A Probabilistic Extension of the Division Property

Integral Distinguishers with Key-Independent Bounds

Jules Baudrin[1], Christina Boura[2,5], Patrick Derbez[3], Gregor Leander[4], and Yevhen Perehuda[4]

[1] Université Paris-Saclay, UVSQ, CNRS, Laboratoire de mathématiques de Versailles, 78000, Versailles, France
jules.baudrin@uvsq.fr
[2] Université Paris Cité, CNRS, IRIF, Paris, France
christina.boura@irif.fr
[3] Univ Rennes, Inria, CNRS, IRISA, Rennes, France
patrick.derbez@inria.fr
[4] Faculty of Computer Science, Ruhr University Bochum, Bochum, Germany
{gregor.leander,yevhen.perehuda}@rub.de
[5] Institut universitaire de France (IUF), Paris, France

Abstract. Classical integral cryptanalysis relies on zero-sum properties that, for every key, hold with probability one. Existing arguments against such deterministic integrals, based on the division property and its algebraic variants, only exclude this extreme case: once the zero-sum probability falls below one, they say nothing about its actual value, nor about a probabilistic bias that could still be exploited. We develop a tool combining MILP-based trail search with a recursive evaluation of assignments to the constant inputs. It estimates the probability that no valid trail remains and uses an additional heuristic to predict the zero-sum probability. Applying this tool to PRESENT and GIFT, we obtain probabilistic integral distinguishers for 6 to 9 rounds, including 6- and 7-round PRESENT distinguishers using 2^4 and 2^{12} plaintexts per structure, respectively. For the ASCON permutation, we obtain 4- and 5-round distinguishers using as few as 2 and 2^{12} inputs per structure. We further explain algebraically the large bias of the 6-round PRESENT distinguisher, showing how the S-box and bit permutation restrict the way active variables combine. Finally, we show how these distinguishers can be used for key recovery, reducing both the data and time complexities compared to their deterministic variants for PRESENT and GIFT-64.

Keywords: Integral attacks · Division Property · Probabilistic distinguishers · PRESENT · GIFT.

1 Introduction

Integral cryptanalysis originated with the attack on SQUARE and was subsequently developed into a general technique by Knudsen and Wagner [13,23].

Rather than considering one or a few encryptions individually, an integral distinguisher analyzes the collective behavior of the ciphertexts obtained from related plaintexts. In its traditional form, it exploits the fact that the XOR-sum of a selected output component over a structured set of plaintexts is always zero.

More precisely, let $E : \mathbb{F}_2^n \times \mathbb{F}_2^\kappa \to \mathbb{F}_2^n$ be a keyed permutation, let $X \subseteq \mathbb{F}_2^n$ be a d-dimensional linear subspace, and let $\beta \in \mathbb{F}_2^n \setminus \{0\}$ be an output mask. For an offset $a \in \mathbb{F}_2^n$ and a key $k \in \mathbb{F}_2^\kappa$, define

$$\mathcal{I}_{X,\beta}(a,k) := \bigoplus_{x \in a+X} \beta \cdot E_k(x). \tag{1}$$

A classical zero-sum integral guarantees that $\mathcal{I}_{X,\beta}(a,k) = 0$ for every key and every offset in the family under consideration. Thus, the usual integral property holds with probability one: it is an algebraic identity (the polynomial $\mathcal{I}_{U,\beta}$ is identically zero) and does not depend on the distribution of the key.

Integral properties are linked to coefficients in the algebraic normal form (ANF) and to the propagation of monomials through the cipher, as described by parity sets and monomial prediction [11,22]. This algebraic viewpoint has led to a mature toolbox for finding deterministic integrals. Todo's division property tracks which products of state coordinates may have nonzero sums, while its bit-based version describes their propagation through the cipher more precisely [29,31]. Parity sets and monomial prediction provide equivalent or complementary descriptions of the underlying ANF behavior, and MILP models make it possible to automate the search for division or monomial trails [11,22,32]. These methods, as well as their mathematical foundations and limitations, are by now well studied.

More recently, algebraic transition matrices have been used to express monomial propagation as matrix multiplication and combine monomial trails and parity sets within a common framework [7]. Ultrametric integral cryptanalysis further lifts this framework from parity to 2-adic divisibility properties, thereby interpolating between zero sums and saturation properties [8]. A complementary line of work derives security properties by studying the coefficient functions in the round keys. In particular, Hebborn et al. [18] and Beierle et al. [3] use linear-independence arguments to prove, under the independent-round-key assumption, that every relevant ciphertext sum is key dependent after sufficiently many rounds.

The key-independence results provide a structural guarantee: they rule out a zero-sum identity of the form $\mathcal{I}_{X,\beta} \equiv 0$ that holds for every key. But, their binary nature is also their limitation. Once the identity fails, they do not quantify how frequently it fails. A function that is nonzero for only a tiny fraction of its inputs and a balanced function are both simply nonzero from this perspective. Similarly, the existing applications of algebraic and ultrametric transition matrices concern exact coefficients or divisibility of occurrence counts and they do not directly quantify the offset-wise probability considered here.

To formulate the probabilistic question, we consider the case where the key is fixed. We define

$$p_k(X, \beta) := \Pr_{a+X \leftarrow \mathbb{F}_2^n / X} \big[\mathcal{I}_{X,\beta}(a, k) = 0\big], \tag{2}$$

where the affine coset $a + X$ is sampled uniformly. For an ideal permutation and a proper low-data input space, this probability is essentially $1/2$. A deterministic integral satisfies $p_k(X, \beta) = 1$ for every key, whereas a probabilistic integral has

$$p_k(X, \beta) = \frac{1}{2} + \varepsilon_k$$

for a non-negligible positive bias ε_k. The setting of interest here requires this bias to persist for every fixed key, or at least to remain essentially uniform across the key space. A proof that excludes deterministic integrals only excludes the endpoint $p_k = 1$. It gives no bound showing that p_k is close to $1/2$.

Probabilistic integral behavior has appeared previously in several forms. Heys explicitly studied probabilistic byte-valued integrals of the `BSPN` block cipher by comparing the observed frequency of a zero sum with the probability that a uniformly random byte is zero [20]. Cube testers also use statistical properties of cube sums to detect nonrandom algebraic behavior [1]. More recently, Zhang et al. derived probabilistic integral properties whose validity depends on the key and used them in attacks that apply only to a subset of keys [33]. These precedents make the underlying probability space important. More precisely, a property that happens with noticeable probability when the key is sampled is a key-averaged or weak-key property. In contrast, Eq. (2) samples only the public offset after fixing the unknown key. The two notions are not interchangeable.

Probabilistic integrals are also closely related to statistical saturation attacks and higher-order differential-linear cryptanalysis. Statistical saturation attacks distinguish a cipher through a non-uniform distribution of selected output values [12]. Their relationship with linear hulls and multidimensional linear cryptanalysis was clarified in [25]. The statistic considered here is different: it is not the marginal distribution of an output projection, but the parity of that projection over an entire affine space.

The relation to higher-order differential–linear cryptanalysis is direct. Writing $F_k(x) = \beta \cdot E_k(x)$, Equation (1) gives $\mathcal{I}_{X,\beta}(a, k) = D_{u_1} \cdots D_{u_d} F_k(a)$, where $D_u F(x) = F(x) \oplus F(x + u)$. A probabilistic integral with a linear output mask is therefore a higher-order differential-linear approximation in the sense introduced by Biham, Dunkelman, and Keller [9]. Accurately evaluating such approximations is known to require dedicated algebraic techniques [21]. This difficulty is natural: the integral, or derivative, combines 2^d evaluations, whose dependencies cannot in general be reconstructed from individual or pairwise behavior. Large biases in individual states or pairs may disappear in their joint parity, while an apparently-ideal behavior may coexist with a maximally-biased parity.

This observation also explains why pairwise independence does not provide an adequate security argument. If $m = 2^d$, then the bias of the integral depends on an m-point moment. For example, consider an m-bit vector chosen uniformly

among all vectors with even parity. Any $m-1$ of its coordinates are independent uniform bits, yet the XOR of all m coordinates is always zero. Thus, in the worst case, no t-wise guarantee with $t < m$ controls the integral parity. Existing work establishes meaningful pairwise-independence results for concrete or AES-like constructions and higher-t results for more idealized SPNs, notably when the S-boxes are chosen randomly and kept secret [27,26,6]. These results do not yet reach the order $t = 2^d$ relevant to practical integral structures in concrete ciphers. Moreover, such guarantees usually concern the joint distribution obtained by sampling the key, whereas $p_k(X, \beta)$ is a fixed-key quantity.

Consequently, current deterministic and limited-independence security arguments leave open a substantial gap. A useful probabilistic integral may result from a collective dependency among many cipher evaluations that is invisible to every pairwise test and is discarded by a purely zero-versus-nonzero algebraic analysis. If such dependencies produce large, key-uniform biases, they can support low-data distinguishers and potentially-powerful key-recovery attacks. Understanding their magnitude, their structural origin, and their behavior across keys is therefore necessary for a more complete evaluation of resistance to integral cryptanalysis.

Our contributions. In this paper, we introduce a probabilistic extension of the division property to derive a key-independent no-trail bound and use it to predict the bias for a fixed secret key. We also explain algebraically one of the bias observed for the block cipher PRESENT and study applications to key recovery. Our main contributions are as follows.

- *A probabilistic extension of the division property.* We extend the division property to describe how the values of the constant inputs affect the validity of division trails. This allows us to study the probability that no valid trail reaches the selected output in our propagation model. Since the absence of a valid trail guarantees a zero sum, this probability provides a lower bound on the zero-sum probability. We further prove that the bound produced by our model is independent of the secret key and therefore applies to every key.
- *An algorithm and its applications.* We develop an algorithm that searches for probabilistic integral distinguishers and estimates their zero-sum probabilities through a recursive computation of the no-trail probability. For PRESENT and GIFT, we find 6- and 7-round distinguishers with low data complexities, 8-round distinguishers with higher but practical data complexities, and 9-round distinguishers requiring more data. We also obtain four- and five-round distinguishers for the ASCON permutation using 2 and 2^{12} inputs per structure, respectively. This application shows that the technique also applies to rotation-and-XOR diffusion. These results use smaller integral structures than the corresponding probability-one configurations considered and are reported in Table 1.
- *An algebraic explanation of the* PRESENT *bias.* We give an algebraic explanation of the six-round PRESENT distinguisher that uses only 2^4 chosen plaintexts per structure. We show how the S-box and its interaction with the

Table 1. Summary of our probabilistic integral distinguishers. $\widehat{p}$ is the heuristic prediction from the tool in Section 3, and $\overline{p}_{\text{emp}}$ is the experimental mean. For `PRESENT`, we show the results for `PRESENT-80`. Those for `PRESENT-128` are essentially identical (see Appendix H). A † means we obtained a certified lower bound on the key-independent probability that there is no trail and this bound is strictly greater than 50%.

PRESENT				GIFT-64				GIFT-128				ASCON			
R	D	$\widehat{p}$	$\overline{p}_{\text{emp}}$	R	D	$\widehat{p}$	$\overline{p}_{\text{emp}}$	R	D	$\widehat{p}$	$\overline{p}_{\text{emp}}$	R	D	$\widehat{p}$	$\overline{p}_{\text{emp}}$
6	2^3	**0.5181**	0.5411	6	2^4	**0.5938**	0.7270	6	2^3	**0.7182**	0.7610	4	2^1	**0.9063**†	0.9373
6	2^4	**0.9158**†	0.8945	6	2^6	**0.5882**	0.6527	6	2^4	**0.9840**†	0.9720	5	2^{12}	**0.5602**	0.5959
6	2^5	**0.8534**†	0.8974	6	2^7	**0.6818**	0.7146	7	2^{10}	**0.7161**	0.7924	5	2^{13}	**0.6973**	0.7467
7	2^{11}	**0.5031**	0.5146	6	2^8	**0.9208**†	0.9239	7	2^{11}	**0.8218**	0.9372				
7	2^{12}	**0.8130**†	0.8570	7	2^{14}	**0.6315**	0.7268	7	2^{12}	**0.9861**†	0.9889				
7	2^{13}	**0.9780**†	0.9882	7	2^{15}	**0.8667**†	0.9406	8	2^{28}	**0.8898**	0.9574				
8	2^{32}	**0.5038**	0.5072	7	2^{16}	**0.9997**†	0.9999	9	2^{60}	**0.8317**	–				
9	2^{56}	**0.5020**	–	8	2^{32}	**0.5031**	0.5418								

linear layer prevent certain monomials from appearing in intermediate expressions. These restrictions limit the ways in which the monomial whose coefficient equals the integral sum can be formed, helping to explain the large observed bias.

– *Applications to partial key recovery.* Finally, we show how probabilistic integral distinguishers can be used to filter key candidates. For `PRESENT` and `GIFT-64`, our analysis shows that these distinguishers can reduce both data and time complexities compared with the corresponding deterministic ones.

The rest of the paper is organized as follows. Section 2 introduces the notation and recalls the connections between integral sums, ANF coefficients, and division trails, and Section 3 presents our probabilistic framework and search tool. Section 4 describes the applications and the algebraic analysis of the `PRESENT` distinguisher. Finally, Section 5 discusses key recovery.

2 Preliminaries

Throughout the paper, the coordinates of vectors in $\mathbb{F}_2^n$ are indexed from 0 to $n-1$. As an ordered tuple, a vector u is written as $(u_0, \ldots, u_{n-1})$. When displayed as a binary string, it is typeset in typewriter font and written as $\mathtt{u_{n-1} \cdots u_1 u_0}$. Thus, coordinate 0 appears on the right in the binary-string representation and corresponds to the least significant bit.

2.1 Monomials and algebraic normal forms

For $x = (x_0, \ldots, x_{n-1}) \in \mathbb{F}_2^n$ and $u \in \mathbb{F}_2^n$, we denote by $x^u := \prod_{i=0}^{n-1} x_i^{u_i}$ the monomial associated with u. The *support* and *Hamming weight* of u are denoted by $\mathrm{supp}(u) := \{i : u_i = 1\}$, and $\mathrm{wt}(u) := |\mathrm{supp}(u)|$. For $u, v \in \mathbb{F}_2^n$, we write $u \preceq v$ if $u_i \leq v_i$ for every i, or equivalently if the monomial x^u divides x^v.

Every Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ admits a unique representation as a multivariate polynomial of $\mathbb{F}_2[x_0, \ldots, x_{n-1}]/(x_0^2 + x_0, \ldots, x_{n-1}^2 + x_{n-1})$, called its *algebraic normal form* (ANF): $f(x) = \bigoplus_{u \in \mathbb{F}_2^n} \lambda_u x^u$, where $\lambda_u \in \mathbb{F}_2$ is the coefficient of the monomial x^u. The *algebraic degree* of a nonzero Boolean function f is $\deg(f) := \max\{\mathrm{wt}(u) : \lambda_u = 1\}$.

Let $F : \mathbb{F}_2^n \longrightarrow \mathbb{F}_2^m$ be a vectorial Boolean function with coordinate functions $F_1, \ldots, F_m : \mathbb{F}_2^n \to \mathbb{F}_2$. For an output monomial mask $v \in \mathbb{F}_2^m$, we define $F(x)^v := \prod_{i=0}^{m-1} F_i(x)^{v_i}$. Its ANF is written as

$$F(x)^v = \bigoplus_{u \in \mathbb{F}_2^n} \lambda_{v,u} x^u, \qquad \lambda_{v,u} \in \mathbb{F}_2.$$

When $v = e_i$, where e_i denotes the i-th unit vector, $F(x)^v = F_i(x)$ is a single coordinate function.

For a linear output mask $\beta \in \mathbb{F}_2^m$, we write $\beta \cdot F(x) := \bigoplus_{i=0}^{m-1} \beta_i F_i(x)$, where $\cdot$ denotes the standard inner product over $\mathbb{F}_2$; this is a linear combination of coordinate functions, whereas $F(x)^v$ is their product.

2.2 Integral distinguishers and division property

Integral cryptanalysis studies XOR-sums of Boolean functions over structured input sets. The following standard identity relates such sums to ANF coefficients.

Lemma 1 (Möbius inversion formula). *Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ have ANF coefficients λ_u as in Section 2.1. For every $u \in \mathbb{F}_2^n$,*

$$\lambda_u = \bigoplus_{x \preceq u} f(x).$$

Consequently, this sum is zero if and only if x^u does not appear in the ANF of f.

The division property was introduced by Todo as an algorithmic tool for finding and propagating integral properties, thereby generalizing classical integral cryptanalysis [29]. Todo and Morii later formulated the division property at the bit level, allowing integral properties to be propagated more precisely through the Boolean operations of the analyzed primitive [31]. Division trails were introduced by Xiang et al. in 2016 to describe propagation through successive operations of a primitive as transitions between their input and output division-property vectors, thereby facilitating automated searches for integral distinguishers [32]. In the following, we recall only the aspects of the division property that are relevant to our work. Throughout this work, we mainly use the parity-set formulation [11,18].

Definition 1 (Parity sets and exact propagation). *For a finite multiset X over $\mathbb{F}_2^n$, its* parity set *is*

$$\mathcal{U}(X) := \left\{ u \in \mathbb{F}_2^n : \bigoplus_{x \in X} x^u = 1 \right\}.$$

For any $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ and any coefficient $\lambda_{v,u}$ (of x^u in $F(x)^v$), we say that there is an exact propagation *(from u to v) if $\lambda_{v,u} = 1$ and denote it by $u \xrightarrow{F} v$.*

Thus, $\mathcal{U}(X)$ records the masks of monomials with nonzero sums over X. Exact propagation uses the ANF coefficients of $F(x)^v$ as in monomial prediction [22]. For the cube $\{x \in \mathbb{F}_2^n : x \preceq u\}$, only x^u has a nonzero sum. Its parity set is therefore $\{u\}$, so propagation starts from the mask u.

Most modern ciphers can be decomposed into simple vectorial Boolean functions. We first consider a general composition $F = F^{(r)} \circ F^{(r-1)} \circ \cdots \circ F^{(1)}$, where r is the number of components and their input and output dimensions are compatible.

Definition 2 (Exact division trail). *An* exact division trail *from u to v through this decomposition is a sequence $(u^{(0)}, \ldots, u^{(r)})$, with $u^{(0)} = u$ and $u^{(r)} = v$, such that*

$$u^{(0)} \xrightarrow{F^{(1)}} u^{(1)} \xrightarrow{F^{(2)}} \cdots \xrightarrow{F^{(r)}} u^{(r)}.$$

Each mask has the dimension of the corresponding intermediate state.

Lemma 2 (Exact propagation & trail parity [18,22]). *For F, X as above, let $F(X)$ be the image multiset, retaining multiplicities. Then:*

$$\mathcal{U}(F(X)) = \left\{ v \in \mathbb{F}_2^m : \bigoplus_{u \in \mathcal{U}(X)} \lambda_{v,u} = 1 \right\},$$

For every input mask u and output mask v,

$$\lambda_{v,u} \equiv |\{\textit{exact division trails from } u \textit{ to } v\}| \pmod 2.$$

In particular, if no such trail exists, then x^u does not occur in $F(x)^v$.

Note in particular that the existence of a trail alone does not imply that the monomial occurs, since a positive even number of trails cancels.

We now apply these statements to a block cipher $E : \mathbb{F}_2^n \times \mathbb{F}_2^\kappa \to \mathbb{F}_2^n$. Its inputs are the plaintext $x \in \mathbb{F}_2^n$ and master key $k \in \mathbb{F}_2^\kappa$. For each fixed k, the map $E_k : x \longmapsto E(x, k)$ is a permutation. A decomposition of E includes both the encryption operations and the actual key schedule.

Let $u, v \in \mathbb{F}_2^n$ be an input and output monomial masks. An input *cube* (associated to u) is an affine subspace consisting of the plaintexts $a \oplus z$ with $z \preceq u$, where the offset $a \in \mathbb{F}_2^n$ satisfies $a_i = 0$ whenever $u_i = 1$. The coordinates a_i with $u_i = 0$ and the key bits remain symbolic. Let $\lambda_u(a, k)$ denote the coefficient of z^u in $E(a \oplus z, k)^v$, viewed as a polynomial in z. This coefficient is a Boolean function of a and k. By Lemma 1, its value at each assignment of a, k is the corresponding cube sum: $\lambda_u(a, k) = \bigoplus_{z \preceq u} E(a \oplus z, k)^v$.

In this paper, we only consider ciphers for which the round keys are added to the state through a key addition $x \mapsto x \oplus k^{(i)}$ only, just as the public offsets

a. The following lemma shows that, if the ANF contains no monomial $x^{u'}$ with $u' \succeq u$, then x^u remains absent after any XOR shift. This conclusion therefore holds regardless of the round-key values, without needing to know them.

Lemma 3 (Unknown shift). *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, $u \in \mathbb{F}_2^n$, $v \in \mathbb{F}_2^m$. If $\lambda_{v,u'} = 0$ for every $u' \succeq u$, then for any $t \in \mathbb{F}_2^n$, the coefficient of x^u in $F(x \oplus t)^v$ is 0.*

Proof. Write $F(x)^v = \bigoplus_{u'} \lambda_{v,u'} x^{u'}$. By substituting $x = y \oplus t$ and expanding $(y \oplus t)^{u'} = \bigoplus_{u'' \preceq u'} y^{u''} t^{u' \oplus u''}$, we observe that the coefficient of y^u in $F(y \oplus t)^v$ is $\bigoplus_{u' \succeq u} \lambda_{v,u'} \, t^{u' \oplus u}$. By hypothesis every term of this sum vanishes. □

The following lemma gives a *sufficient* condition for the absence of the monomial z^u, leading to a sufficient condition for an integral distinguisher.

Lemma 4. *Let $E : \mathbb{F}_2^n \times \mathbb{F}_2^\kappa \to \mathbb{F}_2^n$, $u, v \in \mathbb{F}_2^n$. If no exact trail from (u', w) to v exists for any $u' \succeq u$ and $w \in \mathbb{F}_2^\kappa$, then for every $(a, k) \in \mathbb{F}_2^n \times \mathbb{F}_2^\kappa$ it holds that:*

$$\lambda_u(a, k) = \bigoplus_{z \preceq u} E(a \oplus z, k)^v = 0 \ .$$

Proof. Let k. By Lemma 2, the hypothesis gives $\lambda_{v,(u',w)} = 0$ for every $u' \succeq u$ and $w \in \mathbb{F}_2^\kappa$; substituting the value of k, the coefficient of $x^{u'}$ in $E(x, k)^v$ vanishes for every $u' \succeq u$. Lemma 3, applied with $F = E(\cdot, k)$ and $t = a$, then gives $\lambda_u(a, k) = 0$ for every a. As k was arbitrary, this holds for every key. □

2.3 Unknown key additions and weak propagation

Although the round keys $k^{(0)}, \ldots, k^{(\rho)}$ satisfy the relations imposed by the key schedule, we relax the cipher model by treating them as independent symbolic inputs. Every sequence of round keys produced by the actual key schedule is still allowed in the relaxed model, so this relaxation can only add possible trails. Therefore, if no trail exists in this model, none exists for any master key. A weak transition lets us avoid tracking each round key as a separate variable in the search. This is the idea of the conventional bit-based division property (CBP) [31].

Definition 3 (CBP transitions). *For $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, its CBP transitions are defined by*

$$u \xrightarrow{F}_c v \quad \iff \quad \exists u' \succeq u, \ v' \preceq v, \quad \lambda_{v',u'} = 1.$$

By Lemma 3, the inequalities $u' \succeq u$ and $v' \preceq v$ correspond to unknown key additions before and after F, respectively. The key addition after F may be incorporated into the next layer in the decomposition. CBP allows key additions at every coordinate, even when the actual cipher adds key bits only to some of them. This may introduce additional transitions, but again, the absence of a CBP trail guarantees the absence of an exact trail in the actual cipher.

A *CBP trail* is a sequence as in Definition 2, with each exact transition replaced by a CBP transition. For each layer consisting of a key addition followed by a function F, we use the CBP transitions of F.

At any intermediate position, a reachable mask v is said to be *redundant* if another reachable mask $w \neq v$ satisfies $w \preceq v$. Indeed, by Definition 3, every CBP transition available from v is also available from w. It therefore suffices to keep the minimal reachable masks [31]. For a mask set $\mathcal{K}$, let $\mathrm{Min}_{\preceq}(\mathcal{K})$ denote its set of minimal elements.

Definition 4 (CBP propagation table [19]). *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$. The* CBP propagation table *of F is defined, for each input mask $u \in \mathbb{F}_2^n$, by*

$$\mathrm{DPPT}_F(u) := \mathrm{Min}_{\preceq}\{v \in \mathbb{F}_2^m : u \xrightarrow{F}_c v\}.$$

After each layer, we take the union of the table outputs for the retained input masks and remove redundant masks. A target mask v is reachable by a CBP trail if and only if some retained output mask w satisfies $w \preceq v$. Thus, if this model admits no trail from u to v, then no exact division trail exists for any fixed round-key tuple, and Lemma 4 gives $\lambda_u(a, k) = 0$ for every offset a and every master key k.

In the remainder of the paper, all division transitions are CBP transitions and, for simplicity, we omit the subscript c and write $u \xrightarrow{F} v$ for them.

3 Probabilistic integral distinguishers

Let $X \subseteq \mathbb{F}_2^n$ be a linear subspace of dimension d and let $\beta \in \mathbb{F}_2^n \setminus \{0\}$ be an output mask. For any master key k, we define the zero-sum probability (associated with X and β) by:

$$p_k(X, \beta) := \Pr_a \left[\bigoplus_{x \in a+X} \beta \cdot E_k(x) = 0 \right].$$

From an adversary's perspective, this key-dependent probability is the relevant quantity. Indeed, for an ideal permutation, the integral sum is expected to vanish with probability approximately $\frac{1}{2}$ over the choice of the offset a. This naturally leads to the following definition.

Definition 5 (Probabilistic zero-sum integral distinguisher). *The pair (X, β) defines a* probabilistic zero-sum integral distinguisher *if :*

$$\exists \varepsilon > 0, \forall k \in \mathbb{F}_2^\kappa, \quad p_k(X, \beta) = \frac{1}{2} + \varepsilon_k, \text{ where } \varepsilon \leq \varepsilon_k \leq \frac{1}{2}.$$

The maximal case $\varepsilon_k = \frac{1}{2}$ corresponds to a deterministic zero-sum distinguisher.

Let $u \in \mathbb{F}_2^n$ be the mask of X, *i.e.*, $X = \{z \in \mathbb{F}_2^n : z \preceq u\}$. By Lemma 1, $p_k(X, \beta) = \Pr_a [\lambda_u(a, k) = 0]$. Hence, (X, β) defines a probabilistic integral distinguisher precisely when the family of Boolean functions $\big(a \longmapsto \lambda_u(a, k)\big)_{k \in \mathbb{F}_2^\kappa}$

has a non-negligible bias for every key. In practice, the absence of the monomial corresponding to $\lambda_u(a, k)$ is established using CBP. Such a propagation requires choosing a decomposition $\mathcal{D} = \left(F^{(1)}, \ldots, F^{(r)}\right)$ of the cipher into layers. For a fixed decomposition $\mathcal{D}$ and a key k, we define

$$q_{k,\mathcal{D}}(X, \beta) := \Pr_a\Big[\text{no CBP trail (w.r.t. } \mathcal{D}) \text{ remains valid through } z \mapsto E_k(a \oplus z)\Big].$$

Contrary to $p_k(X, \beta)$, the quantity $q_{k,\mathcal{D}}(X, \beta)$ is not an intrinsic property of the complete function E_k: it depends on the decomposition used for the propagation. Indeed, if no valid trail exists through $\mathcal{D}$, then the corresponding monomial cannot occur in the ANF of the complete function. Therefore,

$$q_{k,\mathcal{D}}(X, \beta) \leq p_k(X, \beta).$$

The converse does not hold in general, since several valid trails through the decomposition may cancel when the layers are composed. Such cancellations are visible in the ANF of the complete function, and hence in p_k, but may not be detected by a trail-searching model. Hence the value of $q_{k,\mathcal{D}}$ may depend strongly on the chosen decomposition, even though p_k is unchanged.

The purpose of the following sections is not to remove the key from the CBP trail structure, since this is already handled by weak propagations, but rather to show that the randomness of the fixed-input assignments can itself be propagated in a key-independent manner, leading to a lower-bound estimate $q_{\mathcal{D}}(X, \beta)$ independent of the key.

We first formalize how the validity of a division trail depends on the fixed-input values specified by a. The main idea is to extend the standard S-box propagation tables by associating each transition with the set of fixed-input assignments under which it remains valid. Then, we explain the process to propagate the fixed-input assignments and finally explain how to efficiently estimate $q_{\mathcal{D}}(X, \beta)$.

3.1 Probabilistic propagation tables

The purpose of this section is to analyze how the fixed input assignment a affects division trails whose input monomial involves only the active variables. In particular, we show that a trail that exists before the fixed inputs are assigned may become invalid for certain assignments of a.

In the following, let $f : \mathbb{F}_2^{s+t} \to \mathbb{F}_2$, be a Boolean function in $s + t$ variables (*e.g.* a product or a sum of coordinates of a cryptographic vectorial function F), and write its input as

$$(a, x) = (a_0, \ldots, a_{s-1}, x_s, \ldots, x_{s+t-1}) \in \mathbb{F}_2^s \times \mathbb{F}_2^t,$$

where $a_0, \ldots, a_{s-1}$ are *fixed* input values and $x_s, \ldots, x_{s+t-1}$ are the *active* variables. We study whether the full active monomial $x^u = \prod_{i=s}^{s+t-1} x_i$, whose input monomial mask is $u = (\underbrace{0, \ldots, 0}_{s}, \underbrace{1, \ldots, 1}_{t})$, appears in the ANF of f, depending on the value of a.

A division trail from an input monomial mask u to an output[6] monomial mask v is valid if and only if every transition composing it is valid. Therefore, to invalidate a given trail, it suffices to invalidate one of its transitions. Suppose that a transition $u' \to v'$ appears through a parallel S-box layer. Since the S-boxes operate on disjoint sets of variables, this transition is valid if and only if all the corresponding local S-box transitions are valid. Hence, it can be invalidated by invalidating any of these local transitions. We may therefore focus on a single S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$. Write $y = S(x) = (y_0, \ldots, y_{m-1})$, and consider a local weak transition $u' \xrightarrow{S} v'$. By definition, there exist $u'' \succeq u'$ and $v'' \preceq v'$ such that the monomial $x^{u''}$ appears in the ANF of $y^{v''}$. Suppose now that some input coordinates of S do not depend on the active variables and *depend only on the fixed-input parameters* a. Once a is fixed, these coordinates are constant, and substituting their values may cause the monomial witnessing the transition to disappear. Consequently, the validity of the local transition may depend on the particular assignment of a.

For a fixed global assignment a, each partially-active S-box receives concrete values on the input coordinates that do not depend on the active variables. We denote the resulting local assignment by α. To capture this dependence, we associate each local transition with the set of assignments α under which it remains valid after the fixed inputs have been substituted.

Fixed and active inputs. Let $c \in \mathbb{F}_2^n$ denote the *fixed-input mask*. Throughout this section, c is fixed and indicates which input variables of the S-box are fixed:

$$c_i = \begin{cases} 1, & \text{if } x_i \text{ is fixed to a constant,} \\ 0, & \text{if } x_i \text{ remains an active variable.} \end{cases}$$

Thus, c specifies the positions of the fixed inputs, whereas α specifies their values.

Propagation tables with partially fixed inputs. Let $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be an S-box. We keep the notation introduced in Section 2: for $u \in \mathbb{F}_2^n$ and $v \in \mathbb{F}_2^m$, the coefficient $\lambda_{v,u}$ denotes the coefficient of x^u in the ANF of $S(x)^v$. An *assignment* to the fixed inputs is encoded by a vector $\alpha \preceq c$. More precisely, if $c_i = 1$, then α_i is the value assigned to x_i and if $c_i = 0$, we set $\alpha_i = 0$, which serves only as padding. We denote the set of such assignments by $\mathcal{A}_c := \{\alpha \in \mathbb{F}_2^n : \alpha \preceq c\}$. For $\alpha \in \mathcal{A}_c$, let $S|_\alpha$ denote the function obtained by substituting $x_i = \alpha_i$ whenever $c_i = 1$, while leaving the other input variables symbolic. Accordingly, for $v \in \mathbb{F}_2^m$, $S(x)^v\big|_\alpha$ is the polynomial obtained by applying the same substitution to $S(x)^v$. This is a polynomial in the $n - \mathrm{wt}(c)$ active variables. Since c is fixed, its dependence is omitted from the notation.

Example 1. Let $n = 3$ and $c = \texttt{110}$, using the bit ordering (x_2, x_1, x_0), so that x_2 and x_1 are fixed, while x_0 remains active. The associated assignment set is

[6] With our notation the only output monomial mask for f is of course $v = \texttt{1}$ but v can be thought of as any monomial mask v' of the vectorial function F if $f = F^{v'}$.

$\mathcal{A}_c = \{\texttt{000}, \texttt{010}, \texttt{100}, \texttt{110}\}$. For instance, the assignment $x_2 = 0$ and $x_1 = 1$ is represented by $\alpha = \texttt{010}$. Since $c_0 = 0$, the least-significant bit of α carries no assigned value: x_0 remains active.

Since the fixed input variables have been substituted, every input monomial of the restricted function involves only active variables. We represent such a monomial mask as a vector $u \in \mathbb{F}_2^n$ whose fixed coordinates are zero. Equivalently, such masks satisfy $c \odot u = 0$, where $\odot$ denotes the component-wise product.

Definition 6 (Transition under a fixed assignment). *Let $c, u \in \mathbb{F}_2^n$ be such that $c \odot u = 0$ and let $v \in \mathbb{F}_2^m$, $\alpha \in \mathcal{A}_c$. The transition from u to v is said to be* valid under the assignment α *if $u \xrightarrow{S|_\alpha} v$. Equivalently, it is valid if there exist $u' \succeq u$ and $v' \preceq v$ such that the ANF of $S(x)^{v'}\big|_\alpha$ contains the monomial $x^{u'}$.*

By definition, for such c, u, α, transitions under fixed assignments are related to standard transition by the following relation: $u \xrightarrow{S} v$ if and only if there exists $\alpha \in \mathcal{A}_c$, such that $u \xrightarrow{S|_\alpha} v$. In particular, the validity of a transition may depend on the assignment α. To record this dependence, we associate each transition with the set of assignments under which it is valid.

Definition 7 (Validity set). *For $c \odot u = 0$, the* validity set *of the transition from u to v is defined as*

$$\mathcal{V}_c^S(u, v) := \left\{ \alpha \in \mathcal{A}_c : u \xrightarrow{S|_\alpha} v \right\}.$$

The equality $\mathcal{V}_c^S(u, v) = \mathcal{A}_c$ means that the transition is valid under every assignment, whereas $\mathcal{V}_c^S(u, v) = \varnothing$ means that it is never valid. Every other validity set identifies precisely the assignments on which the transition is valid. Note that since we are considering *weak transitions*, for all (u, v, u', v') such that $u' \succeq u$ and $v' \preceq v$, we have that $\mathcal{V}_c^S(u', v') \subseteq \mathcal{V}_c^S(u, v)$. An example illustrating the computation of validity sets from restricted ANFs is given in Appendix A.

Extended propagation tables. Let c be a fixed-input mask. For each input monomial mask u, we define the *raw extended propagation table* by

$$\widetilde{\mathcal{T}}_{S,c}(u) := \left\{ \left(v, \mathcal{V}_c^S(u, v)\right) : v \in \mathbb{F}_2^m,\ \mathcal{V}_c^S(u, v) \neq \varnothing \right\}.$$

Each entry records both a possible output monomial mask and the set of assignments under which the corresponding transition is valid. For any fixed assignment $\alpha \in \mathcal{A}_c$, the ordinary propagation table of the restricted S-box $S|_\alpha$ can be recovered from the raw extended table. Indeed, by Definitions 4 and 7 it holds that:

$$\mathrm{DPPT}_{S|_\alpha}(u) = \mathrm{Min}_{\preceq} \left\{ v \in \mathbb{F}_2^m : \alpha \in \mathcal{V}_c^S(u, v) \right\}.$$

Thus, $\widetilde{\mathcal{T}}_{S,c}(u)$ simultaneously represents the propagation table associated with all assignments in $\mathcal{A}_c$. However, some entries of this table are currently redundant.

To identify them, we define the following partial order on pairs $(v, \mathcal{A})$, where $v \in \mathbb{F}_2^m$ and $\mathcal{A} \subseteq \mathcal{A}_c$:

$$(v, \mathcal{A}) \sqsubseteq (v', \mathcal{A}') \quad \Longleftrightarrow \quad v \preceq v' \text{ and } \mathcal{A} \supseteq \mathcal{A}'.$$

If $(v, \mathcal{V}_c^S(u, v)) \sqsubseteq (v', \mathcal{V}_c^S(u, v'))$, then every assignment under which the transition to v' is valid also admits the strictly smaller output mask v. Consequently, v' cannot be minimal for any of these assignments, and the corresponding entry may be discarded. We therefore define the *extended propagation table* as

$$\mathcal{T}_{S,c}(u) := \mathrm{Min}_{\sqsubseteq}\, \widetilde{\mathcal{T}}_{S,c}(u).$$

As already noted, for all (u, v, u', v') such that $u' \succeq u$ and $v' \preceq v$, we have that $\mathcal{V}_c^S(u', v') \subseteq \mathcal{V}_c^S(u, v)$. This implies that the partial order $(v, \mathcal{V}_c^S) \sqsubseteq (v', \mathcal{V}_c'^S)$ is equivalent to $v \preceq v'$ and $\mathcal{V}_c^S = \mathcal{V}_c'^S$. Thus, reducing the raw extended propagation table amounts to removing each pair $\big(v', \mathcal{V}_c^S(u, v')\big)$ for which there exists a strictly smaller mask $v \preceq v'$, with $v \neq v'$, having the same validity set. An example illustrating the construction and reduction of extended propagation tables is given in Appendix B.

3.2 Propagation of offset values and key-independent bounds

As already mentioned, the probability that no division trail is valid depends on the decomposition used to propagate division properties. In this section, we make this dependence explicit by tracking those values derived from the affine offset that remain constant when the active variables range over the integral structure. A key-independent lower bound will then follow from the distribution of these values. Let $\mathbb{F}_2^n = X \oplus \Gamma_0$, where X is the active input space, and let sample the offset a uniformly from Γ_0. Thus, for each fixed a, the integral structure is the affine space $a + X$. It is convenient to describe the constant information carried by the offset through linear forms. To do so, we denote by V^* the dual space of any subspace V. Let $\Lambda_0 = \{\lambda \in (\mathbb{F}_2^n)^* : \lambda(x) = 0 \ \forall x \in X\} \subseteq (\mathbb{F}_2^n)^*$ be the annihilator subspace of X.

Lemma 5 (Initial space Λ_0). *For any $\lambda \in \Lambda_0$, λ is constant on each coset of X. Furthermore for any basis $\lambda_{0,1}, \ldots, \lambda_{0,r_0}$ of Λ_0, the vector $\big(\lambda_{0,1}(a), \ldots, \lambda_{0,r_0}(a)\big)$ is uniformly distributed over $\mathbb{F}_2^{r_0}$ when a is sampled uniformly from Γ_0.*

Proof. The first fact holds since for any $x \in X$, $\lambda(x + a) = \lambda(a)$. Moreover, because Γ_0 is a linear complement of X, the restriction map $\Phi : \Lambda_0 \to \Gamma_0^*, \lambda \mapsto \lambda|_{\Gamma_0}$ is a well-defined isomorphism, which implies the second statement. □

Consider a decomposition of the iterated function into layers $\widehat{S}_i = L_i \circ S^{(i)}$, where the state space decomposes as $\mathbb{F}_2^n = U_i \oplus W_i$ and $\widehat{S}_i(u, w) = (S^{(i)}(u), w)$. Here $S^{(i)}$ denotes the nonlinear function processed at layer i, whereas the coordinates in W_i bypass this nonlinear function and L_i is linear and invertible. Let $\mathcal{F}_i = F_i \circ \cdots \circ F_0$ be the composition of the first layers. We define subspaces

$\Lambda_i \subseteq (\mathbb{F}_2^n)^*$ of tracked constant linear forms at the input of $\widehat{S}_i$. Our goal is that for every $\lambda \in \Lambda_i$, $\lambda \circ \mathcal{F}_i(x+a) = \lambda \circ \mathcal{F}_i(x)$ for $x \in X$, $a \in \Gamma^0$, while the vector of values of any basis of Λ_i, together with all assignments extracted at previous layers (that will be used to specialize the propagation tables of $S^{(0)}, \ldots, S^{(i-1)}$), follows a product of uniform distributions.

In input of a nonlinear layer $\widehat{S}_i$, we identify U_i^* with the forms that vanish on W_i, and W_i^* with the forms that vanish on U_i. Define

$$A_i := \Lambda_i \cap U_i^*, \qquad B_i := \Lambda_i \cap W_i^*.$$

The forms in A_i give constant values involving only the input of $S^{(i)}$, whereas those in B_i give constant values involving only the coordinates that bypass $S^{(i)}$. Let M_i be any complement of $A_i \oplus B_i$ in Λ_i, i.e. $\Lambda_i = A_i \oplus B_i \oplus M_i$. The forms in M_i mix the nonlinear input and the bypassing coordinates. We deliberately do not exploit such relations.

Suppose first that $A_i = U_i^*$. Then the complete input of $S^{(i)}$ is constant throughout each integral structure. Across the different offsets, this input is uniformly distributed over U_i, up to a fixed translation such as a round-key addition. If $S^{(i)}$ maps the uniform distribution over its input space to the uniform distribution over its output space, then all its output coordinates can continue to be tracked after the nonlinear layer, as well as the ones of B_i. Thus we can define $\Lambda_{i+1} \coloneqq (L_i^{-1})^*(A_i \oplus B_i)$, or equivalently (because $A_i = U_i^*$), $\Lambda_{i+1} \coloneqq (L_i^{-1})^*(\Lambda_i)$. Indeed, if $y' = L_i y$, then a constant form λ on the input state induces the form $\lambda' = \lambda \circ L_i^{-1}$ on the output state. Identifying linear forms with column vectors, this corresponds to $\Lambda_{i+1} = (L_i^{-1})^\top(\Lambda_i)$. For every state y, $((L_i^{-1})^\top \lambda)(L_i y) = \lambda(y)$, so the associated constant value is unchanged.

Suppose now that $A_i \subsetneq U_i^*$. The values of a basis of A_i define a local assignment α_i used to specialize the propagation table of $S^{(i)}$.[7] These values are constant throughout the current integral structure. Once they have been used for this local specialization, they are not used again at a later nonlinear layer. Only the constant forms in B_i, which involve exclusively the bypassing coordinates, are retained. The forms in M_i are discarded.

Thus, after a partially-constant nonlinear layer, our conservative propagation retains only B_i so that Λ_{i+1} is defined by $\Lambda_{i+1} \coloneqq (L_i^{-1})^*(B_i)$ in that case. We do not attempt to infer additional constant relations from the output of the restricted function $S^{(i)}|_{\alpha_i}$. This gives the following product-distribution property.

Lemma 6 (Distribution of the extracted assignments). *Assume that:*

- **(i)** *whenever the complete input of a nonlinear function $S^{(i)}$ is constant throughout the integral structure, $S^{(i)}$ maps the uniform distribution on its input space to the uniform distribution on its output space; and*
- **(ii)** *at every partially-constant nonlinear layer, the values associated with A_i are used for the local assignment, those associated with B_i are retained, and those associated with M_i are discarded.*

[7] Fully exploiting A_i might require the application of an invertible linear map at the input of $S^{(i)}$ before computing its propagation table.

Then all local assignments $\alpha_{i_1}, \ldots, \alpha_{i_r}$ extracted during the propagation are mutually independent and each is uniformly distributed over its respective space: the distribution of $(\alpha_{i_1}, \ldots, \alpha_{i_r})$ is a product of uniform distributions.

Assumption **(i)** is in particular satisfied by every bijective S-box.

Proof. We proceed inductively based on the initial case given by Lemma 5. Suppose that the assignments extracted so far, together with the values of a basis of Λ_i, follow a product of uniform distributions. Choose a basis of Λ_i adapted to $\Lambda_i = A_i \oplus B_i \oplus M_i$. Since an invertible change of basis preserves the uniform distribution, the three corresponding blocks of values are jointly uniform. The A_i-block becomes the new assignment α_i and after discarding the M_i-block, the B_i-block remains uniform and independent of all assignments extracted so far.

An invertible linear layer only replaces the tracked forms by their pullbacks through the inverse linear map and therefore leaves their values unchanged. When the complete input of $S^{(i)}$ is constant, Assumption **(i)** replaces a uniform input vector by a uniform output vector, without affecting the distribution of the bypassing values. The claim follows by induction. □

Conservative propagation model. The propagation described above deliberately retains only constant information that can be represented by the local conditional tables. It should therefore not be interpreted as an exact propagation of all affine relations carried by the cipher.

Let $\boldsymbol{\alpha} = (\alpha_{i_1}, \ldots, \alpha_{i_r})$ denote the collection of assignments extracted by the propagation. By Lemma 6, $\boldsymbol{\alpha}$ follows the product of the corresponding uniform distributions. For a fixed $\boldsymbol{\alpha}$, let $\mathcal{T}_{\mathcal{D}}(\boldsymbol{\alpha})$ denote the set of CBP trails through the decomposition $\mathcal{D}$, using the conditional propagation tables associated with the extracted assignments and the generic CBP propagation table whenever constant information has been discarded. We define

$$q_{\mathcal{D}}(u, v) := \Pr_{\boldsymbol{\alpha}} \left[\mathcal{T}_{\mathcal{D}}(\boldsymbol{\alpha}) = \varnothing \right].$$

The propagation is conservative in the following sense. For every concrete offset, each local transition allowed by the corresponding concrete propagation is also allowed by our abstract propagation, because any constant information that is not explicitly exploited is replaced by the generic propagation table. Consequently, the set of concrete trails is contained in $\mathcal{T}_{\mathcal{D}}(\boldsymbol{\alpha})$. Therefore, whenever $\mathcal{T}_{\mathcal{D}}(\boldsymbol{\alpha})$ is empty, no concrete trail exists for the corresponding offset. This is the inclusion that yields the desired lower bound.

Key independence. The quantity $q_{\mathcal{D}}$ is independent of the secret key. Indeed, the candidate CBP trails through $\mathcal{D}$ are already key-independent by weak propagation. A fixed-key addition only acts as a translation of the local constant assignments α_i. Since each α_i is uniformly distributed and the assignments are mutually independent, such translations only permute their assignment spaces and do not change the distribution of $\mathcal{T}_{\mathcal{D}}(\boldsymbol{\alpha})$. We therefore obtain the following result.

Theorem 1 (Key-independent no-valid-trail bound). *For a fixed decomposition $\mathcal{D}$, the probability $q_{\mathcal{D}}(u, v)$ is independent of the secret key and, for every key k, satisfies*

$$q_{\mathcal{D}}(u, v) \leq q_{k,\mathcal{D}}(u, v).$$

In particular, $q_{\mathcal{D}}(X, \beta) \leq p_k(X, \beta)$.

The value of $q_{\mathcal{D}}$ nevertheless depends on the chosen decomposition and on the constant information represented by the propagation model. For instance, when Γ_0 is a coordinate subspace and the linear layers are bit permutations, no mixed direction is introduced and no information is lost: $\sum_i \dim A_i = \dim \Gamma^0$. However, the model may still discard constant relations generated at the output of a partially-active nonlinear function and this does not imply that $q_{\mathcal{D}} = q_{k,\mathcal{D}}$.

Remark 1 (A larger active space is not always better). As already mentioned, if no CBP trail exists from u to v, then none exists either from any $u' \succeq u$. A deterministic zero sum for a cube therefore extends to every larger cube containing it. However, the probability $q_{\mathcal{D}}(u, v)$ does not necessarily follow the same order: it is possible to have $q_{\mathcal{D}}(u, v) > q_{\mathcal{D}}(u', v)$ for $u \preceq u'$. Enlarging the active space from X to $X' \supseteq X$ turns some of the coordinates of Γ_0 into active ones, so they can no longer be exploited to specialize the propagation tables of the functions they used to feed. For specific values of these coordinates, this specialization can have a high probability of invalidating a trail, an effect that a larger (and in particular more costly) cube simply forgoes. This is exactly what happens for the two smallest 6-round `PRESENT` structures of Table 4: enlarging $D = 2^4$ to the strictly larger $D = 2^5$ lowers $\widehat{q}$ from 0.8317 to 0.7068.

3.3 Computing the probability that no trails are valid

We now explain how the extended propagation tables introduced in Section 3.1 can be used to compute the no-valid-trail probability $q_{\mathcal{D}}(u, v)$ for the fixed decomposition $\mathcal{D}$. Throughout this section, the decomposition $\mathcal{D}$ and the monomial masks u and v are fixed. As before, a mask c specifies which coordinates are fixed and which ones remain active. For an m-bit S-box, the set of possible assignments to its fixed coordinates is $\mathcal{A}_c = \{\alpha \in \mathbb{F}_2^m : \alpha \preceq c\}$.

Note that for all trails to be invalid, it is necessary that at least one transition $u' \xrightarrow{S} v'$ is invalid for each trail.

Conditional assignment sets. For every S-box $S^{(j)}$ having some fixed input coordinates, let $c^{(j)}$ denote its fixed-coordinate mask. We also associate to each S-box a current set of admissible assignments $C^{(j)} \subseteq \mathcal{A}_{c^{(j)}}$. Initially, no constraint is imposed on any S-box and $C^{(j)}$ is therefore initialized as $C^{(j)} = \mathcal{A}_{c^{(j)}}$. Then, the set $C^{(j)}$ is progressively restricted as probabilistic transitions involving $S^{(j)}$ are considered. Let $\mathcal{C} = \left\{C^{(j)}\right\}_j$ denote the current collection of admissible assignment sets, and let $P(\mathcal{C})$ denote the *conditional* probability that no valid division trail exists under these constraints.

If no division trail exists under the current constraints, then $P(\mathcal{C}) = 1$. Conversely, if a valid trail exists and all its transitions are deterministic under the current constraints, then $P(\mathcal{C}) = 0$. In any other case, there exists at least one trail containing at least one probabilistic transition under the current constraints and $P(\mathcal{C})$ remains unknown. Let us consider such a trail and such a transition $u' \to v'$ through an S-box $S^{(j)}$, and let $\mathcal{V} = \mathcal{V}_{c^{(j)}}^{S^{(j)}}(u', v')$ be the associated validity set. To compute $P(\mathcal{C})$, we can distinguish two cases, according to whether this transition is valid or invalid. These cases will respectively define the *in* and *out* branches of our recursive algorithm.

In the *in* branch, we assume that $u \to v$ is a deterministic and valid transition through $S^{(j)}$, while we assume that it is invalid in the *out* branch. Given the current constraints $\mathcal{C}$, these events have respective conditional probability:

$$p_{\text{in}} = \frac{|C^{(j)} \cap \mathcal{V}|}{|C^{(j)}|} \quad \text{and} \quad p_{\text{out}} = \frac{|C^{(j)} \setminus \mathcal{V}|}{|C^{(j)}|}.$$

The assignments $\mathcal{C}$ are then restricted to the assignments that allows $u \to v$ to hold or not. In other words, we update $\mathcal{C}$ using $C^{(j)} \leftarrow C^{(j)} \cap \mathcal{V}$ in the *in* branch and using $C^{(j)} \leftarrow C^{(j)} \setminus \mathcal{V}$ in the *out* branch and denote by $\mathcal{C}_{\text{in}}$ and $\mathcal{C}_{\text{out}}$, the respective resulting collections (which coincide with $\mathcal{C}$ except for the updated set $C^{(j)}$). The computation then proceeds recursively under each new collection of constraints.

Recursive relation. The two events above are disjoint and form a partition of the current assignment set. The law of total probability therefore gives

$$P(\mathcal{C}) = p_{\text{in}} P(\mathcal{C}_{\text{in}}) + p_{\text{out}} P(\mathcal{C}_{\text{out}}).$$

Recursively applying this identity provides the tree-search algorithm we use to evaluate the probability that no division trail remains.

The recursion tree. The procedure defines a binary probability tree diagram whose root represents the initial collection $\mathcal{C}_0$, with $C^{(j)} = \mathcal{A}_{c^{(j)}}$ for every j, and whose edge probabilities are p_{in} and p_{out} (whose values depend on the constraints of the parent node). Each root-to-leaf path represents the initial assignments satisfying all conditions imposed along that path. Since each split is disjoint and exhaustive, distinct root-to-leaf paths represent pairwise disjoint subsets that together cover the initial assignment space. The contribution of a path is computed by multiplying the 0 or 1 probability of its leaf with the edge probabilities along the path. Thus, *no-trail* leaves contribute their full path probabilities, whereas *deterministic-trail* leaves contribute zero. By the law of total probability, summing these contributions gives $P(\mathcal{C}_0)$ which is precisely the probability that no trail is valid, where the probability is taken over the uniformly distributed initial fixed-coordinate assignments. Finally, no branch can be extended indefinitely since we only branch on *probabilistic* transitions $u \to v$ which necessarily makes both C_{in} and C_{out} strictly smaller sets of assignments compared to C, so the tree traversal terminates. We thus obtain the following result.

Theorem 2. *The recursive algorithm terminates and computes exactly $q_{\mathcal{D}}(u, v)$, the probability over the initial fixed-coordinate assignments that no valid division trail exists between the prescribed input monomial mask x^u and output mask y^v through the decomposition $\mathcal{D}$. This probability is independent of the secret key.*

An example illustrating how the recursive procedure restricts the admissible assignments and computes the no-valid-trail probability is given in Appendix C.

Approximate computation. To limit the cost of the recursive search, we introduce a cutoff δ on the probability of reaching a node, below which exploration is approximated. The algorithm returns certified lower and upper bounds on $q_{\mathcal{D}}(X, \beta)$, together with a heuristic estimate $\widehat{q}_{\mathcal{D}}(X, \beta)$. This estimate is not itself guaranteed to be a lower bound. The update rules and a motivating example are given in Appendix D.

From the probability of no trail to the probability of a zero sum. The recursive algorithm, combined with the cutoff approximation described in Appendix D, provides an estimate $\widehat{q}_{\mathcal{D}}(X, \beta)$ of the probability $q_{\mathcal{D}}(X, \beta)$ that no valid division trail exists. For a fixed key k, the absence of a valid division trail guarantees $\mathcal{I}_{X,\beta}(a, k) = 0$. Consequently,

$$q_{\mathcal{D}}(X, \beta) \leq p_k(X, \beta).$$

The converse does not hold: valid division trails may exist while their algebraic contributions cancel in the final sum. To isolate precisely what remains heuristic, let $T_{\mathcal{D}}(a)$ denote the event that at least one valid division trail exists, and define the residual conditional bias

$$\varepsilon_{k,\mathcal{D}}(X, \beta) := \Pr_a \left[\mathcal{I}_{X,\beta}(a, k) = 0 \,|\, T_{\mathcal{D}}(a)\right] - \frac{1}{2}.$$

The law of total probability gives the exact identity

$$\begin{aligned} p_k(X, \beta) &= q_{\mathcal{D}}(X, \beta) + \big(1 - q_{\mathcal{D}}(X, \beta)\big) \left(\frac{1}{2} + \varepsilon_{k,\mathcal{D}}(X, \beta)\right) \\ &= \frac{1}{2} + \frac{1}{2} q_{\mathcal{D}}(X, \beta) + \big(1 - q_{\mathcal{D}}(X, \beta)\big) \varepsilon_{k,\mathcal{D}}(X, \beta). \end{aligned}$$

Our additional heuristic is that, once the division-property propagation no longer forces the integral sum to zero, there is no residual bias towards either value. In other words, we assume $\varepsilon_{k,\mathcal{D}}(X, \beta) \approx 0$, and therefore

$$p_k(X, \beta) \approx \frac{1}{2} + \frac{1}{2} q_{\mathcal{D}}(X, \beta).$$

In practice, we replace $q_{\mathcal{D}}(X, \beta)$ by its estimate $\widehat{q}_{\mathcal{D}}(X, \beta)$.

The division-property propagation identifies assignments for which $\mathcal{I}_{X,\beta}(a, k)$ is structurally forced to zero. On the remaining assignments, our heuristic treats

this Boolean function as approximately balanced. This assumption is not valid for arbitrary decompositions and functions. For example, in a composition such as $S \circ S^{-1}$, many intermediate algebraic paths may survive the local propagation while global algebraic identities force their final contributions to cancel systematically. Such a construction can therefore produce a large residual bias $\varepsilon_{k,\mathcal{D}}$. Our heuristic amounts to assuming that no such cancellation occur in the instances we consider. In our experiments, a typical instance admits many valid trails, whose validity conditions involve different subsets and values of the constant assignments. As the offset varies, the collection of surviving trails therefore changes substantially. Although this does not prove that $\mathcal{I}_{X,\beta}(a,k)$ is balanced, it provides empirical motivation for using $1/2$ as the baseline probability in the absence of a structural no-trail certificate.

Finally, the heuristic is not needed to certify a probabilistic zero-sum property. Since $p_k(X,\beta) \geq q_{\mathcal{D}}(X,\beta)$ for every key k, any certified key-independent lower bound $\underline{q}_{\mathcal{D}}(X,\beta) > \frac{1}{2}$ immediately implies $p_k(X,\beta) > \frac{1}{2}$ for every key k, independently of the residual-bias heuristic.

3.4 Implementation

We implemented the search procedure described above in `C++`. Its complete algorithmic description can be found in Appendix D.2 (Algorithm 1). The division-trail searches themselves are formulated as MILP models and solved with `Gurobi`.

A direct enumeration of all input subspaces and output masks would be prohibitively large. We therefore restrict the search to configurations compatible with the S-box and Super-Sbox structure of the cipher: active coordinates are grouped by completing S-boxes first and then Super-Sboxes, leaving at most one partially active S-box and one partially filled Super-Sbox. For a partially active S-box, we additionally enumerate linear subspaces of the required dimension and represent the S-box in a basis adapted to each candidate, since the precision of division-property propagation depends on the chosen linear representation [24]. Similarly, at the output of the final S-box layer, we test relevant linear combinations and retain only the nondominated ones. Thus, the search covers structured placements together with favorable linear representations at the input and output of nonlinear functions, following techniques related to linearly equivalent S-boxes and improved division-property propagation [14].

Even after restricting the input and output spaces as described above, the number of possible configurations remains too large to run the complete probability-tree exploration for every candidate. We therefore used a heuristic ranking, looking at configurations for which the probability of the most probable trails is the smallest.

4 Applications

We tested our tool on the lightweight block ciphers `PRESENT` and `GIFT`, as well as on the `ASCON` permutation. The resulting probabilistic and probability-one

integral distinguishers, together with the corresponding cipher specifications, are presented in detail in Appendix F. Every empirical zero-sum probability reported there is averaged over a large number of trials: most configurations use 10^3 independently sampled master keys with 10^4 offsets per key, and even our least extensive experiment, for the 8-round distinguishers with 32 active bits, still uses 64 master keys with 4, 000 offsets sampled for each of them. For both `PRESENT` and `GIFT`, these experiments also show that the zero-sum probability varies only slightly across the sampled master keys.

We now examine in more detail one of these distinguishers, a 6-round `PRESENT` property with an unusually large bias, and explain it algebraically.

4.1 On the 6-round distinguisher with $D = 2^4$ for `PRESENT`

In this section, we examine in more detail the 6-round distinguisher with $D = 2^4$ and a zero-sum probability of approximately 0.9 presented in Table 4. We first give an algebraic explanation of its bias, highlighting the role of the `PRESENT` S-box and bit permutation. We then connect this analysis to the automated search and explain how it helps interpret the tool's results.

Algebraic explanation of the distinguisher. We consider the distinguisher obtained by varying the rightmost plaintext nibble (corresponding to the input space $X = \langle e_0, e_1, e_2, e_3 \rangle \subseteq \mathbb{F}_2^{64}$ and mask $u = \texttt{0}\ldots\texttt{01111}$) and observing the least significant output bit after six rounds (corresponding to linear mask $\beta = e_0$ or monomial mask $v = e_0$). For a fixed key k and a fixed 60-bit offset a, we thus study the integral sum $\mathcal{I}_{X,\beta}(a,k) = \bigoplus_{z \in \mathbb{F}_2^4} \beta \cdot E_k(a \oplus z_0 e_0 \oplus \cdots \oplus z_3 e_3)$ that we abbreviate as $\mathcal{I}(a,k)$ in the following. By Lemma 1, this sum equals the coefficient of $z^u = z_0 z_1 z_2 z_3$ in the ANF of the selected output bit, expressed as a function of the four active input variables z_0, z_1, z_2, z_3. This output bit can also be expressed as $G_{a,k}(w)$ where the variables $w = (w_0, w_1, w_2, w_3)$ correspond to the output of the rightmost S-box of the first round. Because the initial key addition and the S-box are bijective, the mapping $z \mapsto w$ is a permutation of $\mathbb{F}_2^4$. Consequently, the coefficient $\mathcal{I}(a,k)$ of z^u is also the coefficient of w^u in the ANF of $G_{a,k}$.

Activity constraints. As shown in Figure 1, after the first linear layer, the least significant input coordinate of S-box S_{4i} in Round 2 is $w_i \oplus k$, where k denotes the corresponding round-key bit, while the three other input bit are fixed. This implies that, for any $t \in \{0,1,2,3\}$ the output coordinate of index t can be expressed as an affine function of the single active variable w_i, namely $c_t(\alpha) \oplus d_t(\alpha) w_i$, where $d_t(\alpha)$ is the evaluation of the derivative of S_t along e_0 at point α:

$$\forall \alpha \in \mathcal{A}_{\texttt{1110}}, \quad D_{e_0} S(\alpha) \coloneqq S(\alpha) \oplus S(\alpha \oplus e_0) \text{ and } \; D_{e_0} S(\alpha) =: (d_0, d_1, d_2, d_3)(\alpha)\,.$$

Interestingly, two of the components of the `PRESENT` S-box are linear in x_0:

$$S_0(x) = x_0 \oplus x_1 x_2 \oplus x_2 \oplus x_3, \quad \texttt{1010} \cdot S(x) = x_0 \oplus x_1 x_2 \oplus x_1 x_3 \oplus x_2 x_3 \oplus 1.$$

This naturally implies some *linear structures*: their respective derivatives d_0 and $d_1 \oplus d_3$ are constant and equal to 1. In terms of output of the second S-box layer, this means that the output coordinate 0 always depends on w_i, while exactly one of output coordinates 1 and 3 depends on w_i. All in all, by also taking coordinate 2 into account, the derivative $D_{e_0}S$ admits the following image set where each image admits two assignments α as preimages: $\mathcal{P} \coloneqq D_{e_0}S(\mathcal{A}_{1110}) = \{(1,1,0,0),(1,1,1,0),(1,0,0,1),(1,0,1,1)\}$ These four images are the possible activity patterns in output of the second S-box layer.

A similar situation occurs for the third S-box layer: each S-box admits (at most) one active input depending on a single variable w_i, so all its outputs are affine in that variable. Hence, for any $i \in \{0,1,2,3\}$, the coordinate of index i of each relevant Round 4 S-box S_j, $j \in \{4,\dots,11\}$, can be written as $x_i^{(j)} = \gamma_i^{(j)} w_i \oplus \delta_i^{(j)}$, where $\gamma_i^{(j)}, \delta_i^{(j)}$ depend on the offset and key.

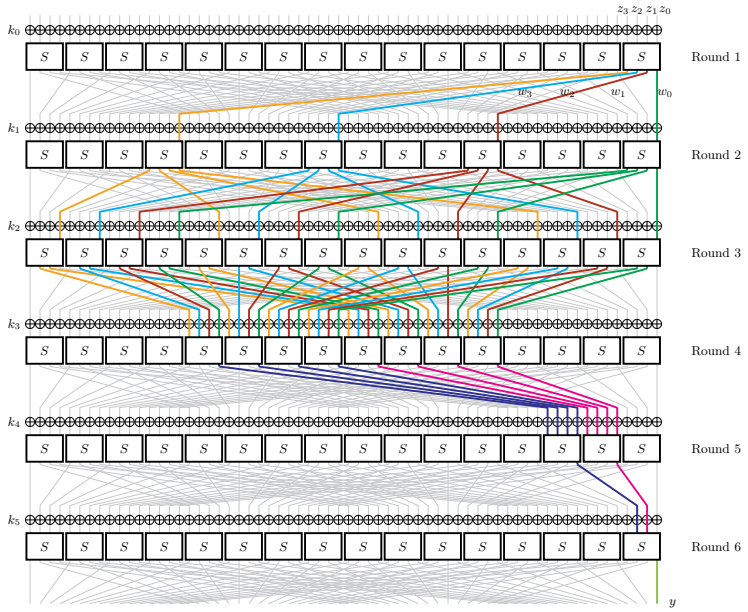

Fig. 1. Propagation of w_0, w_1, w_2, w_3 along the paths relevant to $\mathcal{I}_{X,\beta}(a,k)$.

Relevant propagation paths. From Round 4 to Round 6, we observe that only least significant outputs $S_0(x) = x_0 \oplus x_1x_2 \oplus x_2 \oplus x_3$ of previous S-boxes serve as inputs of the next relevant S-boxes. In particular, the input of the considered S-boxes at Round 5 are at most quadratic: the only possible quadratic term is w_1w_2 while some linear terms in w_i's may also appear. Applying S_0 at Round 5 involves multiplying two such expressions and adding three others, which leads to inputs x_t of the final S-box of degree at most 3 (because $(w_1w_2)^2 = w_1w_2$).

This implies that only the product x_1x_2 of S_0 can contribute to the coefficient $\mathcal{I}(a,k)$ of the monomial w^u of degree 4. As shown in Figure 1, up to a renaming of the variables, the ANFs of x_1 and x_2 in variables $w_i, \gamma_i^{(j)}, \delta_i^{(j)}$ are identical. In particular, the monomials from either branch that can contribute to the quartic coefficient belong to

$$\mathcal{M} = \{w_0,\ w_2, w_3,\ w_0w_1,\ w_0w_2,\ w_0w_3,\ w_1w_2,\ w_1w_3,\ w_2w_3,\ w_0w_1w_2,\ w_1w_2w_3\}.$$

A constant term and w_1 may also occur, but cannot contribute because the other branch contains neither $w_0w_1w_2w_3$ nor $w_0w_2w_3$. The coefficient $\mathcal{I}(a,k)$ is then a sum of products of coefficients of monomials whose product is $w_0w_1w_2w_3$.

The 256 activity scenarios. Let $d^{(i)} \in \mathcal{P}$ be the derivative pattern associated with w_i. A scenario is a tuple $s = (d^{(0)}, d^{(1)}, d^{(2)}, d^{(3)}) \in \mathcal{P}^4$ indicating the activity pattern in output of the four active S-boxes at Round 2. The $4^4 = 256$ scenarios have equal probability because the input assignments are uniformly random and each value in $\mathcal{P}$ has two preimages by $D_{e_0}S$. Each scenario actually constrains some coefficients $\gamma_i^{(j)}$'s in input of Round 4. Indeed, if output t of S_{4i} in Round 2 is inactive, then so is the input 0 of both S_{4+t} and S_{8+t} in Round 4:

$$d_t^{(i)} = 0 \quad \Longrightarrow \quad \gamma_i^{(4+t)} = \gamma_i^{(8+t)} = 0. \tag{3}$$

If $d_t^{(i)} = 1$, the two coefficients may still vanish, depending on the fixed inputs in Round 3.

Classification and algebraic probability. For each scenario s, let $\mathcal{I}_s$ be the polynomial in variables $\gamma_i^{(j)}, \delta_i^{(j)}$ coefficients obtained by imposing Eq. (3). The 256 polynomials fall into 32 classes, where two polynomials belong to the same class if a bijective renaming of their variables transforms one into the other. The biggest class is the class where $\mathcal{I}_s$ is identically zero, which regroup 106 scenarios for which the zero-sum vanishes, independently of the remaining assignments. In the other 150 scenarios, the polynomial is sparse and depends on maximum 40 variables. In particular, we can *exactly* compute the bias of each $\mathcal{I}_s$ assuming that its inputs $\gamma_i^{(j)}, \delta_i^{(j)}$ are independent and uniformly random by evaluating its full look-up table and then compute the average probability of being zero:

$$\frac{1}{256}\sum_{s\in\mathcal{P}^4} \Pr[\mathcal{I}_s = 0] = \frac{60,033,501}{67,108,864} \approx 0.89457.$$

Under the assumption that the coefficients $\gamma_i^{(j)}, \delta_i^{(j)}$ are independent and uniformly random, this provides an estimate of $p_k(X,\beta)$ which is close to the empirical probabilities 0.8945 and 0.8947 for `PRESENT-80` and `PRESENT-128` (see Table 4), respectively. Details on each class (number of monomials, variables, degree and bias) are listed in Table 6 of Appendix G.

The identities satisfied by the S-box derivatives, together with the bit permutation, restrict how the active variables combine. These restrictions help explain the high zero-sum probability.

Connection with the automated search. The aim of this section is to show how the activity restrictions from the algebraic analysis are represented in the automated search. This correspondence allows us to condition the search on each of the 256 scenarios and compute the probability that no valid division trail remains within that scenario.

Consider a Round 2 S-box with an active variable w_i in input of coordinate 0, while coordinates 1, 2, and 3 are fixed to $\alpha \in \mathcal{A}_{\mathtt{1110}}$, and let us study its output coordinate t. With the notation introduced above, the transition $e_0 \to e_t$ is valid exactly when the monomial w_i appears, that is when $d_t(\alpha) = 1$. Consequently, the associated validity set is given by $\mathcal{V}^S_{\mathtt{1110}}(e_0, e_t) = \{\alpha \in \mathcal{A}_{\mathtt{1110}} : d_t(\alpha) = 1\}$. Thus, each activity pattern describes which of the four transitions $e_0 \to e_t$, for $t \in \{0, 1, 2, 3\}$, are valid. The validity sets used by the tool express these same conditions in terms of the fixed-input assignments.

Conditioning on a scenario. For a pattern $d \in \mathcal{P}$, let $\mathcal{A}(d)$ be the preimage of d that is the set of assignments leading to this pattern: $\mathcal{A}(d) := \{\alpha \in \mathcal{A}_{\mathtt{1110}} : D_{e_0}S(\alpha) = d\} = D_{e_0}S^{-1}(\{d\}) \cap \mathcal{A}_{\mathtt{1110}}$. Our algorithmic search can then be restricted to one of the scenarios $s = (d^{(0)}, d^{(1)}, d^{(2)}, d^{(3)})$ by restricting the initial admissible assignments of each Round 2 S-box S_{4i} to $\mathcal{A}(d^{(i)})$. The freedom on the specific values of the assignments (and on S-boxes of later rounds) is handled by the recursive algorithm. All transitions through a given S-box use the same admissible assignment set. Their validity conditions therefore refer to the same fixed input bits, preserving relations such as $d_1(\alpha) \oplus d_3(\alpha) = 1$.

To express the resulting probabilities, fix a key k and let $\sigma(a, k)$ denote the scenario determined by the offset a and this key. Let N be the event that no valid weak division trail reaches the selected output and for each scenario s, define

$$q(s) := \Pr_a[\mathsf{N} \mid \sigma(a, k) = s], \qquad p(s) := \Pr_a[\mathcal{I}(a, k) = 0 \mid \sigma(a, k) = s].$$

Since the 256 scenarios are equally likely, it holds that:

$$q_{\mathcal{D}}(X, \beta) = \frac{1}{256} \sum_{s \in \mathcal{P}^4} q(s), \qquad p_k(X, \beta) = \frac{1}{256} \sum_{s \in \mathcal{P}^4} p(s).$$

The conditional search computes $q(s)$ by averaging over these 256 scenarios and returns a no-trail probability of approximately 0.8316821, within the interval $[0.8312, 0.8318]$ obtained by searching without fixing a scenario. We also find 106 cases with $q(s) = 1$, matching the number of $\mathcal{I}_s$ which are identically zero.

The tool therefore certifies a zero sum for about 83.17% of the offsets. The conversion $\widehat{p} = (1 + \widehat{q})/2$ mentioned above raises the prediction to 91.58%, which is above the experimental value of about 89.46%. This has to be compared with the algebraic model which considers cancellations between nonzero contributions and which estimates the probability to be 0.89457, which is closer to the experiments, despite the hypotheses made on $\gamma_i^{(j)}, \delta_i^{(j)}$. While the no-trail analysis explains much of the observed bias, more accurate predictions will then require a better understanding of the assignments for which a trail survives.

5 Filtering with probabilistic integrals

This section explains how to turn a probabilistic integral distinguisher into a *partial* key-recovery attack and when this approach is preferable to using a deterministic one. In this work, we compute each integral sum by partially decrypting all 2^d ciphertexts of a structure under the candidate key and XORing the recovered values of the selected bit. For simplicity, we do not use FFT-based key recovery [30] or partial-sum techniques [16]. Let ℓ denote the number of effective key bits guessed in the rounds appended after the distinguisher, so that there are 2^ℓ key classes to test, exactly one of which is correct. Under the usual wrong-key-randomization hypothesis, for a key candidate $\widehat{k}$ and an affine coset $a + X$, it holds that:

$$\forall \widehat{k} \neq k, \quad \Pr[\mathcal{I}_{X,\beta}(a, \widehat{k}) = 0] = \frac{1}{2}, \quad \text{whereas } \Pr[\mathcal{I}_{X,\beta}(a, k) = 0] = p > \frac{1}{2}.$$

The deterministic case. With a deterministic zero-sum ($p = 1$), testing s independent structures $a_1 + X, \ldots, a_s + X$ and discarding every candidate whose integral sum is nonzero for any structure leaves an expected number of surviving wrong key classes of $(2^\ell - 1)2^{-s} \approx 2^{\ell-s}$. Since only the keys that pass the first $i - 1$ filterings are tested against $a_i + X$, the i-th structure requires testing approximately $2^{\ell-i+1}$ key candidates on 2^d ciphertexts each. Summing the geometric series gives the following time and data complexities:

$$T_{\text{det}}(s) \approx 2^{d+\ell+1}(1 - 2^{-s}), \qquad D_{\text{det}}(s) = s\,2^d.$$

5.1 Probabilistic integral filtering

For $p < 1$, discarding a key candidate after a single failed structure has probability $1-p$ of discarding the correct key. We therefore filter candidates in stages, basing each decision on the statistics accumulated from the structures evaluated so far. For each structure $a_j + X$, let $Y_j(\widehat{k}) = \mathbf{1}\left[\mathcal{I}_{X,\beta}(a_j, \widehat{k}) = 0\right]$. For a wrong key $\widehat{k} \neq k$, $Y_j(\widehat{k}) \sim \text{Ber}(1/2)$, while for the correct one $Y_j(k) \sim \text{Ber}(p)$. After m structures, we thus consider $S_m(\widehat{k}) = \sum_{j=1}^{m} Y_j(\widehat{k})$. At filtering stage i, we use the Neyman–Pearson test between

$$H_0 : \ S_m \sim \text{Bin}(m, 1/2), \qquad H_1 : \ S_m \sim \text{Bin}(m, p),$$

with probability of not-filtering the wrong key candidate exactly 2^{-i}. Since $p > 1/2$, this test keeps keys with value S_m above a suitable integer threshold t_i, and additionally with probability $\rho_i \in [0, 1]$ if $S_m = t_i$, where ρ_i is chosen so that

$$\Pr_{1/2}[S_m > t_i] + \rho_i \Pr_{1/2}[S_m = t_i] = 2^{-i}.$$

For instance, for the `PRESENT` schedule of Table 2 targeting a reduction by $s = 2$ bits, the second stage uses $m_2 = 5$ structures, with $t_2 = 3$ and $\rho_2 = 0.2$.

Let m_i be the cumulative number of structures available at stage i, with $0 = m_0 \le m_1 \le \cdots \le m_s$ and $\Delta m_i := m_i - m_{i-1}$, so that Δm_i is the number of new structures evaluated at stage i. With these notations, the data complexity is therefore proportional to m_s, and at the beginning of stage i, the expected number of surviving wrong key candidates is about $2^{\ell-i+1}$, leading to

$$D_{\text{prob}}(s) = 2^d m_s \qquad T_{\text{prob}}(s) \approx 2^{\ell+d+1} \sum_{i=1}^{s} \Delta m_i \, 2^{-i}.$$

Only Δm_i new structures are evaluated for candidates that survived the previous stages. Since fewer candidates remain at later stages, each new structure costs less to evaluate, as reflected by the factor 2^{-i} in the time complexity.

A closed form approximation for m_i. The log-likelihood ratio, in bits, associated with the outcome Y of one integral structure is $L(Y) = Y \log_2(2p) + (1 - Y) \log_2(2(1-p))$. Under the correct key, its mean is

$$\mu = p \log_2(2p) + (1-p) \log_2(2(1-p)) = 1 - h_2(p),$$

where $h_2(p)$ is the binary entropy function. Thus, each structure contributes, on average, $1 - h_2(p)$ bits of evidence in favor of the correct key candidate. After m_i structures, the expected accumulated log-likelihood ratio is $m_i\mu$. Using i bits as a first-order target for the stage-i filter gives the following approximation for the cumulative number of structures: $m_i \simeq \frac{i}{1-h_2(p)}$. This approximation recovers the deterministic case: if $p = 1$, then $h_2(p) = 0$ and $m_i = i$, corresponding to one new structure at each stage. For $p = \frac{1}{2} + \varepsilon$ with $\varepsilon \to 0$,

$$1 - h_2\left(\frac{1}{2} + \varepsilon\right) = \frac{2\varepsilon^2}{\ln 2} + O(\varepsilon^4), \qquad m_i \simeq \frac{i \ln 2}{2(p - 1/2)^2}.$$

Thus, near the unbiased case, the estimated number of structures grows proportionally to $(p-1/2)^{-2}$. To include the desired probability of retaining the correct key, we use a second-order approximation. For $1/2 < p < 1$, let ϵ_i be the target probability that the correct key fails the test at stage i, and set $z_i = \Phi^{-1}(1-\epsilon_i)$, where Φ is the standard normal cumulative distribution function. The second-order approximation to the Neyman–Pearson test gives

$$i \simeq m_i\mu - z_i\sqrt{m_i v}, \qquad v = \mathrm{Var}_p[L(Y)] = p(1-p)\left(\log_2 \frac{p}{1-p}\right)^2,$$

where v is the variance of the log-likelihood ratio contributed by one structure under the correct key. Solving this approximation as a quadratic equation in $\sqrt{m_i}$ and rounding up gives

$$m_i \simeq \left\lceil (2\mu)^{-2} \left(z_i\sqrt{v} + \sqrt{z_i^2 v + 4\mu i} \right)^2 \right\rceil, \qquad \mu = 1 - h_2(p).$$

Again, when $p \to 1$ we have $\mu \to 1$ and $v \to 0$, and we recover the procedure of the deterministic case since $m_i \to i$.

Overall success probability and restarts. Let the total key-recovery success probability be $q = 1 - \delta$. The failure probability at each stage is $\epsilon_i = \frac{\delta\, 2^{-i}}{1-2^{-s}}$, for $i = 1, \ldots, s$, for which $\sum_{i=1}^{s} \epsilon_i = \delta$. An union bound then guarantees that the correct class survives all s filtering stages with probability at least q.

If the final candidates are verified by a full encryption or another attack, an unsuccessful run is detectable and the attack can be restarted. Since the actual success probability is at least q, independent runs require at most $1/q$ trials on average. Hence the overall data and time complexities are

$$\overline{T}_{\text{prob}}(s; q) \leq \frac{T_{\text{prob}}(s; q)}{q} \qquad \text{and} \qquad \overline{D}_{\text{prob}} \leq \frac{D_{\text{prob}}}{q}.$$

There is no reason to force q close to one: making a single run extremely reliable increases the m_i, whereas a moderate failure probability can be amortized by restarts. By expressing the $\overline{T}_{\text{prob}}(s; q)$ as a function of q and by numerical minimization we obtain a unique solution $q \simeq 0.729$. Thus, a target success probability of around 70–75% for the whole run is a useful default.[8]

As a useful rule of thumb, reducing the number of surviving candidates by one bit has two simple asymptotic regimes. When the distinguisher is sufficiently strong, say $p \geq 0.80$, a single structure already provides an effective filter: repeating the experiment until success requires on average p^{-1} trials, leading to

$$\overline{D} \simeq p^{-1} 2^d, \qquad \overline{T} \simeq p^{-1} 2^{d+\ell}.$$

On the other hand, when p is close to $1/2$, several independent structures must be combined in order to distinguish the correct key from a wrong one. Using the optimized restart probability $q \simeq 0.729$, the corresponding average complexities are approximately

$$\overline{D} \simeq \frac{1.286\, 2^d}{(p - \frac{1}{2})^2}, \qquad \overline{T} \simeq \frac{1.286\, 2^{d+\ell}}{(p - \frac{1}{2})^2}.$$

Thus, the cost smoothly interpolates between essentially one structure when p is close to 1, and the classical quadratic cost in $(p - \frac{1}{2})^{-1}$ when the bias is small.

5.2 Partial-key filtering costs for `PRESENT`

We illustrate these partial-key filtering costs on the same 6-round property of `PRESENT` with target $(j, \gamma) = (0, \texttt{0x1})$, comparing the deterministic configuration with $D = 2^8$ to the probabilistic one with $D = 2^4$. For the probabilistic configuration, we use $p = 0.8945$, the empirical mean reported in Table 4, giving $\varepsilon = p - \frac{1}{2} = 0.3945$. We append two rounds and partially decrypt to the target output bit. The direct dependency cone of this two-round partial decryption contains 20 round-key bits; exploiting the relations between admissible key differences described in [4] and [5], however, reduces the number of classes that

[8] For very small s, integer effects can shift the optimum noticeably.

Table 2. Partial-key filtering cost for the 2^{12}-representative PRESENT key-dependency space, reduced by s bits. For the probabilistic case, we use cumulative Neyman–Pearson filtering with $p = 0.8945$ and a schedule designed for a target success probability $q = 0.75$; q_{true} is the actual success probability of that schedule, and we report conservative expected complexities, $\overline{D} = D_{\text{run}}/q_{\text{true}}$ and $\overline{T} = T_{\text{run}}/q_{\text{true}}$.

Reduction s	$\log_2 D_{\text{det}}$	$\log_2 T_{\text{det}}$	$m_{s,s}$	q_{true}	$\log_2 \overline{D}_{\text{prob}}$	$\log_2 \overline{T}_{\text{prob}}$
1	8.00	20.00	1	0.894	4.16	16.16
2	9.00	20.58	5	0.852	6.55	17.82
4	10.00	20.91	15	0.847	8.15	18.63
8	11.00	20.99	39	0.845	9.53	18.82
12	11.58	21.00	62	0.845	10.20	18.83

actually need to be tested to only 2^{12} representatives, and all computations below use $\ell = 12$. For the probabilistic filter, the per-stage failure budgets ϵ_i are set from the design target $q = 0.75$, close to the generic optimum $q \simeq 0.729$ discussed above. For the deterministic property, reducing the key space by s bits gives

$$D_{\text{det}}(s) = s\,2^8, \qquad T_{\text{det}}(s) = 2^{21}\left(1 - 2^{-s}\right).$$

For the probabilistic property, let $m_{i,s}$ denote the cumulative number of structures evaluated by stage i when we target a reduction by s bits, with $m_{0,s} = 0$ and $\Delta m_{i,s} := m_{i,s} - m_{i-1,s}$. Then it holds that:

$$D_{\text{prob}}(s) = m_{s,s}2^4, \qquad T_{\text{prob}}(s) = 2^{17} \sum_{1 \le i \le s} \Delta m_{i,s} 2^{-i}.$$

Because both the union bound over stages and the exact-binomial thresholds are conservative, the resulting schedule actually succeeds with a probability q_{true} higher than the 0.75 design target; we compute q_{true} exactly, as the probability that the correct key survives every one of the s thresholds jointly, and report the expected complexities with detectable restarts obtained by dividing $D_{\text{prob}}(s)$ and $T_{\text{prob}}(s)$ by this actual q_{true} rather than by the 0.75 design target.

Table 2 highlights the partial-key filtering costs required to reduce the 2^{12} representatives by s bits. In practice, it is common to only decrease the number of key candidates by a small factor to then make new guesses and reach a better distinguisher. In some cases, several distinguishers can also be interleaved. In Appendix I we provide a similar comparison for GIFT-64 and GIFT-128, as well as a comparison between our 6-round probabilistic integral distinguisher for PRESENT and the seven-round differential-linear distinguisher of Hadipour et al. [17], both extended to 9 rounds for key recovery.

6 Conclusion

In this work, we introduced a probabilistic extension of the division property that describes how the values of the constant inputs affect the validity of division

trails. Under the assumptions of our model, the resulting no-trail probability gives a key-independent lower bound on the zero-sum probability. Our technique relies on a tree traversal to obtain bounds and approximations of this quantity, and uses an additional heuristic to predict the zero-sum probability.

Our applications to `PRESENT`, `GIFT`, and the `ASCON` permutation show how probabilistic integrals can reduce the number of inputs per structure. For 6-round `PRESENT`, our algebraic analysis explains how the interaction between the S-box and bit permutation restrict the combination of active variables, producing a large zero-sum bias. The `ASCON` results demonstrate the tool's applicability to rotation-and-XOR diffusion. For both `PRESENT` and `GIFT-64`, the smaller structures also reduce the estimated data and time complexities of key recovery within our statistical model.

Propagating more information about intermediate bits or linear combinations of bits that are constant over an integral structure could exclude more division trails and give tighter lower bounds on the zero-sum probability. This is a promising research direction. Another direction is to extend the probabilistic analysis to other iterated constructions beyond SPNs, with particular attention to the conditions ensuring that the constant assignments used by the model remain independent and uniform.

Acknowledgements. This work was partially supported by the France 2030 program under grant agreement No. ANR22-PECY-0010 Cryptanalyse, by the French Agence Nationale de la Recherche through the OREO project under Contract ANR-22-CE39-0015 and by the European Union (ERC-2023-COG, SoBaSyC, 101125450 and ERC-2022-ADG, SymTrust, 101097056). 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. Finally, the authors would like to thank the Dagstuhl Seminar 26061 on Symmetric Cryptography that gave the opportunity to the authors to initiate this collaboration.

Use of AI tools. The authors acknowledge that, while all of the ideas presented in this work are theirs, some AI tools were used to generate part of the contents of the present paper. This is the case of some preliminary version of current sections and figures that were largely reworked to convey their personal thoughts, and some of their experimental code. The authors accept responsibility for the veracity and correctness of all material that are presented, including any computer-generated material.

References

1. Aumasson, J.P., Dinur, I., Meier, W., Shamir, A.: Cube testers and key recovery attacks on reduced-round MD6 and Trivium. In: Dunkelman, O. (ed.) FSE 2009.

LNCS, vol. 5665, pp. 1–22. Springer, Berlin, Heidelberg (Feb 2009). https://doi.org/10.1007/978-3-642-03317-9_1
2. Banik, S., Pandey, S.K., Peyrin, T., Sasaki, Y., Sim, S.M., Todo, Y.: GIFT: A small present - towards reaching the limit of lightweight encryption. In: Fischer, W., Homma, N. (eds.) CHES 2017. LNCS, vol. 10529, pp. 321–345. Springer, Cham (Sep 2017). https://doi.org/10.1007/978-3-319-66787-4_16
3. Beierle, C., Hebborn, P., Leander, G., Perehuda, Y.: Integral resistance of block ciphers with key whitening by modular addition. In: Kalai, Y.T., Kamara, S.F. (eds.) CRYPTO 2025, Part V. LNCS, vol. 16004, pp. 497–529. Springer, Cham (Aug 2025). https://doi.org/10.1007/978-3-032-01901-1_16
4. Beierle, C., Leander, G., Perehuda, Y.: When the wrong key lives on: The key-recovery procedure in integral attacks. pp. 33–62. LNCS (2026). https://doi.org/10.1007/978-3-032-25333-0_2
5. Beyne, T., Leander, G., Neumann, P., Perehuda, Y., Verbauwhede, M.: What to guess in key-recovery attacks? Cryptology ePrint Archive, Paper 2026/1897 (2026), https://eprint.iacr.org/2026/1897
6. Beyne, T., Leander, G., Schütt, I.: Pairwise independence of AES-like block ciphers. pp. 181–210. LNCS (2026). https://doi.org/10.1007/978-3-032-25333-0_7
7. Beyne, T., Verbauwhede, M.: Integral cryptanalysis using algebraic transition matrices. IACR Trans. Symm. Cryptol. **2023**(4), 244–269 (2023). https://doi.org/10.46586/tosc.v2023.i4.244-269
8. Beyne, T., Verbauwhede, M.: Ultrametric integral cryptanalysis. In: Chung, K.M., Sasaki, Y. (eds.) ASIACRYPT 2024, Part VII. LNCS, vol. 15490, pp. 392–423. Springer, Singapore (Dec 2024). https://doi.org/10.1007/978-981-96-0941-3_13
9. Biham, E., Dunkelman, O., Keller, N.: New combined attacks on block ciphers. In: Gilbert, H., Handschuh, H. (eds.) FSE 2005. LNCS, vol. 3557, pp. 126–144. Springer, Berlin, Heidelberg (Feb 2005). https://doi.org/10.1007/11502760_9
10. 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, Berlin, Heidelberg (Sep 2007). https://doi.org/10.1007/978-3-540-74735-2_31
11. Boura, C., Canteaut, A.: Another view of the division property. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part I. LNCS, vol. 9814, pp. 654–682. Springer, Berlin, Heidelberg (Aug 2016). https://doi.org/10.1007/978-3-662-53018-4_24
12. Collard, B., Standaert, F.X.: A statistical saturation attack against the block cipher PRESENT. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 195–210. Springer, Berlin, Heidelberg (Apr 2009). https://doi.org/10.1007/978-3-642-00862-7_13
13. Daemen, J., Knudsen, L.R., Rijmen, V.: The block cipher Square. In: Biham, E. (ed.) FSE'97. LNCS, vol. 1267, pp. 149–165. Springer, Berlin, Heidelberg (Jan 1997). https://doi.org/10.1007/BFb0052343
14. Derbez, P., Fouque, P.A.: Increasing precision of division property. IACR Trans. Symm. Cryptol. **2020**(4), 173–194 (2020). https://doi.org/10.46586/tosc.v2020.i4.173-194
15. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon v1.2: Lightweight authenticated encryption and hashing. Journal of Cryptology **34**(3), 33 (Jul 2021). https://doi.org/10.1007/s00145-021-09398-9

16. Ferguson, N., Kelsey, J., Lucks, S., Schneier, B., Stay, M., Wagner, D., Whiting, D.: Improved cryptanalysis of Rijndael. In: Schneier, B. (ed.) FSE 2000. LNCS, vol. 1978, pp. 213–230. Springer, Berlin, Heidelberg (Apr 2001). https://doi.org/10.1007/3-540-44706-7_15
17. Hadipour, H., Derbez, P., Eichlseder, M.: Revisiting differential-linear attacks via a boomerang perspective with application to AES, ascon, CLEFIA, SKINNY, PRESENT, KNOT, TWINE, WARP, LBlock, simeck, and SERPENT. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part IV. LNCS, vol. 14923, pp. 38–72. Springer, Cham (Aug 2024). https://doi.org/10.1007/978-3-031-68385-5_2
18. Hebborn, P., Lambin, B., Leander, G., Todo, Y.: Strong and tight security guarantees against integral distinguishers. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021, Part I. LNCS, vol. 13090, pp. 362–391. Springer, Cham (Dec 2021). https://doi.org/10.1007/978-3-030-92062-3_13
19. Hebborn, P., Leander, G., Udovenko, A.: Mathematical aspects of division property. Cryptogr. Commun. **15**(4), 731–774 (2023). https://doi.org/10.1007/S12095-022-00622-2
20. Heys, H.M.: Integral cryptanalysis of the BSPN block cipher. In: Proceedings of the Queen's Biennial Symposium on Communications (QBSC). IEEE (2014). https://doi.org/10.1109/QBSC.2014.6841204
21. Hu, K., Peyrin, T., Tan, Q.Q., Yap, T.: Revisiting higher-order differential-linear attacks from an algebraic perspective. In: Guo, J., Steinfeld, R. (eds.) ASIACRYPT 2023, Part III. LNCS, vol. 14440, pp. 405–435. Springer, Singapore (Dec 2023). https://doi.org/10.1007/978-981-99-8727-6_14
22. Hu, K., Sun, S., Wang, M., Wang, Q.: An algebraic formulation of the division property: Revisiting degree evaluations, cube attacks, and key-independent sums. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part I. LNCS, vol. 12491, pp. 446–476. Springer, Cham (Dec 2020). https://doi.org/10.1007/978-3-030-64837-4_15
23. Knudsen, L.R., Wagner, D.: Integral cryptanalysis. In: Daemen, J., Rijmen, V. (eds.) FSE 2002. LNCS, vol. 2365, pp. 112–127. Springer, Berlin, Heidelberg (Feb 2002). https://doi.org/10.1007/3-540-45661-9_9
24. Lambin, B., Derbez, P., Fouque, P.A.: Linearly equivalent S-boxes and the division property. DCC **88**(10), 2207–2231 (2020). https://doi.org/10.1007/s10623-020-00773-4
25. Leander, G.: On linear hulls, statistical saturation attacks, PRESENT and a cryptanalysis of PUFFIN. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 303–322. Springer, Berlin, Heidelberg (May 2011). https://doi.org/10.1007/978-3-642-20465-4_18
26. Liu, T., Pelecanos, A., Tessaro, S., Vaikuntanathan, V.: Layout graphs, random walks and the t-wise independence of SPN block ciphers. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part III. LNCS, vol. 14083, pp. 694–726. Springer, Cham (Aug 2023). https://doi.org/10.1007/978-3-031-38548-3_23
27. Liu, T., Tessaro, S., Vaikuntanathan, V.: The t-wise independence of substitution-permutation networks. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part IV. LNCS, vol. 12828, pp. 454–483. Springer, Cham, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84259-8_16
28. Rohit, R., Hu, K., Sarkar, S., Sun, S.: Misuse-free key-recovery and distinguishing attacks on 7-round Ascon. IACR Trans. Symm. Cryptol. **2021**(1), 130–155 (2021). https://doi.org/10.46586/tosc.v2021.i1.130-155

29. Todo, Y.: Structural evaluation by generalized integral property. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 287–314. Springer, Berlin, Heidelberg (Apr 2015). https://doi.org/10.1007/978-3-662-46800-5_12
30. Todo, Y., Aoki, K.: FFT key recovery for integral attack. In: Gritzalis, D., Kiayias, A., Askoxylakis, I.G. (eds.) CANS 14. LNCS, vol. 8813, pp. 64–81. Springer, Cham (Oct 2014). https://doi.org/10.1007/978-3-319-12280-9_5
31. Todo, Y., Morii, M.: Bit-based division property and application to simon family. In: Peyrin, T. (ed.) FSE 2016. LNCS, vol. 9783, pp. 357–377. Springer, Berlin, Heidelberg (Mar 2016). https://doi.org/10.1007/978-3-662-52993-5_18
32. Xiang, Z., Zhang, W., Bao, Z., Lin, D.: Applying MILP method to searching integral distinguishers based on division property for 6 lightweight block ciphers. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part I. LNCS, vol. 10031, pp. 648–678. Springer, Berlin, Heidelberg (Dec 2016). https://doi.org/10.1007/978-3-662-53887-6_24
33. Zhang, L., Yao, Y., Shi, D., Chai, D., Guo, J., Wang, Z.: Neural-inspired advances in integral cryptanalysis. pp. 451–481. LNCS (2026). https://doi.org/10.1007/978-3-032-25333-0_16

Supplementary Material

A Computing validity sets

Example 2 (Validity sets). Let $S : \mathbb{F}_2^3 \to \mathbb{F}_2^2$ be the S-box defined by

$$S(x_0, x_1, x_2) = (x_0x_1 \oplus x_2 \oplus x_0x_2,\ x_1 \oplus x_0x_2)\,.$$

As in Example 1, let $c = \mathtt{110}$, using the bit ordering (x_2, x_1, x_0). The corresponding restricted ANFs are given in the following table where the output masks are written in the order (v_1v_0).

α	$S(x)^{\mathtt{00}}\big\vert_\alpha$	$S(x)^{\mathtt{01}}\big\vert_\alpha$	$S(x)^{\mathtt{10}}\big\vert_\alpha$	$S(x)^{\mathtt{11}}\big\vert_\alpha$
000	1	0	0	0
010	1	x_0	1	x_0
100	1	$x_0 \oplus 1$	x_0	0
110	1	1	$x_0 \oplus 1$	$x_0 \oplus 1$

The monomial x_0 appears in $S(x)^{\mathtt{01}}\big\vert_\alpha$ precisely for $\alpha \in \{\mathtt{010}, \mathtt{100}\}$. Therefore, $\mathcal{V}_c^S(\mathtt{001}, \mathtt{01}) = \{\mathtt{010}, \mathtt{100}\}$. Similarly, $\mathcal{V}_c^S(\mathtt{001}, \mathtt{10}) = \{\mathtt{100}, \mathtt{110}\}$.

For the output mask 11, the monomial x_0 appears in $S(x)^{\mathtt{11}}\big\vert_\alpha$ only for $\alpha \in \{\mathtt{010}, \mathtt{110}\}$. However, because we consider weak transitions, the transition to 11 may also be witnessed by either $\mathtt{01} \preceq \mathtt{11}$ or $\mathtt{10} \preceq \mathtt{11}$. Consequently, $\mathcal{V}_c^S(\mathtt{001}, \mathtt{11}) = \{\mathtt{010}, \mathtt{100}, \mathtt{110}\}$. Finally, $S(x)^{\mathtt{00}}\big\vert_\alpha = 1$ for every $\alpha \in \mathcal{A}_c$. Hence, $\mathcal{V}_c^S(\mathtt{000}, \mathtt{00}) = \mathcal{A}_c$. Since $\mathtt{00} \preceq v$ for every $v \in \mathbb{F}_2^2$, it follows from the definition of a weak transition that $\mathcal{V}_c^S(\mathtt{000}, v) = \mathcal{A}_c$ for every $v \in \mathbb{F}_2^2$.

B Extended propagation tables: an example

Consider as in all previous examples the S-box $S(x_0, x_1, x_2) = \big(x_0x_1 \oplus x_2 \oplus x_0x_2,\ x_1 \oplus x_0x_2\big)$. We first complete the construction of the extended propagation table for the mask $c = \mathtt{110}$ considered in Example 2. We then consider $c = \mathtt{100}$ to illustrate how redundant entries are removed.

Case $c = \mathtt{110}$. Recall that x_1 and x_2 are fixed, whereas x_0 remains active, and

$$\mathcal{A}_c = \{\mathtt{000}, \mathtt{010}, \mathtt{100}, \mathtt{110}\}\,.$$

In Example 2, we obtained the following validity sets.

u	$\mathcal{V}_c^S(u, \mathtt{00})$	$\mathcal{V}_c^S(u, \mathtt{01})$	$\mathcal{V}_c^S(u, \mathtt{10})$	$\mathcal{V}_c^S(u, \mathtt{11})$
000	$\mathcal{A}_c$	$\mathcal{A}_c$	$\mathcal{A}_c$	$\mathcal{A}_c$
001	$\varnothing$	$\{\mathtt{010}, \mathtt{100}\}$	$\{\mathtt{100}, \mathtt{110}\}$	$\{\mathtt{010}, \mathtt{100}, \mathtt{110}\}$

For $u = \texttt{000}$, all output masks have the same validity set $\mathcal{A}_c$. Since $\texttt{00}$ is the smallest output mask, only the pair $(\texttt{00}, \mathcal{A}_c)$ is retained. For $u = \texttt{001}$, the entry corresponding to $\texttt{00}$ is omitted because its validity set is empty. The remaining entries are incomparable under $\sqsubseteq$ and are therefore all retained. Consequently, the extended propagation table is

u	v	$\mathcal{V}_c^S(u, v)$
$\texttt{000}$	$\texttt{00}$	$\mathcal{A}_c$
$\texttt{001}$	$\texttt{01}$	$\{\texttt{010}, \texttt{100}\}$
	$\texttt{10}$	$\{\texttt{100}, \texttt{110}\}$
	$\texttt{11}$	$\{\texttt{010}, \texttt{100}, \texttt{110}\}$

For instance, the entries corresponding to $\texttt{001} \to \texttt{01}$ and $\texttt{001} \to \texttt{11}$ are incomparable. Indeed, although $\texttt{01} \preceq \texttt{11}$, the required reverse inclusion between their validity sets does not hold:

$$\mathcal{V}_c^S(\texttt{001}, \texttt{01}) \not\supseteq \mathcal{V}_c^S(\texttt{001}, \texttt{11}).$$

Case $c = \texttt{100}$. We now fix only x_2, while x_0 and x_1 remain active. Hence, $\mathcal{A}_c = \{\texttt{000}, \texttt{100}\}$. The corresponding restricted ANFs are shown below.

α	$S(x)^{\texttt{00}}\big\vert_\alpha$	$S(x)^{\texttt{01}}\big\vert_\alpha$	$S(x)^{\texttt{10}}\big\vert_\alpha$	$S(x)^{\texttt{11}}\big\vert_\alpha$
$\texttt{000}$	1	x_0x_1	x_1	x_0x_1
$\texttt{100}$	1	$1 \oplus x_0 \oplus x_0x_1$	$x_0 \oplus x_1$	$x_1 \oplus x_0x_1$

The admissible input masks are $\texttt{000}$, $\texttt{001}$, $\texttt{010}$, and $\texttt{011}$. From the restricted ANFs, we obtain the following validity sets.

u	$\mathcal{V}_c^S(u, \texttt{00})$	$\mathcal{V}_c^S(u, \texttt{01})$	$\mathcal{V}_c^S(u, \texttt{10})$	$\mathcal{V}_c^S(u, \texttt{11})$
$\texttt{000}$	$\mathcal{A}_c$	$\mathcal{A}_c$	$\mathcal{A}_c$	$\mathcal{A}_c$
$\texttt{001}$	$\varnothing$	$\mathcal{A}_c$	$\{\texttt{100}\}$	$\mathcal{A}_c$
$\texttt{010}$	$\varnothing$	$\mathcal{A}_c$	$\mathcal{A}_c$	$\mathcal{A}_c$
$\texttt{011}$	$\varnothing$	$\mathcal{A}_c$	$\varnothing$	$\mathcal{A}_c$

For $u = \texttt{000}$, only the smallest output mask $\texttt{00}$ is retained. For $u = \texttt{001}$, the entry associated with $\texttt{11}$ is removed because $\texttt{01} \preceq \texttt{11}$ and both masks have the same validity set $\mathcal{A}_c$. Similarly, the entry associated with $\texttt{11}$ is redundant for $u = \texttt{010}$ and $u = \texttt{011}$. We therefore obtain

u	v	$\mathcal{V}_c^S(u, v)$
$\texttt{000}$	$\texttt{00}$	$\mathcal{A}_c$
$\texttt{001}$	$\texttt{01}$	$\mathcal{A}_c$
	$\texttt{10}$	$\{\texttt{100}\}$
$\texttt{010}$	$\texttt{01}$	$\mathcal{A}_c$
	$\texttt{10}$	$\mathcal{A}_c$
$\texttt{011}$	$\texttt{01}$	$\mathcal{A}_c$

This example illustrates the two steps involved in constructing an extended propagation table. First, the restricted ANFs determine the validity set associated with each transition. Second, the resulting pairs $(v, \mathcal{V}_c^S(u,v))$ are reduced with respect to $\sqsubseteq$. In particular, for a fixed input mask u, an entry associated with an output mask v' is removed only if there exists a strictly smaller output mask $v \preceq v'$ having exactly the same validity set.

The entries $(\mathtt{001}, \mathtt{01})$ and $(\mathtt{010}, \mathtt{01})$ could also be inferred from $(\mathtt{011}, \mathtt{01})$, since $\mathtt{001} \preceq \mathtt{011}, \mathtt{010} \preceq \mathtt{011}$, and all three entries have validity set $\mathcal{A}_c$. Therefore, this table is not "maximally compressed" and an additional "input-side" reduction could therefore retain only the maximal input masks having a given output mask and validity set. We do not use this other kind of compression in our implementation of the propagation tables.

C Example of the no-trail probability computation

We illustrate the recursive procedure of Section 3.3 on a toy example. The example shows how the admissible assignment set is updated and how the resulting branches contribute to the no-trail probability.

Example 3. Consider the S-box $S : \mathbb{F}_2^3 \to \mathbb{F}_2^2$ defined by $S(x_0, x_1, x_2) = \big(x_0x_1 \oplus x_2 \oplus x_0x_2,\ x_1 \oplus x_0x_2\big)$, with fixed-input mask $c = \mathtt{110}$. Thus x_1 and x_2 are fixed, while x_0 remains active. The fixed-input assignment α is uniformly distributed over $\mathcal{A}_c = \{\mathtt{000}, \mathtt{010}, \mathtt{100}, \mathtt{110}\}$.

For the input mask $u = \mathtt{001}$, the relevant validity sets are

$$\mathcal{V}_c^S(\mathtt{001}, \mathtt{01}) = \{\mathtt{010}, \mathtt{100}\},$$
$$\mathcal{V}_c^S(\mathtt{001}, \mathtt{10}) = \{\mathtt{100}, \mathtt{110}\}.$$

Let $L(y_0, y_1) = y_0 \oplus y_1$ and consider the composition $F = L \circ S$. Writing $z = L(y_0, y_1)$, the output mask $\mathtt{1}$ represents the monomial z. Its ANF is $z = y_0 \oplus y_1$. The input masks $\mathtt{01}$ and $\mathtt{10}$ represent y_0 and y_1, respectively. Both monomials occur in this ANF, so both masks admit a transition to $\mathtt{1}$. By contrast, the input mask $\mathtt{11}$ represents y_0y_1. No monomial in the ANF of L is divisible by this product, so $\mathtt{11} \to \mathtt{1}$ is impossible. Moreover, $\mathtt{001} \to \mathtt{00}$ is impossible through S. Consequently, there are exactly two candidate division trails from $\mathtt{001}$ to $\mathtt{1}$:

$$\tau_1 : \quad \mathtt{001} \xrightarrow{S} \mathtt{01} \xrightarrow{L} \mathtt{1},$$
$$\tau_2 : \quad \mathtt{001} \xrightarrow{S} \mathtt{10} \xrightarrow{L} \mathtt{1}.$$

Since this example involves only one S-box instance with fixed inputs, we denote its admissible assignment set by C_S. Initially, $C_S = \mathcal{A}_c$.

Initially, both τ_1 and τ_2 are candidate trails. Suppose that the trail-search algorithm selects τ_1 first. Its only probabilistic transition is $\mathtt{001} \xrightarrow{S} \mathtt{01}$. The corresponding *in* branch has conditional probability

$$p_{\text{in}} = \frac{|C_S \cap \mathcal{V}_c^S(\mathtt{001}, \mathtt{01})|}{|C_S|} = \frac{2}{4} = \frac{1}{2}.$$

In this branch, the admissible set becomes

$$C_S \leftarrow C_S \cap \mathcal{V}_c^S(\mathtt{001}, \mathtt{01}) = \{\mathtt{010}, \mathtt{100}\}.$$

The selected transition is now valid for every remaining assignment. Since the transition through L is also deterministic, τ_1 is a deterministic valid trail. This is therefore a deterministic-trail leaf with conditional value 0.

The *out* branch also has probability $1/2$. In this branch,

$$C_S \leftarrow C_S \setminus \mathcal{V}_c^S(\mathtt{001}, \mathtt{01}) = \{\mathtt{000}, \mathtt{110}\}.$$

For every assignment remaining in C_S, the transition through S in τ_1 is invalid, so τ_1 cannot be a valid trail. The search therefore looks for another trail that is valid for at least one assignment in the updated set C_S. The only such trail is τ_2.

The validity set of the transition through S in τ_2 is $\mathcal{V}_c^S(\mathtt{001}, \mathtt{10}) = \{\mathtt{100}, \mathtt{110}\}$. Its probability must be evaluated conditionally on the current admissible set $C_S = \{\mathtt{000}, \mathtt{110}\}$. Hence,

$$p_{\text{in}} = \frac{|\{\mathtt{000}, \mathtt{110}\} \cap \{\mathtt{100}, \mathtt{110}\}|}{|\{\mathtt{000}, \mathtt{110}\}|} = \frac{1}{2}.$$

In the corresponding *in* branch, $C_S = \{\mathtt{110}\}$. The trail τ_2 is then deterministically valid, giving another deterministic-trail leaf with conditional value 0.

The corresponding *out* branch also has conditional probability $1/2$ and gives $C_S = \{\mathtt{000}\}$. Both τ_1 and τ_2 are invalid for this assignment, and no other division trail exists. This is therefore a no-trail leaf with conditional value 1. Its contribution to the probability computed at the root is the probability of reaching this leaf. The complete recursion tree is shown in Figure 2.

The only root-to-leaf path ending at a no-trail leaf consists of two successive *out* decisions. Its probability is $\frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$. Therefore,

$$P(\mathcal{C}_0) = \Pr_{\alpha}[\text{no valid division trail}] = \frac{1}{4}.$$

Equivalently, applying the recursive relation gives

$$P(\mathcal{C}_0) = \frac{1}{2} \cdot 0 + \frac{1}{2}\left(\frac{1}{2} \cdot 0 + \frac{1}{2} \cdot 1\right) = \frac{1}{4}.$$

This value can also be checked directly from the four assignments in $\mathcal{A}_c$. For $\alpha = \mathtt{010}$, only τ_1 is valid; for $\alpha = \mathtt{110}$, only τ_2 is valid; and for $\alpha = \mathtt{100}$, both trails are valid. The assignment $\alpha = \mathtt{000}$ is the only one for which neither trail is valid. Thus exactly one of the four assignments leads to the absence of a valid division trail.

Finally, this example also illustrates the distinction between the no-trail probability and the probability that the integral sum vanishes. Indeed,

$$F(x_0, x_1, x_2) = x_0x_1 \oplus x_2 \oplus x_1,$$
$$\bigoplus_{x_0 \in \mathbb{F}_2} F(x_0, x_1, x_2) = x_1.$$

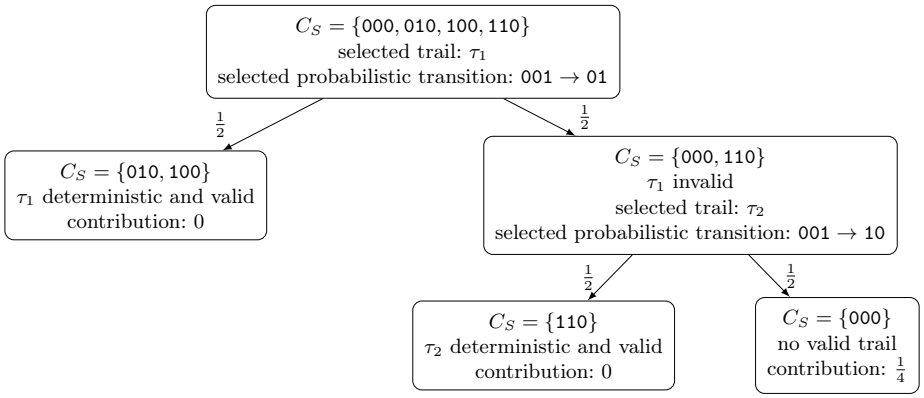

Fig. 2. Recursion tree for the composition $L \circ S$ with $c = 110$ and input mask 001. At each internal node, the left (*in*) branch makes the selected transition deterministically valid, whereas the right (*out*) branch makes it impossible. Edge labels give conditional probabilities under the current admissible set C_S. Leaf contributions include the probability of reaching the leaf from the root.

The sum therefore vanishes for both $\alpha = 000$ and $\alpha = 100$, giving a zero-sum probability of $1/2$. For $\alpha = 100$, both trails are valid, but their contributions cancel. Hence the no-trail probability $1/4$ is a strict lower bound on the zero-sum probability.

D Approximation below a probability cutoff

D.1 Our heuristic

The exhaustive recursive procedure terminates and computes exactly the probability that no valid division trail exists. In practice, however, the recursion tree may be too large to explore completely. We therefore introduce a probability cutoff $\delta \in]0, 1[$, below which exhaustive exploration is replaced by the approximate procedure described below. Each node is reached from the root with probability W, obtained by multiplying the conditional edge probabilities along its path ($W = 1$ at the root); it is used only to decide whether exploration remains exact or switches to the approximation. Since W cannot increase along a path, once a node falls below the cutoff, so does every node below it.

At each node, the algorithm returns certified bounds L and U with $L \leq P(\mathcal{C}) \leq U$, together with a heuristic estimate $\widehat{P}$ of $P(\mathcal{C})$. At a no-trail leaf, $P(\mathcal{C}) = 1$ exactly, so $(L, U) = (1, 1)$ and $\widehat{P} = 1$; at a deterministic-trail leaf, $(L, U) = (0, 0)$ and $\widehat{P} = 0$.

Consider a probabilistic transition $u \to v$ of the currently selected trail τ through an S-box S, and let $\mathcal{V} = \mathcal{V}_c^S(u, v)$ be its validity set. Under the current

admissible assignment set C_S, its conditional *in* and *out* probabilities are

$$p_{\text{in}} = \frac{|C_S \cap \mathcal{V}|}{|C_S|}, \qquad p_{\text{out}} = \frac{|C_S \setminus \mathcal{V}|}{|C_S|}.$$

If $W \geq \delta$ at the current node, both branches are explored exactly, and the bounds and estimate of the children – reached with probability $W \times p_{\text{in}}$ and $W \times p_{\text{out}}$, respectively – are combined by the law of total probability:

$$L = p_{\text{in}} L_{\text{in}} + p_{\text{out}} L_{\text{out}}, \qquad U = p_{\text{in}} U_{\text{in}} + p_{\text{out}} U_{\text{out}}, \qquad \widehat{P} = p_{\text{in}} \widehat{P}_{\text{in}} + p_{\text{out}} \widehat{P}_{\text{out}}.$$

Since both branches are followed exactly here, L and U coincide with $\widehat{P}$: no approximation has yet been introduced.

Once $W < \delta$, we retain only the *out* branch of the selected transition. Let

$$p_{\text{inv}}(\tau \mid \mathcal{C}) = \Pr[\text{at least one transition of } \tau \text{ is invalid} \mid \mathcal{C}].$$

Invalidating the selected transition also invalidates τ, but τ may be invalidated by another transition as well. Therefore, $p_{\text{out}} \leq p_{\text{inv}}(\tau \mid \mathcal{C})$ and the gap $p_{\text{inv}}(\tau \mid \mathcal{C}) - p_{\text{out}}$ corresponds to the other, unexplored ways of invalidating τ. Both probabilities are evaluated under the parent node's constraints, before the admissible sets are updated. We define

$$L = p_{\text{out}} L_{\text{out}}, \qquad U = p_{\text{out}} U_{\text{out}} + \big(p_{\text{inv}}(\tau \mid \mathcal{C}) - p_{\text{out}}\big), \qquad \widehat{P} = p_{\text{inv}}(\tau \mid \mathcal{C})\, \widehat{P}_{\text{out}}.$$

The lower bound L credits only the mass certified by the explored *out* branch: every other case – in particular, whenever τ remains entirely valid – is conservatively assumed not to yield a no-trail event. The upper bound U instead credits every unexplored way of invalidating τ optimistically, by treating it as leading to the no-trail event. The only case that is excluded is the one where τ remains entirely valid, since then τ itself provides a valid trail. Both L and U are therefore exact bounds on $P(\mathcal{C})$, unlike $\widehat{P}$, which heuristically extrapolates the explored branch's estimate $\widehat{P}_{\text{out}}$ to the whole probability $p_{\text{inv}}(\tau \mid \mathcal{C})$ of invalidating τ.

This approximation treats the explored *out* branch as representative of the other ways of invalidating τ. It is heuristic because these other cases may leave different alternative trails available.

Example 4 (Motivating our heuristic). To explain the motivation for this approximation, suppose that the current trail τ contains exactly three probabilistic transitions, ordered as t_0, t_1, t_2, with t_0 being the transition selected for branching. We partition the event that τ is invalid according to the first invalid transition in this order.

For $i \in \{0, 1, 2\}$, let $\mathcal{C}_{\text{out}}^{(i)}$ denote the collection obtained by conditioning on $t_0, \ldots, t_{i-1}$ being valid and t_i being invalid. For $i = 0$, this simply means conditioning on t_0 being invalid. Write $p_{\text{in}}^{(i)}$ and $p_{\text{out}}^{(i)}$ for the validity and invalidity probabilities of t_i, conditioned on the current constraints and on all preceding transitions in this order being valid. The three cases then have conditional probabilities

$$w_0 = p_{\text{out}}^{(0)}, \quad w_1 = p_{\text{in}}^{(0)} p_{\text{out}}^{(1)}, \quad w_2 = p_{\text{in}}^{(0)} p_{\text{in}}^{(1)} p_{\text{out}}^{(2)}.$$

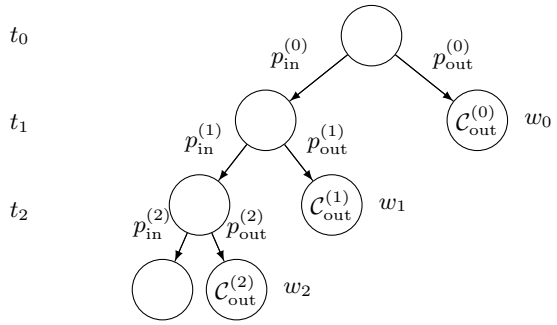

Fig. 3. Example of the cut-off heuristic.

This situation is depicted in Figure 3.

These cases are disjoint and cover all assignments for which τ is invalid. Therefore, $p_{\text{inv}}(\tau \mid \mathcal{C}) = w_0 + w_1 + w_2$. If all three transitions are valid, then τ is valid and the no-trail event cannot occur. Consequently, the exact conditional no-trail probability is

$$P(\mathcal{C}) = \sum_{i=0}^{2} w_i P\left(\mathcal{C}_{\text{out}}^{(i)}\right).$$

Below the cutoff, we explore only the case represented by $\mathcal{C}_{\text{out}}^{(0)}$. The approximation treats its conditional no-trail probability as representative of those in the other two cases: $P(\mathcal{C}_{\text{out}}^{(i)}) \approx P(\mathcal{C}_{\text{out}}^{(0)}), i = 1, 2$. This gives

$$P(\mathcal{C}) \approx (w_0 + w_1 + w_2) P(\mathcal{C}_{\text{out}}^{(0)}) = p_{\text{inv}}(\tau \mid \mathcal{C}) P(\mathcal{C}_{\text{out}}^{(0)}).$$

This is the substitution implemented in $\widehat{P} = p_{\text{inv}}(\tau \mid \mathcal{C})\, \widehat{P}_{\text{out}}$, which extrapolates the estimate of the selected *out* branch to the whole probability of invalidating τ.

D.2 Pseudocode for the probability-tree search

Algorithm 1 describes the recursive search for probabilistic integral distinguishers. It returns certified lower and upper bounds on the no-valid-trail probability, together with a heuristic estimate.

E Specifications of `PRESENT` and `GIFT`

E.1 `PRESENT`

`PRESENT` was proposed by Bogdanov *et al.* in 2007 [10]. Its two variants, `PRESENT-80` and `PRESENT-128`, differ only in their master-key size and key schedule. This appendix describes the round components and the conventions used in our analysis.

Algorithm 1 Certified bounds and a heuristic estimate of the no-trail probability

Require: Current constraints $\mathcal{C}$, exact node mass W, cutoff δ
Ensure: Certified bounds (L, U) such that $L \le P(\mathcal{C}) \le U$ and an estimated value $\widehat{P}$ of $P(\mathcal{C})$

$\tau \leftarrow \textsc{MostProbableTrail}(\mathcal{C})$
if $\tau = \perp$ **then return** $(1,1)$**,** 1 ▷ No valid trail exists
$p_\tau \leftarrow \prod_{t \in \tau} p_{\text{in}}(t)$
if $p_\tau = 1$ **then return** $(0,0)$**,** 0 ▷ τ is deterministic under $\mathcal{C}$
$t^\star \leftarrow \arg\max_{t \in \tau} p_{\text{out}}(t)$
$(p_{\text{in}}, p_{\text{out}}) \leftarrow (p_{\text{in}}(t^\star), p_{\text{out}}(t^\star))$
if $W \ge \delta$ **then** ▷ Exact binary branching
    $\mathcal{C}_{\text{in}} \leftarrow \textsc{Restrict}(\mathcal{C}, t^\star, \text{IN})$
    $\mathcal{C}_{\text{out}} \leftarrow \textsc{Restrict}(\mathcal{C}, t^\star, \text{OUT})$
    $(L_{\text{in}}, U_{\text{in}}), \widehat{P}_{\text{in}} \leftarrow \textsc{CertifiedBounds}(\mathcal{C}_{\text{in}}, W \times p_{\text{in}}, \delta)$
    $(L_{\text{out}}, U_{\text{out}}), \widehat{P}_{\text{out}} \leftarrow \textsc{CertifiedBounds}(\mathcal{C}_{\text{out}}, W \times p_{\text{out}}, \delta)$
    $L \leftarrow p_{\text{in}} L_{\text{in}} + p_{\text{out}} L_{\text{out}}$
    $U \leftarrow p_{\text{in}} U_{\text{in}} + p_{\text{out}} U_{\text{out}}$
    $\widehat{P} \leftarrow p_{\text{in}} \widehat{P}_{\text{in}} + p_{\text{out}} \widehat{P}_{\text{out}}$
    return $(L, U), \widehat{P}$
else ▷ Keep only one exact failure branch
    $q_\tau \leftarrow 1 - p_\tau$
    $\mathcal{C}_{\text{out}} \leftarrow \textsc{Restrict}(\mathcal{C}, t^\star, \text{OUT})$
    $(L_{\text{out}}, U_{\text{out}}), \widehat{P}_{\text{out}} \leftarrow \textsc{CertifiedBounds}(\mathcal{C}_{\text{out}}, W \times p_{\text{out}}, \delta)$
    $L \leftarrow p_{\text{out}} L_{\text{out}}$
    $U \leftarrow p_{\text{out}} U_{\text{out}} + (q_\tau - p_{\text{out}})$
    $\widehat{P} \leftarrow q_\tau \widehat{P}_{\text{out}}$
    return $(L, U), \widehat{P}$
end if

Round structure. `PRESENT` consists of 31 rounds followed by a final key addition. Each round applies the following operations:

- addition of the 64-bit round key by bitwise XOR;
- a substitution layer consisting of 16 parallel applications of a 4-bit S-box S;
- a bit-permutation layer.

Two consecutive rounds are shown in Figure 4.

Numbering conventions. State bits are numbered from 0 to 63, starting from the least significant bit. Within each nibble, we write $x = (x_3, x_2, x_1, x_0)$ and $S(x) = (S_3(x), S_2(x), S_1(x), S_0(x))$, where x_0 and $S_0(x)$ are the least significant input and output bits, respectively.

In all figures, bit positions within a nibble are numbered from right to left, with bit 0 on the right. The sixteen S-boxes are also numbered from right to left, from 0 to 15. Thus, S-box j processes state bits $4j, \ldots, 4j+3$.

S-box. The PRESENT S-box is defined by the following lookup table, whose entries are written in hexadecimal:

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

Under the above bit convention, its coordinate functions have the following algebraic normal forms (ANFs):

$$\begin{aligned}
S_0(x) &= x_0 \oplus x_1x_2 \oplus x_2 \oplus x_3,\\
S_1(x) &= x_0x_1x_2 \oplus x_0x_1x_3 \oplus x_0x_2x_3 \oplus x_1x_3 \oplus x_1 \oplus x_2x_3 \oplus x_3,\\
S_2(x) &= x_0x_1x_3 \oplus x_0x_1 \oplus x_0x_2x_3 \oplus x_0x_3 \oplus x_1x_3 \oplus x_2 \oplus x_3 \oplus 1,\\
S_3(x) &= x_0x_1x_2 \oplus x_0x_1x_3 \oplus x_0x_2x_3 \oplus x_0 \oplus x_1x_2 \oplus x_1 \oplus x_3 \oplus 1.
\end{aligned}$$

Permutation layer. The permutation moves the bit at position i to position

$$P(i) = \begin{cases} 16i \bmod 63, & 0 \le i \le 62,\\ 63, & i = 63. \end{cases}$$

Equivalently,

$$P(4j + b) = j + 16b, \qquad 0 \le j \le 15, \quad 0 \le b \le 3.$$

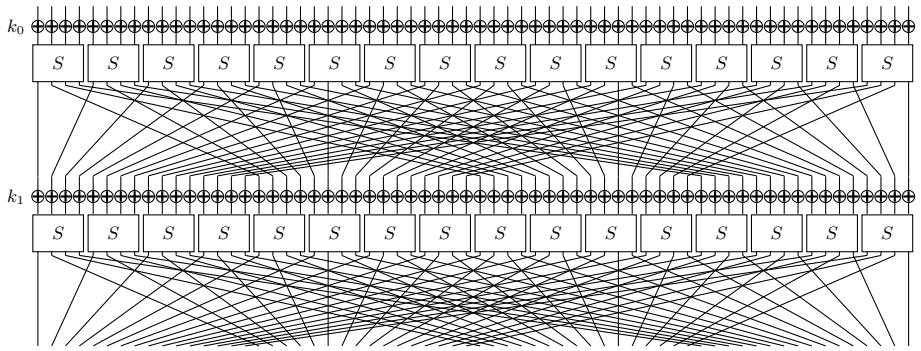

Fig. 4. Two consecutive rounds of PRESENT.

E.2 GIFT

We recall here the details of GIFT required to reproduce our experiments. Let $n \in \{64, 128\}$ be the state size and let $s = n/4$ be the number of 4-bit S-boxes.

The state is represented as

$$X = w_{s-1} \| w_{s-2} \| \cdots \| w_0,$$

where each nibble $w_i = (x_3, x_2, x_1, x_0)$ is processed independently by the same 4-bit S-box. Thus, the substitution layer contains 16 parallel S-boxes for `GIFT-64` and 32 for `GIFT-128`.

Bits within a nibble are numbered from right to left: x_0 is the least significant input bit, and $S_0(x)$ is the least significant output bit. Likewise, S-boxes are numbered from right to left, starting from S-box 0, which processes the least significant nibble. State bits are numbered from 0 to $n-1$ according to the same convention.

The `GIFT` S-box is defined by the following lookup table.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	1	A	4	C	6	F	3	9	2	D	B	7	5	0	8	E

The substitution layer is followed by a fixed bit permutation. For $n \in \{64, 128\}$, the output bit at position i, with $0 \le i < n$, is moved to position

$$P_n(i) = 4 \left\lfloor \frac{i}{16} \right\rfloor + \frac{n}{4} \left(\left(3 \left\lfloor \frac{i \bmod 16}{4} \right\rfloor + (i \bmod 4) \right) \bmod 4 \right) + (i \bmod 4).$$

In particular, the factor $n/4$ is equal to 16 for `GIFT-64` and to 32 for `GIFT-128`.

The round key is XORed into only two bits of each state nibble. Let the extracted round key be written as $RK = U \| V$, where $U = (u_{s-1}, \ldots, u_0)$ and $V = (v_{s-1}, \ldots, v_0)$.

For `GIFT-64`, the round-key addition is

$$b_{4i+1} \leftarrow b_{4i+1} \oplus u_i, \qquad b_{4i} \leftarrow b_{4i} \oplus v_i, \qquad 0 \le i < 16.$$

For `GIFT-128`, it is

$$b_{4i+2} \leftarrow b_{4i+2} \oplus u_i, \qquad b_{4i+1} \leftarrow b_{4i+1} \oplus v_i, \qquad 0 \le i < 32.$$

Finally, a fixed bit and a 6-bit round constant $C = (c_5, c_4, c_3, c_2, c_1, c_0)$ are XORed into the state according to

$$b_{n-1} \leftarrow b_{n-1} \oplus 1$$

and

$$\begin{aligned} b_{23} &\leftarrow b_{23} \oplus c_5, \quad b_{19} \leftarrow b_{19} \oplus c_4, \\ b_{15} &\leftarrow b_{15} \oplus c_3, \quad b_{11} \leftarrow b_{11} \oplus c_2, \\ b_7 &\leftarrow b_7 \oplus c_1, \quad b_3 \leftarrow b_3 \oplus c_0. \end{aligned}$$

Two consecutive rounds of `GIFT-64` are shown in Figure 5.

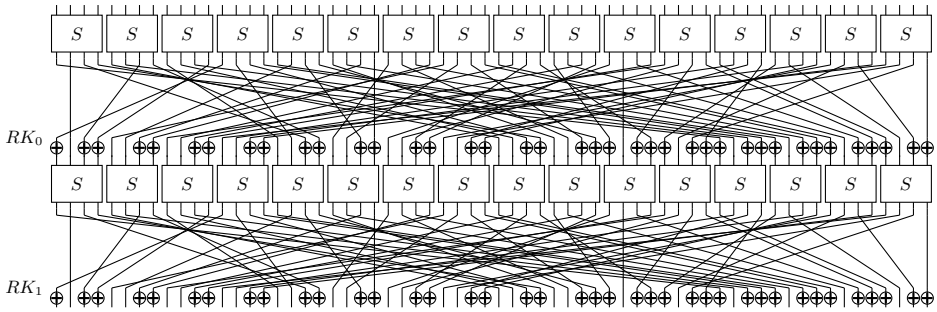

Fig. 5. Two consecutive rounds of GIFT-64.

F Detailed results for PRESENT, GIFT, and ASCON

This appendix collects the full tables of probabilistic and probability-one integral distinguishers for PRESENT, GIFT, and ASCON referred to from Section 4, together with the accompanying cipher descriptions and discussion.

F.1 Application to PRESENT and GIFT

PRESENT is a lightweight block cipher proposed by Bogdanov *et al.* in 2007 [10]. It operates on a 64-bit state and is available in two versions, namely PRESENT-80 and PRESENT-128, which differ only in the size of the master key (80 and 128 bits, respectively) and the key schedule. Both variants use the same round structure, consisting of round-key addition, a layer of 4-bit S-boxes, and a bit permutation. Our analysis applies to both variants, which we refer to simply as PRESENT. The round function specification, the S-box algebraic description and numbering conventions are given in Appendix E.1.

Similarly, GIFT was designed by Banik *et al.* as an improved successor to PRESENT [2]. The cipher has two variants: GIFT-64, operating on a 64-bit state for 28 rounds, and GIFT-128, operating on a 128-bit state for 40 rounds. Both variants use a 128-bit master key and follow the same SPN structure as PRESENT (with a different S-box and a different linear layer). Unlike PRESENT though, a sparse round-constant is added at every round, and the round-key is XORed into only half of the state bits. A more complete specification is given in Appendix E.2.

Table 3 summarizes the probabilistic and probability-one integral distinguishers obtained from the algorithm described in the previous section for both variants of GIFT, while distinguishers for 6 to 9 rounds of PRESENT are summarized in Table 4.

Notation used in Tables 3 and 4. We briefly discuss the notation used in both tables. Inactive and fully active S-boxes in an input structure are represented by their respective hexadecimal activity mask 0 and f. If present, the only

partially-active input S-box is marked by the symbol $\star$ and column $\mathcal{B}$ provides a basis of the input space held constant, so that its active local subspace is $\{x \in \mathbb{F}_2^4 : b \cdot x = 0 \text{ for every } b \in \mathcal{B}\} = \langle\mathcal{B}\rangle^{\perp}$ of dimension $4 - \dim\langle\mathcal{B}\rangle$. If N_{full} denotes the number of fully active input S-boxes, the dimension of the input space X is thus $d = \dim(X) = 4N_{\text{full}} + (4 - \dim\langle\mathcal{B}\rangle)$, where the last contribution is omitted when no partially active S-box is present. The data required to evaluate one integral sum over an affine space $a + X$ is therefore $D = 2^d$. The pair (j, γ) indicates that we consider the γ-component of the j-th S-box after the last S-box layer (*i.e.* before the final bit permutation) from which the output mask β can easily be deduced.

For the probabilistic configurations, $\widehat{q}$ estimates the key-independent value of $q_{\mathcal{D}}(X, \beta)$. Under the heuristic that the integral sum is unbiased whenever a valid trail remains, $\widehat{p} = (1 + \widehat{q})/2$ is the corresponding estimate of the zero-sum probability. The column $\widehat{p}_{\text{emp}}$ reports the empirical zero-sum probability. For the probability-one configurations, the MILP solver proves that no valid division trail exists. These values, marked with † in the table, do not rely on the conditional-unbiasedness heuristic or on a probability estimate. A dash in the experimental-probability column indicates that no empirical probability is reported because the particular experiment is too costly.

Table 3. Integral distinguishers for `GIFT-64` and `GIFT-128`.

Cipher	R	Input structure	D	$\mathcal{B}$	(j, γ)	$\widehat{q}$	$\widehat{p}$	$\widehat{p}_{\text{emp}}$
GIFT-64	6	0x000000000000000f0	2^4	–	(12, 0x3)	0.1877	**0.5938**	0.7270
		0x000000f⋆00000000	2^6	{0001, 1000}	(6, 0x1)	0.1764	**0.5882**	0.6527
		0x00f⋆000000000000	2^7	{0100}	(15, 0x1)	0.3636	**0.6818**	0.7146
		0x00ff000000000000	2^8	–	(15, 0x1)	0.8415	**0.9208**	0.9239
		0x00000000000000ff⋆	2^{11}	{1000}	(7, 0x1)	$1^{\dagger}$	$\mathbf{1}^{\dagger}$	1
	7	0xfff⋆000000000000	2^{14}	{0001, 1000}	(0, 0x1)	0.2630	**0.6315**	0.7268
		0x00000000⋆fff0000	2^{15}	{0100}	(13, 0x1)	0.7335	**0.8667**	0.9406
		0x0000000000000ffff	2^{16}	–	(8, 0x1)	0.9995	**0.9997**	0.9999
		0x0000000000f⋆ffff	2^{23}	{0100}	(4, 0x1)	$1^{\dagger}$	$\mathbf{1}^{\dagger}$	1
	8	0xffffffff00000000	2^{32}	–	(14, 0x1)	0.0061	**0.5031**	0.5418
		0x0000ffffffffffff	2^{48}	–	(2, 0x1)	$1^{\dagger}$	$\mathbf{1}^{\dagger}$	–
	9	0xfff⋆fffffffffffff	2^{61}	{0001, 0010, 1000}	(3, 0x1)	$1^{\dagger}$	$\mathbf{1}^{\dagger}$	–
GIFT-128	6	0x0000000000000000000000000000⋆00	2^3	{1100}	(31, 0x3)	0.4364	**0.7182**	0.7610
		0x00000f0000000000000000000000000	2^4	–	(3, 0x1)	0.9681	**0.9840**	0.9720
		0x0000000000000000000000000000f⋆	2^7	{0100}	(18, 0x3)	$1^{\dagger}$	$\mathbf{1}^{\dagger}$	1
	7	0x0000000000000000000000000000f⋆0f	2^{10}	{0001, 1010}	(12, 0x1)	0.4323	**0.7161**	0.7924
		0x000000000000⋆ff00000000000000000	2^{11}	{1100}	(24, 0x1)	0.6436	**0.8218**	0.9372
		0x00000000000f0ff0000000000000000	2^{12}	–	(8, 0x1)	0.9722	**0.9861**	0.9889
		0x0000000000000000000000000000fff⋆	2^{13}	{0001, 0010, 1000}	(4, 0x1)	$1^{\dagger}$	$\mathbf{1}^{\dagger}$	1
	8	0xffff0000000000000000000000fff0000	2^{28}	–	(6, 0x1)	0.7797	**0.8898**	0.9574
		0x0000000000000000000000000ffffffff	2^{32}	–	(9, 0x1)	$1^{\dagger}$	$\mathbf{1}^{\dagger}$	–
	9	0x00000000000000000fffffffffffff0fff	2^{60}	–	(9, 0x1)	0.6634	**0.8317**	–
		0x00000000000000000fffffffffffff⋆fff	2^{61}	{0001, 0010, 1000}	(8, 0x1)	$1^{\dagger}$	$\mathbf{1}^{\dagger}$	–

Cube-dimension/probability trade-off for integral distinguishers. These results illustrate, for a fixed number of rounds, the tradeoff between the size of the input space and the probability of obtaining a zero sum.

For example, for 6-round `GIFT-64`, our tool finds a probability-one distinguisher with $D = 2^{11}$, whereas an input space of size 2^8 gives an estimated zero-sum probability of 0.9208 and an empirical probability of 0.9239. For 7 rounds, the configuration with $D = 2^{16}$ already has an empirical probability of 0.9999, while the probability-one configuration found by the tool uses $D = 2^{23}$. For `PRESENT`, the experimental probability for the 7-round distinguisher with $D = 2^{13}$ is 0.9882 for both key lengths, using eight times fewer plaintexts per structure than what needed for the deterministic distinguishers.

Thus, a probability close to one can be obtained with substantially fewer plaintexts per integral sum than the corresponding proven configuration found in our search. However, these comparisons concern the data required for a *single* integral sum and do not directly determine the total data needed for a distinguishing experiment since a probabilistic distinguisher may require several trials to detect a given bias. Moreover, the probability-one configurations reported here are results of the structured search, rather than proven minima over all possible input and output spaces.

Comparisons between estimates and empirical probabilities. Among the probabilistic configurations evaluated empirically, the observed zero-sum probabilities are, in several cases, larger than those predicted by $\widehat{p} = (1 + \widehat{q})/2$. This does not contradict the no-trail analysis but instead reflects limitations of the conditional-unbiasedness heuristic, as well as approximation error in $\widehat{q}$.

Dependence on the master key. We also performed additional empirical evaluations for many keys to investigate the dependence on the master key. For `GIFT`, and for `PRESENT` we refer to Table 8 both in Appendix H. The entries $\widehat{p}_{\text{emp}}$ in Tables H and 4 actually report the mean probability over these keys. As seen in these tables, the largest variation occurs for the 7-round `GIFT-128` distinguisher with data complexity 2^{10}, whose probability ranges from 0.7624 to 0.8153 depending on the key. For every other evaluated configuration, the width of the observed interval is at most 0.0352.

F.2 Application to `ASCON`

The `ASCON` permutation. `ASCON` is a family of lightweight authenticated-encryption and hashing algorithms designed by Dobraunig, Eichlseder, Mendel, and Schläffer [15]. It was selected as the primary choice for the lightweight-use case of the CAESAR competition and later became the NIST lightweight cryptography standard. Its permutation operates on a 320-bit state, arranged as five 64-bit words $S_0, \ldots, S_4$, and iterates a round function $p = p_L \circ p_S \circ p_C$ consisting of a round-constant addition p_C, a substitution layer p_S, and a linear diffusion layer p_L.

Table 4. Integral distinguishers for PRESENT.

R	Input structure	D	$\mathcal{B}$	(j,γ)	$\widehat{q}$	$\widehat{p}$	$\widehat{p}_{\text{emp}}$	
							PRESENT-80	PRESENT-128
6	0x000000000000000⋆	2^3	{0001}	(0, 0x1)	0.0362	**0.5181**	0.5411	0.5410
	0x000000000000000f	2^4	–	(0, 0x1)	0.8317	**0.9158**	0.8945	0.8947
	0x000000000000⋆0f	2^5	{1001, 0010, 0100}	(0, 0x1)	0.7068	**0.8534**	0.8974	0.8977
	0x00000000000000ff0	2^8	–	(0, 0x1)	$1^\dagger$	$\mathbf{1}^\dagger$	1	1
7	0x0000f⋆f000000000	2^{11}	{0001}	(0, 0x1)	0.0061	**0.5031**	0.5146	0.5143
	0x000000000000fff0	2^{12}	–	(0, 0x1)	0.6260	**0.8130**	0.8570	0.8568
	0x000000000000fff⋆	2^{13}	{1001, 0010, 0100}	(0, 0x1)	0.9560	**0.9780**	0.9882	0.9882
	0x000000000000ffff	2^{16}	–	(0, 0x1)	$1^\dagger$	$\mathbf{1}^\dagger$	1	1
8	0x0000ffff0000ffff	2^{32}	–	(0, 0x1)	0.0077	**0.5038**	0.5072	0.5083
	0xffffffffffff0000	2^{48}	–	(0, 0x1)	$1^\dagger$	$\mathbf{1}^\dagger$	–	–
9	0xffffffffffff00ff	2^{56}	–	(0, 0x1)	0.0039	**0.5020**	–	–
	0xfffffffffffffff0	2^{60}	–	(0, 0x1)	$1^\dagger$	$\mathbf{1}^\dagger$	–	–

Search setting. We followed the setting used by Rohit *et al.* for their division-property distinguishers of ASCON [28]. There, the initialization permutation is applied to a 320-bit state loaded as $IV\|K_0\|K_1\|N_0\|N_1$ and split into the words $S_0, \ldots, S_4$. The key words K_0, K_1 and the constant IV are secret or fixed, while N_0, N_1 form the public nonce. To obtain a low-data distinguisher on a fixed number of rounds of this permutation, we restrict all active input bits to the last word S_4 (*i.e.*, to N_1), leaving $S_0, \ldots, S_3$ constant; this keeps the input structure entirely inside the part of the state that an adversary can vary without violating the nonce-respecting, misuse-free setting.

Difficulty beyond 5 rounds. ASCON's 5-bit S-box and rotation-based linear layer make the propagation tables and the resulting MILP models considerably denser than for PRESENT or GIFT. From 6 rounds on, even certifying the existence, or the absence, of a single division trail for a given input-output pair already becomes computationally difficult. Consequently, we were unable to run the tree-based probability search of Section 3.4 on ASCON beyond 5 rounds, and we only report distinguishers for 4 and 5 rounds below.

Results. Table 5 summarizes the probabilistic integral distinguishers we obtained for 4- and 5-round ASCON. Since only S_4 can be active, the input structure column only shows the mask of active bit positions within this word; $S_0, \ldots, S_3$ are always constant.

For comparison, the best previously known deterministic (probability-one) division-property distinguishers of ASCON require cube dimensions 2^5, 2^{16}, 2^{31}, and 2^{60} for 4, 5, 6, and 7 rounds, respectively [28]. Our probabilistic 4-round distinguisher already reduces the data complexity from 2^5 to a single active bit, $D = 2^1$, and our 5-round distinguishers reduce it from 2^{16} to 2^{12} or 2^{13}, at the cost of a zero-sum probability bounded away from 1 rather than equal to 1.

Table 5. Probabilistic integral distinguishers for ASCON. The input structure column gives the mask of active bit positions of S_4; $S_0, \ldots, S_3$ are constant. For every entry, the target S-box is S-box 0, and A gives the selected output linear combination γ as a 5-bit mask over $y_0, \ldots, y_4$.

R	Input structure (S_4)	D	A	$\widehat{q}$	$\widehat{p}$	$\widehat{p}_{\text{emp}}$
4	0x0040000000000000	2^1	0x13	0.8125	**0.9063**	0.9373
5	0x860c80041a028000	2^{12}	0x4	0.1203	**0.5602**	0.5959
	0x860c80041a028008	2^{13}	0x4	0.3947	**0.6973**	0.7467

This is the same trade-off between input size and zero-sum probability already observed for PRESENT and GIFT in Appendix F.1.

G Classification of the activity scenarios for PRESENT

The 256 activity scenarios form 32 classes up to a renaming of the polynomial variables. In particular, 106 scenarios give $\mathcal{I}_s \equiv 0$. Averaging the exact probabilities below, weighted by the number of scenarios in each class, gives

$$p_{\text{aff}} = \frac{60033501}{67108864} \approx 0.8945688754.$$

H Experimental analysis of key dependence for the GIFT and PRESENT distinguishers

We provide here a more detailed analysis of the experiments on GIFT and PRESENT presented in Section F.1.

GIFT. Table 7 gives the minimum, maximum, and mean zero-sum probabilities observed over the sampled master keys. All configurations with $D < 2^{17}$ were evaluated using 10^3 keys and 10^4 independently sampled offsets per key; the two 8-round configurations used the smaller sample sizes indicated below the table.

PRESENT. For each version of PRESENT, we sampled 10^3 master keys and estimated $p_k(X, \beta)$ from 10^4 independently sampled offsets for each key. Table 8 gives the minimum, maximum, and mean of the resulting probabilities.

The two key schedules produce very similar results: the mean probabilities differ by at most 0.0011, while the observed ranges are also nearly identical. These experiments therefore suggest that, for the sampled keys, the choice between the PRESENT-80 and PRESENT-128 key schedules has no appreciable effect on the observed biases.

Classes 1–8					Classes 9–16				
Class Id	**Sc.**	**Mon.**	**Var.**	$\Pr[\mathcal{I}_s = 0]$	**Class Id**	**Sc.**	**Mon.**	**Var.**	$\Pr[\mathcal{I}_s = 0]$
1	106	0	0	1.0000	9	4	22	22	0.7935
2	24	2	8	0.8828	10	4	30	28	0.7638
3	28	4	10	0.8828	11	4	34	26	0.7825
4	22	4	12	0.8301	12	4	38	26	0.7825
5	2	8	16	0.8052	13	4	40	28	0.7825
6	3	8	16	0.7583	14	4	42	20	0.8184
7	12	10	16	0.8572	15	4	44	28	0.7825
8	2	20	20	0.7931	16	4	50	24	0.8184

Classes 17–24					Classes 25–32				
Class Id	**Sc.**	**Mon.**	**Var.**	$\Pr[\mathcal{I}_s = 0]$	**Class Id**	**Sc.**	**Mon.**	**Var.**	$\Pr[\mathcal{I}_s = 0]$
17	2	72	36	0.7444	25	1	102	40	0.6720
18	2	72	38	0.7164	26	1	118	38	0.6977
19	2	82	36	0.7357	27	1	122	38	0.6977
20	2	84	38	0.7357	28	1	126	38	0.6977
21	2	86	36	0.7357	29	1	130	38	0.6977
22	2	88	38	0.7357	30	2	134	38	0.6851
23	2	92	38	0.7180	31	2	142	40	0.6851
24	1	94	36	0.7103	32	1	276	40	0.6306

Table 6. Classification of the 256 activity scenarios for `PRESENT` after imposing the corresponding zero constraints on the affine coefficients. Two scenarios belong to the same class when their polynomials $\mathcal{I}_s$ coincide up to a bijective renaming of variables. The columns Sc., Mon., Var. respectively indicate the number of scenarios, monomials and variables for a given class.

Table 7. Variation of the empirical zero-sum probability with the master key for the `GIFT` distinguishers using 10^3 randomly sampled keys and 10^4 independently sampled offsets per key (except for marked entries).

Cipher	R	D	$p_{\text{emp}}^{\min}$	$p_{\text{emp}}^{\max}$	$\overline{p}_{\text{emp}}$
`GIFT-64`	6	2^4	0.7109	0.7424	0.7270
	6	2^6	0.6392	0.6690	0.6527
	6	2^7	0.6970	0.7322	0.7146
	6	2^8	0.9153	0.9327	0.9239
	7	2^{14}	0.7120	0.7430	0.7268
	7	2^{15}	0.9321	0.9484	0.9406
	7	2^{16}	0.9994	1.0000	0.9999
	8	$2^{32\dagger}$	0.5230	0.5615	0.5418
`GIFT-128`	6	2^3	0.7474	0.7750	0.7610
	6	2^4	0.9665	0.9776	0.9720
	7	2^{10}	0.7624	0.8153	0.7924
	7	2^{11}	0.9289	0.9454	0.9372
	7	2^{12}	0.9854	0.9922	0.9889
	8	$2^{28\ddagger}$	0.9470	0.9645	0.9574

† Evaluated using 64 keys and 4×10^3 offsets per key. ‡ Evaluated using 64 keys and 2×10^3 offsets per key.

Table 8. Variation of the empirical zero-sum probability with the master key for `PRESENT-80` and `PRESENT-128` using 10^3 randomly sampled keys and 10^4 independently sampled offsets per key (except for the marked entry).

		`PRESENT-80`			`PRESENT-128`		
R	D	$p_{\text{emp}}^{\min}$	$p_{\text{emp}}^{\max}$	$\overline{p}_{\text{emp}}$	$p_{\text{emp}}^{\min}$	$p_{\text{emp}}^{\max}$	$\overline{p}_{\text{emp}}$
6	2^3	0.5258	0.5581	0.5411	0.5267	0.5578	0.5410
	2^4	0.8842	0.9046	0.8945	0.8841	0.9042	0.8947
	2^5	0.8877	0.9091	0.8974	0.8873	0.9076	0.8977
7	2^{11}	0.4963	0.5304	0.5146	0.4950	0.5305	0.5143
	2^{12}	0.8451	0.8669	0.8570	0.8453	0.8674	0.8568
	2^{13}	0.9850	0.9915	0.9882	0.9846	0.9913	0.9882
8	$2^{32\dagger}$	0.4883	0.5270	0.5072	0.4922	0.5255	0.5083

† Evaluated using 64 keys and 4×10^3 offsets per key.

I Additional Comparison of Key Recovery from Probabilistic Integrals

I.1 Application to GIFT-64

We compare two six-round configurations from Table 3: the deterministic configuration with $d_{\mathrm{det}} = 11$ and target $(j, \gamma) = (7, \texttt{0x1})$, and the probabilistic configuration with $d_{\mathrm{prob}} = 8$ and target $(15, \texttt{0x1})$. Their input structures are, respectively, $\texttt{0x000000000000ff}\star$ with $\mathcal{B} = \{\texttt{1000}\}$, and $\texttt{0x00ff000000000000}$. For the probabilistic configuration, the empirical mean in Table 3 gives

$$p \approx 0.9239, \qquad p - \tfrac{1}{2} \approx 0.4239.$$

We append two rounds in both cases and partially decrypt to the corresponding single-bit target. Each partial decryption uses $k = 10$ raw round-key bits, so both comparisons enumerate 2^{10} guesses for eight-round GIFT-64.

Table 9. Cost of reducing the 10-bit GIFT-64 key-dependency space by s bits in the naive exhaustive-decryption model, using a six-round distinguisher and two appended rounds. The deterministic columns use the deterministic cost formulas above. The probabilistic columns use cumulative Neyman–Pearson filtering with $p = 0.9239$ and target success probability $q = 0.75$.

Reduction s	$\log_2 D_{\mathrm{det}}$	$\log_2 T_{\mathrm{det}}$	m_s	$\log_2 \overline{D}_{\mathrm{prob}}$	$\log_2 \overline{T}_{\mathrm{prob}}$
1	11.00	21.00	1	8.42	18.42
2	12.00	21.58	5	10.74	20.00
4	13.00	21.91	13	12.12	20.58
8	14.00	21.99	31	13.37	20.74
10	14.32	22.00	40	13.74	20.74

For the deterministic property, reducing the key space by s bits gives

$$D_{\mathrm{det}}(s) = s2^{11}, \qquad T_{\mathrm{det}}(s) \approx 2^{22}(1 - 2^{-s}).$$

For the probabilistic property, if m_s denotes the largest cumulative sample size in the schedule for the same target reduction, then

$$D_{\mathrm{prob,stage}}(s) = m_s 2^8, \qquad T_{\mathrm{prob,stage}}(s) \lesssim 2^{19} \sum_{i=1}^{s} \Delta m_i 2^{-i}.$$

At $s = 10$, the exact binomial schedule uses $m_{10} = 40$ structures. The normalized probabilistic estimates are approximately $2^{13.74}$ chosen texts and $2^{20.74}$ partial-decryption evaluations, compared with $2^{14.32}$ and $2^{22.00}$ for the deterministic configuration. Within the naive model, the probabilistic configuration therefore reduces data by a factor 1.5 and filtering time by approximately 2.38.

I.2 Application to GIFT-128

For a matched six-round comparison, we use the deterministic configuration with $d_{\mathrm{det}} = 7$ and target $(18, \texttt{0x3})$, and the probabilistic configuration with $d_{\mathrm{prob}} = 3$ and target $(31, \texttt{0x3})$ from Table 3. The deterministic input has fully active nibble 1 and partially active nibble 0: its suffix is $\texttt{f}\star$, with $\mathcal{B} = \{\texttt{0100}\}$ for the starred nibble. The probabilistic configuration has only nibble 2 partially active, with $\mathcal{B} = \{\texttt{1100}\}$. Its empirical probability is

$$p \approx 0.7610, \qquad p - \tfrac{1}{2} \approx 0.2610.$$

Both targets XOR two coordinate bits. We append two rounds and naively guess all $k = 20$ involved raw round-key bits, obtaining a comparison for eight-round GIFT-128.

For the probabilistic filter we again use $q = 0.75$. For the deterministic property, reducing the key space by s bits gives

$$D_{\mathrm{det}}(s) = s2^7, \qquad T_{\mathrm{det}}(s) \approx 2^{28}(1 - 2^{-s}).$$

For the probabilistic property, if m_s denotes the largest cumulative sample size in the schedule for the same target reduction, then

$$D_{\mathrm{prob,run}}(s) = m_s 2^3, \qquad T_{\mathrm{prob,run}}(s) \lesssim 2^{24} \sum_{i=1}^{s} \Delta m_i 2^{-i}.$$

Table 10 gives the resulting values.

Table 10. Cost of reducing the 20-bit GIFT-128 key-dependency space by s bits in the naive exhaustive-decryption model, using a six-round distinguisher and two appended rounds. The deterministic columns use the deterministic cost formulas above. The probabilistic columns use cumulative Neyman–Pearson filtering with $p = 0.7610$ and target success probability $q = 0.75$.

Reduction s	$\log_2 D_{\mathrm{det}}$	$\log_2 T_{\mathrm{det}}$	$m_{s,s}$	$\log_2 \overline{D}_{\mathrm{prob}}$	$\log_2 \overline{T}_{\mathrm{prob}}$
1	7.00	27.00	1	3.42	23.42
2	8.00	27.58	14	7.22	26.50
4	9.00	27.91	44	8.87	27.38
10	10.32	28.00	142	10.56	27.57
20	11.32	28.00	315	11.71	27.57

At $s = 20$, the exact schedule requires $m_{20} = 315$ structures. The normalized probabilistic time is approximately $2^{27.57}$, compared with $2^{28.00}$ for the deterministic configuration. Its normalized data is approximately $2^{11.71}$, compared with $2^{11.32}$. Thus the probabilistic configuration reduces filtering time by a factor of approximately 1.35, while using 1.3125 times as much data. The smaller integral structure therefore does not imply a reduction in total data for this matched 6-round comparison.

I.3 Comparison with Differential-Linear Key Recovery on PRESENT

We compare two key recoveries for 9-round PRESENT: the seven-round differential-linear (DL) distinguisher of Hadipour et al. [17], followed by two recovery rounds, and our 6-round probabilistic integral, followed by three recovery rounds. The DL distinguisher uses input difference Δ = 0x0000009000000000 and output mask β = 0x0800000008000800 with reported correlation $|C| \approx 2^{-5.32}$. Each pair $\{x, x \oplus \Delta\}$ can be seen as a one-dimensional probabilistic integral: its integral sum. Thus $d_{\mathrm{DL}} = 1$ and the bias magnitude is $|C|/2 \approx 0.012517$. Our probabilistic integral uses $d_{\mathrm{prob}} = 4$, output target $(j, \gamma) = (0, \mathtt{0x1})$, and empirical zero-sum probability $p_{\mathrm{prob}} = 0.8945$.

Since distinguishers produce different output masks, and DL distinguishers often yield high-weight masks, they are not so convernient for key recovery.We also see this as an advantage of the probabilistic integral: it is more likely to produce a good output mask for conducting an attack. Therefore, it makes sense to exploit the fact that probabilistic integrals provide better output masks, so that the admissible-key-difference method of [4] can be applied. That one gives $k_{\mathrm{prob}} = 42$. For DL, the target bits $11, 27, 59$ have disjoint two-round dependency of 20 bits each. Admissible differences therefore yield no reduction, leaving $k_{\mathrm{DL}} = 60$.

Table 11. Comparison of partial-key filtering for nine-round PRESENT, reducing each effective key space by s bits. DL uses a seven-round distinguisher and two recovery rounds ($d = 1$, $|C| = 2^{-5.32}$, $\ell_{DL} = 60$); probabilistic integral uses a six-round distinguisher and three recovery rounds ($d = 4$, $p = 0.8945$, $\ell_{prob} = 42$). Each schedule is recomputed separately for each s at $q = 0.75$.

	DL: $7+2$ rounds, $\ell = 60$			PI: $6+3$ rounds, $\ell = 42$		
Reduction s	m_s	$\log_2 \overline{D}$	$\log_2 \overline{T}$	m_s	$\log_2 \overline{D}$	$\log_2 \overline{T}$
1	2 571	12.74	72.74	1	4.42	46.42
2	10 233	14.74	74.22	5	6.74	48.00
4	25 408	16.05	74.80	15	8.32	48.81
10	73 424	17.58	74.95	51	10.09	49.00
20	157 512	18.68	74.96	112	11.22	49.01
30	243 368	19.31	74.96	174	11.86	49.01
40	329 974	19.75	74.96	237	12.30	49.01
42	347 352	19.82	74.96	249	12.38	49.01
50	416 996	20.08	74.96	—	—	—
60	504 282	20.36	74.96	—	—	—