

A Quasidifferential Analysis of the Wrong-Key Randomization Hypothesis

Tim Beyne¹ , Gregor Leander² , Mariia Mutkovina²  and
Ricardo Rodriguez Reveco² 

¹ COSIC, KU Leuven, Leuven, Belgium

² Faculty of Computer Science, Ruhr University Bochum, Bochum, Germany

Abstract

The Wrong-Key Randomization (WKR) hypothesis governs data-complexity estimates in differential cryptanalysis: wrong-key guesses are assumed to behave as a random permutation would. Exact computation of fixed-key differential probabilities was, until recently, infeasible.

We use quasidifferential trails [BR22] to compute the exact wrong-key distribution for the key-recovery map $G_{k,k'} = F_{k'}^{-1} \circ F_k$ in PRESENT-like SPNs. A mask-first reformulation exposes a Walsh–Hadamard structure; restricting the transform to the low-dimensional support, together with SMT-guided trail enumeration, reduces the cost: for a 16-bit toy cipher, from 2^{80} to 2^{13} ; for PRESENT, from 2^{192} to 2^{30} ; and for GIFT-64, from 2^{192} to 2^{32} .

For the toy cipher, PRESENT and GIFT-64, the computed distribution is a structured mixture: a large zero-probability class coexists with bottleneck classes orders of magnitude above the random-permutation mean, and nothing lies between them. Such a distribution is not unimodal, so no Poisson or binomial law fits it for any parameter and the hypothesis is formally false for all three targets. For PRESENT, however, we show that this deviation does not affect the security of Wang’s 14-round differential attack. We cast the computed distribution as a structured composite hypothesis—the differential counterpart of the random-permutation/composite-hypothesis model used for wrong keys in linear cryptanalysis [BR25]—and show that the shape of the wrong-key distribution, not merely its mean, governs how many wrong keys survive the key-recovery filter. For PRESENT with Wang’s distinguisher, the structural signal is carried only by the right pairs, whose weight is too small for the deviation to surface; the hypothesis remains a safe heuristic in this case despite being formally false. Our SMT-based enumeration tool is publicly available.

Keywords: Differential cryptanalysis, Wrong-Key Randomization, Quasidifferential Trails

1 Introduction

Differential cryptanalysis is among the oldest and best-studied attack vectors against symmetric cryptography [BS91]. Since Biham and Shamir introduced it, cryptanalysts have examined all aspects of differential cryptanalysis: finding optimal distinguishers, proving resistance bounds, and automating the search for characteristics. There are many variants, ranging from impossible differential [Knu98; BBS99] and multiple-differential [BG11] to boomerang [Wag99] and differential-linear attacks [LH94]. Despite this maturity, still fundamental questions remain.

Emails: tim.beyne@esat.kuleuven.be (Tim Beyne), gregor.leander@rub.de (Gregor Leander), mariia.mutkovina@rub.de (Mariia Mutkovina), ricardo.rodriguezreveco@rub.de (Ricardo Rodriguez Reveco)

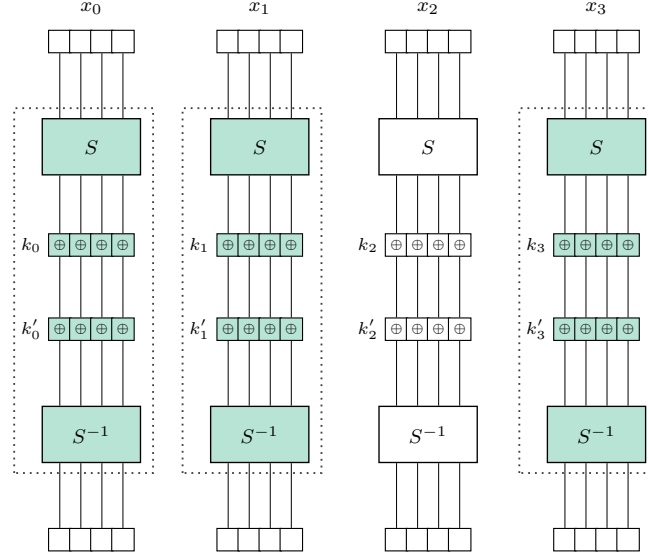


Figure 1: S-box collapse in a key-recovery round. Nibbles where the correct and guessed subkeys are equal (green): S followed by S^{-1} reduces to the identity; the S-box layers cancel out. Nibbles where the subkeys disagree (white): both S-box layers remain in the path.

One is the key-recovery phase. In a standard differential attack, an attacker uses a distinguisher over several rounds and guesses subkeys for r additional rounds to check for consistency between the distinguisher and the guessed keys. The standard statistical model relies on the *Wrong-Key Randomization* (WKR) hypothesis: for any wrong-key guess, the cipher behaves like a random permutation. In practice the model is applied in a simplified form, with the probability that the differential holds fixed at 2^{-n} for every wrong key.

The problem: the WKR hypothesis is unverified. While the WKR hypothesis simplifies analysis, it is rarely verified. Theoretical intuition suggests it should fail in several contexts. For example, consider a key-recovery step with a single round. If a wrong-key guess differs from the correct key by only a few bits, the decryption is almost identical to the correct one, preserving the differential property with high probability (Figure 1).

Until recently, exact verification was infeasible. Computing the exact differential probability for a *fixed* key (rather than an average over all keys) was an open problem. Standard tools only computed expected probabilities (p_{EDP}) over random keys, hiding the specific behavior of individual key candidates. In the linear-cryptanalysis setting, an analogous limitation has been addressed through composite-hypothesis techniques and the random-permutation model for wrong keys [BR25, §7.3.3].

The solution: quasidifferential trails. Beyne and Rijmen [BR22] made fixed-key probabilities *expressible* exactly, through quasidifferential trails. This is what lets us *test* the WKR hypothesis instead of assuming it: it is a claim about the wrong-key hit probabilities, and quasidifferential trails are the first general method that expresses them exactly. Key-averaged tools such as the DDT or the expected differential probability average them away, and direct experiments cannot measure probabilities as small as 2^{-n} —distinguishing a fixed-key probability $p_G = 2^{-n}$ from $p_G = 0$ would need on the order of 2^n samples per key. Expressing these probabilities and *computing* them are different problems, however:

evaluated naively, the expression is a doubly exponential sum, infeasible even for our toy cipher. Making it computable is the technical core of this paper (Section 4).

1.1 Related work

Fixed-Key Differential Behavior. The fact that differential probabilities depend on the specific key has long been known. Knudsen [Knu93] already observed significant deviations from stochastic equivalence for the iterative characteristics used in the differential analysis of DES. Experimental studies, such as those by Heys [Hey20], further showed that such deviations are the norm rather than the exception. Daemen and Rijmen [DR07a] analyzed, for two rounds of the AES, the case of “plateau characteristics,” which take only two possible values per key. This line of work was recently extended by Canteaut and Fruchon [CF25] to cover more general structures. Beyne and Rijmen [BR22] introduced quasidifferential trails, providing the first rigorous method to compute exact fixed-key probabilities. This technique was later extended by Boura et al. [BDG25] to cover the expected differential probability computation, while Peyrin and Tan [PT22] investigated key dependencies to demonstrate that valid trails can often be incompatible with the cipher’s key schedule. Exact computation matters in practice: for the SPEEDY family, Boura et al. [BDBN23] mounted a full-round differential attack on SPEEDY-7-192, computing the distinguisher probability under the standard product-of-DDT heuristic. Beyne and Neyt [BN24; BN25] later showed, using quasidifferential trails, that the main characteristic underlying this attack has probability *zero*. Similarly, Dobraunig et al. [DEKM16] showed, in a practical attack on MANTIS5, that some wrong keys are differentially equivalent to the correct key. For PRESENT specifically, Tezcan [Tez16] showed that differential factors in the PRESENT S-box reduce the advantage of Wang’s attack [Wan08] by a further 6 bits, independently confirming that the attack’s key recovery is more constrained than the nominal 32-bit claim suggests.

Statistical key-recovery models. Evaluating the success of a differential attack requires a statistical model. Selçuk [Sel08] formalized the *ranking approach*, using order statistics to estimate the probability that the correct key ranks among the top candidates. Blondeau et al. [BGT11] later introduced the *hypothesis-testing formulation*, which models key-recovery as a binary classification problem and provides accurate data complexity estimates using large deviation theory. Both approaches heavily rely on the WKR hypothesis to model the behavior of incorrect key candidates. While the WKR hypothesis simplifies analysis, its validity has been questioned, particularly in the context of linear cryptanalysis, where correlation biases for wrong keys are known to vary [BT14]; see [BR25, §7.3.3] for a textbook treatment. In differential cryptanalysis, generic key-recovery tools like those by Boura et al. [BDD⁺24] optimize the search algorithmically but generally operate efficiently under the assumption that the underlying statistical model (and the WKR hypothesis) holds.

WKR Violations in Practice. Concrete deviations from the WKR hypothesis appear across cipher families. In DEFAULT [BBB⁺21], whose equivalent-key structure is a design property, Nageler et al. [NDE22] exploited large classes of equivalent keys: many wrong-key guesses produce identical differential behavior rather than the uniform distribution predicted by the WKR hypothesis, and combining information across rounds recovered the full key with negligible complexity. In the ARX setting, *probabilistic neutral bits* (PNB) [AFK⁺08; FT25] capture a structurally analogous phenomenon: wrong-key guesses that differ from the correct key only in neutral bit positions preserve the elevated differential probability, creating wrong-key distributions in which a subset of keys retains a high differential probability instead of all wrong keys behaving alike. A third class of extreme

WKR violation appears in LOWMC [DEM16; LSW⁺22], where a sparse S-box layer forces the key-recovery map onto a near-linear structure, enabling algebraic attacks that exploit predictable wrong-key behavior. The hypothesis also fails outside differential cryptanalysis: Todo et al. [TIM⁺18] showed that the corresponding assumption does not hold for fast correlation attacks on stream ciphers, and exploited the resulting wrong-key behavior to attack Grain-128a, Grain-128 and Grain-v1 in full. These examples underscore that structured wrong-key distributions are not a curiosity but a recurring feature of lightweight designs and that methods to compute, rather than assume, them are a necessary tool.

1.2 Our contributions

In this work, we revisit key-recovery differential attacks without relying on the WKR hypothesis. Our contributions are threefold.

Formalization. We recast the WKR hypothesis as a falsifiable claim about the wrong-key hit probability of the chain $F_{k'}^{-1} \circ E_k$: for $k' \neq k$, the probability that a pair with input difference a triggers the counter is distributed as it would be for a random permutation. This fixes the whole law and not just a mean; the simplified form used in practice replaces it by the single value 2^{-n} . The object we compute with is the key-recovery map $G_{k,k'} := F_{k'}^{-1} \circ F_k$; the distinguisher E'_k enters through the right-pair probability p_R (Section 3.3). Quantitative wrong-key models exist for linear cryptanalysis [BT14; BR25]; in the differential setting the hypothesis is usually stated qualitatively. We state it as a claim that can be tested cipher by cipher.

Tool. We develop a general method to compute the fixed-key probability distribution of a differential through a keyed permutation; it may be of independent interest beyond the key-recovery application. By reversing the summation order in the fixed-key characteristic formula of [BR22], we expose a Walsh–Hadamard structure over the intermediate-mask space and exploit that the relevant masks lie in a low-dimensional subspace. Combined with an SMT-guided trail enumerator for PRESENT-like SPNs, this reduces a computation exponential in both the state size and the number of rounds to one exponential only in the dimension of the mask support. For PRESENT this is a reduction from 2^{192} to 2^{30} , and from 2^{80} to 2^{13} for the toy cipher of Section 2.5.³

Phenomenon and model. For the toy cipher, PRESENT and GIFT-64, we exhibit the fixed-key spectrum of $G_{k,k'}$ as a structured mixture rather than the dispersed but unimodal law a random permutation would give. A large zero-probability class appears alongside classes exceeding the random-permutation mean by several orders of magnitude, with nothing in between. We compare throughout against the random-permutation model, not against the point mass at 2^{-n} that the standard formulas use: a point mass is refuted by any dispersion at all, so refuting it would establish nothing. Beyond the histogram, we identify the algebraic structure behind the partition into equivalence classes: they arise from cosets of a vector space determined by the quasidifferential trails. We use the computed spectrum as the wrong-key model in a composite-hypothesis test, in place of the single alternative that the WKR hypothesis assumes. Within this model we show that the *shape* of the wrong-key distribution governs how many wrong keys survive the key-recovery filter, a quantity the mean alone cannot capture. We also identify a *bottleneck class*: a small set of wrong keys whose differential probability is so high that they pass the filter almost as often as the right key. The two are separate effects, and we compute both. A bottleneck limits how far ranking can narrow the candidate list, while its contribution to the survivor count depends on how many keys it holds — for our toy cipher, four. Crucially,

³All of our source code can be found at <https://doi.org/10.5281/zenodo.22129299>.

the method shows not only *that* the WKR hypothesis fails but *whether the failure matters*: for PRESENT with Wang’s distinguisher [Wan08] we show, by direct computation, that the deviation does not reach the attack, whereas for the toy cipher it does.

1.3 Outline

Section 2 fixes notation, recalls quasidifferential trails, and states the classical counter-based key-recovery setup together with its standard Neyman–Pearson statistical framework. Section 3 takes a closer look at wrong-key randomization: it introduces the key-recovery map as a formal object, reviews the two pre-existing models of wrong-key behavior, states our accurate refinement, and analyzes how the shape of the wrong-key distribution—not just its mean—controls the post-filter survivor count. Section 4 gives the method: the mask-first formulation, the dimension-reduction argument, the SMT trail enumerator, and the certified mask dimensions of the three targets. Section 5 concludes with open directions.

2 Preliminaries

This section fixes notation, recalls quasidifferential trails, and states the classical counter-based key-recovery setup together with its standard Neyman–Pearson statistical framework.

2.1 Differential distinguishers

Differential cryptanalysis, introduced by Biham and Shamir [BS91], is a statistical method used to distinguish a block cipher from a random permutation by analyzing how input differences propagate to output differences.

2.1.1 Keyless case

For a function $F: \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$, the differential behavior is captured by the *Difference Distribution Table* (DDT), a $2^n \times 2^n$ table where each entry

$$\text{DDT}_F(a, b) := |\{x \in \mathbb{F}_2^n : F(x) \oplus F(x \oplus a) = b\}| \quad (1)$$

counts the number of input pairs with difference a that map to output difference b . We call these input pairs *right pairs*. The *differential probability* of the input-output difference tuple (a, b) , called a *differential*, is

$$p(a, b) = \frac{\text{DDT}_F(a, b)}{2^n}. \quad (2)$$

When it is clear from the context to which differential we refer, we will drop the (a, b) from the notation.

2.1.2 Keyed case

A block cipher with an l -bit key is a family of permutations $E_k: \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$, where $k \in \mathbb{F}_2^l$. The permutation being analyzed depends on the choice of key, making the differential probability key-dependent:

$$p(a, b, k) := \frac{\text{DDT}_{E_k}(a, b)}{2^n}. \quad (3)$$

We call this the *fixed-key differential probability*. The *expected differential probability* is

$$p_{\text{EDP}}(a, b) := \mathbb{E}_k[p(a, b, k)], \quad (4)$$

where the expectation is over uniform random keys.

2.1.3 Differential characteristics

Usually, E_k is constructed as the composition of round functions: $E_k = E_{r,k} \circ \dots \circ E_{1,k}$. A *differential characteristic* (or *differential trail*) is a sequence of intermediate differences $\mathbf{a} = (a_1, a_2, \dots, a_{r+1})$ specifying how a difference propagates through each round.

The probability of a characteristic is

$$p_{\text{char}}(\mathbf{a}, k) = \Pr_x [E_{i,k}(x_i) \oplus E_{i,k}(x_i \oplus a_i) = a_{i+1} \text{ for all } i \in \{1, \dots, r\}], \quad (5)$$

where $x_1 = x$ is uniformly random, and $x_i = E_{i-1,k} \circ \dots \circ E_{1,k}(x)$ for $i > 1$.

A single differential (a_1, a_{r+1}) may contain *multiple* characteristics with the same input and output differences but different intermediate values. The total differential probability is the sum over all such characteristics, and in practice, many low-probability trails can contribute significantly to the overall differential probability.

2.1.4 Key-dependent behavior

Under the *Markov Cipher Assumption*, i.e., independent round behavior with independent uniform round keys, and the *hypothesis of stochastic equivalence*, the probability of a characteristic can be heuristically approximated as the product of round probabilities [LMM91]:

$$p_{\text{char}}(\mathbf{a}, k) \approx \prod_{i=1}^r \Pr[E_{i,k}(x) \oplus E_{i,k}(x \oplus a_i) = a_{i+1}]. \quad (6)$$

This “product of round probabilities” heuristic is accurate only for key-averaged probabilities and does not capture key-dependent behavior. For a single round function, the 1-round differential probability is key-independent and captured by the DDT, but across multiple rounds the differential probability becomes key-dependent. Traditional differential cryptanalysis focuses on expected probabilities $p_{\text{EDP}}(a, b)$, but the actual fixed-key probabilities $p(k)$ can vary significantly, which is important for understanding the true security of block ciphers against differential attacks.

2.2 Quasidifferential trails

Computing fixed-key probabilities requires a more sophisticated tool. Beyne and Rijmen [BR22] introduced quasidifferential trails to address this limitation.

2.2.1 Quasidifferential transition matrix

The key computational tool is the *quasidifferential transition matrix* (QDTM) D^F , which extends the difference-distribution table. The entries of this matrix are indexed by pairs (u, a) and (v, b) in $\mathbb{F}_2^n \times \mathbb{F}_2^n$ where u and v are masks and a and b are differences. For $u, x \in \mathbb{F}_2^n$ we write $\langle u, x \rangle = \bigoplus_{i=1}^n u_i x_i$ for the dot product on $\mathbb{F}_2^n$; it is nondegenerate, but not an inner product, since positive definiteness is meaningless in characteristic 2. Each entry has the value:

$$D_{(v,b),(u,a)}^F = \left(2 \Pr_x [\langle v, F(x) \rangle = \langle u, x \rangle \mid F(x+a) = F(x)+b] - 1 \right) \times \Pr_x [F(x+a) = F(x)+b]. \quad (7)$$

That is, $D_{(v,b),(u,a)}^F$ is the correlation of the linear approximation (u, v) restricted to pairs that follow the differential (a, b) , multiplied by the probability of the differential (a, b) . The DDT appears as a submatrix of D^F corresponding to zero masks ($u = v = 0$).

Crucially, the quasidifferential transition matrices compose by multiplication: for $E = E_2 \circ E_1$, it holds that

$$D^E = D^{E_2} D^{E_1}. \quad (8)$$

This property, which does *not* hold for DDTs, enables efficient round-by-round analysis through matrix multiplication and is fundamental to computing fixed-key probabilities. For DDTs, composition only works on average under independence assumptions.

2.2.2 Fixed-key probability formula

The fundamental result for computing fixed-key differential probabilities is the following:

Theorem 1 (Fixed-key probability for a characteristic [BR22, Thm. 4.1]). *Let $F_k : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ be a keyed function such that $F_k = F_{r,k} \circ \dots \circ F_{1,k}$, where $F_{i,k}$ is round i under the key k . The probability of a characteristic $\mathbf{a}$ with differences $a_1, \dots, a_{r+1}$ is equal to:*

$$p_{char}(\mathbf{a}, k) = \sum_{u_2, \dots, u_r} \prod_{i=1}^r D_{(u_{i+1}, a_{i+1}), (u_i, a_i)}^{F_{i,k}}, \quad (9)$$

with $u_1 = u_{r+1} = 0$.

2.2.3 Quasidifferential trails

The sum in Theorem 1 ranges over all sequences of linear masks $(u_2, \dots, u_r)$. Each such sequence, together with the fixed differences $(a_1, \dots, a_{r+1})$, defines a *quasidifferential trail*: a sequence of mask-difference pairs (u_i, a_i) for $i = 1, \dots, r+1$. Each term in the sum corresponds to one such trail and equals the product of the corresponding quasidifferential transition matrix entries. When all masks are zero ($u_i = 0$ for all i), a quasidifferential trail reduces to an ordinary differential characteristic.

2.3 Key recovery

A differential (a, b) with probability $p_R \gg 2^{-n}$ across all but the last r rounds of a cipher can be turned into a key-recovery attack against those last r rounds. Following Biham and Shamir [BS91], the attacker collects a set

$$\mathcal{P} = \{(P_i, P'_i, C_i, C'_i)\}_{i=1}^N, \quad P_i \oplus P'_i = a, \quad (C_i, C'_i) = (E_k(P_i), E_k(P'_i)),$$

of N known plaintext pairs with input difference a . For each candidate guess k' of the subkey material in the last r rounds, the attacker partially decrypts each ciphertext pair over those rounds and maintains a counter

$$T(k') = \#\{(P_i, P'_i, C_i, C'_i) \in \mathcal{P} : \text{partial decryption under } k' \text{ yields difference } b\}. \quad (10)$$

The attacker hopes that the correct key will be among the guesses with the largest counter values. In practice, a diffusion-based pre-filter discards pairs whose output difference cannot propagate to the distinguisher's output difference via probability-1 differentials; the counter model applies to the surviving pairs.

2.3.1 Structured key guessing

An alternative approach, proposed by Boura et al. [BDD⁺24], incrementally builds partial keys by solving S-boxes in an optimized order, pruning inconsistent candidates early.

Starting from $\mathcal{P}$, the procedure maintains a list of triplets

$$\mathcal{L}_i = \{(P_j, P'_j, C_j, C'_j, k^{(i)}) : (P_j, P'_j, C_j, C'_j) \in \mathcal{P}, k^{(i)} \in \{0, 1\}^{m_i}\}, \quad (11)$$

where $k^{(i)}$ is a partial key of length m_i after step i . We initialize the list of triplets as $\mathcal{L}_0 = \{(P_j, P'_j, C_j, C'_j, \emptyset)\}$, and then at each step the attacker guesses new key bits and keeps only extensions consistent with local differential constraints. After the end of the procedure, the output key candidate set is

$$\mathcal{K}_{\geq 1} = \{k' : \exists (P_j, P'_j, C_j, C'_j) \in \mathcal{P}, \Delta G(C_j, C'_j) = b\} \quad (12)$$

where $\Delta G(C_j, C'_j)$ denotes the output difference when partially decrypting (C_j, C'_j) over the last r rounds under k' .

Equivalence to counters. By construction, $k' \in \mathcal{K}_{\geq 1} \iff T(k') \geq 1$. While several algorithmic approaches exist for computing the set of keys with at least one hit, the resulting statistical object is the same in all cases. The difference from classical counters is algorithmic: structured key-guessing avoids storing 2^m values, achieving lower time and memory complexity [BDD⁺24].

2.4 Statistical models for key recovery

Blondeau et al. [BGT11] recast the counter-based key-recovery attack as a binary hypothesis testing problem, parametrized by two quantities: the *success probability* P_S (the probability that the right key is included in the output set) and the *gain* g (measuring the reduction in key space). When the attack outputs an expected 2^{m-g} candidates from a key space of size 2^m , the gain is g bits [Sel08].

We model the key-recovery procedure by viewing the counters $T(k')$ as random variables. Since structured key-guessing outputs exactly the set $\{k' : T(k') \geq 1\}$, the same model applies to both approaches.

For the right key k , the key-recovery map is the identity function, so the probability that a pair suggests the key equals the distinguisher probability p_R , the fraction of pairs that are right pairs. We assume this is a key-independent probability.⁴ When the distinguisher probability has heterogeneous behavior over keys, we are in the setting of *weak key attacks*.

Hence, the counter distribution after testing N pairs follows a binomial distribution:

$$T(k) \sim \text{Binomial}(N, p_R). \quad (13)$$

2.4.1 Hypothesis testing approach

Blondeau et al. assume the wrong-key randomization hypothesis (coined by Harpes, Kramer, and Massey [HKM95] for linear cryptanalysis and first stated explicitly for the differential setting by Blondeau and Gérard [BG11]), and apply it in its simplified form: the differential hits a wrong key with probability 2^{-n} , so the wrong-key counter follows $\text{Binomial}(N, 2^{-n})$. We call this form the point-mass simplification and separate it from the hypothesis proper in Section 3.2; the formulas of this subsection are the ones the literature derives from it. For each key candidate k' , we test

$$\begin{aligned} H_0 : T(k') &\sim \text{Binomial}(N, p_R) \\ H_1 : T(k') &\sim \text{Binomial}(N, 2^{-n}). \end{aligned}$$

Instead of ranking all counters and keeping a fixed-size list of candidates, we make accept/reject decisions through a threshold τ : reject H_0 (discard the key) if $T(k') < \tau$, that is, keep k' only when $T(k') \geq \tau$.

⁴This assumption corresponds to the *right key randomization hypothesis*, a different topic from the one of this work.

In practical differential attacks, the distinguisher probability p_R is typically such that the expected number of right pairs for the correct key is small (often around one). In this regime, keeping only the top-ranked key (ranking with $r = 1$) is equivalent to accepting keys with at least one hit (hypothesis testing with $\tau = 1$). Moreover, this is precisely the procedure implemented by structured key-guessing strategies. For these two reasons, we focus on hypothesis testing throughout this work.

Classical hypothesis tests are parametrized by two types of errors:

- *Non-detection probability* P_{ND} : The probability of rejecting the right key (Type I error).
- *False alarm probability* P_{FA} : The probability of accepting a wrong key (Type II error).

The success probability and gain introduced earlier correspond to $P_S = 1 - P_{\text{ND}}$ and $g = -\log_2(P_{\text{FA}})$, respectively.

2.4.2 Neyman–Pearson test

The Neyman–Pearson lemma [NP33] provides the most powerful test for distinguishing between two simple hypotheses at a given significance level. For our binary test with threshold τ , we reject H_0 if $T(k') < \tau$, where the threshold is chosen to satisfy:

$$\Pr_{H_0}[T(k') < \tau] = \sum_{j=0}^{\tau-1} \binom{N}{j} p_R^j (1 - p_R)^{N-j} = P_{\text{ND}}. \quad (14)$$

The success probability is then $P_S = 1 - P_{\text{ND}}$, and the false alarm probability under H_1 is:

$$\Pr_{\text{Binomial}(N, 2^{-n})}[T(k') \geq \tau] \leq P_{\text{FA}}. \quad (15)$$

2.4.3 Data complexity determination

To determine the required number of samples N for given target probabilities P_{ND} and P_{FA} , we must find a solution (N, τ) to the system:

$$\begin{cases} \Pr_{H_0}[T(k') < \tau] \leq P_{\text{ND}} \\ \Pr_{\text{Binomial}(N, 2^{-n})}[T(k') \geq \tau] \leq P_{\text{FA}} \end{cases} \quad (16)$$

Blondeau et al. [BGT11] provided an algorithm to find the smallest N solving Equation (16) and also sharp asymptotic estimates using large deviation inequalities for the binomial distribution [AG89].

2.5 Target ciphers

To validate our approach, we focus on PRESENT-like SPN block ciphers. These ciphers operate on an n -bit state and employ a round function that consists of parallel 4-bit S-boxes followed by a bit-permutation linear layer.

Assumption on round keys. Throughout this work, we analyze the key-recovery map (defined formally in Section 3.1) under the assumption of independent round keys. That is, we treat the subkeys in the backward and forward rounds as independent variables, ignoring the constraints imposed by the cipher’s key schedule. Round keys are also the natural granularity here: a key-recovery attack guesses the subkeys entering the key-recovery rounds, not the master key. This simplifies the SMT modeling by avoiding the need to

encode complex key-schedule relations. The computed classes then have to be reconciled with a real key schedule, which we do for PRESENT in Section 3.4: there every class turns out to be consistent with the key schedule and the relative frequencies are unaffected. In general the key schedule is better encoded as a constraint in the enumeration itself than applied as a filter to the classes it produces, since filtering can only discard what the enumeration has already paid to compute.

We apply this model to three primitives: the 16-bit toy SPN of Boura et al. [BDD⁺24], PRESENT [BKL⁺07] and GIFT-64 [BPP⁺17].

Toy cipher. The small cipher we obtained data for is the same SPN used by the authors in [BDD⁺24]. It operates on a 16-bit state with 8-bit subkeys, applied on the two rightmost bits of each nibble. The round function is a parallel application of a 4-bit S-box to each of the 4-bit blocks of the state, followed by a bit permutation. The S-box is the same as used by GIFT [BPP⁺17]. Both it and the bit permutation are given in Tables 5 and 6 of Appendix A, which also depicts the round function in Figure 7.

GIFT-64. Our third target adds only two key bits per nibble: the round key is 32 bits over a 64-bit state, reaching bits $4i$ and $4i + 1$, while the other two bits of each nibble receive no key material. The round function is again a parallel 4-bit S-box layer followed by a bit permutation. Its S-box is the one of Table 6 and its permutation is given in Table 10 of Appendix A.

PRESENT. We chose the lightweight block cipher PRESENT as our real-world target. The state size is 64 bits and the round function is once again a composition of a simultaneous application of a 4-bit S-box on each 4-bit word of the state and a linear bit permutation. Both the S-box and the bit permutation are given in Tables 8 and 9 of Appendix A.

The point-mass hypothesis, together with the Neyman–Pearson test of Section 2.4, determines the standard data-complexity formulas. The rest of this paper questions the WKR side of Equation (16): the $\text{Binomial}(N, 2^{-n})$ term of Equation (15), which gives every wrong key hit probability 2^{-n} .

3 A closer look at wrong-key randomization

This section asks whether a more accurate model of wrong-key behavior could prevent the failures collected in Section 1.1. Our answer is a model computed directly from the differential transitions of the cipher at hand: the wrong-key distribution becomes a composite hypothesis with a *structured, cipher-specific* prior instead of a law fixed in advance. We do not claim the model is exact, and it probably is not. What is exact is its computation, which quasidifferential trails (Section 2.2) carry out with no statistical assumption. One simplifying assumption remains, made for feasibility: the prior is computable for the on-diagonal transition, while the full prior over all transitions is not (Section 4.6), so off-diagonal transitions keep the point mass at 2^{-n} . We then test the model against the random-permutation hypothesis on three ciphers and find that, for key-recovery attacks, the difference matters for the toy cipher and not for PRESENT.

Section 3.1 introduces the key-recovery map and Section 3.3 states the model. Section 3.2 reviews the existing hypotheses, Section 3.4 computes all three distributions, and Section 3.5 identifies the regime in which they remain safe heuristics despite being formally wrong.

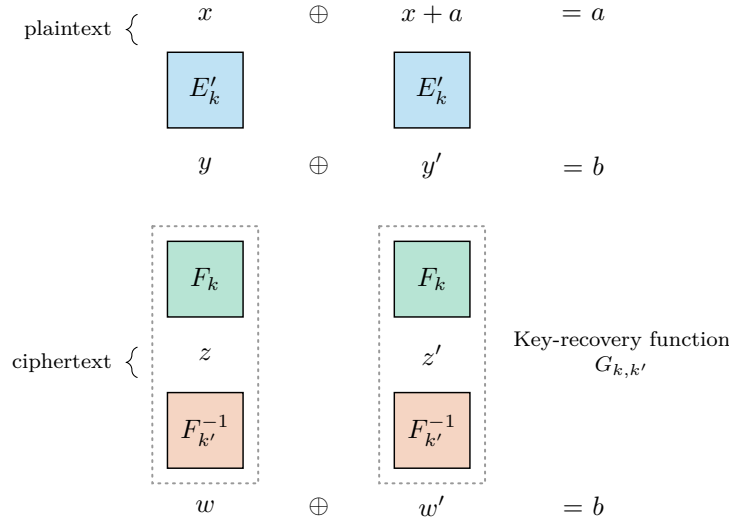


Figure 2: Distinguisher rounds E'_k , key-recovery rounds F_k and $F_{k'}$, and the key-recovery map $G_{k,k'}$.

3.1 The key-recovery map

The counter model of Section 2.3 leaves one object implicit: the permutation that is effectively applied to each ciphertext when a wrong key is tested. We define and study this object. Our analysis uses this map. The statistical treatment appears in Section 3.3; the computational treatment, in Section 4.

Write the cipher as $E_k = F_k \circ E'_k$, where E'_k absorbs the distinguisher rounds and F_k denotes the last r key-recovery rounds. Partially decrypting $C = F_k(u)$ under guess k' gives $F_{k'}^{-1}(C) = (F_{k'}^{-1} \circ F_k)(u)$, the key-recovery map at the distinguisher-boundary state $u = E'_k(P)$.

Definition 1 (Key-recovery map). Let $E_k = F_k \circ E'_k$ be a block cipher with r key-recovery rounds F_k . The *key-recovery map* for the pair (k, k') is

$$G_{k,k'} := F_{k'}^{-1} \circ F_k.$$

Figure 2 illustrates the construction.

Two basic properties are used throughout. First, for the right key $k' = k$, the map $G_{k,k}$ is the identity, so every right pair contributes to $T(k)$. Second, $G_{k,k'}$ depends only on the partial subkeys involved in the key-recovery rounds, reducing the fixed-key analysis to $m \ll |k|$ bits rather than the full master key.

Define the *wrong-key chain* and *wrong-key hit probability*:

$$D_{k,k'} := G_{k,k'} \circ E'_k = F_{k'}^{-1} \circ E_k, \quad (17)$$

$$p_w(k') := \Pr_P[D_{k,k'}(P) \oplus D_{k,k'}(P') = b \mid P \oplus P' = a].$$

Then $T(k')$ counts pairs (P_i, P'_i) for which $D_{k,k'}(P_i) \oplus D_{k,k'}(P'_i) = b$.

3.2 Existing models of wrong-key behavior

Prior work has answered this question with two models of the wrong-key counter distribution. We add a third, tractable model (Hypothesis 3) and the exact but intractable ground truth. Hypotheses 1 and 2 are special cases of Hypothesis 3.

Hypothesis 1 (Point-mass simplification). *For every wrong key $k' \neq k$,*

$$p_w(k') = 2^{-n}.$$

Hypothesis 2 (Wrong-key randomization). *For $k' \neq k$, the wrong-key chain $D_{k,k'}$ is indistinguishable from a uniform random permutation.*

3.2.1 Hypothesis 1: fixed probability

Hypothesis 1 is the form in which the hypothesis is normally applied: every wrong key gives hit probability 2^{-n} . It is a simplification of **Hypothesis 2** and not the hypothesis itself. Blondeau and Gérard draw the same distinction, stating the hypothesis as uniform behavior and then adopting 2^{-n} in place of $1/(2^n - 1)$ for their analysis [BG11].

We state **Hypothesis 1** because it is what the standard data-complexity formulas of Section 2.4 use, and because our own model keeps it for the off-diagonal transitions (Section 3.3). It is not what we test against. A point mass is refuted by any dispersion at all, so refuting it would say nothing about wrong-key randomization. Every comparison in this paper — the figures of Section 3.4, the survivor counts of Section 3.5, and the claims of Section 5 — is made against **Hypothesis 2**.

3.2.2 Hypothesis 2: random permutation

Hypothesis 2 is the wrong-key randomization hypothesis proper. As mentioned in Section 2.4, the term originated with Harpes, Kramer, and Massey [HKM95] and was first stated for the differential setting by Blondeau and Gérard [BG11]. It allows hit probabilities to vary across wrong keys, and it fixes the whole law rather than just a mean. A permutation of $\mathbb{F}_2^n$ has 2^{n-1} pairs with a given nonzero input difference, so $p_w(k') = j 2^{1-n}$ where j counts the pairs mapped to the output difference b . For a uniform random permutation

$$\Pr[p_w(k') = j 2^{1-n}] \approx \text{Poisson}(j; \lambda), \quad \lambda = \frac{2^{n-1}}{2^n - 1} \approx \frac{1}{2}, \quad (18)$$

by [DR05, Cor. 4] (journal version [DR07b]). The mean is $1/(2^n - 1)$ rather than 2^{-n} : a permutation sends distinct inputs to distinct outputs, so a nonzero input difference never gives a zero output difference, and each of the $2^n - 1$ nonzero output differences is on average equally likely. The rate barely depends on n — it is 0.5000076 at $n = 16$ and $1/2$ to twenty decimals at $n = 64$ — and the Poisson approximation is accurate to within 2^{-17} and 2^{-65} for those two block sizes.

What Hypothesis 2 predicts. The two hypotheses agree on the mean hit probability up to a gap of order 2^{-2n} , but they are not otherwise close. Hypothesis 1 is a point mass; Hypothesis 2 is widely dispersed. The distribution in Equation (18) has standard deviation $\sqrt{1/2}$ about a mean of $1/2$, a relative dispersion of $\sqrt{2}$, and it places

$$\Pr[p_w(k') = 0] = e^{-1/2} = 60.65\%$$

of its mass at zero. A large class of wrong keys with hit probability exactly zero is therefore not on its own evidence against the random-permutation model, and we do not treat it as such; Section 3.4 makes the comparison quantitative.

3.3 A more accurate model

Constructing a better model requires computing fixed-key differential probabilities for all transitions through $G_{k,k'}$. For the on-diagonal transition (b, b) , quasidifferential trails make this tractable. For off-diagonal transitions (γ, b) with $\gamma \neq b$, computing fixed-key

differential probabilities for all 2^n values of γ is infeasible (see Section 4.6). Hypothesis 3 computes the dominant on-diagonal contribution exactly and applies to the tail the *residual WKR assumption*: the point-mass simplification of Hypothesis 1, at the constant 2^{-n} , retained for the off-diagonal transitions only. It is a modeling ingredient here, not a benchmark; Section 3.5 states where it limits what the computed survivor counts can show. Write $p_G(k, k'; b)$ for the on-diagonal fixed-key probability,

$$p_G(k, k'; b) = \Pr_x[G_{k,k'}(x) \oplus G_{k,k'}(x \oplus b) = b].$$

These values partition the key space into ℓ equivalence classes with representative probabilities $p_{G,1}, \dots, p_{G,\ell}$ and weights $\omega_1, \dots, \omega_\ell$ given by the class sizes.

This partition reflects algebraic structure, not enumeration noise. The equivalence classes arise from cosets of a vector space determined by the quasidifferential trails. The computational consequences appear in Section 4.

The on-diagonal probability $p_G(k, k'; b)$ is computed exactly via quasidifferential trails (Section 4). A pair contributes to the counter only if it carries the distinguisher output difference b after E'_k ; this occurs with probability p_R . For these pairs, the on-diagonal differential through $G_{k,k'}$ holds with probability $p_G(k, k'; b)$. The remaining fraction $1 - p_R$ of pairs carry a different output difference $\gamma \neq b$; under the residual WKR assumption, each such pair satisfies the differential with probability 2^{-n} . The wrong-key hit probability $p_w(k')$ of Equation (17) is therefore

$$p_w(k') = p_R \cdot p_G(k, k'; b) + (1 - p_R) \cdot 2^{-n}.$$

Applying this per class gives

$$p_i = p_R \cdot p_{G,i} + (1 - p_R) \cdot 2^{-n}, \quad i = 1, \dots, \ell,$$

with class weights ω_i proportional to the class sizes.

Hypothesis 3 (Structured mixture). *The wrong-key counter distribution is a finite mixture*

$$T(k') \sim \sum_{i=1}^{\ell} \omega_i \text{Binomial}(N, p_i), \quad k' \neq k, \quad (19)$$

where the weights ω_i are positive reals with $\sum_{i=1}^{\ell} \omega_i = 1$, and both they and the probabilities p_i are determined by the fixed-key differential probabilities of the key-recovery map.

Hypothesis 3 is exact for the on-diagonal transition (b, b) through $G_{k,k'}$. The residual WKR assumption applied to off-diagonal transitions (γ, b) with $\gamma \neq b$ is the sole remaining approximation; resolving it exactly would require computing fixed-key probabilities for 2^n different differentials (Section 4.6).

Hypothesis 1 is recovered as $\ell = 1$, $p_1 = 2^{-n}$. Hypothesis 2 is recovered by taking $p_j = j 2^{1-n}$ with weights $\omega_j = \text{Poisson}(j; 1/2)$, a mixture whose components are fixed by the random-permutation model. Our refinement is *structural*: the ω_i and p_i are computed from the cipher by the tool of Section 4, rather than posited.

Table 1 summarizes the progression.

3.4 Structured mixture models of three ciphers

We compute the wrong-key probability distributions of Hypothesis 3 for three differential key-recovery settings. In each of them we enumerate all pairs (k, k') the cipher can realise, so a class counts pairs rather than the wrong keys for one particular k . An attacker instead fixes k and varies only k' . Whether the two distributions agree is a property of the cipher, not a general fact.

Table 1: Progression of wrong-key models. Hypotheses 1–3 are modeling assumptions; the exact benchmark is the ground truth, intractable because it needs fixed-key probabilities for all 2^n output differences. Hypothesis 2 is the reference against which we compare throughout; Hypothesis 1 is the simplification the standard data-complexity formulas use.

Hypothesis	Model	Wrong-key probabilities	Status
Hyp. 1	Point mass	$p_i = 2^{-n}$ for all i	Simplified
Hyp. 2	Random permutation	$p_j = j 2^{1-n}$, $j \sim \text{Poisson}(1/2)$	Reference
Hyp. 3	Structured mixture	On-diag. exact, off-diag. point mass	This paper
Exact	Full computation	All transitions exact	Intractable

3.4.1 Toy cipher

We consider the 1-round differential $(a, b) = (0x0001, 0x0800)$ of the 16-bit toy SPN introduced in [BDD⁺24] as a distinguisher, and add three key-recovery rounds. In this setting, we analyze the differential $(b, b) = (0x0800, 0x0800)$ through the key-recovery map $G_{k,k'} = F_{k'}^{-1} \circ F_k$, and compute the fixed-key probabilities $p_G(k, k')$ for all wrong-key candidates $k' \neq k$. The attacker guesses 14 key bits (2 per active S-box across the three backward rounds; Appendix A traces the propagation). One key bit is redundant, so the effective key space contains 2^{13} key pairs: four of them (the correct key and three equivalent keys due to symmetries) satisfy $p_G(k, k') = 1$, leaving 8,188 wrong keys. For a 16-bit state, Hypothesis 2 predicts $p_w = j 2^{-15}$ with $j \sim \text{Poisson}(1/2)$: 60.65% of wrong keys at $p_w = 0$, a mean of $1/(2^{16} - 1)$, and fewer than one wrong key expected beyond $j = 5$.

Table 2 and Figure 3 present the computed distribution of $p_G(k, k')$ across all 8,188 wrong keys. The mean over wrong keys is $\bar{p} = 30/2047 \approx 2^{-6.09}$, differing from the product-of-DDT heuristic, which assumes independent round behavior; the toy cipher adds only 2 key bits per S-box, inducing inter-round correlation that the quasidifferential computation captures. We find a large zero-probability class: 5,396 wrong keys (65.9%) have $p_G = 0$, i.e., the differential (b, b) never occurs through $G_{k,k'}$ for these keys. Hypothesis 2 places 60.65% of its mass at zero, so this class is close to what the model predicts and is not on its own the discrepancy; the discrepancy is in the tail. The non-zero probabilities cluster into 14 equivalence classes with values ranging from 0.0078 (class 1, containing 1,152 keys) to 0.625 (class 14, containing 4 keys). Read as pair counts, class 1 already has $j = 256$ of the 2^{15} pairs and class 14 has $j = 20,480$, against a Hypothesis 2 mean of $j = 1/2$ — factors of 512 and 40,960. Nothing we compute lies between $j = 0$ and $j = 256$, where Hypothesis 2 puts 39.3% of its mass.

Which right keys see this distribution. For the toy cipher, not every one of them. The computed basis decides this: p_G depends on the pair only through $k \oplus k'$ when every basis vector carries the same mask on the forward and the backward occurrence of a round key, and then fixing k only relabels the classes. Here that fails, so p_G depends on k itself. Fixing k , the wrong keys reach an 11-dimensional subspace of the 13-dimensional class space, so there are four right-key families, one for each of the four classes the diagonal $k = k'$ maps to. The four give only two distributions: they pair up, each pair assigning the same multiset of probabilities under a relabelling of the classes, so the right key selects the distribution through a single bit. The two have mean p_G of 0.015144 and 0.014167, and Table 2 is their average, class by class. Seven of the fourteen nonzero classes move between them, by factors of up to four. The zero class does not move: the 65.9% of Figure 3 is what every right key sees.

Table 2: Distribution of wrong-key probabilities $p_G(k, k')$ for the toy cipher. For a right key k , the guessed key k' ranges over the 8,192 values of the effective guessed subkey: 4 with $p_G = 1$ (right key and equivalents) and 8,188 wrong keys shown here. The entries are averaged over right keys: a fixed k sees one of two histograms, with mean p_G equal to 0.015144 and 0.014167, and the table is their average, class by class. The last two rows give Hypothesis 2 for comparison: $p_G = j 2^{-15}$ with $j \sim \text{Poisson}(1/2)$, which puts 4,966 of the wrong keys at $p_G = 0$ and expects fewer than one of the remaining 3,222 above $j = 5$, that is, above $p_G = 2^{-12.7}$. Every computed class lies above it.

Class	p_G	$\log_2(p_G)$	Keys	% of Keys
0	0	$-\infty$	5,396	65.90%
1	0.0078125	-7.00	1,152	14.07%
2	0.0156250	-6.00	224	2.74%
3	0.0312500	-5.00	672	8.21%
4	0.0468750	-4.42	128	1.56%
5	0.0625000	-4.00	212	2.59%
6	0.0781250	-3.68	96	1.17%
7	0.1250000	-3.00	146	1.78%
8	0.1562500	-2.68	32	0.39%
9	0.1875000	-2.42	58	0.71%
10	0.2500000	-2.00	16	0.20%
11	0.3125000	-1.68	30	0.37%
12	0.3750000	-1.42	10	0.12%
13	0.5000000	-1.00	12	0.15%
14	0.6250000	-0.68	4	0.05%
Hyp. 2, $p_G = 0$		$-\infty$	4,966	60.65%
Hyp. 2, $p_G > 0$		≤ -12.7	3,222	39.35%

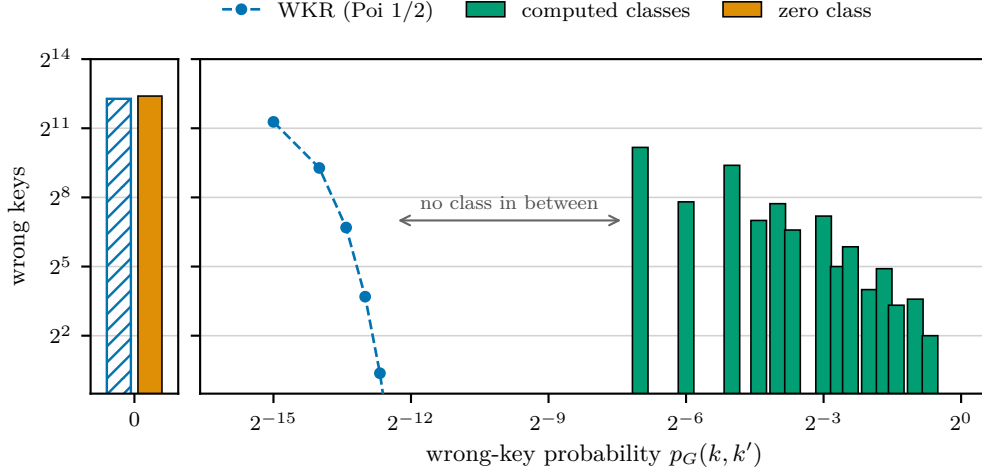


Figure 3: Computed wrong-key distribution for the toy cipher (Table 2) against the random-permutation model of Equation (18). The zero class is drawn in its own panel because $p_G = 0$ has no place on a logarithmic axis: 65.9% of wrong keys have $p_G = 0$, against the 60.65% the model predicts. The computed classes begin at 2^{-7} , by which point the model is already below one expected key, and no class lies in between. The computed bars, like the table, are averaged over right keys.

3.4.2 PRESENT

For the PRESENT block cipher [BKL⁺07], we use Wang’s 14-round differential $(a, b) = (0x0700000000000700, 0x0000000900000009)$ from [Wan08] as a distinguisher, and add two key-recovery rounds on the ciphertext side. This allows us to recover 32 bits of the round keys (2^{32} key candidates). We then analyze the differential $(b, b) = (0x0000000900000009, 0x0000000900000009)$ through $G_{k,k'}$ and compute the fixed-key probabilities $p_G(k, k')$ for all wrong-key candidates. Here the state size is $n = 64$, so Hypothesis 2 predicts $p_w = j 2^{-63}$ with $j \sim \text{Poisson}(1/2)$, again placing 60.65% of the mass at $p_w = 0$, with fewer than one of the 2^{32} candidates expected beyond $j = 9$.

Figure 4 presents the computed distribution for PRESENT against the random-permutation model of Equation (18). A zero-probability class contains 35.4% of all wrong candidates, and the non-zero probabilities span 271 distinct values. The model places its nonzero mass on the multiples of 2^{-63} , and by 2^{-60} it already expects fewer than one wrong key. Every class we compute lies at 2^{-16} or above.

These classes are computed under the independent-round-key assumption of Section 2.5: the pair (k, k') ranges over all round-key values, not over those a PRESENT master key can produce. We have checked that this costs nothing here. The count is exact: the key schedule is affine apart from one S-box between consecutive round keys, so branching on it splits the master keys into 16 pieces on which the class map is affine.⁵ Fixing a right key and letting the wrong key range over all 2^{80} PRESENT master keys, the class map is onto and every class receives exactly 2^{50} master keys, so all 2^{30} classes—and therefore every probability value in Table 11—occur for genuine key-schedule-consistent keys, with the relative frequencies shown. The key schedule does constrain the classes: its linear part alone reaches only a 28-dimensional subspace, and the two remaining dimensions are supplied by the S-box in the key schedule.

⁵A solver-free checker for this count is available as supplementary material.

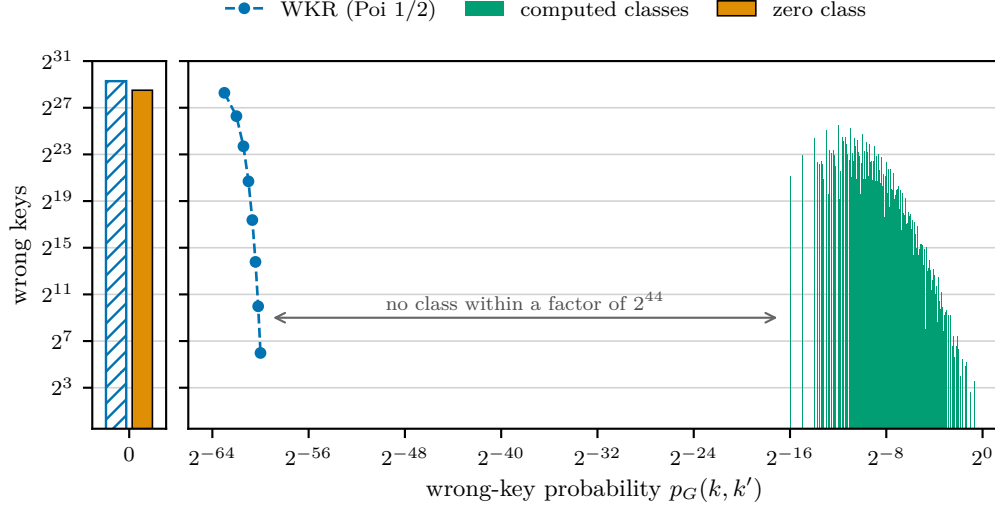


Figure 4: Computed wrong-key distribution for PRESENT (Table 11) against the random-permutation model of Equation (18). The zero class is drawn in its own panel because $p_G = 0$ has no place on a logarithmic axis: here the computed mass at zero is *smaller* than the model predicts, 35.4% against 60.65%. The discrepancy is in the tail. The model’s nonzero support ends at 2^{-60} and the smallest computed class is 2^{-16} , a factor of 2^{44} away, while the largest gap within the computed distribution is a single factor of two.

Which right keys see this distribution. For PRESENT, every one of them. Here the condition holds, and it is checked rather than argued: all 30 basis vectors carry the same mask on the forward and the backward occurrence of a round key. Fixing k then translates the class labels without changing the distribution over them; and since the wrong keys are spread evenly over the 2^{30} classes, that distribution is uniform for every choice of k . Table 11 and Figure 4 therefore describe the wrong-key distribution seen by an attacker, for every right key, not an average over k .

3.4.3 GIFT-64

For GIFT-64 [BPP⁺17] we use the 12-round iterative characteristic of Zhu, Dong and Yu [ZDY18], $\Delta_{\text{in}} = \Delta_{\text{out}} = 0x0000000000001010$ with $p_R = 2^{-60}$, as a distinguisher and add two key-recovery rounds. The block size is again $n = 64$, so Hypothesis 2 predicts $p_w = j 2^{-63}$ with $j \sim \text{Poisson}(1/2)$, placing 60.65% of the mass at $p_w = 0$ and expecting fewer than one candidate beyond $j = 5$. The trail enumeration is uncapped: it returned 98 533 differential characteristics and a trail space of dimension $\dim W = 32$, so the basis is exact rather than a lower bound. We report the distribution over the 2^{16} of those classes that GIFT-64’s 32-bit round key can reach.

Figure 5 presents the computed distribution against the random-permutation model. Of the 65 520 wrong-key classes among them (listed in Table 12), 42.82% have $p_G = 0$ and the remaining 86 distinct values run from 2^{-15} to $2^{-1.91}$, with no gap inside that range wider than a factor of $2^{1.09}$. The model’s own support ends at 2^{-61} , the largest value whose expected population is one candidate, so the two are separated by a factor of 2^{46} with nothing in between. Table 3 places the three ciphers side by side.

Which right keys see this distribution. For GIFT-64, sixteen groups of them. Its restricted trail space splits as $16 = 12 + 4$, so a fixed right key reaches a 12-dimensional subspace of the 16-dimensional class space and there are $2^4 = 16$ right-key families, one for each of the sixteen classes with $p_G = 1$. They give ten distinct distributions: six pairs of families share a histogram and four stand alone. As for the toy cipher, the families that share one carry the same multiset of probabilities under a relabelling of the classes, and no nonzero shift of the class space leaves p_G unchanged, so the coincidence is one of values and not a symmetry. What separates the families is shape rather than location: their means span a factor of 1.07, from $2^{-9.43}$ to $2^{-9.52}$, while their mass at zero ranges from 41.6% to 45.4% and the number of distinct nonzero values they carry from 66 to 82. A summary by mean alone would call all sixteen identical.

What the three ciphers have in common. The classes a cipher’s keys can reach split, in each case, into the subspace the wrong keys reach for a fixed right key and the space of right-key families: $30 = 30 + 0$ for PRESENT, $13 = 11 + 2$ for the toy cipher and $16 = 12 + 4$ for GIFT-64, whose 32-bit round key reaches 16 of the $\dim W = 32$ dimensions the enumeration returns. The second term is the part of the computed basis whose vectors do not carry the same mask on the forward and the backward occurrence of a round key, and it decides how many families there are: one, four and sixteen, each holding exactly one class of probability 1. PRESENT is the only one of the three whose distribution a single right key sees in full, and that is a property of its basis rather than a general fact — all thirty of its vectors happen to be paired. The number of *distinct* distributions is smaller than the number of families in both ciphers that have more than one, and by an amount the dimensions do not predict.

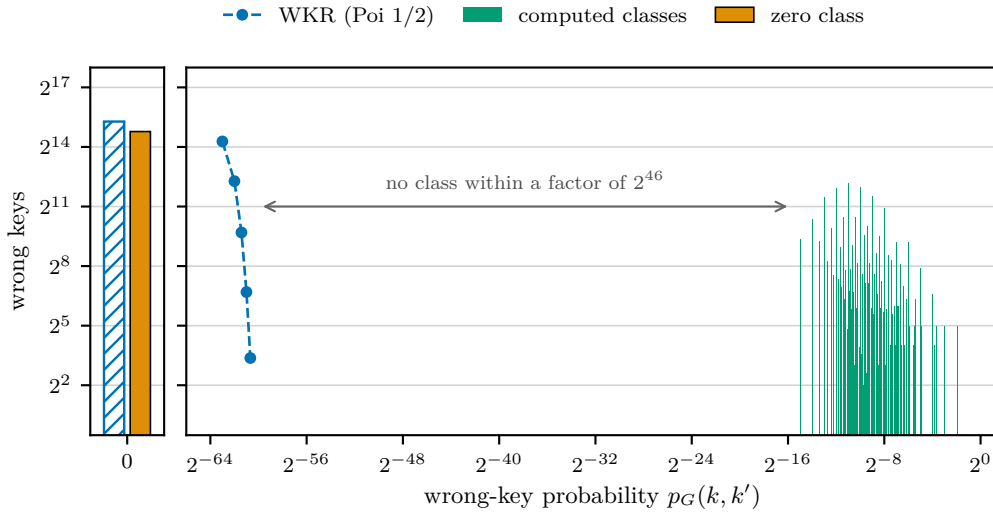


Figure 5: Computed wrong-key distribution for GIFT-64 (Table 12) against the random-permutation model of Equation (18), over the 65 520 wrong-key classes its 32-bit round key can reach, averaged over its sixteen right-key families. The zero class is drawn in its own panel because $p_G = 0$ has no place on a logarithmic axis: 42.82% of the classes have $p_G = 0$ against the 60.65% the model predicts, so as for PRESENT the discrepancy is in the tail. The model’s support ends at 2^{-61} and the smallest computed class is 2^{-15} , a factor of 2^{46} away, while the largest gap within the computed distribution is a factor of $2^{1.09}$.

Table 3: Computed wrong-key distributions against the random-permutation model, Equation (18). The predicted class count is the number of values of j whose expected number of wrong keys is at least one, over 8,188 wrong keys for the toy cipher, 2^{32} for PRESENT and 65 520 for GIFT-64; the Poisson support is unbounded, so it is a typical count rather than a maximum. These three figures are the same for every right key for PRESENT and for the toy cipher, whose individual class weights are not (Table 2); for GIFT-64 they differ, so its column gives the range over its sixteen right-key families.

Quantity	Toy cipher		PRESENT		GIFT-64	
	Hyp. 2	Computed	Hyp. 2	Computed	Hyp. 2	Computed
Mass at $p_w = 0$	60.65%	65.9%	60.65%	35.4%	60.65%	41.6–45.4%
Mass beyond 3 std.	1.44%	34.1%	1.44%	64.6%	1.44%	54.6–58.4%
Distinct classes	≈ 6	15	≈ 10	272	≈ 6	67–83

3.4.4 Comparison with Hypothesis 2

Table 3 puts the three computed distributions against the random-permutation model of Equation (18). The mass at zero behaves differently in each: the toy cipher has somewhat more zeros than the model predicts, PRESENT has considerably fewer, and GIFT-64 sits between them. It is not the main discrepancy in any of the three. The tail is: essentially every nonzero class we compute lies beyond three standard deviations of the model, where it allows 1.44% of the mass, and the number of distinct classes exceeds the prediction by a factor of two for the toy cipher, of twenty-seven for PRESENT and of eleven to fourteen for GIFT-64.

All three rows point at the same thing. What Hypothesis 2 cannot produce is the *shape* of the distributions we compute. They are not unimodal: a mass at zero, a second mass far out in the tail, and nothing between them. For the toy cipher the smallest nonzero class already has $j = 256$ pairs out of 2^{15} , against a mean of $1/2$; for PRESENT it has $j = 2^{47}$.

The obstruction is the family, not the value of its parameter. Call a distribution on $\{0, 1, 2, \dots\}$ unimodal if its probabilities are nondecreasing up to some point and nonincreasing after it; the Poisson and the binomial laws are unimodal for every value of their parameters. No such law describes the distribution of j across wrong keys. (This is a different object from the counter $T(k')$ of Section 2.4, which is binomial over the N pairs of the attack for each fixed k' .) For the toy cipher $\Pr[j = 0] = 65.9\%$ while $\Pr[j] = 0$ for $1 \leq j \leq 255$, so the mode must be at $j = 0$ and the probabilities are nonincreasing from there; hence $\Pr[j = 256] \leq \Pr[j = 1] = 0$, yet 14.1% of wrong keys have exactly $j = 256$. For PRESENT the empty range is $1 \leq j < 2^{47}$.

Fitting the rate only moves the failure. For the toy cipher $\lambda = 0.417$ reproduces the mass at zero and leaves probability 2^{-2008} at $j = 256$, while $\lambda = 480.2$ matches the observed mean and gives $\Pr[j = 0] = 2^{-693}$ against the 65.9% we observe. The class counts move the same way: the rate matching the observed mean inflates the predicted count to about 138 for the toy cipher and 2^{29} for PRESENT, further from what we compute rather than closer. No choice of rate removes these discrepancies together.

These deviations from the WKR hypothesis change the wrong-key counter distribution and therefore can alter the post-filter cost of key recovery. We make this precise next.

3.5 Statistical impact: pairs and distribution shape

We now address two questions. First, among the N pairs used in the attack, which type — right or wrong — contributes more to a wrong key’s counter on average? Second, when does the shape of the wrong-key distribution, not just its mean, change the number of surviving wrong keys?

Pair origin and per-key hitting probability. Of the N pairs, a fraction p_R are right and $1 - p_R$ are wrong. Right pairs carry the distinguisher difference b into $G_{k,k'}$ and hit with class-dependent probability $p_{G,i}$; wrong pairs carry a random $\gamma \neq b$ and hit with 2^{-n} under the residual WKR assumption (Section 3.3). For a differential of probability $p_R = 2^{t-n}$ over an n -bit block, the per-key hitting probability of equivalence class i is

$$p_i = p_R p_{G,i} + (1 - p_R) 2^{-n}.$$

Split the expected counter into the contribution C_1 from wrong pairs and C_2 from right pairs:

$$\mathbb{E}[C_1] = (1 - p_R) N 2^{-n}, \quad \mathbb{E}[C_2] = p_R N \bar{p}_G, \quad (20)$$

where $\bar{p}_G = \sum_i \omega_i p_{G,i}$ is the key-weighted mean class probability. Right pairs dominate when $\mathbb{E}[C_2] > \mathbb{E}[C_1]$. Since $N > 0$ we cancel it from both sides; dividing further by p_R and substituting $p_R = 2^{t-n}$ yields

$$\begin{aligned} \mathbb{E}[C_2] > \mathbb{E}[C_1] &\iff p_R \bar{p}_G > (1 - p_R) 2^{-n} \\ &\iff \bar{p}_G > \frac{1 - p_R}{p_R} 2^{-n} = 2^{-t} (1 - 2^{t-n}) = 2^{-t} - 2^{-n} \approx 2^{-t}. \end{aligned} \quad (21)$$

Equivalently, right pairs dominate iff

$$p_R > \frac{2^{-n}}{\bar{p}_G + 2^{-n}} = \frac{2^{-n}}{2^{-n}(1 + 2^n \bar{p}_G)} = \frac{1}{1 + 2^n \bar{p}_G} \approx \frac{2^{-n}}{\bar{p}_G}. \quad (22)$$

The last step uses $2^n \bar{p}_G \gg 1$, which holds for all three targets. For the toy cipher ($t = 13$, $n = 16$) we have $\bar{p}_G \approx 2^{-6.1}$, so the crossover sits at $p_R^* \approx 2^{-9.9}$, well below the actual $p_R = 2^{t-n} = 2^{-3}$: right pairs dominate. For PRESENT ($t = 2$, $n = 64$) we have $\bar{p}_G \approx 2^{-10.5}$, giving $p_R^* \approx 2^{-53.5}$, above the actual $p_R = 2^{-62}$: wrong pairs dominate by about 8.5 bits.

Bottleneck classes. A wrong key for which the differential $b \rightarrow b$ through $G_{k,k'}$ holds almost always behaves like the right key on right pairs: its per-pair probability $p_i \approx p_R$ matches the correct key's. We call its equivalence class a *bottleneck*. Such a key is hard to separate from the correct one by counter value, so a bottleneck class limits how far ranking can narrow the candidate list. How many keys the filter leaves is a separate question, since that also depends on the class weight ω_i . A bottleneck of small weight obstructs ranking and contributes almost nothing to the survivor count, which is what happens for the toy cipher, as the survivor counts below show.

Does the shape matter, or only the mean? The crossover in Equation (22) settles which pair type dominates. It says nothing about the second question. We answer that one with a lemma, then evaluate it for the toy cipher and PRESENT.

Write $\mathcal{K}_w$ for the set of wrong keys. Let $\mathcal{D}$ be the wrong-key distribution $\{(\omega_i, p_i)\}$ from Hypothesis 3: drawing a wrong key $K_w \in \mathcal{K}_w$ uniformly selects class i with probability ω_i , giving per-key hitting probability $p_{K_w} = p_i$. Its mean is $\bar{p} = p_R \bar{p}_G + (1 - p_R) 2^{-n}$. Fix $\tau \geq 1$ and let $\varphi_\tau(q) := \Pr[\text{Bin}(N, q) \geq \tau]$ be the probability that a wrong key with hitting probability q reaches the threshold. We write $\mathcal{N}_{\text{surv}}^{(\tau)}$ for the expected number of wrong keys that survive the filter at threshold τ . A second superscript marks the same quantity computed under another model: $\mathcal{N}_{\text{surv}}^{(\tau), \text{mean}}$ evaluates every wrong key at the mean $\bar{p}$; $\mathcal{N}_{\text{surv}}^{(\tau), \text{H2}}$ draws the wrong-key probabilities from Equation (18), and is the reference we compare against; $\mathcal{N}_{\text{surv}}^{(\tau), \text{H1}}$ evaluates every wrong key at 2^{-n} .

Lemma 1 (General wrong-key distributions govern post-filter complexity). *The expected number of surviving wrong keys for a filter threshold τ , under any arbitrary wrong-key distribution $\mathcal{D}$, is*

$$\mathcal{N}_{\text{surv}}^{(\tau)} = |\mathcal{K}_w| \mathbb{E}_{K_w \sim \mathcal{D}}[\varphi_\tau(p_{K_w})].$$

Approximating by the mean alone assumes all wrong keys behave identically at $\bar{p}$:

$$\mathcal{N}_{\text{surv}}^{(\tau), \text{mean}} = |\mathcal{K}_w| \varphi_\tau(\bar{p}).$$

Because φ_τ is nonlinear, the two values diverge depending on the threshold:

1. **Threshold $\tau = 1$:** Here $\varphi_1(q) = 1 - (1 - q)^N$ is strictly concave on $[0, 1]$, so Jensen's inequality applies directly,

$$\mathbb{E}_{K_w \sim \mathcal{D}}[\varphi_1(p_{K_w})] \leq \varphi_1(\mathbb{E}_{K_w \sim \mathcal{D}}[p_{K_w}]) = \varphi_1(\bar{p}),$$

that is, $\mathcal{N}_{\text{surv}}^{(1)} \leq \mathcal{N}_{\text{surv}}^{(1), \text{mean}}$: the mean-only approximation overestimates the surviving keys, with equality iff all wrong keys in $\mathcal{D}$ share the same probability.

2. **Higher thresholds ($\tau \geq 2$):** Here φ_τ is convex on $[0, p^*)$ and concave on $(p^*, 1]$, with inflection $p^* = (\tau - 1)/(N - 1)$. Since $Np^* \approx \tau - 1$, the inflection sits just below the probability where a wrong key's expected counter Np equals the threshold. The sign of the mean-only error is fixed by where the wrong-key probabilities fall relative to p^* :

- (a) *all wrong-key probabilities below p^* : φ_τ is convex, so Jensen's inequality gives the reverse bound and $\mathcal{N}_{\text{surv}}^{(\tau)} \geq \mathcal{N}_{\text{surv}}^{(\tau), \text{mean}}$ — the mean underestimates.*
- (b) *all wrong-key probabilities above p^* : φ_τ is concave, as for $\tau = 1$, so $\mathcal{N}_{\text{surv}}^{(\tau)} \leq \mathcal{N}_{\text{surv}}^{(\tau), \text{mean}}$ — the mean overestimates.*

When the wrong-key probabilities fall on both sides of p^* , curvature alone does not fix the sign and the two values have to be compared directly. Since p^* grows with τ while the probabilities do not, a single $\mathcal{D}$ is overestimated at low thresholds and underestimated at high ones. Where the crossover falls is a property of $\mathcal{D}$ and is not determined by $\bar{p}$ alone: for the toy cipher of [Section 3.4](#) it lies between $\tau = 2$ and $\tau = 3$, and for $\mathcal{D}_B$ of the bimodal example below near $\tau = 5.6$, against $N\bar{p} = 0.9$ and 1 respectively.

Proof. By definition, the expected number of survivors is

$$\mathcal{N}_{\text{surv}}^{(\tau)} = \sum_{K_w \in \mathcal{K}_w} \Pr[T(K_w) \geq \tau] = |\mathcal{K}_w| \mathbb{E}[\varphi_\tau(p_{K_w})].$$

For a threshold of $\tau = 1$, the survival function $\varphi_1(q) = 1 - (1 - q)^N$ is strictly concave since $\varphi_1''(q) = -N(N - 1)(1 - q)^{N-2} < 0$ for $N \geq 2$. By Jensen's inequality, $\mathbb{E}[\varphi_1(p_{K_w})] \leq \varphi_1(\bar{p})$, with equality if and only if p_{K_w} takes a single value with probability 1 under $\mathcal{D}$. For $\tau \geq 2$, write $\varphi_\tau(q) = \sum_{j=\tau}^N \binom{N}{j} q^j (1 - q)^{N-j}$ and differentiate term by term. Consecutive terms cancel and only one survives:

$$\varphi'_\tau(q) = N \binom{N-1}{\tau-1} q^{\tau-1} (1 - q)^{N-\tau} > 0,$$

so φ_τ is strictly increasing. Differentiating once more,

$$\varphi''_\tau(q) = N \binom{N-1}{\tau-1} q^{\tau-2} (1 - q)^{N-\tau-1} [(\tau - 1) - (N - 1)q],$$

whose sign is that of the bracket: positive for $q < p^* := (\tau - 1)/(N - 1)$ and negative for $q > p^*$. Hence φ_τ is convex below p^* and concave above it. When the support of $\mathcal{D}$ lies entirely below p^* — as in the sparse regime ($N\bar{p} \leq 1$) for a concentrated distribution — φ_τ is convex there, so the inequality goes the other way and $\mathbb{E}[\varphi_\tau(p_{K_w})] \geq \varphi_\tau(\bar{p})$ (case (a)). Symmetrically, when the support lies entirely above p^* , φ_τ is concave and Jensen gives the upper bound $\mathbb{E}[\varphi_\tau(p_{K_w})] \leq \varphi_\tau(\bar{p})$ (case (b)); in both cases equality holds if and only if p_{K_w} takes a single value with probability 1 under $\mathcal{D}$. $\square$

Survivor counts against the random-permutation model. Lemma 1 says the survivor count depends on the whole wrong-key distribution and not on its mean. We therefore evaluate it under the computed mixture and under **Hypothesis 2**, which is the reference. Under **Equation (18)** the wrong-key probability is $j 2^{1-n}$ with j Poisson, so

$$\mathcal{N}_{\text{surv}}^{(\tau)} = |\mathcal{K}_w| \sum_i \omega_i \varphi_\tau(p_i), \quad \mathcal{N}_{\text{surv}}^{(\tau), \text{H2}} = |\mathcal{K}_w| \sum_{j \geq 0} \text{Poisson}(j; 1/2) \varphi_\tau(j 2^{1-n}),$$

with $p_i = p_R p_{G,i} + (1 - p_R) 2^{-n}$ as above. Which of the two is larger depends on the threshold, through the convexity switch of Lemma 1: below p^* dispersion raises the survivor count, above it lowers it. **Table 4** evaluates both for the toy cipher and PRESENT, together with $\mathcal{N}_{\text{surv}}^{(\tau), \text{mean}}$, which collapses our own distribution to its mean. The ratio $\mathcal{N}_{\text{surv}}^{(\tau)} / \mathcal{N}_{\text{surv}}^{(\tau), \text{mean}}$ is the part of the survivor count that the mean cannot account for. The rest of the gap to **Hypothesis 2** has two further sources: the difference between the two means, and the dispersion of the reference model itself. The second is not always small, contributing a factor of 1.01 at $\tau = 1$ for the toy cipher and 249 at $\tau = 5$.

For the toy cipher the computed distribution puts 31 times more wrong keys through the filter than **Hypothesis 2** predicts at $\tau = 1$. Most of that comes from the means, $\bar{p} = 2^{-9.08}$ against the model's $1/(2^{16} - 1)$: a ratio of 121, which the saturation of φ_1 turns into a factor of 79 in the survivor count. The shape works the other way here, *lowering* the count by a factor of 2.6. Lemma 1 gives that direction at $\tau = 1$, where $p^* = 0$ and φ_1 is concave over the whole range. By $\tau = 5$ the shape has reversed and raises the count by a factor of 21, with the crossover between $\tau = 2$ and $\tau = 3$. At $\tau = 5$ the inflection sits at $p^* = 7.8 \cdot 10^{-3}$, with 7,784 of the 8,188 wrong keys below it and 404 above, so this is the lemma's mixed case: curvature does not fix the sign, and we computed it.

The gap for the toy cipher comes from the whole nonzero part of the spectrum, not from the bottleneck class. All 2,792 wrong keys with $p_G > 0$ sit at $p_i \geq 9.9 \cdot 10^{-4}$, six times the largest probability **Hypothesis 2** assigns to any wrong key it expects among 8,188 ($j = 5$, or $1.5 \cdot 10^{-4}$). The bottleneck class contributes 4 of the 1,953 survivors, as anticipated on **Page 20**: it obstructs ranking without moving the survivor count.

For PRESENT the shape contributes nothing measurable, $\mathcal{N}_{\text{surv}}^{(\tau)}$ and $\mathcal{N}_{\text{surv}}^{(\tau), \text{mean}}$ agreeing to within 0.03% at every threshold. At $p_R = 2^{-62}$ the structured term $p_R p_{G,i}$ is negligible beside the residual term $(1 - p_R) 2^{-n}$ for every class, so the computed distribution is nearly a point mass, with almost no dispersion left for the threshold to act on. The crossover of **Equation (22)** reached this conclusion for the expected counters; the survivor counts now show it directly.

The comparison with **Hypothesis 2** nonetheless reverses at $\tau \geq 2$, the model predicting *more* survivors than we compute. How much more depends on where it is read. At $\tau = 2$ the factor is 1.5 and the right key still survives with probability 0.59; at $\tau = 5$ it is 47, but P_S has fallen to 0.05, so the largest discrepancy sits at a threshold where the attack has already failed.

That gap comes from the model's own dispersion rather than from anything we compute for PRESENT. All our probabilities satisfy $p_i \leq 0.44 p^*$ at $\tau = 5$, so φ_τ is convex across our whole distribution and case (a) of Lemma 1 applies exactly. The random-permutation model instead places 0.02% of its mass at or above p^* , which lifts its survivor count without

Table 4: Expected surviving wrong keys under the computed mixture (**Hypothesis 3**), under that mixture collapsed to its mean, and under the random-permutation model (**Hypothesis 2**), for a filter threshold τ . The toy cipher uses $N = 2^9$ pairs, 64 expected right pairs, over 8,188 candidates. PRESENT uses $N = 2^{63}$, which is the whole codebook: a fixed input difference admits at most 2^{n-1} pairs, so no attack on a 64-bit block has more. That gives two expected right pairs at $p_R = 2^{-62}$, over 2^{32} candidates. P_S is the probability that the right key survives, and is reported because the largest discrepancies sit at thresholds where the attack itself is failing. The *shape* column is $\mathcal{N}_{\text{surv}}^{(\tau)} / \mathcal{N}_{\text{surv}}^{(\tau), \text{mean}}$, the part of the survivor count that the mean cannot account for. For the toy cipher it crosses 1 between $\tau = 2$ and $\tau = 3$, in the direction Lemma 1 gives; for PRESENT it is 1 to within 0.03% at every threshold.

τ	Expected survivors			Ratio to		
	$\mathcal{N}_{\text{surv}}^{(\tau)}$	$\mathcal{N}_{\text{surv}}^{(\tau), \text{mean}}$	$\mathcal{N}_{\text{surv}}^{(\tau), \text{H2}}$	shape	Hyp. 2	P_S
Toy cipher, $N = 2^9$, 8,188 wrong keys						
1	1,953	5,008	63.2	0.39	31	1.00
2	1,262	1,997	0.73	0.63	$1.7 \cdot 10^3$	1.00
5	486	23.3	$4.8 \cdot 10^{-7}$	21	$1.0 \cdot 10^9$	1.00
PRESENT, $N = 2^{63}$, 2^{32} wrong keys						
1	$2^{30.66}$	$2^{30.66}$	$2^{30.12}$	1.00	1.46	0.86
2	$2^{28.54}$	$2^{28.54}$	$2^{29.13}$	1.00	0.66	0.59
5	$2^{19.51}$	$2^{19.51}$	$2^{25.06}$	1.00	0.021	0.05

changing the direction. Spreading mass below the inflection raises the count. The model is dispersed and our distribution is not, so any cipher with a distinguisher probability this small would behave the same way.

Two limits on what Table 4 shows. First, its PRESENT column agrees with the point-mass value $\mathcal{N}_{\text{surv}}^{(\tau), \text{H1}} = |\mathcal{K}_w| \varphi_\tau(2^{-n})$ to within 1.4% at every entry, but the residual WKR assumption produces that agreement, being itself the point mass at 2^{-n} . It is not independent evidence for **Hypothesis 1**. What the column does establish is that the on-diagonal structure, the part we compute rather than assume, moves no survivor count at this distinguisher probability. Second, **Hypothesis 2** caps the survivor count at $(1 - e^{-1/2}) |\mathcal{K}_w|$, because 60.65% of wrong keys have hit probability exactly zero and can never survive. The computed mixture has no such cap, since the residual term keeps every class strictly positive.

A 2^{10} complexity gap hidden by the mean. The shape factors in Table 4 are limited by how far apart the class probabilities of those ciphers happen to lie. The following example shows there is no general bound. It is the same comparison as the shape column of that table — a distribution against a point mass at its own mean — with the ratio quoted the other way up, and it behaves the same way with τ : the mean overestimates at low thresholds and underestimates above $\tau \approx 5.6$. Consider two wrong-key distributions with an identical mean probability of $\bar{p} = 2^{-40}$, using $N = 2^{40}$ pairs (so $N\bar{p} = 1$):

- $\mathcal{D}_A$ (point mass): Every wrong key has exactly $\Pr[p_{K_w} = 2^{-40}] = 1$.
- $\mathcal{D}_B$ (structured): A small fraction of keys hit a differential plateau, giving $\Pr[p_{K_w} = 2^{-30}] = 2^{-10}$, while the rest sit at 0.

Under a threshold of $\tau = 1$, Lemma 1 yields drastically different survivor counts:

$$\mathbb{E}_{\mathcal{D}_A}[\varphi_1(p_{K_w})] = 1 - (1 - 2^{-40})^{2^{40}} \approx 1 - e^{-1} \approx 0.632,$$

$$\mathbb{E}_{\mathcal{D}_B}[\varphi_1(p_{K_w})] = 2^{-10}(1 - (1 - 2^{-30})^{2^{40}}) \approx 2^{-10}.$$

The residual exhaustive search complexity differs by roughly a factor of 2^{10} ($\mathcal{N}_{\text{surv}}^{(A)}/\mathcal{N}_{\text{surv}}^{(B)} \approx 2^{9.34}$), despite identical means. This gap grows without bound as the structured mass shifts to larger values, so collapsing the distribution to its mean is unsound.

However, this gap is highly τ -dependent. At a stricter threshold of $\tau = 5$, the flat distribution $\mathcal{D}_A$ yields a binomial tail probability of ≈ 0.0037 , while the structured class in $\mathcal{D}_B$ still contributes $\approx 2^{-10}$. The ratio collapses from roughly 2^{10} to 4. Higher filtering thresholds compress the gap because the survival probability becomes dominated by the binomial tail rather than the structural spikes.

When does $\bar{p}$ alone suffice? The mean is only an adequate proxy for $\mathbb{E}[\varphi_\tau(p_{K_w})]$ when the wrong-key probabilities concentrate tightly around $\bar{p}$. Let $\delta = \sqrt{\text{Var}_{K_w}(p_{K_w})}$. A second-order Taylor expansion at $\tau = 1$ shows how variance distorts the approximation:

$$\mathbb{E}[\varphi_1(p_{K_w})] \approx \varphi_1(\bar{p}) - \frac{1}{2}N(N-1)(1-\bar{p})^{N-2}\delta^2$$

The correction term matters once δ is comparable to $1/N$. In differential cryptanalysis, this approximation fails because δ grows when (i) the key-recovery map depends on only a fraction of the wrong-key bits, so equivalent keys cluster; or (ii) the key schedule or the differential trail makes the fixed-key probability take the same, larger value across a whole subset of the wrong keys. A large expected hit count $\lambda = N\bar{p}$ does not by itself make the mean safe. Take a class of weight 2^{-10} at 2^{-20} with the remaining wrong keys at zero, and $N = 2^{40}$: here $\lambda = 2^{10}$, yet the mean overestimates the survivor count by a factor of 2^{10} : the correction term depends on δ , not on λ .

Table 4 and the crossover of Equation (22) answer the two questions this subsection opened with. The shape of the wrong-key distribution changes the survivor count when the right pairs carry enough weight. For PRESENT with Wang’s distinguisher they do not, and $p_R = 2^{-62}$ leaves the residual term in control. The numbers use the spectra of Section 3.4; Section 4 gives the method that produces them.

4 Computing fixed-key probabilities

This section describes the computation: the mask-first reformulation of the fixed-key probability formula, the dimension-reduction argument, the SMT trail enumerator, and the certified mask dimensions of the three targets, whose distributions Section 3.4 reports.

The mixture model requires the exact class weights ω_i and probabilities $p_{G,i}$ for $G_{k,k'}$. Computing these quantities directly from the quasidifferential formula is infeasible, and even a naive application of the fast Walsh–Hadamard transform (FWHT) does not help. We outline the two baselines before describing our method.

Naive double sum. Theorem 1 expresses the fixed-key probability $p_G(\kappa)$ as a double sum over all differential characteristics and all intermediate mask sequences $(u_2, \dots, u_s) \in \mathbb{F}_2^{(s-1)n}$. Evaluating this sum for a single key requires iterating over both spaces simultaneously, giving a doubly exponential cost in the number of rounds. For the toy cipher ($r = 3$ key-recovery rounds, so $s = 2r = 6$ in the key-recovery map, $n = 16$), this is infeasible; for PRESENT ($r = 2$, so $s = 4$, $n = 64$), it is entirely infeasible.

Naive FWHT. One can reorganize the computation by first assembling the full mask-correlation function $B: \mathbb{F}_2^{(s-1)n} \rightarrow \mathbb{R}$, which aggregates quasidifferential trail correlations over all characteristics for each intermediate mask vector. The mask-correlation function is formally defined in Section 4.1. We then apply the FWHT to recover all key values

simultaneously. This eliminates the double sum, but the transform still operates over the full $(s-1)n$ -dimensional space at cost $O((s-1)n \cdot 2^{(s-1)n})$. For the toy cipher, this is $80 \cdot 2^{80}$; for PRESENT, $192 \cdot 2^{192}$; both are entirely infeasible.

Our approach. For a mask vector u , let $B(u)$ collect the correlations of all quasidifferential trails carrying that mask vector; it is defined in Equation (24). We show that the support $T = \{u \in \mathbb{F}_2^{(s-1)n} : B(u) \neq 0\}$ of the mask-correlation function lies in a subspace $W = \text{span}(T)$ of dimension $d \ll (s-1)n$. Restricting the FWHT to W reduces the transform cost to $O(d \cdot 2^d)$, and a basis of W is recovered by an SMT-guided two-stage search exploiting the algebraic structure of Proposition 1. The reduction is substantial: for the toy cipher, $d = 13$ versus $(s-1)n = 80$, cutting cost from $80 \cdot 2^{80}$ to $13 \cdot 2^{13}$; for PRESENT, $d = 30$ versus 192, from $192 \cdot 2^{192}$ to $30 \cdot 2^{30}$. This independently corroborates Kumar [Kum12], who observed that Wang’s attack on PRESENT recovers at most 30 of the 32 guessed subkey bits. The SMT basis construction is the dominant runtime cost in practice; the FWHT step itself is negligible once the basis is known. Applying the same two-stage basis search with $r = 2$ key-recovery rounds yields $d = 35$ for Rectangle [ZBL⁺15] and $d = 32$ for GIFT-64 [BPP⁺17], against $(s-1)n = 192$ in both cases, so the transform cost falls from $192 \cdot 2^{192}$ to $35 \cdot 2^{35}$ and $32 \cdot 2^{32}$ respectively. We carried the computation through for GIFT-64 (Section 3.4); for Rectangle the restricted FWHT is at the edge of practicality and partial trail enumeration via our SMT tool is future work. The dimensions for PRESENT, Rectangle and GIFT-64 are each backed by a machine-checkable certificate in the supplementary material: d independent quasidifferential trails, each checked against the S-box tables without a solver, establish $\dim W \geq d$, and the matching upper bound is the completeness query of Section 4.3 returning unsatisfiable. The following subsections present the method for a generic keyed permutation, then specialize to $G_{k,k'}$.

4.1 Mask-first fixed-key probability computation

Generic setting. Let $G_\kappa : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ be a keyed s -round permutation whose round keys are independent (Section 2.5). Only the $s-1$ round keys added between the rounds affect the quantity we compute, since Theorem 1 fixes the outer masks to zero and a round key paired with a zero mask contributes nothing; we write $\kappa \in \mathbb{F}_2^{(s-1)n}$ for their concatenation and call it the round-key vector. We compute the distribution of fixed-key probabilities $p_G(\kappa) = \Pr_x[\Delta G_\kappa(x) = \beta \mid \Delta x = \alpha]$ over all κ .

Application to key recovery. For the key-recovery map $G_{k,k'} = F_{k'}^{-1} \circ F_k$ of Definition 1, we specialize to $(\alpha, \beta) = (b, b)$ where b is the output difference of the distinguisher, with $s = 2r$ rounds (r forward in F_k , r backward in $F_{k'}^{-1}$). The guessed pair $k||k'$ determines the round-key vector κ ; the attacker’s m guessed bits per key therefore enter through an embedding of $\mathbb{F}_2^{2m}$ into $\mathbb{F}_2^{(s-1)n}$, and it is the latter over which the transform below runs.

From characteristics to masks. By Theorem 1, quasidifferential trails express the fixed-key probability of the differential (α, β) through G_κ as a double sum:

$$p_G(\kappa) = \sum_{a_2, \dots, a_s} \sum_{u_2, \dots, u_s} \prod_{i=1}^s D_{(u_{i+1}, a_{i+1}), (u_i, a_i)}^{G_{i,\kappa}} \quad (23)$$

where the outer sum ranges over all characteristics $(a_1, \dots, a_{s+1})$ with $a_1 = \alpha$ and $a_{s+1} = \beta$, and the inner sum ranges over all intermediate mask sequences $u_2, \dots, u_s \in \mathbb{F}_2^n$ with $u_1 = u_{s+1} = 0$ (restricting to the DDT submatrix of the quasidifferential transition matrix).

Key idea: reverse summation order. Equation (23) can be computed in two ways: fix characteristics first and sum over masks (as in [BR22]), or fix masks first and sum over characteristics. We take the latter approach because it exposes Walsh–Hadamard structure and sparsity in the mask space.

We define the mask correlation function. For each mask vector $u = (u_2, \dots, u_s) \in \mathbb{F}_2^{(s-1)n}$, we aggregate correlations over all characteristics:

$$B : \mathbb{F}_2^{(s-1)n} \rightarrow \mathbb{R}, \quad B(u) = \sum_{a_2, \dots, a_s} \prod_{i=1}^s D_{(u_{i+1}, a_{i+1}), (u_i, a_i)}^{G_i}, \quad (24)$$

where $a_1 = \alpha$, $a_{s+1} = \beta$ and $u_1 = u_{s+1} = 0$, following the convention of Theorem 1.

Most mask vectors $u \in \mathbb{F}_2^{(s-1)n}$ do not induce any quasidifferential trails with nonzero correlation. High-correlation trails tend to be local [BR22, Sec. 4.3], with nonzero masks concentrated in a few rounds rather than all $(s-1)$ intermediate states. Consequently, the support T lies in a low-dimensional subspace with $d = \dim(\text{span}(T)) \ll (s-1)n$ (Section 3.4).

The key dependence of each entry factors as a sign $(-1)^{\langle u, \kappa \rangle}$: the mask vector u paired with the round-key vector κ , so that replacing κ by $\kappa + \delta$ multiplies the entry by $(-1)^{\langle u, \delta \rangle}$. Reversing the summation order in Equation (23) yields:

$$p_G(\kappa) = \sum_{u \in T} B(u) (-1)^{\langle u, \kappa \rangle}. \quad (25)$$

Exploiting structure via the Fast Walsh–Hadamard transform. The sum in Equation (25) is the Walsh–Hadamard transform of B over $\mathbb{F}_2^{(s-1)n}$. The transform $\hat{B}$ maps each key κ to $p_G(\kappa)$; its preimages are the key classes.

The FWHT [Car21] computes $\hat{B}$ efficiently. However, applying the FWHT directly over the full $(s-1)n$ -dimensional space yields complexity $O((s-1)n \cdot 2^{(s-1)n})$, which remains prohibitively expensive.

4.2 Dimension reduction and algebraic structure

Recall from Section 4.1 that $B(u)$ is nonzero only for masks in the support $T = \{u \in \mathbb{F}_2^{(s-1)n} : B(u) \neq 0\}$. Write $W = \text{span}(T)$ for the subspace it spans and $d = \dim(W)$. We restrict the FWHT to W .

Let $\{e_1, \dots, e_d\}$ be a basis for W and define the change-of-basis matrix $M = [e_1 \mid \dots \mid e_d] \in \mathbb{F}_2^{(s-1)n \times d}$.

We restrict $\hat{B}$ to W by defining $h : \mathbb{F}_2^d \rightarrow \mathbb{R}$ as

$$h(\xi) := B(M \cdot \xi) \quad (26)$$

where $\xi \in \mathbb{F}_2^d$ represents coordinates in the basis $\{e_1, \dots, e_d\}$. Substituting into Equation (25) yields

$$\hat{B}(\kappa) = \sum_{\xi \in \mathbb{F}_2^d} (-1)^{\langle \kappa, M\xi \rangle} \cdot h(\xi). \quad (27)$$

Recursive FWHT. The FWHT can be computed recursively by decomposing along coordinates. Partitioning $\mathbb{F}_2^d = \Xi_0 \sqcup \Xi_1$ where $\Xi_b = \{\xi : \xi_d = b\}$ gives

$$\hat{B}(\kappa) = \sum_{\xi \in \Xi_0} (-1)^{\langle \kappa, M'\xi' \rangle} h(\xi) + (-1)^{\langle \kappa, e_d \rangle} \sum_{\xi \in \Xi_1} (-1)^{\langle \kappa, M'\xi' \rangle} h(\xi),$$

where $M' = [e_1 \mid \dots \mid e_{d-1}]$ and $\xi' = (\xi_1, \dots, \xi_{d-1})$. Applying this recursively yields complexity $O(d \cdot 2^d)$.

4.2.1 Mask space structure

The basis computation exploits algebraic structure in the mask space. For a fixed differential characteristic $\mathbf{a} = (a_1, \dots, a_{s+1})$ with $a_1 = \alpha$ and $a_{s+1} = \beta$, define

$$V(\mathbf{a}) = \{u \in \mathbb{F}_2^{(s-1)n} : B(u) \neq 0 \text{ and } u \text{ induces a trail for } \mathbf{a}\} \quad (28)$$

as the set of mask vectors inducing trails for this characteristic. Equivalently, $T = \bigcup_{\mathbf{a}} V(\mathbf{a})$.

Proposition 1 ([BR22, Thm. 4.2(a)]). *For a function $G_\kappa = G_{s,\kappa} \circ \dots \circ G_{1,\kappa}$ and a characteristic $\mathbf{a} = (a_1, \dots, a_{s+1})$, let $p_{\mathbf{a}}$ denote the correlation of the quasidifferential trail for $\mathbf{a}$ in which all masks are zero; by Section 2.2 this trail is the ordinary differential characteristic, so $p_{\mathbf{a}}$ is its probability. The following holds:*

If $(u_1, a_1), \dots, (u_{s+1}, a_{s+1})$ is a quasidifferential trail with correlation $(-1)^b p_{\mathbf{a}}$ where $b \in \{0, 1\}$, then for every quasidifferential trail $(v_1, a_1), \dots, (v_{s+1}, a_{s+1})$ with correlation c , the trail $(v_1 + u_1, a_1), \dots, (v_{s+1} + u_{s+1}, a_{s+1})$ has correlation $(-1)^b c$.

An immediate consequence is that $V_0(\mathbf{a}) = \{u \in V(\mathbf{a}) : |B(u)| = p_{\mathbf{a}}\}$, the set of masks preserving the characteristic's correlation absolute value, forms a vector space. Moreover, defining $W(\mathbf{a}) = \text{span}(V(\mathbf{a}))$, Proposition 1 implies that masks in the same coset $u + V_0(\mathbf{a})$ induce trails with equal correlation absolute value. Thus, the quotient space $W(\mathbf{a})/V_0(\mathbf{a})$ partitions masks by correlation absolute values.

Basis computation. This analysis assumes a basis for $W = \text{span}(T)$ is available. To enumerate all quasidifferential trails for a characteristic $\mathbf{a}$, it suffices to compute two bases: $\mathcal{B}_0$ for $V_0(\mathbf{a})$ (correlation-preserving masks) and $\mathcal{B}_1$ for coset representatives of $W(\mathbf{a})/V_0(\mathbf{a})$ (distinct correlation absolute values). We recover them by SMT search and Gaussian elimination (Section 4.3, Section 4.4).

4.3 Two-stage basis search (algorithm)

This section describes how we exploit this structure to avoid exhaustive enumeration of the $2^{(s-1)n}$ mask space. Our implementation uses SMT (Section 4.4).

The strategy applies to any s -round keyed permutation; our implementation specializes to PRESENT-like SPNs.

Two-stage basis search. For each differential characteristic $\mathbf{a}$ separately, we compute bases $\mathcal{B}_0$ for $V_0(\mathbf{a})$ and $\mathcal{B}_1$ for representatives of $W(\mathbf{a})/V_0(\mathbf{a})$ via SMT-guided enumeration:

Stage 1: Finding $\mathcal{B}_0$. We query the SMT solver for masks $u \in \mathbb{F}_2^{(s-1)n}$ satisfying two constraints:

- $B(u) \neq 0$ (induces at least one trail), encoded via QDTM (Section 2.2) consistency.
- $|B(u)| = p_{\mathbf{a}}$ (preserves correlation absolute value).

Each solution is tested for linear independence from previously found basis elements; if independent, it extends $\mathcal{B}_0$.

Stage 2: Finding $\mathcal{B}_1$. We search for masks u with $B(u) \neq 0$ that lie in distinct cosets $u + V_0(\mathbf{a})$. The SMT constraint $u \notin \text{span}(\mathcal{B}_0 \cup \mathcal{B}_1)$ ensures each solution represents a new coset. Stage 2 ends when this query returns unsatisfiable, certifying that no coset is missing.

Sieving invalid cosets. Not every vector in $\text{span}(\mathcal{B}_1)$ induces valid trails; Proposition 1 guarantees structure only for $V(\mathbf{a})$, not its span. After Stage 2, we enumerate all $2^{|\mathcal{B}_1|}$ linear combinations and verify each against the QDTM via direct table lookup (not SMT). Invalid combinations are discarded.

This enumeration remains tractable because $|\mathcal{B}_1| \ll (s-1)n$ in practice: most trails have similar correlation absolute values, yielding few distinct cosets. For the toy cipher with $(s-1)n = 80$, we observe $|\mathcal{B}_1| \leq 5$; for PRESENT with $(s-1)n = 192$, we find $|\mathcal{B}_1| \leq 8$.

Merging across characteristics. After computing bases $\mathcal{B}(\mathbf{a}) = \mathcal{B}_0(\mathbf{a}) \cup \mathcal{B}_1(\mathbf{a})$ for each characteristic $\mathbf{a}$ separately, we obtain the overall basis via Gaussian elimination on the union $S = \bigcup_{\mathbf{a}} \mathcal{B}(\mathbf{a})$. Since different characteristics may share mask vectors, $|S|$ is typically larger than the dimension $d = \dim(W)$. Gaussian elimination over $\mathbb{F}_2$ reduces S to a linearly independent set $\{e_1, \dots, e_d\}$ spanning $W = \sum_{\mathbf{a}} W(\mathbf{a})$. This basis is used in the FWHT (Section 4.2).

The per-characteristic dimension $\dim W(\mathbf{a})$ varies with $\mathbf{a}$, though within a narrow band: across the 160 characteristics of the toy cipher it ranges over $\{8, 9, 10\}$, from 16 to 22 across the 11,664 characteristics of PRESENT, and from 15 to 25 across the 98,533 characteristics of GIFT-64. The unions of those spaces have dimensions $d = 13, 30$ and 32 , respectively, matching the dimensions reported in Section 4.2. Thus, the merge recovers the full trail space from the per-characteristic pieces.

4.4 SMT-based trail enumeration for PRESENT-like SPNs

We instantiate the two-stage basis search for PRESENT-like SPNs. The SMT solver encodes trail validity as bit-vector constraints, enabling systematic enumeration without exhaustive search.

4.4.1 SMT Model for Quasidifferential Trails

Variable encoding. A quasidifferential trail is represented by two families of bit-vector variables (see Figure 6):

- *Input variables* (u_i, a_i) for $i = 1, \dots, s$, where $u_i \in \mathbb{F}_2^n$ is the mask and $a_i \in \mathbb{F}_2^n$ is the difference at the input to the nonlinear layer in round i .
- *Output variables* (v_i, b_i) for $i = 1, \dots, s$, representing the mask and difference at the output of the nonlinear layer.

The first r rounds correspond to the forward direction (F_k) and the remaining r rounds to the backward direction (F_k^{-1}). The bit-permutation layer $\mathcal{P}$ enforces deterministic linear constraints between consecutive rounds: $u_{i+1} = \mathcal{P}(v_i)$. For key-recovery on the plaintext side, the model uses $\mathcal{P}^{-1}$ and $\mathcal{S}^{-1}$ instead.

Constraint types. We use the Bitwuzla SMT solver [NP23] with constraints in two categories:

Validity: Each S-box transition must appear in the QDTM. For differential trails where all masks are zero, the DDT suffices. Bit-permutation constraints are equality constraints between bit-vector variables.

Cost: Each QDTM (or DDT) entry with correlation c is assigned integer weight $w = -\lceil \log_2 |c| \rceil$, following standard practice [ARS⁺22; RR22]. Our basis search performs full enumeration of all masks with $B(u) \neq 0$, but the cost function enables prioritized search when only the highest-probability trails are needed.

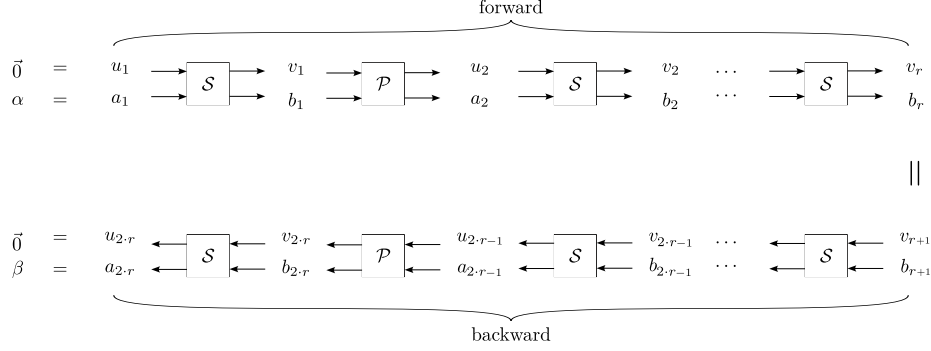


Figure 6: SMT variable structure for quasidifferential trails through the key-recovery map $G_{k,k'} = F_{k'}^{-1} \circ F_k$. The top path shows the forward direction (F_k) and the bottom shows the backward direction ($F_{k'}^{-1}$). $\mathcal{S}$ denotes the S-box layer and $\mathcal{P}$ the bit-permutation layer. Variables (u_i, a_i) represent masks and differences at S-box inputs; (v_i, b_i) at S-box outputs.

QDTM encoding strategy. The full QDTM for a 4-bit S-box has 2^{16} entries (input mask, input difference, output mask, output difference). Encoding the complete table is feasible for small S-boxes but becomes prohibitive when considering all possible differential characteristics simultaneously.

We fix a characteristic and enumerate trails for it. This allows encoding only the QDTM submatrices corresponding to the differential transitions (a_i, a_{i+1}) in the characteristic, reducing table size by orders of magnitude. When switching to a different characteristic, we reset the solver and reload the relevant submatrices. This requires recomputation, but the memory savings justify the overhead. This aligns with the two-stage search of Section 4.3.

4.5 Performance and scope

Our implementation is available as open-source software at <https://doi.org/10.5281/zenodo.22129299> and consists of Python modules for SMT model construction, trail enumeration, and post-processing (FWHT and Gaussian elimination for basis computation). For the PRESENT experiments in Section 3.4, enumerating quasidifferential trails over 11,664 differential characteristics with $r = 2$ key-recovery rounds required approximately 8 hours on a single Intel Core i7-8665U CPU without parallelization. The dominant cost is SMT solving during basis construction. Memory usage is controlled by processing characteristics sequentially and resetting the solver between instances.

Our tool is specialized for 4-bit S-boxes and bit-permutations, but the algebraic structure applies generically. Extending to more general linear layers would require encoding the corresponding constraints.

4.6 Scope: On-diagonal vs. off-diagonal in key recovery

The method of this section computes fixed-key probabilities for a *single* differential (α, β) through G_κ . Its tractability depends only on the dimension d of the mask correlation support, not on the specific differential.

In the key-recovery context of Section 3.3, the on-diagonal transition (b, b) requires computing fixed-key probabilities for one differential. For PRESENT, our SMT search finds $d = 30$, giving a cost of $30 \cdot 2^{30}$ (Section 4.5).

The off-diagonal transitions (γ, b) for $\gamma \neq b$ would require fixed-key probabilities for 2^n different differentials. Even with comparable dimension reduction for each, 2^n independent searches would be infeasible. This asymmetry is handled by the residual WKR assumption.

Clustering gives the distinguisher a set of likely output differences $b_1, \dots, b_\ell$ rather than a single b . Our method extends to that setting without change: once the cluster and its probabilities are known, run the tool once per b_i under the same key constraints, at ℓ times the cost. We did not do so for computational reasons and leave it to future work.

5 Conclusion

We gave the Wrong-Key Randomization hypothesis a concrete mathematical object—the key-recovery map $G_{k,k'} = F_{k'}^{-1} \circ F_k$ —and a computable quantity. Three advances made this possible.

First, the mask-first reformulation of the fixed-key probability formula exposes a Walsh–Hadamard structure in the intermediate-mask space. Second, exploiting the algebraic structure of quasidifferential trails restricts the transform to a subspace of dimension $d \ll (s-1)n$, reducing complexity from $\mathcal{O}(2^{(s-1)n})$ to $\mathcal{O}(d \cdot 2^d)$. For the toy cipher $d = 13$; for PRESENT, $d = 30$; for GIFT-64, $d = 32$. Third, an SMT-guided two-stage basis search makes this subspace computable for practical targets.

The resulting wrong-key distributions are structured mixtures. For the toy cipher, every wrong key with positive class probability p_G sits well above anything the random-permutation model expects, and by [Lemma 1](#) the filter therefore leaves 31 times more wrong keys than that model predicts at a threshold of $\tau = 1$. A bottleneck class—four keys out of 2^{13} with $p_G = 0.625$ —behaves almost like the right key on right pairs; it contributes little to that count, but it is what makes those four keys hard to rank apart from the right one. For PRESENT, the subspace dimension $d = 30$ independently corroborates Kumar’s observation that Wang’s attack recovers at most 30 effective subkey bits [[Kum12](#)]; the analogous bottleneck class exists, but its signal reaches the wrong-key counters only through the right pairs, whose weight is the distinguisher probability $p_R = 2^{-62}$, which is negligible. Even at the full codebook of 2^{63} pairs the on-diagonal structure we compute changes the survivor count by less than 0.03%. For GIFT-64 the mixture has the same shape, but its sixteen right-key families give ten distinct distributions, so the one an attacker faces depends on which right key is in use—and they differ far more in shape than in mean.

WKR therefore remains a safe heuristic for PRESENT with Wang’s 14-round distinguisher even though it is formally false: the structured deviation is real, but at this distinguisher probability it cannot surface. The random-permutation model still differs from the computed survivor count, by a factor of 1.5 at a usable threshold, but that difference comes from the dispersion the model itself carries, not from anything specific to PRESENT. Whether it surfaces for other ciphers or distinguishers depends on the shape of the computed distribution and on the attack’s data volume—exactly the quantities our method makes accessible.

These conclusions rest on treating the round keys as independent, so the classes we compute are classes of the pair (k, k') ranging over independent round keys rather than over master keys. For PRESENT this costs nothing: every class is realised by key-schedule-consistent keys, each by exactly 2^{50} master keys, so the classes, the weights ω_i and the survivor counts are the same whether the round keys are treated as independent or derived from a master key. The mask space and its dimension d are unaffected in either case, being properties of the round function and the linear layer rather than of the key values. We have not checked the other ciphers, and the independent-round-key model remains an assumption for them. For those, the key schedule is better imposed natively—as a constraint inside the trail enumeration—than as a filter on the classes afterwards: a filter can only discard classes already computed, whereas a constraint keeps the enumeration from producing them at all.

Several directions remain open. The SMT-based basis search is the runtime bottle-

neck; specialised solvers exploiting the vector-space structure of $V_0(\mathbf{a})$ could push the method to ciphers with larger mask dimensions, such as Rectangle. Extending beyond bit-permutations to MDS linear layers requires only new SMT constraint encoding, though the growth of d under stronger diffusion is an empirical question. The method applies to any keyed permutation—not only the key-recovery map—and applying it to the forward cipher E_k would provide per-key distinguisher probabilities, the natural companion to the present work. Our tool is publicly available at <https://doi.org/10.5281/zenodo.22129299>.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. This work was (in part) supported by the European Research Council (ERC) project 101097056 (SYMTRUST) and the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy – EXC 2092 CASA – 390781972. Tim Beyne is supported by a postdoctoral fellowship from the Research Foundation – Flanders (FWO) with reference number 1274724N.

References

- [AFK⁺08] Jean-Philippe Aumasson, Simon Fischer, Shahram Khazaei, Willi Meier, and Christian Rechberger. New features of latin dances: analysis of Salsa, ChaCha, and Rumba. In Kaisa Nyberg, editor, *FSE 2008*, volume 5086 of *LNCS*, pages 470–488. Springer, Berlin, Heidelberg, February 2008. DOI: [10.1007/978-3-540-71039-4_30](https://doi.org/10.1007/978-3-540-71039-4_30).
- [AG89] R. Arratia and L. Gordon. Tutorial on large deviations for the binomial distribution. *Bulletin of Mathematical Biology*, 51(1):125–131, January 1989. ISSN: 1522-9602. DOI: [10.1007/bf02458840](https://doi.org/10.1007/bf02458840).
- [ARS⁺22] Seyyed Arash Azimi, Adrián Ranea, Mahmoud Salmasizadeh, Javad Mohajeri, Mohammad Reza Aref, and Vincent Rijmen. A bit-vector differential model for the modular addition by a constant and its applications to differential and impossible-differential cryptanalysis. *Designs, Codes and Cryptography*, 90(8):1797–1855, 2022. DOI: [10.1007/s10623-022-01074-8](https://doi.org/10.1007/s10623-022-01074-8).
- [BBB⁺21] Anubhab Baksi, Shivam Bhasin, Jakub Breier, Mustafa Khairallah, Thomas Peyrin, Sumanta Sarkar, and Siang Meng Sim. DEFAULT: cipher level resistance against differential fault attack. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part II*, volume 13091 of *LNCS*, pages 124–156. Springer, Cham, December 2021. DOI: [10.1007/978-3-030-92075-3_5](https://doi.org/10.1007/978-3-030-92075-3_5).
- [BBS99] Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In Jacques Stern, editor, *EUROCRYPT’99*, volume 1592 of *LNCS*, pages 12–23. Springer, Berlin, Heidelberg, May 1999. DOI: [10.1007/3-540-48910-X_2](https://doi.org/10.1007/3-540-48910-X_2).
- [BDBN23] Christina Boura, Nicolas David, Rachelle Heim Boissier, and María Naya-Plasencia. Better steady than speedy: full break of SPEEDY-7-192. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part IV*, volume 14007 of *LNCS*, pages 36–66. Springer, 2023. DOI: [10.1007/978-3-031-30634-1_2](https://doi.org/10.1007/978-3-031-30634-1_2).

- [BDD⁺24] Christina Boura, Nicolas David, Patrick Derbez, Rachelle Heim Boissier, and María Naya-Plasencia. A generic algorithm for efficient key recovery in differential attacks - and its associated tool. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part I*, volume 14651 of *LNCS*, pages 217–248. Springer, Cham, May 2024. DOI: [10.1007/978-3-031-58716-0_8](https://doi.org/10.1007/978-3-031-58716-0_8).
- [BDG25] Christina Boura, Patrick Derbez, and Baptiste Germon. Extending the quasidifferential framework: from fixed-key to expected differential probability. *IACR Transactions on Symmetric Cryptology*, 2025(1):515–541, 2025. DOI: [10.46586/tosc.v2025.i1.515-541](https://doi.org/10.46586/tosc.v2025.i1.515-541).
- [BG11] Céline Blondeau and Benoît Gérard. Multiple differential cryptanalysis: theory and practice. In Antoine Joux, editor, *FSE 2011*, volume 6733 of *LNCS*, pages 35–54. Springer, Berlin, Heidelberg, February 2011. DOI: [10.1007/978-3-642-21702-9_3](https://doi.org/10.1007/978-3-642-21702-9_3).
- [BGT11] Céline Blondeau, Benoît Gérard, and Jean-Pierre Tillich. Accurate estimates of the data complexity and success probability for various cryptanalyses. *Designs, Codes and Cryptography*, 59(1-3):3–34, 2011. DOI: [10.1007/s10623-010-9452-2](https://doi.org/10.1007/s10623-010-9452-2).
- [BKL⁺07] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsøe. PRESENT: an ultra-lightweight block cipher. In Pascal Paillier and Ingrid Verbauwhede, editors, *CHES 2007*, volume 4727 of *LNCS*, pages 450–466. Springer, Berlin, Heidelberg, September 2007. DOI: [10.1007/978-3-540-74735-2_31](https://doi.org/10.1007/978-3-540-74735-2_31).
- [BN24] Tim Beyne and Addie Neyt. Note on the cryptanalysis of speedy. Cryptology ePrint Archive, Report 2024/262, 2024. URL: <https://eprint.iacr.org/2024/262>.
- [BN25] Tim Beyne and Addie Neyt. Improved differential cryptanalysis of SPEEDY. *IACR Transactions on Symmetric Cryptology*, 2025(3):444–474, 2025. DOI: [10.46586/tosc.v2025.i3.444-474](https://doi.org/10.46586/tosc.v2025.i3.444-474).
- [BPP⁺17] Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Yosuke Todo. GIFT: A small present - towards reaching the limit of lightweight encryption. In Wieland Fischer and Naofumi Homma, editors, *CHES 2017*, volume 10529 of *LNCS*, pages 321–345. Springer, Cham, September 2017. DOI: [10.1007/978-3-319-66787-4_16](https://doi.org/10.1007/978-3-319-66787-4_16).
- [BR22] Tim Beyne and Vincent Rijmen. Differential cryptanalysis in the fixed-key model. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part III*, volume 13509 of *LNCS*, pages 687–716. Springer, Cham, August 2022. DOI: [10.1007/978-3-031-15982-4_23](https://doi.org/10.1007/978-3-031-15982-4_23).
- [BR25] Tim Beyne and Vincent Rijmen. *Linear Cryptanalysis*. Cambridge University Press, 2025. ISBN: 978-1-009-60787-2. DOI: [10.1017/9781009607872](https://doi.org/10.1017/9781009607872).
- [BS91] Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems. In Alfred J. Menezes and Scott A. Vanstone, editors, *CRYPTO'90*, volume 537 of *LNCS*, pages 2–21. Springer, Berlin, Heidelberg, August 1991. DOI: [10.1007/3-540-38424-3_1](https://doi.org/10.1007/3-540-38424-3_1).
- [BT14] Andrey Bogdanov and Elmar Tischhauser. On the wrong key randomisation and key equivalence hypotheses in Matsui's algorithm 2. In Shiho Moriai, editor, *FSE 2013*, volume 8424 of *LNCS*, pages 19–38. Springer, Berlin, Heidelberg, March 2014. DOI: [10.1007/978-3-662-43933-3_2](https://doi.org/10.1007/978-3-662-43933-3_2).

- [Car21] Claude Carlet. *Boolean Functions for Cryptography and Coding Theory*. Cambridge University Press, 2021.
- [CF25] Anne Canteaut and Merlin Fruchon. Understanding unexpected fixed-key differential behaviours: how to avoid major weaknesses in lightweight designs. In Goichiro Hanaoka and Bo-Yin Yang, editors, *ASIACRYPT 2025, Part I*, volume 16245 of *LNCS*, pages 97–130. Springer, Singapore, December 2025. DOI: [10.1007/978-981-95-5018-0_4](https://doi.org/10.1007/978-981-95-5018-0_4).
- [DEKM16] Christoph Dobraunig, Maria Eichlseder, Daniel Kales, and Florian Mendel. Practical key-recovery attack on MANTIS5. *IACR Transactions on Symmetric Cryptology*, 2016(2):248–260, 2016. ISSN: 2519-173X. DOI: [10.13154/tosc.v2016.i2.248-260](https://doi.org/10.13154/tosc.v2016.i2.248-260).
- [DEM16] Christoph Dobraunig, Maria Eichlseder, and Florian Mendel. Higher-order cryptanalysis of LowMC. In Soonhak Kwon and Aaram Yun, editors, *ICISC 15*, volume 9558 of *LNCS*, pages 87–101. Springer, Cham, November 2016. DOI: [10.1007/978-3-319-30840-1_6](https://doi.org/10.1007/978-3-319-30840-1_6).
- [DR05] Joan Daemen and Vincent Rijmen. Probability distributions of correlation and differentials in block ciphers. Cryptology ePrint Archive, Report 2005/212, 2005. URL: <https://eprint.iacr.org/2005/212>.
- [DR07a] Joan Daemen and Vincent Rijmen. Plateau characteristics. *IET Information Security*, 1(1):11–17, March 2007. DOI: [10.1049/iet-ifs:20060099](https://doi.org/10.1049/iet-ifs:20060099).
- [DR07b] Joan Daemen and Vincent Rijmen. Probability distributions of correlation and differentials in block ciphers. *Journal of Mathematical Cryptology*, 1(3):221–242, 2007. DOI: [10.1515/JMC.2007.011](https://doi.org/10.1515/JMC.2007.011).
- [FT25] Antonio Flórez-Gutiérrez and Yosuke Todo. Improved cryptanalysis of ChaCha: beating PNBs with bit puncturing. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part I*, volume 15601 of *LNCS*, pages 427–457. Springer, Cham, May 2025. DOI: [10.1007/978-3-031-91107-1_15](https://doi.org/10.1007/978-3-031-91107-1_15).
- [Hey20] Howard M. Heys. Key dependency of differentials: experiments in the differential cryptanalysis of block ciphers using small S-boxes. Cryptology ePrint Archive, Report 2020/1349, 2020. URL: <https://eprint.iacr.org/2020/1349>.
- [HKM95] Carlo Harpes, Gerhard G. Kramer, and James L. Massey. A generalization of linear cryptanalysis and the applicability of Matsui’s piling-up lemma. In Louis C. Guillou and Jean-Jacques Quisquater, editors, *EUROCRYPT’95*, volume 921 of *LNCS*, pages 24–38. Springer, Berlin, Heidelberg, May 1995. DOI: [10.1007/3-540-49264-X_3](https://doi.org/10.1007/3-540-49264-X_3).
- [Knu93] Lars R. Knudsen. Iterative characteristics of DES and s^2 -DES. In Ernest F. Brickell, editor, *CRYPTO’92*, volume 740 of *LNCS*, pages 497–511. Springer, Berlin, Heidelberg, August 1993. DOI: [10.1007/3-540-48071-4_35](https://doi.org/10.1007/3-540-48071-4_35).
- [Knu98] Lars R. Knudsen. DEAL – A 128-bit Block Cipher. Technical report 151, Department of Informatics, University of Bergen, Bergen, Norway, February 1998. Submitted as an AES candidate.
- [Kum12] Manoj Kumar. Rectified differential cryptanalysis of 16 round PRESENT. *International Journal on Cryptography and Information Security (IJCIS)*, 2(4), December 2012. URL: <https://ssrn.com/abstract=3663887>. Available at SSRN.
- [LH94] Susan K. Langford and Martin E. Hellman. Differential-linear cryptanalysis. In Yvo Desmedt, editor, *CRYPTO’94*, volume 839 of *LNCS*, pages 17–25. Springer, Berlin, Heidelberg, August 1994. DOI: [10.1007/3-540-48658-5_3](https://doi.org/10.1007/3-540-48658-5_3).

- [LMM91] Xuejia Lai, James L. Massey, and Sean Murphy. Markov ciphers and differential cryptanalysis. In Donald W. Davies, editor, *EUROCRYPT'91*, volume 547 of *LNCS*, pages 17–38. Springer, Berlin, Heidelberg, April 1991. DOI: [10.1007/3-540-46416-6_2](https://doi.org/10.1007/3-540-46416-6_2).
- [LSW⁺22] Fukang Liu, Santanu Sarkar, Gaoli Wang, Willi Meier, and Takanori Isobe. Algebraic meet-in-the-middle attack on LowMC. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part I*, volume 13791 of *LNCS*, pages 225–255. Springer, Cham, December 2022. DOI: [10.1007/978-3-031-22963-3_8](https://doi.org/10.1007/978-3-031-22963-3_8).
- [NDE22] Marcel Nageler, Christoph Dobraunig, and Maria Eichlseder. Information-combining differential fault attacks on DEFAULT. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part III*, volume 13277 of *LNCS*, pages 168–191. Springer, Cham, May 2022. DOI: [10.1007/978-3-031-07082-2_7](https://doi.org/10.1007/978-3-031-07082-2_7).
- [NP23] Aina Niemetz and Mathias Preiner. Bitwuzla. In Constantin Enea and Akash Lal, editors, *Computer Aided Verification - 35th International Conference, CAV 2023, Paris, France, July 17-22, 2023, Proceedings, Part II*, volume 13965 of *Lecture Notes in Computer Science*, pages 3–17. Springer, 2023. DOI: [10.1007/978-3-031-37703-7_1](https://doi.org/10.1007/978-3-031-37703-7_1).
- [NP33] Jerzy Neyman and Egon S. Pearson. On the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A*, 231(694–706):289–337, 1933. DOI: [10.1098/rsta.1933.0009](https://doi.org/10.1098/rsta.1933.0009).
- [PT22] Thomas Peyrin and Quan Quan Tan. Mind your path: on (key) dependencies in differential characteristics. *IACR Transactions on Symmetric Cryptology*, 2022(4):179–207, 2022. DOI: [10.46586/tosc.v2022.i4.179-207](https://doi.org/10.46586/tosc.v2022.i4.179-207).
- [RR22] Adrián Ranea and Vincent Rijmen. Characteristic automated search of cryptographic algorithms for distinguishing attacks (CASCADA). *IET Information Security*, 16(6):470–481, 2022. DOI: [10.1049/ise2.12077](https://doi.org/10.1049/ise2.12077).
- [Sel08] Ali Aydin Selçuk. On probability of success in linear and differential cryptanalysis. *Journal of Cryptology*, 21(1):131–147, January 2008. DOI: [10.1007/s00145-007-9013-7](https://doi.org/10.1007/s00145-007-9013-7).
- [Tez16] Cihangir Tezcan. Differential factors revisited: corrected attacks on PRESENT and SERPENT. In Tim Güneysu, Gregor Leander, and Amir Moradi, editors, *Lightweight Cryptography for Security and Privacy – LightSec 2015*, volume 9542 of *LNCS*, pages 21–33. Springer, 2016. DOI: [10.1007/978-3-319-29078-2_2](https://doi.org/10.1007/978-3-319-29078-2_2).
- [TIM⁺18] Yosuke Todo, Takanori Isobe, Willi Meier, Kazumaro Aoki, and Bin Zhang. Fast correlation attack revisited - cryptanalysis on full Grain-128a, Grain-128, and Grain-v1. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 129–159. Springer, Cham, August 2018. DOI: [10.1007/978-3-319-96881-0_5](https://doi.org/10.1007/978-3-319-96881-0_5).
- [Wag99] David Wagner. The boomerang attack. In Lars R. Knudsen, editor, *FSE'99*, volume 1636 of *LNCS*, pages 156–170. Springer, Berlin, Heidelberg, March 1999. DOI: [10.1007/3-540-48519-8_12](https://doi.org/10.1007/3-540-48519-8_12).
- [Wan08] Meiqin Wang. Differential cryptanalysis of reduced-round PRESENT. In Serge Vaudenay, editor, *AFRICACRYPT 08*, volume 5023 of *LNCS*, pages 40–49. Springer, Berlin, Heidelberg, June 2008. DOI: [10.1007/978-3-540-68164-9_4](https://doi.org/10.1007/978-3-540-68164-9_4).

- [ZBL⁺15] Wentao Zhang, Zhenzhen Bao, Dongdai Lin, Vincent Rijmen, Bohan Yang, and Ingrid Verbauwhede. RECTANGLE: a bit-slice lightweight block cipher suitable for multiple platforms. *Science China Information Sciences*, 58(12):1–15, 2015. DOI: [10.1007/s11432-015-5459-7](https://doi.org/10.1007/s11432-015-5459-7).
- [ZDY18] Baoyu Zhu, Xiaoyang Dong, and Hongbo Yu. MILP-based differential attack on round-reduced GIFT. Cryptology ePrint Archive, Report 2018/390, 2018. URL: <https://eprint.iacr.org/2018/390>.

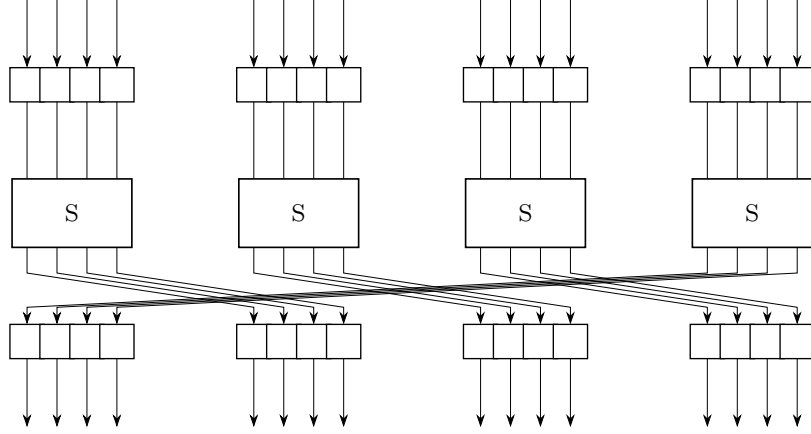


Figure 7: Schematic view of the round function of the toy-cipher (*bit 0 on the right*)

Table 5: Permutation table for the toy cipher

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P(i)$	4	5	2	3	12	13	10	11	0	1	6	7	8	9	14	15

Table 6: The GIFT S-box

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S(i)$	1	10	4	12	6	15	3	9	2	13	11	7	5	0	8	14

Table 7: Active S-boxes per backward key-recovery round. S-box 1 is the rightmost nibble (bits 0–3).

Backward round	S-box 1 (0–3)	S-box 2 (4–7)	S-box 3 (8–11)	S-box 4 (12–15)	Key bits guessed
1 (outermost)	•	•	•	•	8
2	•	•	–	–	4
3 (innermost)	–	–	•	–	2
	Total				14

A Cipher specifications

Toy cipher

The tables below give the S-box and the bit permutation of the toy cipher of Section 2.5, and Figure 7 shows its round function.

Toy cipher key-recovery propagation

For the distinguisher output difference $b = 0x0800$, the backward key-recovery rounds activate S-box nibbles as follows:

The attacker guesses 14 key bits, but $\dim(W) = 13$ means one bit is linearly dependent in the mask space and does not create a distinct counter-distribution class; the effective key space under the mixture model is $2^{13} = 8,192$.

Table 8: S-box of the PRESENT cipher

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S(i)$	12	5	6	11	9	0	10	13	3	14	15	8	4	7	1	2

Table 9: Permutation table for the PRESENT cipher

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P(i)$	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P(i)$	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$P(i)$	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P(i)$	12	28	44	60	13	29	45	61	14	30	46	62	15	31	47	63

Table 10: The GIFT-64 bit permutation

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P(i)$	0	17	34	51	48	1	18	35	32	49	2	19	16	33	50	3
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P(i)$	4	21	38	55	52	5	22	39	36	53	6	23	20	37	54	7
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$P(i)$	8	25	42	59	56	9	26	43	40	57	10	27	24	41	58	11
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P(i)$	12	29	46	63	60	13	30	47	44	61	14	31	28	45	62	15

PRESENT

The tables below give the S-box and the bit permutation of PRESENT, described in [Section 2.5](#).

GIFT-64

Table 10 gives the bit permutation of GIFT-64; its S-box is the one of [Table 6](#), which the toy cipher shares.

B Complete PRESENT wrong-key distribution

Table 11: All wrong-key probability equivalence classes for PRESENT. Here, **probability** is the class probability p_G , **nreprs** the number of classes at that probability, **keys** the number of 32-bit subkey candidates those classes contain (four per class, $\sum = 2^{32}$), and **rel_freq** their fraction of the total. The 2^{30} classes are realised by key-schedule-consistent PRESENT-80 master keys, exactly 2^{50} each, so this is the distribution an attacker sees for any fixed right key rather than an average over right keys (Section 3.4).

probability	nreprs	keys	rel_freq
0.00000000	380209599	1 520 838 396	0.35410
0.00001526	2359296	9 437 184	0.00220
0.00003052	7864320	31 457 280	0.00732
0.00006104	22282240	89 128 960	0.02075
0.00007629	5111808	20 447 232	0.00476
0.00008392	4718592	18 874 368	0.00439
0.00009155	5505024	22 020 096	0.00513
0.00009918	4718592	18 874 368	0.00439
0.00010681	1966080	7 864 320	0.00183
0.00012207	35528704	142 114 816	0.03309
0.00013733	786432	3 145 728	0.00073
0.00015259	10878976	43 515 904	0.01013
0.00016785	8847360	35 389 440	0.00824
0.00018311	11042816	44 171 264	0.01028
0.00019836	7864320	31 457 280	0.00732
0.00021362	4063232	16 252 928	0.00378
0.00024414	47046656	188 186 624	0.04382
0.00025940	589824	2 359 296	0.00055
0.00027466	3080192	12 320 768	0.00287
0.00030518	23035904	92 143 616	0.02145
0.00033569	18055168	72 220 672	0.01682
0.00036621	23134208	92 536 832	0.02155
0.00038147	2768896	11 075 584	0.00258
0.00039673	15728640	62 914 560	0.01465
0.00041962	5111808	20 447 232	0.00476
0.00042725	8306688	33 226 752	0.00774
0.00045776	6062080	24 248 320	0.00565
0.00046158	2359296	9 437 184	0.00220
0.00048828	40792064	163 168 256	0.03799
0.00049591	5111808	20 447 232	0.00476
0.00050354	5505024	22 020 096	0.00513
0.00051880	983040	3 932 160	0.00092
0.00053406	2129920	8 519 680	0.00198
0.00054550	4718592	18 874 368	0.00439
0.00054932	8978432	35 913 728	0.00836
0.00058746	1966080	7 864 320	0.00183
0.00059509	5505024	22 020 096	0.00513
0.00061035	21641216	86 564 864	0.02015
0.00064087	2293760	9 175 040	0.00214
0.00064468	2359296	9 437 184	0.00220
0.00067139	13737984	54 951 936	0.01279

Continued on next page

Continued from previous page

probability	nreprs	keys	rel_freq
0.00068665	851968	3 407 872	0.00079
0.00069427	1966080	7 864 320	0.00183
0.00073242	20013056	80 052 224	0.01864
0.00074768	409600	1 638 400	0.00038
0.00075531	786432	3 145 728	0.00073
0.00076294	2555904	10 223 616	0.00238
0.00079346	9314304	37 257 216	0.00867
0.00082397	917504	3 670 016	0.00085
0.00083923	3424256	13 697 024	0.00319
0.00085449	7387136	29 548 544	0.00688
0.00089264	786432	3 145 728	0.00073
0.00091553	4939776	19 759 104	0.00460
0.00092316	983040	3 932 160	0.00092
0.00096130	327680	1 310 720	0.00031
0.00097656	28490304	113 961 216	0.02653
0.00099182	2359296	9 437 184	0.00220
0.00100708	3014656	12 058 624	0.00281
0.00103760	1966080	7 864 320	0.00183
0.00106812	1835008	7 340 032	0.00171
0.00109100	983040	3 932 160	0.00092
0.00109863	9981952	39 927 808	0.00930
0.00117493	1196032	4 784 128	0.00111
0.00119019	1867776	7 471 104	0.00174
0.00122070	17711104	70 844 416	0.01649
0.00123596	65536	262 144	0.00006
0.00128174	1695744	6 782 976	0.00158
0.00129700	638976	2 555 904	0.00060
0.00134277	9542656	38 170 624	0.00889
0.00137329	2310144	9 240 576	0.00215
0.00138855	786432	3 145 728	0.00073
0.00142670	589824	2 359 296	0.00055
0.00146484	15718912	62 875 648	0.01464
0.00149536	327680	1 310 720	0.00031
0.00151062	1933312	7 733 248	0.00180
0.00152588	2826240	11 304 960	0.00263
0.00155640	688128	2 752 512	0.00064
0.00158691	5308416	21 233 664	0.00494
0.00164795	2375680	9 502 720	0.00221
0.00167847	3301376	13 205 504	0.00307
0.00168610	589824	2 359 296	0.00055
0.00170898	5484544	21 938 176	0.00511
0.00178528	1769472	7 077 888	0.00165
0.00181580	245760	983 040	0.00023
0.00183105	5480448	21 921 792	0.00510
0.00184631	790528	3 162 112	0.00074
0.00192261	868352	3 473 408	0.00081
0.00195312	13657088	54 628 352	0.01272
0.00198364	2064384	8 257 536	0.00192
0.00201416	2863104	11 452 416	0.00267

Continued on next page

Continued from previous page

probability	nreprs	keys	rel_freq
0.00207520	1164288	4 657 152	0.00108
0.00213623	1732608	6 930 432	0.00161
0.00218201	688128	2 752 512	0.00064
0.00219727	7393280	29 573 120	0.00689
0.00228882	106496	425 984	0.00010
0.00233459	98304	393 216	0.00009
0.00234985	892928	3 571 712	0.00083
0.00238037	1671168	6 684 672	0.00156
0.00244141	8559104	34 236 416	0.00797
0.00247192	294912	1 179 648	0.00027
0.00251770	98304	393 216	0.00009
0.00256348	1523712	6 094 848	0.00142
0.00259399	294912	1 179 648	0.00027
0.00268555	3361280	13 445 120	0.00313
0.00274658	1556480	6 225 920	0.00145
0.00277710	442368	1 769 472	0.00041
0.00285339	122880	491 520	0.00011
0.00292969	6968832	27 875 328	0.00649
0.00297546	98304	393 216	0.00009
0.00299072	249856	999 424	0.00023
0.00302124	679936	2 719 744	0.00063
0.00305176	1298432	5 193 728	0.00121
0.00311279	233472	933 888	0.00022
0.00317383	897024	3 588 096	0.00084
0.00320435	40960	163 840	0.00004
0.00329590	1208320	4 833 280	0.00113
0.00335693	1019904	4 079 616	0.00095
0.00341797	2623232	10 492 928	0.00244
0.00357056	196608	786 432	0.00018
0.00363159	98304	393 216	0.00009
0.00366211	2279936	9 119 744	0.00212
0.00369263	143360	573 440	0.00013
0.00384521	450560	1 802 240	0.00042
0.00390625	5133504	20 534 016	0.00478
0.00396729	245760	983 040	0.00023
0.00402832	755712	3 022 848	0.00070
0.00411987	16384	65 536	0.00002
0.00415039	663552	2 654 208	0.00062
0.00427246	827392	3 309 568	0.00077
0.00439453	3429376	13 717 504	0.00319
0.00440979	36864	147 456	0.00003
0.00457764	49152	196 608	0.00005
0.00466919	221184	884 736	0.00021
0.00469971	354304	1 417 216	0.00033
0.00476074	135168	540 672	0.00013
0.00488281	3444224	13 776 896	0.00321
0.00494385	364544	1 458 176	0.00034
0.00503540	20480	81 920	0.00002
0.00512695	681984	2 727 936	0.00064

Continued on next page

Continued from previous page

probability	nreprs	keys	rel_freq
0.00518799	258048	1 032 192	0.00024
0.00537109	1068544	4 274 176	0.00100
0.00549316	978944	3 915 776	0.00091
0.00555420	147456	589 824	0.00014
0.00570679	86016	344 064	0.00008
0.00585938	2770752	11 083 008	0.00258
0.00598145	135168	540 672	0.00013
0.00604248	323584	1 294 336	0.00030
0.00610352	587776	2 351 104	0.00055
0.00622559	208896	835 584	0.00019
0.00634766	107520	430 080	0.00010
0.00640869	16384	65 536	0.00002
0.00659180	755712	3 022 848	0.00070
0.00671387	364544	1 458 176	0.00034
0.00683594	983040	3 932 160	0.00092
0.00714111	24576	98 304	0.00002
0.00726318	55296	221 184	0.00005
0.00732422	1067520	4 270 080	0.00099
0.00738525	50176	200 704	0.00005
0.00769043	233472	933 888	0.00022
0.00778198	12288	49 152	0.00001
0.00781250	1257232	5 028 928	0.00117
0.00793457	12288	49 152	0.00001
0.00805664	314880	1 259 520	0.00029
0.00823975	36864	147 456	0.00003
0.00830078	112128	448 512	0.00010
0.00854492	313344	1 253 376	0.00029
0.00878906	1023616	4 094 464	0.00095
0.00915527	43008	172 032	0.00004
0.00933838	24576	98 304	0.00002
0.00939941	95232	380 928	0.00009
0.00952148	43008	172 032	0.00004
0.00976562	818368	3 273 472	0.00076
0.00988770	77824	311 296	0.00007
0.01007080	14336	57 344	0.00001
0.01025391	255488	1 021 952	0.00024
0.01037598	30720	122 880	0.00003
0.01074219	162432	649 728	0.00015
0.01098633	227328	909 312	0.00021
0.01171875	607968	2 431 872	0.00057
0.01196289	45312	181 248	0.00004
0.01208496	33792	135 168	0.00003
0.01220703	133632	534 528	0.00012
0.01245117	16896	67 584	0.00002
0.01269531	9216	36 864	0.00001
0.01281738	9216	36 864	0.00001
0.01318359	137216	548 864	0.00013
0.01342773	56832	227 328	0.00005
0.01367188	271744	1 086 976	0.00025

Continued on next page

Continued from previous page

probability	nreprs	keys	rel_freq
0.01373291	1024	4096	0.00000
0.01452637	18432	73 728	0.00002
0.01464844	212736	850 944	0.00020
0.01538086	77824	311 296	0.00007
0.01562500	234656	938 624	0.00022
0.01586914	18432	73 728	0.00002
0.01611328	28672	114 688	0.00003
0.01647949	4096	16 384	0.00000
0.01660156	13440	53 760	0.00001
0.01708984	97280	389 120	0.00009
0.01757812	171776	687 104	0.00016
0.01831055	5120	20 480	0.00000
0.01867676	3072	12 288	0.00000
0.01879883	21504	86 016	0.00002
0.01953125	151904	607 616	0.00014
0.01977539	13312	53 248	0.00001
0.02050781	72064	288 256	0.00007
0.02075195	1536	6144	0.00000
0.02148438	20672	82 688	0.00002
0.02197266	31488	125 952	0.00003
0.02343750	119392	477 568	0.00011
0.02392578	13824	55 296	0.00001
0.02416992	3584	14 336	0.00000
0.02441406	20992	83 968	0.00002
0.02490234	5376	21 504	0.00001
0.02539062	3072	12 288	0.00000
0.02563477	3072	12 288	0.00000
0.02636719	30464	121 856	0.00003
0.02685547	5632	22 528	0.00001
0.02734375	40928	163 712	0.00004
0.02929688	39744	158 976	0.00004
0.03076172	8448	33 792	0.00001
0.03125000	36872	147 488	0.00003
0.03222656	6272	25 088	0.00001
0.03295898	512	2048	0.00000
0.03320312	1152	4608	0.00000
0.03417969	11904	47 616	0.00001
0.03515625	30288	121 152	0.00003
0.03662109	256	1024	0.00000
0.03906250	34240	136 960	0.00003
0.03955078	1024	4096	0.00000
0.04101562	8256	33 024	0.00001
0.04150391	2304	9216	0.00000
0.04296875	1984	7936	0.00000
0.04394531	9600	38 400	0.00001
0.04687500	15736	62 944	0.00001
0.04785156	2304	9216	0.00000
0.04882812	8704	34 816	0.00001
0.05273438	2496	9984	0.00000

Continued on next page

Continued from previous page

probability	nreprs	keys	rel_freq
0.05371094	2688	10 752	0.00000
0.05468750	4736	18 944	0.00000
0.05859375	9312	37 248	0.00001
0.06152344	768	3072	0.00000
0.06250000	6433	25 732	0.00001
0.06640625	384	1536	0.00000
0.06835938	2112	8448	0.00000
0.07031250	3328	13 312	0.00000
0.07324219	384	1536	0.00000
0.07812500	5608	22 432	0.00001
0.07910156	64	256	0.00000
0.08203125	1344	5376	0.00000
0.08593750	448	1792	0.00000
0.08789062	832	3328	0.00000
0.09375000	2332	9328	0.00000
0.09765625	976	3904	0.00000
0.10546875	224	896	0.00000
0.10937500	576	2304	0.00000
0.11718750	704	2816	0.00000
0.12500000	794	3176	0.00000
0.13671875	576	2304	0.00000
0.14062500	372	1488	0.00000
0.15625000	604	2416	0.00000
0.17578125	96	384	0.00000
0.18750000	172	688	0.00000
0.19531250	48	192	0.00000
0.21875000	96	384	0.00000
0.23437500	168	672	0.00000
0.25000000	79	316	0.00000
0.28125000	16	64	0.00000
0.31250000	44	176	0.00000
0.37500000	28	112	0.00000
0.39062500	36	144	0.00000
0.50000000	6	24	0.00000
0.62500000	12	48	0.00000
1.00000000	1	4	0.00000

C Complete GIFT-64 wrong-key distribution

Table 12: All wrong-key probability equivalence classes for GIFT-64 under the 12-round iterative characteristic $\Delta_{in} = \Delta_{out} = 0x0000000000001010$, over all combinations of k and k' . Here **probability** is the class probability p_G , **nreprs** the number of key classes with that probability and **rel_freq** their fraction of the total. The classes listed are those GIFT-64's 32-bit round key can reach, which is why the column sums to 2^{16} and not 2^{32} .

probability	$\log_2 p_G$	nreprs	rel_freq
0.00000000	—	28 058	0.42813
0.00003052	−15.0000	662	0.01010
0.00006104	−14.0000	1327	0.02025
0.00009155	−13.4150	616	0.00940
0.00012207	−13.0000	2848	0.04346
0.00015259	−12.6781	300	0.00458
0.00018311	−12.4150	971	0.01482
0.00021362	−12.1926	184	0.00281
0.00024414	−12.0000	3901	0.05952
0.00027466	−11.8301	162	0.00247
0.00030518	−11.6781	502	0.00766
0.00033569	−11.5406	124	0.00189
0.00036621	−11.4150	1422	0.02170
0.00039673	−11.2996	82	0.00125
0.00042725	−11.1926	226	0.00345
0.00045776	−11.0931	28	0.00043
0.00048828	−11.0000	4596	0.07013
0.00051880	−10.9125	106	0.00162
0.00054932	−10.8301	234	0.00357
0.00057983	−10.7521	56	0.00085
0.00061035	−10.6781	531	0.00810
0.00064087	−10.6077	8	0.00012
0.00067139	−10.5406	104	0.00159
0.00070190	−10.4764	8	0.00012
0.00073242	−10.4150	1424	0.02173
0.00076294	−10.3561	4	0.00006
0.00079346	−10.2996	59	0.00090
0.00085449	−10.1926	286	0.00436
0.00091553	−10.0931	15	0.00023
0.00097656	−10.0000	3991	0.06090
0.00103760	−9.9125	12	0.00018
0.00109863	−9.8301	192	0.00293
0.00115967	−9.7521	4	0.00006
0.00122070	−9.6781	759	0.01158
0.00128174	−9.6077	8	0.00012
0.00134277	−9.5406	140	0.00214
0.00140381	−9.4764	6	0.00009
0.00146484	−9.4150	1021	0.01558
0.00152588	−9.3561	8	0.00012
0.00158691	−9.2996	140	0.00214
0.00170898	−9.1926	284	0.00433
0.00183105	−9.0931	58	0.00089

Continued on next page

Continued from previous page

probability	$\log_2 p_G$	nreprs	rel_freq
0.00195312	-9.0000	2933	0.04475
0.00201416	-8.9556	4	0.00006
0.00207520	-8.9125	48	0.00073
0.00213623	-8.8707	4	0.00006
0.00219727	-8.8301	192	0.00293
0.00244141	-8.6781	404	0.00616
0.00256348	-8.6077	20	0.00031
0.00268555	-8.5406	96	0.00146
0.00280762	-8.4764	8	0.00012
0.00292969	-8.4150	736	0.01123
0.00305176	-8.3561	8	0.00012
0.00317383	-8.2996	60	0.00092
0.00329590	-8.2451	4	0.00006
0.00341797	-8.1926	152	0.00232
0.00366211	-8.0931	52	0.00079
0.00390625	-8.0000	1968	0.03003
0.00402832	-7.9556	4	0.00006
0.00415039	-7.9125	8	0.00012
0.00439453	-7.8301	56	0.00085
0.00488281	-7.6781	372	0.00568
0.00561523	-7.4764	16	0.00024
0.00585938	-7.4150	320	0.00488
0.00634766	-7.2996	48	0.00073
0.00683594	-7.1926	64	0.00098
0.00732422	-7.0931	16	0.00024
0.00781250	-7.0000	584	0.00891
0.00830078	-6.9125	64	0.00098
0.00878906	-6.8301	64	0.00098
0.00976562	-6.6781	272	0.00415
0.01074219	-6.5406	16	0.00024
0.01171875	-6.4150	128	0.00195
0.01269531	-6.2996	16	0.00024
0.01367188	-6.1926	80	0.00122
0.01562500	-6.0000	596	0.00909
0.01660156	-5.9125	32	0.00049
0.02050781	-5.6077	16	0.00024
0.02246094	-5.4764	32	0.00049
0.02343750	-5.4150	80	0.00122
0.03125000	-5.0000	240	0.00366
0.03320312	-4.9125	32	0.00049
0.06250000	-4.0000	96	0.00146
0.07226562	-3.7905	16	0.00024
0.07812500	-3.6781	32	0.00049
0.12500000	-3.0000	32	0.00049
0.26562500	-1.9125	32	0.00049
1.00000000	0.0000	16	0.00024