Delta'_m, \lambda_m, \lambda_o).$$

Now we rewrite the DLCT using the UDLCT (2) to get:

$$\sum_{\lambda_m, \Delta'_m, \Delta_m} \texttt{UDLCT}(\Delta_i, \Delta_m, \lambda_m) \cdot \texttt{LDLCT}(\Delta'_m, \lambda_m, \lambda_o).$$

We then simplify both the UDLCT and the LDLCT by using equalities 3 and 4:

$$\sum_{\lambda_m, \Delta'_m, \Delta_m} (-1)^{\lambda_m \cdot (\Delta_m \oplus \Delta'_m)} \texttt{DDT}(\Delta_i, \Delta_m) \cdot \texttt{DLCT}(\Delta'_m, \lambda_o).$$

This sum can be rewritten as:

$$\sum_{\Delta'_m, \Delta_m} \texttt{DDT}(\Delta_i, \Delta_m) \cdot \texttt{DLCT}(\Delta'_m, \lambda_o) \sum_{\lambda_m} (-1)^{\lambda_m \cdot (\Delta_m \oplus \Delta'_m)},$$

and thus, by evaluating the nested sum we finally reach:

$$\sum_{\Delta'_m, \Delta_m} \texttt{DDT}(\Delta_i, \Delta_m) \cdot \texttt{DLCT}(\Delta'_m, \lambda_o) \cdot 2^n \cdot \mathbb{1}_{\Delta_m = \Delta'_m}.$$

At this point we have obtained that the sum over Δ_m and Δ'_m can be reduced to a sum over $\Delta_m = \Delta'_m$, achieving the proof.

It is worth noting that our formulation for DDLCT aligns with Theorem 2 in [13], which addresses computing the correlation of DL distinguishers.

We emphasize that the DDLCT relies on the assumptions of round independence and round-key independence. Nevertheless, it significantly reduces the time complexity in evaluating correlations. Additionally, it can be employed to analyze the behavior of S-boxes against DL attacks. One can extend this approach by defining the Triple DLCT or t-DLCT to precompute correlations for different portions of the distinguisher.

Definition 11 (t-DLCT). *For a vectorial Boolean function $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$, the t-DLCT of S is a $2^n \times 2^m$ table such that the entry at index (Δ_i, λ_o) is*

$$t\text{-}\mathit{DLCT}(\Delta_i, \lambda_o) = \sum_{\Delta_m} \mathit{DDT}(\Delta_i, \Delta_m) \ \cdot \ (t-1)\text{-}\mathit{DLCT}(\Delta_m, \lambda_o)\,. \tag{4}$$

Definition 12. *Let $\Delta_i, \Delta_o, \lambda_i, \lambda_o \in \mathbb{F}_2^n$. We define the following quantities:*

$$\begin{aligned}
\mathbb{P}_{\mathit{DDT}}(\Delta_i, \Delta_o) &= \mathit{DDT}(\Delta_i, \Delta_o)/2^n & \mathbb{C}_{\mathit{LAT}}(\lambda_i, \lambda_o) &= \mathit{LAT}(\lambda_i, \lambda_o)/2^n \\
\mathbb{C}_{\mathit{UDLCT}}(\Delta_i, \Delta_o, \lambda_o) &= \mathit{UDLCT}(\Delta_i, \Delta_o, \lambda_o)/2^n & \mathbb{C}_{\mathit{DLCT}}(\Delta_i, \lambda_o) &= \mathit{DLCT}(\Delta_i, \lambda_o)/2^n \\
\mathbb{C}_{\mathit{LDLCT}}(\Delta_i, \lambda_i, \lambda_o) &= \mathit{LDLCT}(\Delta_i, \lambda_i, \lambda_o)/2^{2n} & \mathbb{C}_{\mathit{DDLCT}}(\Delta_i, \lambda_o) &= \mathit{DDLCT}(\Delta_i, \lambda_o)/2^{2n} \\
\mathbb{C}_{\mathit{EDLCT}}(\Delta_i, \Delta_o, \lambda_i, \lambda_o) &= \mathit{EDLCT}(\Delta_i, \Delta_o, \lambda_i, \lambda_o)/2^{2n} & \mathbb{C}_{t\text{-}\mathit{DLCT}}(\Delta_i, \lambda_o) &= t\text{-}\mathit{DLCT}(\Delta_i, \lambda_o)/2^{t \cdot n}
\end{aligned}$$

3.2 Practical Examples with the Generalized DLCT Framework

Now that we have defined all the tables related to DL distinguishers, let us explore some practical examples demonstrating how we can use these tables to accurately formulate the correlation of a DL distinguisher.

Example 1. We start by studying a simple example on 3-round AES as shown in Figure 3. The internal differences/masks have been replaced by variables to be as precise as possible regarding potential clusters.

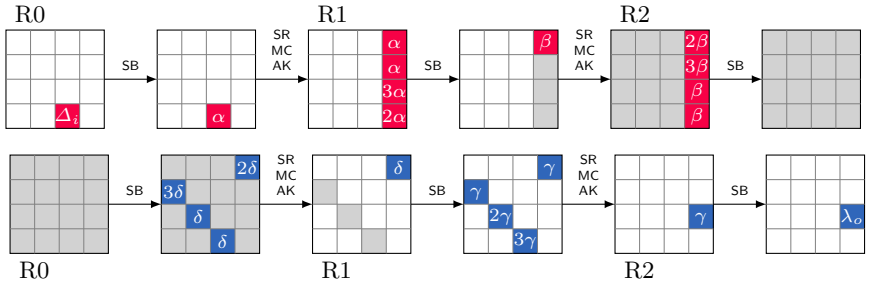


Fig. 3: DL distinguisher for 3-round AES in the single-key model. The upper trail represents the differential trail while the lower one represents the linear trail. White, blue/red, and gray colors indicate respectively that the difference/mask on the byte is null, fixed, and unknown.

The correlation of this DL distinguisher can be computed as follows:

$$\mathbb{C}\left(\Delta_i, \lambda_o\right) = \sum_{\alpha,\beta,\gamma,\delta} \mathbb{C}_{\texttt{UDLCT}}(\Delta_i, \alpha, \delta) \cdot \mathbb{C}_{\texttt{EDLCT}}(\alpha, \beta, \delta, \gamma) \cdot \mathbb{C}_{\texttt{LDLCT}}(\beta, \gamma, \lambda_o)$$

The terms of this formula are obtained by looking at each S-box cell through the encryption, exactly as for boomerang distinguishers. For instance, the term $\mathbb{C}_{\texttt{UDLCT}}(\Delta_i, \alpha, \delta)$ comes from the *red* cell in the first round since both the input and output of the differential transition and the output mask are set.

According to Proposition 1, all extended tables can be expressed using only DDT, LAT and DLCT. Thus the above formula can be rewritten as:

$$\mathbb{C}\left(\Delta_i, \lambda_o\right) = 2^{-5n} \sum_{\alpha,\beta} \texttt{DDT}(\Delta_i, \alpha) \cdot \texttt{DDT}(\alpha, \beta) \cdot \texttt{DLCT}(\beta, \lambda_o) \sum_{\gamma,\delta} (-1)^{\alpha\cdot\delta \oplus \alpha\cdot\delta \oplus \beta\cdot\gamma \oplus \beta\cdot\gamma}$$

The nested sum equals 2^{2n}, so the formula for the correlation is reduced to

$$\mathbb{C}\left(\Delta_i, \lambda_o\right) = 2^{-3n} \sum_{\alpha,\beta} \texttt{DDT}(\Delta_i, \alpha) \cdot \texttt{DDT}(\alpha, \beta) \cdot \texttt{DLCT}(\beta, \lambda_o). \tag{5}$$

According to Equation 5, we observe the maximum (absolute) correlation of $2^{-7.66}$ when $(\Delta_i, \lambda_o) = (\texttt{0xb4}, \texttt{0x67})$. Table 2 compares the theoretical value

obtained from Equation 5 with the experimental results, confirming the high accuracy of the formula. Using Equation 5, we computed the correlation for all possible input/output differences/masks. Figure 8 visualizes the correlation matrix for our 3-round distinguisher for AES. An interesting observation is that for all (Δ_i, λ_o) where Δ_i, and λ_o are nonzero, we have $\mathbb{C}(\Delta_i, \lambda_o) < 0$, indicating that the sign of the correlation remains unchanged.

Table 2: Theoretical vs. experimental results for the distinguisher in Figure 3.

Input/Output Differences/Linear-mask	Equation 5	Exp. Correlation
$(\Delta_i, \lambda_o) = (\texttt{0xb4}, \texttt{0x67})$	$-2^{-7.66}$	$-2^{-7.64}$
$(\Delta_i, \lambda_o) = (\texttt{0x02}, \texttt{0x02})$	$-2^{-7.92}$	$-2^{-7.93}$
$(\Delta_i, \lambda_o) = (\texttt{0x55}, \texttt{0x55})$	$-2^{-7.99}$	$-2^{-7.98}$
$(\Delta_i, \lambda_o) = (\texttt{0xbf}, \texttt{0xef})$	$-2^{-8.05}$	$-2^{-8.06}$
$(\Delta_i, \lambda_o) = (\texttt{0xfe}, \texttt{0x06})$	$-2^{-8.26}$	$-2^{-8.25}$
$(\Delta_i, \lambda_o) = (\texttt{0x4b}, \texttt{0x1a})$	$-2^{-8.43}$	$-2^{-8.44}$

Example 2. Figure 4 displays the 9-round middle part of a DL distinguisher for 13 rounds of TWINE, as depicted in Figure 64. In this distinguisher, we can divide the commonly active S-boxes into two groups of three consecutive S-boxes along with several key additions and linear layers in between. We highlight the two identified groups in ■ and ■ within Figure 4d. Each group of active S-boxes contributes a correlation value of $\mathbb{C}$, as defined in Equation 6, to the total correlation:

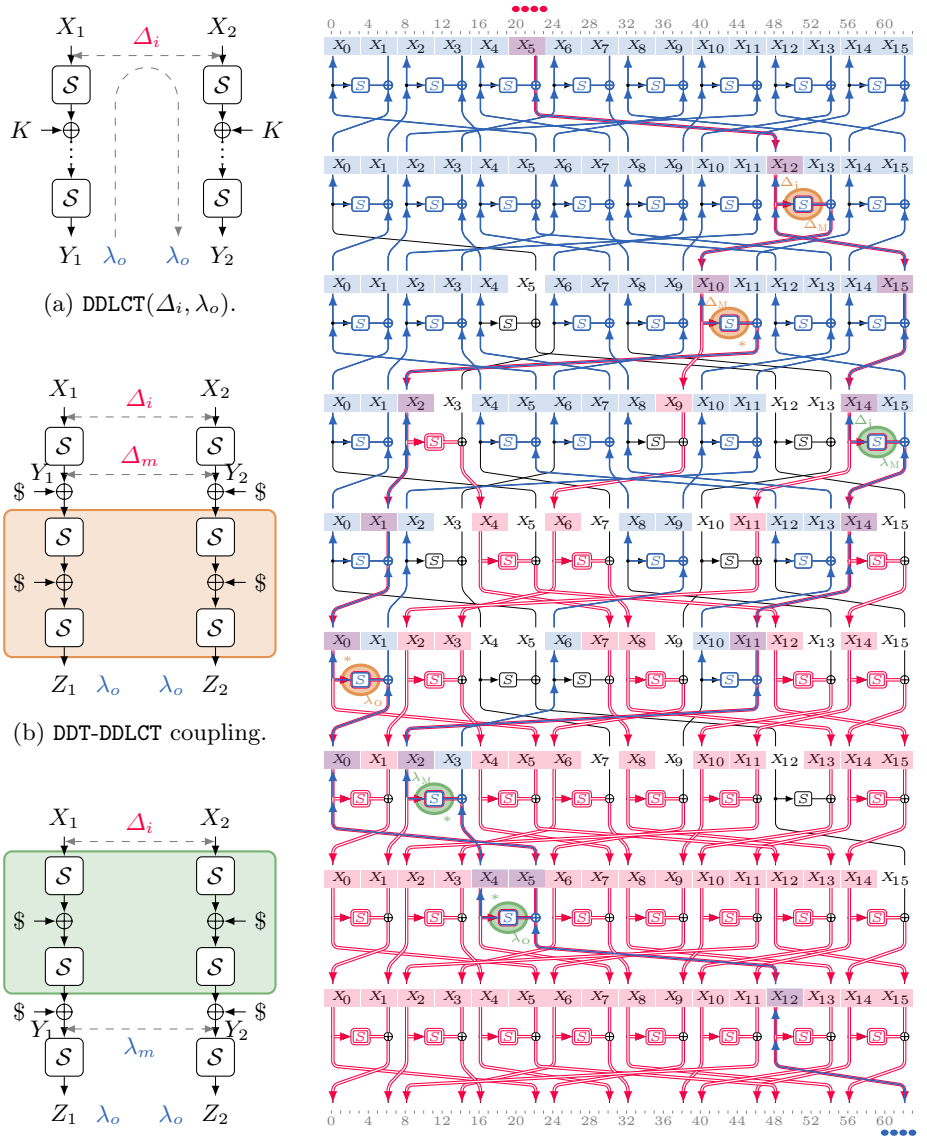
$$\mathbb{C}(\Delta_i, \lambda_o) = \sum_{\Delta_m} \mathbb{P}_{\texttt{DDT}}(\Delta_i, \Delta_m) \cdot \mathbb{C}_{\texttt{DDLCT}}(\Delta_m, \lambda_o) = \sum_{\lambda_m} \mathbb{C}_{\texttt{DDLCT}}(\Delta_i, \lambda_m) \cdot \mathbb{C}^2_{\texttt{LAT}}(\lambda_m, \lambda_o). \tag{6}$$

The two groups of commonly activated S-boxes, highlighted in ■ and ■, are independent. Therefore, the total correlation is $\mathbb{C}_t = \mathbb{C}^2$.

Table 3 provides a comparison between experimental and theoretical results for the correlation of the DL distinguisher depicted in Figure 4d. As demonstrated in Table 3, the outcomes derived from Equation 6 closely align with the experimental findings. Expressing the correlation is simpler using just one 3-DLCT: $\mathbb{C}_t(\Delta_i, \lambda_o) = \mathbb{C}^2_{\text{3-DLCT}}(\Delta_i, \lambda_o)$. Table 39 represents the 3-DLCT of TWINE's S-box, demonstrating that $(\Delta_i, \lambda_o) \in \{(\texttt{4}, \texttt{5}), (\texttt{5}, \texttt{a})\}$ yields the maximum correlation for the DL distinguisher.

3.3 Cell-wise Switches and Bit-wise Switches

The examples in Section 3.2 convey some key points. If an S-box is active in both differential and linear trails, referred to as *common* active S-box, it contributes to the correlation by a specific generalized DLCT table. However, if an S-box is



(a) DDLCT(Δ_i, λ_o).

(b) DDT-DDLCT coupling.

(c) DDLCT-LAT coupling.

(d) DL distinguisher for 9-round TWINE (— differential — linear).

Fig. 4: Application of t-DLCT to formulate the correlation of DL distinguishers.

active in at most one of the differential or linear trails, it does not contribute to the correlation of the distinguisher, allowing us to freely bypass it. We term these activeness patterns *ladder switches* or *cell-wise switches*. The cell-wise switches can also be explained by the maximal entries in the first row and the first column of all generalized DLCT tables. Moreover, the DLCT of many S-boxes also

Table 3: Theoretical vs. experimental results for the distinguisher in Figure 4d.

Input/Output Differences/Linear-mask	Equation 6	Exp. Correlation
$(\Delta_i, \lambda_o) = (\mathtt{4}, \mathtt{5})$	$2^{-5.83}$	$2^{-5.80}$
$(\Delta_i, \lambda_o) = (\mathtt{a}, \mathtt{a})$	$2^{-6.39}$	$2^{-6.39}$
$(\Delta_i, \lambda_o) = (\mathtt{f}, \mathtt{c})$	$2^{-7.36}$	$2^{-7.31}$
$(\Delta_i, \lambda_o) = (\mathtt{e}, \mathtt{9})$	$2^{-8.39}$	$2^{-8.39}$
$(\Delta_i, \lambda_o) = (\mathtt{d}, \mathtt{a})$	$2^{-10.00}$	$2^{-9.98}$
$(\Delta_i, \lambda_o) = (\mathtt{f}, \mathtt{1})$	0	0

have maximal entries (2^n) in the middle rows/columns. For example, referring to Table 30, we can see that if $(\Delta_i, \lambda_o) = (2, 2)$, then $\mathbb{C}_{\mathtt{DLCT}}(\Delta_i, \lambda_o) = 1$. We refer to these bit-wise differential-linear transitions with correlation 1 as *bit-wise switches*. It is worth noting that according to Proposition 16 in [17], any Boolean component of degree 2 leads to such bit-wise switches.

The correlation of a DL distinguisher depends on three main factors: the probability of the differential transition over E_u (denoted by p), the correlation over the middle part E_m (denoted by r), and the squared correlation of the linear approximation over E_ℓ (denoted by q^2). We already know that p and q^2 are determined by the number of differentially and linearly active S-boxes through E_u and E_ℓ, respectively. Nevertheless, r depends on the number of common active S-boxes between the differential and linear trails over E_m. As a result, minimizing the number of differentially (rep. linearly) active S-boxes through E_u (resp. E_ℓ) while maximizing the cell-wise/bit-wise switches over E_m can yield a better distinguisher. Therefore, finding a good DL distinguisher is a non-trivial combinatorial optimization problem. CP/MILP-based methods have been shown to effectively solve optimization problems stemming from symmetric-key cryptanalysis. With a systematic CP/MILP-based method, we can take advantage of cell-wise/bit-wise switches during the search for DL distinguishers. If the middle part (E_m) includes only one S-box layer, the `DLCT` captures all the switches, and one can model all valid transitions through `DLCT` by CP/MILP constraints as done in [40]. However, what if the middle part includes much more than one round, e.g., 10 rounds? Section 4 addresses this research gap.

4 CP/MILP Model to Search for DL Distinguishers

Here, we describe how to model the problem of finding differential-linear distinguishers as a CP/MILP problem. We present two approaches: the *cell-wise* and *bit-wise* models. Our cell-wise model treats the S-boxes as black-boxes and only takes advantage of cell-wise switches. The cell-wise model offers simplicity in creation as well as solving, and as we will show, it performs very well for strongly aligned ciphers such as AES, SKINNY, CLEFIA, WARP, and TWINE. On ther other hand, our bit-wise model takes advantage of the internal structure

of S-boxes (generally non-linear operations) and is suitable for weakly aligned primitives like Ascon, SERPENT, KNOT, and Simeck.

4.1 Decomposing the Distinguisher

Before proposing our CP/MILP-based methods for exploring DL distinguishers, we would like to discuss how to decompose the DL distinguishers. In Section 3, we demonstrated that the correlation r of the middle part of a DL distinguisher is essentially the sum of products of fractional values proportional to the generalized `DLCT` tables of common active S-boxes. As a result, an increase in common active S-boxes (more generally, non-linear operations) in the middle part results in more tables in the formula, thereby complicating the correlation evaluation. The computational complexity of evaluating the correlation r of the middle part is approximately r^{-2}, which is exponential in the number of common active S-boxes or generally common active non-linear operations. Therefore, regardless of the approach to computing the correlation of the middle part, the number of common active S-boxes exponentially increases the computational complexity of evaluating the correlation of the middle part, whether we use experimental or theoretical approaches for computation.

The number of common active S-boxes (bits) in the middle part depends on the diffusion effect of the linear layer (and also the diffusion effect of the S-box in the bit-level) and the density of the differential and linear trails within the boundaries of E_m. For example, the more differentially (resp. linearly) active bits at the input of E_m, the sooner the differences (resp. linear masks) propagate to the entire state. On the other hand, the accuracy of correlation estimation based on prq^2 highly depends on the length of the middle part of r_m rounds. If we set r_m to be too short, then we are considering the dependency in only a few rounds, ignoring many potential cell-wise switches, and also some harmful dependencies between the two parts that can spoil the distinguisher.

Therefore, when decomposing the DL distinguishers, we should make a trade-off between the quality of the distinguisher and the computational complexity of evaluating the correlation of the middle part. The middle part should be long enough for the dependency between the two parts almost to disappear. On the other hand, it should be short enough to evaluate the correlation of the middle part efficiently. As a result, a reasonable starting point for the length of the middle part is the length of full diffusion of the corresponding primitive in terms of differential and linear trails. Then, we can prepend/append some rounds before/after E_m to obtain E_u and E_ℓ.

After deriving the distinguisher, we can also conduct experiments on a reduced number of rounds to enhance the confidence in the accuracy of correlation estimation. To this end, we can extend E_m by a few rounds before and after and check if the prq^2 formula gives a reasonable estimate for the correlation of the extended E_m. If we have chosen the length of the middle part properly, the correlation of the extended E_m derived by the prq^2 formula should be close to the actual value derived by experiments; otherwise, we should increase the length of the middle part and repeat the process.

4.2 Cell-Wise Modeling for Distinguishers

We first explain our cell-wise model because it is more straightforward and intuitive. In this model, we consider the S-boxes as black-boxes and set up the truncated differential and linear trails as constraints. We use the term *cell-wise* because we focus on how the differential and linear trails propagate at the cell level. A cell is active if its difference or linear mask is nonzero; otherwise, it is inactive. Our cell-wise model has three main parts:

- A truncated CP/MILP model for finding truncated differential-linear trails.
- A bit-wise CP/MILP model to instantiate the discovered truncated trail and compute p and q^2.
- A module to compute r using the `DLCT` framework or experimental approach. Lastly, we estimate the total correlation using the prq^2 formula.

We implement these three modules in a unified tool that receives three integer numbers r_u, r_m, r_ℓ, as an input and outputs the distinguisher together with an estimated correlation. The round numbers r_u, r_m, and r_ℓ determine the decomposition of the distinguisher, i.e., the length of E_u, E_m, and E_ℓ, respectively.

The most critical step is finding good truncated differential-linear trails (first part), as it determines the position of differentially/linearly active bits, and the second and third modules should align with the activation pattern of the first module's output. In addition, the CP/MILP models to find ordinary differential/linear trails have already been studied in the literature [1,35,53]. So, in what follows, we mainly focus on the first module.

First, we give an overall view of our approach. As illustrated in Figure 5, we decompose the distinguisher into three parts: E_u, E_m, and E_ℓ, with the lengths of r_u, r_m, and r_ℓ, respectively. Let p be the probability of the differential transition over E_u, r be the correlation of the middle part, and q^2 be the squared correlation of the linear approximation for E_ℓ. Based on Section 4.1, as long as the length of E_m is long enough and Δ_m, λ_m are well-chosen, prq^2 gives a reasonable estimate for the correlation of the distinguisher. Hence, we aim to create a CP/MILP model to find a truncated differential-linear trail that maximizes the value of prq^2. We know that p and q depend on the number of differentially/linearly active S-boxes in E_u and E_ℓ, respectively. The middle term r is the sum of products of fractional values, with each fractional value corresponding to a specific generalized `DLCT` table, associated with the involved S-boxes through E_m. Due to the cell-wise switches, we expect a higher value for r if the number of common active S-boxes in E_m is minimized. Consequently, we create a unified CP model to find a truncated differential-linear trail such that the total number of differentially (resp. linearly) active S-boxes in E_u (resp. E_ℓ) as well as common active S-boxes in E_m is minimized. To count the number of common active S-boxes in E_m (considering cell-wise switches), we employ the idea of cell-wise deterministic differential and linear propagation. More precisely, while we propagate truncated differential/linear trails through E_u/E_ℓ as usual, we switch to deterministic propagation of differential/linear trails in the level of

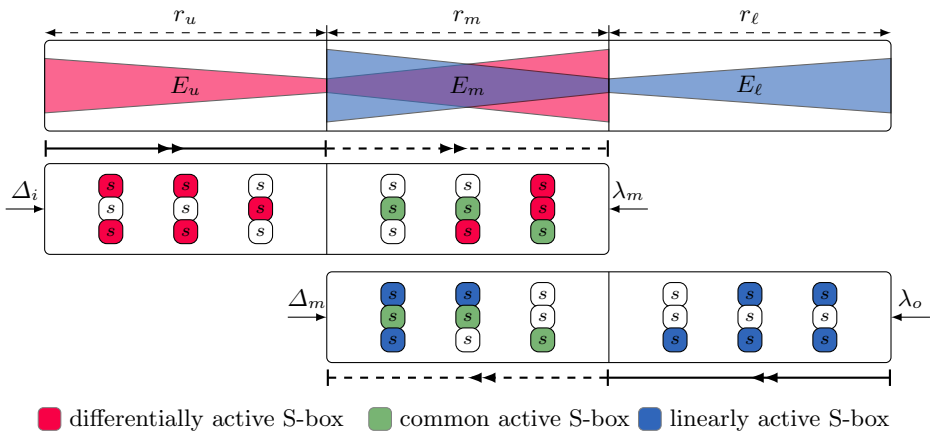


Fig. 5: Overall view of our CP/MILP model for finding DL distinguishers.

cells through E_m and identify which cells are only active in at most one of the trails.

Now, we explain the details of our cell-wise model. As discussed in Section 4.1, we typically set the length of the middle part to the size of the full diffusion of the primitive. Next, as Figure 5 visualizes, we model the propagation of the truncated differential (resp. linear) trails through the $E_m \circ E_u$ (resp. $E_\ell \circ E_m$) forward (resp. backward). Let $n = m \cdot c$ be the block size of E, where c represents the cell size and m is the number of cells in each state. We define the binary variables $\mathtt{XU}_t[i]$ (resp. $\mathtt{XL}_t[i]$) to represent the activeness pattern in the ith cell of the internal state in round t of E_u (resp. E_ℓ) for differential (resp. linear) propagation. We also define the binary variables $\mathtt{XMU}_t[i]$ (resp. $\mathtt{XML}_t[i]$) to represent the activeness pattern of the internal state in round t of E_m in the differential (resp. linear) propagation.

Let $\mathit{CSP}_u\,(\mathtt{XU}_0, \cdots, \mathtt{XU}_{r_u})$ represent the CSP model for the differential propagation over E_u, and $\mathit{CSP}_\ell\,(\mathtt{XL}_0, \cdots, \mathtt{XL}_{r_\ell})$ denote the CSP model for the linear propagation over E_ℓ. Similarly, let $\mathit{CSP}_{mu}\,(\mathtt{XMU}_0, \cdots, \mathtt{XMU}_{r_m})$ be the CSP model for the differential propagation over E_m, and $\mathit{CSP}_{m\ell}\,(\mathtt{XML}_0, \cdots, \mathtt{XML}_{r_m})$ represent the CSP model for the linear propagation over E_m^{-1}. While CSP_u, and CSP_ℓ encode the propagation of truncated trails over E_u and E_ℓ as usual, CSP_{mu} and $\mathit{CSP}_{m\ell}$ encode the propagation over E_m in a deterministic manner. Deterministic means that the model does not allow cancellation through the diffusion layers, though it can happen with a certain probability. The main reason for this choice is that almost all common active S-boxes in the middle affect the correlation for the middle part. If we let the cancellation happen in the middle part, we are essentially ignoring the effect of some potentially active S-boxes in the middle part this would lead to many false positive solutions, i.e., solutions in which the number of common active S-boxes in the middle is essentially much higher in practice, but we have underestimated it. So, to avoid underestimating

the number of common active S-boxes in the middle part, we use deterministic propagation over E_m. This way, we can ensure that the number of actually effective S-boxes in the middle part is as high as the number of common active S-boxes in the solution.

Next, we merge the CSP models CSP_u, CSP_ℓ, CSP_{mu}, and $\mathit{CSP}_{m\ell}$ to form a unified CSP model CSP_d for the truncated differential-linear distinguisher. Additionally, we include constraints $\sum_{i=0}^{m} \mathtt{XU}_0[i] \neq 0$ and $\sum_{i=0}^{m} \mathtt{XL}_{r_\ell}[i] \neq 0$ to exclude trivial solutions. In our CP model, $\mathtt{XU}_{r_u}$ and $\mathtt{XMU}_0$ correspond to the same internal state at the junction of E_u and E_m, and similarly for $\mathtt{XL}_0$ and $\mathtt{XML}_{r_m}$. Therefore, we enforce constraints $\mathtt{XU}_{r_u}[i] = \mathtt{XMU}_0[i]$ and $\mathtt{XL}_0[i] = \mathtt{XML}_{r_m}[i]$ for all i. Finally, using constant integer weights w_u, w_m, and w_ℓ, we incorporate the following objective function to build a unified optimization problem COP_d:

$$\sum_{t=0}^{r_u} w_u \cdot \mathtt{XU}_t[i] + \sum_{t=0}^{r_d} w_m \cdot \mathtt{bool2int}\left(\mathtt{XMU}_t[i] + \mathtt{XML}_t[i] = 2\right) + \sum_{t=0}^{r_\ell} w_\ell \cdot \mathtt{XL}_t[i]. \quad (7)$$

The integer weights w_u, w_m, and w_ℓ should be proportional to the differential uniformity ($\mathcal{DU}$), differential-linear uniformity ($\mathcal{DLU}$), and squared linearity ($\mathcal{L}^2$) of the S-box, respectively. For example, in the case of WARP, we have $\mathcal{DU} = 2^2, \mathcal{L} = 2^3$ and $\mathcal{DLU} = 2^4$, or equivalently, $\mathbb{P}_{\mathtt{DDT}} \leq 2^{-2}$, $\mathbb{C}^2_{\mathtt{LAT}} \leq 2^{-2}$, and $\mathbb{C}_{\mathtt{DLCT}} \leq 1$. In addition, in our CP/MILP models, we work with the absolute logarithm of probability transitions or squared correlation, i.e., $|\log_2(\mathbb{P}_{\mathtt{DDT}})|, |\log_2(\mathbb{C}^2_{\mathtt{LAT}})|$, and $|\log_2(\mathbb{C}_{\mathtt{DLCT}})|$. Therefore, the cost of each active S-box in E_u and E_ℓ is approximately twice as much as an active S-box in E_m. As a result, $(2, 1, 2)$ is a reasonable choice for (w_u, w_m, w_ℓ) for WARP. For ciphers that employ different S-boxes with different $\mathcal{DU}, \mathcal{L}$, and $\mathcal{DLU}$ in the design, e.g., CLEFIA, we can use different appropriate weights for each S-box in the CP model.

After finding a solution for COP_d, we find concrete differential (resp. linear) trails for E_u (resp. E_ℓ) that satisfy the activeness pattern of the solution for COP_d. To this end, we generate bit-wise CP/MILP models for differential (resp. linear) trails over E_u (resp. E_ℓ) and set all inactive bits to zero. After solving the bit-wise models and deriving concrete differential (resp. linear) trails for E_u (resp. E_ℓ), we only fix the differences (resp. linear masks) at the input and output of E_u, and E_ℓ, i.e., $\Delta_i, \Delta_m, \lambda_m, \lambda_o$ in Figure 5. Then we compute p and q^2, where we consider the clustering effect for computing p. Lastly, we compute r using the generalized DLCT framework or experimental approach and use the prq^2 formula to estimate the total correlation of the distinguisher. We applied our cell-wise model to AES, SKINNY, CLEFIA, WARP, LBlock, LBlock-s and TWINE and found new DL distinguishers for these ciphers. In what follows, we briefly discuss the discoveries based on our cell-wise model.

Application to AES We applied our method to search for DL distinguishers of AES in the single-key setting. Refer to Section C for a brief specification of AES. Given that AES achieves full diffusion within cells after 2 rounds, we set the length of the middle part (E_m) in our searches to 3. This choice sufficiently captures the effect of the middle part while also modeling the dependency between

differential and linear trails. Table 5 summarizes the results of our searches for AES with 2 to 5 rounds. When running our tool for 3 to 5 rounds of AES, due to the symmetry of AES design, it discovered nearly identical activeness patterns for the middle part of our distinguishers, with 3 common active S-boxes. Additionally, the correlation of the 3-round middle part of our DL distinguishers for AES can be accurately evaluated using Equation 5. Evaluating Equation 5 for all possible input/output differences and masks reveals that the absolute correlation of the middle part ranges from $2^{-8.43}$ to $2^{-7.66}$ (as shown in Figure 8). This observation confirms that for strongly aligned ciphers such as AES, the choice of the optimal cell-wise pattern for the distinguisher has a greater impact on its correlation than selecting the actual values for the active input/output cells of the middle part. Referring to the previous literature on DL distinguishers for AES, there are only two results thus far, exclusively in the related-key setting and limited to (up to 5 rounds) AES-192 [52, 60]. Therefore, we are the first to find DL distinguishers for up to 5 rounds of (all versions of) AES in the single-key setting.

Application to CLEFIA As one of the most important Feistel ciphers, we applied our tool to the ISO standard (ISO/IEC 29192-2) block cipher CLEFIA. Section L provides a brief specification of CLEFIA. The full diffusion of CLEFIA at the byte level is 5. So, we set the length of the middle part to 5 in our searches. Table 34 summarizes the results for CLEFIA. The most interesting property of CLEFIA is the Diffusion Switching Mechanism (DSM) [50, 51]. Thanks to this mechanism, the number of differentially/linearly active S-boxes of CLEFIA is 40% higher than an ordinary Generalized Feistel Structure (GFS) without DSM. The diffusion switching mechanism of CLEFIA comes into effect for more than 3 (resp. 7) rounds in the linear (resp. differential) analysis. So, one may expect a much higher resistance against DL distinguishers for CLEFIA compared to boomerang distinguishers. Due to this feature, and also considering that there is no previous result on DL distinguishers for CLEFIA, comparing our DL distinguishers to boomerang distinguishers of CLEFIA is of great interest. To date, the best boomerang distinguishers proposed for CLEFIA cover up to 9 rounds [33]. As can be seen in Table 34, our DL distinguishers also reach up to 9 rounds. However, for up to 8 rounds of CLEFIA, our DL distinguishers significantly surpass the best boomerang distinguishers. For example, whereas the data complexity of the best boomerang distinguisher for 7 (resp. 8) rounds of CLEFIA is $2^{32.67}$ (resp. $2^{76.03}$), the data complexity of our 7-round (resp. 8-round) DL distinguisher is $2^{23.50}$ (resp. $2^{66.86}$).

Application to SKINNY As an application from lightweight tweakable block ciphers, we targeted SKINNY in both single-tweakey and related-tweakey settings. Section F briefly describes the SKINNY family of block ciphers. The full diffusion of SKINNY on the word-level is 6 rounds. So we set the length of the middle part to 6 or more in our searches. Table 8 summarizes our results for SKINNY in the single-key setting. Notably, we achieved an 11-round DL distin-

guisher, matching the effectiveness of the best-known single-key distinguishers of SKINNY in terms of the number of rounds. We also achieved interesting results for the related-tweakey setting (refer to Table 9,Table 10). The most potent combined distinguishers on SKINNY in the related-tweakey setting are the boomerang distinguisher [30]. However, its efficacy comes at the cost of requiring a minimum of four related tweakeys. Among the related-tweakey distinguishers of SKINNY with only two related tweakeys, the impossible-differential distinguishers are the longest ones which cover up to 14 rounds (resp. 16 rounds) of SKINNY-n-$2n$ (resp. SKINNY-n-$3n$) [34]. However, our related-tweakey DL distinguisher covers one round more, reaching 15 rounds of SKINNY-n-$2n$, and 17 rounds of SKINNY-n-$3n$.

Appication to WARP Here we propose DL distinguishers for WARP for the first time. Section K briefly describes the lightweight block cipher WARP. According to the designers of WARP [2], the nibble-wise full diffusion of WARP is achieved after 10 rounds. So, we set the length of the middle part to 10 or 11 in our searches. Although the designers of WARP claimed nibble-wise full diffusion after 10 rounds, we discovered a deterministic DL distinguisher for 11 rounds of WARP that is noteworthy. This observation arises not only from the diffusion layer but also from the differential-linear behavior of WARP's S-box. Table 30 illustrates the DLCT of WARP's S-box. As shown, the differential-linear uniformity of WARP's S-box is 16 ($\mathcal{DLU}_{\mathsf{WARP}} = 16$), indicating that the correlation of the common active S-box in the middle can reach 1 for certain input/output differences/linear masks. For instance, in our 11-round distinguisher for WARP, we can express the correlation as $\mathbb{C}^2_{\mathtt{DLCT}}(\Delta_i, \lambda_o)$, where Δ_i and λ_o denote the difference and linear mask of the active cell at the input and output of the middle part, respectively. According to Table 30, we have $\mathbb{C}_{\mathtt{DLCT}}(\mathsf{2},\mathsf{2}) = 1$. Table 32 and Table 33 summarize our results for WARP. As another interesting example of our new DLCT tables, the correlation of the 11-round middle part of our DL distinguishers for 16 to 22 rounds of WARP can be compactly formulated by only one DDLCT. In particular, let Δ_i and λ_o denote the difference and linear mask of the active cell at the input and output of the middle part, respectively. Then, the correlation of the middle part of our 16- to 22-round distinguishers is given by $\mathbb{C}^3_{\mathtt{DDLCT}}(\Delta_i, \lambda_o)$. Table 31 shows the DDLCT of WARP's S-box. As seen in Table 31, if $(\Delta_i, \lambda_o) \in \{(\mathsf{a},\mathsf{2}), (\mathsf{2},\mathsf{b}), (\mathsf{2},\mathsf{e})\}$, the absolute correlation of the middle part is maximum.

Application to TWINE Section M provides a brief specification of TWINE. TWINE achieves full nibble-wise diffusion after 8 rounds. Thus, we set r_m to more than 9 rounds in our searches to ensure that we capture the effect of the middle part and consider the dependencies between differential and linear trails. Table 37 shows the DLCT of TWINE's S-box. In contrast to WARP's S-box, which exhibits maximal $\mathcal{DLU}$, TWINE's S-box has $\mathcal{DLU}_{\mathsf{TWINE}} = 8$, not maximal. Thus, TWINE's S-box is more resistant to DL distinguishers. Table 40 summarizes our results for TWINE. As an application of our new DLCT tables, we can formu-

late the correlation of our 8-round distinguisher for TWINE using one DDLCT. Table 38 illustrates the DDLCT of TWINE's S-box, indicating that the maximum absolute correlation is achieved by setting the active input/output differences/-linear masks to $(8, 2)$. As another interesting example, we can formulate the correlation of our 9-round distinguisher for TWINE by only one 3-DLCT. More precisely, if Δ_i and λ_o denote the difference and linear mask of the active cell at the input and output of the middle part, respectively, then the correlation of our 9-round distinguisher is given by $\mathbb{C}^2_{\text{3-DLCT}}(\Delta_i, \lambda_o)$. Table 39 describes the 3-DLCT of TWINE's S-box. According to Table 39, choosing the active input/output differences/linear masks from $\{(4, 5), (5, a)\}$, results in the maximum absolute correlation. As seen in Table 40, we propose DL distinguisher for up to 17 rounds of TWINE for the first time. Interestingly, our 17-round DL distinguisher is one round longer than all previous distinguishers for TWINE, including its longest boomerang distinguisher in [33].

Application to LBlock and LBlock-s Section J provides a brief specification of LBlock and LBlock-s. Although LBlock employs 8 different S-boxes in its design, for all of them, $\mathcal{DU} = \mathcal{L}^2 = \mathcal{DLU}/4 = 4$. Therefore, in our cell-wise CP model, we treat all S-boxes equally (similar to the cell-wise CP model for LBlock-s). Like TWINE, LBlock exhibits full nibble-wise diffusion after 8 rounds for both encryption and decryption. Consequently, we set the middle part length in our searches to at least 8 rounds. Table 24 and Table 25 summarize our results for LBlock-s and LBlock, respectively. As an application of our new DLCT tables, the middle part of our 8- to 12-round distinguishers for LBlock-s can be compactly formulated by only one DDLCT. Table 23 illustrates the DDLCT of LBlock-s's S-box. According to Table 23, the maximum absolute correlation is achieved by setting the active input/output differences/linear masks to $(a, 1)$. We managed to propose DL distinguishers for up to 17 rounds of LBlock-s and LBlock for the first time. Our 17-round DL distinguisher for LBlock and LBlock-s exceeds the length of the longest boomerang distinguishers for both ciphers reported in [33], highlighting the superiority of DL distinguishers over boomerang ones.

The advantage of our cell-wise model lies in its high efficiency and sufficient accuracy, particularly for strongly aligned ciphers. For instance, results for AES and CLEFIA are typically obtained within minutes, while those for TWINE, SKINNY-64, and WARP are often solvable within seconds on a standard laptop. However, when dealing with weakly aligned designs such as Ascon, KNOT, SERPENT, or Simeck, a cell-wise CP model lacks the precision required to track the deterministic differential/linear transitions in the middle part. To tackle this issue, we introduce a bit-wise model in Section 4.3.

4.3 Bit-Wise Modeling for Distinguishers

To capture the bit-wise switches for one S-box layer, DLCT is reasonably sufficient. Indeed, one can even model all valid bit-wise differential-linear transitions for a single S-box layer by using some CP constraints encoding the DLCT [40].

Nevertheless, this approach is not extendable to more than one round. In this section, we first show that the idea of deterministic bit-wise differential/linear transitions can capture many bit-wise switches across multiple rounds. Then, we propose a CP/MILP-based method to model the deterministic bit-wise differential/linear trails. Lastly, we use our deterministic bit-wise model to create a CP model to explore DL distinguishers of weakly aligned primitives.

Table 4: DLCT of KNOT's S-box.

$\Delta \setminus \lambda$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	0	0	0	**-16**	0	0	0	0	0	0	0	0	0	0	0
2	16	-8	-8	0	0	0	8	-8	0	-8	0	8	0	0	0	0
3	16	0	-8	-8	0	-8	8	0	0	0	0	0	0	-8	0	8
4	16	0	-8	0	0	0	-8	0	**-16**	0	8	0	0	0	8	0
5	16	0	-8	0	0	0	-8	0	0	0	8	0	-16	0	8	0
6	16	-8	8	-8	0	0	-8	0	0	-8	0	0	0	0	0	8
7	16	0	8	0	0	-8	-8	-8	0	0	0	8	0	-8	0	0
8	16	0	0	0	**-16**	0	0	0	**-16**	0	0	0	**16**	0	0	0
9	16	-8	0	-8	**16**	-8	0	-8	0	8	0	-8	0	8	0	-8
a	16	0	0	8	0	8	0	0	0	0	-8	0	0	-8	-8	-8
b	16	8	0	0	0	0	0	8	0	-8	-8	-8	0	0	-8	0
c	16	0	0	-8	0	0	0	-8	**16**	0	0	-8	0	0	0	-8
d	16	-8	0	0	0	-8	0	0	0	8	0	0	-16	8	0	0
e	16	0	0	0	0	8	0	8	0	0	-8	-8	0	-8	-8	0
f	16	8	0	8	0	0	0	0	0	-8	-8	0	0	0	-8	-8

To explain why the deterministic bit-wise differential/linear transitions can capture many bit-wise switches, we give a basic example in the level of one S-box layer. However, this approach can be extended beyond one S-box layer. Let S be the S-box of KNOT. Table 4 illustrates the DLCT of KNOT's S-box. Also assume that we represent the difference (resp. linear mask) in each bit of differential (resp. linear) trail by $\{0, 1, ?\}$, where ? denotes the unknown bits in terms of difference (resp. linear mask) value. Table 14 shows the DDT of KNOT's S-box. Let Δ_i, and Δ_o be the input and output difference of the S-box, respectively. By checking all input differences in $\{0, 1, ?\}^4$, we can identify the following nontrivial deterministic differential transitions $\Delta_i \xrightarrow{S} \Delta_o$:

$$\begin{aligned}
&\Delta_i = (0,0,0,1) \xrightarrow{S} \Delta_o = (?,1,?,?) \quad \Delta_i = (0,1,0,0) \xrightarrow{S} \Delta_o = (1,?,?,?)\\
&\Delta_i = (1,0,0,0) \xrightarrow{S} \Delta_o = (1,1,?,?) \quad \Delta_i = (1,0,0,1) \xrightarrow{S} \Delta_o = (?,0,?,?)\\
&\Delta_i = (1,1,0,0) \xrightarrow{S} \Delta_o = (0,?,?,?)
\end{aligned} \tag{8}$$

Referring to Equation 8, when $\Delta_i = (1,0,0,0)$, then $\Delta_o = (1,1,?,?)$. Thus, if $\lambda_o \in (0,1,0,0), (1,0,0,0), (1,1,0,0)$, $\lambda_o \cdot \Delta_o$ remains constant for all input pairs

with difference Δ_i. This explains the bit-wise switches at indices $(8,4),(8,8),(8,c)$ in DLCT. Similarly, when $\Delta_i = (1,0,0,1)$, $\Delta_o = (?,0,?,?)$, indicating that for $\lambda_o = (0,1,0,0)$, $\lambda_o \cdot \Delta_o$ remains constant for all input pairs with difference Δ_i. This accounts for the bit-wise switch at index $(9,4)$ in DLCT. Similarly, other bit-wise switches, such as $(1,4),(4,8)$, and $(c,8)$, can be explained using only the bit-wise deterministic differential transitions.

We can also explain the same bit-wise switches by using deterministic backward linear transitions. Table 15 shows the LAT of KNOT's S-box. Let λ_i and λ_o be the input and output linear masks of the S-box, respectively. One can see that there are 3 nontrivial deterministic linear transitions for $\lambda_i \xleftarrow{S} \lambda_o$:

$$\begin{aligned} &\lambda_i = (1,?,?,1) \xleftarrow{S} \lambda_o = (0,1,0,0) \quad \lambda_i = (1,1,?,?) \xleftarrow{S} \lambda_o = (1,0,0,0) \\ &\lambda_i = (0,?,?,?) \xleftarrow{S} \lambda_o = (1,1,0,0) \end{aligned} \tag{9}$$

Based on Equation 9 if $\lambda_o = (1,0,0,0)$ then $\lambda_i = (1,1,?,?)$. Therefore, $\lambda_i \cdot \Delta_i$ is fixed for all $\Delta_i \in \{(0,1,0,0),(1,0,0,0),(1,1,0,0)\}$, explaining the bit-wise switches $\{(4,8),(8,8),(c,8)\}$. As another example, if $\lambda_o = (0,1,0,0)$, then $\lambda_i = (1,?,?,1)$. Thus, if $\Delta_i \in \{(1,0,0,1),(1,0,0,0),(0,0,0,1)\}$ then $\lambda_i \cdot \Delta_i$ is constant. It explains the bit-wise switches $\{(9,4),(8,4),(1,4)\}$ in DLCT. The bit-wise switch $(8,c)$ can also be identified by only using the bit-wise deterministic linear transitions.

Now, we explain how to model deterministic bit-wise differential/linear transitions. For each bit of the internal state in our bit-wise model, we define an integer variable with domain $\{-1,0,1\}$ to represent if the difference (linear mask) is unknown, zero, or one, respectively. Next, we define some constraints to model the truncated differential/linear trails at the bit level. In what follows, we define the constraints for the common cryptographic operations, e.g., XOR, Copy, and S-boxes. We describe our method for truncated differential trails, and a similar method applies to truncated linear trails.

Proposition 3 (XOR). *For $f : \mathbb{F}_2^n \to \mathbb{F}_2$, $f(x_0,x_1,\ldots,x_{n-1}) = y$, where $y = x_0 \oplus x_1 \oplus \cdots \oplus x_{n-1}$, the valid deterministic differential transitions satisfy:*

$$\mathit{XOR}_b(Y, X[0],\ldots,X[n-1]) := \begin{cases} \mathit{if}\ \bigvee_{i=0}^{n-1} (X[i] = -1)\ \ \mathit{then}\ Y = -1 \\ \mathit{else}\ Y = X[0] + X[1] + \cdots + X[n-1] \mod 2\ \ \mathit{endif}, \end{cases}$$

where $X[i]$, and Y are integer variables in $\{-1,0,1\}$ for $0 \le i \le n-1$.

Proposition 4 (Copy). *For $f : \mathbb{F}_2 \to \mathbb{F}_2^n$, $f(x) = (y_0,y_1,\ldots,y_{n-1})$ where $y_0 = y_1 = \cdots = x$, valid transitions for deterministic differential trails satisfy*

$$\mathit{Branch}_b(X, Y[0],\ldots,Y[n-1]) := \bigwedge_{i=0}^{n-1} (Y[i] = X),$$

where X, and $Y[i]$ are integer variables with domain $\{-1,0,1\}$ for all $0 \le i \le n-1$.

Regarding S-boxes or generally a non-linear vectorial Boolean function, we refer to its DDT (resp. LAT) to identify deterministic bit-wise differential (resp. linear) transitions. Then, we define some constraints to model these deterministic transitions. To explain our bit-wise model for S-boxes, we give a basic example. Take the 4-bit S-box of KNOT as an example. Let $\mathtt{X}[i]$ and $\mathtt{Y}[i]$ be integer variables with domain $\{-1, 0, 1\}$ for $0 \leq i \leq 3$, to denote the input and output difference bits, respectively, where $\mathtt{X}[0], \mathtt{Y}[0]$ correspond the MSB. The bit-wise deterministic differential transitions of KNOT's S-box are represented in Equation 9. In CP modeling, we can model these transitions using the following constraints:

$$\begin{cases} \textit{if } (\mathtt{X}[0] = 0 \wedge \mathtt{X}[1] = 0 \wedge \mathtt{X}[2] = 0 \wedge \mathtt{X}[3] = 0) \textit{ then } (\mathtt{Y}[0] = 0 \wedge \mathtt{Y}[1] = 0 \wedge \mathtt{Y}[2] = 0 \wedge \mathtt{Y}[3] = 0) \\ \textit{elseif } (\mathtt{X}[0] = 0 \wedge \mathtt{X}[1] = 0 \wedge \mathtt{X}[2] = 0 \wedge \mathtt{X}[3] = 1) \textit{ then } (\mathtt{Y}[0] = -1 \wedge \mathtt{Y}[1] = 1 \wedge \mathtt{Y}[2] = -1 \wedge \mathtt{Y}[3] = -1) \\ \textit{elseif } (\mathtt{X}[0] = 0 \wedge \mathtt{X}[1] = 1 \wedge \mathtt{X}[2] = 0 \wedge \mathtt{X}[3] = 0) \textit{ then } (\mathtt{Y}[0] = 1 \wedge \mathtt{Y}[1] = -1 \wedge \mathtt{Y}[2] = -1 \wedge \mathtt{Y}[3] = -1) \\ \textit{elseif } (\mathtt{X}[0] = 1 \wedge \mathtt{X}[1] = 0 \wedge \mathtt{X}[2] = 0 \wedge \mathtt{X}[3] = 0) \textit{ then } (\mathtt{Y}[0] = 1 \wedge \mathtt{Y}[1] = 1 \wedge \mathtt{Y}[2] = -1 \wedge \mathtt{Y}[3] = -1) \\ \textit{elseif } (\mathtt{X}[0] = 1 \wedge \mathtt{X}[1] = 0 \wedge \mathtt{X}[2] = 0 \wedge \mathtt{X}[3] = 1) \textit{ then } (\mathtt{Y}[0] = -1 \wedge \mathtt{Y}[1] = 0 \wedge \mathtt{Y}[2] = -1 \wedge \mathtt{Y}[3] = -1) \\ \textit{elseif } (\mathtt{X}[0] = 1 \wedge \mathtt{X}[1] = 1 \wedge \mathtt{X}[2] = 0 \wedge \mathtt{X}[3] = 0) \textit{ then } (\mathtt{Y}[0] = 0 \wedge \mathtt{Y}[1] = -1 \wedge \mathtt{Y}[2] = -1 \wedge \mathtt{Y}[3] = -1) \\ \textit{else } (\mathtt{Y}[0] = -1 \wedge \mathtt{Y}[1] = -1 \wedge \mathtt{Y}[2] = -1 \wedge \mathtt{Y}[3] = -1) \textit{ endif}; \end{cases}$$

We can model the backward deterministic bit-wise linear transitions in Equation 9 in the same way.

Now, we explain our bit-wise model for finding DL distinguishers. As before, we split the primitive E into three parts E_u, E_m, and E_ℓ of lengths r_u, r_m, and r_ℓ, respectively. Let $\mathtt{XU}_t[i]$ (resp. $\mathtt{XL}_t[i]$) represents the value of the difference (resp. linear mask) in the ith bit of the internal state in round t of E_u (resp. E_ℓ). Besides, let $\mathtt{XMU}_t[i]$ (resp. $\mathtt{XML}_t[i]$) represents the ith bit of the internal state in round t of E_m for differential (resp. linear) propagation. We create CP models $\mathit{CSP}_u\,(\mathtt{XU}_t, \cdots, \mathtt{XU}_{r_u})$ and $\mathit{CSP}_\ell(\mathtt{XL}_0, \cdots, \mathtt{XL}_{r_\ell})$ to model the bit-wise differential and linear trails through E_u and E_ℓ, respectively. For this purpose, we employ the methods outlined in prior studies such as [1, 35, 53] for the propagation of differential and linear trails at the bit level. However, for E_m, we switch to deterministic propagation at the bit level. For this, we create $\mathit{CSP}_{mu}\,(\mathtt{XMU}_0, \cdots, \mathtt{XMU}_{r_m})$ (resp. $\mathit{CSP}_{m\ell}\,(\mathtt{XML}_0, \cdots, \mathtt{XML}_{r_m})$) to model the deterministic propagation of differential (resp. linear) trails through E_m (resp. E_m^{-1}).

In CSP_u, we encode the DDT by CP constraints to model the probability of differential transition over E_u. In CSP_ℓ, we encode $\mathtt{LAT}^2$ by CP constraints to model the squared correlation of linear approximation over E_ℓ. We use the open-source S-box Analyzer tool [33] (refer to Section N) to effectively encode the differential and linear behavior of S-boxes and other operations in our CP models. Any feasible solution for CSP_u (resp. CSP_ℓ) is a differential (resp. linear) trail for E_u (resp. E_ℓ). Assume that the probability of differential transition over E_u is p and the squared correlation of linear transition over E_ℓ is q^2. In our CP model, we define the variables PU and CL to encode $-\log_2(p)$ and $-\log_2(q^2)$, respectively. Depending on the DDT and LAT, the variables PU and CL can be integer or real-valued variables. Lastly, we combine CSP_u, CSP_ℓ, CSP_{mu}, and $\mathit{CSP}_{m\ell}$ to create a unified CP model for the DL distinguisher. Like our cell-wise CP model, we add some constraints to link the internal states at the join points of E_u and E_m and E_m and E_ℓ. To identify the number of bit positions that are active in both

differential and linear propagations in the middle part and take unknown values in at least one of the differential and linear propagations, we define the following integer variable:

$$\texttt{CM} = \sum_{t=0}^{r_m} \mathit{bool2int}\left((\texttt{XMU}_t[i] = -1 \vee \texttt{XML}_t[i] = -1) \wedge (\texttt{XMU}_t[i] \neq 0 \wedge \texttt{XML}_t[i] \neq 0)\right) \tag{10}$$

Next, assuming that w_u, w_m, and w_ℓ are some integer constants, we set the objective function to: $\min(w_u \cdot \texttt{PU} + w_m \cdot \texttt{CM} + w_\ell \cdot \texttt{CL})$. If the number of common active bits in the middle is high, then computing the correlation of the middle part becomes more difficult. Therefore, we typically use these integer weights to make a trade-off between the weight of differential and linear transitions over E_u and E_ℓ, and the number of common active bits in the middle part. After finding a solution for the unified CP model, we use the `DLCT` framework or experimental approach to compute the correlation of the middle part. Lastly, we put $p = 2^{-\texttt{PU}}$, $q^2 = 2^{-\texttt{CL}}$, and r together in the prq^2 formula to estimate the total correlation of the distinguisher.

It is worth noting that, while our cell-wise model is inspired by techniques for finding boomerang distinguishers [30], there is no equivalent method for our bit-wise model in the context of boomerang distinguishers. For example, very recently, Bonnetain and Lallemand [15] demonstrated at ToSC 2023 that previous tools for finding boomerang distinguishers are not applicable to bit-sliced designs like Simeck, and they had to propose a different approach to find boomerang distinguishers. However, our bit-wise model can be applied to both strongly aligned primitives like WARP and weakly aligned primitives like Simeck.

Application to Ascon Permutation To illustrate the usefulness of our bit-wise model, we first applied it to Ascon [23,24], the winner of the NIST lightweight cryptography standardization (LWC) process. Section D.2 briefly describes the Ascon permutation. Ascon achieves full diffusion at the bit-level after 3.5 rounds. Therefore, we set the length of the middle part to 2, 3, or 4 in our searches. Table 6 summarizes our results for Ascon. We discovered the first 4-round deterministic DL distinguisher for Ascon. To find the 4-round DL distinguisher, we set r_m to 4. In this case, our tool returns a solution with no overlap between the active bits in the differential and linear propagations at the output of the S-box layer. As a result, due the bit-wise switches, the correlation of our 4-round DL distinguisher is 1, which we also verified experimentally. Nevertheless, as Figure 6 shows, there are many common active S-boxes whose input difference and output linear masks are non-zero, and all switches are bit-wise switches that are not detectable by the cell-wise models. This example showcases the effectiveness of our bit-wise model. Moreover, we uncovered a 5-round DL distinguisher for Ascon that raises the correlation from 2^{-9} in the best previous 5-round distinguishers [22] to $2^{-4.33}$. Due to the rotational invariant property of Ascon with respect to differences and linear trails, each of our DL distinguishers represent 64 different DL distinguishers with the same correlation.

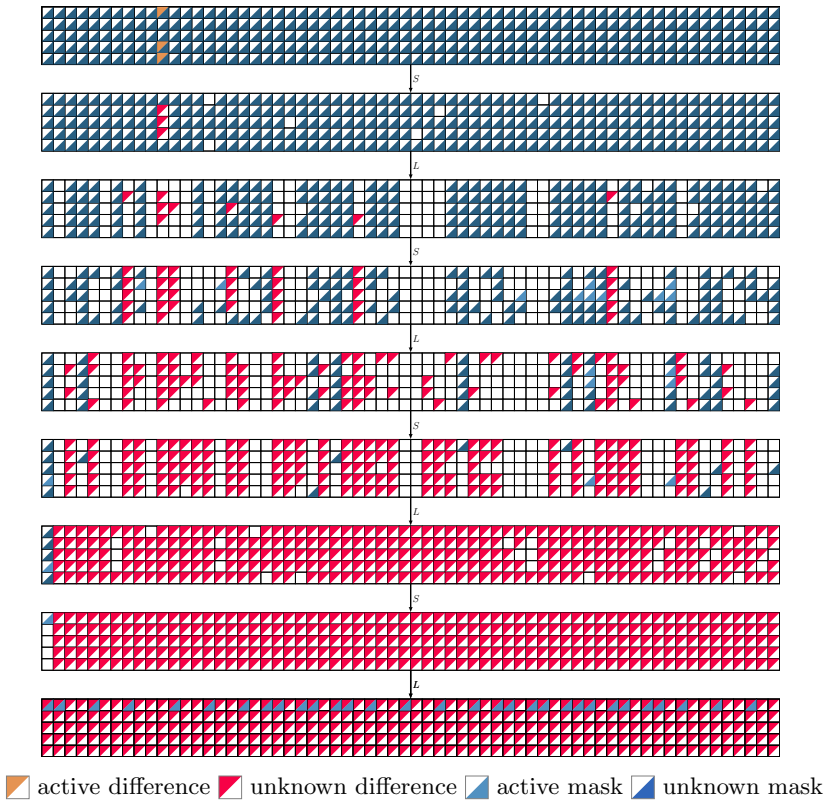


Fig. 6: DL distinguisher I for 4 rounds of Ascon.

Application to SERPENT As another important bit-sliced cipher, we applied our method to the runner-up of the AES competition, namely SERPENT [8]. Refer to Section E.1 for a brief specification of SERPENT. The full diffusion of SERPENT at the bit-level is achieved after 3 rounds. Thus, we set the length of the middle part to 3 in our searches. Since SERPENT uses different S-boxes in each round, its differential/linear behavior depends on the starting round. So, we also searched for DL distinguishers for different starting rounds. Table 7 summarizes our results for SERPENT. One of the first applications of DL analysis was on SERPENT in [11], and this cipher has been the center of attention for much research on DL analysis. The longest DL distinguisher for SERPENT is a 9-round distinguisher with a correlation of 2^{-58} proposed in [11]. Later, in [26], the authors performed experiments on a reduced-round version of the 9-round DL distinguisher and showed that the correlation is $2^{-56.5}$. Using our bit-wise model, we found a 9-round DL distinguisher with a correlation of $2^{-50.95}$.

Application to Simeck Section I.1 briefly describes the Simeck family of lightweight block ciphers. Here, we discuss the application of our method to

Simeck. Table 17, Table 18, and Table 19 describe the specification of our discovered DL distinguishers. The number of rounds required for full diffusion in Simeck-32, Simeck-48, and Simeck-64 is 8, 9, and 11, respectively [37]. Our discoveries also match this fact: we found deterministic 7-, 8-, and 10-round DL distinguishers for Simeck-32, Simeck-48, and Simeck-64, respectively. Very recently, the authors of [61] applied the new MILP/MIQCP-based tool [7] to Simeck and proposed the DL distinguishers for all variants of Simeck for the first time. Using our bit-wise model, we could significantly improve all of the results of [61]. The authors of [61] proposed two types of estimations for the correlation of their distinguishers: the first one is based on the MILP/MIQCP model, and the second one is based on experimental measurements of the correlation for smaller parts of the distinguishers. Referring to [61], one can see that the correlation derived by the MILP/MIQCP model extremely underestimates the real correlation. To have a fair comparison, we compare the correlation of our new distinguishers with the (experimental) correlation of distinguishers reported in [61]. The authors of [61] proposed 14-, 18-, and 25-round DL distinguishers for Simeck-32, Simeck-48, and Simeck-64, with (experimentally measured) correlations of $2^{-15.57}$, $2^{-17.88}$, and $2^{-29.65}$, respectively. However, we found 14-, 18, and 25-round DL distinguishers for the corresponding variants of Simeck with correlations of $2^{-13.92}$, $2^{-15.89}$, and $2^{-27.07}$, respectively. Interestingly, we discovered a 20-round DL distinguisher for Simeck-48 with a data complexity of $2^{43.78}$, improving its DL distinguisher by 2 rounds. Moreover, we provided a 26-round DL distinguisher for Simeck-64, improving its best known DL distinguisher by 1 round.

Application to KNOT Permutation and PRESENT We also applied our method to the KNOT permutation, one of the second-round candidates of LWC. We analyzed the main variant of the KNOT family of lightweight authenticated encryption algorithms, denoted as KNOT-256. The length of the middle part was set to 9, sufficient to capture its effect. Table 16 briefly describes our results for KNOT-256. The only previous result on DL distinguishers for KNOT is a conditional DL distinguisher for 15 rounds of KNOT-256 in [56], applicable to the initialization phase. However, as illustrated in Figure 40b, we propose an unconditional DL distinguisher for 15 rounds of KNOT-256 with a correlation of $2^{-17.20}$, also targeting the initialization phase, with a data limit of 2^{64}. Another significant finding is our discovery of up to 23-round DL distinguishers for KNOT-256, the longest for this permutation, surpassing the 17-round integral distinguisher proposed in [28]. To demonstrate the versatility of our method, we also applied it to PRESENT and proposed DL distinguishers for up to 13 rounds of this cipher for the first time (see Table 12).

5 Conclusion and Future Works

In this paper, we presented a general framework for formalizing the correlation of DL distinguishers, along with new CP-based models designed to search for high-quality distinguishers efficiently. We proposed two CP/MILP-based models:

a cell-wise model suitable for strongly aligned primitives and a bit-wise model suitable for weakly aligned primitives. Our new CP/MILP-based models are efficient and user-friendly, allowing for easy incorporation of future improvements in the search for either differential or linear characteristics to find better DL distinguishers. To demonstrate the usefulness and versatility of our new tools, we applied them to a wide range of symmetric-key primitive designs. In all applications, our DL distinguishers exhibit superior correlation and/or cover more rounds than previously known ones. In several instances, we showed that DL distinguishers can surpass boomerang distinguishers or even the best-known (integral) distinguishers. Our work enables further exploration of the similarities between boomerang and DL cryptanalysis and demonstrates that one can adapt many advances in one area to the other.

Our research also raises several open questions and suggests directions for future studies. Our method for identifying DL distinguishers can be applied to other cryptographic primitives. Extending our bit-wise model to ARX primitives could be a potential future work. We also suggest considering neutral bits [9] while searching for DL distinguishers as another future work. Moreover, it is worth inspecting the impact of our approach for finding rotational DL distinguishers [42, 45]. Another interesting avenue for future work is extending our models to a unified CP model for finding complete DL key recovery.

In our applications to SKINNY, we observed that the experimental correlation of the distinguisher is often much higher than the approximation provided by the prq^2 formula. It might be due to partial key addition. Therefore, another interesting area for future work would be to accurately consider the dependencies between rounds for ciphers with partial key addition, potentially leading to the discovery of better distinguishers. Additionally, during our analyses of several symmetric-key primitives, we noticed that the DLCT of many S-boxes contains numerous bit-wise switches, rendering them weaker against DL cryptanalysis. Thus, further studies on constructing strong S-boxes against DL cryptanalysis appear necessary. For example, investigating the application of DDLCT or t-DLCT in the design and analysis of S-boxes resistant to DL cryptanalysis would be intriguing.

Acknowledgments. This work has been supported in part by the Austrian Science Fund (FWF SFB project SPyCoDe), by the France 2030 program under grant agreement No. ANR-22-PECY-0010, by the French Agence Nationale de la Recherche through the OREO project under Contract ANR-22-CE39-0015 and it has been partially funded by the European Union (ERC-2023-COG, SoBaSyC, 101125450). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. The first author would like to thank Fredrick Meisingseth for reading the manuscript and providing valuable feedback.

References

1. Abdelkhalek, A., Sasaki, Y., Todo, Y., Tolba, M., Youssef, A.M.: MILP modeling for (large) s-boxes to optimize probability of differential characteristics. IACR Trans. Symmetric Cryptol. **2017**(4), 99–129 (2017). https://doi.org/10.13154/TOSC.V2017.I4.99-129
2. Banik, S., Bao, Z., Isobe, T., Kubo, H., Liu, F., Minematsu, K., Sakamoto, K., Shibata, N., Shigeri, M.: WARP : Revisiting GFN for lightweight 128-bit block cipher. In: SAC. LNCS, vol. 12804, pp. 535–564. Springer (2020). https://doi.org/10.1007/978-3-030-81652-0_21
3. Bar-On, A., Dunkelman, O., Keller, N., Weizman, A.: DLCT: A new tool for differential-linear cryptanalysis. In: EUROCRYPT 2019. LNCS, vol. 11476, pp. 313–342. Springer (2019). https://doi.org/10.1007/978-3-030-17653-2_11
4. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: The SIMON and SPECK lightweight block ciphers. In: DAC 2015. pp. 175:1–175:6. ACM (2015). https://doi.org/10.1145/2744769.2747946
5. Beierle, C., Jean, J., Kölbl, S., Leander, G., Moradi, A., Peyrin, T., Sasaki, Y., Sasdrich, P., Sim, S.M.: The SKINNY family of block ciphers and its low-latency variant MANTIS. In: CRYPTO 2016. pp. 123–153. Springer (2016). https://doi.org/10.1007/978-3-662-53008-5_5
6. Beierle, C., Leander, G., Todo, Y.: Improved differential-linear attacks with applications to ARX ciphers. In: CRYPTO 2020. LNCS, vol. 12172, pp. 329–358. Springer (2020). https://doi.org/10.1007/978-3-030-56877-1_12
7. Bellini, E., Gérault, D., Grados, J., Makarim, R.H., Peyrin, T.: Fully automated differential-linear attacks against ARX ciphers. In: CT-RSA 2023. LNCS, vol. 13871, pp. 252–276. Springer (2023). https://doi.org/10.1007/978-3-031-30872-7_10
8. Biham, E., Anderson, R.J., Knudsen, L.R.: Serpent: A new block cipher proposal. In: Vaudenay, S. (ed.) FSE'98. LNCS, vol. 1372, pp. 222–238. Springer (1998). https://doi.org/10.1007/3-540-69710-1_15
9. Biham, E., Chen, R.: Near-collisions of SHA-0. In: Franklin, M.K. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 290–305. Springer (2004). https://doi.org/10.1007/978-3-540-28628-8_18
10. Biham, E., Dunkelman, O., Keller, N.: Enhancing differential-linear cryptanalysis. In: ASIACRYPT 2002. LNCS, vol. 2501, pp. 254–266. Springer (2002). https://doi.org/10.1007/3-540-36178-2_16
11. Biham, E., Dunkelman, O., Keller, N.: Differential-linear cryptanalysis of Serpent. In: Johansson, T. (ed.) FSE 2003. LNCS, vol. 2887, pp. 9–21. Springer (2003). https://doi.org/10.1007/978-3-540-39887-5_2
12. Biham, E., Shamir, A.: Differential cryptanalysis of DES-like cryptosystems. In: Menezes, A., Vanstone, S.A. (eds.) CRYPTO '90. LNCS, vol. 537, pp. 2–21. Springer (1990). https://doi.org/10.1007/3-540-38424-3_1
13. Blondeau, C., Leander, G., Nyberg, K.: Differential-linear cryptanalysis revisited. J. Cryptol. **30**(3), 859–888 (2017). https://doi.org/10.1007/s00145-016-9237-5
14. Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C.: PRESENT: an ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer (2007). https://doi.org/10.1007/978-3-540-74735-2_31

15. Bonnetain, X., Lallemand, V.: On boomerang attacks on quadratic feistel ciphers new results on KATAN and simon. IACR Trans. Symmetric Cryptol. **2023**(3), 101–145 (2023). https://doi.org/10.46586/TOSC.V2023.I3.101-145
16. Boukerrou, H., Huynh, P., Lallemand, V., Mandal, B., Minier, M.: On the feistel counterpart of the boomerang connectivity table introduction and analysis of the FBCT. IACR Trans. Symmetric Cryptol. **2020**(1), 331–362 (2020). https://doi.org/10.13154/TOSC.V2020.I1.331-362
17. Canteaut, A., Kölsch, L., Li, C., Li, C., Li, K., Qu, L., Wiemer, F.: Autocorrelations of vectorial boolean functions. In: Longa, P., Ràfols, C. (eds.) LATINCRYPT 2021. LNCS, vol. 12912, pp. 233–253. Springer (2021). https://doi.org/10.1007/978-3-030-88238-9_12
18. Chen, Y., Bao, Z., Yu, H.: Differential-linear approximation semi-unconstrained searching and partition tree: Application to LEA and Speck. IACR Cryptology ePrint Archive, Paper 2023/1414 (2023), https://eprint.iacr.org/2023/1414
19. Cid, C., Huang, T., Peyrin, T., Sasaki, Y., Song, L.: Boomerang connectivity table: A new cryptanalysis tool. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10821, pp. 683–714. Springer (2018). https://doi.org/10.1007/978-3-319-78375-8_22
20. Daemen, J., Rijmen, V.: The Design of Rijndael: AES – The Advanced Encryption Standard. Information Security and Cryptography, Springer (2002). https://doi.org/10.1007/978-3-662-04722-4
21. Delaune, S., Derbez, P., Vavrille, M.: Catching the fastest boomerangs application to SKINNY. IACR Trans. Symmetric Cryptol. **2020**(4), 104–129 (2020). https://doi.org/10.46586/tosc.v2020.i4.104-129
22. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Cryptanalysis of Ascon. In: Nyberg, K. (ed.) CT-RSA 2015. LNCS, vol. 9048, pp. 371–387. Springer (2015). https://doi.org/10.1007/978-3-319-16715-2_20
23. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon v1.2: Lightweight authenticated encryption and hashing. Journal of Cryptology **34**(3), 33 (2021). https://doi.org/10.1007/s00145-021-09398-9
24. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon v1.2 (Submission to NIST). Finalist submission to the NIST lightweight cryptography standardization process (2021), https://csrc.nist.gov/Projects/Lightweight-Cryptography
25. Dong, X., Qin, L., Sun, S., Wang, X.: Key guessing strategies for linear key-schedule algorithms in rectangle attacks. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022. LNCS, vol. 13277, pp. 3–33. Springer (2022). https://doi.org/10.1007/978-3-031-07082-2_1
26. Dunkelman, O., Indesteege, S., Keller, N.: A differential-linear attack on 12-round Serpent. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) INDOCRYPT 2008. LNCS, vol. 5365, pp. 308–321. Springer (2008). https://doi.org/10.1007/978-3-540-89754-5_24
27. Dunkelman, O., Keller, N., Shamir, A.: A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3G telephony. J. Cryptol. **27**(4), 824–849 (2014). https://doi.org/10.1007/s00145-013-9154-9
28. Ghosh, S., Dunkelman, O.: Automatic search for bit-based division property. In: Longa, P., Ràfols, C. (eds.) LATINCRYPT 2021. LNCS, vol. 12912, pp. 254–274. Springer (2021). https://doi.org/10.1007/978-3-030-88238-9_13
29. Gurobi Optimization, LLC: Gurobi Optimizer Reference Manual (2022), https://www.gurobi.com

30. Hadipour, H., Bagheri, N.: Improved rectangle attacks on SKINNY and CRAFT. IACR Trans. Symmetric Cryptol. **2021**(2), 140–198 (2021). https://doi.org/10.46586/tosc.v2021.i2.140-198
31. Hadipour, H., Eichlseder, M.: Integral cryptanalysis of WARP based on monomial prediction. IACR Trans. Symmetric Cryptol. **2022**(2), 92–112 (2022). https://doi.org/10.46586/tosc.v2022.i2.92-112
32. Hadipour, H., Gerhalter, S., Sadeghi, S., Eichlseder, M.: Improved search for integral, impossible differential and zero-correlation attacks application to ascon, forkskinny, skinny, mantis, PRESENT and qarmav2. IACR Trans. Symmetric Cryptol. **2024**(1), 234–325 (2024). https://doi.org/10.46586/TOSC.V2024.I1.234-325
33. Hadipour, H., Nageler, M., Eichlseder, M.: Throwing boomerangs into feistel structures application to CLEFIA, WARP, LBlock, LBlock-s and TWINE. IACR Trans. Symmetric Cryptol. **2022**(3), 271–302 (2022). https://doi.org/10.46586/tosc.v2022.i3.271-302
34. Hadipour, H., Sadeghi, S., Eichlseder, M.: Finding the impossible: Automated search for full impossible-differential, zero-correlation, and integral attacks. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023. LNCS, vol. 14007, pp. 128–157. Springer (2023). https://doi.org/10.1007/978-3-031-30634-1_5
35. Hadipour, H., Sadeghi, S., Niknam, M.M., Song, L., Bagheri, N.: Comprehensive security analysis of CRAFT. IACR Trans. Symmetric Cryptol. **2019**(4), 290–317 (2019). https://doi.org/10.13154/TOSC.V2019.I4.290-317
36. Huang, T., Tjuawinata, I., Wu, H.: Differential-linear cryptanalysis of ICEPOLE. In: Leander, G. (ed.) FSE 2015. LNCS, vol. 9054, pp. 243–263. Springer (2015). https://doi.org/10.1007/978-3-662-48116-5_12
37. Kölbl, S., Roy, A.: A brief comparison of Simon and Simeck. In: Bogdanov, A. (ed.) Lightweight Cryptography for Security and Privacy - 5th International Workshop, LightSec 2016, Aksaray, Turkey, September 21-22, 2016, Revised Selected Papers. LNCS, vol. 10098, pp. 69–88. Springer (2016). https://doi.org/10.1007/978-3-319-55714-4_6
38. Langford, S.K., Hellman, M.E.: Differential-linear cryptanalysis. In: CRYPTO '94. vol. 839, pp. 17–25. Springer (1994). https://doi.org/10.1007/3-540-48658-5_3
39. Leurent, G.: Improved differential-linear cryptanalysis of 7-round Chaskey with partitioning. In: Fischlin, M., Coron, J. (eds.) EUROCRYPT 2016. LNCS, vol. 9665, pp. 344–371. Springer (2016). https://doi.org/10.1007/978-3-662-49890-3_14
40. Li, M., Sun, L., Wang, M.: Automated key recovery attacks on round-reduced Orthros. In: Batina, L., Daemen, J. (eds.) AFRICACRYPT 2022. LNCS, vol. 13503, pp. 189–213. Springer Nature Switzerland (2022). https://doi.org/10.1007/978-3-031-17433-9_9
41. Liu, M., Lu, X., Lin, D.: Differential-linear cryptanalysis from an algebraic perspective. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12827, pp. 247–277. Springer (2021). https://doi.org/10.1007/978-3-030-84252-9_9
42. Liu, Y., Sun, S., Li, C.: Rotational cryptanalysis from a differential-linear perspective - practical distinguishers for round-reduced friet, xoodoo, and alzette. In: Canteaut, A., Standaert, F. (eds.) EUROCRYPT 2021. LNCS, vol. 12696, pp. 741–770. Springer (2021). https://doi.org/10.1007/978-3-030-77870-5_26
43. Matsui, M.: Linear cryptanalysis method for DES cipher. In: Helleseth, T. (ed.) EUROCRYPT '93. LNCS, vol. 765, pp. 386–397. Springer (1993). https://doi.org/10.1007/3-540-48285-7_33

44. Nethercote, N., Stuckey, P.J., Becket, R., Brand, S., Duck, G.J., Tack, G.: Minizinc: Towards a standard CP modelling language. In: CP 2007. LNCS, vol. 4741, pp. 529–543. Springer (2007)
45. Niu, Z., Sun, S., Liu, Y., Li, C.: Rotational differential-linear distinguishers of ARX ciphers with arbitrary output linear masks. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022. LNCS, vol. 13507, pp. 3–32. Springer (2022). https://doi.org/10.1007/978-3-031-15802-5_1
46. Nyberg, K.: Differentially uniform mappings for cryptography. In: Helleseth, T. (ed.) EUROCRYPT '93. LNCS, vol. 765, pp. 55–64. Springer (1993). https://doi.org/10.1007/3-540-48285-7_6
47. Perron, L., Furnon, V.: OR-Tools, https://developers.google.com/optimization/
48. Qin, L., Dong, X., Wang, X., Jia, K., Liu, Y.: Automated search oriented to key recovery on ciphers with linear key schedule applications to boomerangs in SKINNY and forkskinny. IACR Trans. Symmetric Cryptol. **2021**(2), 249–291 (2021). https://doi.org/10.46586/TOSC.V2021.I2.249-291
49. Sage Developers: SageMath, the Sage Mathematics Software System (Version 9.5.0) (2022), https://www.sagemath.org
50. Shirai, T., Shibutani, K.: Improving immunity of feistel ciphers against differential cryptanalysis by using multiple MDS matrices. In: FSE 2004. LNCS, vol. 3017, pp. 260–278. Springer (2004). https://doi.org/10.1007/978-3-540-25937-4_17
51. Shirai, T., Shibutani, K., Akishita, T., Moriai, S., Iwata, T.: The 128-bit blockcipher CLEFIA (extended abstract). In: FSE 2007. LNCS, vol. 4593, pp. 181–195. Springer (2007). https://doi.org/10.1007/978-3-540-74619-5_12
52. Sun, L., Gérault, D., Wang, W., Wang, M.: On the usage of deterministic (related-key) truncated differentials and multidimensional linear approximations for SPN ciphers. IACR Trans. Symmetric Cryptol. **2020**(3), 262–287 (2020). https://doi.org/10.13154/tosc.v2020.i3.262-287
53. Sun, L., Wang, W., Wang, M.: More accurate differential properties of LED64 and midori64. IACR Trans. Symmetric Cryptol. **2018**(3), 93–123 (2018). https://doi.org/10.13154/TOSC.V2018.I3.93-123
54. Suzaki, T., Minematsu, K., Morioka, S., Kobayashi, E.: Twine: A lightweight block cipher for multiple platforms. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 339–354. Springer (2012). https://doi.org/10.1007/978-3-642-35999-6_22
55. Wagner, D.A.: The boomerang attack. In: Knudsen, L.R. (ed.) FSE '99. LNCS, vol. 1636, pp. 156–170. Springer (1999). https://doi.org/10.1007/3-540-48519-8_12
56. Wang, S., Hou, S., Liu, M., Lin, D.: Differential-linear cryptanalysis of the lightweight crytographic algorithm KNOT. In: Yu, Y., Yung, M. (eds.) Inscrypt 2021. LNCS, vol. 13007, pp. 171–190. Springer (2021). https://doi.org/10.1007/978-3-030-88323-2_9
57. Wu, W., Zhang, L.: Lblock: A lightweight block cipher. In: López, J., Tsudik, G. (eds.) ACNS 2011. LNCS, vol. 6715, pp. 327–344 (2011). https://doi.org/10.1007/978-3-642-21554-4_19
58. Yang, G., Zhu, B., Suder, V., Aagaard, M.D., Gong, G.: The simeck family of lightweight block ciphers. In: Güneysu, T., Handschuh, H. (eds.) CHES 2015. LNCS, vol. 9293, pp. 307–329. Springer (2015). https://doi.org/10.1007/978-3-662-48324-4_16

59. Zhang, W., Ding, T., Yang, B., Bao, Z., Xiang, Z., Ji, F., Zhao, X.: KNOT: Algorithm specifications and supporting document. Submission to NIST lightweight cryptography project (2019)
60. Zhang, W., Zhang, L., Wu, W., Feng, D.: Related-key differential-linear attacks on reduced AES-192. In: Srinathan, K., Rangan, C.P., Yung, M. (eds.) INDOCRYPT 2007. LNCS, vol. 4859, pp. 73–85. Springer (2007). https://doi.org/10.1007/978-3-540-77026-8_7
61. Zhou, Y., Wang, S., Hu, B.: MILP/MIQCP-based fully automatic method of searching for differential-linear distinguishers for SIMON-like ciphers. IET Information Security **2024** (2024). https://doi.org/10.1049/2024/8315115

— Supplementary Material —

A Comparison of our Approach with the State-of-the-Art

Here, we briefly compare our approach with the state-of-the-art in DL analysis. After introducing DL cryptanalysis in CRYPTO 1994 [38], efforts to formalize computing the correlation of DL distinguishers have been extensive, while automation of the search for DL distinguishers has received less attention. One of the most interesting works is that of Blondeau, Leander, and Nyberg [13]. They relaxed one of the two critical assumptions from the original work of Langford and Hellman, providing a closed formula to compute the correlation of DL distinguishers. In EUROCRYPT 2019, Bar-On et al. [3] further relaxed another critical assumption in computing the correlation of DL distinguishers. They proposed the DLCT and utilized the sandwich framework to formulate the correlation of DL distinguishers. Another interesting work is that of Liu et al. [41], presented at CRYPTO 2021. They introduce a purely algebraic approach to compute the correlation of DL distinguishers across multiple rounds. Recently, at EUROCRYPT 2021, Liu et al. [42] combined the concepts of rotational differentials and DL analysis. They proposed a novel combined attack known as rotational DL analysis. This work was further extended in CRYPTO 2022 [45], presenting an efficient algorithm for computing the (rotational) differential-linear correlation of modular additions for arbitrary output linear masks. However, most of these interesting works have focused mainly on formalizing the correlation of DL distinguishers and rarely on providing a (generic CP-based) automatic tool for finding DL distinguishers. Other interesting works have also aimed to improve the key recovery of DL attacks, such as the one by Beierle et al. presented at CRYPTO 2020 [6].

Only very recently have there been some interesting works regarding the automatic search for DL distinguishers. For example, in 2023, Bellini et al. [7] introduced an automated tool based on MILP/MIQCP for identifying DL distinguishers in ARX ciphers and applied it to Speck-32. However, the authors acknowledge that their CP model is resource-intensive, limiting its application to smaller variants of Speck like Speck-32. Furthermore, the tool's efficiency concerning SPN ciphers remains an open question. Later, Zhou et al. [61] applied this method to Simon and Simeck. In another recent work from ASIACRYPT 2023, Chen et al. [18] proposed an alternative method for searching for DL distinguishers. However, their approach is also tailored to ARX ciphers and does not leverage general-purpose CP/MILP solvers. Developing an efficient and generic approach to automatically search for effective DL distinguishers across various classes of primitives, especially SPN designs, remained an open problem. In our paper, we initially examine the interdependency between two DL distinguishers components from the boomerang analysis perspective to extend the framework proposed in [3]. Subsequently, we introduce a novel tool for automating the search for DL distinguishers.

The advantage of our approach is that it allows us to capitalize on the progress made in boomerang analysis within DL analysis. Our CP-based automatic tool also addresses the need for a generic and efficient CP-based method for exploring DL distinguishers, especially for strongly aligned SPN ciphers. Additionally, our approach can be integrated with some intriguing previous approaches, such as the algebraic one presented in CRYPTO 2021 [41]. For instance, while our generalized DLCT framework is very efficient for formulating/computing the correlation across multiple rounds of strongly aligned primitives, it is more complicated to use it for weakly aligned primitives, where the algebraic approach in [41] might be more efficient. However, our tool for finding the DL distinguishers performs well for strongly and weakly aligned primitives. Therefore, one can use our automatic tool for finding DL distinguishers and then use either the generalized DLCT framework or the algebraic approach in [41] to compute the correlation of the middle part.

B Constraint Satisfaction and Constraint Optimization Problems

A constraint satisfaction problem (CSP) is a mathematical problem including a set of constraints over a set of variables that should be satisfied. The following definition is a formal definition of CSPs.

Definition 13. *A CSP is a triple* $(\mathcal{X}, \mathcal{D}, \mathcal{C})$*, where*

- $\mathcal{X} = \{X_0, X_1, \ldots, X_{n-1}\}$ *is a set of variables;*
- $\mathcal{D} = \{\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_{n-1}\}$ *is the set of domains such that* $X_i \in \mathcal{D}_i$*,* $0 \leq i \leq n-1$*; and*
- $\mathcal{C} = \{\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_{n-1}\}$ *is a set of constraints.*

Each constraint $\mathcal{C}_j \in \mathcal{C}$ *is a tuple* $(\mathcal{S}_j, \mathcal{R}_j)$*, where* $\mathcal{S}_j = \{X_{i_0}, \ldots, X_{i_{k-1}}\} \subseteq \mathcal{X}$ *and* $\mathcal{R}_j$ *is a relation on the corresponding domains, i.e.,* $\mathcal{R}_j \subseteq \mathcal{D}_{i_0} \times \cdots \times \mathcal{D}_{i_{k-1}}$*.*

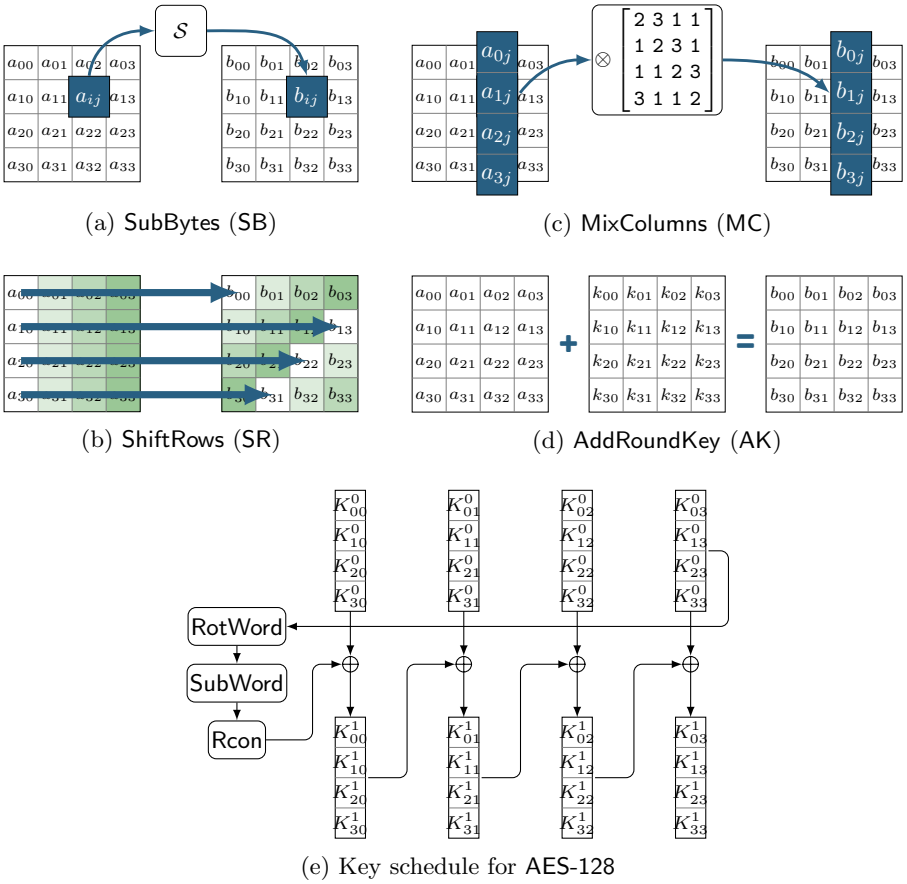
Any assignment of domain values to the variables that satisfies all constraints of a CSP problem is a feasible solution. Including an objective function to be minimized (or maximized) in a CSP problem results in a constraint optimization problem (COP). We refer to finding a feasible solution for a CSP or COP problem as constraint programming (CP). The tools that are used to solve CSP and COP problems are called CP solvers.

To generate our COP models in this paper, we use MiniZinc [44]. The main advantage of MiniZinc is that it allows modeling the CSP and COP problems in a high-level and solver-independent way. It compiles the model into FlatZinc, a standard language supported by a wide range of CP solvers. As a result, once a model is written in MiniZinc, it can be solved by any CP solver that supports FlatZinc, which includes a wide range of powerful CP solvers, e.g., Gurobi [29], and Or-Tools [47]. In this paper we use Gurobi and Or-Tools as the CP solvers.

C Application to AES

C.1 Brief Specification of AES

The AES family of 128-bit block ciphers with key sizes $k \in \{128, 192, 256\}$ bits was designed by Rijmen and Daemen [20] and standardized in NIST FIPS PUB 197 in 2001. The state of all family members is a 4×4 matrix of bytes, while the key is a $4 \times \{4, 6, 8\}$ matrix depending on the key size. The state is updated in 10, 12, or 14 rounds, respectively. The round function is illustrated in Figure 7 and consists of the operations SubBytes, ShiftRows, MixColumns, and AddRoundKey. An additional round key is added before the first round, while MixColumns is omitted in the last round.



(a) SubBytes (SB)

(c) MixColumns (MC)

(b) ShiftRows (SR)

(d) AddRoundKey (AK)

(e) Key schedule for AES-128

Fig. 7: Round function of AES.

C.2 The DL Distinguishers of AES

Table 5: DL distinguishers for 2 to 5 rounds of AES (single-key).

2 Rounds, Figure 9a $r_0 = 0, r_m = 2, r_1 = 0,\ p = 1,\ r = 1, q = 1,\ prq^2 = \mathbf{1}$	
ΔX_0 01000000000100000000010000000001	ΓX_2 00000000008500000006000000000000
3 Rounds, Figure 9b $r_0 = 0, r_m = 3, r_1 = 0,\ p = 1,\ r = 2^{-7.66}, q = 1,\ prq^2 = \mathbf{2^{-7.66}}$	
ΔX_0 00000000000000000000000000b400	ΓX_3 0032000000ab00000066000000980000
4 Rounds, Figure 9c $r_0 = 1, r_m = 3, r_1 = 0,\ p = 2^{-24},\ r = 2^{-7.66}, q^2 = 1,\ prq^2 = \mathbf{2^{-31.66}}$	
ΔX_0 00005200000000f58f000000007b0000	ΔX_1 00000000000000000000000000b400
ΓX_4 0032000000ab00000066000000980000	-
5 Rounds, Figure 9d $r_0 = 1, r_m = 3, r_1 = 1,\ p = 2^{-24.00}, r = 2^{-7.66},\ q^2 = 2^{-24.00},\ prq^2 = 2^{-55.66}$	
ΔX_0 00005200000000f58f000000007b0000	ΔX_1 00000000000000000000000000b400
ΓX_4 0032000000ab00000066000000980000	ΓX_5 208acd121f4b3ff232f46e51299eda33

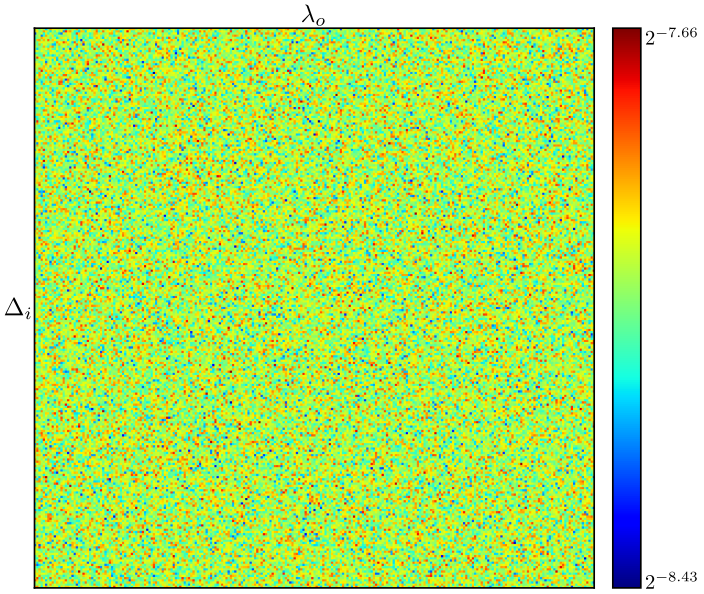
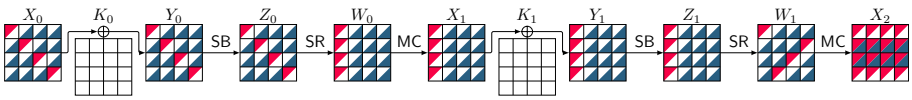
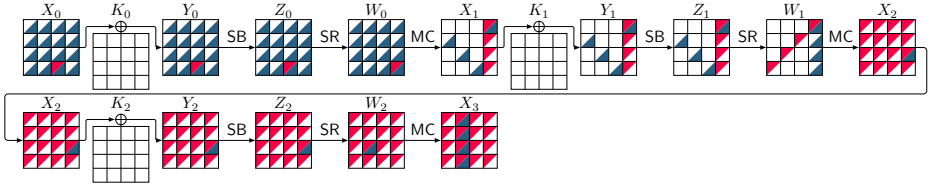


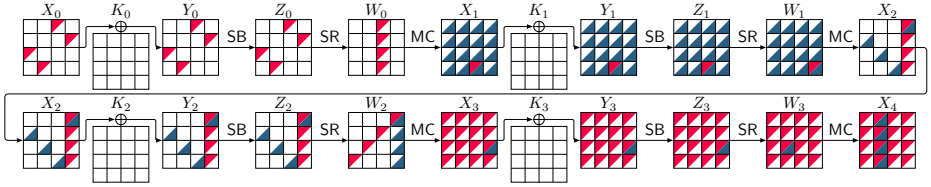
Fig. 8: Correlation matrix visualization for 3-round AES DL distinguisher.



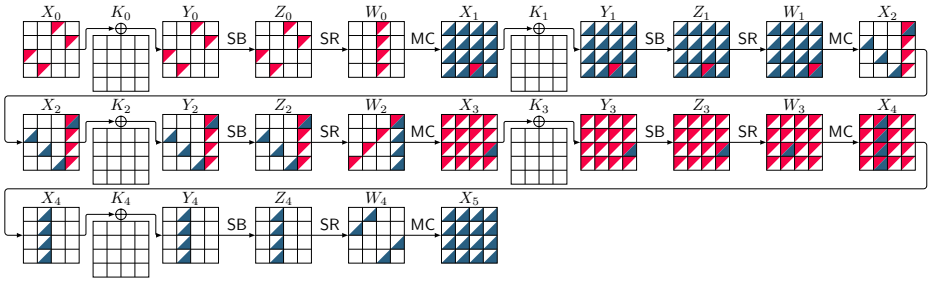
(a) DL distinguisher for 2 rounds of AES



(b) DL distinguisher for 3 rounds of AES



(c) DL distinguisher for 4 rounds of AES



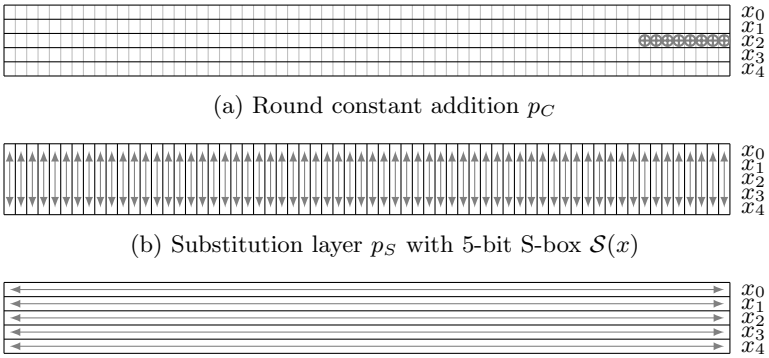
(d) DL distinguisher for 5 rounds of AES

Fig. 9: DL distinguishers for 2 to 5 rounds of AES (difference linear mask).

D Application to Ascon

D.1 Brief Specification of Ascon

Ascon is a family of authenticated encryption and hashing designed by the Ascon team [23,24] and has been selected by NIST as the new standard for lightweight cryptography (LWC). Its underlying primitive is a 320-bit permutation. This permutation is defined with a different number of rounds (6, 8, or 12) for different phases of the encryption scheme. The SPN-based round transformation p consists of three steps, $p = p_L \circ p_S \circ p_C$. Ascon's 320-bit state S is split into five 64-bit registers words x_i, $S = x_0 \| x_1 \| x_2 \| x_3 \| x_4$ (see Figure 10).



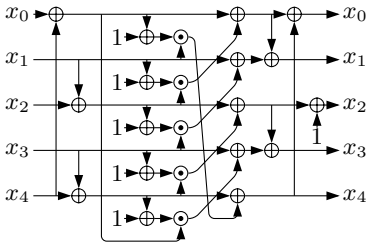
(a) Round constant addition p_C

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

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

Fig. 10: 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 11a to each bit-slice of the five registers $x_0 \dots x_4$. The linear diffusion layer p_L applies a linear function $\Sigma_i(x_i)$ defined in Figure 11b 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 $\Sigma_i(x_i)$

Fig. 11: Ascon's substitution layer and linear diffusion layer.

D.2 The DL Distinguishers of Ascon

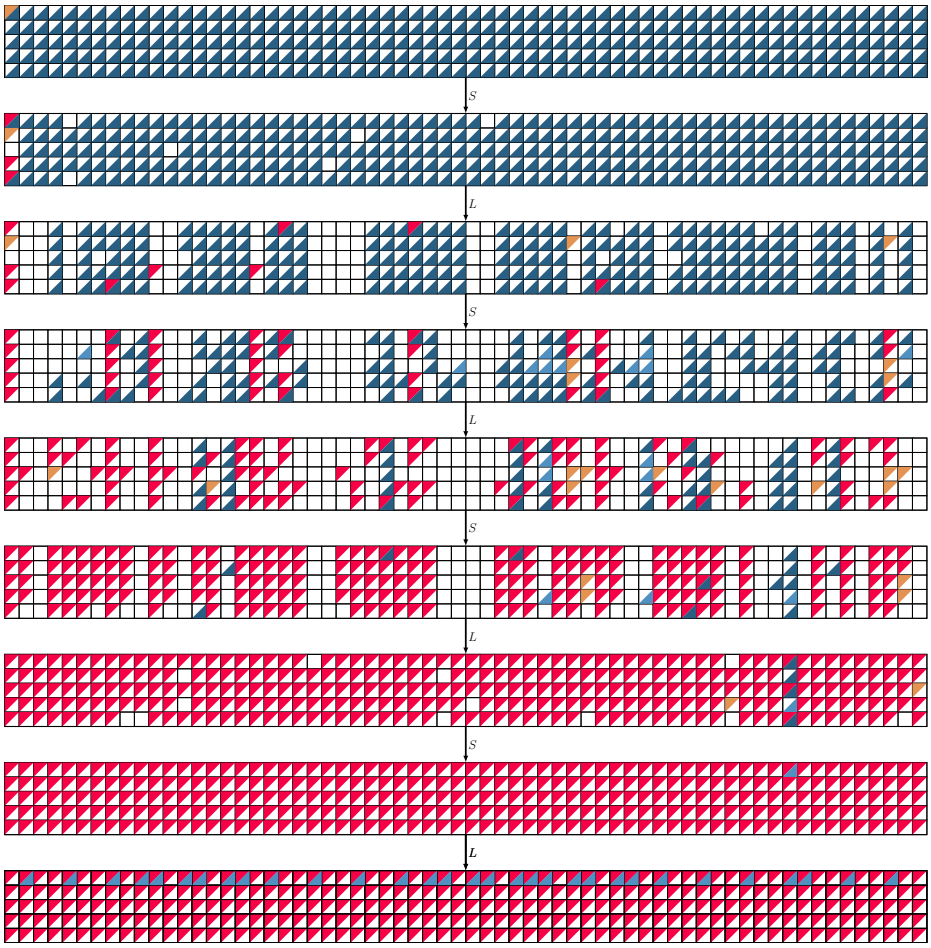


Fig. 12: DL distinguisher II for 4 rounds of Ascon.

Table 6: Specification of DL distinguishers for Ascon ($\hat{\mathbb{C}}$: experimental correlatin).

4 Rounds, Figure 6

$r_u = 0, r_m = 4, r_\ell = 0,\ p = 1,\ r = 1,\ q^2 = 1,\ prq^2 = \mathbf{1},\ \hat{\mathbb{C}} = \mathbf{1}$

ΔX_0	ΓX_4
0020000000000000	c9125b6925b76d24
0000000000000000	0000000000000000
0000000000000000	0000000000000000
0020000000000000	0000000000000000
0020000000000000	0000000000000000

4 Rounds, Figure 12

$r_u = 0, r_m = 4, r_\ell = 0,\ p = 1,\ r = 2^{-1},\ q^2 = 1,\ prq^2 = \mathbf{2^{-1}},\ \hat{\mathbb{C}} = \mathbf{2^{-1}}$

ΔX_0	ΓX_4
8000000000000000	496da496ddb49324
0000000000000000	0000000000000000
0000000000000000	0000000000000000
0000000000000000	0000000000000000
0000000000000000	0000000000000000

4 Rounds, Figure 13

$r_u = 0, r_m = 4, r_\ell = 0,\ p = 1,\ r = 2^{-1},\ q^2 = 1,\ prq^2 = \mathbf{2^{-1}},\ \hat{\mathbb{C}} = \mathbf{2^{-1}}$

ΔX_0	ΓX_4
0000000000000000	892db492dbb69264
0100000000000000	ba6e221eea5a47cc
0100000000000000	0000000000000000
0000000000000000	0000000000000000
0000000000000000	0000000000000000

5 Rounds, Figure 14

$r_u = 1, r_m = 3, r_\ell = 1,\ p = 2^{-2},\ r = 1,\ q^2 = 2^{-2},\ prq^2 = 2^{-4},\ \hat{\mathbb{C}} = 2^{-4.33}$

ΔX_0	ΔX_1	ΓX_4	ΓX_5
0000000000000080	0000000000000000	0000000000000000	6da496ddb4932449
0000000000000000	0000000000000000	0000000000000000	7110f752d23e65d3
0000000000000000	00000000000000c2	0000000000020000	0000000000000000
0000000000000080	0000000000000000	0000000000000000	0000000000000000
0000000000000080	0000000000000000	0000000000000000	e631e6e25c7f614b

5 Rounds, Figure 14

$r_u = 1, r_m = 3, r_\ell = 1,\ p = 2^{-2},\ r = 2^{-0.83},\ q^2 = 2^{-4},\ prq^2 = 2^{-6.83},\ \hat{\mathbb{C}} = 2^{-7.61}$

ΔX_0	ΔX_1	ΓX_4	ΓX_5
0000000000000000	0100002010000000	0000000000000080	125b6925b76d24c9
0100000000000000	0000000000000000	0000000000000000	74dc443dd4b48f99
0100000000000000	0000000000000000	0000000000000000	0000000000000000
0000000000000000	0000000000000000	0000000000000000	0000000000000000
0000000000000000	0000000000000000	0000000000000000	0000000000000000

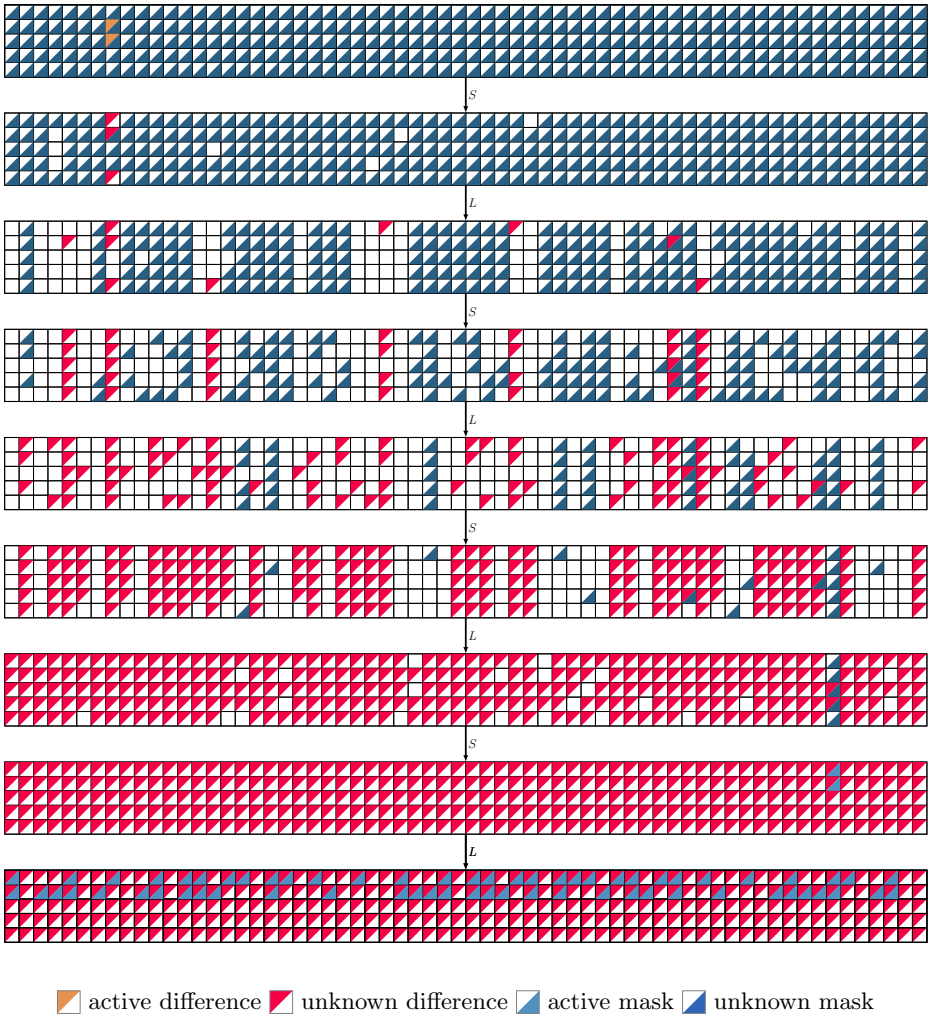


Fig. 13: DL distinguisher III for 4 rounds of Ascon.

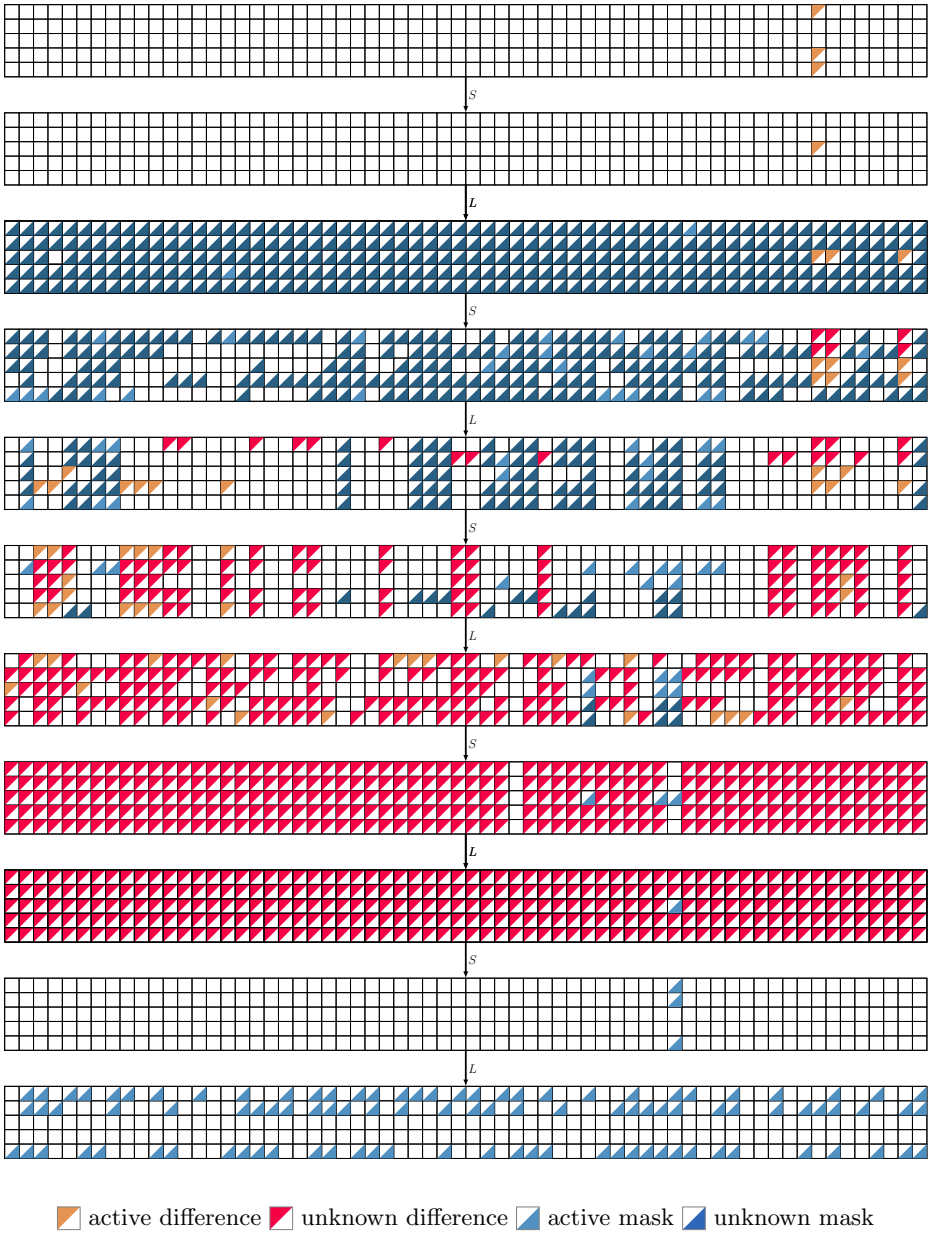


Fig. 14: DL distinguisher I for 5 rounds of Ascon.

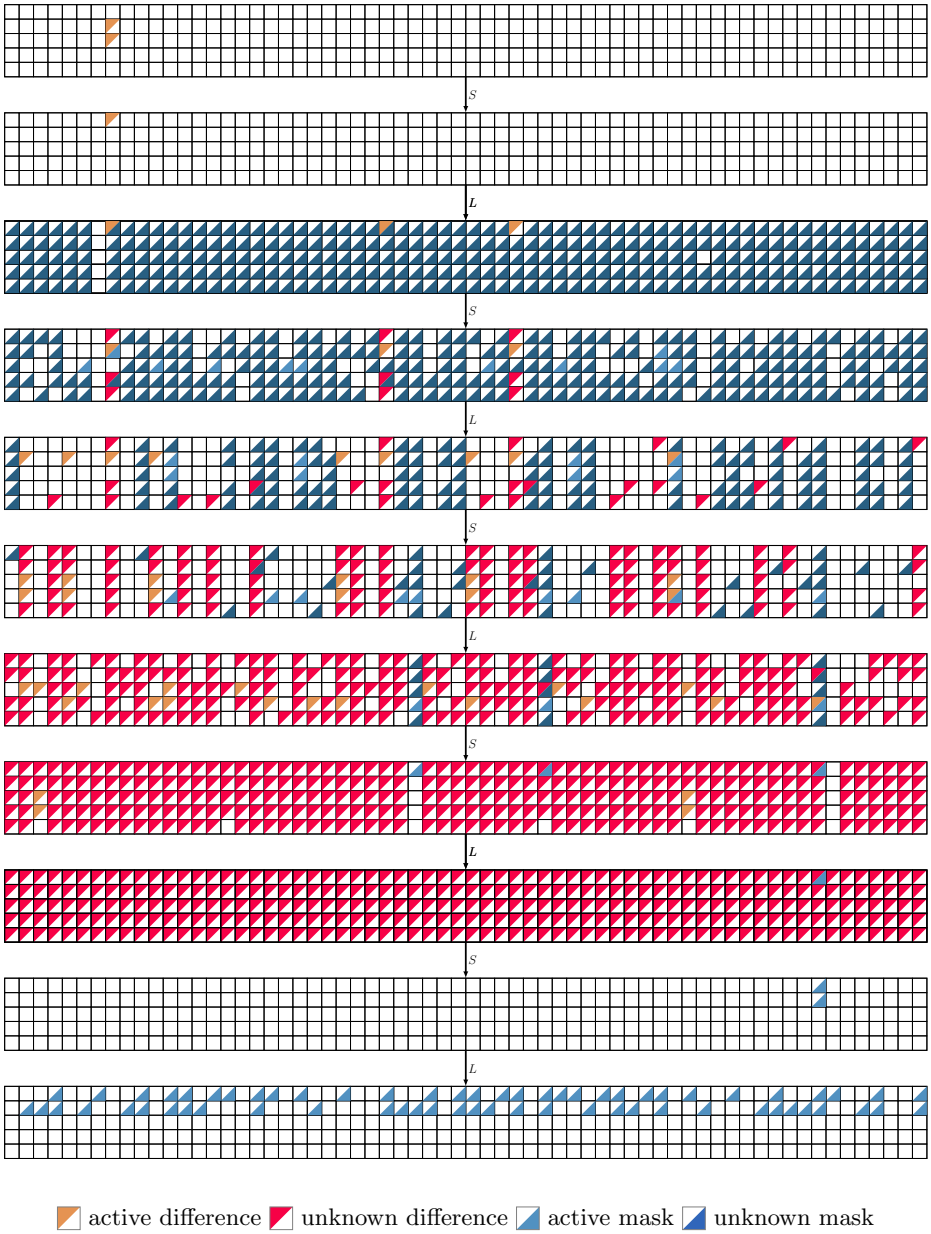
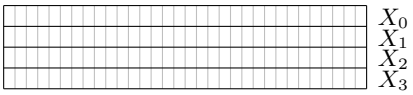


Fig. 15: DL distinguisher II for 5 rounds of Ascon.

E Application to SERPENT

E.1 Brief Specification of SERPENT

The SERPENT family of 128-bit block ciphers with key sizes $k \in \{128, 192, 256\}$ bits was designed by Anderson, Biham, and Knudsen [8] and was a finalist in the AES competition. The state of all family members is a 4×32 matrix of bits, i.e., it is organized in four 32-bit words $X_0, \ldots, X_3$. SERPENT has a very generous security margin with a total of 32 SPN rounds. The round function consists of a round key addition, a bitsliced S-box layer across words, and a linear layer (omitted in the last round) that mixes all four words. The S-box layer alternates between different S-boxes $\mathcal{S}_0, \ldots, \mathcal{S}_7$ in consecutive rounds. Overall, the round function in round i is defined by the instructions in Figure 16, where Y_i is the output state of round $i-1$, K_i is the current round key, and $\lll$, $\ll$ denote left rotation and left shift, respectively.



(a) State layout

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$\mathcal{S}_0(x)$	3	8	f	1	a	6	5	b	e	d	4	2	7	0	9	c
$\mathcal{S}_1(x)$	f	c	2	7	9	0	5	a	1	b	e	8	6	d	3	4
$\mathcal{S}_2(x)$	8	6	7	9	3	c	a	f	d	1	e	4	0	b	5	2
$\mathcal{S}_3(x)$	0	f	b	8	c	9	6	3	d	1	2	4	a	7	5	e
$\mathcal{S}_4(x)$	1	f	8	3	c	0	b	6	2	5	4	a	9	e	7	d
$\mathcal{S}_5(x)$	f	5	2	b	4	a	9	c	0	3	e	8	d	6	7	1
$\mathcal{S}_6(x)$	7	2	c	5	8	4	6	b	e	9	1	f	d	3	a	0
$\mathcal{S}_7(x)$	1	d	f	0	e	8	2	b	7	4	c	a	9	3	5	6

(b) S-boxes of SERPENT

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

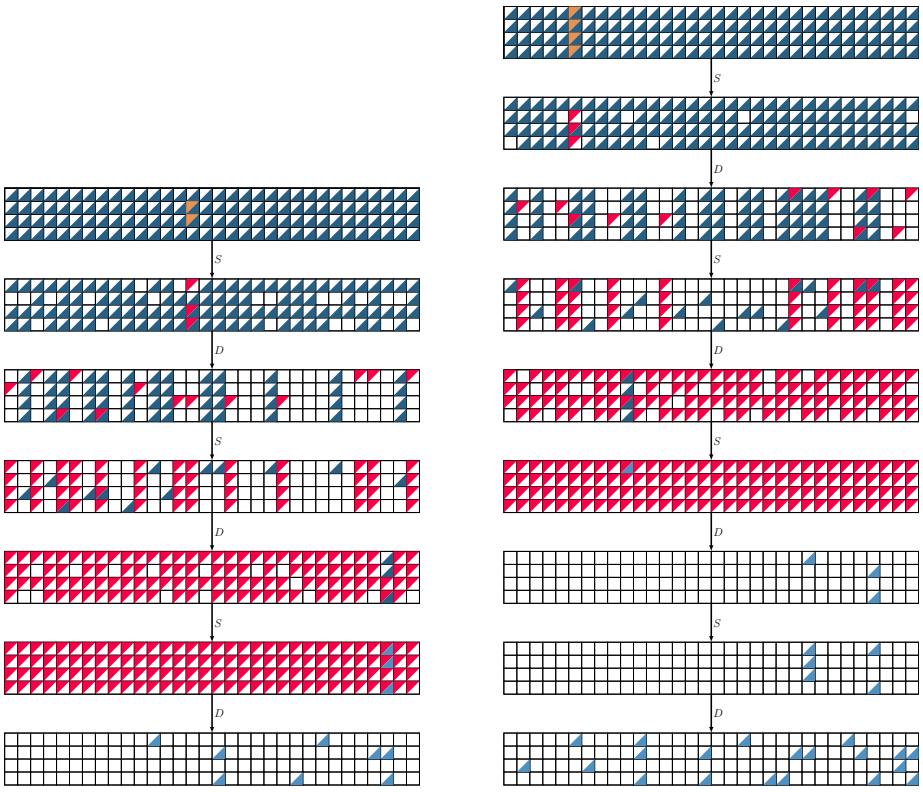
(c) Instructions for round i

Fig. 16: Round function of SERPENT.

E.2 The DL Distinguishers of SERPENT

Table 7: DL distinguishers for 3 to 9 rounds of SERPENT.

3 Rounds, Figure 17a $\texttt{offset} = 6,\ r_0 = 0, r_m = 3, r_1 = 0,\ p = 1,\ r = 2^{-0.68}, q = 1,\ prq^2 = \mathbf{2^{-0.68}}$	
ΔX_0 00000000000200000002000000000000	ΓX_3 00008204000000000000800c00100080
4 Rounds, Figure 17b $\texttt{offset} = 4, r_0 = 0, r_m = 3, r_1 = 1,\ p = 1,\ r = 2^{-1.54}, q = 2^{-4},\ prq^2 = \mathbf{2^{-5.54}}$	
ΔX_0 04000000040000000400000004000000	
ΓX_3 00000080000000000000000800000100	ΓX_4 00210c094200000200210313042020020
5 Rounds, Figure 18a $\texttt{offset} = 2,\ r_0 = 1, r_m = 3, r_1 = 1,\ p = 2^{-4.00},\ r = 2^{-3.10}, q^2 = 2^{-4},\ prq^2 = \mathbf{2^{-11.10}}$	
ΔX_0 00000000090000000100000001000000	ΔX_1 00000000000200000000000000000000
ΓX_4 00000080000000000000008000001000	ΓX_5 02181600000000050210022842004000
6 Rounds, Figure 18b $\texttt{offset} = 2,\ r_0 = 1, r_m = 3, r_1 = 2,\ p = 2^{-4.00}, r = 2^{-8.58},\ q^2 = 2^{-8.00},\ prq^2 = 2^{-20.58}$	
ΔX_0 04000000080000004800000000800000	ΔX_1 00000000000100000000000000000000
ΓX_4 00000100000000000000000000000000	ΓX_6 2000084000908000c28408484001080
7 Rounds, Figure 19a $\texttt{offset} = 1,\ r_0 = 1, r_m = 3, r_1 = 3,\ p = 2^{-5.00}, r = 2^{-7.45},\ q^2 = 2^{-16.00},\ prq^2 = 2^{-28.45}$	
ΔX_0 00800000000000000000000004800000	ΔX_1 00000000000100000000000000000000
ΓX_4 00000004000000000000000200000000	ΓX_7 34a400860009080000cc408410801080
8 Rounds, Figure 19b $\texttt{offset} = 1,\ r_0 = 1, r_m = 3, r_1 = 4,\ p = 2^{-4.00}, r = 2^{-9.18},\ q^2 = 2^{-26.00},\ prq^2 = 2^{-39.18}$	
ΔX_0 40020000400200000002000040020000	ΔX_1 00000000000000000000000000000008
ΓX_4 00000000800000000000000400000000	ΓX_8 00102c0000000042000166100024800
9 Rounds - I, Figure 20a $\texttt{offset} = 2,\ r_0 = 2, r_m = 3, r_1 = 4,\ p = 2^{-7.00}, r = 2^{-7.43},\ q^2 = 2^{-40},\ prq^2 = 2^{-54.43}$	
ΔX_0 00000000000000090000000100000001	ΔX_2 00000001004000000400000080000020
ΓX_5 00001000000000000000100000020000	ΓX_9 04590204080010010009842403208480
9 Rounds - II, Figure 20b $\texttt{offset} = 1,\ r_0 = 2, r_m = 3, r_1 = 4,\ p = 2^{-7.00}, r = 2^{-13.95},\ q^2 = 2^{-30.00},\ prq^2 = 2^{-50.95}$	
ΔX_0 00001000000000000000900000000000	ΔX_2 00001000000000c0000004000020800
ΓX_5 00000100000000000000000000000000	ΓX_9 4216902300808500c847b80008520009



(a) 3-round SERPENT (b) 4-round SERPENT

active difference unknown difference active mask unknown mask

Fig. 17: DL distinguishers for 3 to 4 rounds of SERPENT.

(a) 5-round SERPENT (b) 6-round SERPENT

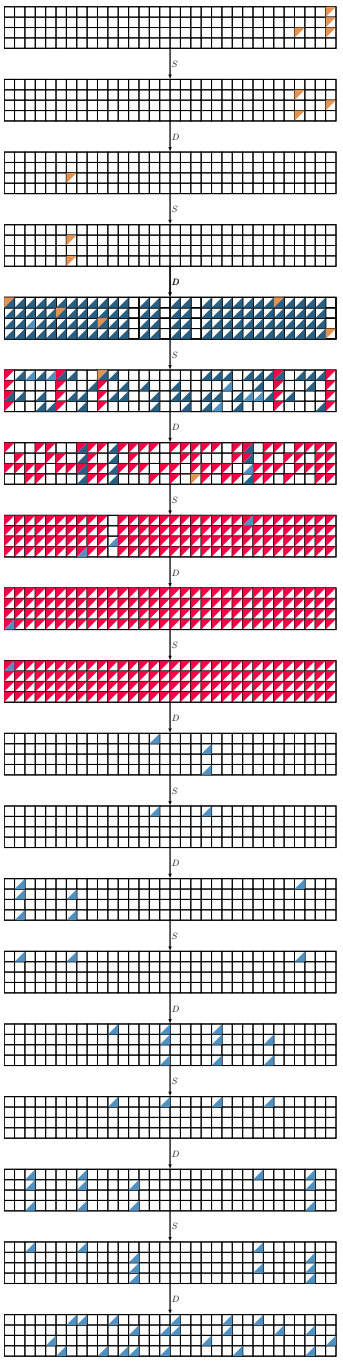
active difference unknown difference active mask unknown mask

Fig. 18: DL distinguishers for 5 to 6 rounds of SERPENT.

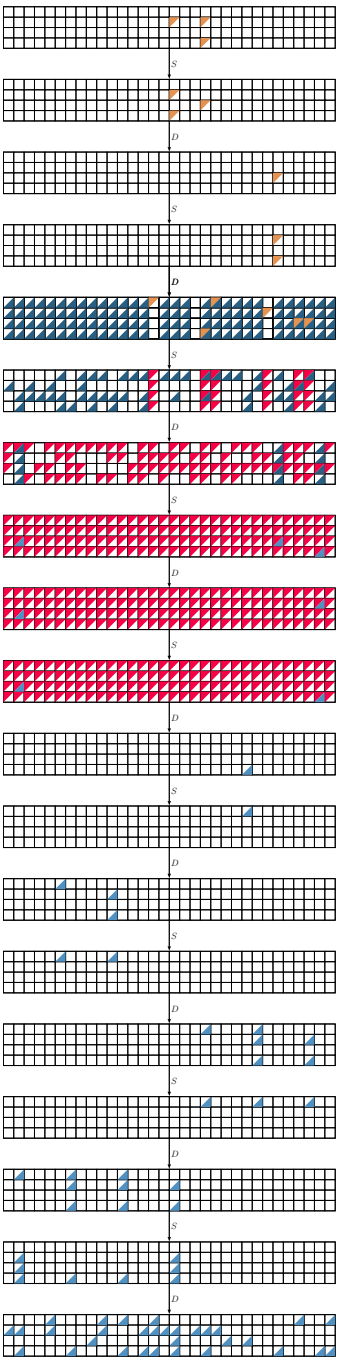
(a) 7-round SERPENT (b) 8-round SERPENT

active difference unknown difference active mask unknown mask

Fig. 19: DL distinguishers for 7 to 8 rounds of SERPENT.



(a) 9-round SERPENT, variant I



(b) 9-round SERPENT, variant II

active difference unknown difference active mask unknown mask

Fig. 20: DL distinguishers for 9 rounds of SERPENT.

F Application to SKINNY

F.1 Brief Specification of SKINNY

SKINNY is a family of tweakable block ciphers introduced by Beierle et al. at CRYPTO 2016 [5]. The SKINNY family offers two block sizes, $n \in \{64, 128\}$, and for each block size there are three tweakey sizes available, $t \in \{n, 2n, 3n\}$. SKINNY-n-t denotes SKINNY with n-bit blocks and t-bit tweakey. The internal state is a 4×4 array of cells, where the cell size is 4 (or 8) bits when $n = 64$ (resp. $n = 128$). The tweakey state consists of z arrays $TK1, TK2, \ldots, TKz$ of 4×4 cells, where $z = \frac{t}{n} \in \{1, 2, 3\}$ depends on the tweakey size.

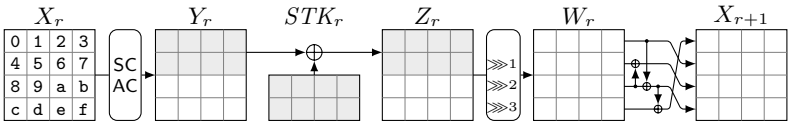


Fig. 21: Round function of SKINNY

Each round of SKINNY applies five basic operations to the internal state: SubCells (SC), AddConstants (AC), AddRoundTweakey (ART), ShiftRows (SR), and MixColumns (MC) (see Figure 21). The SC operation applies a 4-bit (or 8-bit) S-box on each cell. AC XORs the round constant to the internal state. In the ART layer, the cells in the first and the second rows of subtweakey are XORed to the corresponding cells in the internals state. SR applies a permutation P on the position of the state cells, where $P = [0, 1, 2, 3, 7, 4, 5, 6, 10, 11, 8, 9, 13, 14, 15, 12]$. MC multiplies each column of the internal state by a non-MDS matrix M:

$$M = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}, \; M^{-1} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}.$$

The tweakey schedule of SKINNY updates each n-bit tweakey array $TK1, \ldots, TKz$ independently with a linear update function: First, a permutation h is applied to each tweakey array, such that $TKm_r[n] \leftarrow TKm_{r-1}[h(n)]$ for all $0 \leq n \leq 15$, and $m \in \{1, 2, 3\}$. Next, an LFSR is applied to each cell of the first and the second rows of $TK2_r$ and $TK3_r$. The final subtweakey added in the rth round is then the XOR of these z arrays.

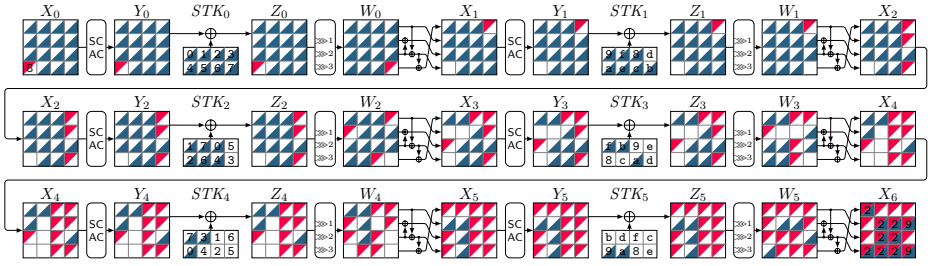
F.2 The DL Distinguishers of SKINNY

Table 8: Specification of DL distinguishers for SKINNY-64 (Single-Tweakey) ($\hat{\mathbb{C}}$: experimental correlation)

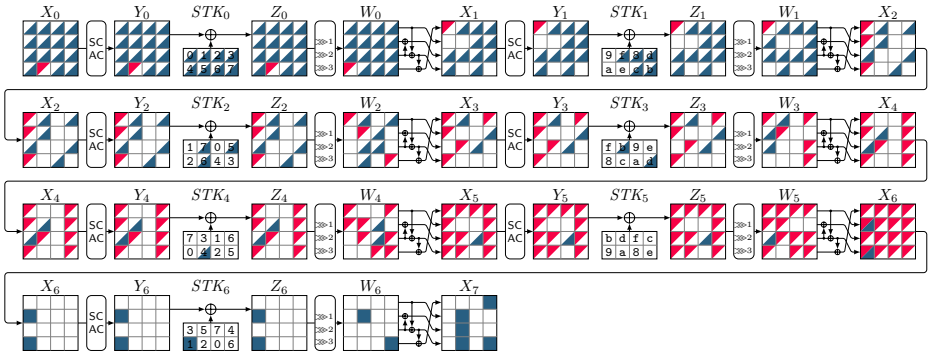
6 Rounds, Figure 22a $r_u = 0, r_m = 6, r_\ell = 0,\ p = 1,\ r = 1, q^2 = 1,\ prq^2 = 1$			
ΔX_0	0000000000008000	ΓX_6	2000022902202229
7 Rounds, Figure 22b $r_u = 0, r_m = 6, r_\ell = 1,\ p = 1,\ r = 1, q^2 = 2^{-4},\ prq^2 = 2^{-4},\ \hat{\mathbb{C}} = 2^{-2.32}$			
ΔX_0	0000000000000100		
ΓX_6	0000a0000000a000	ΓX_7	0004040004000404
8 Rounds, Figure 22c $r_u = 0, r_m = 7, r_\ell = 1,\ p = 1,\ r = 2^{-3.87}, q^2 = 2^{-4},\ prq^2 = 2^{-7.87},\ \hat{\mathbb{C}} = 2^{-6.75}$			
ΔX_0	0000000000000040		
ΓX_7	9000000000009000	ΓX_8	0008c00000000008
9 Rounds, Figure 23a $r_u = 1, r_m = 7, r_\ell = 1,\ p = 2^{-6},\ r = 2^{-3.94}, q^2 = 2^{-4},\ prq^2 = 2^{-13.94},\ \hat{\mathbb{C}} = 2^{-10.86}$			
ΔX_0	0000010010000001	ΔX_1	0000000000000090
ΓX_8	9000000000009000	ΓX_9	0008c00000000008
10 Rounds, Figure 23b $r_u = 1, r_m = 8, r_\ell = 1,\ p = 2^{-6},\ r = 2^{-7.72}, q^2 = 2^{-6},\ prq^2 = 2^{-19.72}$			
ΔX_0	0000010010000001	ΔX_1	00000000000000b0
ΓX_9	000000b000b000b0	ΓX_{10}	0900900900099909
11 Rounds, Figure 23c $r_u = 1, r_m = 8, r_\ell = 2,\ p = 2^{-6},\ r = 2^{-10.36}, q^2 = 2^{-8.87},\ prq^2 = 2^{-26.36}$			
ΔX_0	0000040010000004	ΔX_1	00000000000000b0
ΓX_{10}	0100000000000100	ΓX_{11}	000bb0b000b000bb

Table 9: DL distinguishers for SKINNY-64-128 (Related-Tweakey)

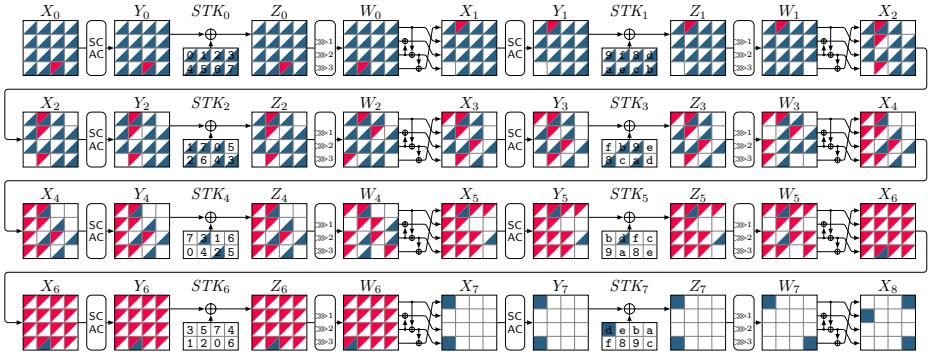
8 Rounds, Figure 24 $r_u = 0, r_m = 8, r_\ell = 0,\ p = 1,\ r = 1, q^2 = 1,\ prq^2 = \mathbf{1}$			
	ΔTK 00000000004000000000000020000		
ΔX_0	0000000000000000	ΓX_8	0010170010001710
14 Rounds, Figure 25a $r_u = 2, r_m = 10, r_\ell = 2,\ p = 2^{-4},\ r = 2^{-9.03}, q^2 = 2^{-10},\ prq^2 = \mathbf{2^{-23.03}}$			
	ΔTK 00000000400000000000000010000000		
ΔX_0	000000000000000c	ΔX_2	0000000000000000
ΓX_{12}	a00000000000a000	ΓX_{14}	0020020f02000220
15 Rounds, Figure 25b $r_u = 3, r_m = 10, r_\ell = 2,\ p = 2^{-10.42},\ r = 2^{-8.30}, q^2 = 2^{-10},\ prq^2 = \mathbf{2^{-28.72}}$			
	ΔTK c000000000000000f000000000000000		
ΔX_0	200000100d00d000	ΔX_3	0000000000000000
ΓX_{13}	a00000000000a000	ΓX_{15}	00400f040f000f40



(a) DL distinguisher for 6 rounds of SKINNY

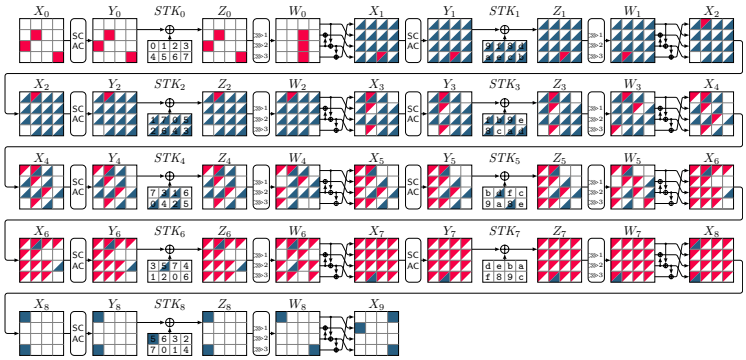


(b) DL distinguisher for 7 rounds of SKINNY

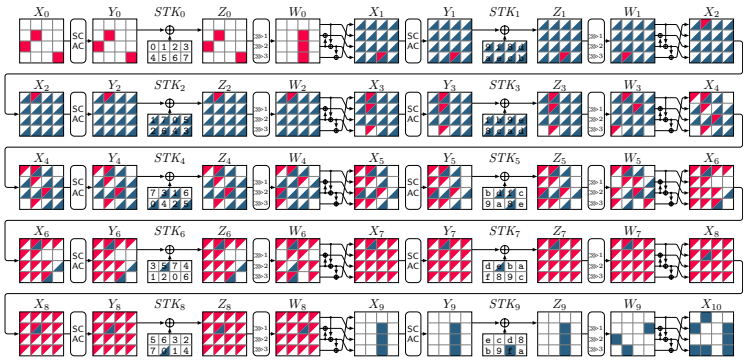


(c) DL distinguisher for 8 rounds of SKINNY

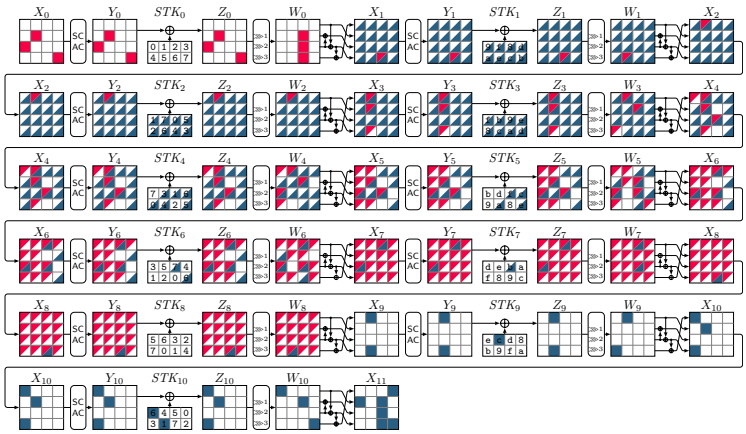
Fig. 22: DL distinguishers for 6 to 8 rounds of SKINNY-64 (Single-Tweakey) (difference linear mask)



(a) DL distinguisher for 9 rounds of SKINNY



(b) DL distinguisher for 10 rounds of SKINNY



(c) DL distinguisher for 11 rounds of SKINNY

Fig. 23: DL distinguishers for 9 to 11 rounds of SKINNY-64 (Single-Tweakey) (difference linear mask)

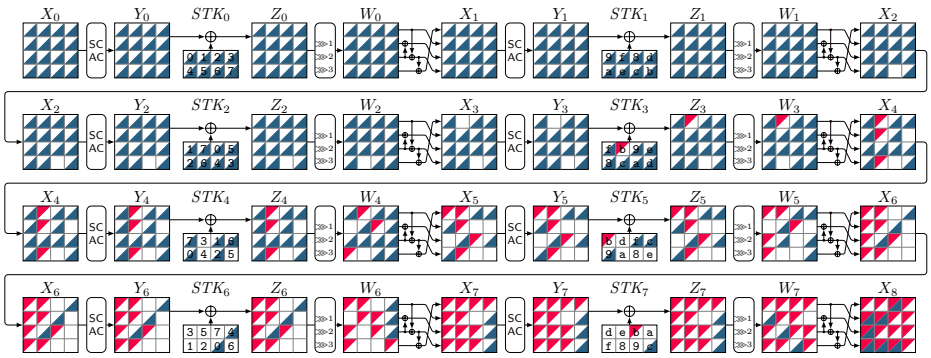
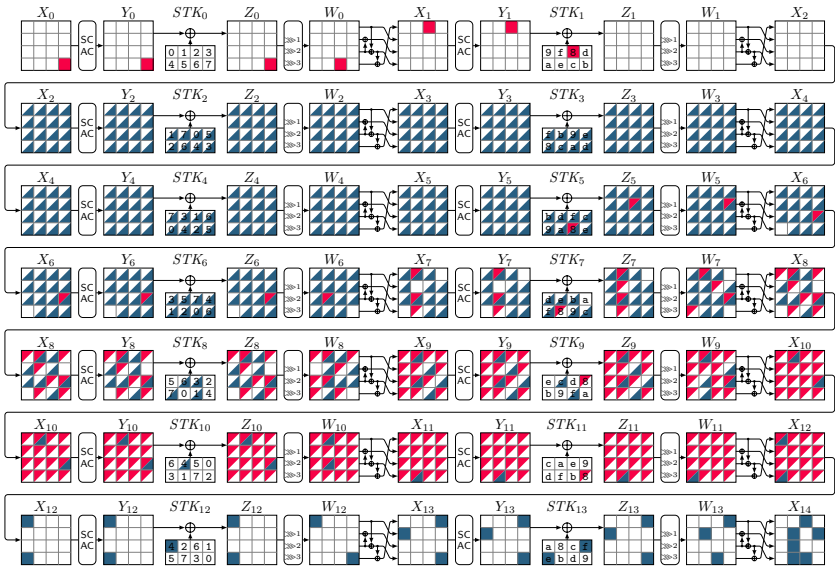


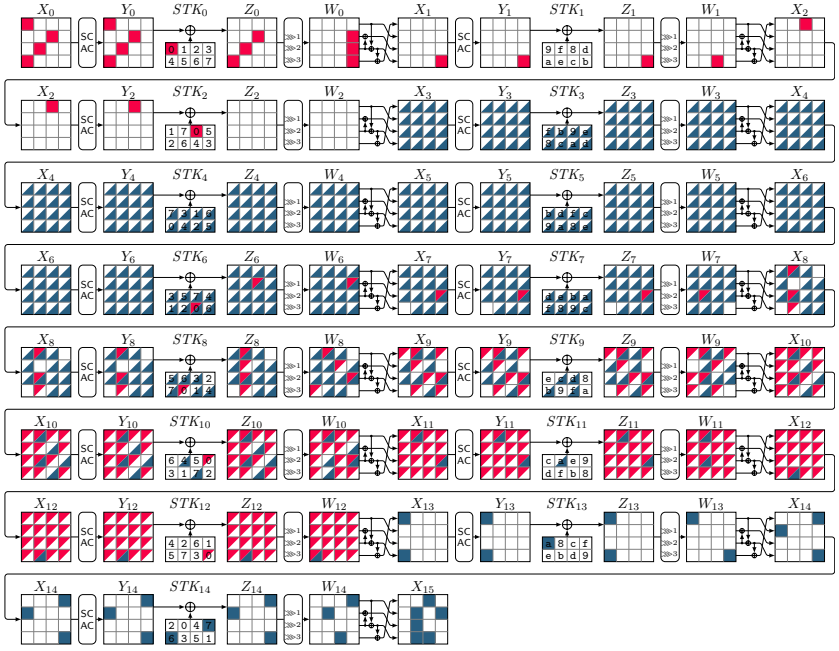
Fig. 24: DL distinguisher for 8 rounds of SKINNY-64-128 (Related-Tweakey) (difference linear mask)

Table 10: DL distinguishers for SKINNY-64-192 (Related-Tweakey)

10 Rounds, Figure 26a $r_u = 0, r_m = 10, r_\ell = 0,\ p = 1,\ r = 1, q^2 = 1,\ prq^2 = \mathbf{1}$			
ΔTK	00000000000008000000000000000b000000000000000e00		
ΔX_0	0000000000000000	ΓX_8	0010130010001310
16 Rounds, Figure 25a $r_u = 7, r_m = 7, r_\ell = 2,\ p = 2^{-2.42},\ r = 2^{-8.15}, q^2 = 2^{-10},\ prq^2 = \mathbf{2^{-20.57}}$			
ΔTK	00000000010000000000000b0000000000000008000000		
ΔX_0	0000000000000200	ΔX_7	0000000000000000
ΓX_{14}	3000000000003000	ΓX_{16}	00c00c0c0c000cc0
17 Rounds, Figure 25b $r_u = 8, r_m = 7, r_\ell = 2,\ p = 2^{-9.09},\ r = 2^{-8.50}, q^2 = 2^{-10},\ prq^2 = \mathbf{2^{-27.59}}$			
ΔTK	02000000000000000700000000000000010000000000000		
ΔX_0	0900200000020020	ΔX_8	0000000000000000
ΓX_{15}	3000000000003000	ΓX_{17}	00c00c0c0c000cc0

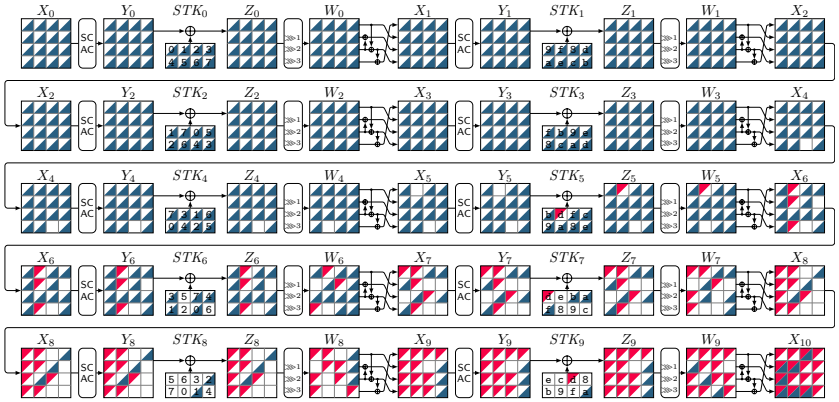


(a) DL distinguisher for 14 rounds of SKINNY-64-128

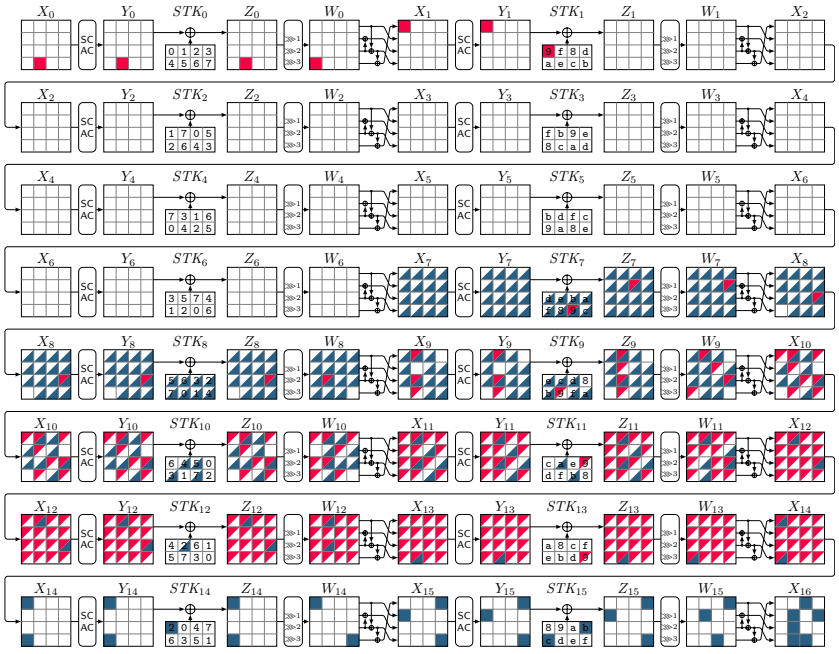


(b) DL distinguisher for 15 rounds of SKINNY-64-128

Fig. 25: DL distinguishers for 14 to 15 rounds of SKINNY-64-128 (Related-Tweakey) (◩ difference ◪ linear mask)



(a) DL distinguisher for 10 rounds of SKINNY-64-192 (Related-Tweakey) (difference linear mask)



(b) DL distinguisher for 16 rounds of SKINNY-64-192

Fig. 26: DL distinguishers for 10 and 16 rounds of SKINNY-64-192 (Related-Tweakey) (difference linear mask)

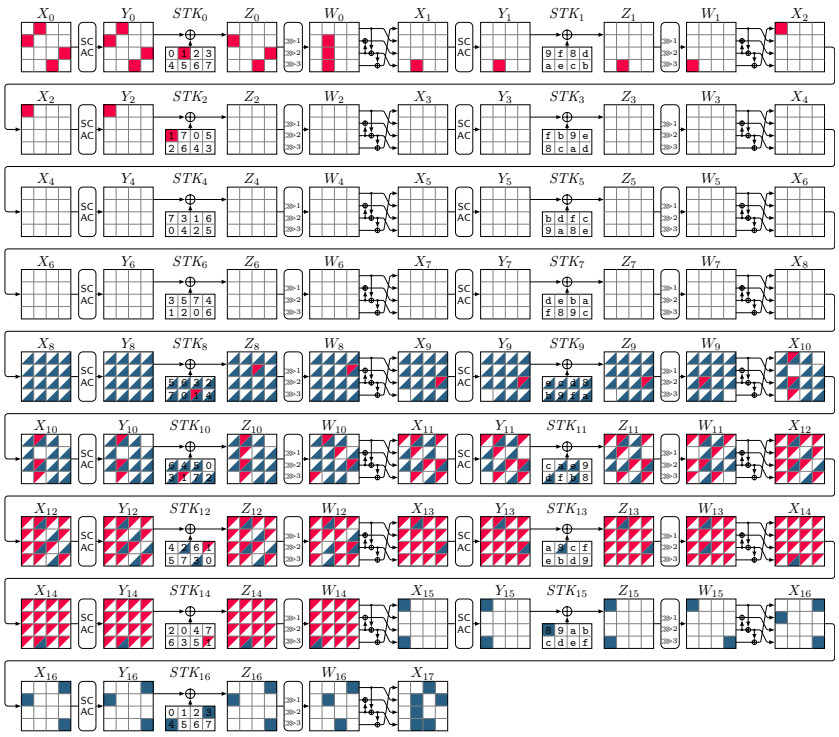


Fig. 27: DL distinguishers for 17 rounds of SKINNY-64-192 (Related-Tweakey) (difference linear mask)

G Application to PRESENT

G.1 Brief Specification of PRESENT

PRESENT is a 64-bit block cipher supporting 80-bit and 128-bit keys designed by Bogdanov et al. in 2007 [14]. The design is a minimalistic SPN construction consisting of a round key addition, a 4-bit S-box layer, and a bit permutation layer. The S-box is specified in Table 11. The bit permutation and the entire round function are both illustrated in Figure 28. The full diffusion of PRESENT is achieved after 4 rounds.

Table 11: S-box of PRESENT

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$\mathcal{S}(x)$	c	5	6	b	9	0	a	d	3	e	f	8	4	7	1	2

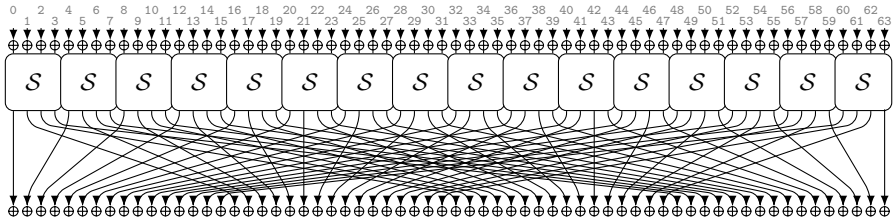


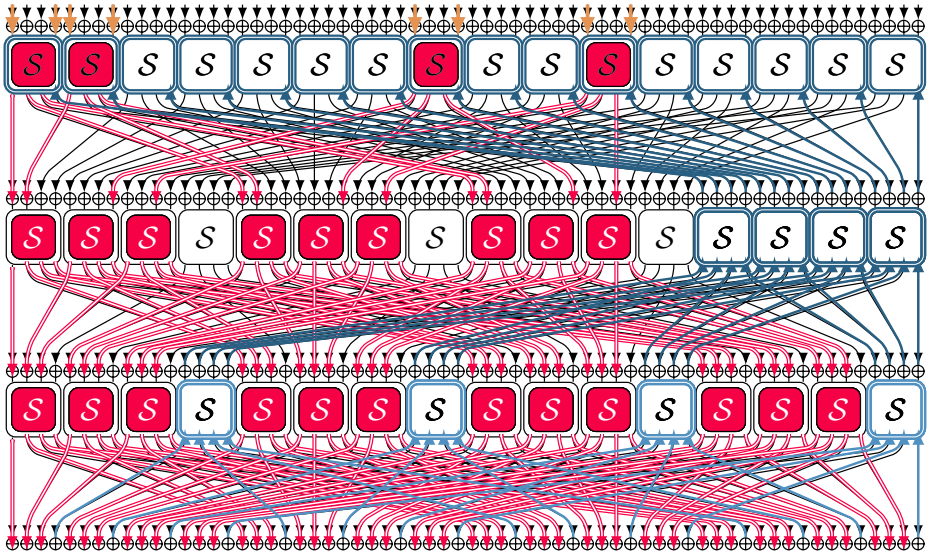
Fig. 28: Round function of PRESENT.

G.2 The DL Distinguishers of PRESENT

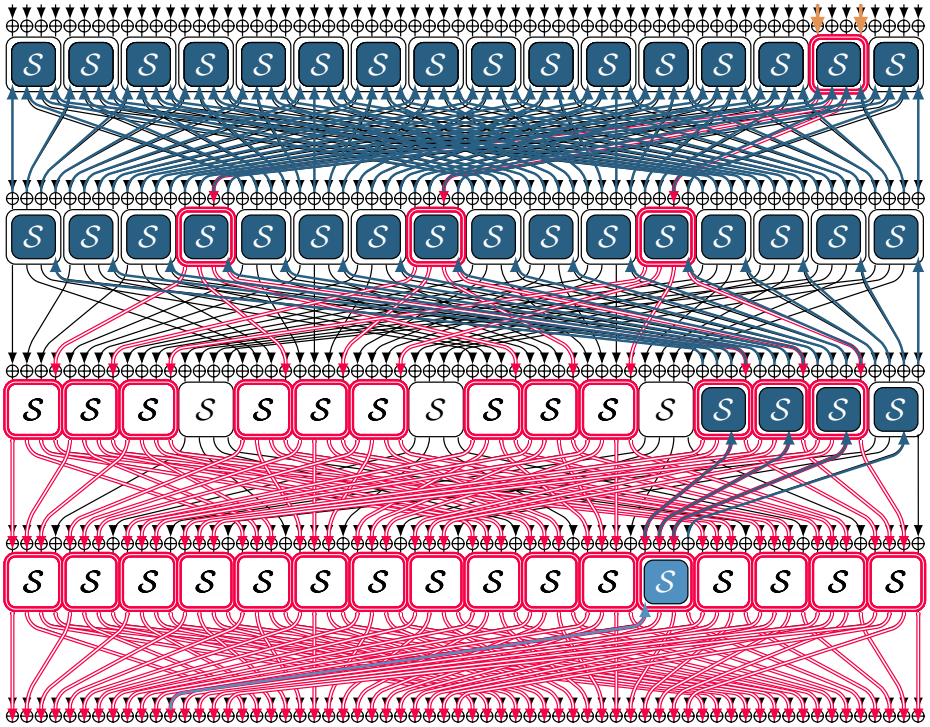
Table 12 briefly describes the DL distinguishers of PRESENT.

Table 12: Specification of DL distinguishers for PRESENT

3 Rounds, Figure 29a $r_u = 0, r_m = 3, r_\ell = 0,\ p = 1,\ r = 1, q^2 = 1,\ prq^2 = \mathbf{1}$			
ΔX_0	9900000900900000	ΓX_3	1111111111111111
4 Rounds, Figure 29b $r_u = 0, r_m = 4, r_\ell = 0,\ p = 1,\ r = 2^{-0.82}, q^2 = 1,\ prq^2 = 2^{-0.82}$			
ΔX_0	0000000000000090	ΓX_4	0010000000000000
5 Rounds, Figure 30 $r_u = 0, r_m = 5, r_\ell = 0,\ p = 1,\ r = 2^{-1.19}, q^2 = 1,\ prq^2 = 2^{-1.19}$			
ΔX_0	0000000000090000	ΓX_5	0010000000100010
6 Rounds, Figure 31 $r_u = 0, r_m = 6, r_\ell = 0,\ p = 1,\ r = 2^{-2.85}, q^2 = 1,\ prq^2 = 2^{-2.85}$			
ΔX_0	0000000000000009	ΓX_6	0800000008000800
7 Rounds, Figure 32 $r_u = 0, r_m = 7, r_\ell = 0,\ p = 1,\ r = 2^{-5.32}, q^2 = 1,\ prq^2 = 2^{-5.32}$			
ΔX_0	0000009000000000	ΓX_7	0800000008000800
8 Rounds, Figure 33 $r_u = 0, r_m = 8, r_\ell = 0,\ p = 1,\ r = 2^{-6.43}, q^2 = 1,\ prq^2 = 2^{-6.43}$			
ΔX_0	0000000000000090	ΓX_8	0080000000800080
9 Rounds, Figure 34 $r_u = 0, r_m = 9, r_\ell = 0,\ p = 1,\ r = 2^{-9.23}, q^2 = 1,\ prq^2 = 2^{-9.23}$			
ΔX_0	0000009000000000	ΓX_9	0020000000200020
10 Rounds, Figure 35 $r_u = 1, r_m = 8, r_\ell = 1,\ p = 2^{-4},\ r = 2^{-7.97}, q^2 = 2^{-2},\ prq^2 = 2^{-13.97}$			
ΔX_0	0000000000007007	ΔX_1	0000000000000009
ΓX_9	0000000000800000	ΓX_{10}	0020002000200000
11 Rounds, Figure 36 $r_u = 2, r_m = 9, r_\ell = 0,\ p = 2^{-6},\ r = 2^{-11.36}, q^2 = 1,\ prq^2 = 2^{-17.36}$			
ΔX_0	0000000090090000	ΔX_2	0000020000000000
ΓX_{11}	0800000008000800	-	-
12 Rounds, Figure 37 $r_u = 2, r_m = 9, r_\ell = 1,\ p = 2^{-6},\ r = 2^{-11.77}, q^2 = 2^{-6},\ prq^2 = 2^{-23.77}$			
ΔX_0	00000000700f0000	ΔX_2	0000000200000000
ΓX_{11}	0800000008000800	ΓX_{12}	4044404440444044
13 Rounds, Figure 38 $r_u = 1, r_m = 8, r_\ell = 4,\ p = 2^{-4},\ r = 2^{-9.01}, q^2 = 2^{-14},\ prq^2 = 2^{-27.01}$			
ΔX_0	9009000000000000	ΔX_1	0000900000000000
ΓX_9	0080000000000000	ΓX_{13}	4000400000004000



(a) DL distinguisher for 3 rounds of PRESENT



(b) DL distinguisher for 4 rounds of PRESENT

Fig. 29: DL distinguishers for 3-4 rounds of PRESENT (– differential – linear)

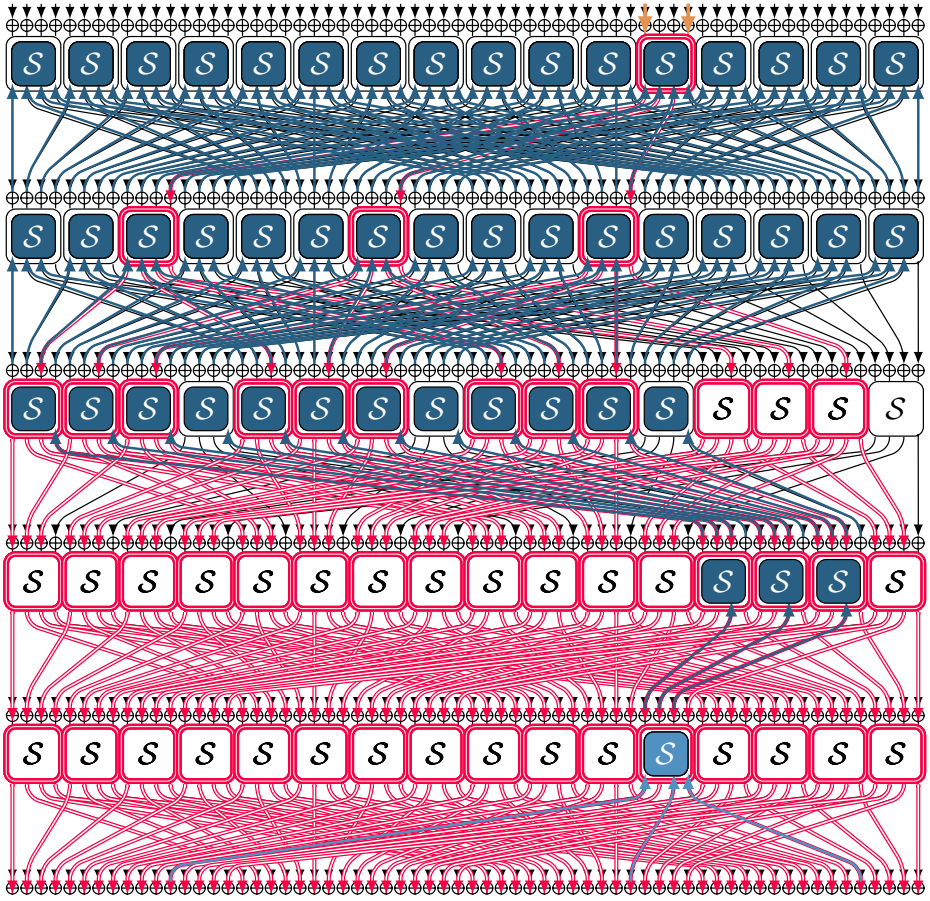


Fig. 30: DL distinguisher for 5 rounds of PRESENT

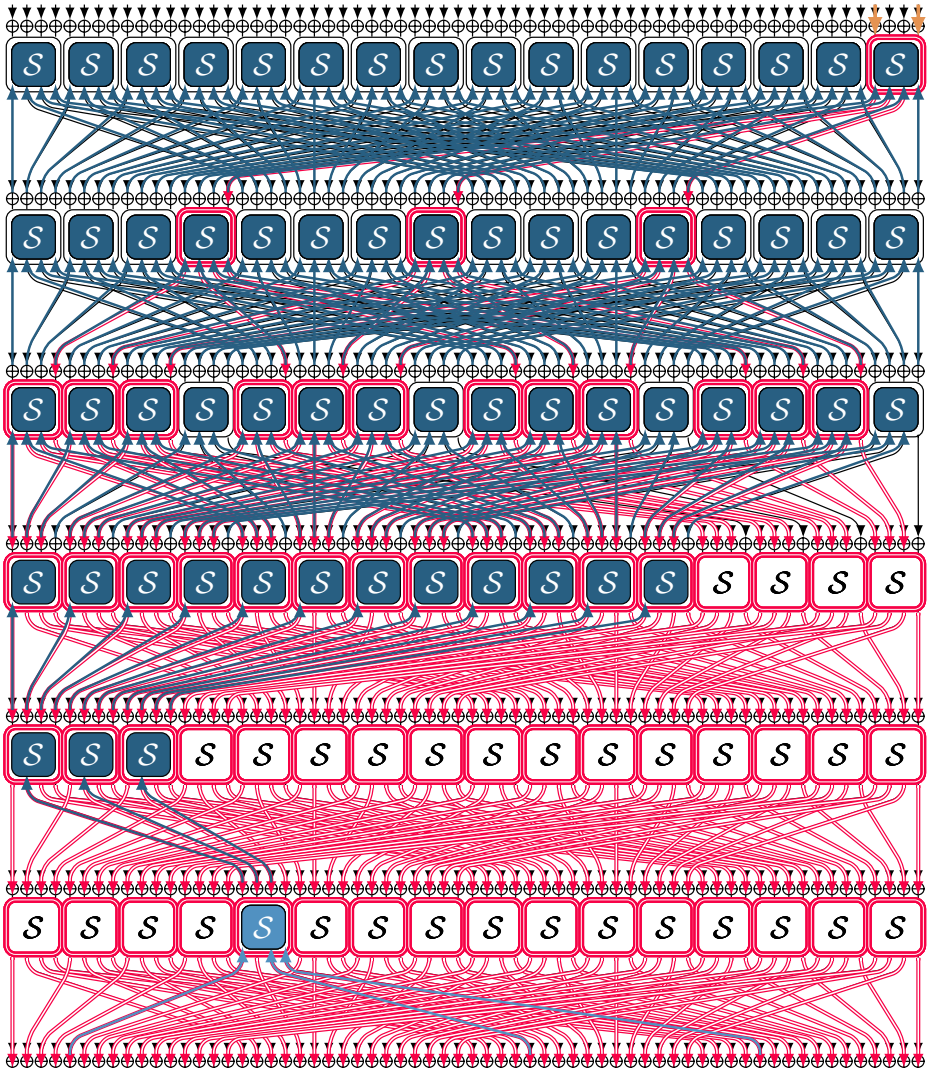


Fig. 31: DL distinguisher for 6 rounds of PRESENT

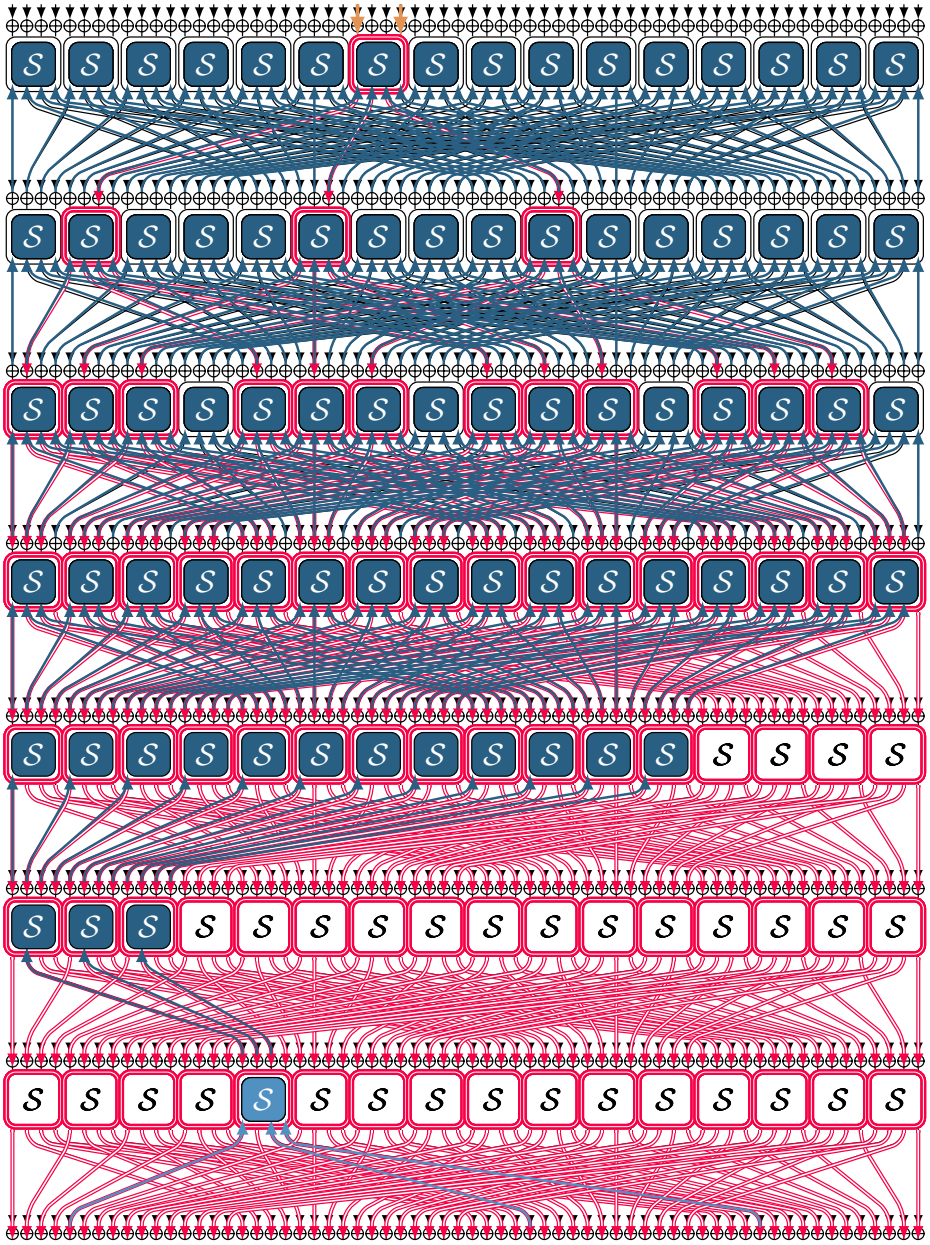


Fig. 32: DL distinguisher for 7 rounds of PRESENT

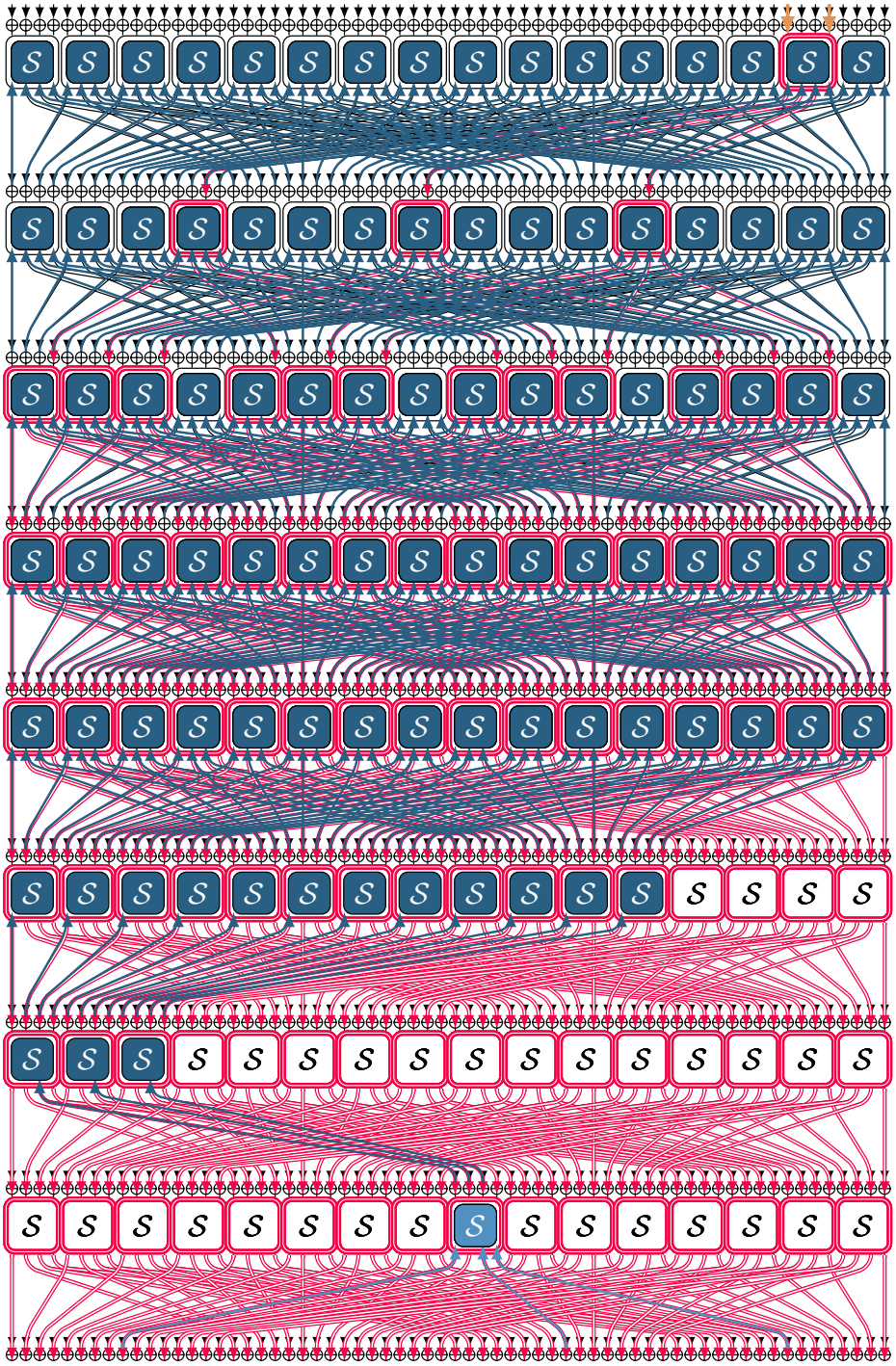


Fig. 33: DL distinguisher for 8 rounds of PRESENT (– differential – linear)

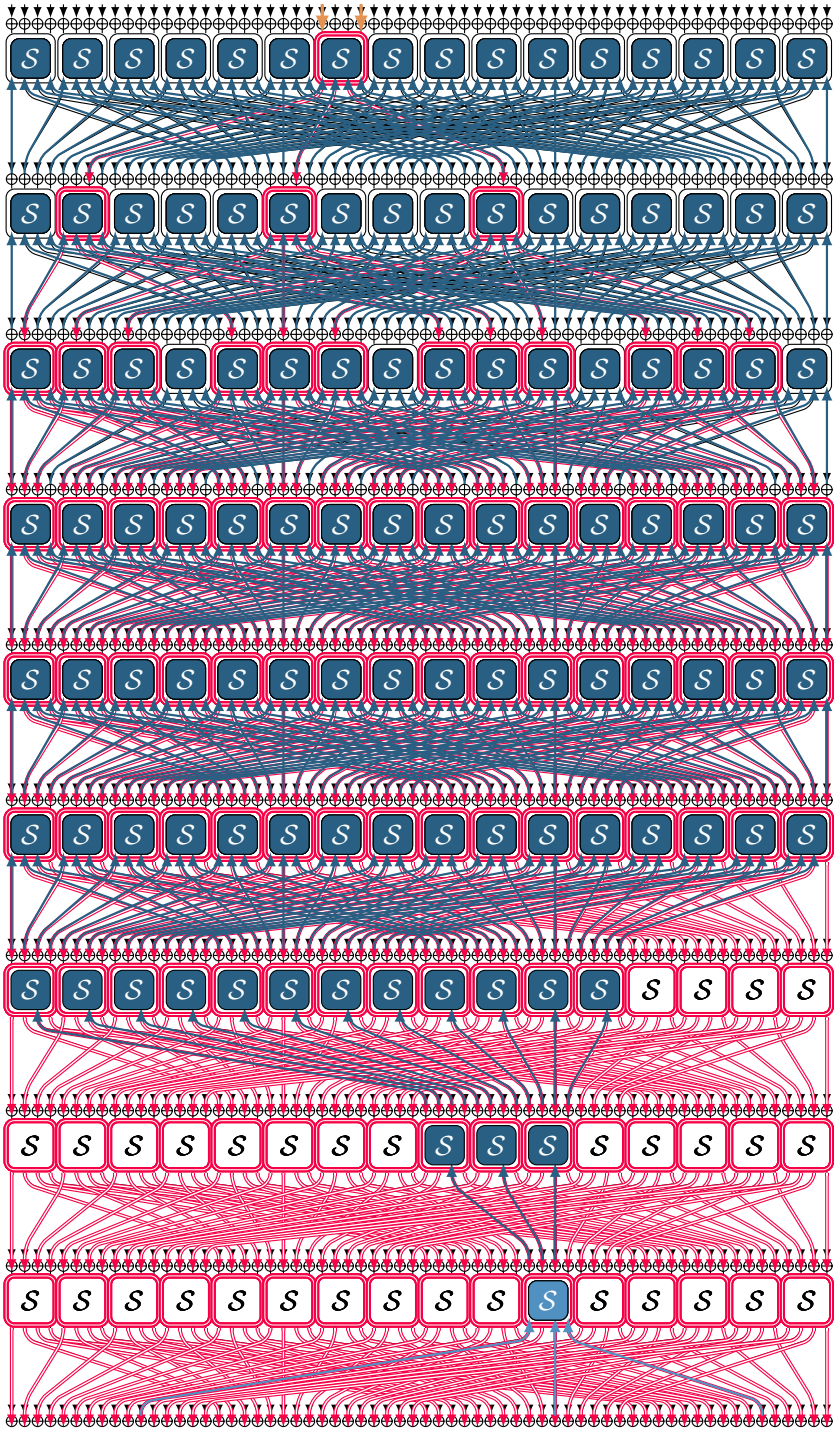


Fig. 34: DL distinguisher for 9 rounds of PRESENT (– differential – linear)

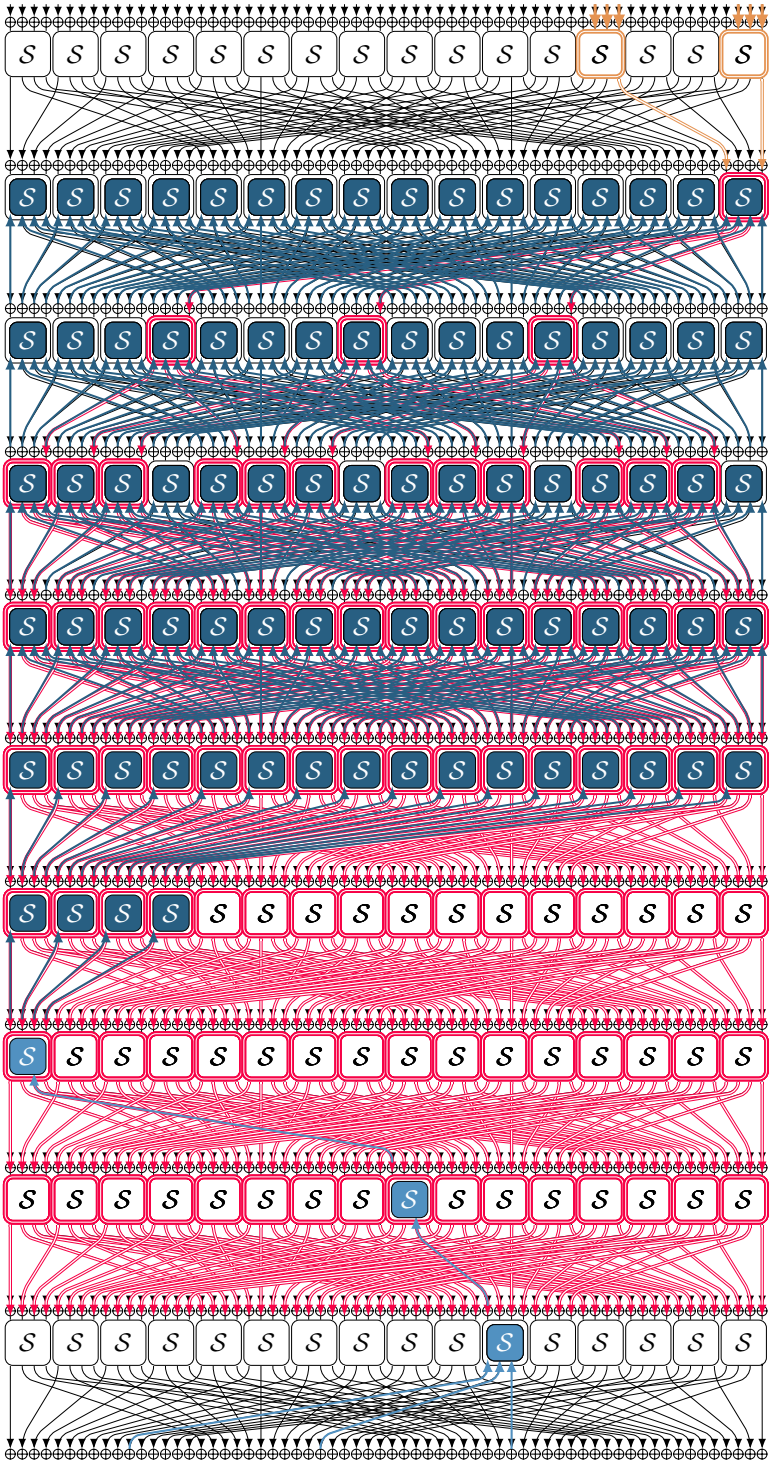


Fig. 35: DL distinguisher for 10 rounds of PRESENT (– differential – linear)

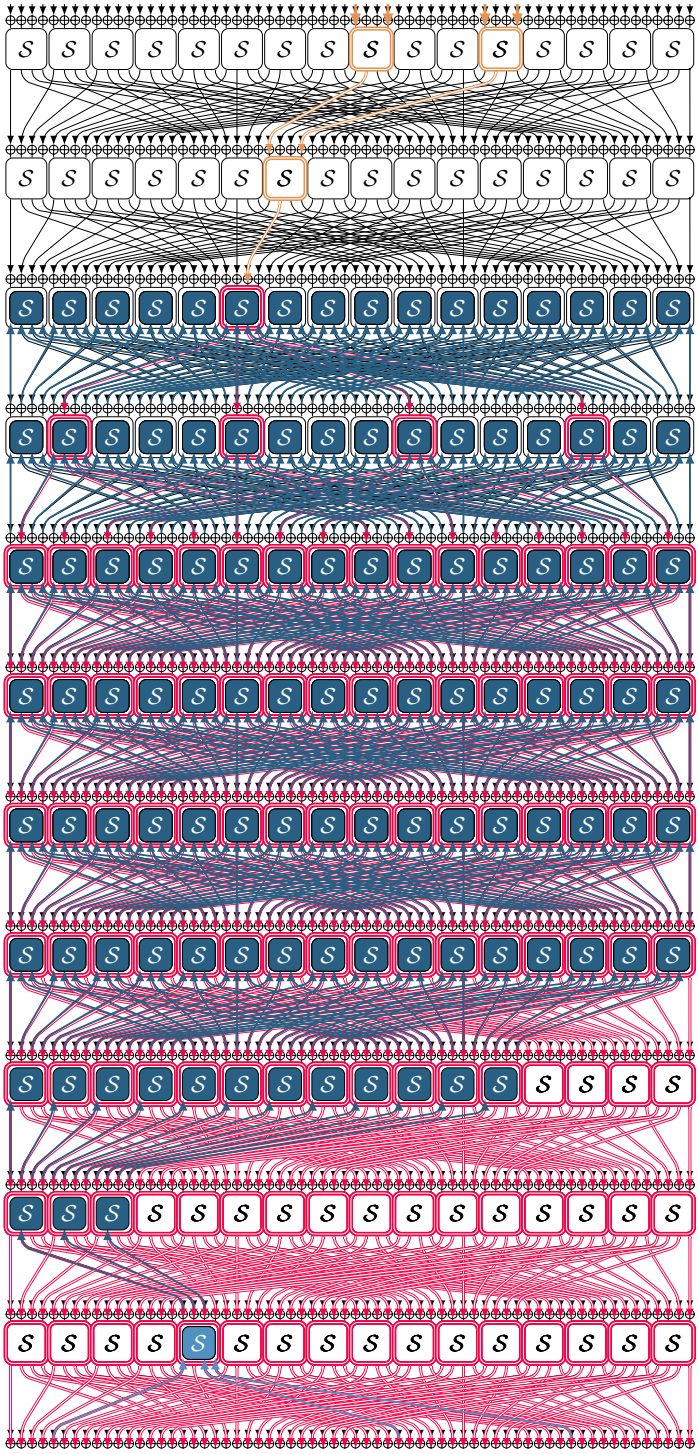


Fig. 36: DL distinguisher for 11 rounds of PRESENT (– differential – linear)

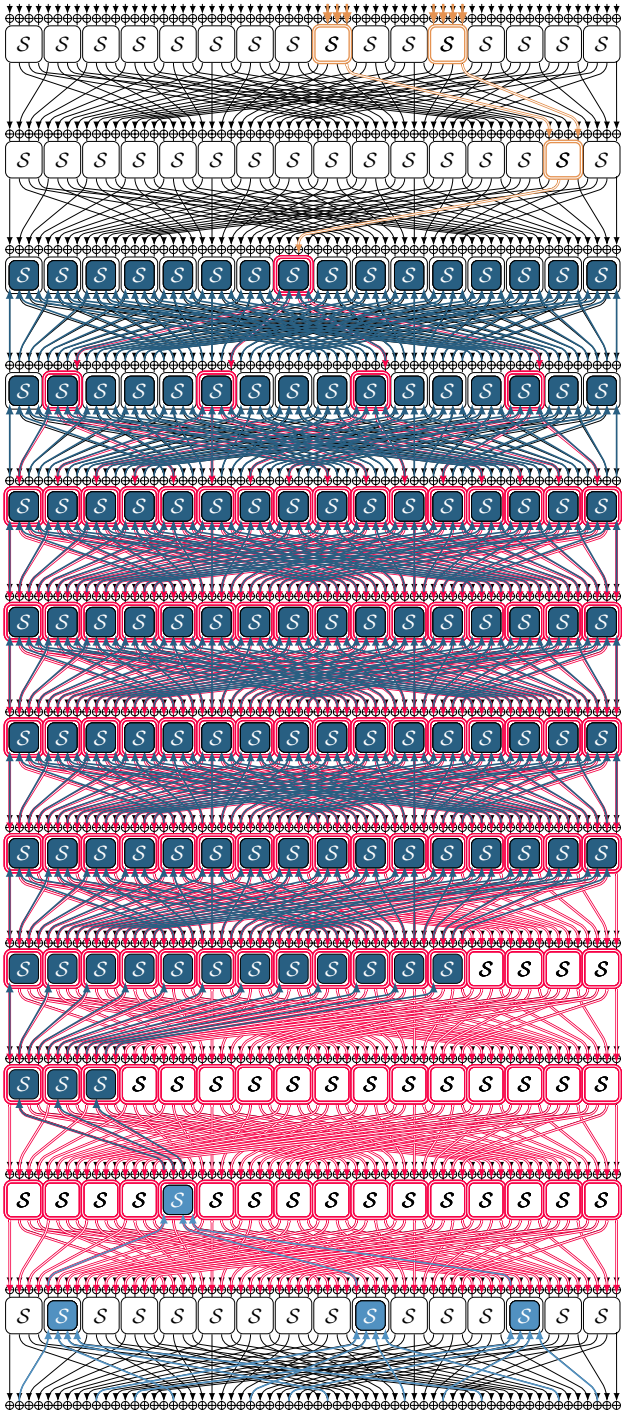


Fig. 37: DL distinguisher for 12 rounds of PRESENT (– differential – linear)

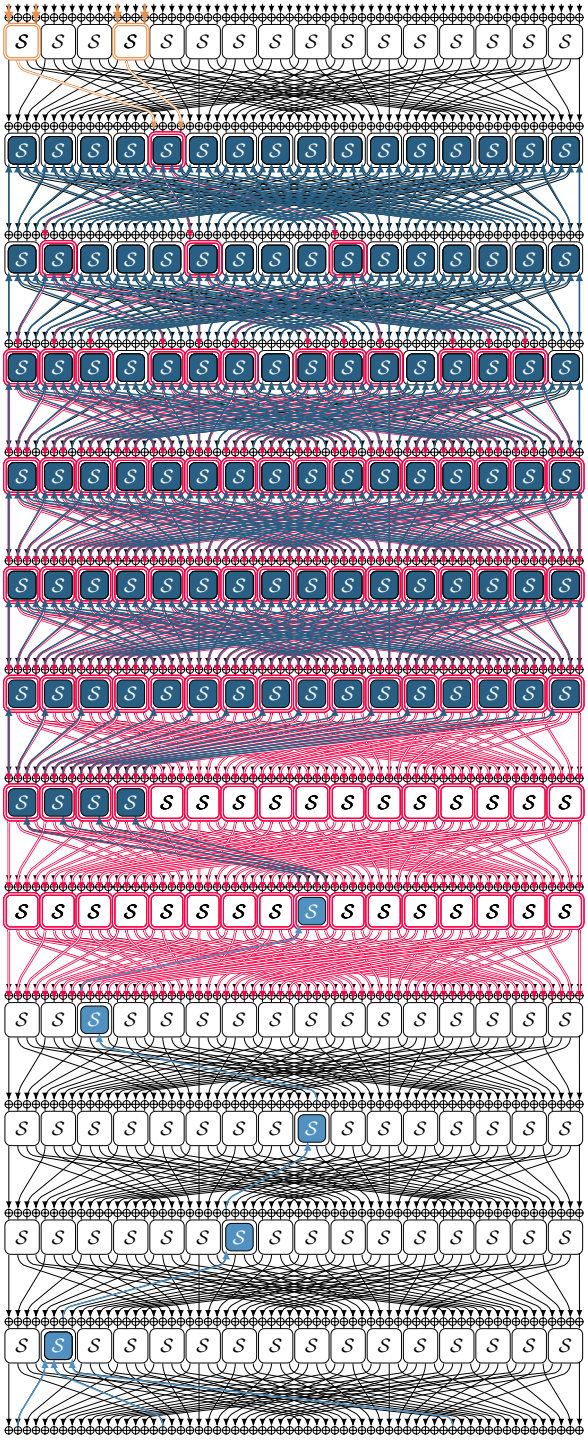


Fig. 38: DL distinguisher for 13 rounds of PRESENT (– differential – linear)

H Application to KNOT

H.1 Brief Specification of KNOT

KNOT was a round-2 candidate in the NIST Lightweight Cryptography project designed by Zhang et al. [59]. Its core primitive is the KNOT permutation, available in three variants with 256-bit, 384-bit, or 512-bit state size, respectively. The state is organized in 4 words of 64, 96, or 128 bits. We focus on the 256-bit variant with 64-bit words used in the primary recommendation for LWC in the following. Each of the variants uses a similar round function consisting of a round constant addition, a bitsliced S-box layer (SubColumn), and a word-wise rotation (ShiftRow). The number of rounds differs between the phases of the authenticated encryption scheme and is 52 (initialization), 28 (data), or 32 (finalization). The S-box is specified in Table 13, and its differential and linear properties are given in Table 14 and Table 15, respectively. In the rotation layer of the 256-bit variant, the words a_0, a_1, a_2, a_3 are rotated left as follows:

$$\begin{aligned} a_0 &\leftarrow a_0 \lll 0 \\ a_1 &\leftarrow a_1 \lll 1 \\ a_2 &\leftarrow a_2 \lll 8 \\ a_3 &\leftarrow a_3 \lll 25 \end{aligned}$$

Table 13: S-box of KNOT

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$\mathcal{S}(x)$	4	0	a	7	b	e	1	d	9	f	6	8	5	2	c	3

H.2 The DL Distinguishers of KNOT

Table 14: DDT of KNOT's S-box

$\Delta_i \setminus \Delta_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	2	2	2	2	0	0	0	0	2	2	2	2
2	0	2	0	2	0	0	0	4	0	2	2	0	0	0	2	2
3	0	2	0	2	0	0	4	0	0	2	2	0	0	0	2	2
4	0	0	0	0	0	0	0	0	0	0	4	4	2	2	2	2
5	0	0	0	0	2	2	2	2	0	0	4	4	0	0	0	0
6	0	2	0	2	0	4	0	0	0	2	2	0	2	2	0	0
7	0	2	0	2	4	0	0	0	0	2	2	0	2	2	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	4	4	4	4
9	0	4	4	0	0	0	0	0	0	4	0	4	0	0	0	0
a	0	0	2	2	2	0	0	2	4	0	0	0	0	2	0	2
b	0	0	2	2	0	2	2	0	4	0	0	0	2	0	2	0
c	0	4	4	0	2	2	2	2	0	0	0	0	0	0	0	0
d	0	0	0	0	2	2	2	2	0	4	0	4	0	0	0	0
e	0	0	2	2	0	2	2	0	4	0	0	0	0	2	0	2
f	0	0	2	2	2	0	0	2	4	0	0	0	2	0	2	0

Table 15: LAT of KNOT's S-box (scale: $2^4 \cdot$ correlation)

$\lambda_i \setminus \lambda_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	4	-4	0	-8	-4	-4	0	0	4	-4	-8	0	4	4
2	0	0	0	0	0	0	0	0	0	8	8	0	0	8	-8	0
3	0	-8	4	4	0	0	-4	4	0	0	-4	4	-8	0	-4	-4
4	0	4	0	4	0	4	8	-4	0	4	0	4	-8	-4	0	4
5	0	4	-4	-8	0	-4	-4	0	0	4	-4	8	0	-4	-4	0
6	0	-4	8	4	0	-4	0	-4	0	4	0	4	8	-4	0	4
7	0	4	4	0	0	-4	4	-8	0	-4	-4	0	0	4	-4	-8
8	0	0	0	0	0	0	0	0	0	0	8	8	0	0	8	-8
9	0	0	-4	4	8	0	-4	-4	0	0	4	-4	0	-8	-4	-4
a	0	8	0	8	0	-8	0	8	0	0	0	0	0	0	0	0
b	0	0	-4	4	-8	0	-4	-4	0	8	-4	-4	0	0	4	-4
c	0	4	0	4	0	4	-8	-4	8	-4	0	4	0	4	0	4
d	0	4	4	0	-8	4	-4	0	-8	-4	4	0	0	-4	-4	0
e	0	4	8	-4	0	4	0	4	8	4	0	-4	0	-4	0	-4
f	0	-4	-4	0	-8	-4	4	0	8	-4	4	0	0	-4	-4	0

Table 16: Specification of DL distinguishers for KNOT-256 permutation

8 Rounds (I), Figure 39a
$r_u = 0, r_m = 8, r_\ell = 0,\ p = 1,\ r = 1,\ q^2 = 1,\ prq^2 = \mathbf{1}$

ΔX_0	ΓX_8
0000000000000000	0000000000000000
0000000000000000	0000000000000000
0000000010000000	0000000000080000
0000000010000000	0000001000000000

8 Rounds (II), Figure 39b
$r_u = 0, r_m = 8, r_\ell = 0,\ p = 1,\ r = 2^{-1},\ q^2 = 1,\ prq^2 = \mathbf{2^{-1}}$

ΔX_0	ΓX_8
0000000000000000	0000800000000000
0000000000000001	0000000000000000
0000000000000000	0000000000000000
0000000000000000	0000000000000000

15 Rounds (I), Figure 40a
$r_u = 4, r_m = 9, r_\ell = 2,\ p = 2^{-10},\ r = 2^{-3.20},\ q^2 = 2^{-4},\ prq^2 = 2^{-17.20}$

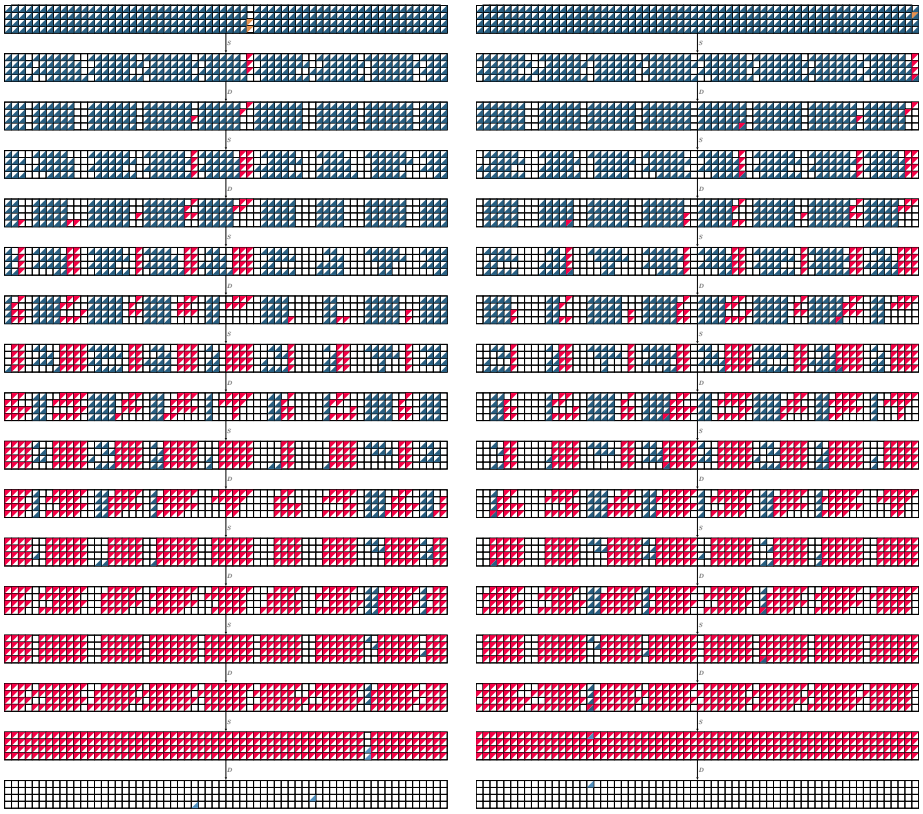
ΔX_0	ΔX_4	ΓX_{13}	ΓX_{15}
0000000000000000	0000000000000000	0000000000000000	0000000000000000
0000000000000000	0008000000000000	0000100000000000	0000000000000000
0000020000000000	0000000000000000	0000000000000000	0010000000000000
0000020000000000	0000000000000800	0000100000000000	0000000000000020

15 Rounds (II), Figure 40b
$r_u = 4, r_m = 9, r_\ell = 2,\ p = 2^{-13},\ r = 2^{-6.15},\ q^2 = 2^{-6},\ prq^2 = 2^{-25.15}$

ΔX_0	ΔX_4	ΓX_{13}	ΓX_{15}
0000000000000000	0000000000000000	0000000200000000	0000020000000000
0000000000800000	0000010000000000	0000000000000000	0000000000000000
0000000000000000	0000000000000001	0000000000000000	0000000000000000
0000000000000000	0000000000020001	0000000200000000	0000000000000000

23 Rounds , Figure 41
$r_u = 6, r_m = 9, r_\ell = 8,\ p = 2^{-21},\ r = 2^{-3.77},\ q^2 = 2^{-34},\ prq^2 = 2^{-58.88}$

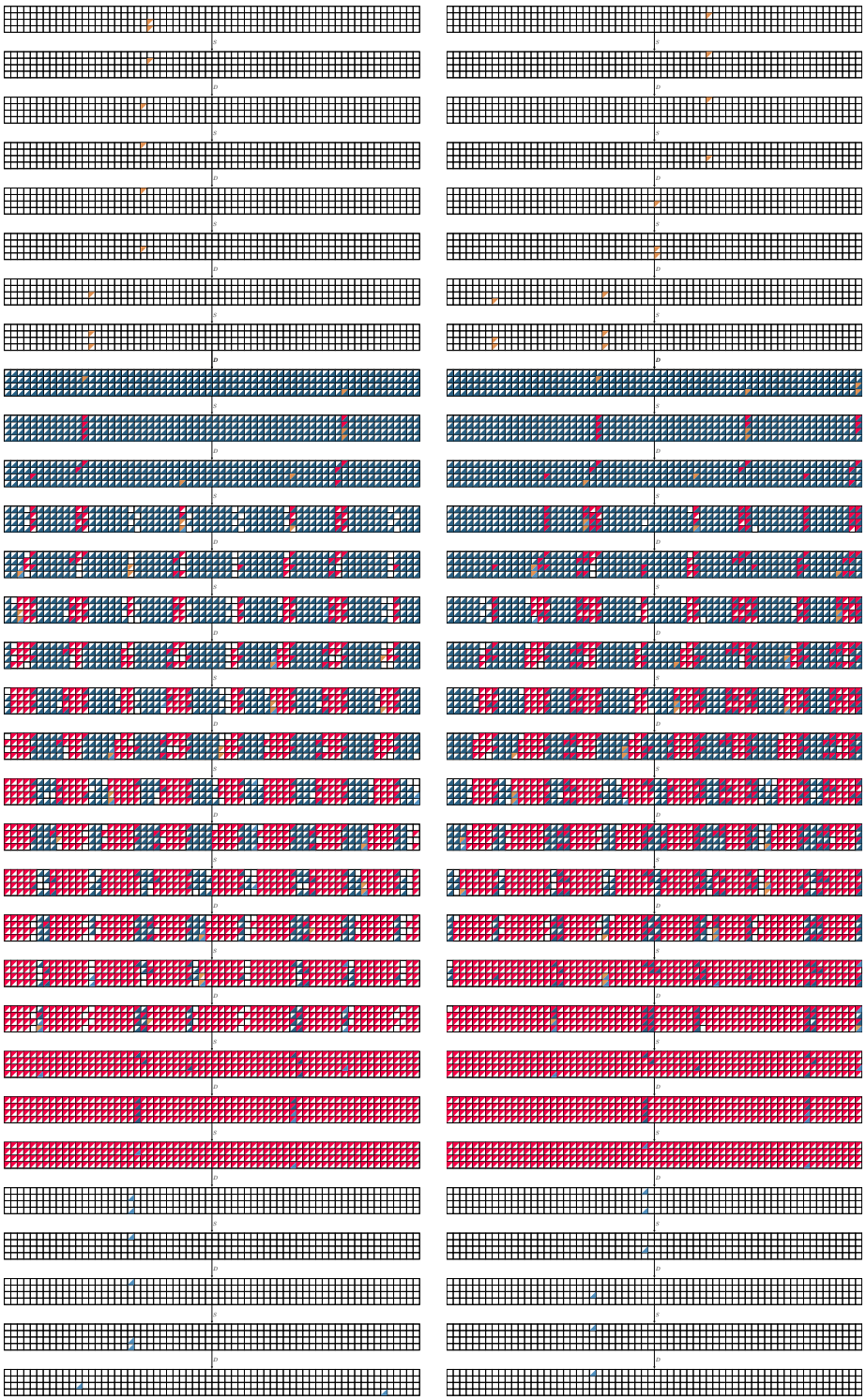
ΔX_0	ΔX_6	ΓX_{15}	ΓX_{23}
0000000000000000	0000000000000000	0000002000000000	0000010000000000
0000000000000002	2000000000000000	0000004000000000	0000000000000002
0004000002000000	0000000000000000	0000002000000000	0001000000000000
0004000002000002	0000000000200000	0000004000000000	0000000002000002



(a) 8-round KNOT (I) (b) 8-round KNOT (II)

active difference unknown difference active mask unknown mask

Fig. 39: DL distinguishers for 8 rounds of KNOT.



(a) 15-round (I) KNOT (b) 15-round (II) KNOT

active difference unknown difference active mask unknown mask

Fig. 40: DL distinguishers for 15 rounds of KNOT.

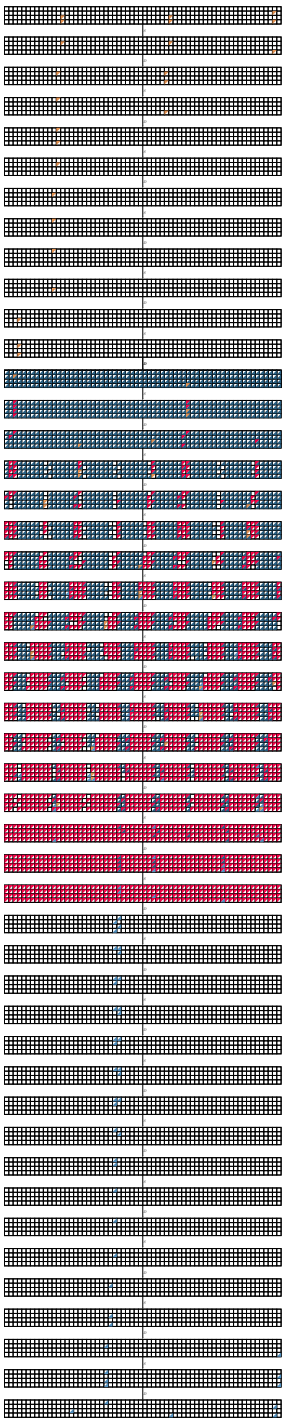


Fig. 41: DL distinguishers for 23 rounds of KNOT (active difference unknown difference active mask unknown mask).

I Application to Simeck

I.1 Brief Specification of Simeck

Simeck is a family of lightweight block ciphers designed introduced in CHES 2015 [58]. The design is inspired by Simon and Speck [4], combining an AndRX round function similar to Simon with a nonlinear key schedule as in Speck. The Simeck family consists of several family members Simeck$2n/4n$ operating on n-bit words with a state size of $2n$ bits and a key size of $4n$ bits for $n \in \{16, 24, 32\}$. In round i, the $2n$-bit input state of round i is split into two n-bit words (L_i, R_i) and updated with a Feistel-based round function F to produce (L_{i+1}, R_{i+1}) using the n-bit round key K_i. The round function is a quadratic Feistel function using bitwise XOR ($x \oplus y$), bitwise AND ($x \odot y$), and cyclic left-shifts by c bits ($x \lll c$):

$$\begin{aligned} R_{i+1} &= L_i \\ L_{i+1} &= R_i \oplus K_i \oplus (L_i \odot (L_i \lll 5)) \oplus (L_i \lll 1), \end{aligned}$$

as illustrated in Figure 42. The round key K_i is produced using a similar nonlinear update function. The total number of rounds is 32 rounds for Simeck32-64 (also referred to as Simeck-32 for short), 36 rounds for Simeck48-96 (aka Simeck-48), and 44 rounds for Simeck64-128 (aka Simeck-64). For a more detailed specification, we refer to the design paper [58].

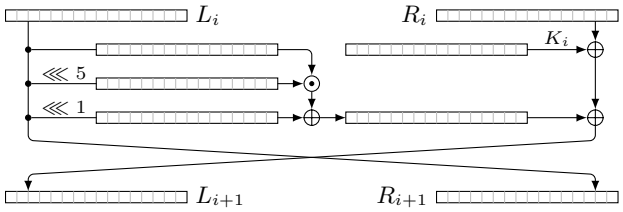


Fig. 42: Round function of Simeck-32.

I.2 The DL Distinguishers of Simeck

Table 17: Specification of DL distinguishers for Simeck-32

7 Rounds, Figure 43 $r_u = 0, r_m = 7, r_\ell = 0,\ p = 1,\ r = 1, q^2 = 1,\ prq^2 = 1$			
ΔX_0	00001000	ΓX_7	00000400
14 Rounds, Figure 44 $r_u = 1, r_m = 10, r_\ell = 3,\ p = 2^{-2},\ r = 2^{-7.92}, q^2 = 2^{-4},\ prq^2 = 2^{-13.92},\ \hat{\mathbb{C}} = 2^{-13.35}$			
ΔX_0	00020005	ΔX_1	00010002
ΓX_{11}	80000000	ΓX_{14}	4000a000

Table 18: Specification of DL distinguishers for Simeck-48

8 Rounds, Figure 45 $r_u = 0, r_m = 8, r_\ell = 0,\ p = 1,\ r = 1, q^2 = 1,\ prq^2 = 1$			
ΔX_0	000000020000	ΓX_6	000000010000
17 Rounds $r_u = 1, r_m = 14, r_\ell = 2,\ p = 2^{-2},\ r = 2^{-9.89}, q^2 = 2^{-2},\ prq^2 = 2^{-13.89},\ \hat{\mathbb{C}} = 2^{-13.25}$			
ΔX_0	000010000020	ΔX_1	000000000010
ΓX_{15}	000010000000	ΓX_{17}	000010000008
18 Rounds, Figure 46 $r_u = 2, r_m = 14, r_\ell = 2,\ p = 2^{-4},\ r = 2^{-9.89}, q^2 = 2^{-2},\ prq^2 = 2^{-15.89}$			
ΔX_0	000020000050	ΔX_2	000000000010
ΓX_{16}	000010000000	ΓX_{18}	000010000008
19 Rounds $r_u = 2, r_m = 14, r_\ell = 3,\ p = 2^{-4},\ r = 2^{-9.89}, q^2 = 2^{-4},\ prq^2 = 2^{-17.89}$			
ΔX_0	000020000050	ΔX_2	000000000010
ΓX_{16}	000010000000	ΓX_{19}	000008000014
20 Rounds $r_u = 3, r_m = 14, r_\ell = 3,\ p = 2^{-8},\ r = 2^{-9.89}, q^2 = 2^{-4},\ prq^2 = 2^{-21.89}$			
ΔX_0	000050000080	ΔX_3	000000000010
ΓX_{17}	000010000000	ΓX_{20}	000008000014

Table 19: Specification of DL distinguishers for Simeck-64

10 Rounds, Figure 47 $r_u = 0, r_m = 10, r_\ell = 0,\ p = 1,\ r = 1, q^2 = 1,\ prq^2 = 1$			
ΔX_0	0000000000000010	ΓX_6	0000000000000002
24 Rounds $r_u = 3, r_m = 17, r_\ell = 4,\ p = 2^{-4},\ r = 2^{-13.14}, q^2 = 2^{-8},\ prq^2 = 2^{-25.14}$			
ΔX_0	00000040000008e0	ΔX_3	0000002000000000
ΓX_{20}	0000001a00000004	ΓX_{24}	0000000000000010
25 Rounds, Figure 48 $r_u = 3, r_m = 17, r_\ell = 5,\ p = 2^{-4},\ r = 2^{-13.14}, q^2 = 2^{-10},\ prq^2 = 2^{-27.14}$			
ΔX_0	00000040000008e0	ΔX_3	0000002000000000
ΓX_{20}	0000001a00000004	ΓX_{25}	0000001080000018
26 Rounds, Figure 49 $r_u = 5, r_m = 16, r_\ell = 5,\ p = 2^{-9.30},\ r = 2^{-13.05}, q^2 = 2^{-8},\ prq^2 = 2^{-30.35}$			
ΔX_0	0000001000000020	ΔX_5	0000008000000050
ΓX_{21}	0000000e00000004	ΓX_{26}	000000040000000e

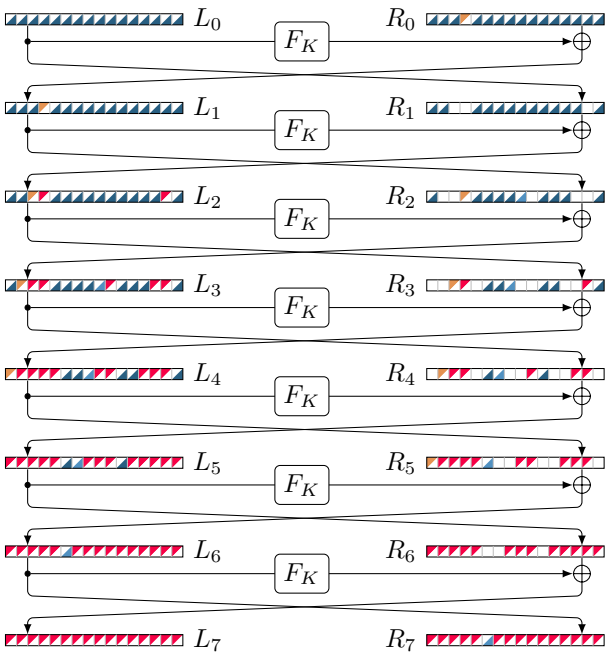


Fig. 43: DL distinguisher for 7-round Simeck-32 (active difference unknown difference active mask unknown mask).

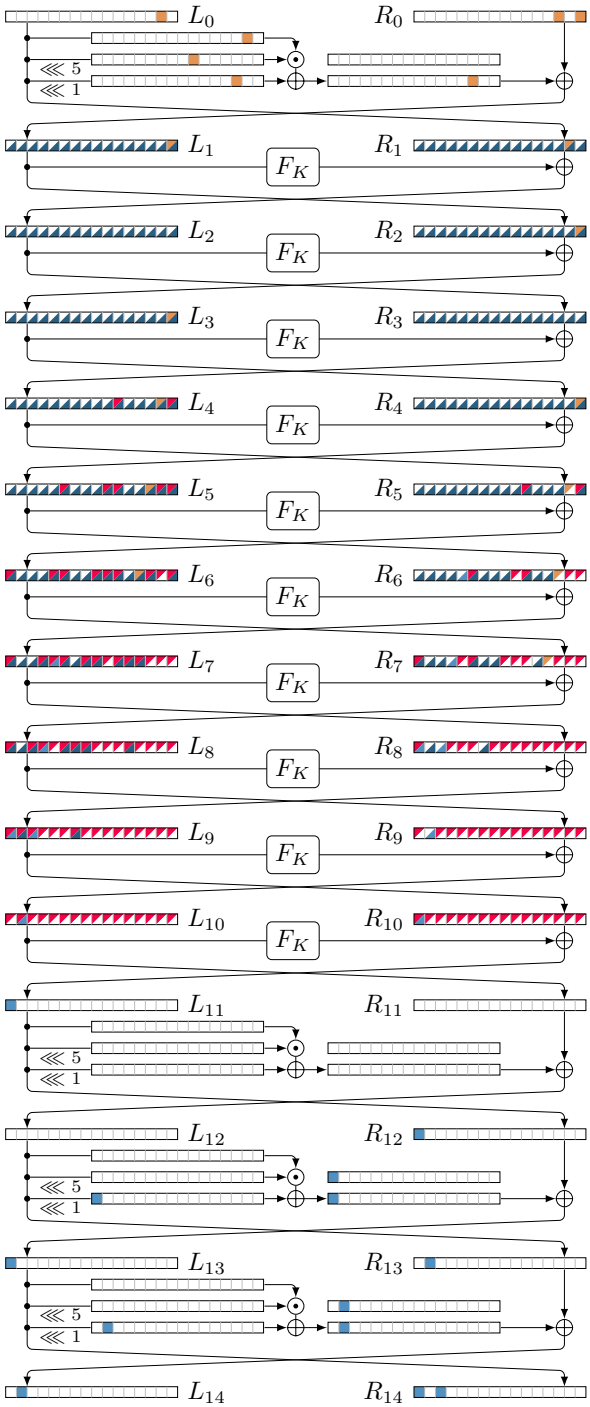


Fig. 44: DL distinguisher for 14-round Simeck-32 (active difference unknown difference active mask unknown mask).

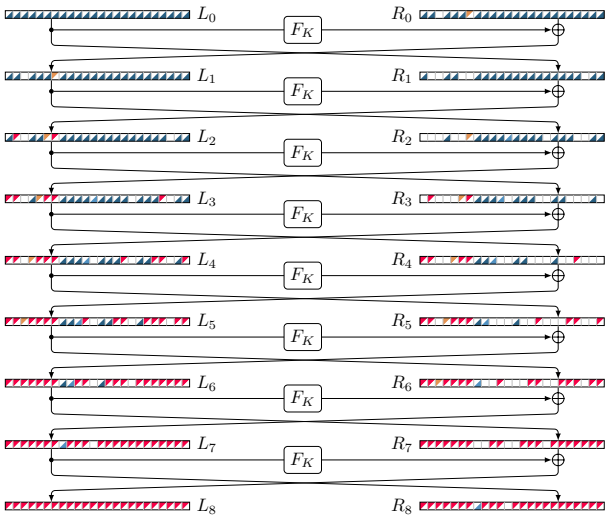


Fig. 45: DL distinguisher for 8-round Simeck-48 (active difference unknown difference active mask unknown mask).

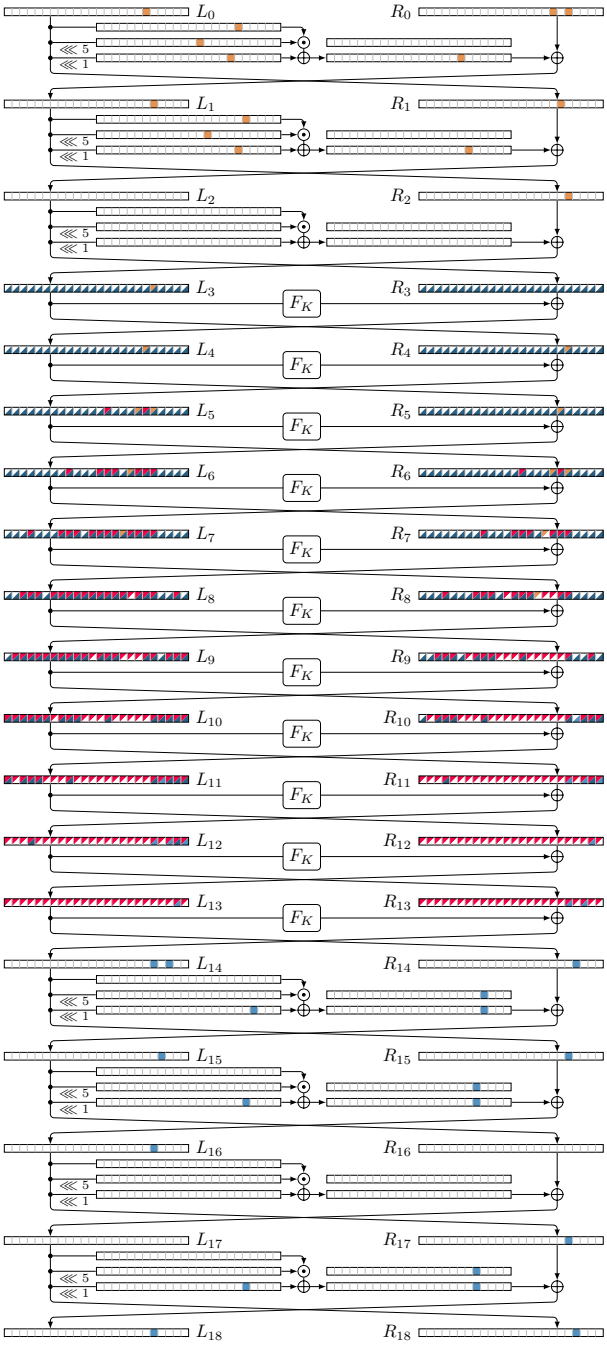


Fig. 46: DL distinguisher for 18-round Simeck-48 (active difference unknown difference active mask unknown mask).

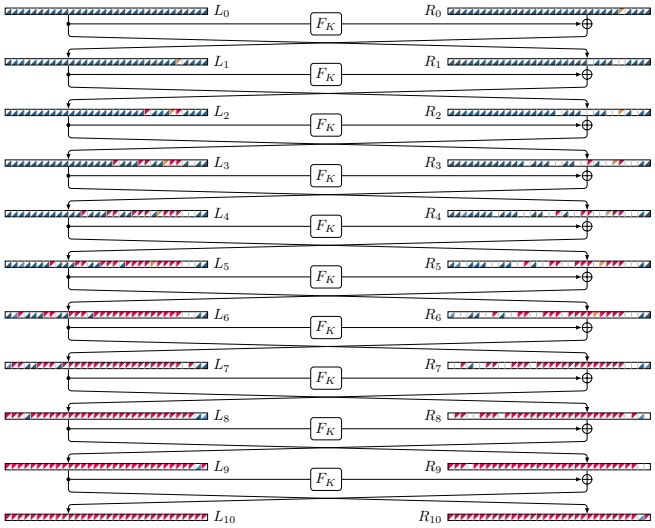


Fig. 47: DL distinguisher for 10-round Simeck-64 (active difference unknown difference active mask unknown mask).

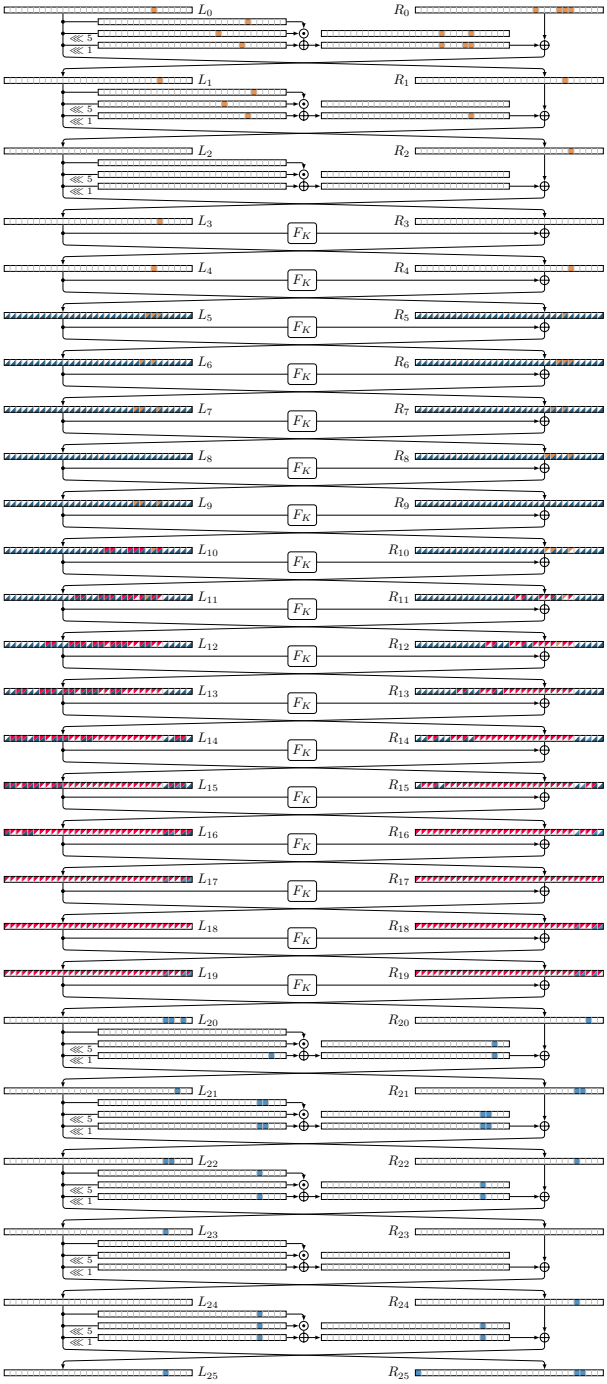


Fig. 48: DL distinguisher for 25-round Simeck-64 (active difference unknown difference active mask unknown mask).

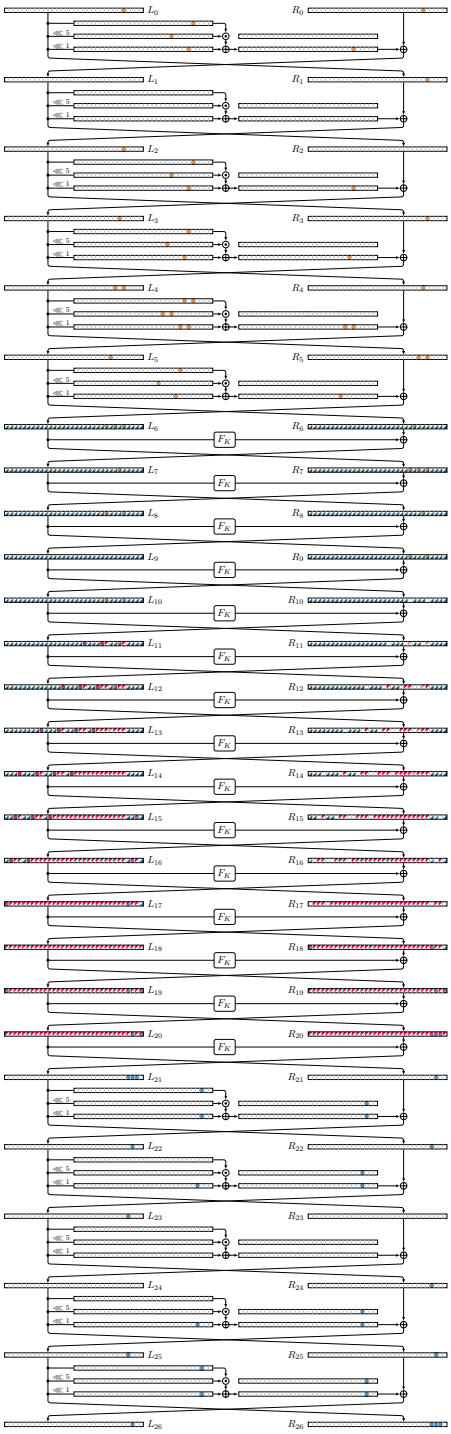
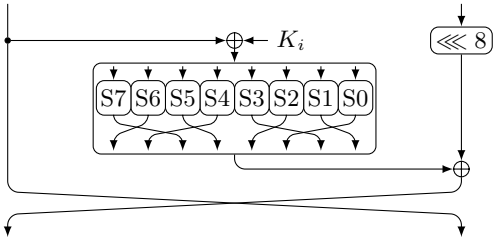


Fig. 49: DL distinguisher for 26-round Simeck-64 (active difference unknown difference active mask unknown mask).

J Application to LBlock and LBlock-s

J.1 Brief Specification of LBlock

LBlock is a block cipher with a 64-bit block size and an 80-bit key size, introduced in ACNS 2011 [57]. As shown in Figure 50a, the round function of LBlock is a 2-branch balanced Feistel structure that applies an 8-bit left rotation to the right branch. The round function uses 8 different 4×4 S-boxes, denoted by $\mathcal{S}_i$ for $0 \leq i \leq 7$. Notably, the differential uniformity and linearity of the S-boxes are the same, being 4 and 8, respectively. LBlock achieves full nibble-wise diffusion after 8 rounds. LBlock-s is a simplified version of LBlock, employing exactly the same structure as LBlock, except that it only uses the S-box $\mathcal{S}_0$ for all nibbles in the round function. The `DDT`, `LAT`, `DLCT`, and `DDLCT` of $\mathcal{S}_0$ are given in Table 20, Table 21, Table 22, and Table 23, respectively.



(a) The round function of LBlock.

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$\mathcal{S}_0(x)$	e	9	f	0	d	4	a	b	1	2	8	3	7	6	c	5
$\mathcal{S}_1(x)$	4	b	e	9	f	d	0	a	7	c	5	6	2	8	1	3
$\mathcal{S}_2(x)$	1	e	7	c	f	d	0	6	b	5	9	3	2	4	8	a
$\mathcal{S}_3(x)$	7	6	8	b	0	f	3	e	9	a	c	d	5	2	4	1
$\mathcal{S}_4(x)$	e	5	f	0	7	2	c	d	1	8	4	9	b	a	6	3
$\mathcal{S}_5(x)$	2	d	b	c	f	e	0	9	7	a	6	3	1	8	4	5
$\mathcal{S}_6(x)$	b	9	4	e	0	f	a	d	6	c	5	7	3	8	1	2
$\mathcal{S}_7(x)$	d	a	f	0	e	4	9	b	2	1	8	3	7	5	c	6

(b) S-boxes of LBlock.

Fig. 50: LBlock block cipher.

J.2 The DL Distinguishers of LBlock and LBlock-s

Table 20: DDT of the S-box $\mathcal{S}_0$ in LBlock

$\Delta_i \setminus \Delta_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	4	0	2	0	0	0	2	0	4	0	2	0	0	0	2
2	0	4	0	2	0	0	0	2	0	4	0	2	0	0	0	2
3	0	0	4	0	0	0	4	0	0	0	4	0	0	0	4	0
4	0	0	0	2	4	2	4	0	0	0	0	2	0	2	0	0
5	0	0	0	0	4	2	0	2	0	0	4	0	0	2	0	2
6	0	0	4	0	4	2	0	2	0	0	0	0	0	2	0	2
7	0	0	0	2	4	2	0	0	0	0	0	2	0	2	4	0
8	0	0	2	2	0	0	2	2	0	0	2	2	0	0	2	2
9	0	0	0	2	0	0	0	2	4	0	0	2	4	0	0	2
a	0	4	2	0	0	0	2	0	0	0	2	0	0	4	2	0
b	0	4	0	0	0	0	0	0	4	0	0	0	4	4	0	0
c	0	0	2	2	0	2	2	0	2	2	0	0	2	0	0	2
d	0	0	0	2	0	2	0	0	2	2	2	0	2	0	2	2
e	0	0	2	0	0	2	2	2	2	2	0	2	2	0	0	0
f	0	0	0	0	0	2	0	2	2	2	2	2	2	0	2	0

Table 21: LAT of S-box $\mathcal{S}_0$ in LBlock (scale: $2^4 \cdot$ correlation)

$\lambda_i \setminus \lambda_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	-4	4	4	-4	-8	-8	0	0	4	-4	4	-4
2	0	0	0	-8	-4	-4	4	-4	4	4	-4	4	0	0	0	-8
3	0	-8	0	0	0	0	0	8	4	-4	4	4	4	4	4	-4
4	0	0	0	0	8	0	8	0	0	0	0	0	8	0	-8	0
5	0	0	0	0	-4	-4	4	4	0	0	8	-8	-4	-4	-4	-4
6	0	0	0	-8	4	-4	-4	-4	-4	-4	4	-4	0	8	0	0
7	0	8	0	0	0	-8	0	0	4	-4	4	4	4	-4	4	4
8	0	0	0	0	0	0	0	0	-8	8	8	8	0	0	0	0
9	0	0	-8	-8	-4	4	-4	4	0	0	0	0	4	-4	-4	4
a	0	0	8	0	-4	-4	-4	4	-4	-4	-4	4	0	0	-8	0
b	0	8	0	0	0	0	0	8	-4	4	-4	-4	4	4	4	-4
c	0	0	0	0	-8	0	8	0	0	0	0	0	0	8	0	8
d	0	0	-8	8	-4	-4	-4	-4	0	0	0	0	4	4	-4	-4
e	0	0	-8	0	4	-4	4	4	-4	-4	-4	4	-8	0	0	0
f	0	8	0	0	0	8	0	0	4	-4	4	4	-4	4	-4	-4

Table 22: DLCT of S-box $\mathcal{S}_0$ in LBlock

$\Delta_i \setminus \lambda_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	-16	0	0	8	-8	8	-8	0	0	0	0	0	0	0	0
2	16	-16	0	0	8	-8	8	-8	0	0	0	0	0	0	0	0
3	16	16	-16	-16	0	0	0	0	0	0	0	0	0	0	0	0
4	16	0	0	0	-8	-8	-8	8	8	8	0	0	-8	-8	0	0
5	16	0	0	0	-8	8	-8	-8	0	0	8	8	-8	-8	0	0
6	16	0	0	0	-8	8	-8	-8	8	8	0	0	0	0	-8	-8
7	16	0	0	0	-8	-8	-8	8	0	0	8	8	0	0	-8	-8
8	16	0	-16	0	0	0	0	0	0	0	0	0	0	0	0	0
9	16	0	0	16	0	0	0	0	-8	-8	-8	-8	0	0	0	0
a	16	0	0	-16	0	0	0	0	0	0	0	0	8	-8	8	-8
b	16	0	16	0	0	0	0	0	-8	-8	-8	-8	8	-8	8	-8
c	16	0	0	0	0	0	0	0	0	0	-8	-8	0	0	-8	8
d	16	0	0	0	0	0	0	0	-8	-8	0	0	0	0	-8	8
e	16	0	0	0	0	0	0	0	0	0	-8	-8	-8	8	0	0
f	16	0	0	0	0	0	0	0	-8	-8	0	0	-8	8	0	0

Table 23: DDLCT of S-box S_0 in LBlock

$\Delta_i \setminus \lambda_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	256	256	256	256	256	256	256	256	256	256	256	256	256	256	256	256
1	256	-32	0	32	16	-48	16	-16	-64	-64	-32	-32	0	0	0	-32
2	256	-32	0	32	16	-48	16	-16	-64	-64	-32	-32	0	0	0	-32
3	256	-64	0	-64	0	0	0	-64	32	32	-32	-32	0	0	0	-64
4	256	32	0	-32	-80	16	-80	-16	32	32	0	0	-32	-64	-32	-32
5	256	0	0	-64	-64	-32	-64	32	0	0	32	32	-32	-64	0	-32
6	256	-64	0	0	-32	-64	-32	0	0	0	32	32	-64	-32	-32	0
7	256	32	0	-32	-48	-16	-48	16	0	0	-32	-32	-64	-32	0	0
8	256	0	0	-64	-16	-16	-16	-16	-16	-16	-16	-16	0	0	0	-64
9	256	32	-64	-32	-16	-16	-16	16	-32	-32	-32	-32	0	0	-32	0
a	256	**-96**	0	-32	32	-32	32	-64	-16	-16	-16	-16	0	0	-32	0
b	256	-64	-64	0	32	-32	32	-32	-32	-32	-32	-32	0	0	-64	64
c	256	0	-64	0	-16	16	-16	-48	-16	-16	-16	-16	-32	0	-32	0
d	256	32	-64	-32	-16	16	-16	-16	-32	-32	-32	-32	-32	0	0	0
e	256	-32	0	32	-32	0	-32	-32	-16	-16	-16	-16	0	-32	-32	-32
f	256	0	0	0	-32	0	-32	0	-32	-32	-32	-32	0	-32	0	-32

Table 24: Specification of the DL Distinguishers for LBlock-s.

7 Rounds $\delta, \lambda \in \mathbb{F}_2^4 \setminus \{0\}, r_u = 0, r_m = 7, r_\ell = 0,\ p = 1,\ r = \mathbf{1}, q^2 = 1,\ prq^2 = \mathbf{1}, \hat{\mathbb{C}} = \mathbf{1}$			
ΔX_0	000000000δ000000	ΔX_3	00000000000000λ0
8 Rounds $r_u = 0, r_m = 8, r_\ell = 0,\ p = 1,\ r = 2^{-1.42}, q^2 = 1,\ prq^2 = 2^{-1.42}, \hat{\mathbb{C}} = 2^{-1.42}$			
ΔX_0	000000000a000000	ΓX_8	0000000000010000
9 Rounds $r_u = 1, r_m = 8, r_\ell = 0,\ p = 2^{-2},\ r = 2^{-1.42}, q^2 = 1,\ prq^2 = 2^{-3.42}, \hat{\mathbb{C}} = 2^{-3.45}$			
ΔX_0	0000000a00000001	ΔX_1	000000000000000a
ΓX_9	0000000000000100		
10 Rounds $r_u = 1, r_m = 8, r_\ell = 1,\ p = 2^{-2},\ r = 2^{-1.42}, q^2 = 2^{-2},\ prq^2 = 2^{-5.42}, \hat{\mathbb{C}} = 2^{-5.43}$			
ΔX_0	000000a001000000	ΔX_1	00000000000000a0
ΓX_9	0000000000001000	ΓX_{10}	0010000070000000
11 Rounds $r_u = 2, r_m = 8, r_\ell = 1,\ p = 2^{-4},\ r = 2^{-1.42}, q^2 = 2^{-2},\ prq^2 = 2^{-7.42}, \hat{\mathbb{C}} = 2^{-7.43}$			
ΔX_0	000000100900000a	ΔX_2	0000000000000a00
ΓX_{10}	0000000000000001	ΓX_{11}	0000010000000007
12 Rounds $r_u = 2, r_m = 8, r_\ell = 2,\ p = 2^{-4},\ r = 2^{-1.42}, q^2 = 2^{-4},\ prq^2 = 2^{-9.42}, \hat{\mathbb{C}} = 2^{-9.42}$			
ΔX_0	000000100900000a	ΔX_2	0000000000000a00
ΓX_{10}	0000000000000001	ΓX_{11}	0000010000000007
13 Rounds $r_u = 2, r_m = 9, r_\ell = 2,\ p = 2^{-4},\ r = 2^{-4.18}, q^2 = 2^{-4},\ prq^2 = 2^{-12.10}, \hat{\mathbb{C}} = 2^{-12.10}$			
ΔX_0	0200000030100000	ΔX_2	0000000000000030
ΓX_{11}	0000000010000000	ΓX_{13}	00b0000080000010
14 Rounds $r_u = 2, r_m = 9, r_\ell = 3,\ p = 2^{-4},\ r = 2^{-6.80}, q^2 = 2^{-4},\ prq^2 = 2^{-14.80}$			
ΔX_0	000000100900000a	ΔX_2	0000000000000a00
ΓX_{11}	0100000000000000	ΓX_{14}	0000700000000b01
15 Rounds $r_u = 3, r_m = 9, r_\ell = 3,\ p = 2^{-8},\ r = 2^{-6.80}, q^2 = 2^{-4},\ prq^2 = 2^{-18.80}$			
ΔX_0	90a000000000c110	ΔX_3	00000000a0000000
ΓX_{12}	1000000000000000	ΓX_{15}	00700000b0000010
16 Rounds $r_u = 4, r_m = 9, r_\ell = 3,\ p = 2^{-8},\ r = 2^{-6.80}, q^2 = 2^{-8},\ prq^2 = 2^{-22.80}$			
ΔX_0	000090a081100000	ΔX_4	00a0000000000000
ΓX_{13}	0000000000001000	ΓX_{16}	10500000770000b0
17 Rounds $r_u = 4, r_m = 9, r_\ell = 4,\ p = 2^{-8},\ r = 2^{-6.80}, q^2 = 2^{-14},\ prq^2 = 2^{-28.80}$			
ΔX_0	0a00000901100008	ΔX_4	00000a0000000000
ΓX_{13}	0000000000000001	ΓX_{17}	f00705000c310b07

Table 25: Specification of the DL Distinguishers for LBlock.

7 Rounds $\delta, \lambda \in \mathbb{F}_2^4 \setminus \{0\}, r_u = 0, r_m = 7, r_\ell = 0,\ p = 1,\ r = \mathbf{1}, q^2 = 1,\ prq^2 = \mathbf{1}, \hat{\mathbb{C}} = \mathbf{1}$			
ΔX_0	000000000δ000000	ΔX_3	00000000000000λ0
8 Rounds $r_u = 0, r_m = 8, r_\ell = 0,\ p = 1,\ r = 2^{-2.00}, q^2 = 1,\ prq^2 = 2^{-2.00}, \hat{\mathbb{C}} = 2^{-2.00}$			
ΔX_0	00000000000000a0	ΓX_8	0000000020000000
9 Rounds $r_u = 0, r_m = 9, r_\ell = 0,\ p = 1,\ r = 2^{-3.89}, q^2 = 1,\ prq^2 = 2^{-3.89}, \hat{\mathbb{C}} = 2^{-3.89}$			
ΔX_0	0000000000000030	ΓX_9	0000000010000000
10 Rounds $r_u = 1, r_m = 8, r_\ell = 1,\ p = 2^{-2},\ r = 2^{-2.00}, q^2 = 2^{-2},\ prq^2 = 2^{-6.00}, \hat{\mathbb{C}} = 2^{-6.03}$			
ΔX_0	000000a00e000000	ΔX_1	00000000000000a0
ΓX_9	0000000000001000	ΓX_{10}	00100000a0000000
11 Rounds $r_u = 2, r_m = 8, r_\ell = 1,\ p = 2^{-4},\ r = 2^{-2.00}, q^2 = 2^{-2},\ prq^2 = 2^{-8.00}, \hat{\mathbb{C}} = 2^{-8.04}$			
ΔX_0	000000010a000001	ΔX_2	000000000000000a
ΓX_{10}	0000000001000000	ΓX_{11}	00000001000000a0
12 Rounds $r_u = 2, r_m = 8, r_\ell = 2,\ p = 2^{-4},\ r = 2^{-2.00}, q^2 = 2^{-4},\ prq^2 = 2^{-10.00}, \hat{\mathbb{C}} = 2^{-9.98}$			
ΔX_0	b0000000000080a0	ΔX_2	000000000000a000
ΓX_{10}	0000000000000010	ΓX_{11}	0004000000901000
13 Rounds $r_u = 2, r_m = 9, r_\ell = 2,\ p = 2^{-4},\ r = 2^{-3.89}, q^2 = 2^{-4},\ prq^2 = 2^{-11.89}, \hat{\mathbb{C}} = 2^{-11.54}$			
ΔX_0	0100000030200000	ΔX_2	0000000000000030
ΓX_{11}	0000000010000000	ΓX_{13}	0030000020000010
14 Rounds $r_u = 2, r_m = 9, r_\ell = 3,\ p = 2^{-4},\ r = 2^{-7.17}, q^2 = 2^{-4},\ prq^2 = 2^{-15.17}$			
ΔX_0	000000700400000a	ΔX_2	0000000000000a00
ΓX_{11}	0100000000000000	ΓX_{14}	0000a00000000901
15 Rounds $r_u = 3, r_m = 9, r_\ell = 3,\ p = 2^{-4},\ r = 2^{-7.13}, q^2 = 2^{-8},\ prq^2 = 2^{-19.13}$			
ΔX_0	00200000000a0100	ΔX_3	0000000a00000000
ΓX_{12}	0000000000000100	ΓX_{15}	01000008b00b0010
16 Rounds $r_u = 3, r_m = 9, r_\ell = 4,\ p = 2^{-8},\ r = 2^{-7.13}, q^2 = 2^{-8},\ prq^2 = 2^{-23.13}$			
ΔX_0	000a010000012020	ΔX_3	000000000a000000
ΓX_{12}	0000010000000000	ΓX_{16}	01000008b00b0010
17 Rounds $r_u = 4, r_m = 9, r_\ell = 4,\ p = 2^{-8},\ r = 2^{-7.15}, q^2 = 2^{-14},\ prq^2 = 2^{-29.15}$			
ΔX_0	0a00000a20e00001	ΔX_4	000a000000000000
ΓX_{13}	0000000001000000	ΓX_{17}	e0900f008604010c

K Application to WARP

K.1 Brief Specification of WARP

WARP is a block cipher with 128-bit plaintext and key designed by Banik et al. [2]. It performs 40 full rounds plus 1 partial round to produce the ciphertext. The internal state of WARP is organized in nibbles $X = X_0||\cdots||X_{31}$, where $X_i \in \{0,1\}^4$. WARP splits the 128-bit key K into two 64-bit halves $K = K^0||K^1$, where $K^{(r-1) \bmod 2}$ is used as the rth round-key. The ith nibble of the round input $X^{(r-1)}$ and round-key $K^{((r-1) \bmod 2)}$ in round r are denoted by $X_i^{(r-1)}$ and $K_i^{(b)}$, where $1 \le r \le 41$, $b \in \{0,1\}$, and $0 \le i \le 15$.

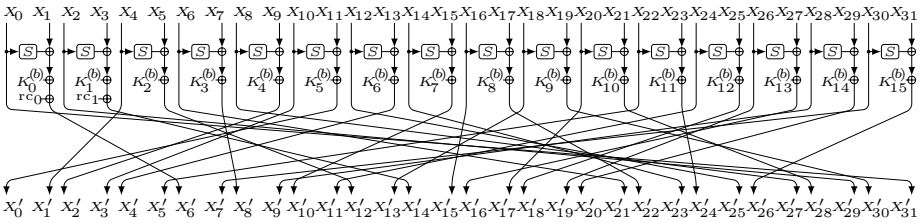


Fig. 51: The round function of WARP.

The round function of WARP, illustrated in Figure 51, first applies an S-box $S : \{0,1\}^4 \to \{0,1\}^4$ (Table 27) as well as the round-key and round-constant addition on each of two consecutive nibbles of internal state. Next, a nibble permutation π (Table 26) is applied, except in the last round.

Table 26: Nibble permutation π of WARP.

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\pi(x)$	31	6	29	14	1	12	21	8	27	2	3	0	25	4	23	10
x	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$\pi(x)$	15	22	13	30	17	28	5	24	11	18	19	16	9	20	7	26

Table 27: 4-bit S-box S of WARP.

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S(x)$	c	a	d	3	e	b	f	7	8	9	1	5	0	2	4	6

Table 28: DDT of WARP's S-box.

$\Delta_i \setminus \Delta_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	2	4	0	2	2	2	0	2	0	0	0	0	0	2	0
2	0	4	0	0	4	0	0	0	0	4	0	0	4	0	0	0
3	0	0	0	0	2	0	4	2	2	2	0	0	0	2	0	2
4	0	2	4	2	2	2	0	0	2	0	0	2	0	0	0	0
5	0	2	0	0	2	0	0	4	0	2	4	0	2	0	0	0
6	0	2	0	4	0	0	0	2	2	0	0	0	2	2	0	2
7	0	0	0	2	0	4	2	0	0	0	0	2	0	4	2	0
8	0	2	0	2	2	0	2	0	0	2	0	2	2	0	2	0
9	0	0	4	2	0	2	0	0	2	2	0	2	2	0	0	0
a	0	0	0	0	0	4	0	0	0	0	4	0	0	4	0	4
b	0	0	0	0	2	0	0	2	2	2	0	4	0	2	0	2
c	0	0	4	0	0	2	2	0	2	2	0	0	2	0	2	0
d	0	0	0	2	0	0	2	4	0	0	4	2	0	0	2	0
e	0	2	0	0	0	0	0	2	2	0	0	0	2	2	4	2
f	0	0	0	2	0	0	2	0	0	0	4	2	0	0	2	4

Table 29: LAT of WARP's S-box (scale: $2^4 \cdot$ correlation).

$\lambda_i \setminus \lambda_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	4	8	4	-4	0	4	0	-4	0	4	0	8	-4	0	-4
2	0	8	0	0	8	0	0	0	-8	0	0	0	0	8	0	0
3	0	4	0	4	-4	0	4	8	4	-8	-4	0	0	4	0	4
4	0	-4	8	-4	4	0	-4	0	-4	-8	-4	0	0	-4	0	4
5	0	0	0	0	0	0	0	0	0	0	-8	-8	0	0	8	-8
6	0	4	0	4	-4	0	4	-8	-4	0	-4	0	-8	-4	0	4
7	0	0	0	8	0	0	-8	0	0	0	0	-8	0	0	-8	0
8	0	-4	-8	4	-4	0	-4	0	-8	-4	0	4	4	0	4	0
9	0	0	0	-8	-8	0	0	0	-4	4	-4	-4	4	4	-4	4
a	0	4	0	-4	-4	-8	-4	0	0	-4	8	-4	-4	0	4	0
b	0	0	0	0	0	-8	0	-8	4	-4	-4	4	4	4	-4	-4
c	0	8	0	0	0	0	-8	0	4	4	-4	4	4	-4	4	4
d	0	-4	8	4	-4	0	-4	0	0	4	0	4	-4	8	4	0
e	0	0	0	0	0	8	0	-8	4	-4	4	-4	4	4	4	4
f	0	-4	0	4	4	-8	4	0	0	4	0	-4	4	0	4	8

Table 30: DLCT of WARP's S-box.

$\Delta_i \setminus \lambda_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	8	0	-8	0	0	0	0	8	0	0	-8	0	0	-8	-8
2	16	0	16	0	0	-16	0	-16	0	0	0	0	0	0	0	0
3	16	0	0	0	-8	0	8	0	0	8	-8	0	-8	-8	0	0
4	16	0	0	0	8	0	-8	0	8	0	0	-8	0	0	-8	-8
5	16	0	0	0	0	0	0	-16	0	-8	0	8	-8	0	8	0
6	16	-8	0	8	0	0	0	0	0	-8	-8	0	8	-8	0	0
7	16	-8	0	-8	-8	0	-8	16	0	0	0	0	0	0	0	0
8	16	0	0	0	0	-16	0	0	0	0	0	0	0	0	0	0
9	16	0	0	0	8	0	-8	0	0	0	-8	-8	0	8	-8	0
a	16	-8	0	-8	-8	16	-8	0	-8	0	8	0	0	-8	0	8
b	16	-8	0	8	0	0	0	0	-8	8	0	0	-8	0	0	-8
c	16	8	0	-8	0	0	0	0	0	0	-8	-8	0	8	-8	0
d	16	0	-16	0	0	0	0	0	0	-8	0	8	-8	0	8	0
e	16	0	0	0	-8	0	8	0	-8	-8	0	0	8	0	0	-8
f	16	0	-16	0	0	0	0	0	-8	0	8	0	0	-8	0	8

Table 31: DDLCT of WARP's S-box.

$\Delta_i \setminus \lambda_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	256	256	256	256	256	256	256	256	256	256	256	256	256	256	256	256
1	256	0	64	0	0	-96	0	-96	16	-48	-16	-16	16	-16	-16	-48
2	256	64	0	-64	64	0	-64	0	64	0	-64	**-128**	0	64	**-128**	-64
3	256	-48	-64	16	16	-32	-48	32	0	-48	-32	-16	16	-32	-16	0
4	256	0	64	0	0	-96	0	-96	16	16	-16	-16	-48	-16	-16	-48
5	256	-32	0	-96	-32	64	-96	64	0	0	0	-64	0	0	-64	0
6	256	16	-64	-48	-48	-32	16	32	0	16	-32	-16	-48	-32	-16	0
7	256	-32	-64	32	-32	0	32	-64	-32	-64	-32	64	-64	-32	64	-32
8	256	0	0	0	0	0	0	0	0	0	-64	-64	0	0	-64	-64
9	256	0	64	0	0	-96	0	-96	-16	16	-48	-16	-48	16	-16	-16
a	256	-32	**-128**	-32	-32	64	-32	-64	-64	-64	64	64	-64	-64	64	64
b	256	-48	-64	16	16	-32	-48	32	-32	16	0	-16	-48	0	-16	-32
c	256	0	64	0	0	-96	0	-96	-16	-48	-48	-16	16	16	-16	-16
d	256	-96	0	-32	-96	64	-32	64	-64	0	0	0	0	-64	0	0
e	256	16	-64	-48	-48	-32	16	32	-32	-48	0	-16	16	0	-16	-32
f	256	-64	-64	0	-64	64	0	0	-96	0	32	0	0	-96	0	32

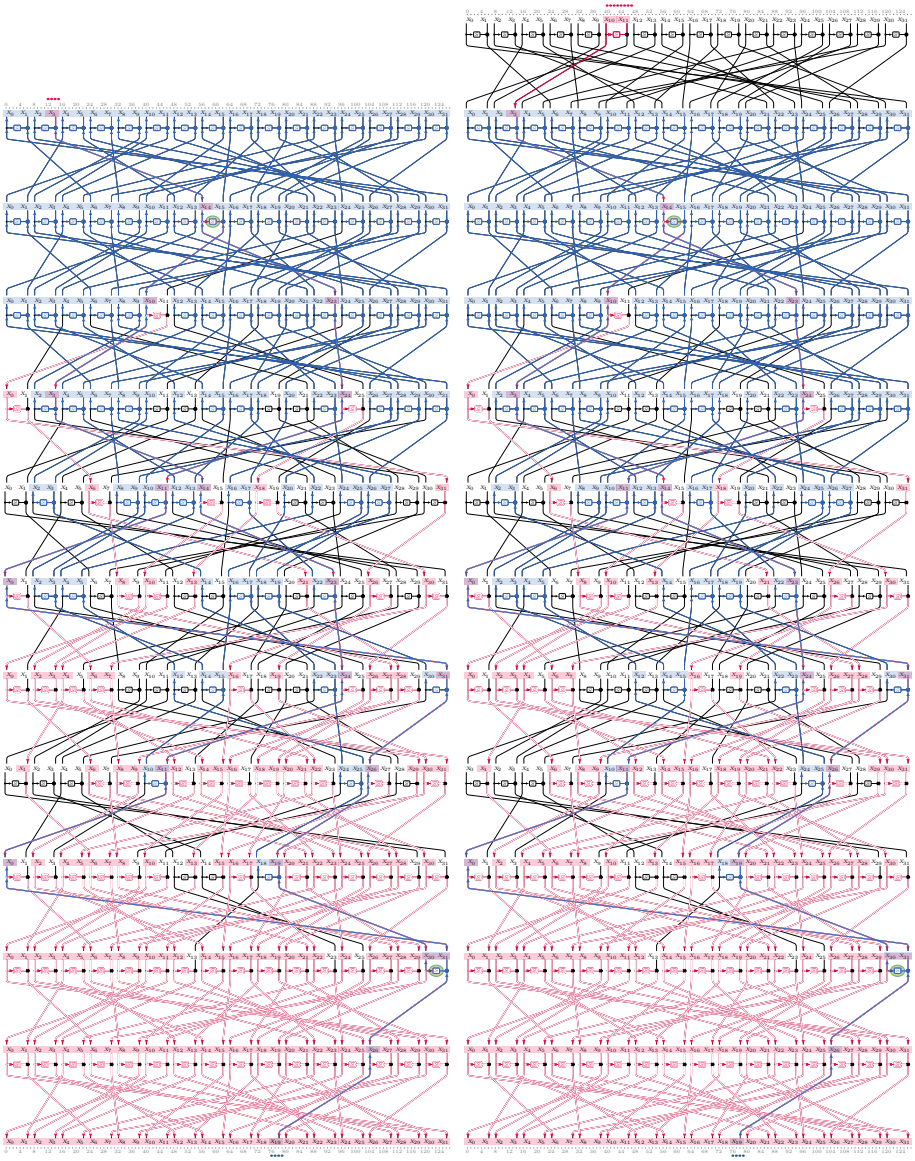
K.2 The DL Distinguishers of WARP

Table 32: Specification of DL distinguishers for 11 to 22 rounds of WARP.

11 Rounds, Figure 52a $r_u = 0, r_m = 11, r_\ell = 0,\ p = 1,\ r = 1,\ q^2 = 1,\ prq^2 = \mathbf{1}$	
ΔX_0 00020000000000000000000000000000	ΓX_{11} 00000000000000000002000000000000
12 Rounds, Figure 52b $r_u = 1, r_m = 11, r_\ell = 0,\ p = 2^{-2},\ r = 1,\ q^2 = 1,\ prq^2 = 2^{-2}$	
ΔX_0 00000000029000000000000000000000	ΔX_1 00020000000000000000000000000000
ΓX_{12} 00000000000000000002000000000000	
13 Rounds, Figure 53a $r_u = 1, r_m = 11, r_\ell = 1,\ p = 2^{-2},\ r = 1,\ q^2 = 2^{-2},\ prq^2 = 2^{-4}$	
ΔX_0 00000000000000002900000000000000	ΔX_1 00000000000000020000000000000000
ΓX_{12} 00000000000000020000000000000000	ΓX_{13} 00000000020000000000000400000000
14 Rounds, Figure 53b $r_u = 2, r_m = 11, r_\ell = 1,\ p = 2^{-4},\ r = 1,\ q^2 = 2^{-2},\ prq^2 = 2^{-6}$	
ΔX_0 00000000000000020000000012000000	ΔX_2 00020000000000000000000000000000
ΓX_{13} 00000000000000000002000000000000	ΓX_{14} 00000000000004000000000000000020
15 Rounds, Figure 54a $r_u = 2, r_m = 11, r_\ell = 2,\ p = 2^{-4},\ r = 1,\ q^2 = 2^{-4},\ prq^2 = 2^{-8}$	
ΔX_0 00000000000000020000000012000000	ΔX_2 00020000000000000000000000000000
ΓX_{13} 00000000000000000002000000000000	ΓX_{15} 00004002000000000000000009000000
16 Rounds, Figure 54b $r_u = 3, r_m = 11, r_\ell = 2,\ p = 2^{-4},\ r = 2^{-3.00},\ q^2 = 2^{-4},\ prq^2 = 2^{-11}$	
ΔX_0 000000000000000a00000000af000000	ΔX_3 00000000000000a00000000000000000
ΓX_{14} 00000000000000020000000000000000	ΓX_{16} 00020900000000000000000040000000
17 Rounds, Figure 55a $r_u = 3, r_m = 11, r_\ell = 3,\ p = 2^{-4},\ r = 2^{-3.00},\ q^2 = 2^{-8},\ prq^2 = 2^{-15}$	
ΔX_0 000000000000000a00000000af000000	ΔX_3 00000000000000a00000000000000000
ΓX_{14} 00000000000000020000000000000000	ΓX_{17} 08000000000820200000000000000d00

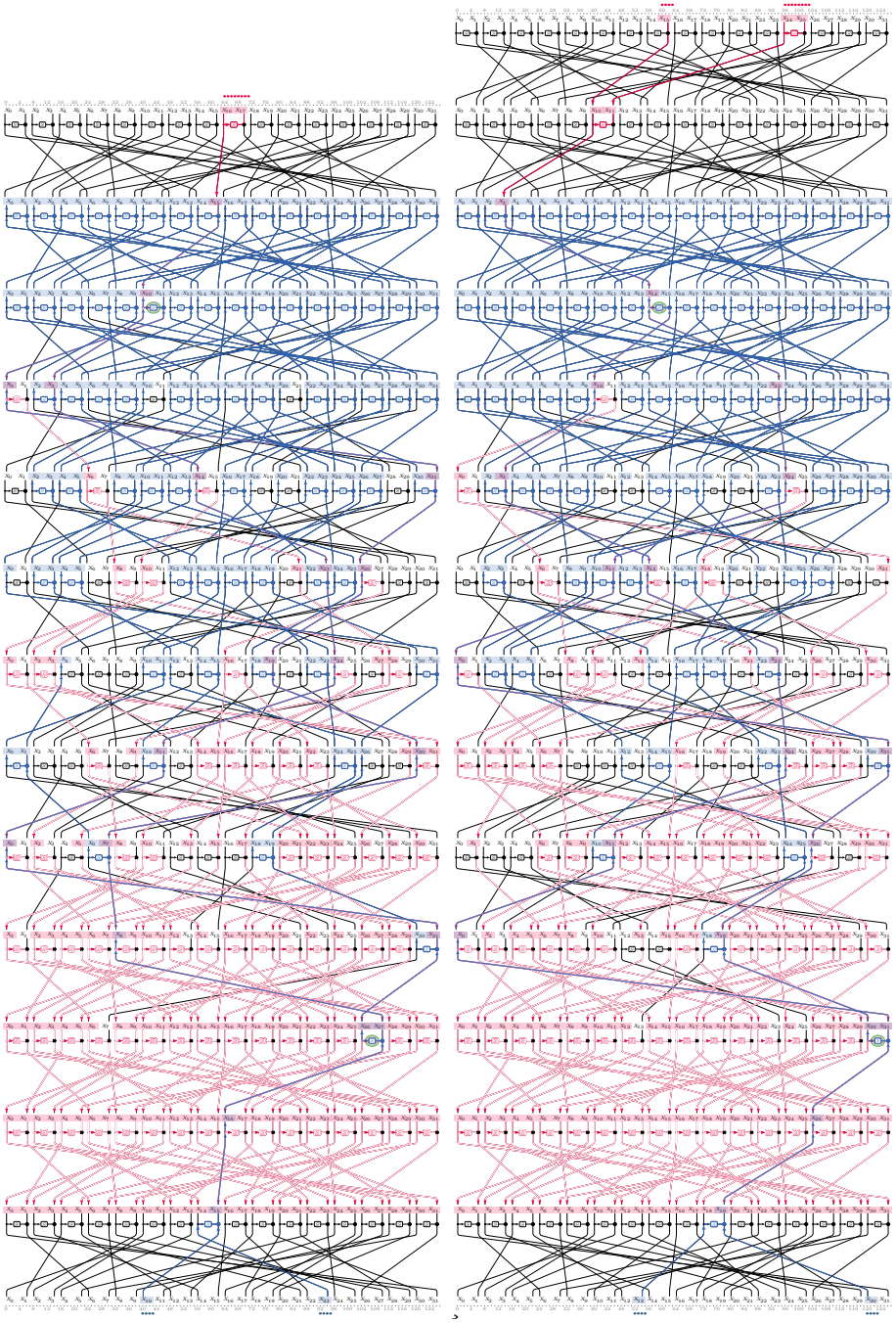
Table 33: Specification of DL distinguishers for 18 to 22 rounds of WARP.

18 Rounds, Figure 55b $r_u = 4, r_m = 11, r_\ell = 3,\ p = 2^{-8},\ r = 2^{-3.00},\ q^2 = 2^{-8},\ prq^2 = 2^{-19.00}$	
ΔX_0 29000004000000000000000000002100	ΔX_4 00000000000000000000000000000020
ΓX_{15} 0000000000000000000000000000000b	ΓX_{18} 00000000000050005000000005e0b0
19 Rounds, Figure 55c $r_u = 4, r_m = 11, r_\ell = 4,\ p = 2^{-14},\ r = 2^{-3.00},\ q^2 = 2^{-8},\ prq^2 = 2^{-25.00}$	
ΔX_0 00124200000200000000020000000042	ΔX_4 00000000000000000002000000000000
ΓX_{15} b0000000000000000000000000000000	ΓX_{19} 00000000000050005000000005e0b0
20 Rounds, Figure 56a $r_u = 5, r_m = 11, r_\ell = 4,\ p = 2^{-14},\ r = 2^{-3.00},\ q^2 = 2^{-14},\ prq^2 = 2^{-31.00}$	
ΔX_0 00124200000200000000020000000042	ΔX_5 00000000000000000000000000000020
ΓX_{16} 0000000000000000000000000000000b	ΓX_{20} 0000500b03000003700b00900a000000
21 Rounds,Figure 56b $r_u = 5, r_m = 11, r_\ell = 5,\ p = 2^{-24},\ r = 2^{-3.00},\ q^2 = 2^{-14},\ prq^2 = 2^{-41.00}$	
ΔX_0 00000200000000420012420000020000	ΔX_5 00000000000000200000000000000000
ΓX_{16} 000000000000000b0000000000000000	ΓX_{21} 00f00502000c006005600c00b01f0c07
22 Rounds, Figure 56c $r_u = 6, r_m = 10, r_\ell = 6,\ p = 2^{-24},\ r = 2^{-3.00},\ q^2 = 2^{-24},\ prq^2 = 2^{-51.00}$	
ΔX_0 00040024290000002100002104210400	ΔX_6 00000000000000200000000000000000
ΓX_{16} 0000000000000000b000000000000000	ΓX_{22} 0060050c000c006007600c00b06c0c07



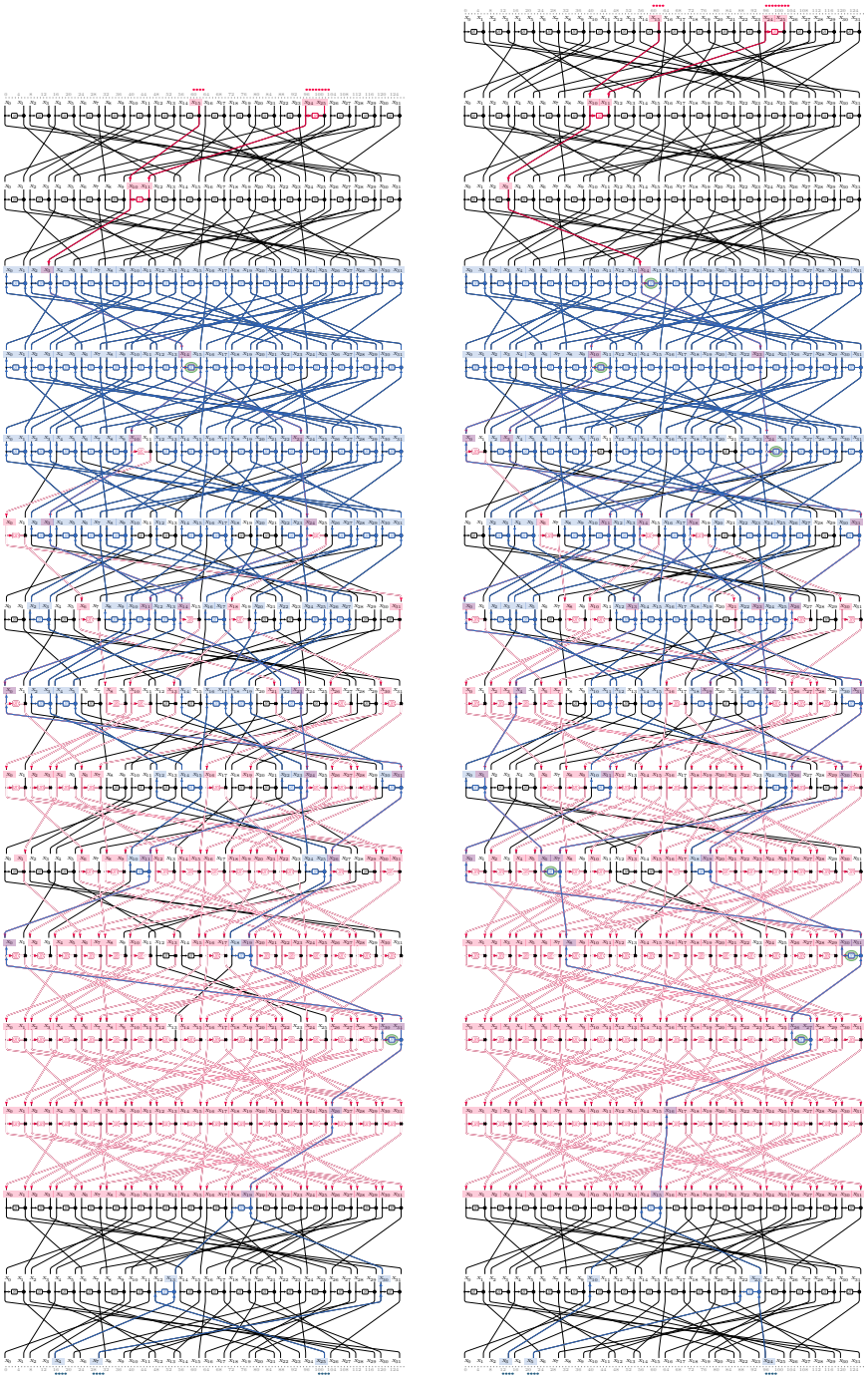
(a) 11-round WARP (b) 12-round WARP

Fig. 52: DL distinguishers for 11 to 12 rounds of WARP (– differential – linear).



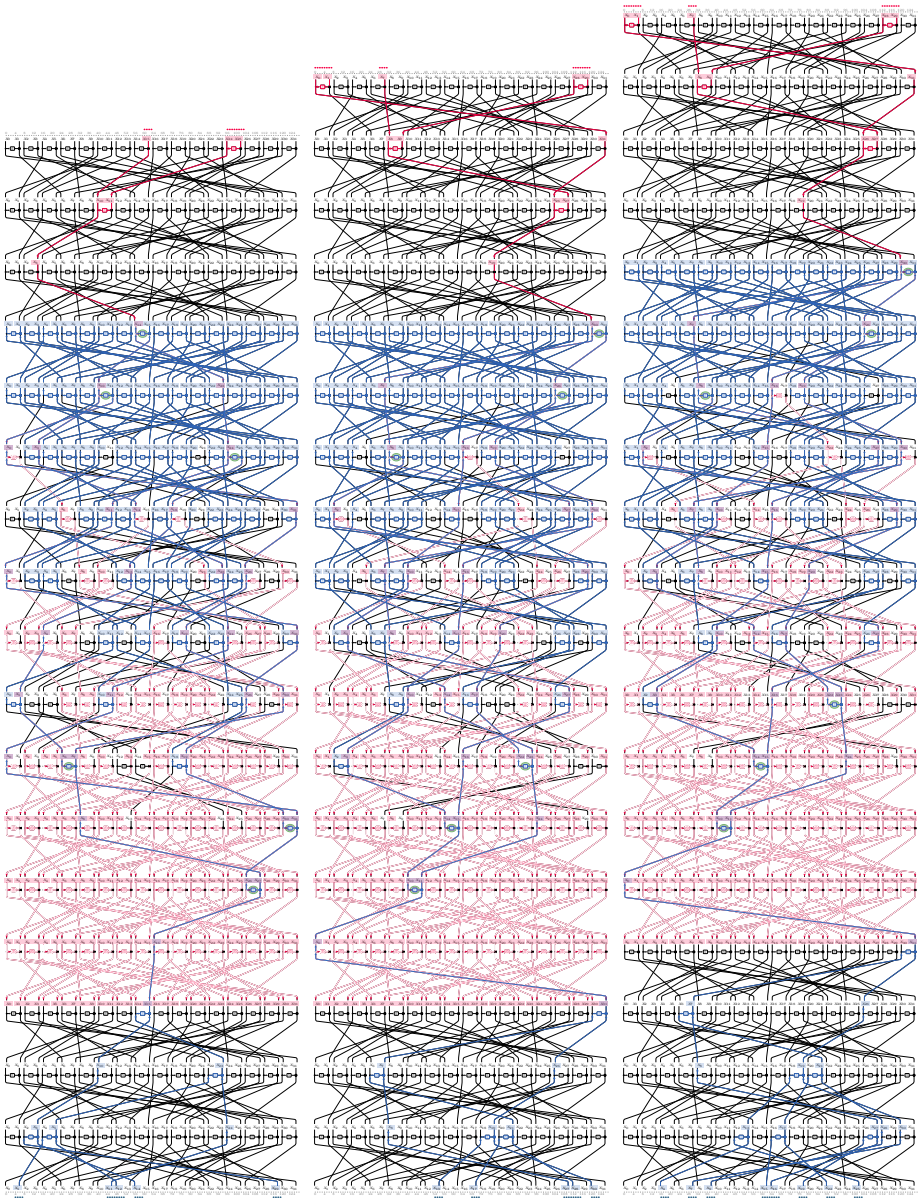
(a) 13-round WARP (b) 14-round WARP

Fig. 53: DL distinguishers for 13 to 14 rounds of WARP (– differential – linear).



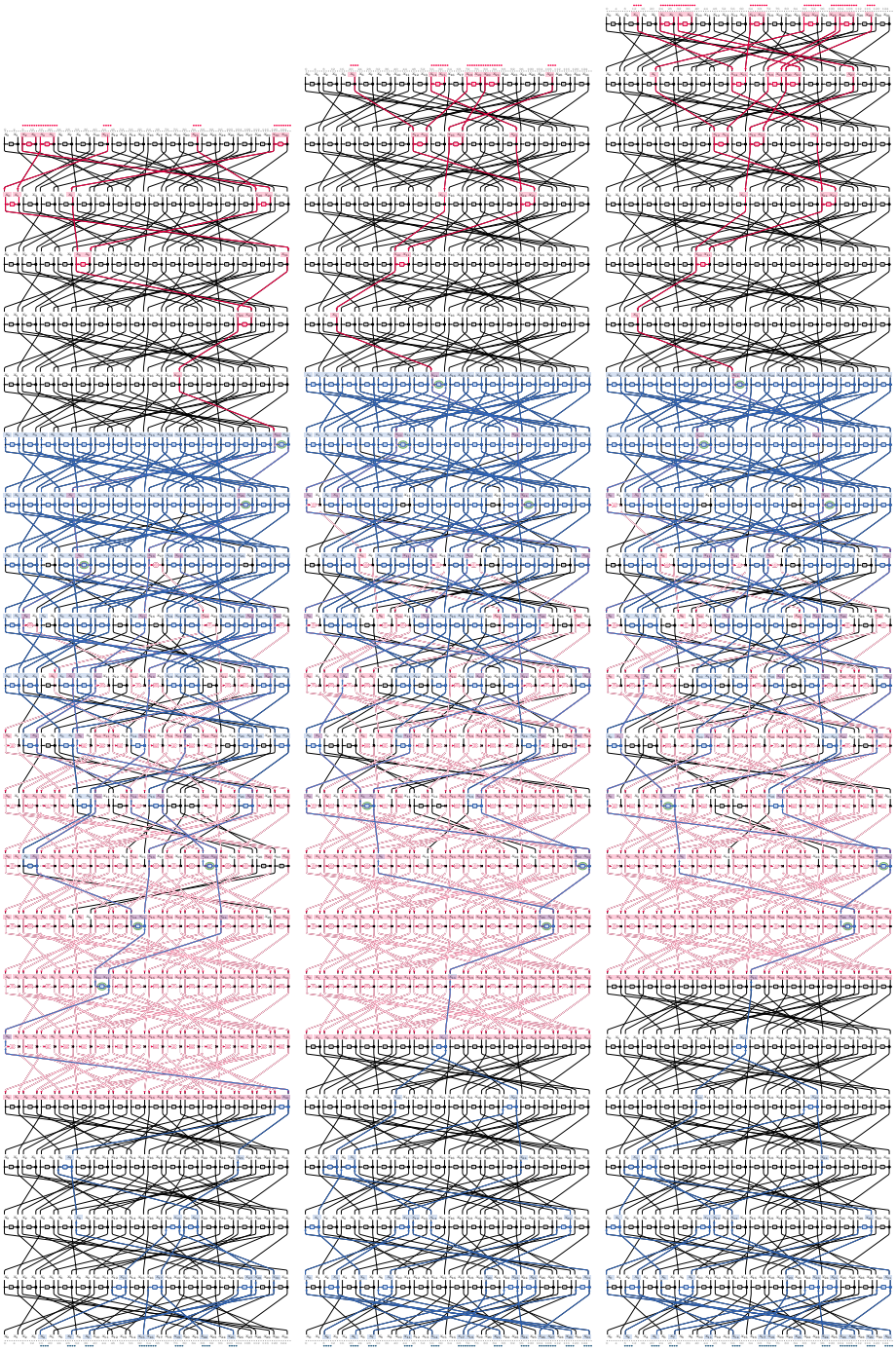
(a) 15-round WARP (b) 16-round WARP

Fig. 54: DL distinguishers for 15 to 16 rounds of WARP (– differential – linear).



(a) 17-round WARP (b) 18-round WARP (c) 19-round WARP

Fig. 55: DL distinguishers for 17 to 19 rounds of WARP (– differential – linear).



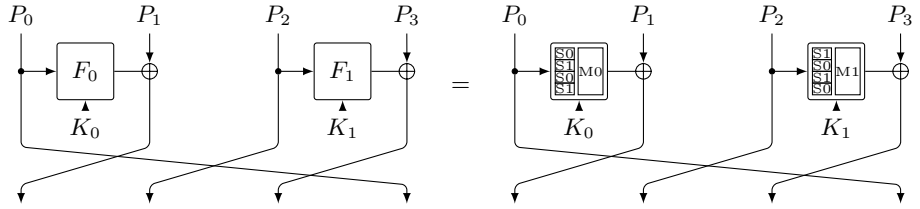
(a) 20-round WARP (b) 21-round WARP (c) 22-round WARP

Fig. 56: DL distinguishers for 20 to 22 rounds of WARP (– differential – linear).

L Application to CLEFIA

L.1 Brief Specification of CLEFIA

CLEFIA [51] is a 128-bit block cipher supporting 128-bit, 192-bit, and 256-bit keys. CLEFIA was designed by a team from Sony Corporation and published at FSE 2007 by Shirai et al. It is internationally standardized in ISO/IEC 29192-2. Depending on the key size, the number of rounds is 18 (128-bit key), 22 (192-bit key), or 26 (256-bit key). The round function of CLEFIA uses the generalized Feistel structure with four 32-bit branches in which two 32-bit functions F_0 and F_1 are applied in parallel (Figure 57). F_0 and F_1 follow the SP structure and perform three basic operations, including sub-key addition, application of four 8-bit S-boxes in parallel, and diffusing the output bytes of the S-box layer by applying a 4×4 MDS matrix over $\mathbb{F}_{2^8}$. CLEFIA employs two different S-boxes which are used in different order in F_0 and F_1. The diffusion layer was designed based on the new Diffusion Switching Mechanism technique [50, 51] to obtain a larger minimum number of active S-boxes. The Hadamard-type MDS matrices are specified in Figure 57b. The 8-bit S-boxes S_0, S_1 are specified as follows. For S_0, the 4-bit S-boxes SS_0, SS_1 (see Figure 57c) are applied to the input halves; then each half is updated by adding 2 times the other half; then SS_2 and SS_3 are applied. The other S-box S_1 is defined using modular inversion, like AES.



(a) Round function

$$M_0 = \begin{pmatrix} 01 & 02 & 04 & 06 \\ 02 & 01 & 06 & 04 \\ 04 & 06 & 01 & 02 \\ 06 & 04 & 02 & 01 \end{pmatrix}, \qquad M_1 = \begin{pmatrix} 01 & 08 & 02 & 0a \\ 08 & 01 & 0a & 02 \\ 02 & 0a & 01 & 08 \\ 0a & 02 & 08 & 01 \end{pmatrix}.$$

(b) Specification of CLEFIA's matrices

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$SS_0(x)$	e	6	c	a	8	7	2	f	b	1	4	0	5	9	d	3
$SS_1(x)$	b	8	5	e	a	6	4	c	f	7	2	3	1	0	d	9

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$SS_0(x)$	6	4	0	d	2	b	a	3	9	c	e	f	8	7	5	1
$SS_1(x)$	a	2	6	d	3	4	5	e	0	7	8	9	b	f	c	1

(c) Specification of CLEFIA's helper S-boxes for S_0

Fig. 57: CLEFIA round function.

L.2 The DL Distinguishers of CLEFIA

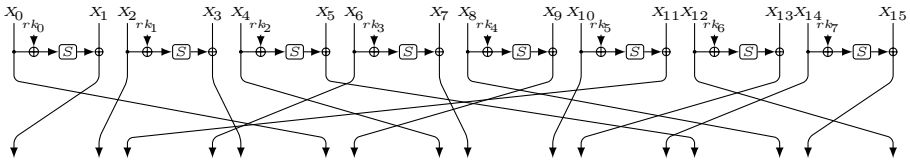
Table 34: DL distinguishers for 4 to 9 rounds of CLEFIA (with final permutation).

4 Rounds $r_0 = 0, r_m = 4, r_1 = 0,\ p = 1,\ r = 1, q = 1,\ prq^2 = \mathbf{1}$	
ΔX_0 00000000000000010000000000000000	ΓX_4 00000000010000000000000000000000
5 Rounds $r_0 = 0, r_m = 5, r_1 = 0,\ p = 1,\ r = 2^{-2.68}, q = 1,\ prq^2 = \mathbf{2^{-2.68}}$	
ΔX_0 00000000008000000000000000000000	ΓX_5 00000000381c8e920000000000000000
6 Rounds $r_0 = 0, r_m = 5, r_1 = 1,\ p = 1,\ r = 2^{-2.68}, q^2 = 2^{-4.39},\ prq^2 = \mathbf{2^{-7.07}}$	
ΔX_0 00000000008000000000000000000000 ΓX_5 00000000381c8e920000000000000000	- ΓX_6 381c8e9200000000000000000f5000000
7 Rounds $r_0 = 1, r_m = 5, r_1 = 1,\ p = 2^{-4.68}, r = 2^{-2.68},\ q^2 = 2^{-4.39},\ prq^2 = \mathbf{2^{-11.75}}$	
ΔX_0 00000000000000000080000d77e2bfc ΓX_6 00000000381c8e920000000000000000	ΔX_1 00000000008000000000000000000000 ΓX_7 381c8e9200000000000000000f5000000
8 Rounds $r_0 = 2, r_m = 5, r_1 = 1,\ p = 2^{-26.36}, r = 2^{-2.68},\ q^2 = 2^{-4.39},\ r = 2^{-13.95},\ prq^2 = 2^{-33.43}$	
ΔX_0 d77e2bfcbe919d960000000000080000 ΓX_7 00000000381c8e920000000000000000	ΔX_2 00000000008000000000000000000000 ΓX_8 381c8e9200000000000000000f5000000
9 Rounds $r_0 = 2, r_m = 5, r_1 = 2,\ p = 2^{-26.36}, r = 2^{-3},\ q^2 = 2^{-25.93},\ r = 2^{-13.95},\ prq^2 = 2^{-55.29}$	
ΔX_0 2bfcd77e9d96be910000000000000008 ΓX_7 00000000f11200000000000000000000	ΔX_2 00000000000000008000000000000000 ΓX_9 0000000000ae00ee69bffc00f1120000

M Application to TWINE

M.1 Brief Specification of TWINE

TWINE is a 64-bit block cipher designed by Suzaki et al. [54] which supports key sizes of 80 and 128 bits. This cipher uses a Type-2 generalized Feistel structure with 16 4-bit branches. Both variants apply 36 rounds of the round function illustrated in Figure 58. The round function includes a nonlinear layer consisting of 8 parallel applications of the same 4-bit S-box and a diffusion layer permuting the 16 nibbles.



(a) Round function.

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S(x)$	c	0	f	a	2	b	9	5	8	3	d	7	1	e	6	4

(b) 4-bit S-box S of TWINE.

Fig. 58: The round function of TWINE.

M.2 The DL Distinguishers of TWINE

Table 35: DDT of TWINE's S-box.

$\Delta_i \setminus \Delta_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	2	0	0	2	0	0	0	2	2	2	4	0	0	2
2	0	0	0	2	2	2	0	2	0	0	4	2	0	0	2	0
3	0	0	2	0	0	2	2	2	2	0	0	0	0	0	2	4
4	0	0	0	2	0	0	2	0	0	2	0	4	0	2	2	2
5	0	2	4	2	0	0	2	2	0	2	2	0	0	0	0	0
6	0	2	0	0	0	4	0	2	0	2	0	0	2	2	2	0
7	0	0	0	2	2	2	2	0	2	4	0	0	2	0	0	0
8	0	2	2	4	2	2	0	0	0	0	0	0	0	2	0	2
9	0	0	0	2	0	0	0	2	4	0	2	0	2	2	0	2
a	0	2	0	0	2	0	0	4	2	2	0	2	0	0	0	2
b	0	0	2	0	2	0	2	2	0	0	0	2	2	4	0	0
c	0	0	2	0	2	0	0	0	2	2	2	0	0	2	4	0
d	0	4	2	2	0	0	0	0	2	0	0	2	2	0	2	0
e	0	2	0	0	4	0	2	0	0	0	2	0	2	0	2	2
f	0	2	0	0	0	2	4	0	2	0	2	2	0	2	0	0

Table 36: LAT of TWINE's S-box (scale: $2^4 \cdot$ correlation).

$\lambda_i \setminus \lambda_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	4	-4	0	0	4	-4	-4	4	-8	-8	4	-4	0	0
2	0	4	0	4	8	-4	0	4	0	-4	0	-4	8	4	0	-4
3	0	4	4	0	0	4	-4	-8	4	-8	0	-4	-4	0	0	-4
4	0	0	0	8	0	8	0	0	-4	-4	-4	4	4	-4	4	4
5	0	0	4	4	-8	0	-4	4	-8	0	4	-4	0	0	-4	-4
6	0	4	8	4	0	-4	0	4	4	0	-4	0	-4	0	-4	8
7	0	4	-4	0	0	-4	4	0	-8	-4	-4	0	-8	4	4	0
8	0	0	0	0	4	4	-4	-4	-4	4	4	-4	0	8	0	8
9	0	0	-4	4	-4	-4	0	-8	0	0	-4	4	4	4	-8	0
a	0	4	0	4	-4	0	-4	0	4	8	-4	0	0	4	8	-4
b	0	4	-4	-8	-4	0	-8	4	0	-4	-4	0	4	0	0	4
c	0	8	0	0	-4	-4	4	-4	0	0	8	0	4	-4	4	4
d	0	-8	-4	4	-4	-4	0	0	4	-4	0	-8	0	0	4	4
e	0	-4	8	-4	-4	0	4	0	0	-4	0	4	4	8	4	0
f	0	-4	4	0	4	-8	-8	-4	-4	0	0	4	0	-4	4	0

Table 37: DLCT of TWINE's S-box.

$\Delta_i \setminus \lambda_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	0	0	0	0	0	-8	-8	-8	0	0	-8	0	8	0	8
2	16	0	-8	0	0	0	-8	0	0	-8	8	8	-8	0	0	0
3	16	0	-8	0	-8	8	8	0	0	0	0	-8	0	0	-8	0
4	16	-8	-8	0	0	-8	0	8	-8	8	0	0	0	0	0	0
5	16	0	-8	-8	8	0	0	-8	8	0	-8	0	0	0	0	0
6	16	-8	8	-8	-8	0	0	0	0	-8	0	0	0	8	0	0
7	16	0	8	0	0	-8	0	0	0	0	-8	0	-8	0	-8	8
8	16	-8	0	0	0	0	-8	0	8	0	0	0	8	-8	-8	0
9	16	0	0	8	0	8	0	0	-8	-8	-8	0	0	-8	0	0
a	16	-8	0	8	0	0	8	-8	0	0	0	0	-8	0	0	-8
b	16	0	0	0	-8	0	-8	0	0	8	-8	0	0	0	8	-8
c	16	8	0	-8	0	0	0	0	-8	0	0	8	0	0	-8	-8
d	16	0	0	0	8	-8	0	0	0	-8	0	-8	8	0	0	-8
e	16	8	0	0	-8	-8	0	-8	0	0	8	0	0	-8	0	0
f	16	0	0	-8	0	0	0	8	0	0	0	-8	-8	-8	8	0

Table 38: DDLCT of TWINE's S-box.

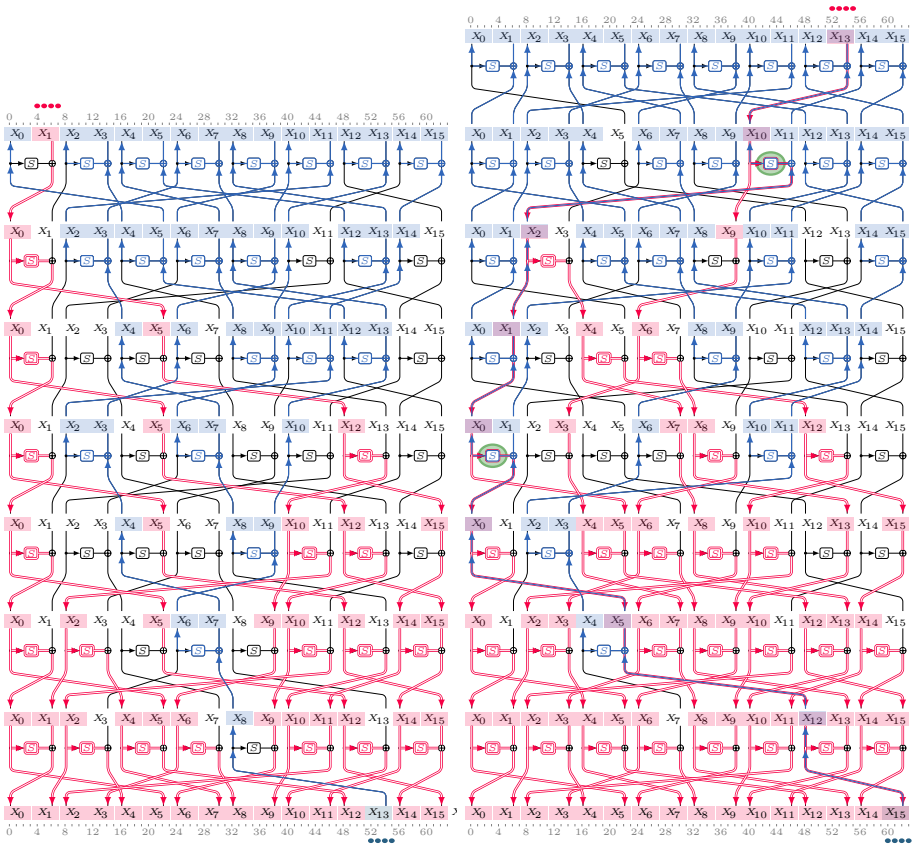
$\Delta_i \setminus \lambda_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	256	256	256	256	256	256	256	256	256	256	256	256	256	256	256	256
1	256	16	-32	-32	0	16	-16	-16	-32	-16	-32	32	-48	-32	0	-64
2	256	-32	-32	16	-32	-32	32	-48	0	32	-32	-16	-48	-16	-16	-32
3	256	-16	0	-64	-16	-32	-32	0	32	-32	0	-16	-48	-48	0	16
4	256	0	0	-16	-64	0	-16	0	-16	-16	-32	-48	0	-32	32	-48
5	256	-32	-16	16	-32	16	-16	-32	-32	-64	0	0	-64	16	-32	16
6	256	32	-16	-32	32	-32	-16	-64	-16	-32	-48	-16	0	-16	-32	0
7	256	-32	-32	-16	-16	32	0	0	-32	-32	-48	0	16	-32	-48	-16
8	256	-16	**-80**	-32	0	0	0	0	-16	-16	0	-64	-16	0	-16	0
9	256	-32	0	-16	0	-16	0	0	16	-16	-16	-32	0	-48	-64	-32
a	256	-32	16	0	-16	-32	-48	16	-32	16	-64	-32	-32	-32	-16	32
b	256	-16	0	-32	0	-64	-32	16	-32	-32	-16	0	0	16	-16	-48
c	256	-16	-32	32	-16	-48	-16	-32	-16	-32	32	0	0	-64	-16	-32
d	256	16	-32	-16	-48	0	-64	-48	-32	0	16	-16	0	0	-32	0
e	256	-32	-16	-32	-32	-48	0	0	-64	16	16	-16	-32	0	0	-16
f	256	-64	16	-32	-16	-16	-32	-48	16	-32	-32	-32	16	32	0	-32

Table 39: 3-DLCT of TWINE's S-box.

$\Delta_i \setminus \lambda_o$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	4096	4096	4096	4096	4096	4096	4096	4096	4096	4096	4096	4096	4096	4096	4096	4096
1	4096	-480	-160	32	-256	-480	-256	-320	-192	-320	-192	-224	-256	-320	-352	-320
2	4096	-384	-64	-288	-384	-320	-384	32	-416	-256	-416	-288	-384	-288	-192	-64
3	4096	-480	-320	-288	-224	-192	-128	-480	-256	-320	-352	-352	-224	32	-288	-224
4	4096	-256	-96	-512	-160	**-544**	-416	-256	-224	-320	-192	-256	-128	-96	-320	-320
5	4096	-256	-256	-256	-160	-256	-96	-320	-128	-96	**-544**	-192	-416	-480	-384	-256
6	4096	-288	-352	-96	-352	-64	-256	-320	-448	-416	-64	-64	-384	-288	-448	-256
7	4096	-224	-288	-256	-192	-256	-192	-256	-64	-448	-160	-416	-256	-480	-384	-224
8	4096	-256	-192	-384	-448	-160	-352	-384	-64	-320	-224	-224	-480	-256	-96	-256
9	4096	-352	-448	-320	-256	-192	-384	-224	-192	-288	-192	-448	-160	-288	-288	-64
a	4096	-352	-320	-384	-224	-32	-192	-96	-256	-384	-448	-288	-32	-256	-320	-512
b	4096	-64	-352	-160	-384	-288	-352	-448	-352	-224	-224	-224	-64	-288	-320	-352
c	4096	-320	-320	-256	-448	-352	-192	-160	-416	64	-192	-480	-320	-256	-224	-224
d	4096	-192	-448	-352	-192	-384	-160	-192	-320	-192	-128	-96	-480	-352	-128	-480
e	4096	-192	-128	-256	-352	-320	-320	-288	-352	-224	-384	-320	-192	-352	0	-416
f	4096	0	-352	-320	-64	-256	-416	-384	-416	-352	-384	-224	-320	-128	-352	-128

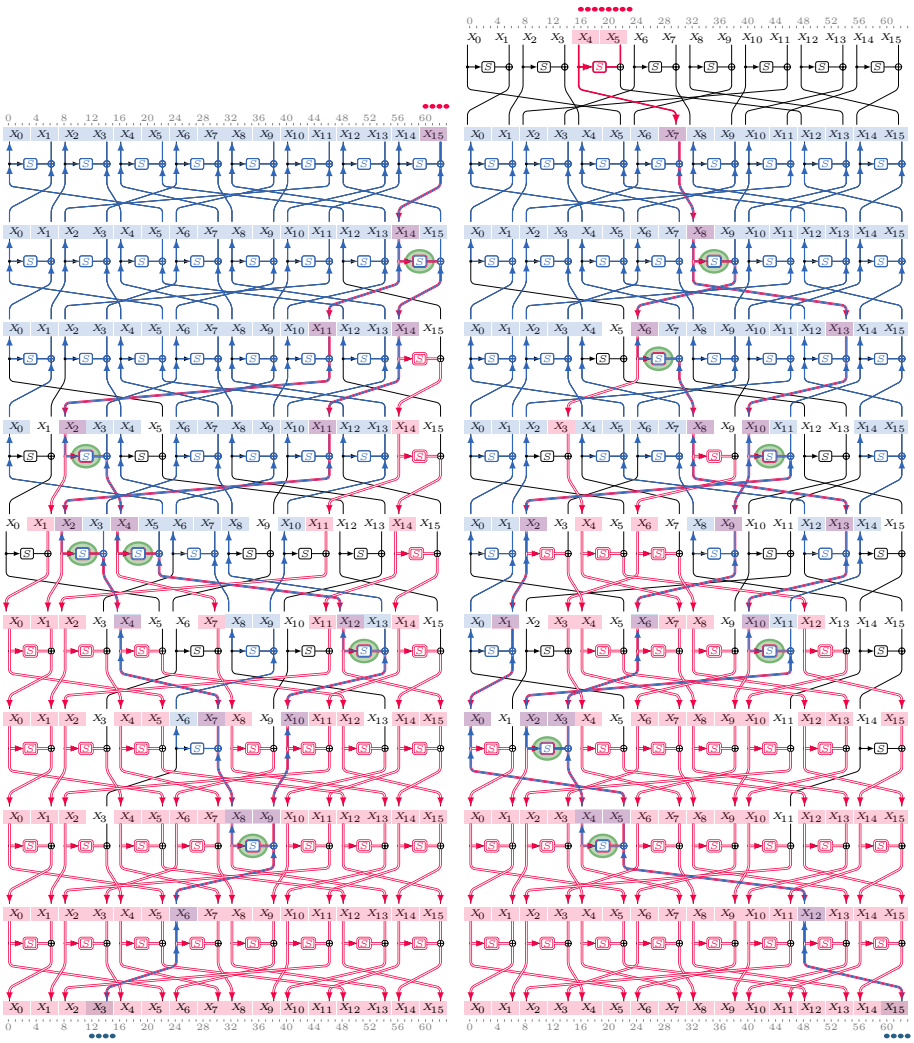
Table 40: Specification of DL distinguishers for TWINE.

7 Rounds, Figure 59a $\delta, \lambda \in \mathbb{F}_2^4,\ r_u = 0, r_m = 7, r_\ell = 0,\ p = 1,\ r = 1, q^2 = 1,\ prq^2 = 1$			
ΔX_0	0δ00000000000000	ΓX_7	0000000000000λ00
8 Rounds, Figure 59b $r_u = 0, r_m = 8, r_\ell = 0,\ p = 1,\ r = 2^{-1.68}, q^2 = 1,\ prq^2 = 2^{-1.68}$			
ΔX_0	0000000000000800	ΓX_8	0000000000000002
9 Rounds, Figure 60a $r_u = 0, r_m = 9, r_\ell = 0,\ p = 1,\ r = 2^{-5.82}, q^2 = 1,\ prq^2 = 2^{-5.82}$			
ΔX_0	0000000000000004	ΓX_9	0005000000000000
10 Rounds, Figure 60b $r_u = 1, r_m = 9, r_\ell = 0,\ p = 2^{-2},\ r = 2^{-5.85}, q^2 = 1,\ prq^2 = 2^{-7.85}$			
ΔX_0	0000a70000000000	ΔX_1	0000000a00000000
ΓX_{10}	0000000000000001		
11 Rounds, Figure 61a $r_u = 1, r_m = 9, r_\ell = 1,\ p = 2^{-2},\ r = 2^{-6.49}, q^2 = 2^{-2},\ prq^2 = 2^{-10.49}$			
ΔX_0	0000000000000a700	ΔX_1	000000000000000a
ΓX_{10}	00000a0000000000	ΓX_{11}	0000000010000a000
12 Rounds, Figure 61b $r_u = 1, r_m = 10, r_\ell = 1,\ p = 2^{-2},\ r = 2^{-7.49}, q^2 = 2^{-2},\ prq^2 = 2^{-11.49}$			
ΔX_0	0000a70000000000	ΓX_{11}	0a00000000000000
ΔX_1	0000000a00000000	ΓX_{12}	a000010000000000
13 Rounds, Figure 62a $r_u = 1, r_m = 10, r_\ell = 2,\ p = 2^{-2},\ r = 2^{-7.58}, q^2 = 2^{-4},\ prq^2 = 2^{-13.58}$			
ΔX_0	00a7000000000000	ΔX_1	0a00000000000000
ΓX_{11}	0000000a00000000	ΓX_{13}	0200c00000000a00
14 Rounds, Figure 62b $r_u = 2, r_m = 10, r_\ell = 2,\ p = 2^{-4},\ r = 2^{-7.64}, q^2 = 2^{-4},\ prq^2 = 2^{-15.64}$			
ΔX_0	0000000a00790000	ΔX_2	0000000000000a00
ΓX_{12}	00000000000a0000	ΓX_{14}	0a0000c000000200
15 Rounds, Figure 63a $r_u = 2, r_m = 10, r_\ell = 3,\ p = 2^{-4},\ r = 2^{-7.49}, q^2 = 2^{-8},\ prq^2 = 2^{-19.49}$			
ΔX_0	0000000000000a79	ΔX_2	000000000a000000
ΓX_{12}	000000000000000a	ΓX_{15}	01a000d001000800
16 Rounds, Figure 63b $r_u = 3, r_m = 10, r_\ell = 3,\ p = 2^{-8},\ r = 2^{-7.64}, q^2 = 2^{-8},\ prq^2 = 2^{-23.64}$			
ΔX_0	00000000a7009807	ΔX_3	000000000a000000
ΓX_{13}	000000000000000a	ΓX_{16}	01a000d001000800
17 Rounds, Figure 63c $r_u = 3, r_m = 10, r_\ell = 4,\ p = 2^{-8.00},\ r = 2^{-8}, q^2 = 2^{-14},\ prq^2 = 2^{-29.62}$			
ΔX_0	0000a70000000798	ΔX_3	0000000000000a00
ΓX_{13}	00000000000a0000	ΓX_{17}	0c001a0d0c061020



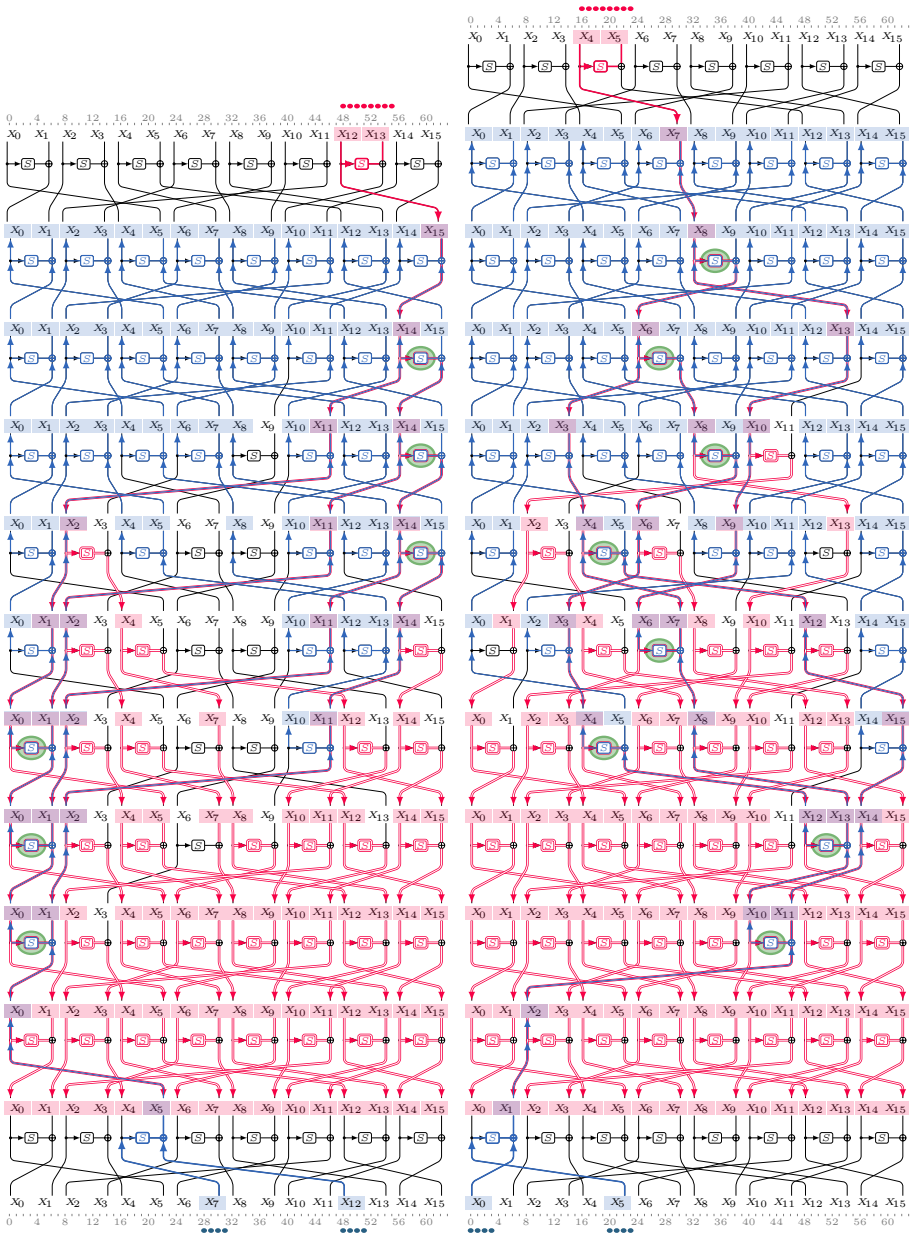
(a) 7-round TWINE. (b) 8-round TWINE.

Fig. 59: DL distinguishers for 7 to 8 rounds of TWINE (– differential – linear).



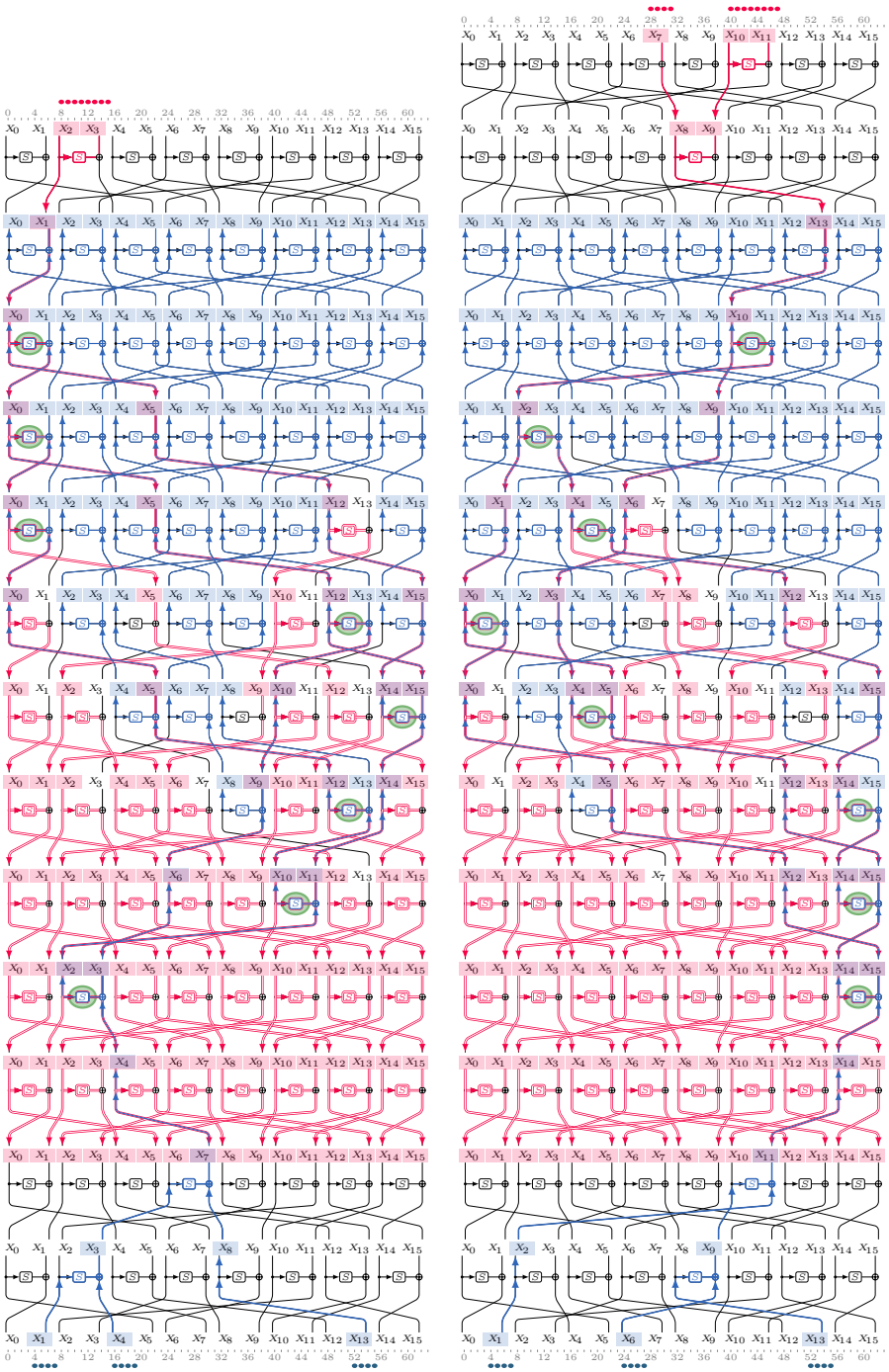
(a) 9-roundTWINE. (b) 10-round TWINE.

Fig. 60: DL distinguishers for 9 to 10 rounds of TWINE (– differential – linear).



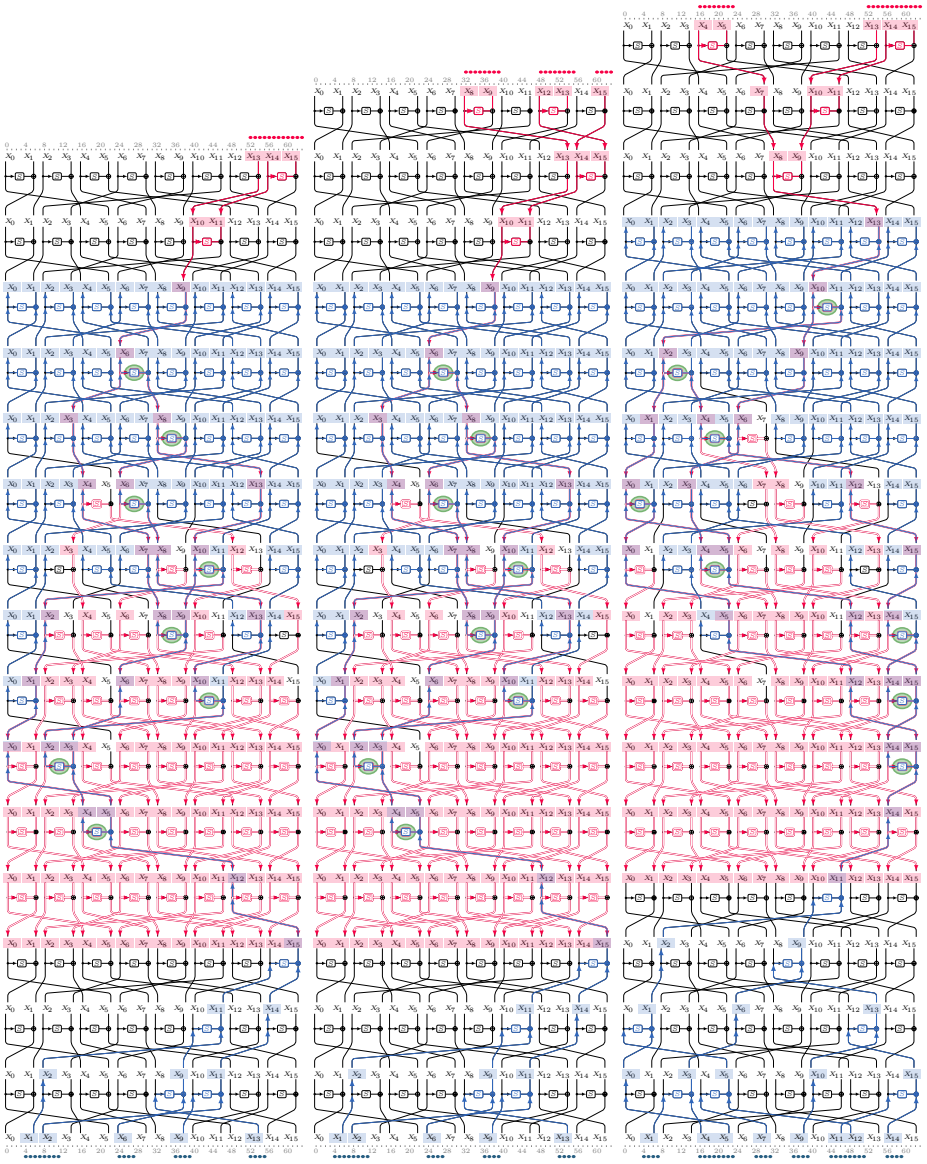
(a) 11-round TWINE. (b) 12-round TWINE.

Fig. 61: DL distinguishers for 11 to 12 rounds of TWINE (– differential – linear).



(a) 13-round TWINE. (b) 14-round TWINE.

Fig. 62: DL distinguishers for 13 to 14 rounds of TWINE (– differential – linear).



(a) 15-round TWINE. (b) 16-round TWINE. (c) 17-round TWINE.

Fig. 63: DL distinguishers for 15 to 17 rounds of TWINE (– differential – linear).

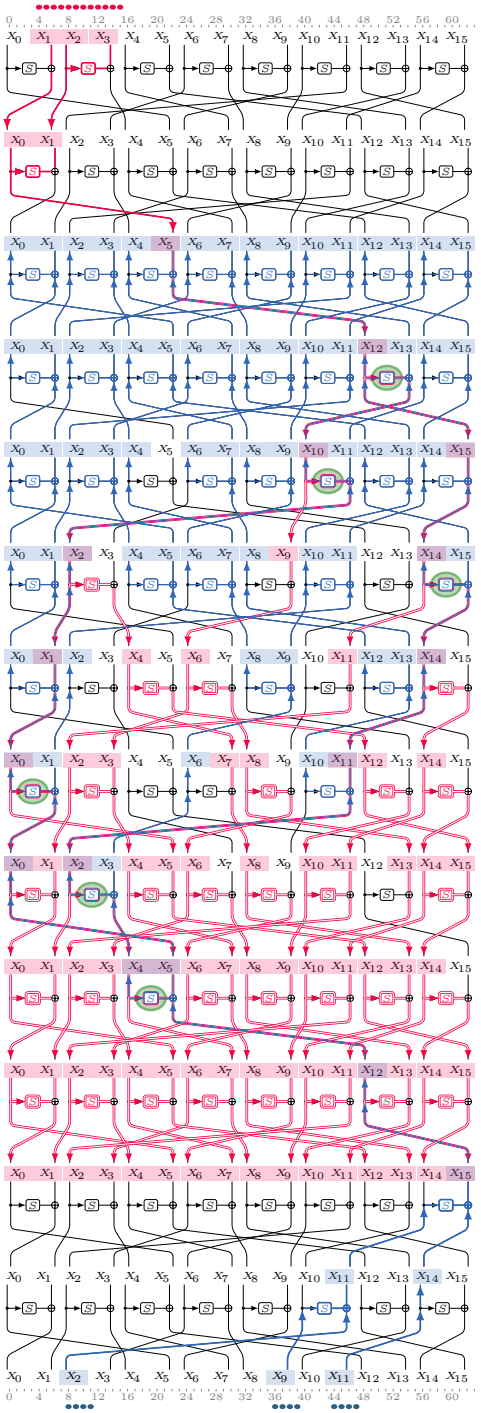


Fig. 64: The 13-round DL distinguisher of TWINE in Example 2 (— differential — linear). $p = 2^{-4}$, $r = 2^{-5.83}$ (for $\Delta X_2[5] = 4, \Gamma X_{11}[15] = 5$), $q^2 = 2^{-4}$, $prq^2 = 2^{-13.83}$.

N Encoding S-boxes

The S-box Analyzer [33], a SageMath [49] module, is an open-source tool designed to efficiently encode the differential, linear, and integral properties of S-boxes through MILP, SMT/SAT, and CP models. It has been applied for differential, integral, impossible-differential, zero-correlation, and boomerang attacks on various block ciphers, as detailed in [31–33]. The tool is openly available to the public at: https://github.com/hadipourh/sboxanalyzer

We also added some new features regarding differential-linear analysis to the S-box Analyzer, e.g., deriving the DLCT, DDLCT, and encoding the DLCT tables. Listing 1.1, and Listing 1.2 briefly demonstrates how to use the S-box Analyzer to derive the CP constraints modeling the DDT, $\texttt{LAT}^2$, deterministic differential/-linear behavior and also the DLCT of S-boxes.

```
sage: from sboxanalyzer import *
sage: from sage.crypto.sboxes import KNOT as sa
sage: sa = SboxAnalyzer(sa)


# Model the DDT
sage: cnf, milp = sa.minimized_diff_constraints()
Time used to simplify the constraints: 0.01 seconds
Number of constraints: 37
Input:  a0||a1||a2||a3; a0: msb
Output: b0||b1||b2||b3; b0: msb
Weight: 3.0000 p0 + 2.0000 p1

sage: pretty_print(milp)
['- p0 - p1 >= -1',
 '- a0 - a2 + b0 + p0 >= -1',
 'a0 - b0 - b1 + p0 >= -1',
 '- a3 - b0 - b1 + p0 >= -2',
 'a2 + b0 - b1 + p0 >= 0',
 'a0 - a2 + b1 + p0 >= 0',
 '- a2 - b0 - b3 + p0 >= -2',
 'a2 + b1 - b2 + p1 >= 0',
 '- a0 + b1 + b2 + p1 >= 0',
 '- a0 - a1 + a2 + a3 - b0 >= -2',
 'a0 - a1 + a2 + a3 + b0 >= 0',
 'a0 + a1 + a2 - a3 + b1 >= 0',
 'a0 - a1 - a2 - b1 - b2 >= -3',
 'a0 + a1 - a2 - b1 + b2 >= -1',
 'a0 + a2 + b1 + b2 - b3 >= 0',
 '- a0 - b0 + b1 - b2 + b3 >= -2',
 '- a1 - a3 + b0 + b1 + p0 >= -1',
 'b0 + b1 - b2 - b3 - p1 >= -2',
 '- a1 + a2 - a3 - b0 + p1 >= -2',
 '- a0 + a1 + a2 - b1 + p1 >= -1',
 'a1 + a2 + a3 - b1 + p1 >= 0',
 'a0 - a2 + b0 - b1 + p1 >= -1',
 '- a0 - a3 - b0 - b3 + p1 >= -3',
 '- b0 + b1 - b2 - b3 + p1 >= -2',
 '- b0 + b1 + b2 + b3 + p1 >= 0',
 'a1 + a3 + b0 - b1 + b2 - b3 >= -1',
 '- a0 - a2 + a3 - b0 - b1 + b3 >= -3',
 'a1 - a2 - a3 + b0 + b2 + b3 >= -1',
 'a1 + a2 + a3 + b1 + p0 - p1 >= 0',
```

```
 'a2 + b1 + b2 + b3 + p0 - p1 >= 0',
 'a0 + b0 + b1 + b3 - p0 + p1 >= 0',
 'a1 - a2 - a3 + b0 - b1 - b2 - b3 >= -4',
 '- a1 - a2 - a3 + b0 - b1 + b2 - b3 >= -4',
 'a1 - a2 + a3 + b0 - b1 - b2 + b3 >= -2',
 '- a1 - a2 + a3 + b0 - b1 + b2 + b3 >= -2',
 '- a1 - a2 - a3 + b0 - b1 - b2 + b3 >= -4',
 '- a1 - a2 + a3 + b0 - b1 - b2 - b3 >= -4']


# Model the squared LAT
sage: cnf, milp = sa.minimized_linear_constraints()
Time used to simplify the constraints: 0.01 seconds
Number of constraints: 34
Input:  a0||a1||a2||a3; a0: msb
Output: b0||b1||b2||b3; b0: msb
Weight: 4.0000 p0 + 2.0000 p1

sage: pretty_print(milp)
['- p0 - p1 >= -1',
 'a1 + a3 - p0 >= 0',
 'a1 - a3 - b2 + p0 >= -1',
 '- a0 - a1 - b3 + p0 >= -2',
 '- a1 + b2 - b3 + p0 >= -1',
 '- a0 + a1 + a3 + p1 >= 0',
 'a1 - a2 + b2 + p1 >= 0',
 'a1 - b1 + b2 + p1 >= 0',
 'a3 - b2 + b3 + p1 >= 0',
 '- a1 + b2 + b3 + p1 >= 0',
 '- a0 + a1 + a3 - b0 + b2 >= -1',
 'a1 + a2 - a3 + b1 + b2 >= 0',
 '- a0 + a1 - b0 + b2 + b3 >= -1',
 'a1 + a3 - b0 + b2 + b3 >= 0',
 '- a0 - a1 - b0 - b1 + p0 >= -3',
 '- a0 - a3 + b0 + b1 + p0 >= -1',
 '- a0 - a2 - b0 - b2 + p0 >= -3',
 '- a0 + a2 + b0 - b3 + p0 >= -1',
 'a0 + a3 + b0 - b3 + p0 >= 0',
 '- a1 - a2 + b1 - b3 + p0 >= -2',
 'a0 + a3 - b2 - b3 + p0 >= -1',
 'a0 - a3 + b0 + b3 + p0 >= 0',
 'a1 + a3 + b0 + b3 - p1 >= 0',
 'a3 + b0 + b2 + b3 - p1 >= 0',
 '- a1 - a3 - b2 - b3 + p1 >= -3',
 'a0 + a1 + a2 - b0 + b2 - b3 >= -1',
 'a0 - a2 - a3 - b0 + b1 + p0 >= -2',
 '- a1 + a2 - a3 - b1 - b2 + p0 >= -3',
 '- a2 - a3 - b1 + b2 - b3 + p0 >= -3',
 '- a1 - a3 - b0 - b1 + b3 + p0 >= -3',
 'a0 - a1 - b0 + b1 + b3 + p0 >= -1',
 'a0 + a2 - b0 - b2 + b3 + p0 >= -1',
 '- a1 - a2 - b1 - b2 + b3 + p0 >= -3',
 '- a1 + a2 + b1 - b2 + b3 + p0 >= -1']


# Model the deterministic differential behavior
sage: detdiff = sa.encode_deterministic_differential_behavior()
sage: cpdetdiff = sa.generate_cp_constraints(detdiff)
Input:  a0||a1||a2||a3; a0: msb
Output: b0||b1||b2||b3; b0: msb
```

```
sage: print(cpdetdiff)
if (a0==0/\a1==0/\a2==0/\a3==0) then (b0=0/\b1=0/\b2=0/\b3=0)
elseif (a0==0/\a1==0/\a2==0/\a3==1) then (b0=-1/\b1=1/\b2=-1/\b3=-1)
elseif (a0==0/\a1==1/\a2==0/\a3==0) then (b0=1/\b1=-1/\b2=-1/\b3=-1)
elseif (a0==1/\a1==0/\a2==0/\a3==0) then (b0=1/\b1=1/\b2=-1/\b3=-1)
elseif (a0==1/\a1==0/\a2==0/\a3==1) then (b0=-1/\b1=0/\b2=-1/\b3=-1)
elseif (a0==1/\a1==1/\a2==0/\a3==0) then (b0=0/\b1=-1/\b2=-1/\b3=-1)
else (b0=-1/\b1=-1/\b2=-1/\b3=-1)
endif


# Model the deterministic linear behavior
sage: detlin = sa.encode_deterministic_linear_behavior()
sage: cpdetlin = sa.generate_cp_constraints(detlin)
Input:  a0||a1||a2||a3; a0: msb
Output: b0||b1||b2||b3; b0: msb
sage: print(cpdetlin)
if (a0==0/\a1==0/\a2==0/\a3==0) then (b0=0/\b1=0/\b2=0/\b3=0)
elseif (a0==0/\a1==0/\a2==1/\a3==0) then (b0=1/\b1=-1/\b2=-1/\b3=-1)
elseif (a0==1/\a1==0/\a2==0/\a3==0) then (b0=1/\b1=-1/\b2=1/\b3=-1)
elseif (a0==1/\a1==0/\a2==1/\a3==0) then (b0=0/\b1=-1/\b2=-1/\b3=1)
else (b0=-1/\b1=-1/\b2=-1/\b3=-1)
endif
```

Listing 1.1: Encoding differential-linear behavior of S-boxes in S-box Analyzer

```
sage: dlct = sa.differential_linear_connectivity_table()
sage: udlct = sa.upper_differential_linear_connectivity_table()
sage: ldlct = sa.lower_differential_linear_connectivity_table()
sage: ddlct = sa.double_differential_linear_connectivity_table()
sage: sa.print_table(dlct)
16 16 16 16  16 16 16 16  16 16 16 16  16 16 16 16
16  0  0  0 -16  0  0  0   0  0  0  0   0  0  0  0
16 -8 -8  0   0  0  8 -8   0 -8  0  8   0  0  0  0
16  0 -8 -8   0 -8  8  0   0  0  0  0   0 -8  0  8
16  0 -8  0   0  0 -8  0 -16  0  8  0   0  0  8  0
16  0 -8  0   0  0 -8  0   0  0  8  0 -16  0  8  0
16 -8  8 -8   0  0 -8  0   0 -8  0  0   0  0  0  8
16  0  8  0   0 -8 -8 -8   0  0  0  8   0 -8  0  0
16  0  0  0 -16  0  0  0 -16  0  0  0  16  0  0  0
16 -8  0 -8  16 -8  0 -8   0  8  0 -8   0  8  0 -8
16  0  0  8   0  8  0  0   0  0 -8  0   0 -8 -8 -8
16  8  0  0   0  0  0  8   0 -8 -8 -8   0  0 -8  0
16  0  0 -8   0  0  0 -8  16  0  0 -8   0  0  0 -8
16 -8  0  0   0 -8  0  0   0  8  0  0 -16  8  0  0
16  0  0  0   0  8  0  8   0  0 -8 -8   0 -8 -8  0
16  8  0  8   0  0  0  0   0 -8 -8  0   0  0 -8 -8
# Model the DLCT
sage: cnf,milp=sa.minimized_differential_linear_constraints(subtable='star')
Number of constraints: 34, a0: msb, b0: msb
sage: pretty_print(milp)
['- a2 - b0 + b2 + b3 >= -1',
 '- a1 + a2 + a3 - b1 + b2 >= -1',
 '- a0 + a1 + a2 + a3 - b3 >= -1',
 '- a1 + a2 - a3 - b2 - b3 >= -3',
 'a1 + a2 - a3 - b2 + b3 >= -1',
 'a1 - a3 - b0 + b2 + b3 >= -1',
 '- a3 - b0 + b1 + b2 + b3 >= -1',
 '- a0 - a1 - a3 + b0 - b1 - b2 >= -4',
 '- a0 - a1 - a2 + a3 + b0 + b1 - b3 >= -3',
 'a0 - a1 - a3 - b0 - b1 - b2 - b3 >= -5',
 'a0 - a1 + a3 - b0 + b1 - b2 - b3 >= -3',
 '- a1 + a2 - a3 + b0 - b1 + b2 + b3 >= -2',
 'a0 + a1 - a2 + a3 + b0 + b1 - b2 - b3 >= -2',
 '- a0 - a1 - a2 + a3 - b0 - b1 - b2 - b3 >= -6',
 'a0 - a3 + b1 + b2 - b3 >= -1',
 'a0 + a1 - a3 + b0 - b1 - b2 - b3 >= -3',
 '- a0 - a1 - a3 - b0 + b1 - b2 - b3 >= -5',
 '- a0 + a1 - a2 - a3 + b0 + b1 - b2 >= -3',
 '- a0 + a1 - a2 - a3 - b0 - b1 - b3 >= -5',
 '- a0 + a1 + a3 + b0 - b1 - b2 >= -2',
 'a0 + a2 - a3 - b3 >= -1',
 '- a0 + a3 + b1 + b2 - b3 >= -1',
 'a0 - a1 - a3 + b0 + b1 - b3 >= -2',
 'a0 - a1 + a3 + b0 - b1 - b3 >= -2',
 '- a0 + b0 - b2 + b3 >= -1',
 '- a0 + a1 + a3 - b0 + b1 - b3 >= -2',
 'a0 + a1 - a2 + a3 - b0 - b1 >= -2',
 '- a0 - a2 - a3 - b1 + b2 >= -3',
 '- a2 - b1 + b2 + b3 >= -1',
 'a0 - a2 - b0 + b3 >= -1',
 'a0 + a1 - a3 - b0 + b1 >= -1',
 '- a0 + a2 - b2 + b3 >= -1',
 'a0 - a2 + a3 - b1 + b2 >= -1',
 'a0 - a1 + a2 - b3 >= -1']
```

Listing 1.2: Encoding differential-linear behavior of S-boxes in S-box Analyzer