# Impossible Polytopic Cryptanalysis: A Framework for Block Ciphers with Oversized Keys

Lin Jiao, Xichao Hu, Yongqiang Li, Senpeng Wang, Dengguo Feng

**Abstract**

Block ciphers whose key size exceeds their block size, such as the 192- and 256-bit variants of AES, the tweakable block ciphers Deoxys-BC-256 and Deoxys-BC-384 (standardized by ISO/IEC as Deoxys-TBC), and ChiLow-32 (EUROCRYPT 2025), achieve a security level beyond the block size and are now common. This paper develops a framework for impossible polytopic (IP) cryptanalysis in the oversized-key regime. We formalize the bound-estimation-based IP transitions construction, establish a general framework for distinguisher construction, and introduce two cost-reduction strategies: the large-Sbox strategy and the subtlety-aware strategy. For key recovery, we propose plaintext grouping, which mounts key recovery on both sides of an IP transition, and partition-guess-filter, which allows the entire master key to be guessed in principle. Together they yield the first key-recovery framework for IP attacks, available in a target-table and in a candidate-table variant; with it, our IP distinguishers and attacks reach more rounds and IP attacks extend to the known-plaintext (KP) setting. We apply the framework to AES, Deoxys-BC, Joltik-BC, LED-128, and ChiLow-32 in the single-key setting, under both chosen-plaintext/chosen-ciphertext (CP/CC) and KP settings. Our main objective is low-data cryptanalysis. We obtain 10-round low-data attacks on Deoxys-BC-384 and Joltik-BC-192, improving the best known impossible differential attacks by one round; for AES-192/256, the proposed IP attack achieves the lowest data complexity among all known single-key attacks with the same number of rounds; for ChiLow-32, a full-round CP attack requires only 8 chosen plaintexts, and a full-round KP attack only 107 known plaintexts, improving upon the best known full-round attacks requiring 64 and 160 plaintexts, respectively.

**Index Terms**

Impossible polytopic attack, Impossible polytopic distinguisher, Low-data cryptanalysis, Block cipher

## I. Introduction

BLOCK ciphers with a key size larger than their block size are a prevalent design paradigm. Prominent examples include two variants of AES [1], namely AES-192 and AES-256; two instances of Deoxys-BC [2]—the underlying tweakable block ciphers of Deoxys-II, a portfolio-selected scheme of the CAESAR competition—namely Deoxys-BC-256 and Deoxys-BC-384; and ChiLow [3], proposed at EUROCRYPT 2025 and designed for code encryption. These designs achieve a security level exceeding their block size by enlarging the key size. The paradigm is further endorsed by standardization efforts: the submission requirements for block ciphers recently issued by the Institute of Commercial Cryptography Standards, China [4] mandate the support of the 256-256 and 256-512 block-key bit-length pairs and allow the optional support of 256-384 and 128-256 pairs. Such ciphers are now widespread, so the question is how the extra degrees of freedom of an oversized key can be turned into low-data cryptanalysis.

The impossible polytopic (IP) attack [5] was proposed by Tyge Tiessen at EUROCRYPT 2016. It achieves prominent improvements for low-data attacks. Unlike the impossible differential (ID) attack [6] that declares a single input-output difference pair impossible (a one-dimensional point constraint), the IP attack aggregates $d$ differences into a $d$-difference tuple and captures incompatibility across the entire tuple. This structural distinction implies that the IP attack filters key candidates via a high-dimensional set intersection rather than a one-dimensional point constraint.

Like ID attacks, the IP attack proceeds in two stages: constructing a distinguisher (an impossible polytopic (IP) transition) and extending it over additional rounds to recover the key. A distinguisher has two uses in modern block cipher cryptanalysis: (i) as the core enabler of key-recovery attacks, and (ii) as a measure of the statistical discrepancy between the primitive and a random permutation. The latter purpose has been extensively explored in AES-related works, including the multiple-of-$n$ property [7], the yoyo trick [8], and the exchange attack [9]. The construction of distinguishers therefore deserves separate treatment.

To date, there exist two primary methodologies for constructing IP transitions.

- **Tiessen's method.** This manual bound-estimation-based technique propagates input $d$-differences round by round to enumerate all feasible output $d$-differences; given the full set of reachable output $d$-differences, one deduces the impossible ones.

Xichao Hu is with Zhongguancun Laboratory, Beijing, China
Lin Jiao and Dengguo Feng are with State Key Laboratory of Cryptology, Beijing, China
Yongqiang Li is with State Key Laboratory of Cyberspace Security Defense, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China, and is also with School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
Senpeng Wang is with PLA Information Engineering University, Zhengzhou, China

- **State-based method ([10, 11]).**This SAT/SMT-based automatic approach incorporates the full cipher specifications and has yielded novel distinguishers for lightweight block ciphers.

Key-recovery techniques constitute another critical factor determining the overall attack performance. Tyge Tiessen laid the groundwork and proposed low-data attacks against AES and DES, demonstrating the advantages of the IP attack.

***Motivation.*** Block ciphers with a key size strictly larger than their block size (e.g., AES-256, Deoxys-BC-384, and ChiLow-32) provide extra degrees of freedom for adversaries through oversized tweakeys. The IP attack is well placed to exploit this structure through high-dimensional filtering, but it has not been developed for it, and four obstacles stand in the way:

(i) *Prohibitive construction cost.* Unlike state-based methods applicable to lightweight ciphers with small $d$, Tiessen's method offers greater generality but incurs exponential complexity. Enumerating reachable output $d$-differences in high-security ciphers is computationally expensive, necessitating rigorous formalization and optimized strategies.

(ii) *One-sided limitation.* Existing key-recovery techniques can only extend attacks to extra cipher rounds on one side of the distinguisher: prepending even a few rounds upfront requires enumerating massive backward-propagated d-difference candidates and incurs prohibitive data complexity, which prevents full exploitation of the oversized key schedule's extra degrees of freedom to cover more attack rounds.

(iii) *Prohibitive guessing complexity.* The oversized key implies that the involved key bits can quickly approach the master key size. Existing one-time full-key guessing strategies thus lead to unmanageable attack complexity.

(iv) *The absence of low-data KP cryptanalysis.* Existing differential-type distinguishers (including ID and IP) are intrinsically tied to the CP model. The two existing KP attack families, MITM and linear attacks, fail to fill this gap: MITM attacks rest on the $\delta$-set distinguisher built under the CP or CC setting [12] and therefore do not transfer, whereas linear distinguishers incur prohibitively high data costs (e.g., the 3-round ChiLow-32 distinguisher already requires $2^{28}$ texts as shown in Table II). No low-data KP distinguisher is known; this is where the present work enters.

***Our Contributions.*** Tiessen's original work [5] already observed that a $d$-difference tuple can capture incompatibilities that no single difference pair reveals, and that this translates into additional filtering power. What this paper contributes is not the observation itself but its operational form for oversized-key ciphers: each of the four obstacles above is met by a technique of its own. We summarize our contributions as follows:

- **Distinguisher construction.** We formalize the *bound-estimation-based IP transition* and give a general construction framework, within which we introduce two cost-reduction strategies, the *large-Sbox strategy* and the *subtlety-aware strategy*, to efficiently enumerate reachable output $d$-differences. The former treats a series unkeyed operations as a large Sbox, while the latter performs a refined analysis of the nonlinear functions for more exact $d$-difference propagation.
- **Two-sided key recovery.** We propose the *plaintext-grouping technique*. By building a plaintext-grouping table from the keys guessed in the pre-distinguisher rounds, it gives key recovery on both sides of the distinguisher and hence attacks on more rounds.
- **Full-key guessing.** We propose the *partition-guess-filter technique*, which partitions the output $d$-difference positions into groups and performs stepwise key guessing and filtering. This technique allows $|K_{\text{in}} \cup K_{\text{out}}|$ to reach the full master key size $|K|$ in principle.
- **Low-data KP cryptanalysis.** In the KP setting, the adversary need not choose plaintexts to realize a specific input. Instead, the $d$-difference is naturally carried by the known plaintexts, since the input $d$-difference could cover the full state. The IP transition can still be derived online because we only target a narrow window of output positions rather than the full state.

Combining these techniques yields the first key-recovery framework for IP attacks, in target-table and candidate-table variants. Applying our framework to AES, Deoxys-BC, Joltik-BC [22], LED-128 [23], and ChiLow-32 in the single-key setting for lower data cost, we obtain significantly improved results:

- For ChiLow-32, we mount full-round attacks in both settings: a CP attack requiring only 8 chosen plaintexts (improving upon 64 [21]), and a KP attack requiring only 107 known plaintexts (improving upon 160 [21]).
- For the 7-round attack on AES-192 and the 8-round attack on AES-256, our work achieves the lowest data complexity among all known attacks at these round counts, reaching $2^{16}$ and $2^{24}$ respectively.
- For Deoxys-BC-384 and Joltik-BC-192, we construct 10-round attacks from 5-round IP transitions, extending the best known ID attacks by one round in the CP/CC setting. This gain is not an artifact of parameter tuning; it originates from the high-dimensional filtering of the $d$-difference tuple, which structurally bypasses the four-round bottleneck restricting the corresponding ID distinguishers. To the best of our knowledge, it first demonstrates that IP attacks achieve a higher round count than ID attacks among all known results.
- To the best of our knowledge, our method yields the first low-data distinguishers in the KP setting.

TABLE I: Our distinguishers and attacks in the chosen-plaintext/chosen-ciphertext (CP/CC) setting. Our distinguishers can distinguish $\#R+0.5$ rounds of the cipher from a random permutation. ID: impossible differential; IP: impossible polytopic transition; MITM: meet-in-the-middle; LC: low-complexity; LD: low-data; †: obtained by implementing and running the one-step method proposed in [13] (the best impossible differential attack obtained for comparison); $\mathsf{IP}^{\mathrm{T}}$/$\mathsf{IP}^{\mathrm{C}}$: the target-table/candidate-table variant of the IP attack.

| Cipher | #R | Data | Time | Mem. | Method | Type | Ref |
|---|---|---|---|---|---|---|---|
| | | | Distinguishers | | | | |
| AES/AES-like | 4 | - CP | - | - | ID | - | [14, 15] |
| Deoxys-BC-384 | **5.5** | 44 CP | $2^{337.30}$ | $2^{339.98}$ | IP | LC | Section IV-A |
| | **5.5** | 13 CP | $2^{361.35}$ | $2^{364.09}$ | IP | LD | Section IV-A |
| Joltik-BC-192 | **5.5** | 16 CP | $2^{177.32}$ | $2^{180.05}$ | IP | LC | Section IV-C |
| | **5.5** | 13 CP | $2^{181.34}$ | $2^{184.09}$ | IP | LD | Section IV-C |
| LED-128 | **7.5** | 15 CP | $2^{109.21}$ | $2^{111.35}$ | IP | LC | Section IV-G |
| | **7.5** | 9 CP | $2^{117.25}$ | $2^{119.47}$ | IP | LD | Section IV-G |
| AES-256 / | **4.5** | 28 CP | $2^{209.23}$ | $2^{211.29}$ | IP | LC | Section IV-B |
| Deoxys-BC-256 | **4.5** | 9 CP | $2^{233.29}$ | $2^{235.47}$ | IP | LD | Section IV-B |
| Joltik-BC-128 | **4.5** | 15 CP | $2^{109.21}$ | $2^{111.34}$ | IP | LC | Section IV-D |
| | **4.5** | 9 CP | $2^{117.25}$ | $2^{119.47}$ | IP | LD | Section IV-D |
| ChiLow-32 | 3 | $2^{19}$ CP | $2^{19}$ | - | Differential | - | [3] |
| | 3.5 | $2^{8}$ CP | - | - | Integral | - | [16] |
| | **4.5** | 11 CC | $2^{118.64}$ | $2^{121.48}$ | IP | - | Section IV-H |
| | **4** | 5 CP | $2^{112.79}$ | $2^{115.83}$ | IP | - | Section IV-H |
| | | | Attacks | | | | |
| Deoxys-BC-384 | 9 | $2^{124}$ CP | $2^{374.83}$ | $2^{245}$ | ID† | - | Appendix A |
| | **10** | $2^{24}$ CP | $2^{323.80}$ | $2^{372.15}$ | $\mathsf{IP}^{\mathrm{T}}$ | LD | Section IV-A |
| | **10** | $2^{24}$ CP | $2^{369.17}$ | $2^{324.17}$ | $\mathsf{IP}^{\mathrm{C}}$ | LD | Section IV-A |
| | **10** | $2^{32}$ CP | $2^{331.73}$ | $2^{331.98}$ | $\mathsf{IP}^{\mathrm{T}}$ | LC | Section IV-A |
| Deoxys-BC-256 | 8 | $2^{117}$ CP | $2^{248}$ | $2^{182}$ | ID† | - | Appendix B |
| | **8** | $2^{24}$ CP | $2^{194.93}$ | $2^{243.56}$ | $\mathsf{IP}^{\mathrm{T}}$ | LD | Section IV-B |
| | **8** | $2^{24}$ CP | $2^{241.04}$ | $2^{195.47}$ | $\mathsf{IP}^{\mathrm{C}}$ | LD | Section IV-B |
| | **8** | $2^{32}$ CP | $2^{202.89}$ | $2^{203.09}$ | $\mathsf{IP}^{\mathrm{T}}$ | LC | Section IV-B |
| Joltik-BC-192 | 9 | $2^{64}$ CP | $2^{186.83}$ | $2^{125}$ | ID† | - | Appendix A |
| | **10** | $2^{16}$ CP | $2^{167.80}$ | $2^{188.15}$ | $\mathsf{IP}^{\mathrm{T}}$ | - | Section IV-C |
| | **10** | $2^{16}$ CP | $2^{185.19}$ | $2^{168.21}$ | $\mathsf{IP}^{\mathrm{C}}$ | - | Section IV-C |
| Joltik-BC-128 | 8 | $2^{60}$ CP | $2^{123}$ | $2^{125}$ | ID† | - | Appendix B |
| | **8** | $2^{16}$ CP | $2^{102.93}$ | $2^{123.56}$ | $\mathsf{IP}^{\mathrm{T}}$ | - | Section IV-D |
| | **8** | $2^{16}$ CP | $2^{121.07}$ | $2^{103.52}$ | $\mathsf{IP}^{\mathrm{C}}$ | - | Section IV-D |
| AES-256 | 7 | $2^{26}$ CP | $2^{146}$ | $2^{40}$ | MITM | - | [17] |
| | 8 | $2^{107}$ CP | $2^{196}$ | $2^{96}$ | MITM | - | [18] |
| | 8 | $2^{113}$ CP | $2^{196}$ | $2^{129}$ | ID | - | [19] |
| | **8** | $2^{24}$ CP | $2^{186.95}$ | $2^{243.57}$ | $\mathsf{IP}^{\mathrm{T}}$ | LD | Section IV-E |
| | **8** | $2^{24}$ CP | $2^{241.04}$ | $2^{187.42}$ | $\mathsf{IP}^{\mathrm{C}}$ | LD | Section IV-E |
| | **8** | $2^{32}$ CP | $2^{194.92}$ | $2^{203.30}$ | $\mathsf{IP}^{\mathrm{T}}$ | LC | Section IV-E |
| | **8** | $2^{32}$ CP | $2^{201.06}$ | $2^{195.47}$ | $\mathsf{IP}^{\mathrm{C}}$ | LC | Section IV-E |
| AES-192 | 7 | $2^{23.25}$ CP | $2^{137.36}$ | $2^{112}$ | Mixture-differential | - | [20] |
| | 7 | $2^{26}$ CP | $2^{146.3}$ | $2^{40}$ | Multiple-of-8 | - | [17] |
| | 7 | $2^{103}$ CP | $2^{103}$ | $2^{129}$ | ID | - | [19] |
| | **7** | $2^{16}$ CP | $2^{162.35}$ | $2^{154.90}$ | $\mathsf{IP}^{\mathrm{T}}$ | - | Section IV-F |
| ChiLow-32 | 6 | $2^{6.32}$ CP | $2^{76.55}$ | - | Integral | - | [16] |
| | 7 | $2^{6.32}$ CP | $2^{108.55}$ | - | Integral | - | [16] |
| | Full-round | 64 CC | $2^{102.09}$ | $2^{94.56}$ | MITM | - | [21] |
| | **Full-round** | 8 CP | $2^{120.35}$ | $2^{121.35}$ | IP | - | Section IV-H |

All new results are summarized in Tables I and II[1]. Throughout, all complexity figures are accounted for in the single unit-cost model fixed in Section II-A, so that the reported results are comparable across ciphers and across attack families.

[1]For several of the ciphers considered, other techniques—e.g., meet-in-the-middle attacks on 9-round AES-192/256—reach one or two more rounds than our attacks; at the same number of rounds, however, our data complexity is the lowest among all known attacks, which is precisely the objective of this work.

TABLE II: Our distinguishers and attacks in the KP setting (LC: low-complexity, LD: low-data. Our distinguishers can distinguish $\#R + 0.5$ rounds of the cipher from a random permutation.)

| Cipher | #R | Data | Time | Mem. | Method | Type | Ref |
|---|---|---|---|---|---|---|---|
| | | | Distinguishers | | | | |
| ChiLow-32 | 3 | $2^{28}$ | $2^{28}$ | - | Linear | - | [3] |
| | 3 | 5 | $2^{97.87}$ | $2^{100.25}$ | IP | - | Section V-A |
| Deoxys-BC-384 | 3.5 | 39 | $2^{297.38}$ | $2^{299.80}$ | IP | LC | Section V-B |
| | 3.5 | 9 | $2^{369.50}$ | $2^{372.12}$ | IP | LD | Section V-B |
| AES-256 and Deoxys-BC-256 | 2.5 | 23 | $2^{169.31}$ | $2^{170.99}$ | IP | LC | Section V-C |
| | 2.5 | 7 | $2^{241.52}$ | $2^{243.54}$ | IP | LD | Section V-C |
| AES-192 | 2.5 | 23 | $2^{169.31}$ | $2^{170.99}$ | IP | LC | Section V-C |
| | 2.5 | 9 | $2^{185.37}$ | $2^{187.12}$ | IP | LD | Section V-C |
| | | | Attacks | | | | |
| ChiLow-32 | Full-round | 160 | $2^{120.34}$ | $2^{98.32}$ | MITM | - | [21] |
| | **Full-round** | 107 | $2^{126.08}$ | $2^{101.32}$ | IP | - | Section V-A |

**Implications for design margins.** The results reveal a sharp contrast in design margins. For AES, Deoxys-BC, and Joltik-BC, our attacks do not break the full versions; their significance lies in penetrating one more round or achieving lower data cost than the best known ID attacks (see Tables I and II). This suggests that a key schedule rapidly randomizing single differences may still leave a zero-probability $d$-difference transition over an extra round. Conversely, for ChiLow-32, the full-round penetration with minimal data indicates that its nested tweakey schedule and local $\chi$ layer diffuse $d$-differences too slowly. For highly optimized, low-latency primitives, the design margin against low-data $d$-difference cryptanalysis is remarkably thin. We therefore recommend incorporating IP-style $d$-difference analysis as a standard benchmark for both high-security and lightweight oversized-key ciphers.

**An information-theoretic view.** The impossibility behind an IP transition concerns *support* rather than small probability: for every key the cipher's output $d$-difference law $P_E$ is supported on a set $\Gamma$ of size at most $2^t$, whereas a random permutation induces the uniform law $P_R$ on the whole $2^{d\eta}$-point space. The attack is therefore a one-sided hypothesis test between these two laws. The first quantity that governs it is the mass the test may spend: $P_R$ places $|\Gamma|/2^{d\eta} \le 2^{t-d\eta}$ on the support of $P_E$ and $1 - |\Gamma|/2^{d\eta}$ on its complement, where $P_E$ places none, so

$$\Delta(P_E, P_R) = \tfrac{1}{2}\sum_x \big|P_E(x) - P_R(x)\big| = 1 - \frac{|\Gamma|}{2^{d\eta}} \;\ge\; 1 - 2^{t-d\eta},$$

the equality holding exactly because the two laws share no mass outside $\Gamma$.

**Outline.** Section II covers the preliminaries and related work. The new techniques for distinguisher construction and key-recovery attacks are presented in Section III. In Section IV and Section V, we apply our method to the CP/CC setting and the KP setting, respectively. Finally, Section VI concludes this paper.

## II. Preliminaries

### A. Notations and Conventions

**Definition 1** ($(d+1)$-polytope and $d$-difference [5])**.** *A $(d+1)$-polytope in $\mathbb{F}_2^n$ is a $(d+1)$-tuple $\boldsymbol{X} = (X_0, X_1, \ldots, X_d) \in (\mathbb{F}_2^n)^{d+1}$. The associated $d$-difference is $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d) = (X_0 \oplus X_1, \ldots, X_0 \oplus X_d)$, where $X_0$ is the reference text.*

The notations used throughout this paper are as follows:

- The superscript $r$ denotes the round; for $X^r_{\ell,i}$ or $\alpha^r_{\ell,i}$, the first subscript $\ell$ identifies the component of $\boldsymbol{X}^r$ or $\boldsymbol{\alpha}^r$, and the second subscript $i$ specifies the cell (bit, nibble, or byte depending on the context) position within that component.
- Let $\boldsymbol{X}^r_{*,i} = (X^r_{0,i}, \ldots, X^r_{d,i})$; similarly, let $\boldsymbol{\alpha}^r_{*,i}$ be defined analogously. Hereafter, both the target cell and the active cell refer to a cell of the form $\boldsymbol{\alpha}^r_{*,i}$, and the two notions are position-consistent across all components. A target cell denotes the distinguisher's output cell of primary interest in our analysis. An intermediate target cell is a cell at an intermediate round whose $d$-difference propagates forward to affect the target cell at the output.
- For AES-like ciphers, when the AddRoundKey operation and the MixColumns operation are commuted, the equivalent round key is denoted by $K'^r$. The output of the equivalent AddRoundKey operation is denoted by $\boldsymbol{X}'^r$ or $\boldsymbol{\alpha}'^r$.

**Definition 2** (Polytopic transition [5])**.** *For a function $f : \mathbb{F}_2^n \to \mathbb{F}_2^m$, the $(d+1)$-polytopic transition $\boldsymbol{\alpha} \xrightarrow{f} \boldsymbol{\beta}$ means that $f(x \oplus \alpha_1) \oplus f(x) = \beta_1, f(x \oplus \alpha_2) \oplus f(x) = \beta_2, \ldots, f(x \oplus \alpha_d) \oplus f(x) = \beta_d$. The probability of $\boldsymbol{\alpha} \xrightarrow{f} \boldsymbol{\beta}$ is taken over a uniformly random $x \in \mathbb{F}_2^n$.*

Hence, an impossible $(d+1)$-polytopic transition is a $(d+1)$-polytopic transition that occurs with probability zero.

**Lemma 1** ([5]). *Let $S : \mathbb{F}_2^s \to \mathbb{F}_2^s$ be an Sbox. For a given input $d$-difference $\boldsymbol{\alpha}$, the number of output $d$-differences to which $\boldsymbol{\alpha}$ is mapped with non-zero probability is upper bounded by $2^s$, the cardinality of the reference text space.*

**Impossible Polytopic Attack [5].** The impossible $(d+1)$-polytopic transition, which occurs with probability zero, can be used to filter out wrong round key guesses [5]. Let $n$ be the block size of the cipher and let $\kappa$ be the number of bits in the last round key.

1) Choose a $d$-difference such that the ratio of possible $(d+1)$-polytopic transitions is lower than $2^{-\kappa}$.
2) Get the $r$-round encryption of $d+1$ plaintexts adhering to the input $d$-difference.
3) For each guess of the round key $k_r$, decrypt the last round. If the obtained $d$-difference after the $(r-1)$-th round is possible, keep the key as a candidate; otherwise, discard it.

*Complexity*: the data complexity is low and limited to $(d+1)$ chosen plaintexts; the time complexity depends on that of determining whether an obtained $(r-1)$-round $(d+1)$-difference is possible or not. Note that $d$ cannot exceed the number of degrees of freedom of the input active cells.

The color convention adopted in the figures of this paper is as follows.

- □ : Nonactive cell.
- ■ : Known active cell.
- ■ : Unknown active cell.
- □ : (Intermediate) target cell. For clarity, an additional ✓ is attached to the target cell.
- □ : Key recovery required cell.

**Cost units.** All complexities in this paper are stated in a single unit-cost model, which we fix here once and use without exception, so that figures are comparable across attack families and across ciphers. Time is counted in units of one full $R$-round encryption of the cipher under analysis, $R$ being the round count of that cipher; memory is counted in $n$-bit blocks, $n$ being its block size; data is counted in texts. Partial operations,a partial decryption over a subset of the rounds, a single substitution-layer evaluation, or a hash-table probe, are charged by the ratio they bear to one full encryption, and these ratios are reported explicitly wherever they occur (the factors $\phi$ of Sections III-C to III-E). Because the unit refers to the cipher under analysis rather than to an absolute operation count, a figure $2^c$ is comparable across attack families on the same cipher, and across ciphers only together with the round counts and block sizes recorded in Tables I and II.

### *B. Specification of Applications*

#### *(1) AES*

AES [1] is among the most prominent block ciphers to date, whose design principles have exerted a lasting influence on the development of block cipher constructions. Adopted as FIPS-197, AES operates on 128-bit blocks and supports key lengths of 128, 192, and 256 bits, which translate into 10, 12, and 14 rounds, respectively. Following the SPN structure, AES adopts an 8-bit Sbox and an MDS matrix. Its internal state can be represented as a square array of bytes as follows, where $s_i \in \mathbb{F}_2^8$ for $0 \le i \le 15$.

$$S = \begin{pmatrix} s_0 & s_4 & s_8 & s_{12} \\ s_1 & s_5 & s_9 & s_{13} \\ s_2 & s_6 & s_{10} & s_{14} \\ s_3 & s_7 & s_{11} & s_{15} \end{pmatrix}. \tag{1}$$

One encryption round of AES is depicted in Figure 1, and it consists of the following four operations:

- AddRoundKey (AK): The round key derived from the key schedule is XORed with the internal state.
- SubBytes (SB): The 8-bit Sbox is applied to each byte of the cipher's internal state in parallel.
- ShiftRows (SR): The $i$-th row, where $0 \le i \le 3$, of the internal state is rotated right by $i$ bytes.
- MixColumns (MC): Each column of the internal state is multiplied by the MDS matrix.

Prior to the first round an initial AddRoundKey is applied, and the final round omits the MixColumns operation.

The key schedules of AES-192 and AES-256 are shown in Figure 2 and Figure 3, respectively. The master key is split into 6 and 8 32-bit words for AES-192 and AES-256, respectively, where each word consists of four bytes. These words are subsequently processed to produce the round keys. The function $g$ rotates an input word left by one byte, followed by four parallel Sbox lookups and the XOR of the round constant $Rcon$. The function $h$ performs four parallel Sboxes on its input word; for AES-256 it is additionally applied whenever the word index satisfies $i \bmod 8 = 4$.

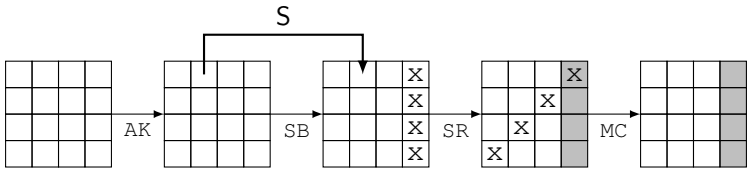


Fig. 1: One round of block cipher AES

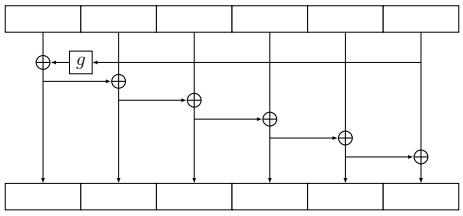


Fig. 2: The key schedule of AES-192

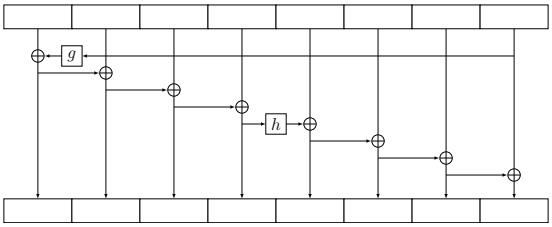


Fig. 3: The key schedule of AES-256

*(2) Deoxys-BC*

Deoxys-BC is an AES-based tweakable block cipher (TBC) within the TWEAKEY framework, designed by Jean, Nikolić, and Peyrin [24]. It is the underlying primitive of the Deoxys authenticated encryption family; Deoxys-II was selected in the final portfolio of the CAESAR competition [2]. Deoxys-BC operates on a 128-bit internal state and combines the secret key and public tweak into a unified input referred to as the *tweakey*. The cipher employs the standard AES round functions, namely AddRoundTweakey (ART), SB, SR, and MC; the subtweakeys are generated by the Super TWEAKEY (STK) schedule.

Deoxys-BC-384 (resp. Deoxys-BC-256) is an instance of Deoxys-BC with a 384-bit (resp. 256-bit) tweakey; the tweakey material $KT = K\|T$ is divided into 128-bit words $TK_0, TK_1, TK_2$ (resp. $TK_0, TK_1$). Let $STK^r$ be the round subtweakey. As Figure 4 shows, the $i$-th cell of the subtweakey after $r$ rounds is given by

$$STK_i^r = \begin{cases} TK_{0,h^r[i]} \oplus LFSR_2^r\big(TK_{1,h^r[i]}\big) \oplus LFSR_3^r\big(TK_{2,h^r[i]}\big) \oplus RC_i^r, & \text{for Deoxys-BC-384,} \\ TK_{0,h^r[i]} \oplus LFSR_2^r\big(TK_{1,h^r[i]}\big) \oplus RC_i^r, & \text{for Deoxys-BC-256,} \end{cases}$$

where $RC^r$ denotes the round constant. Here, $h$ is a byte permutation defined as

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 1 & 6 & 11 & 12 & 5 & 10 & 15 & 0 & 9 & 14 & 3 & 4 & 13 & 2 & 7 & 8 \end{pmatrix},$$

and $LFSR_2$ and $LFSR_3$ are byte-oriented LFSR functions; the superscript in $h^r$, $LFSR_i^r$ denotes applying the corresponding operation $r$ times. Accordingly, observing $STK^r_{h^r[i]}$ over three consecutive rounds yields three independent linear equations in the unknowns $TK_{0,i}$, $TK_{1,i}$, and $TK_{2,i}$ for Deoxys-BC-384, and over two consecutive rounds yields two independent linear equations in the unknowns $TK_{0,i}$ and $TK_{1,i}$ for Deoxys-BC-256; in both cases, the corresponding master tweakey cells are uniquely determined. Deoxys-BC-256 iterates 14 rounds, while Deoxys-BC-384 iterates 16 rounds.

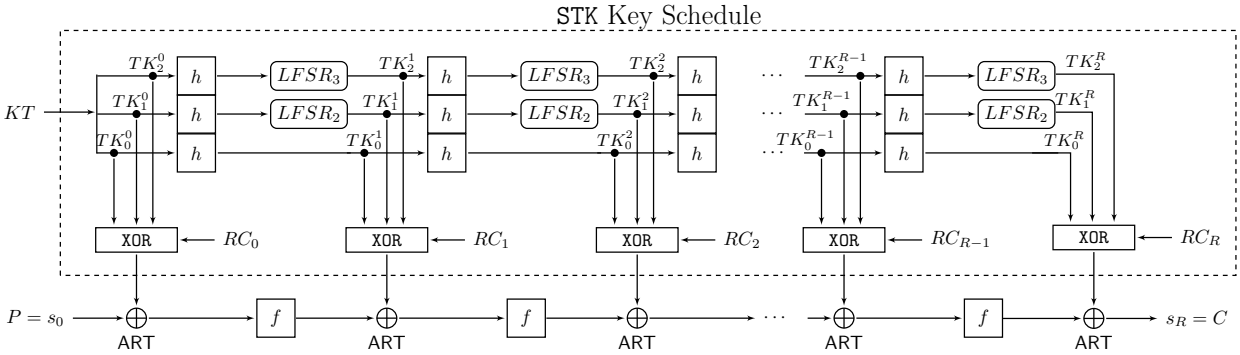


Fig. 4: The TWEAKEY framework for Deoxys-BC-384

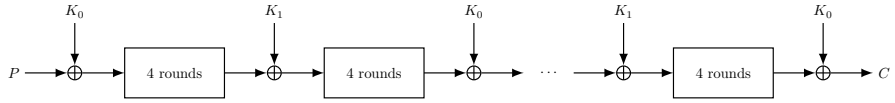


Fig. 5: Overview of LED-128

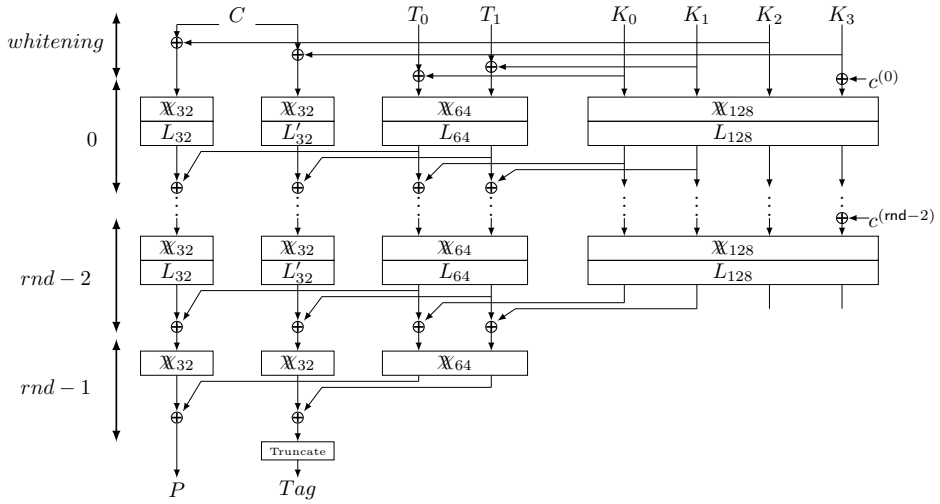


Fig. 6: Full ChiLow-32 decryption and tag computation ($rnd = 8$).

*(3) Joltik-BC*

Joltik-BC is the resource-constrained counterpart of Deoxys-BC designed by the same team [22], with all parameters scaled down to a 64-bit block size on 4-bit nibbles, while preserving the TWEAKEY paradigm. It was submitted as a core primitive of Joltik for the CAESAR Competition's second round.

Joltik-BC also supports two tweakey sizes, i.e., Joltik-BC-128 and Joltik-BC-192, which iterate 24 and 32 encryption rounds, respectively. Joltik-BC utilizes a 4-bit Sbox and an involutory MDS matrix for state updating, as well as 4-bit parallel LFSRs and a dedicated nibble permutation $h$ defined as

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 9 & 15 & 8 & 13 & 10 & 14 & 12 & 11 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{pmatrix},$$

sharing the same core design philosophy as Deoxys-BC.

*(4) LED-128*

LED-128 is the high-security variant of the LED lightweight block cipher family, proposed by Guo et al. at CHES 2011 [23]. It is an AES-like block cipher with 64-bit block size and 128-bit key length. The 64-bit block is arranged into a $4 \times 4$ nibble matrix. One round consists of AddConstants (AC), SubCells (SC), ShiftRows SR, and MixColumns MC, adopting PRESENT's 4-bit Sbox and an MDS optimized for serial hardware. As Figure 5 shows, there is no dedicated key expansion. The 128-bit original key is reused periodically to participate in subkey addition, which is performed only once every four rounds (defined as a *step*). Specifically, LED-128 iterates over 12 steps, i.e., 48 rounds in total.

*(5) ChiLow-32*

ChiLow-32 is one of the main variants in the ChiLow family proposed by Belkheyar et al. at EUROCRYPT 2025 [3]. It is designed for authenticated code encryption (ACE) on embedded microcontrollers, where instructions stored in untrusted external memory must be decrypted with ultra-low latency before execution.

ChiLow-32 combines a 32-bit tweakable block cipher $D_{32}$ with a companion pseudorandom function (PRF) $D'_{32}$ for authentication, which outputs a 32-bit plaintext block $P$ together with a $\tau$-bit authentication tag ($\tau \in \{8, 16\}$). ChiLow-32 takes a 128-bit master key $K = (K_0 \| K_1 \| K_2 \| K_3) \in \mathbb{F}_2^{128}$ and a 64-bit tweak $T = (T_0 \| T_1) \in \mathbb{F}_2^{64}$. It uses an SPN structure, and its encryption consists of $rnd = 8$ rounds. Figure 6 outlines the complete decryption and tag computation procedure for ChiLow-32. The operations in Figure 6 are detailed as follows.

- Non-linear layer: $\chi\!\!\chi_n (n = 32, 64, 128)$ is extended from the classic $\chi$ function[2]. Let $m = n/2$. For any input $x \in \mathbb{F}_2^n$, the output is computed as:

$$\chi\!\!\chi_n(x) = (\chi_{m-1}(x_0, x_1, \ldots, x_{m-2}) \| \chi_{m+1}(x_{m-1}, x_m, \ldots, x_{n-1})) \oplus \lambda(x).$$

  Here $\lambda(x) \in \mathbb{F}_2^n$ is defined by:

$$\lambda_i(x) = \begin{cases} x_m \oplus x_{m-3}, & i = m-3, \\ x_{m-1} \oplus x_{m-2}, & i = m-2, \\ x_{m-3} \oplus x_m \oplus x_{m-1}, & i = m-1, \\ x_m \oplus x_{m-2}, & i = m, \\ 0, & \text{otherwise.} \end{cases}$$

- Linear layer: $L_n$ $(n = 32, 64, 128)$ and $L'_{32}$ adopt the same construction. For input $x$ and output $y$, each output bit satisfies: $y_i = x_{a\cdot i+b_0} \oplus x_{a\cdot i+b_1} \oplus x_{a\cdot i+b_2}$ $(0 \le i \le n-1)$. The offsets are summarized as follows:

| Layer | $a$ | $b_0$ | $b_1$ | $b_2$ |
|---|---|---|---|---|
| $L_{32}$ | 11 | 5 | 9 | 12 |
| $L'_{32}$ | 11 | 1 | 26 | 30 |
| $L_{64}$ | 3 | 1 | 26 | 50 |
| $L_{128}$ | 17 | 7 | 11 | 14 |

ChiLow employs a nested tweakey schedule, where the round key is partially injected into the tweak, and the round tweak is further added to the data path. Let $K^r = \big(K_0^r \| K_1^r \| K_2^r \| K_3^r\big) \in \mathbb{F}_2^{128}$ and $T^r = \big(T_0^r \| T_1^r\big) \in \mathbb{F}_2^{64}$ denote the round key and round tweak for round $r$, respectively. The initial tweak state is set as $T^0 = T \oplus (K_0 \| K_1)$, and the initial key state is $K^0 = K$. For all $0 \le i < rnd - 1$:

$$K^{i+1} = L_{128}\big(\chi\!\!\chi_{128}\big(K^i \oplus c^i\big)\big), \quad T^{i+1} = L_{64}\big(\chi\!\!\chi_{64}\big(T^i\big)\big) \oplus K^{i+1}_{0\cdots 63}.$$

Here $c^i$ denotes the round constant, which is only applied to bits 96 to 127 of $K^i$. For the final round, the tweak state is updated only by linear mapping: $T^{rnd} = L_{64}(T^{rnd-1})$.

### C. Storage and Membership Queries for Static Sets

The Fredman–Komlós–Szemerédi (FKS) two-level perfect hashing scheme [25] provides a static dictionary for a fixed key set $\Gamma$ of size $N = |\Gamma|$. It achieves $\mathcal{O}(1)$ worst-case lookup time and $\mathcal{O}(N)$ worst-case space, with $\mathcal{O}(N)$ expected construction time. The original formulation further sharpens the pointer overhead space bound to $o(N)$.

The structure employs two levels of 2-universal hash functions. Firstly, a primary hash $h$ partitions the elements into buckets $B_i$ of sizes $c_i = |B_i|$ and $h$ is resampled until $\sum_i c_i^2 \le \rho N$. Subsequently, each bucket $B_i$ is assigned a secondary table of size $\mu c_i^2$ and an independent small hash $h_i$, parameterized to ensure injectivity on $B_i$. The descriptors and offsets are maintained in a first-level directory $T_1$, while all secondary slots reside in a global table $T_2$.

A membership query for an element $x$ evaluates the primary index $i = h(x)$, retrieves the corresponding descriptor from $T_1[i]$, evaluates the secondary index $k = h_i(x)$, and validates the candidate pointer from $T_2[i][k]$. This process entails two hash function evaluations (one first-level hash evaluation plus one negligible second-level evaluation) and one key comparison, guaranteeing worst-case $\mathcal{O}(1)$ lookup time independent of $N$. Construction is expected $\mathcal{O}(N)$ due to the constant success probability at both sampling stages, alongside the bounded sum of squared bucket sizes.

A canonical parameterization sets $\rho = 4$ and allocates secondary tables of size $c_i^2$. Under this configuration, the expected number of primary hash trials is bounded by 2, yielding approximately $\mathcal{TC}_T = 2N$ evaluations. The aggregate secondary slots total at most $4N$, which, combined with the $N$-slot primary directory, results in a pointer structure of at most $\mathcal{MC}_T = 5N$ slots. The 2-universal hash functions used in FKS (e.g., $h(x) = ((ax + b) \bmod p) \bmod M$) are lightweight, entailing merely a handful of basic arithmetic operations (multiplication, addition, and modulo). We therefore take the cost of one membership query $\mathcal{QC}_T$ to be at most that of one encryption.

## III. New Techniques for Impossible Polytopic Distinguisher and Key Recovery

This section develops new techniques for distinguisher construction and for key recovery, and integrates them into two complementary key-recovery frameworks. For distinguisher construction, we give a formal treatment of the bound-estimation-based $\mathsf{IP}$ transition: we state its definition, develop a general construction methodology with a precise complexity evaluation, and propose two strategies that yield tighter size estimates of the distinguisher. For key recovery, we introduce the plaintext-grouping technique and the partition-guess-filter technique: the former removes

[2]For an $m$-bit input $x = (x_0, x_1, \ldots, x_{m-1})$, the $\chi$ function defines the output $y = (y_0, y_1, \ldots, y_{m-1})$ bitwise as: $y_i = x_i + (x_{i+1} + 1) \cdot x_{i+2}$ where subscripts are taken modulo $m$.

the data-complexity bottleneck that arises when rounds are prepended to the distinguisher, and the latter reduces the overall attack complexity by sieving the key candidates in several steps. Finally, we present the attack framework in two variants, the target-table framework and the candidate-table framework: the former precomputes and stores the distinguisher table offline, whereas the latter inverts the two matching sides of the sieve, which lowers the memory whenever $|\Gamma|$ exceeds $2^{|K_{\text{in}} \cup K_{\text{out}}|}$.

### *A. Formalizing and Refining Bound-Estimation-Based IP transitions*

Distinguishers have a second use besides key recovery: they tell a block cipher from a random permutation. The bound-estimation-based method of Tiessen [5] was proposed with key recovery in mind. The setting in which the key is longer than the block has features that this method matches, yet the method has not been worked out there for distinguishing. We formalize the distinguisher for that setting and systematically develop its construction framework.

**Definition 3.** *Let $E$ denote a block cipher, $\boldsymbol{\alpha}$ an input $d$-difference carried by $d+1$ plaintexts, $\eta$ the number of target bits of the output $d$-difference, and $\Gamma$ the set of all possible output target $d$-differences with $|\Gamma| \le 2^t$.*

*The pair $(\boldsymbol{\alpha}, \Gamma)$ is called a* bound-estimation-based IP transition *of $E$, and $(d, t, \eta)$ is referred to as its parameter tuple.*

Here $t$ is not a property of the cipher but an upper bound chosen for the analysis: in every instantiation it is read off the structural bound (Theorem 1) or from one of its tightenings, and a loose $t$ only weakens the exponent of Corollary 1 without affecting the soundness of the test.

Throughout this paper, the notation IP transition always denotes a bound-estimation-based IP transition. The complexity of constructing an IP transition lies in the precomputation and storage of a list of possible $d$-differences. An IP transition is called valid only if the precomputation and memory costs of its construction stay below the cipher's claimed security level. For concreteness, we illustrate the complexity evaluation on an Sbox-based cipher. Let $E = E_{R-1} \circ \cdots \circ E_1 \circ E_0$ denote an $R$-round block cipher. Let $\mathcal{A}_r$ denote the set of active-cell positions at round $r$, where $\mathcal{A}_r = \{j \mid \text{the } j\text{-th cell is affected by cells in } \mathcal{A}_{r-1}\}$ and $\mathcal{A}_0$ corresponds to the active cells of the input $d$-difference. Let $\mathcal{B}_r$ denote the set of target-cell positions at round $r$, where $\mathcal{B}_r = \{j \mid \text{the } j\text{-th cell affects cells in } \mathcal{B}_{r+1}\}$ and $\mathcal{B}_R$ corresponds to the target cells of the output $d$-difference. Two propagation trails are derived: the upper trail is obtained by forward-propagating the active cells in $\boldsymbol{\alpha}$, and the lower trail by backward-propagating the target cells from $\Gamma$.

$$\text{Upper trail}: \mathcal{A}_0 \xrightarrow[\text{fw}]{E_0} \mathcal{A}_1 \xrightarrow[\text{fw}]{E_1} \cdots \xrightarrow[\text{fw}]{E_{R'-1}} \mathcal{A}_{R'} \xrightarrow[\text{fw}]{E_{R'}} \cdots \xrightarrow[\text{fw}]{E_{R-1}} \mathcal{A}_R,$$
$$\text{Lower trail}: \mathcal{B}_0 \xrightarrow[\text{bw}]{E_0} \mathcal{B}_1 \xrightarrow[\text{bw}]{E_1} \cdots \xrightarrow[\text{bw}]{E_{R'-1}} \mathcal{B}_{R'} \xrightarrow[\text{bw}]{E_{R'}} \cdots \xrightarrow[\text{bw}]{E_{R-1}} \mathcal{B}_R.$$

By choosing a switch round $R'$, we first derive, along the upper trail, all possible $d$-differences at round $R'$ from $\boldsymbol{\alpha}$; the set $\Gamma$ is then obtained from these $d$-differences along the lower trail. Let

$$\Gamma_r = \begin{cases} \{\text{all possible } d\text{-differences at round } r\}, & \text{if } r \le R' \\ \{\text{all possible target } d\text{-differences at round } r\}, & \text{if } r > R'. \end{cases}$$

The sizes of these sets are bounded by the recurrence below, in which each factor $2^{s|\mathcal{A}_r|}$ charges every active cell $2^s$ possibilities and follows the counting convention of [5]:

$$\begin{cases} |\Gamma_0| & = 2^0, \\ |\Gamma_1| & \le |\Gamma_0| 2^{s|\mathcal{A}_0|} \le 2^{s|\mathcal{A}_0|}, \\ & \cdots, \\ |\Gamma_{R'}| & \le |\Gamma_{R'-1}| 2^{s|\mathcal{A}_{R'-1}|} \le 2^{s(\sum_{r=0}^{R'-1} |\mathcal{A}_r|)}, \\ |\Gamma_{R'+1}| & \le |\Gamma_{R'}| 2^{s|\mathcal{B}_{R'}|} \le 2^{s(\sum_{r=0}^{R'-1} |\mathcal{A}_r| + |\mathcal{B}_{R'}|)}, \\ |\Gamma_{R'+2}| & \le |\Gamma_{R'+1}| 2^{s|\mathcal{B}_{R'+1}|} \le 2^{s(\sum_{r=0}^{R'-1} |\mathcal{A}_r| + \sum_{r=R'}^{R'+1} |\mathcal{B}_r|)}, \\ & \cdots, \\ |\Gamma_R| & \le |\Gamma_{R-1}| 2^{s|\mathcal{B}_{R-1}|} \le 2^{s(\sum_{r=0}^{R'-1} |\mathcal{A}_r| + \sum_{r=R'}^{R-1} |\mathcal{B}_r|)}. \end{cases}$$

Thus, $|\Gamma| = |\Gamma_R| \le 2^{s(\sum_{r=0}^{R'-1} |\mathcal{A}_r| + \sum_{r=R'}^{R-1} |\mathcal{B}_r|)}$. The inequalities are strict whenever distinct values collide under the nonlinear layer, the equality being attained only if every extension is collision-free; Lemma 3 replaces these worst-case factors by exact counts.

The bound below is a statement about the size of a reachable set, and every such statement rests on a per-round extension factor. We state that factor explicitly rather than leaving it inside the enumeration. For each round $r$ and

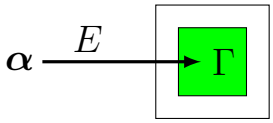


Fig. 7: Illustration for bound-estimation-based IP transition

each reachable input $d$-difference at that round, the number of reachable output $d$-differences after the substitution layer of round $r$ is at most $2^{s|\mathcal{A}_r|}$, i.e., at most $2^s$ per active cell; the linear layer then acts bijectively on differences and contributes no growth. This is the convention of [5], and it is exactly what the round-by-round enumeration above computes; the exact counts of Lemma 3 replace the factor whenever the layer is small enough to be enumerated exhaustively.

Therefore, to formalize the size estimation of the reachable output $d$-differences, we establish the following structural bound theorem.

**Theorem 1** (Structural Bound of Reachable $d$-Differences)**.** *Let $E = E_{R-1} \circ \cdots \circ E_1 \circ E_0$ be an $R$-round block cipher with cell size $s$. Given an input $d$-difference $\boldsymbol{\alpha}$ and a switch round $R'$ $(0 \le R' \le R)$, let $\mathcal{A}_r$ be the set of forward active cells at round $r$ (for $r < R'$) and $\mathcal{B}_r$ be the set of backward target cells at round $r$ (for $r \ge R'$). The size of the set of reachable output target $d$-differences, $|\Gamma|$, is upper bounded by*

$$|\Gamma| \le 2^{s\left(\sum_{r=0}^{R'-1}|\mathcal{A}_r| + \sum_{r=R'}^{R-1}|\mathcal{B}_r|\right)}. \tag{2}$$

*Proof.* Let $\Gamma_r$ be the set of $d$-differences reachable from $\boldsymbol{\alpha}$ after the substitution and linear layers of rounds $0, \ldots, r-1$, and let $\Delta_r$ be the set of $d$-differences reachable at round $r$ when starting from a single $d$-difference at the switch round $R'$ and applying rounds $R', \ldots, r-1$. Both sets obey the same recursion: a set $S$ at round $r$ is replaced at round $r+1$ by the union of the images of its elements under the substitution layer, each element contributing at most $2^{s|\mathcal{A}_r|}$ (for $\Gamma$) or $2^{s|\mathcal{B}_r|}$ (for $\Delta$) images by the counting hypothesis, while the linear layer, being bijective on differences, contributes nothing. Induction on the round index therefore yields

$$|\Gamma_{R'}| \le 2^{s\sum_{r=0}^{R'-1}|\mathcal{A}_r|}, \qquad |\Delta_{R-1}| \le 2^{s\sum_{r=R'}^{R-1}|\mathcal{B}_r|}.$$

The target cells are a subset of the state, so the reachable target set is the projection of $\Gamma_{R'} \times \Delta_{R-1}$ and therefore has at most $|\Gamma_{R'}||\Delta_{R-1}| \le 2^{s(\sum_{r<R'}|\mathcal{A}_r| + \sum_{r\ge R'}|\mathcal{B}_r|)}$ elements, which is Equation (2). The inequality is strict whenever two distinct $d$-differences reach the same value at some layer, since the two preimages then contribute one element instead of two; equality would require every extension to be collision-free, which no layer with an active cell can satisfy once its input differences take at least two distinct values. The bound is thus strictly one-sided. □

**Corollary 1** (Distinguishing Advantage)**.** *Let $(\boldsymbol{\alpha}, \Gamma)$ be a bound-estimation-based IP transition of $E$ with parameter tuple $(d, t, \eta)$, where $|\Gamma| \le 2^t$. Let the test accept a ciphertext tuple whenever its target $d$-difference lies in $\Gamma$. For any $d\eta > t$, the distinguisher separates $E$ from a random permutation with the advantage of $1 - 2^{t-d\eta}$.*

*Proof.* Under $E$ the target $d$-difference always lies in $\Gamma$, so the test accepts with probability one and never rejects the cipher. Under a random permutation $\Pi$ the $d+1$ ciphertexts are distinct, so the target $d$-difference is uniform on $\{0,1\}^{d\eta}$. Hence $\Pr_\Pi[\boldsymbol{\beta} \in \Gamma] \le |\Gamma| 2^{-d\eta}$, and the advantage, i.e., the acceptance probability under $E$ minus that under $\Pi$, is about $1 - 2^{t-d\eta}$. □

The switch round $R'$ determines the relative sizes of the forward and backward active sets. To minimize the overall complexity, one simply iterates over all admissible values of $R'$ and selects the one minimizing Equation (2). To further sharpen the bound in Equation (2), we introduce two optimization strategies, formalized as follows.

**Lemma 2** (Large-Sbox Tightening)**.** *If there exists a sequence of consecutive unkeyed rounds $E_{r_1}, \ldots, E_{r_2}$ (where $0 \le r_1 < r_2 \le R$) that can be regarded as a single large Sbox of size $s'$, and if $s' < s\sum_{r=r_1}^{r_2-1}|\mathcal{A}_r|$, then the bound on $|\Gamma|$ can be strictly tightened by replacing the corresponding sum in Equation (2) with $s'$.*

**Lemma 3** (Subtlety-Aware Tightening)**.** *If the nonlinear function in round $r$ does not achieve the worst-case bound of $2^{s|\mathcal{A}_r|}$ (e.g., due to local diffusion properties or algebraic constraints), let $N_r < 2^{s|\mathcal{A}_r|}$ denote the exact number of reachable output $d$-differences. Then, the size of $\Gamma$ is strictly upper bounded by replacing the term $2^{s|\mathcal{A}_r|}$ with $N_r$ in Equation (2).*

**Remark 1.** *Lemmas 2 and 3 justify the two strategies used in our applications: the large-Sbox strategy for LED-128 (Section IV-G) and the subtlety-aware strategy for ChiLow-32 (Section IV-H). Both strategies aim at a tighter bound than the generic worst-case count, and so at a smaller $d$.*

**Memory complexity.** Since the sets $\Gamma_r$ grow monotonically, the memory complexity is dominated by the largest set $\Gamma = \Gamma_R$. As discussed in Section II-C, the storage overhead consists of $5|\Gamma|$ slots, each holding a $\log_2 |\Gamma|$-bit pointer, and $|\Gamma|$ stored $d$-difference elements. Accordingly, the overall memory complexity is given by $\mathcal{MC}^D = |\Gamma|\cdot(d\cdot\eta + 5\cdot\log_2|\Gamma|)/n$ blocks.

**Membership-query complexity.** As discussed in Section II-C, we treat each query as an $R$-round encryption.

**Precomputation complexity.** Let $\mathcal{TC}_r$ denote the time complexity to compute $\Gamma_r$. Then,

$$\mathcal{TC}_r = \begin{cases} |\Gamma_{r-1}|2^{s|\mathcal{A}_{r-1}|}(d+1)|\mathcal{A}_{r-1}|\phi^D_{r-1} = 2^{s\left(\sum_{r'=0}^{r-1}|\mathcal{A}_{r'}|\right)}(d+1)|\mathcal{A}_{r-1}|\phi^D_{r-1}, & \text{if } 1 \le r \le R', \\ |\Gamma_{r-1}|2^{s|\mathcal{B}_{r-1}|}(d+1)|\mathcal{B}_{r-1}|\phi^D_{r-1} = 2^{s\left(\sum_{r'=0}^{R'-1}|\mathcal{A}_{r'}| + \sum_{r'=R'}^{r-1}|\mathcal{B}_{r'}|\right)}(d+1)|\mathcal{B}_{r-1}|\phi^D_{r-1}, & \text{if } r > R', \end{cases}$$

where $\phi^D_r$ $(1 \le r \le R)$ denote the equivalent factors with respect to the operation count of the encryption function. In addition, storing all possible $d$-differences via the FKS two-level perfect hashing scheme requires $2|\Gamma|$ evaluations of 2-universal hash functions, which we directly regard as $2|\Gamma|$ $R$-round encryptions. The time complexity to precompute $\Gamma$ is dominated by $\mathcal{TC}^D = \mathcal{TC}_R + 2|\Gamma|$.

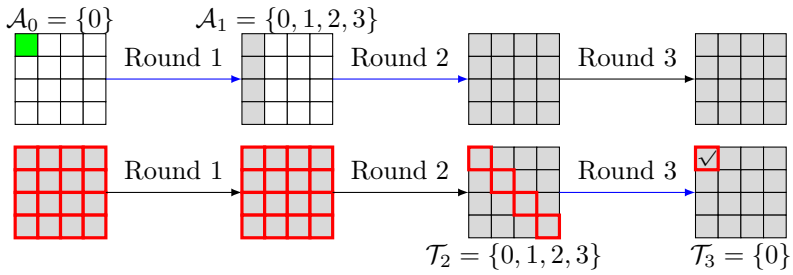


Fig. 8: Example for calculating complexity

**Example 1.** *As an example, we use the 3-round distinguisher employed to mount a 5-round attack on AES in [5], where the original work sets $d = 14$. As Figure 8 shows, the active byte of the input $d$-difference is 0, and the target byte of the output $d$-difference is 0. By choosing $R' = 2$, we obtain $|\Gamma| \le 2^{72}$. The 3-round AES is then distinguished from a random permutation with success probability $1 - 2^{72-14\cdot 8} = 1 - 2^{-40}$. The memory complexity is $2^{72}\cdot(14\cdot 8 + 5\cdot 72)/128 \approx 2^{73.88}$ blocks. The precomputation complexity is $2^{72}\cdot 15\cdot 4/(3\cdot 16) + 2^{72}\cdot 2 \approx 2^{73.70}$ 3-round encryptions.*

### *B. The plaintext-grouping technique*

The attack paradigm of Tiessen [5] performs key recovery only at the bottom of the distinguisher, which keeps the data complexity minimal but inherently limits the total number of attackable rounds. A natural remedy is to mount key recovery on both sides of the distinguisher. However, prepending rounds directly before the distinguisher forces the adversary to enumerate a large set of candidate backward-propagated $d$-differences, so the data complexity quickly becomes prohibitive even for a small number of prepended rounds. The plaintext-grouping technique removes this bottleneck by organizing the plaintexts so that a fixed small set of input differences can be realized. Let $\Theta$ be the set of the input $d$-difference components, i.e., the input $d$-difference of the distinguisher is a $d$-tuple of components drawn from $\Theta$. Let $r_b$ denote the number of rounds prepended before the distinguisher. Propagating the components $\theta \in \Theta$ backward through the $r_b$ rounds yields the set of active input bit positions; let $\nu$ denote the number of these positions. Let $\mathbb{P}$ be the plaintext set in which the $\nu$ active bits range over all $2^\nu$ values while every other bit is fixed. Fix a plaintext $P^0 \in \mathbb{P}$. Let $K_{\text{in}}$ denote the subkey material involved in the $r_b$ prepended rounds, and let $E^{r_b}_{K_{\text{in}}}$ be the corresponding partial encryption.

**Definition 4.** *A plaintext-grouping table organizes the plaintext set $\mathbb{P}$ by the key value: its entry indexed by $K_{\text{in}}$ is*

$$\mathcal{P}_{K_{\text{in}}} = \left\{P^1 \in \mathbb{P} \,\middle|\, E^{r_b}_{K_{\text{in}}}(P^0) \oplus E^{r_b}_{K_{\text{in}}}(P^1) = \theta,\ \theta \in \Theta\right\}.$$

Next, we show that, for fixed $K_{\text{in}}$ and $P^0 \in \mathbb{P}$, any prescribed input $d$-difference admits a unique corresponding group of $(d+1)$ plaintexts, which can be obtained by a single key-indexed lookup $(\mathcal{P}_{K_{\text{in}}})$.

**Theorem 2.** *Fix $K_{\text{in}}$ and $P^0 \in \mathbb{P}$. For every $\theta \in \Theta$, there is exactly one $P' \in \mathbb{P}$ such that $E^{r_b}_{K_{\text{in}}}(P^0) \oplus E^{r_b}_{K_{\text{in}}}(P') = \theta$.*

*Proof.* Since $E^{r_b}_{K_{\text{in}}}(P^0) \oplus \theta$ is fixed and $E^{r_b}_{K_{\text{in}}}$ is a bijection, the equation admits the unique solution

$$P' = (E^{r_b}_{K_{\text{in}}})^{-1}\left(E^{r_b}_{K_{\text{in}}}(P^0) \oplus \theta\right).$$

It remains to argue that $P' \in \mathbb{P}$. By construction of the active positions, any input difference that propagates to the component $\theta$ after $r_b$ rounds is necessarily zero outside the active set. The difference $P^0 \oplus P'$ is such an input difference, so $P'$ coincides with $P^0$ on every non-active position and differs from $P^0$ only on the $\nu$ active positions, which is precisely what $\mathbb{P}$ permits. Hence $P' \in \mathbb{P}$. □

The plaintext set can thus be grouped into $2^{|K_{\text{in}}|}$ (possibly overlapping) entries, each of size $|\mathcal{P}_{K_{\text{in}}}| = |\Theta|$, since distinct $\theta$ values give distinct solutions $P'$. The memory cost of the plaintext-grouping table is $|\Theta| \cdot 2^{|K_{\text{in}}|} \cdot \nu/n$ blocks. The table is constructed by evaluating $E^{r_b}_{K_{\text{in}}}(P^1)$ for all $K_{\text{in}}$ and $P^1 \in \mathbb{P}$, which costs $2^{|K_{\text{in}}|+\nu}\,\phi^B$ encryption equivalents, where $\phi^B$ denotes the computational cost equivalent to one full encryption operation. The data complexity is $2^\nu$ plaintexts. When the table is used in the attack, for a guessed key $K_{\text{in}}$, the entry $\mathcal{P}_{K_{\text{in}}}$ provides up to $|\Theta|$ plaintext pairs whose $r_b$-round output difference lies in $\Theta$. We pre-enumerate key candidates and, for each candidate, gather $(d+1)$ plaintexts $P^0, \ldots, P^d$ such that, for the prescribed $d$-difference $\boldsymbol{\alpha} = (\theta_1, \ldots, \theta_d)$, the pair $(P^0, P^i)$ satisfies $E^{r_b}_{K_{\text{in}}}(P^0) \oplus E^{r_b}_{K_{\text{in}}}(P^i) = \theta_i$ for $1 \le i \le d$. These plaintexts are then fed through the $r_b$ prepended rounds to realize $\boldsymbol{\alpha}$, followed by the distinguisher-based sieve.

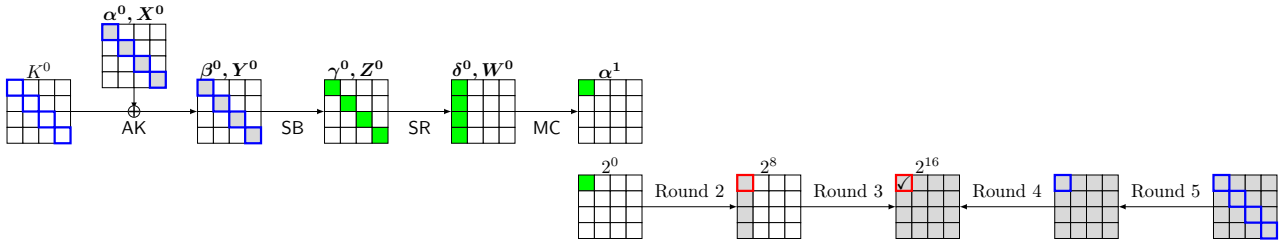


Fig. 9: Add one round at the top of the example IP transition

**Example 2.** *As an example, we prepend one more round at the top of the 4-round IP attack against AES in [5]. The 4-round IP attack, as illustrated in the bottom part of Figure 9, takes an input 7-difference $\boldsymbol{\alpha}^1$ activating only the first byte. It yields at most $2^8$ feasible 7-differences after round 2, and at most $2^{16}$ on the first byte after round 3 as the target. Exchanging AK and MC, one guesses 40 bits of subkey to recover the round-3 first-byte difference; as only $2^{16}$ of the $2^{56}$ possible 7-differences are reachable, a guess survives with probability $2^{-40}$, leaving one candidate on average. The attack needs 8 chosen plaintexts, with time complexity $2^{38}$ 4-round encryptions and memory $2^{15}$ blocks. Building on this attack, we prepend one more round at the top, as shown in the top part of Figure 9. Accordingly, we take $\Theta$ as the set of all nonzero byte differences of $\boldsymbol{\alpha}^1$ on byte 0, so $|\Theta| = 7$. Since the single active byte spreads to a full column after one round, each $\theta \in \Theta$ induces exactly 32 active input bits, governed by the key bytes $K_{\text{in}} = K^0_0 \| K^0_5 \| K^0_{10} \| K^0_{15}$. Fix $P^0$ and let $P^1$ vary over only these 32 active bits; for the one-round encryption $E^1_{K_{\text{in}}}$, for each $\theta \in \Theta$ the equation $E^1_{K_{\text{in}}}(P^0) \oplus E^1_{K_{\text{in}}}(P^1) = \theta$ has exactly one solution $P^1$ for a fixed key. Selecting eight plaintexts $P^0, \ldots, P^7$ accordingly (placing the variable bytes at bytes $0, 5, 10, 15$ of the state, leaving all other bytes unchanged) realizes $\boldsymbol{\alpha}^1$ after one round. Guessing the 32 bits of $K^0$ and the 40 bits of $K^4 \| K^5$ then extends the attack to five rounds: the data complexity becomes $2^{32}$, the time complexity is $2^{32}$ times that of the original attack, and the memory complexity is unchanged.*

### C. The partition-guess-filter technique

In contrast to existing approaches that guess all required subkeys at once, we propose the partition-guess-filter technique, which performs key recovery in several steps and lowers the overall attack complexity. The $\eta$ output target bits are partitioned into $Q$ groups, each associated with at least one distinct subkey involved in key recovery; let the $i$-th group correspond to $\iota_i$ output target bits ($0 \le i \le Q-1$). The distinguisher is split into $Q$ sub-distinguishers. For the $i$-th sub-distinguisher, we focus on the target output bits indexed by $\sum_{j=0}^{i} \iota_j$ and store the corresponding distinguisher for these bits independently; the result is $Q$ sub-distinguisher tables. Meanwhile, the key recovery proceeds in $Q$ steps: at step $i$, we exploit target output bits indexed by $\sum_{j=0}^{i} \iota_j$ whose states are available so far to filter key candidates by querying the corresponding sub-distinguisher tables. Suppose that step $i$ further guesses $NK_i$ bits of subkeys and retains a fraction $\varphi_i$ of the candidates. Then

$$\varphi_0 = |\Gamma|/2^{d\iota_0}, \quad \varphi_i = \frac{|\Gamma|/2^{d\sum_{j=0}^{i}\iota_j}}{|\Gamma|/2^{d\sum_{j=0}^{i-1}\iota_j}} = 2^{-d\iota_i}, \tag{3}$$

The first factor is measured against the whole target space because no filter has yet been applied, whereas the later factors are ratios of successive survival fractions. since the $\sum_{j=0}^{i-1} \iota_j$ output target bits have already been exploited in the previous steps. The overall time complexity is

$$\mathcal{TC} = (d+1)\cdot 2^{NK_0}\cdot \phi_0^P + (d+1)\cdot 2^{NK_0}\cdot \varphi_0 \cdot 2^{NK_1}\cdot \phi_1^P + \cdots + (d+1)\cdot (\prod_{i=0}^{Q-2} 2^{NK_i}\cdot \varphi_i)\cdot 2^{NK_{Q-1}}\cdot \phi_{Q-1}^P, \quad (4)$$

where $\phi_i^P$ denotes the ratio of the cost of the partial decryption to that of a full encryption[3].

To formalize the advantage of this technique, we present the following theorem, which specifies the condition under which the stepwise sieve strictly outperforms the one-time full-key guessing.

**Theorem 3.** *Let $NK_{\text{total}} = \sum_{i=0}^{Q-1} NK_i$ be the total number of guessed subkey bits. The time complexity of the one-time full-key guessing is $\mathcal{TC}_{\text{naive}} = 2^{NK_{\text{total}}}$. If for every step $i \in \{1,\ldots,Q-1\}$, the guessed key bits satisfy*

$$NK_i < d\cdot \iota_i, \quad (5)$$

*then the partition-guess-filter technique strictly reduces the overall attack complexity, i.e., $\mathcal{TC} < \mathcal{TC}_{\text{naive}}$. Specifically, the time complexity is dominated by the step that maximizes $\sum_{j=0}^{i} NK_j + t - d\sum_{j=0}^{i-1}\iota_j$, which is typically the initial step.*

### *D. The target-table framework of Impossible Polytopic Key-Recovery Attacks*

Given an $r_d$-round $\mathsf{IP}$ transition, we prepend $r_b$ rounds before the distinguisher and append $r_f$ rounds after it, mounting an $(r_b + r_d + r_f)$-round attack. We describe the case $r_b \neq 0$; the case $r_b = 0$ is similar. The attack proceeds in four phases.

1) **Output side (partition-guess-filter).** Partition the $\eta$ output target bits of the distinguisher into $Q$ groups, the $i$-th group ($0 \le i \le Q-1$) containing $\iota_i$ bits and being associated with a distinct set of subkeys. If $Q > 1$, we focus on the target output bits indexed by $\sum_{j=0}^{i}\iota_j$ and store the corresponding distinguisher for these bits independently, therefore, we generate $Q$ sub-distinguisher tables $\Gamma_i$ ($0 \le i \le Q-1$).
2) **Input side (plaintext-grouping).** Let $\Theta$ be the component set of the input $d$-difference $\boldsymbol{\alpha}$, and let $K_{\text{in}}$ denote the subkeys involved in the $r_b$ prepended rounds. Propagating each $\theta \in \Theta$ backward through $r_b$ rounds yields $\nu$ active plaintext bits; let $\mathbb{P}$ be the plaintext set that is free exactly on these bits. Fix $P^0 \in \mathbb{P}$ and build the plaintext-grouping table
$$\mathcal{P}_{K_{\text{in}}} = \left\{P^1 \in \mathbb{P} \;\middle|\; E_{K_{\text{in}}}^{r_b}(P^0) \oplus E_{K_{\text{in}}}^{r_b}(P^1) = \theta,\ \theta \in \Theta\right\}.$$
By Theorem 2, each entry contains exactly $|\Theta|$ plaintexts, one per component.
3) **Online sieve.** For each candidate $K_{\text{in}}$:
   a) Fetch $d+1$ plaintexts realizing $\boldsymbol{\alpha}$ from $\mathcal{P}_{K_{\text{in}}}$ via a key-indexed lookup, and obtain their ciphertexts;
   b) For each candidate $K_{\text{out}}$, partially decrypt the last $r_f$ rounds, compute the output $d$-difference on the $\eta$ target bits, and discard the candidate unless it belongs to $\Gamma$. In particular, when $Q > 1$, $K_{\text{out}}$ is partitioned into $Q$ sets $K_{\text{out},i}$, and the key recovery is performed in $Q$ steps. For each candidate $K_{\text{out},i}$, we partially decrypt the last $r_f$ rounds, compute the output $d$-difference over the target output bits indexed by $\sum_{j=0}^{i}\iota_j$ by querying the corresponding sub-distinguisher tables, and discard the candidate unless it lies in $\Gamma_i$.
4) **Exhaustive search.** After the sieve, $2^{|K_{\text{in}}\cup K_{\text{out}}|}\cdot 2^{t-d\eta}$ candidates for $K_{\text{in}}\cup K_{\text{out}}$ survive; for each of them, the remaining $|K| - |K_{\text{in}}\cup K_{\text{out}}|$ key bits are enumerated.

**Complexity.**

- The data complexity is $\mathcal{DC} = 2^{\nu}$.
- The memory consists of the distinguisher table and the plaintext-grouping table. As discussed in Section III-A, storing the distinguisher table requires $\mathcal{MC}^D = \sum_{i=0}^{Q-1} |\Gamma|\cdot(d\cdot\sum_{j=0}^{i}\iota_j + 5\cdot\log_2|\Gamma|)/n$ blocks, while the plaintext-grouping table consumes $|\Theta|\cdot 2^{|K_{\text{in}}|}\cdot\nu/n$ blocks. Overall, the total memory complexity is
$$\mathcal{MC} = (\sum_{i=0}^{Q-1}|\Gamma|\cdot(d\cdot\sum_{j=0}^{i}\iota_j + 5\cdot\log_2|\Gamma|) + |\Theta|\cdot 2^{|K_{\text{in}}|}\cdot\nu)/n$$
blocks.

[3]Since every sub-distinguisher table is the projection of the common set $\Gamma$ onto its own target bits, and projections cannot enlarge a set, the number of admissible projections on the cumulative target bits is at most $|\Gamma|$; hence the cumulative survival fraction is $|\Gamma|/2^{d\sum_{j\le i}\iota_j}$, and the conditional fraction of step $i$ is $2^{-d\iota_i}$.

- Distinguisher construction and plaintext grouping are treated as precomputation and are not charged to the attack. However, we need to ensure that the complexity of these precomputations does not exceed the security level, which is satisfied in all applications of this work. The time consists of the sieve in Phase 3 and the exhaustive search in Phase 4. The nested loops of Algorithm 1 prune the candidates after every step, so that step $i$ only processes the survivors of step $i-1$; the bound below nevertheless charges every step with the full candidate set, and is therefore an upper bound. This bound concerns the sieve alone, i.e., the nested enumeration of Algorithm 1; the partition-guess-filter of Section III-C is a separate optimization layered on top of this enumeration, and it is that optimization which Theorem 3 compares with one-time full-key guessing, so the upper-bound statement here and the strict reduction proved there are consistent. Phase 3 costs

$$\mathcal{TC}_{\text{sieve}} = \sum_{i=0}^{Q-1} (d+1) \cdot 2^{|K_{\text{in}} \cup K_{\text{out},i}|} \cdot \phi^{r_f,i} + 2^{|K_{\text{in}} \cup K_{\text{out},i}|},$$

  where $\phi^{r_f,i}$ denotes the partial-decryption cost ratio, measured relative to one full encryption.
- The exhaustive search costs

$$\mathcal{TC}_{\text{search}} = 2^{|K_{\text{in}} \cup K_{\text{out}}| + t - d\eta} \cdot 2^{|K| - |K_{\text{in}} \cup K_{\text{out}}|} = 2^{|K| + t - d\eta}.$$

  The total time complexity is $\mathcal{TC} = \mathcal{TC}_{\text{sieve}} + \mathcal{TC}_{\text{search}}$.

The entire target-table key-recovery procedure is summarized in Algorithm 1.

**Algorithm 1:** Target-Table Framework for IP Key-Recovery

**Input:** Input $d$-difference $\boldsymbol{\alpha}$, target bits $\eta$ with partition $\{\iota_i\}_{i=0}^{Q-1}$, rounds $r_b, r_f$, target tables $\Gamma_0, \ldots, \Gamma_{Q-1}$, plaintext-grouping tables $\mathcal{P}_{K_{\text{in}}}$

**Output:** Surviving key candidates $K_{\text{in}} \cup K_{\text{out}}$

```
foreach candidate K_in do
    Fetch d+1 plaintexts realizing α from P_{K_in} via key-indexed lookup;
    Encrypt the plaintexts to obtain ciphertexts;
    foreach candidate K_{out,0} do
        Partially decrypt under K_{out,0}; check the d-difference on bits indexed by ι_0 against Γ_0; discard the
         candidate if it is not in Γ_0;
        foreach candidate K_{out,1} only for surviving K_{out,0} do
            Partially decrypt under K_{out,1}; check the d-difference on bits indexed by ι_0 + ι_1 against Γ_1; discard if
             not in Γ_1;
                ⋮
            foreach candidate K_{out,Q-1} only for surviving K_{out,Q-2} do
                Partially decrypt under K_{out,Q-1}; keep the candidate if the d-difference on all η bits lies in Γ_{Q-1};
foreach surviving (K_in ∪ K_out) do
    Enumerate the remaining |K| − |K_in ∪ K_out| key bits to recover the full master key;
```

### *E. The candidate-table framework of Impossible Polytopic Key-Recovery Attacks*

The large memory of the distinguisher table can be reduced by swapping the two matching sides of the sieve whenever $|\Gamma|$ exceeds $2^{|K_{\text{in}} \cup K_{\text{out}}|}$. That is, instead of precomputing and storing the set $\Gamma$ of possible output $d$-differences offline and checking each decrypted $d$-difference against it online, we partially decrypt the ciphertexts under every subkey candidate and store the resulting output $d$-difference candidates in a candidate table, in the same layout as the target table (Section III-D). The elements of $\Gamma$, now computed online, are then matched against this table. The plaintext-grouping technique of Section III-B is still applied at the top of the distinguisher.

In this variant, an $r_d$-round IP transition is specified solely by its input $d$-difference $\boldsymbol{\alpha}$ and its target positions, without materializing the set $\Gamma$ of possible target $d$-differences. We prepend $r_b$ rounds before the distinguisher and append $r_f$ rounds after it, mounting an $(r_b + r_d + r_f)$-round attack. The attack proceeds in three phases.

1) **Input side (plaintext-grouping).** Identical to Phase 2 in Section III-D: build the plaintext-grouping table $\mathcal{P}_{K_{\text{in}}}$ and realize the input $d$-difference $\boldsymbol{\alpha}$ via Theorem 2.
2) **Online sieve.**

a) For each candidate $K_{\text{in}}$, fetch $d+1$ plaintexts realizing $\boldsymbol{\alpha}$ from $\mathcal{P}_{K_{\text{in}}}$ via a key-indexed lookup and obtain their corresponding ciphertexts ; this yields $2^{|K_{\text{in}}|}$ ciphertext tuples in total. Each tuple is partially decrypted under every $K_{\text{out}}$ candidate, producing $2^{|K_{\text{in}} \cup K_{\text{out}}|}$ candidate target $d$-differences. These differences are stored in the table $\overline{\Gamma}$ together with the associated guessed key material.

3) **Exhaustive search.** After the sieve, $2^{|K_{\text{in}} \cup K_{\text{out}}|} \cdot 2^{t-d\eta}$ candidates for $K_{\text{in}} \cup K_{\text{out}}$ survive; for each of them, the remaining $|K| - |K_{\text{in}} \cup K_{\text{out}}|$ key bits are enumerated.

**Complexity.** The data complexity is $\mathcal{DC} = 2^{\nu}$. The memory cost is dominated by the storage of the candidate table and the plaintext-grouping table, the reachable:

$$\mathcal{MC} = (2^{|K_{\text{in}} \cup K_{\text{out}}|} \cdot \big(d \cdot \eta + 5 \cdot |K_{\text{in}} \cup K_{\text{out}}| + |K_{\text{in}} \cup K_{\text{out}}|\big) + |\Theta| \cdot 2^{|K_{\text{in}}|} \cdot \nu)/n.$$

The three terms inside the parentheses are distinct: $d \cdot \eta$ bits hold the $d$-difference stored for the candidate, $5 \cdot |K_{\text{in}} \cup K_{\text{out}}|$ bits account for the five FKS pointers of Section II-C (each of $\log_2 |\Gamma| = |K_{\text{in}} \cup K_{\text{out}}|$ bits in this variant), and the trailing $|K_{\text{in}} \cup K_{\text{out}}|$ bits store the guessed key by which the candidate is indexed, so that the entry can be translated back into a key candidate after the sieve. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = \big((d+1) \cdot 2^{|K_{\text{in}} \cup K_{\text{out}}|} \cdot \phi^{r_f} + 2 \cdot 2^{|K_{\text{in}} \cup K_{\text{out}}|}\big) + \big((d+1) \cdot |\Gamma| \cdot \phi^{D} + |\Gamma|\big),$$

where $\phi^{r_f}$ denotes the partial-decryption cost ratio relative to one full encryption, and $\phi^{D}$ denotes the cost ratio of computing one target $d$-difference of $\Gamma$. The exhaustive search costs

$$\mathcal{TC}_{\text{search}} = 2^{|K| + t - d\eta},$$

and the total time complexity is $\mathcal{TC} = \mathcal{TC}_{\text{sieve}} + \mathcal{TC}_{\text{search}}$.

The candidate-table framework, which trades time for memory, is summarized in Algorithm 2.

**Algorithm 2:** Candidate-Table Framework for IP Key-Recovery

**Input:** Input $d$-difference $\boldsymbol{\alpha}$, target positions, rounds $r_b, r_f$, plaintext-grouping tables $\mathcal{P}_{K_{\text{in}}}$
**Output:** Surviving key candidates $K_{\text{in}} \cup K_{\text{out}}$

```
Initialize candidate table Γ̄ ← ∅;
foreach candidate K_in do
    Fetch d+1 plaintexts realizing α from P_{K_in} via key-indexed lookup;
    Encrypt the plaintexts to obtain ciphertexts;
    foreach candidate K_out do
        Partially decrypt the last r_f rounds under K_out;
        Compute the candidate target d-difference β;
        Store the tuple (β, K_in, K_out) into Γ̄;
Enumerate the reachable target d-differences of α;
foreach entry (β, K_in, K_out) whose β occurs in the enumeration do
    Mark (K_in, K_out) as a surviving candidate;
foreach surviving (K_in ∪ K_out) do
    Enumerate the remaining |K| − |K_in ∪ K_out| key bits to recover the full master key;
```

## IV. The Chosen-Plaintext/Chosen-Ciphertext Distinguishers and Attacks

We apply our method in the Chosen-Plaintext (CP) setting to present IP transitions and key-recovery attacks for the block ciphers Deoxys-BC-384, Deoxys-BC-256, Joltik-BC-192, Joltik-BC-128, AES-256, AES-192, LED-128 and ChiLow-32, and in the Chosen-Ciphertext (CC) setting to present IP transitions for ChiLow-32. The results are summarized in Table I.

**Distinguishers.** Impossible differentials deployed in key-recovery attacks on AES and AES-like ciphers reach at most 4 rounds; ours go beyond that by 1.5, 0.5, 1.5, 0.5, 0.5 and 3.5 rounds on Deoxys-BC-384, Deoxys-BC-256, Joltik-BC-192, Joltik-BC-128, AES-256 and LED-128, respectively; for LED-128, whose round keys are applied only every four rounds, this relies on the large-Sbox strategy. ChiLow-32 is the only cipher we treat in both the CP and CC settings: in the single-tweakey setting the data complexity drops from $2^8$ chosen plaintexts over 3.5 rounds [16] to 5 chosen plaintexts over 4 rounds, which is, to our knowledge, at once the longest and the lowest distinguisher recorded in that setting.

**Attacks.** Our 10-round IP attacks on Deoxys-BC-384 and Joltik-BC-192 go one round beyond the best known ID attacks, and they do so for a structural reason: the IP transition spans 5 rounds where its ID counterpart spans 4. To

our knowledge no earlier IP attack exceeds an ID attack in round count on any cipher. On Deoxys-BC-256 and Joltik-BC-128 our 8-round attacks match the maximal round count of the known ID attacks while using less data and less total time. On AES-192 (7 rounds) and AES-256 (8 rounds), whose strongest low-data competitors are the multiple-of-8 and MITM attacks, respectively, we attain the lowest data complexity of all known attacks at these round counts, and for ChiLow-32 the full-round attack in the CP setting needs 8 plaintexts against 64.

### A. Deoxys-BC-384

#### (1) IP Distinguishers

- **Low-complexity 5.5-round distinguishers.** Activating a single byte of the input $d$-difference and selecting a single target byte of the output $d$-difference gives $16 \times 16 \times \binom{255}{d}$ IP transitions in total. An example of such a distinguisher is given below.

  **Distinguisher 1.** *Let $\boldsymbol{\alpha}^0$ denote the input $d$-difference, where only the components $\alpha^0_{\ell,0}$ are non-zero for $0 \le \ell \le d-1$. Let $\boldsymbol{\alpha}'^6$ denote the output $d$-difference without the final MC operation, and let $\Gamma$ be the set of all possible tuples $(\alpha'^6_{0,0}, \ldots, \alpha'^6_{d-1,0})$. As Figure 10 shows, $|\Gamma| \le 2^{336}$. Setting $d = 43$ lets us distinguish the 6-round Deoxys-BC-384 from a random permutation with success probability $1 - 2^{336-43\cdot 8} = 1 - 2^{-8}$.*
  *Building the table requires $2^{336}\cdot(43\cdot 8+5\cdot 336)/128 \approx 2^{339.98}$ blocks of storage and $2^{336}\cdot(43+1)\cdot 1/(6\cdot 16)+2^{336}\cdot 2 \approx 2^{337.30}$ counts of 6-round encryptions.*

- **Low-data 5.5-round distinguishers.** Increasing the number of output target bytes of the above IP transitions reduces the data complexity. When the target bytes correspond to the four bytes lying on an anti-diagonal, we obtain $16 \times 4 \times \binom{255}{d}$ low-data IP transitions in total. A concrete instance follows.

  **Distinguisher 2.** *Let $\boldsymbol{\alpha}^0$ denote the input $d$-difference, where only the components $\alpha^0_{*,0}$ are non-zero. Let $\boldsymbol{\alpha}'^6$ denote the output $d$-difference without the final MC operation, and let $\Gamma$ be the set of all possible tuples*

  $$\big(\alpha'^6_{0,0} \| \alpha'^6_{0,7} \| \alpha'^6_{0,10} \| \alpha'^6_{0,13}, \ldots, \alpha'^6_{d-1,0} \| \alpha'^6_{d-1,7} \| \alpha'^6_{d-1,10} \| \alpha'^6_{d-1,13}\big).$$

  *As Figure 10 shows, all possible values of $\boldsymbol{\beta}^5$ amount to $2^{328}$, and hence $|\Gamma| \le 2^{328} \times 2^{32} = 2^{360}$. With $d = 12$ we distinguish the 6-round Deoxys-BC-384 from a random permutation with success probability $1 - 2^{360-12\cdot 32} = 1 - 2^{-24}$. The table occupies $2^{360}\cdot(12\cdot 32+5\cdot 360)/128 \approx 2^{364.09}$ blocks; its construction takes $2^{360}\cdot(12+1)\cdot 4/(6\cdot 16)+2^{360}\cdot 2 \approx 2^{361.35}$ counts of 6-round encryptions.*

#### (2) IP Key-Recovery Attacks

- **Low-data 10-round attack.** We exploit the 5-round IP transition for key recovery to construct a 10-round IP attack on Deoxys-BC-384 by prepending one round ($r_b = 1$) and appending four rounds ($r_f = 4$), shown in Figure 11. The distinguisher is instantiated with the input $d$-difference $\boldsymbol{\alpha}^1$, where only the components $\alpha^1_{\ell,0}$ and $\alpha^1_{\ell,1}$ are non-zero and

  $$\mathsf{MC}^{-1} \cdot (\alpha^1_{\ell,0}, \alpha^1_{\ell,1}, 0, 0)^T = (*, *, *, 0)^T \tag{6}$$

  for $0 \le \ell \le d-1$; and the target byte $\boldsymbol{\beta}^6_{*,0}$, for which $|\Gamma| \le 2^{368}$. The subkeys involved in the appended rounds are

  $$K_{\text{out}} = \{K'^7_0,\ K'^8_0,\ K'^8_7,\ K'^8_{10},\ K'^8_{13}\} \cup K'^9 \cup K'^{10}, \qquad |K_{\text{out}}| = 296,$$

  and the subkeys involved in the prepended round are $K_{\text{in}} = \{K^0_0, K^0_5, K^0_{10}\}$ with $|K_{\text{in}}| = 24$, hence $|K_{\text{in}} \cup K_{\text{out}}| = 320$. The following plaintext-grouping step is shared by both the target-table and the candidate-table frameworks.

  *Plaintext-grouping.* Let $\Theta$ be the nonzero component set of $\boldsymbol{\alpha}^1$. Propagating each $\theta \in \Theta$ backward through one round yields $\nu = 24$ active plaintext bits, namely the bytes $P_0, P_5, P_{10}$; let $\mathbb{P}$ be the plaintext set that is free exactly on these bytes ($|\mathbb{P}| = 2^{24}$). Fix $P^0 \in \mathbb{P}$ and build the plaintext-grouping table

  $$\mathcal{P}_{K_{\text{in}}} = \left\{ P^1 \in \mathbb{P} \;\middle|\; \mathsf{MC} \cdot \begin{pmatrix} S(P^0_0 \oplus K^0_0) \oplus S(P^1_0 \oplus K^0_0) \\ S(P^0_5 \oplus K^0_5) \oplus S(P^1_5 \oplus K^0_5) \\ S(P^0_{10} \oplus K^0_{10}) \oplus S(P^1_{10} \oplus K^0_{10}) \\ 0 \end{pmatrix} = \begin{pmatrix} * \\ * \\ 0 \\ 0 \end{pmatrix} \right\}. \tag{7}$$

  Theorem 2 guarantees that each $K_{\text{in}}$ admits exactly one $(d+1)$-tuple in $\mathbb{P}$ realizing $\boldsymbol{\alpha}^1$.

  - **Target-table variant.** Setting $d = 55$ leaves the sieve fraction $|\Gamma|/2^{d\eta} = 2^{368-55\cdot 8} = 2^{-72}$.
    1) $\Gamma$*-precomputing.* We build and store $\Gamma$ in $2^{368} \cdot (55 \cdot 8 + 5 \cdot 368)/128 \approx 2^{372.15}$ blocks.
    2) *Online sieve.* For each candidate $K_{\text{in}}$, retrieve the unique plaintext tuple $X^0_0 = P^0, X^0_1, \ldots, X^0_d$ from $\mathcal{P}_{K_{\text{in}}}$ that realizes $\boldsymbol{\alpha}^1$ via a key-indexed lookup, and obtain their ciphertexts $(Y^{10}_0, \ldots, Y^{10}_d)$. For each candidate $K_{\text{out}}$, partially decrypt the last 4 rounds of the ciphertexts, compute the candidate target $d$-difference $\boldsymbol{\beta}^6_{*,0}$, and discard the candidate unless it belongs to $\Gamma$.

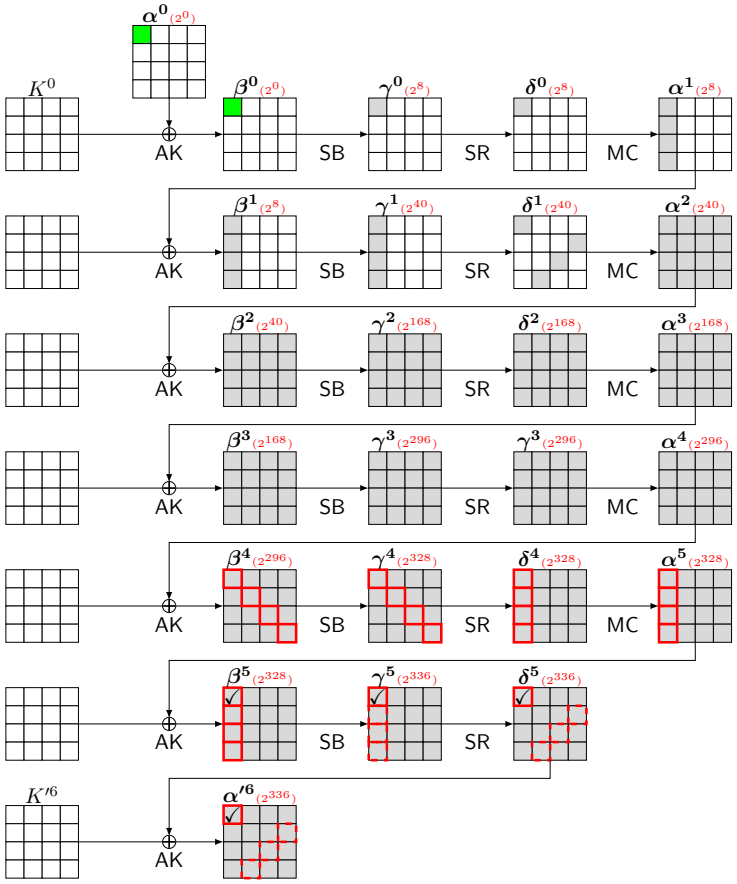


Fig. 10: The low-complexity 5.5-round IP transition for Deoxys-BC-384. The low-data distinguishers differ only in the selection of the target bytes (marked by the dashed line): the low-complexity variant targets a single byte, which keeps $|\Gamma|$ small and hence the memory low, whereas the low-data variant targets the four bytes on the anti-diagonal, which enlarges $|\Gamma|$ but needs a much smaller $d$, and hence far less data.

3) *Exhaustive search.* The sieve leaves $2^{|K_{\text{in}} \cup K_{\text{out}}|} \cdot 2^{t-d\eta} = 2^{320-72} = 2^{248}$ candidates for $K_{\text{in}} \cup K_{\text{out}}$; enumerating the remaining $|K| - |K_{\text{in}} \cup K_{\text{out}}| = 64$ key bits for each survivor recovers the full key.

*Complexity.* The attack uses $\mathcal{DC} = 2^{\nu} = 2^{24}$ plaintexts. The distinguisher table dominates the memory footprint, $\mathcal{MC} \approx 2^{372.15}$ blocks, since the plaintext-grouping table which costs $|\Theta| \cdot 2^{|K_{\text{in}}|} \cdot \nu/n$ blocks is negligible. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = (55+1) \cdot 2^{320} \cdot \frac{1+4+16+16}{16 \cdot 10} + 2^{320} \approx 2^{323.80},$$

while the exhaustive search adds $2^{248+64} = 2^{312}$. Hence $\mathcal{TC} \approx 2^{323.80}$ counts of 10-round encryptions, the sieve being the dominant term.

- **Candidate-table variant.** With $d = 49$, the sieve retains a fraction $|\Gamma|/2^{d\eta} = 2^{368-49 \cdot 8} = 2^{-24}$.
  1) *Online sieve.* Compute and store $\overline{\Gamma}$. For the input $d$-difference $\boldsymbol{\alpha}$, derive the possible target $d$-differences online, and query the candidate table $\overline{\Gamma}$ with $|\overline{\Gamma}| \leq 2^{320}$; a candidate $(K_{\text{in}}, K_{\text{out}})$ survives iff its stored $d$-difference is matched.
  2) *Exhaustive search.* $2^{|K_{\text{in}} \cup K_{\text{out}}|} \cdot 2^{t-d\eta} = 2^{320-24} = 2^{296}$ candidates for $K_{\text{in}} \cup K_{\text{out}}$ survive the sieve, and enumerating the remaining $|K| - |K_{\text{in}} \cup K_{\text{out}}| = 64$ key bits per survivor completes the key recovery.

  *Complexity.* The data complexity is $\mathcal{DC} = 2^{\nu} = 2^{24}$. The memory is dominated by the candidate table, which costs $\mathcal{MC} \approx 2^{320} \cdot (49 \cdot 8 + 5 \cdot 320 + 320)/128 \approx 2^{324.17}$ blocks. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = \left((49+1) \cdot 2^{320} \cdot \frac{1+4+16+16}{16 \cdot 10} + 2 \cdot 2^{320}\right) + \left((49+1) \cdot 2^{368} \cdot \frac{4}{16 \cdot 10} + 2^{368}\right) \approx 2^{369.17}$$

and exhaustive key search accounts for a further $2^{296+64} = 2^{360}$. The sieve dominates the total cost, which is $\mathcal{TC} \approx 2^{369.17}$ counts of 10-round encryptions.

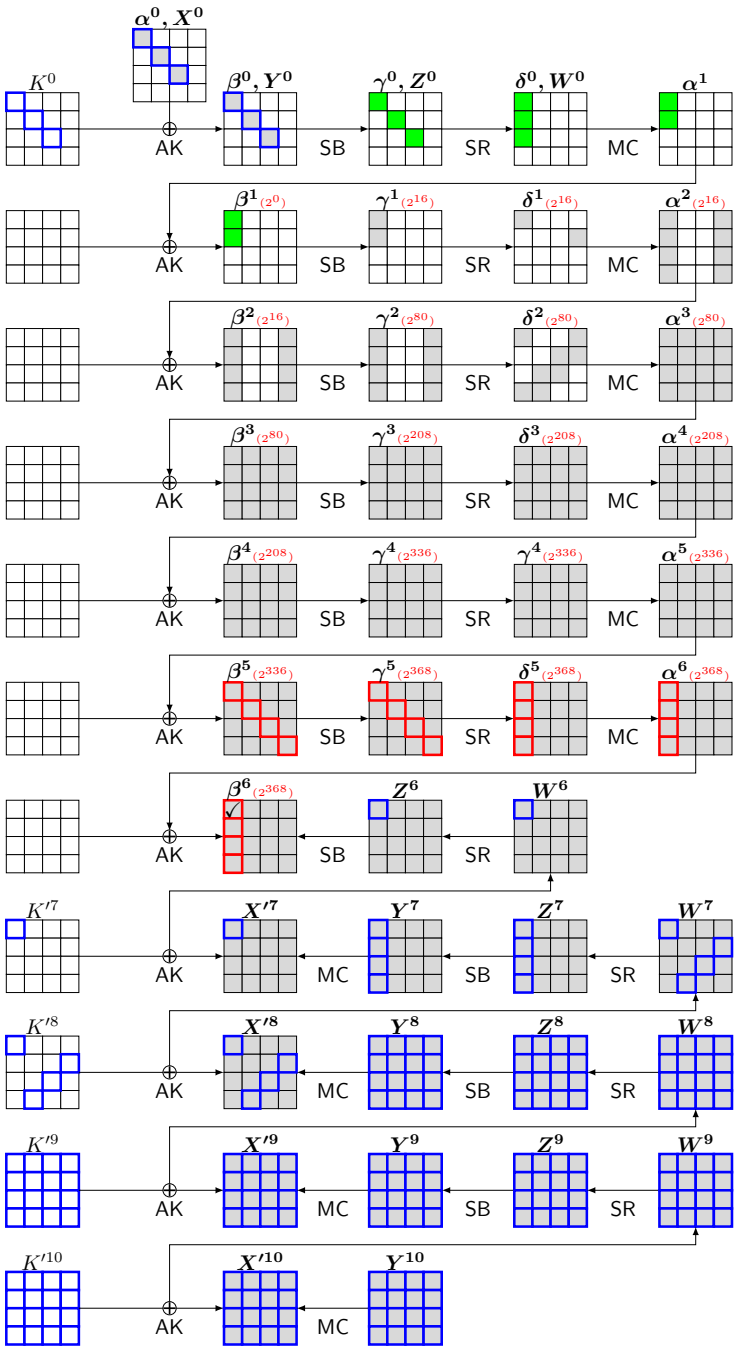


Fig. 11: The low-data 10-round IP attacks of Deoxys-BC-384

- **Low-complexity 10-round attack.** Similar to the low-data attack, we exploit another 5-round IP transition to mount a 10-round IP attack on Deoxys-BC-384 by prepending one round (as in Figure 9) and appending four rounds (as in Figure 11). The distinguisher is instantiated with the input $d$-difference $\boldsymbol{\alpha}^1$, corresponding to a single active input byte (only the components $\alpha^1_{\ell,0}$ are non-zero for $0 \le \ell \le d-1$), and the target byte $\boldsymbol{\beta}^6_{*,0}$, i.e., the pattern $(\boldsymbol{\beta}^0, \boldsymbol{\beta}^5)$ of the low-complexity 5.5-round IP transition in Figure 10, with $|\Gamma| \le 2^{328}$. The subkeys involved in the appended rounds are

$$K_{\text{out}} = \{K'^7_0, K'^8_0, K'^8_7, K'^8_{10}, K'^8_{13}\} \cup K'^9 \cup K'^{10}, \qquad |K_{\text{out}}| = 296,$$

and the subkeys involved in the prepended round are $K_{\text{in}} = \{K^0_0, K^0_5, K^0_{10}, K^0_{15}\}$ with $|K_{\text{in}}| = 32$, hence $|K_{\text{in}} \cup K_{\text{out}}| = 328$. Since $\log_2 |\Gamma| = |K_{\text{in}} \cup K_{\text{out}}| = 328$, the target-table and the candidate-table frameworks achieve the same memory; the latter is not beneficial. Therefore, we describe only the target-table variant.

*Plaintext-grouping.* Let $\Theta$ be the nonzero component set of $\boldsymbol{\alpha}^1$. Propagating each $\theta \in \Theta$ backward through one round yields $\nu = 32$ active plaintext bits, namely the bytes $P_0, P_5, P_{10}, P_{15}$; let $\mathbb{P}$ be the plaintext set that is free exactly on

these bytes ($|\mathbb{P}| = 2^{32}$). Fix $P^0 \in \mathbb{P}$ and build the plaintext-grouping table

$$\mathcal{P}_{K_{\text{in}}} = \left\{ P^1 \in \mathbb{P} \;\middle|\; \mathsf{MC} \cdot \begin{pmatrix} S(P_0^0 \oplus K_0^0) \oplus S(P_0^1 \oplus K_0^0) \\ S(P_5^0 \oplus K_5^0) \oplus S(P_5^1 \oplus K_5^0) \\ S(P_{10}^0 \oplus K_{10}^0) \oplus S(P_{10}^1 \oplus K_{10}^0) \\ S(P_{15}^0 \oplus K_{15}^0) \oplus S(P_{15}^1 \oplus K_{15}^0) \end{pmatrix} = \begin{pmatrix} * \\ 0 \\ 0 \\ 0 \end{pmatrix} \right\}. \tag{8}$$

For each $K_{\text{in}}$, Theorem 2 pins down exactly one such $(d+1)$-tuple in $\mathbb{P}$.

- **Target-table variant.** Fixing $d = 48$ gives the sieve fraction $|\Gamma|/2^{d\eta} = 2^{328-48\cdot 8} = 2^{-56}$.
  1) $\Gamma$-*precomputing.* $\Gamma$ is precomputed and stored in $2^{328} \cdot (48 \cdot 8 + 5 \cdot 328)/128 \approx 2^{331.98}$ blocks.
  2) *Online sieve.* For each candidate $K_{\text{in}}$, look up the unique plaintext tuple $X_0^0 = P^0, X_1^0, \ldots, X_{48}^0$ from $\mathcal{P}_{K_{\text{in}}}$ that realizes $\boldsymbol{\alpha}^1$ via a key-indexed lookup, and obtain their ciphertexts $(Y_0^{10}, \ldots, Y_{48}^{10})$. For each candidate $K_{\text{out}}$, partially decrypt the last 4 rounds of the ciphertexts, compute the candidate target $d$-difference $\boldsymbol{\beta}_{*,0}^6$, and discard the candidate unless it belongs to $\Gamma$.
  3) *Exhaustive search.* What the sieve leaves is $2^{|K_{\text{in}} \cup K_{\text{out}}|} \cdot 2^{t-d\eta} = 2^{328-56} = 2^{272}$ candidates for $K_{\text{in}} \cup K_{\text{out}}$; the remaining $|K| - |K_{\text{in}} \cup K_{\text{out}}| = 56$ key bits are then enumerated for each of them, recovering the full key.

  *Complexity.* Only $2^{32}$ plaintexts are needed, i.e., $\mathcal{DC} = 2^{\nu} = 2^{32}$. Memory is governed by the distinguisher table, $\mathcal{MC} \approx 2^{331.98}$ blocks; the plaintext-grouping table costs $|\Theta| \cdot 2^{|K_{\text{in}}|} \cdot \nu/n$ blocks, which is negligible. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = (48+1) \cdot 2^{328} \cdot \frac{1+4+16+16}{16 \cdot 10} + 2^{328} \approx 2^{331.62},$$

  with the final enumeration of the key bits costing $2^{272+56} = 2^{328}$. The total time complexity is $\mathcal{TC} = 2^{331.62} + 2^{328} \approx 2^{331.73}$ counts of 10-round encryptions.

### *B. Deoxys-BC-256*

#### *(1) IP Distinguishers*

- **Low-complexity 4.5-round distinguishers.** Activating a single byte of the input $d$-difference and selecting a single target byte of the output $d$-difference gives $16 \times 16 \times \binom{255}{d}$ IP transitions in total. An example distinguisher is as follows.

  **Distinguisher 3.** *Let $\boldsymbol{\alpha}^0$ denote the input $d$-difference, where only the components $\alpha_{*,0}^0$ are non-zero. Let $\boldsymbol{\alpha}'^5$ denote the output $d$-difference without the final MC operation, and let $\Gamma$ be the set of all possible tuples $(\alpha_{0,0}'^5, \ldots, \alpha_{d-1,0}'^5)$. As Figure 12 shows, $|\Gamma| \le 2^{208}$. With $d = 27$ the 5-round Deoxys-BC-256 is distinguished from a random permutation with success probability $1 - 2^{208-27\cdot 8} = 1 - 2^{-8}$.*
  *$2^{208} \cdot (27+1) \cdot 1/(5 \cdot 16) + 2^{208} \cdot 2 \approx 2^{209.23}$ counts of 5-round encryptions, and the table is stored in $2^{208} \cdot (27 \cdot 8 + 5 \cdot 208)/128 \approx 2^{211.29}$ blocks.*

- **Low-data 4.5-round distinguishers.** Increasing the number of output target bytes of the above IP transitions reduces the data complexity. When the target bytes correspond to the four bytes lying on an anti-diagonal, we obtain $16 \times 4 \times \binom{255}{d}$ low-data IP transitions in total. An example distinguisher is as follows.

  **Distinguisher 4.** *Let $\boldsymbol{\alpha}^0$ denote the input $d$-difference, where only the components $\alpha_{*,0}^0$ are non-zero. Let $\boldsymbol{\alpha}'^5$ denote the output $d$-difference without the final MC operation, and let $\Gamma$ be the set of all possible tuples*

$$\big(\alpha_{0,0}'^5 \| \alpha_{0,7}'^5 \| \alpha_{0,10}'^5 \| \alpha_{0,13}'^5, \ldots, \alpha_{d-1,0}'^5 \| \alpha_{d-1,7}'^5 \| \alpha_{d-1,10}'^5 \| \alpha_{d-1,13}'^5\big).$$

  *As Figure 12 shows, all possible values of $\boldsymbol{\beta}^4$ amount to $2^{200}$, and hence $|\Gamma| \le 2^{200} \times 2^{32} = 2^{232}$. $d = 8$ yields a distinguisher that tells apart 5-round Deoxys-BC-256 from a random permutation with success probability $1 - 2^{232-8\cdot 32} = 1 - 2^{-24}$. Storage for the table is $2^{232} \cdot (8 \cdot 32 + 5 \cdot 232)/128 \approx 2^{235.47}$ blocks, while construction consumes $2^{232} \cdot (8+1) \cdot 4/(5 \cdot 16) + 2^{232} \cdot 2 \approx 2^{233.29}$ counts of 5-round encryptions.*

#### *(2) IP Key-Recovery Attacks*

- **Low-data 8-round attack.** We exploit the 4-round IP transition for key recovery to construct an 8-round IP attack on Deoxys-BC-256 by prepending one round ($r_b = 1$) and appending three rounds ($r_f = 3$). The distinguisher is instantiated with the input $d$-difference $\boldsymbol{\alpha}^1$, where only the components $\alpha_{\ell,0}^1$ and $\alpha_{\ell,1}^1$ are non-zero and satisfy the same constraint as in Equation (6) for $0 \le \ell \le d-1$; and the target byte $\boldsymbol{\beta}_{*,0}^5$, for which $|\Gamma| \le 2^{240}$. The subkeys involved in the appended rounds are

$$K_{\text{out}} = \{K_0'^6, K_0'^7, K_7'^7, K_{10}'^7, K_{13}'^7\} \cup K'^8, \qquad |K_{\text{out}}| = 168,$$

  and the subkeys involved in the prepended round are $K_{\text{in}} = \{K_0^0, K_5^0, K_{10}^0\}$ with $|K_{\text{in}}| = 24$, hence $|K_{\text{in}} \cup K_{\text{out}}| = 192$. The following plaintext-grouping step is shared by both the target-table and the candidate-table frameworks.

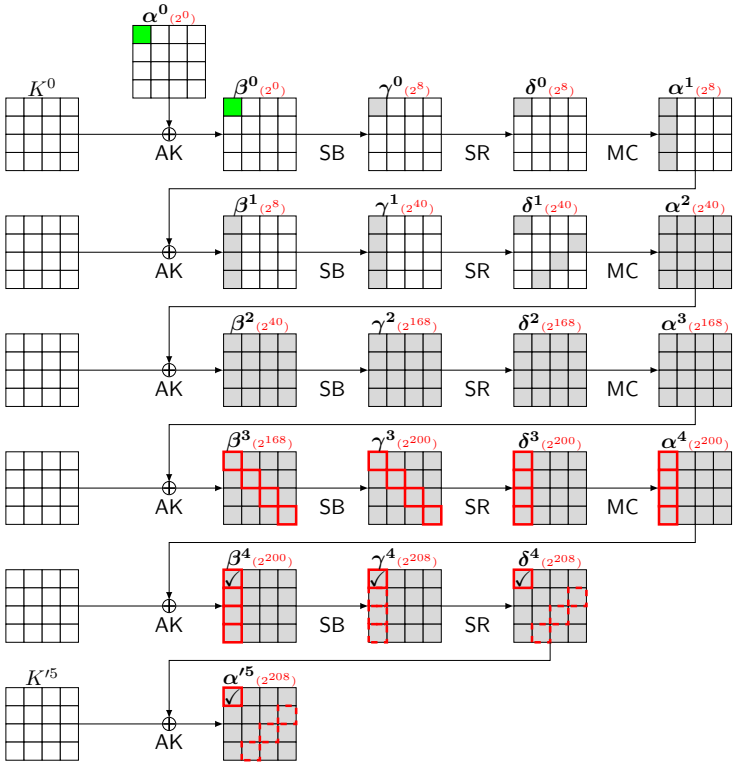


Fig. 12: The low-complexity 4.5-round IP transition for Deoxys-BC-256. The low-data distinguishers differ only in the selection of the target bytes (marked by the dashed line).

*Plaintext-grouping.* Let $\Theta$ be the nonzero component set of $\boldsymbol{\alpha}^1$. Propagating each $\theta \in \Theta$ backward through one round yields $\nu = 24$ active plaintext bits, namely the bytes $P_0, P_5, P_{10}$; let $\mathbb{P}$ be the plaintext set that is free exactly on these bytes ($|\mathbb{P}| = 2^{24}$). Fix $P^0 \in \mathbb{P}$ and build the plaintext-grouping table as in Equation (7). Exactly one $(d+1)$-tuple in $\mathbb{P}$ realizes $\boldsymbol{\alpha}^1$ per $K_{\text{in}}$, by Theorem 2.

- **Target-table variant.** $d$ is set to 39, which yields the sieve fraction $|\Gamma|/2^{d\eta} = 2^{240-39\cdot 8} = 2^{-72}$.
  1) $\Gamma$-*precomputing.* Storing the precomputed $\Gamma$ takes $2^{240} \cdot (39 \cdot 8 + 5 \cdot 240)/128 \approx 2^{243.56}$ blocks.
  2) *Online sieve.* For each candidate $K_{\text{in}}$, fetch the single plaintext tuple $X_0^0 = P^0, X_1^0, \ldots, X_d^0$ from $\mathcal{P}_{K_{\text{in}}}$ that realizes $\boldsymbol{\alpha}^1$ via a key-indexed lookup, and obtain their ciphertexts $(Y_0^8, \ldots, Y_d^8)$. For each candidate $K_{\text{out}}$, partially decrypt the last 3 rounds of the ciphertexts, compute the candidate target $d$-difference $\boldsymbol{\beta}^5_{*,0}$, and discard the candidate unless it belongs to $\Gamma$.
  3) *Exhaustive search.* $2^{|K_{\text{in}} \cup K_{\text{out}}|} \cdot 2^{t-d\eta} = 2^{192-72} = 2^{120}$ candidates for $K_{\text{in}} \cup K_{\text{out}}$ pass the sieve; a final enumeration of $|K| - |K_{\text{in}} \cup K_{\text{out}}| = 64$ key bits per survivor yields the full key.

  *Complexity.* The data requirement is $\mathcal{DC} = 2^{\nu} = 2^{24}$. The dominant memory term is the distinguisher table, $\mathcal{MC} \approx 2^{243.56}$ blocks; the plaintext-grouping table costs $|\Theta| \cdot 2^{|K_{\text{in}}|} \cdot \nu/n$ blocks, which is negligible. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = (39+1) \cdot 2^{192} \cdot \frac{1+4+16}{16 \cdot 8} + 2^{192} \approx 2^{194.93},$$

  and the final key search contributes $2^{120+64} = 2^{184}$. Overall, the attack costs $\mathcal{TC} \approx 2^{194.93}$ counts of 8-round encryptions, dominated by the sieve.
- **Candidate-table variant.** Setting $d = 33$ leaves the sieve fraction $|\Gamma|/2^{d\eta} = 2^{240-33\cdot 8} = 2^{-24}$.
  1) *Online sieve.* Compute and store $\overline{\Gamma}$. For the input $d$-difference $\boldsymbol{\alpha}$, derive the possible target $d$-differences online, and query the candidate table $\overline{\Gamma}$ with $|\overline{\Gamma}| \leq 2^{192}$; a candidate $(K_{\text{in}}, K_{\text{out}})$ survives iff its stored $d$-difference is matched.
  2) *Exhaustive search.* The sieve leaves $2^{|K_{\text{in}} \cup K_{\text{out}}|} \cdot 2^{t-d\eta} = 2^{192-24} = 2^{168}$ candidates for $K_{\text{in}} \cup K_{\text{out}}$; enumerating the remaining $|K| - |K_{\text{in}} \cup K_{\text{out}}| = 64$ key bits for each survivor recovers the full key.

  *Complexity.* The attack uses $2^{24}$ plaintexts, i.e., $\mathcal{DC} = 2^{\nu} = 2^{24}$. The memory is dominated by the candidate table, which costs $\mathcal{MC} = 2^{192} \cdot (33 \cdot 8 + 5 \cdot 192 + 192)/128 \approx 2^{195.47}$ blocks. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = \left((33+1) \cdot 2^{192} \cdot \frac{1+4+16}{16 \cdot 8} + 2 \cdot 2^{192}\right) + \left((33+1) \cdot 2^{240} \cdot \frac{4}{16 \cdot 8} + 2^{240}\right) \approx 2^{241.04}$$

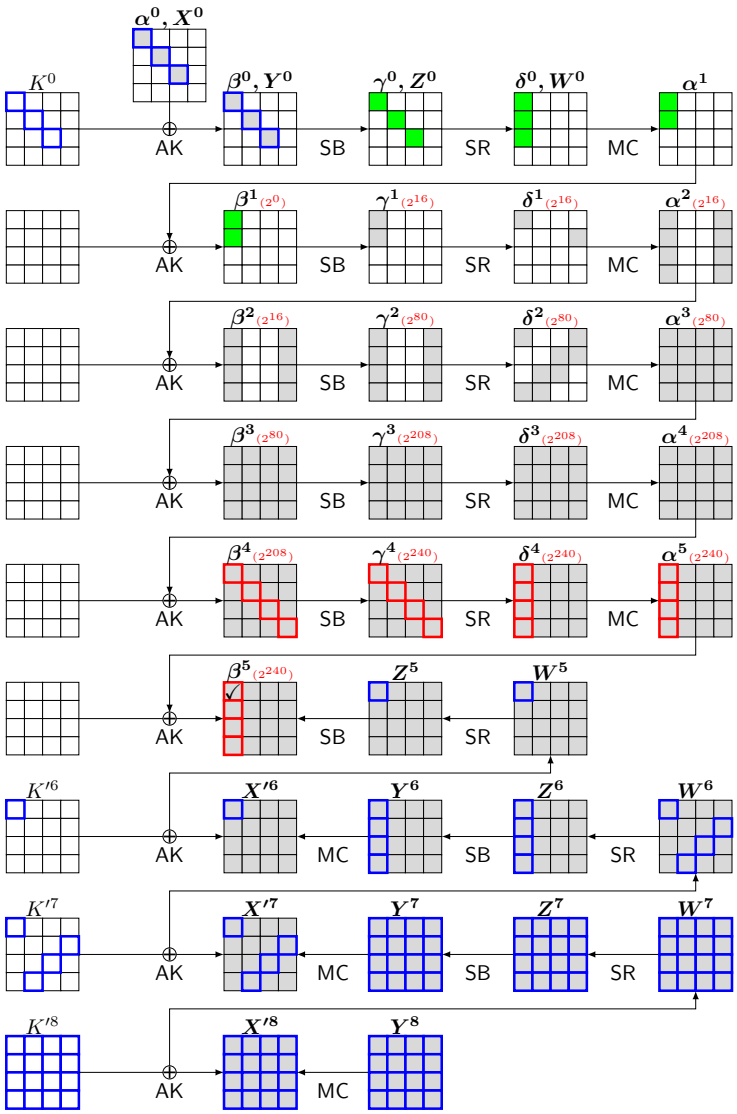


Fig. 13: The low-data 8-round IP attack of Deoxys-BC-256

while the exhaustive search adds $2^{168+64} = 2^{232}$. Thus $\mathcal{TC} \approx 2^{241.04}$ counts of 8-round encryptions, with the sieve as the leading term.

- **Low-complexity 8-round attack.** Similar to the low-data attack, we exploit another 4-round IP transition, with a single active input byte, to mount an 8-round IP attack on Deoxys-BC-256 by prepending one round (as in Figure 9) and appending three rounds (as in Figure 13). The distinguisher is instantiated with the input $d$-difference $\boldsymbol{\alpha}^1$, where only the components $\alpha^1_{\ell,0}$ are non-zero for $0 \le \ell \le d-1$, and the target byte $\boldsymbol{\beta}^5_{*,0}$, i.e., the pattern $(\boldsymbol{\beta}^0, \boldsymbol{\beta}^5)$ of the low-complexity 4.5-round IP transition in Figure 12, with $|\Gamma| \le 2^{200}$. The subkeys involved in the appended rounds are

$$K_{\text{out}} = \{K'^6_0, K'^7_0, K'^7_7, K'^7_{10}, K'^7_{13}\} \cup K'^8, \qquad |K_{\text{out}}| = 168,$$

and the subkeys involved in the prepended round are $K_{\text{in}} = \{K^0_0, K^0_5, K^0_{10}, K^0_{15}\}$ with $|K_{\text{in}}| = 32$, hence $|K_{\text{in}} \cup K_{\text{out}}| = 200$. Since $\log_2 |\Gamma| = |K_{\text{in}} \cup K_{\text{out}}|$, the candidate-table framework brings no memory advantage; we therefore describe only the target-table variant.

*Plaintext-grouping.* Let $\Theta$ be the nonzero component set of $\boldsymbol{\alpha}^1$. Propagating each $\theta \in \Theta$ backward through one round yields $\nu = 32$ active plaintext bits, namely the bytes $P_0, P_5, P_{10}, P_{15}$; let $\mathbb{P}$ be the plaintext set that is free exactly on these bytes ($|\mathbb{P}| = 2^{32}$). Fix $P^0 \in \mathbb{P}$ and build the plaintext-grouping table as in Equation (8). Theorem 2 guarantees that each $K_{\text{in}}$ admits exactly one $(d+1)$-tuple in $\mathbb{P}$ realizing $\boldsymbol{\alpha}^1$.

- **Target-table variant.** With $d = 32$, the sieve retains a fraction $|\Gamma|/2^{d\eta} = 2^{200-32\cdot 8} = 2^{-56}$.

  1) $\Gamma$-*precomputing.* The table $\Gamma$ occupies $2^{200} \cdot (32 \cdot 8 + 5 \cdot 200)/128 \approx 2^{203.29}$ blocks of memory.

2) *Online sieve.* For each candidate $K_{\text{in}}$, retrieve the unique plaintext tuple $X_0^0 = P^0, X_1^0, \ldots, X_{32}^0$ from $\mathcal{P}_{K_{\text{in}}}$ that realizes $\boldsymbol{\alpha}^1$ via a key-indexed lookup, and obtain their ciphertexts $(Y_0^8, \ldots, Y_{32}^8)$. For each candidate $K_{\text{out}}$, partially decrypt the last 3 rounds of the ciphertexts, compute the candidate target $d$-difference $\boldsymbol{\beta}_{*,0}^5$, and discard the candidate unless it belongs to $\Gamma$.
3) *Exhaustive search.* $2^{|K_{\text{in}} \cup K_{\text{out}}|} \cdot 2^{t-d\eta} = 2^{200-56} = 2^{144}$ candidates for $K_{\text{in}} \cup K_{\text{out}}$ survive the sieve, and enumerating the remaining $|K| - |K_{\text{in}} \cup K_{\text{out}}| = 56$ key bits per survivor completes the key recovery.

*Complexity.* Data complexity: $\mathcal{DC} = 2^\nu = 2^{32}$. The distinguisher table dominates the memory footprint, $\mathcal{MC} \approx 2^{203.09}$ blocks; the plaintext-grouping table costs $|\Theta| \cdot 2^{|K_{\text{in}}|} \cdot \nu/n$ blocks, which is negligible. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = (32+1) \cdot 2^{200} \cdot \frac{1+4+16}{16 \cdot 8} + 2^{200} \approx 2^{202.68},$$

and exhaustive key search accounts for a further $2^{144+56} = 2^{200}$. The total time complexity is $\mathcal{TC} = 2^{200} + 2^{202.68} \approx 2^{202.89}$ counts of 8-round encryptions.

### C. Joltik-BC-192

Joltik-BC-192 is the lightweight variant of Deoxys-BC-384 obtained by shrinking the state cells from bytes to nibbles. The description in this section therefore parallels Section IV-A, and we point out only the differences.

#### (1) IP Distinguishers

- **Low-complexity 5.5-round distinguishers.** Unlike Deoxys-BC, Joltik-BC uses a 4-bit Sbox, which restricts the $d$-difference case of a single active Sbox to $d \leq 15$. To keep the filtering fraction $|\Gamma|/2^{d\eta}$ small under this bound on $d$, we enlarge the number $\eta$ of target bits: the target is set to the three nibbles lying on an anti-diagonal, which yields $16 \times 16 \times \binom{15}{d}$ IP transitions in total.
An example distinguisher is as follows.

**Distinguisher 5.** *Let $\boldsymbol{\alpha}^0$ denote the input $d$-difference, where only the components $\alpha_{*,0}^0$ are non-zero. Let $\boldsymbol{\alpha}'^6$ denote the output $d$-difference without the final MC operation, and let $\Gamma$ be the set of all possible tuples*

$$(\alpha_{0,0}'^6 \| \alpha_{0,10}'^6 \| \alpha_{0,13}'^6, \ldots, \alpha_{d-1,0}'^6 \| \alpha_{d-1,10}'^6 \| \alpha_{d-1,13}'^6).$$

*As Figure 14 shows, $|\Gamma| \leq 2^{176}$. Setting $d = 15$ lets us distinguish the 6-round Joltik-BC-192 without the final MC operation from a random permutation with success probability $1 - 2^{176-15\cdot 12} = 1 - 2^{-4}$.*
*We allocate $2^{176} \cdot (15 \cdot 12 + 5 \cdot 176)/64 \approx 2^{180.05}$ blocks to the table and pay $2^{176} \cdot (15+1) \cdot 3/(6 \cdot 16) + 2^{176} \cdot 2 \approx 2^{177.32}$ counts of 6-round encryptions for building it.*

- **Low-data 5.5-round distinguishers.** These are the nibble analogues of the low-data distinguishers of Deoxys-BC-384: The target bytes correspond to the four nibbles lying on an anti-diagonal, and we obtain $16 \times 4 \times \binom{15}{d}$ low-data IP transitions in total. We give a concrete instance below.

**Distinguisher 6.** *Let $\boldsymbol{\alpha}^0$, $\boldsymbol{\alpha}'^6$, and $\Gamma$ be defined as in Distinguisher 2, with all 8-bit bytes replaced by 4-bit nibbles. As Figure 14 shows, $|\Gamma| \leq 2^{180}$. With $d = 12$ we distinguish the 6-round Joltik-BC-192 from a random permutation with success probability $1 - 2^{-12}$. Construction of the table costs $2^{180} \cdot (12+1) \cdot 4/(6 \cdot 16) + 2^{180} \cdot 2 \approx 2^{181.34}$ counts of 6-round encryptions and occupies $2^{180} \cdot (12 \cdot 16 + 5 \cdot 180)/64 = 2^{184.09}$ blocks.*

#### (2) IP Key-Recovery Attacks

- **The 10-round attack.** This is the nibble analogue of the low-data 10-round attack on Deoxys-BC-384: the 5-round IP transition is instantiated with the input $d$-difference $\boldsymbol{\alpha}^1$, where only the components $\alpha_{*,0}^1$ and $\alpha_{*,1}^1$ are non-zero; and the target nibble $\boldsymbol{\beta}_{*,0}^6$, for which $|\Gamma| \leq 2^{184}$. The subkeys involved are

$$K_{\text{out}} = \{K_0'^7, K_0'^8, K_7'^8, K_{10}'^8, K_{13}'^8\} \cup K'^9 \cup K'^{10}, \qquad |K_{\text{out}}| = 148,$$

and $K_{\text{in}} = \{K_0^0, K_5^0, K_{10}^0, K_{15}^0\}$ with $|K_{\text{in}}| = 16$, hence $|K_{\text{in}} \cup K_{\text{out}}| = 164$. Both the target-table and the candidate-table frameworks are applicable. The $\Gamma$-precomputing, the online sieve, and the exhaustive search proceed exactly as in Section IV-A; the only difference is that the plaintext-grouping table is built on the nibbles $P_0, P_5, P_{10}, P_{15}$ ($\nu = 16$, $\mathcal{DC} = 2^{16}$):

$$\mathcal{P}_{K_{\text{in}}} = \left\{ P^1 \in \mathbb{P} \;\middle|\; \mathsf{MC} \cdot \begin{pmatrix} S(P_0^0 \oplus K_0^0) \oplus S(P_0^1 \oplus K_0^0) \\ S(P_5^0 \oplus K_5^0) \oplus S(P_5^1 \oplus K_5^0) \\ S(P_{10}^0 \oplus K_{10}^0) \oplus S(P_{10}^1 \oplus K_{10}^0) \\ S(P_{15}^0 \oplus K_{15}^0) \oplus S(P_{15}^1 \oplus K_{15}^0) \end{pmatrix} = \begin{pmatrix} * \\ * \\ 0 \\ 0 \end{pmatrix} \right\}. \tag{9}$$

For each $K_{\text{in}}$, the lemma pins down exactly one such $(d+1)$-tuple in $\mathbb{P}$.

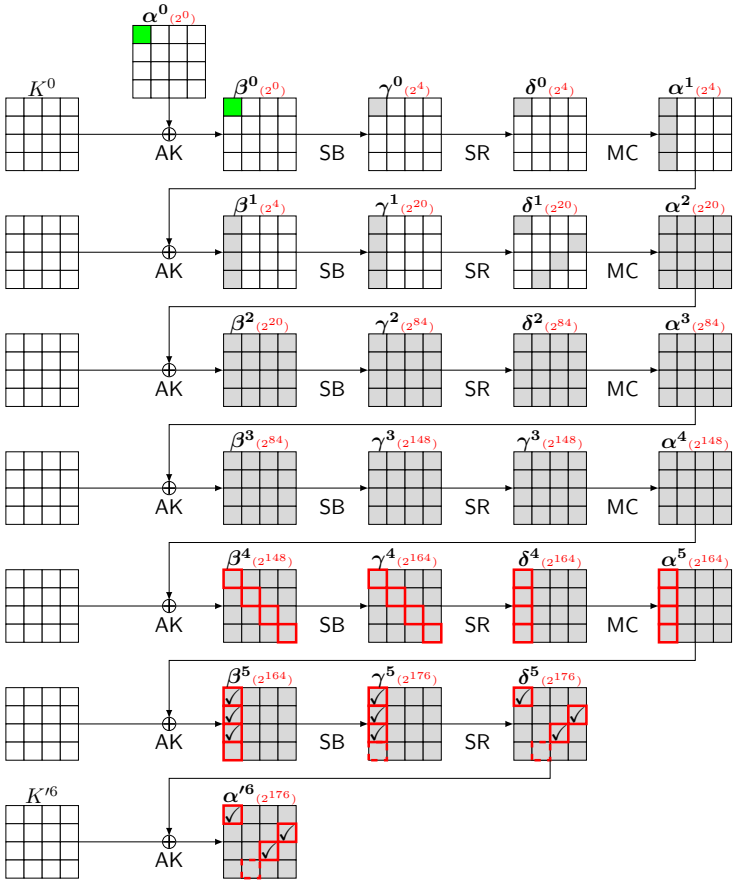


Fig. 14: The low-complexity 5.5-round IP transition for Joltik-BC-192. The low-data distinguishers differ only in the selection of the target bytes (marked by the dashed line).

- **Target-table variant.** Fixing $d = 55$ gives the sieve fraction $2^{184-55\cdot4} = 2^{-36}$. Only $2^{16}$ plaintexts are needed, so $\mathcal{DC} = 2^{\nu} = 2^{16}$. Memory is governed by the distinguisher table, $\mathcal{MC} = 2^{184} \cdot (55 \cdot 4 + 5 \cdot 184)/64 = 2^{188.15}$ blocks. The sieve costs
$$\mathcal{TC}_{\text{sieve}} = (55+1) \cdot 2^{164} \cdot \frac{1+4+16+16}{16 \cdot 10} + 2^{164} \approx 2^{167.80},$$
with the final enumeration of the key bits costing $2^{192-36} = 2^{156}$. Hence $\mathcal{TC} \approx 2^{167.80}$ counts of 10-round encryptions, the sieve being the dominant term.
- **Candidate-table variant.** $d$ is set to 50, which yields the sieve fraction $2^{184-50\cdot4} = 2^{-16}$. Still $\mathcal{DC} = 2^{\nu} = 2^{16}$ plaintexts are needed. The memory is dominated by the candidate table $\overline{\Gamma}$ with $|\overline{\Gamma}| \leq 2^{164}$, $\mathcal{MC} = 2^{164} \cdot (50 \cdot 4 + 5 \cdot 164 + 164)/64 \approx 2^{168.21}$ blocks. The sieve costs
$$\mathcal{TC}_{\text{sieve}} = \left((50+1) \cdot 2^{164} \cdot \frac{1+4+16+16}{16 \cdot 10} + 2 \cdot 2^{164}\right) + \left((50+1) \cdot 2^{184} \cdot \frac{4}{16 \cdot 10} + 2^{184}\right) \approx 2^{185.19}$$
and the final key search contributes $2^{192-16} = 2^{176}$. The sieve dominates the total cost, which is $\mathcal{TC} \approx 2^{185.19}$ counts of 10-round encryptions.

### *D. Joltik-BC-128*

Joltik-BC-128 is the lightweight variant of Deoxys-BC-256 obtained by shrinking the state cells from bytes to nibbles. The description in this section therefore parallels Section IV-B, and we point out only the differences.

#### *(1) IP Distinguishers*

- **Low-complexity 4.5-round distinguishers.** Activating a single nibble of the input $d$-difference and selecting two target nibbles lying on an anti-diagonal of the output $d$-difference gives $16 \times 24 \times \binom{15}{d}$ IP transitions in total. One such distinguisher is displayed below.

**Distinguisher 7.** *Let $\boldsymbol{\alpha}^0$ denote the input $d$-difference, where only the components $\alpha^0_{*,0}$ are non-zero. Let $\boldsymbol{\alpha}'^5$ denote the output $d$-difference without the final MC operation, and let $\Gamma$ be the set of all possible tuples $(\alpha'^5_{0,0} \| \alpha'^5_{0,13}, \ldots, \alpha'^5_{d-1,0} \| \alpha'^5_{d-1,13})$.*

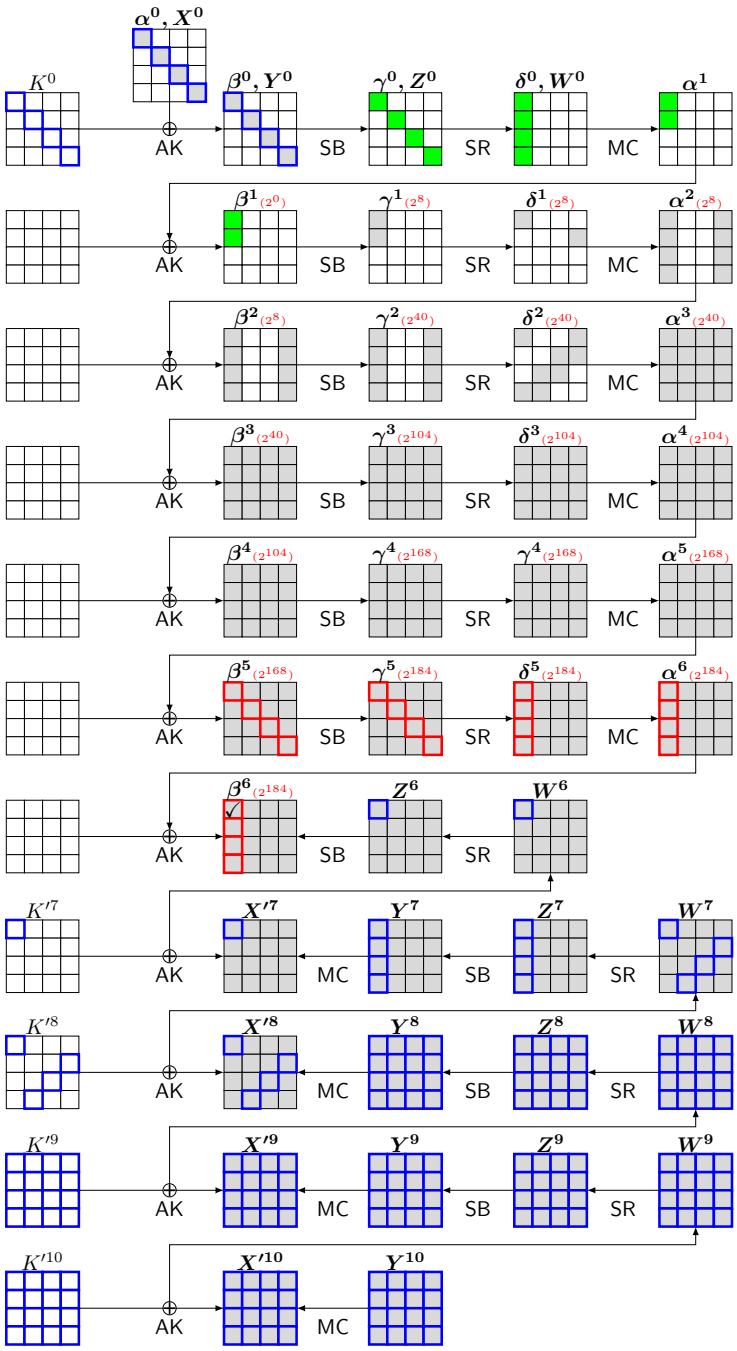


Fig. 15: The 10-round IP attack of Joltik-BC-192

*As Figure 16 shows,* $|\Gamma| \leq 2^{108}$. *With* $d = 14$ *the 5-round Joltik-BC-128 without the final MC operation is distinguished from a random permutation with success probability* $1 - 2^{108-14\cdot 8} = 1 - 2^{-4}$.
*Building the table requires* $2^{108} \cdot (14 \cdot 8 + 5 \cdot 108)/64 \approx 2^{111.34}$ *blocks of storage and* $2^{108} \cdot (14+1) \cdot 2/(6 \cdot 16) + 2^{108} \cdot 2 \approx 2^{109.21}$ *counts of 5-round encryptions.*

- **Low-data 4.5-round distinguishers.** These are the nibble analogues of the low-data distinguishers of Deoxys-BC-256: the target bytes correspond to the four nibbles lying on an anti-diagonal, and we obtain $16 \times 4 \times \binom{15}{d}$ low-data IP transitions in total. An example distinguisher is as follows.

  **Distinguisher 8.** *Let* $\boldsymbol{\alpha}^0$, $\boldsymbol{\alpha}'^5$, *and* $\Gamma$ *be defined as in Distinguisher 2, with all 8-bit bytes replaced by 4-bit nibbles. As Figure 16 shows,* $|\Gamma| \leq 2^{116}$. $d = 8$ *yields a distinguisher that tells apart the 5-round Joltik-BC-128 from a random permutation with success probability* $1 - 2^{-12}$. *The table occupies* $2^{116} \cdot (8 \cdot 16 + 5 \cdot 116)/64 \approx 2^{119.47}$ *blocks; its construction takes* $2^{116} \cdot (8+1) \cdot 4/(6 \cdot 16) + 2^{116} \cdot 2 \approx 2^{117.25}$ *counts of 5-round encryptions.*

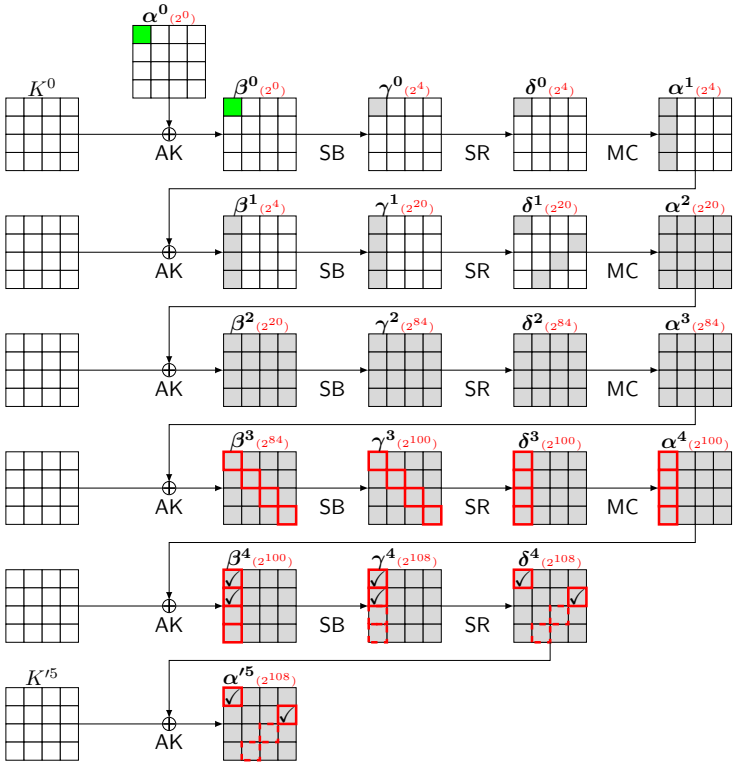


Fig. 16: The low-complexity 4.5-round IP transition for Joltik-BC-128. The low-data distinguishers differ only in the selection of the target nibbles (marked by the dashed line).

*(2) IP Key-Recovery Attacks*

- **The 8-round attack.** This is the nibble analogue of the low-data 8-round attack on Deoxys-BC-256: the 4-round IP transition is instantiated with the input $d$-difference $\boldsymbol{\alpha}^1$, where only the components $\alpha^1_{*,0}$ and $\alpha^1_{*,1}$ are non-zero; and the target nibble $\boldsymbol{\beta}^5_{*,0}$, for which $|\Gamma| \le 2^{120}$. The subkeys involved are

$$K_{\text{out}} = \{K'^6_0,\, K'^7_0,\, K'^7_7,\, K'^7_{10},\, K'^7_{13}\} \cup K'^8, \qquad |K_{\text{out}}| = 84,$$

and $K_{\text{in}} = \{K^0_0, K^0_5, K^0_{10}, K^0_{15}\}$ with $|K_{\text{in}}| = 16$, hence $|K_{\text{in}} \cup K_{\text{out}}| = 100$. Both the target-table and the candidate-table frameworks are applicable. The $\Gamma$-precomputing, the online sieve, and the exhaustive search proceed exactly as in Section IV-B; the only difference is that the plaintext-grouping table is built on the nibbles $P_0, P_5, P_{10}, P_{15}$ ($\nu = 16$, $\mathcal{DC} = 2^{16}$), as in Equation (9). Exactly one $(d+1)$-tuple in $\mathbb{P}$ realizes $\boldsymbol{\alpha}^1$ per $K_{\text{in}}$, by Theorem 2.

  - **Target-table variant.** Setting $d = 39$ leaves the sieve fraction $2^{120-39\cdot4} = 2^{-36}$. The dominant memory term is the distinguisher table, Only $\mathcal{DC} = 2^{\nu} = 2^{16}$ plaintexts are needed. $\mathcal{MC} = 2^{120} \cdot (39 \cdot 4 + 5 \cdot 120)/64 = 2^{123.56}$ blocks. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = (39+1) \cdot 2^{100} \cdot \frac{1+4+16}{16 \cdot 8} + 2^{100} \approx 2^{102.93},$$

  while the exhaustive search adds $2^{128-36} = 2^{92}$; the sieve dominates the total cost, $\mathcal{TC} \approx 2^{102.93}$ counts of 8-round encryptions.
  - **Candidate-table variant.** With $d = 34$, the sieve retains a fraction $2^{120-34\cdot4} = 2^{-16}$. Still $\mathcal{DC} = 2^{\nu} = 2^{16}$ plaintexts are needed. The memory is dominated by the candidate table $\overline{\Gamma}$ with $|\overline{\Gamma}| \le 2^{100}$, $\mathcal{MC} = 2^{100} \cdot (34 \cdot 4 + 5 \cdot 100 + 100)/64 \approx 2^{103.52}$ blocks. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = \left((34+1) \cdot 2^{100} \cdot \frac{1+4+16}{16 \cdot 8} + 2 \cdot 2^{100}\right) + \left((34+1) \cdot 2^{120} \cdot \frac{4}{16 \cdot 8} + 2^{120}\right) \approx 2^{121.07}$$

  and exhaustive key search accounts for a further $2^{128-16} = 2^{112}$; with the sieve as the leading term, the total time is $\mathcal{TC} \approx 2^{121.07}$ counts of 8-round encryptions.

### *E. AES-256*

The 4.5-round low-complexity and low-data distinguishers for AES-256 are identical to those of Deoxys-BC-256 described in Section IV-B; we do not repeat them here. However, due to the difference in key schedules, the complexities of the attacks differ.

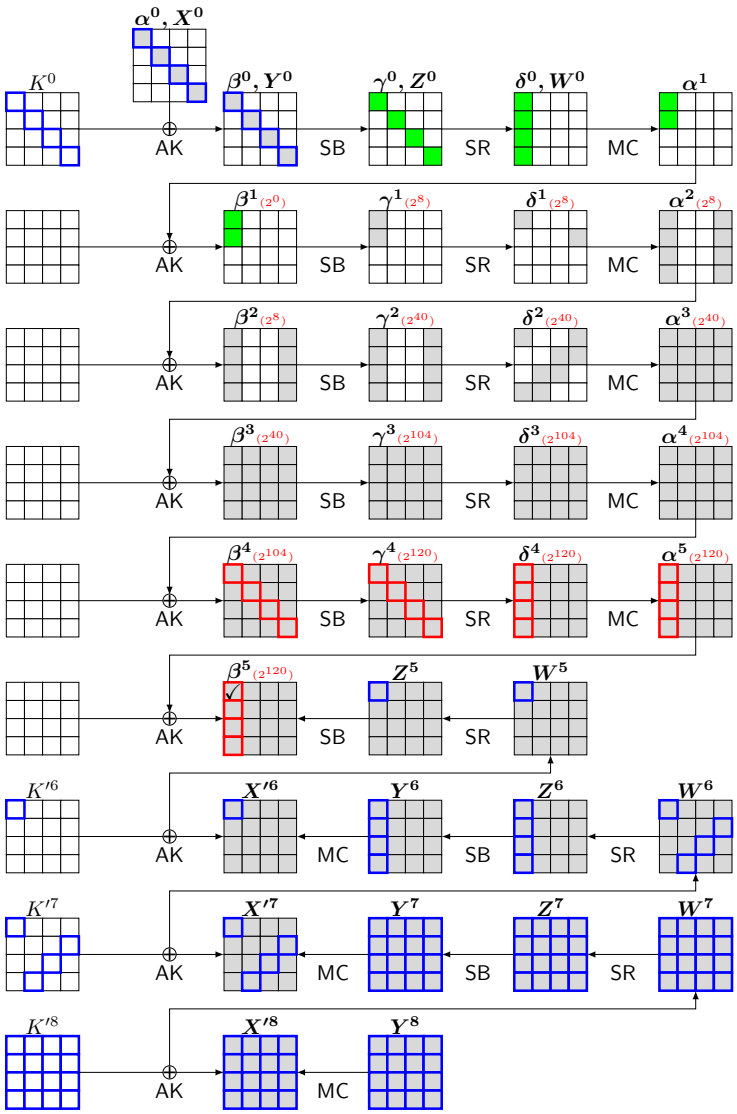


Fig. 17: The 8-round IP attack of Joltik-BC-128

- **Low-data 8-round attack.** We exploit the 4-round IP transition for key recovery to construct an 8-round IP attack on AES-256 by prepending one round ($r_b = 1$) and appending three rounds ($r_f = 3$), exactly as in Section IV-B except that the target byte is $\boldsymbol{\beta}^5_{*,4}$ rather than $\boldsymbol{\beta}^5_{*,0}$. The distinguisher is instantiated with the input $d$-difference $\boldsymbol{\alpha}^1$, where only the components $\alpha^1_{\ell,0}$ and $\alpha^1_{\ell,1}$ are non-zero and

$$\mathsf{MC}^{-1} \cdot (\alpha^1_{\ell,0}, \alpha^1_{\ell,1}, 0, 0)^T = (*, *, *, 0)^T \tag{10}$$

for $0 \le \ell \le d-1$; and the target byte $\boldsymbol{\beta}^5_{*,4}$, for which $|\Gamma| \le 2^{240}$.
The subkeys involved are

$$K_{\text{out}} = \left\{K'^6_4,\, K'^7_1,\, K'^7_4,\, K'^7_{11},\, K'^7_{14}\right\} \cup K'^8, \qquad |K_{\text{out}}| = 160,$$

and $K_{\text{in}} = \{K^0_0, K^0_5, K^0_{10}\}$ with $|K_{\text{in}}| = 24$.

The size of $K_{\text{out}}$ equals 160 bits rather than 168 bits, because the structure of the AES-256 key schedule yields the following key recovery property: Once all 16 bytes of the equivalent subkey $K'^8$ are fully known, the complete value of the corresponding subkey $K^8$ can be uniquely determined via MC operation; from the recovered full $K^8$, the four bytes $K^6_4, K^6_5, K^6_6, K^6_7$ can be derived by reversing two steps of the key expansion process; these four bytes further allow direct deduction of the byte $K'^6_4$ via MC operation. This property reduces the total number of key bits needed to be guessed to 184 bits ($|K_{\text{in}} \cup K_{\text{out}}| = 184$). The following plaintext-grouping step is shared by both the target-table and the candidate-table frameworks.

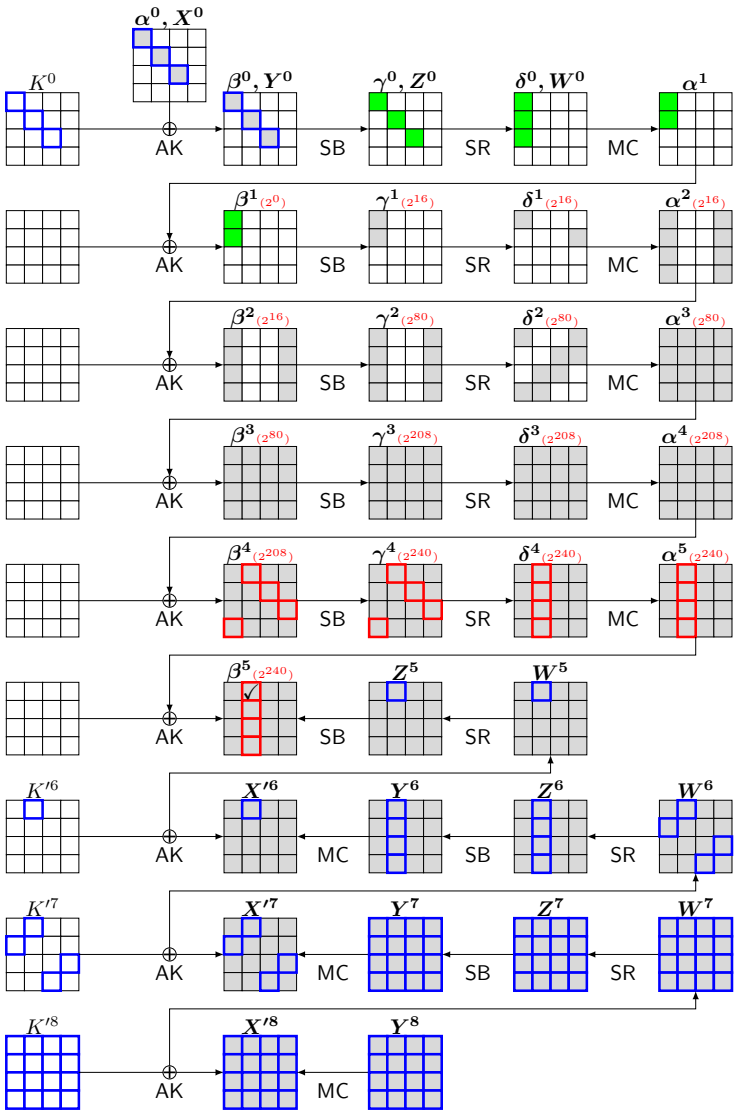


Fig. 18: The low-data 8-round IP attack of AES-256

*Plaintext-grouping.* Let $\Theta$ be the nonzero component set of $\boldsymbol{\alpha}^1$. Propagating each $\theta \in \Theta$ backward through one round yields $\nu = 24$ active plaintext bits, namely the bytes $P_0, P_5, P_{10}$; let $\mathbb{P}$ be the plaintext set that is free exactly on these bytes ($|\mathbb{P}| = 2^{24}$). Fix $P^0 \in \mathbb{P}$ and build the plaintext-grouping table as in Equation (7). Theorem 2 guarantees that each $K_{\text{in}}$ admits exactly one $(d+1)$-tuple in $\mathbb{P}$ realizing $\boldsymbol{\alpha}^1$.

- **Target-table variant.** Fixing $d = 40$ gives the sieve fraction $|\Gamma|/2^{d\eta} = 2^{240-40\cdot 8} = 2^{-80}$.
  1) $\Gamma$-*precomputing.* We build and store $\Gamma$ in $2^{240} \cdot (40 \cdot 8 + 5 \cdot 240)/128 \approx 2^{243.57}$ blocks.
  2) *Online sieve.* For each candidate $K_{\text{in}}$, look up the unique plaintext tuple $X_0^0 = P^0, X_1^0, \ldots, X_d^0$ from $\mathcal{P}_{K_{\text{in}}}$ that realizes $\boldsymbol{\alpha}^1$ via a key-indexed lookup, and obtain their ciphertexts $(Y_0^8, \ldots, Y_d^8)$. For each candidates $K_{\text{out}}$, partially decrypt the last 3 rounds of the ciphertexts, compute the candidate target $d$-difference $\boldsymbol{\beta}^5_{*,4}$, and discard the candidate unless it belongs to $\Gamma$.
  3) *Exhaustive search.* What the sieve leaves is $2^{184-80} = 2^{104}$ candidates for $K_{\text{in}} \cup K_{\text{out}}$; the remaining $|K| - |K_{\text{in}} \cup K_{\text{out}}| = 72$ key bits are then enumerated for each of them, recovering the full key.

  *Complexity.* Only $2^{24}$ plaintexts are needed, so $\mathcal{DC} = 2^{\nu} = 2^{24}$. The distinguisher table dominates the memory footprint, $\mathcal{MC} \approx 2^{243.57}$ blocks. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = (40+1) \cdot 2^{184} \cdot \frac{1+4+16}{16 \cdot 8} + 2^{184} \approx 2^{186.95},$$

  with the final enumeration of the key bits costing $2^{104-72} = 2^{176}$. Hence $\mathcal{TC} \approx 2^{186.95}$ counts of 8-round encryptions, the sieve being the dominant term.

- **Candidate-table variant.** $d$ is set to 33, which yields the sieve fraction $|\Gamma|/2^{d\eta} = 2^{240-33\cdot 8} = 2^{-24}$.
  1) *Online sieve.* Compute and store $\overline{\Gamma}$. For the input $d$-difference $\boldsymbol{\alpha}$, derive the possible target $d$-differences online, and query the candidate table $\overline{\Gamma}$ with $|\overline{\Gamma}| \leq 2^{184}$; a candidate $(K_{\text{in}}, K_{\text{out}})$ survives iff its stored $d$-difference is matched.
  2) *Exhaustive search.* $2^{|K_{\text{in}} \cup K_{\text{out}}|} \cdot 2^{t-d\eta} = 2^{184-24} = 2^{160}$ candidates for $K_{\text{in}} \cup K_{\text{out}}$ pass the sieve; a final enumeration of $|K| - |K_{\text{in}} \cup K_{\text{out}}| = 72$ key bits per survivor yields the full key.

  *Complexity.* The data requirement is $\mathcal{DC} = 2^{\nu} = 2^{24}$. The memory is dominated by the candidate table, which costs $\mathcal{MC} = 2^{184} \cdot (33 \cdot 8 + 5 \cdot 184 + 184)/128 \approx 2^{187.42}$ blocks. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = \left((33+1) \cdot 2^{184} \cdot \frac{1+4+16}{16 \cdot 8} + 2 \cdot 2^{184}\right) + \left((33+1) \cdot 2^{240} \cdot \frac{4}{16 \cdot 8} + 2^{240}\right) \approx 2^{241.04}$$

  and the final key search contributes $2^{160+72} = 2^{232}$. The sieve dominates the total cost, which is $\mathcal{TC} \approx 2^{241.04}$ counts of 8-round encryptions.

- **Low-complexity 8-round attack.** Similar to the low-complexity attack on Deoxys-BC-256, we exploit the 4-round IP transition with a single active input byte to mount an 8-round IP attack on AES-256. The distinguisher is instantiated with the input $d$-difference $\boldsymbol{\alpha}^1$, where only the components $\alpha^1_{\ell,0}$ are non-zero for $0 \leq \ell \leq d-1$, and the target byte is $\boldsymbol{\beta}^5_{*,4}$, with $|\Gamma| \leq 2^{200}$. The subkeys involved are:

$$K_{\text{out}} = \left\{K'^6_4,\, K'^7_1,\, K'^7_4,\, K'^7_{11},\, K'^7_{14}\right\} \cup K'^8, \qquad |K_{\text{out}}| = 160,$$

  and $K_{\text{in}} = \{K^0_0, K^0_5, K^0_{10}, K^0_{15}\}$ with $|K_{\text{in}}| = 32$. The same as in the low-data attack, $K'^6_4$ is deduced from $K'^8$ directly. Thus, $|K_{\text{in}} \cup K_{\text{out}}| = 192$.

  *Plaintext-grouping.* Let $\Theta$ be the nonzero component set of $\boldsymbol{\alpha}^1$. Propagating each $\theta \in \Theta$ backward through one round yields $\nu = 32$ active plaintext bits, namely the bytes $P_0, P_5, P_{10}, P_{15}$; let $\mathbb{P}$ be the plaintext set that is free exactly on these bytes ($|\mathbb{P}| = 2^{32}$). Fix $P^0 \in \mathbb{P}$ and build the plaintext-grouping table as in Equation (8).
  - **Target-table variant.** Setting $d = 33$ leaves the sieve fraction $|\Gamma|/2^{d\eta} = 2^{200-33\cdot 8} = 2^{-64}$.
    1) $\Gamma$-*precomputing.* $\Gamma$ is precomputed and stored in $2^{200} \cdot (33 \cdot 8 + 5 \cdot 200)/128 \approx 2^{203.30}$ blocks.
    2) *Online sieve.* For each candidate $K_{\text{in}}$, fetch the single plaintext tuple $X^0_0 = P^0, X^0_1, \ldots, X^0_d$ from $\mathcal{P}_{K_{\text{in}}}$ that realizes $\boldsymbol{\alpha}^1$ via a key-indexed lookup, and obtain their ciphertexts $(Y^{10}_0, \ldots, Y^{10}_d)$. For each candidate $K_{\text{out}}$, partially decrypt the last 3 rounds of the ciphertexts, compute the candidate target $d$-difference $\boldsymbol{\beta}^5_{*,4}$, and discard the candidate unless it belongs to $\Gamma$.
    3) *Exhaustive search.* The sieve leaves $2^{|K_{\text{in}} \cup K_{\text{out}}|} \cdot 2^{t-d\eta} = 2^{192-64} = 2^{128}$ candidates for $K_{\text{in}} \cup K_{\text{out}}$; enumerating the remaining $|K| - |K_{\text{in}} \cup K_{\text{out}}| = 64$ key bits for each survivor recovers the full key.

    *Complexity.* The attack uses $2^{32}$ plaintexts, i.e., $\mathcal{DC} = 2^{\nu} = 2^{32}$. Memory is governed by the distinguisher table, $\mathcal{MC} \approx 2^{203.30}$ blocks; the plaintext-grouping table costs $|\Theta| \cdot 2^{|K_{\text{in}}|} \cdot \nu/n$ blocks, which is negligible. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = (33+1) \cdot 2^{192} \cdot \frac{1+4+16}{16 \cdot 8} + 2^{192} \approx 2^{194.72},$$

    while the exhaustive search adds $2^{128+64} = 2^{192}$. The total time complexity is $\mathcal{TC} = 2^{194.72} + 2^{192} \approx 2^{194.92}$ counts of 8-round encryptions.
  - **Candidate-table variant.** With $d = 33$, the sieve retains a fraction $|\Gamma|/2^{d\eta} = 2^{200-33\cdot 8} = 2^{-64}$.
    1) *Online sieve.* Compute and store $\overline{\Gamma}$. For the input $d$-difference $\boldsymbol{\alpha}$, derive the possible target $d$-differences online, and query the candidate table $\overline{\Gamma}$ with $|\overline{\Gamma}| \leq 2^{192}$; a candidate $(K_{\text{in}}, K_{\text{out}})$ survives iff its stored $d$-difference is matched.
    2) *Exhaustive search.* $2^{|K_{\text{in}} \cup K_{\text{out}}|} \cdot 2^{t-d\eta} = 2^{192-64} = 2^{128}$ candidates for $K_{\text{in}} \cup K_{\text{out}}$ survive the sieve, and enumerating the remaining $|K| - |K_{\text{in}} \cup K_{\text{out}}| = 64$ key bits per survivor completes the key recovery.

    *Complexity.* Data complexity: $\mathcal{DC} = 2^{\nu} = 2^{32}$. The memory is dominated by the candidate table, which costs $\mathcal{MC} = 2^{192} \cdot (33 \cdot 8 + 5 \cdot 192 + 192)/128 \approx 2^{195.47}$ blocks. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = \left((33+1) \cdot 2^{192} \cdot \frac{1+4+16}{16 \cdot 8} + 2^{192}\right) + \left((33+1) \cdot 2^{200} \cdot \frac{4}{16 \cdot 8} + 2^{200}\right) \approx 2^{201.06}$$

    and exhaustive key search accounts for a further $2^{128+64} = 2^{192}$. Overall, the attack costs $\mathcal{TC} \approx 2^{201.06}$ counts of 8-round encryptions, dominated by the sieve.

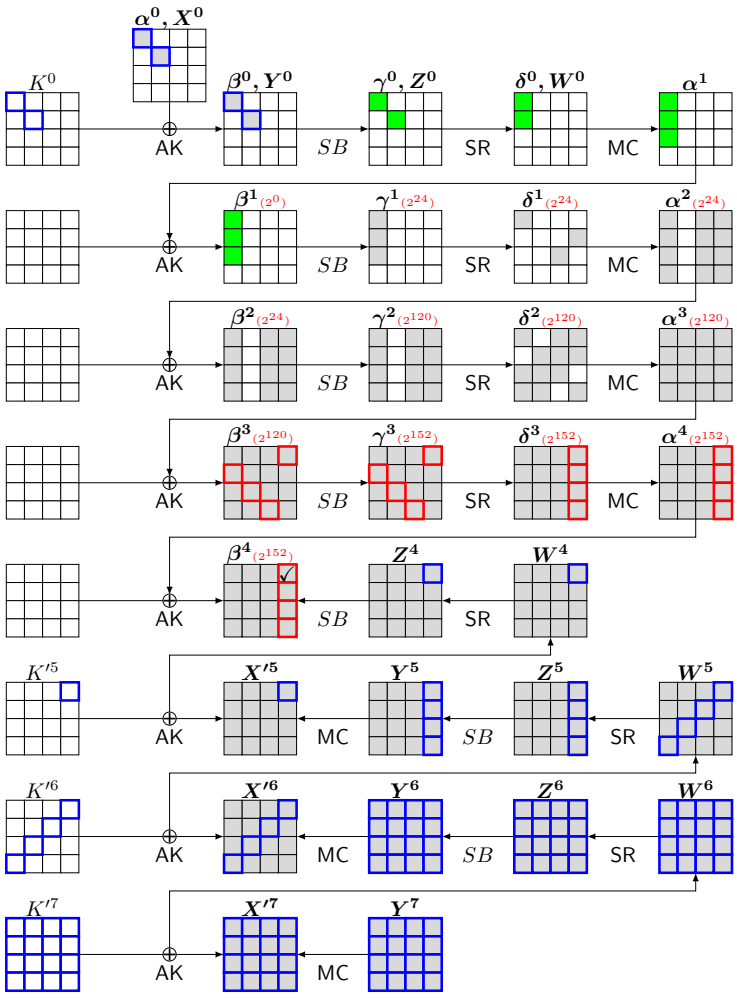


Fig. 19: The 7-round IP attack of AES-192

### F. AES-192

- **The 7-round attack.** As illustrated in Figure 19, we exploit a 3-round IP transition for key recovery to construct a 7-round IP attack on AES-192 by prepending one round ($r_b = 1$) and appending three rounds ($r_f = 3$). The distinguisher is instantiated with the input $d$-difference $\boldsymbol{\alpha}^1$, where only the components $\alpha^1_{\ell,0}$, $\alpha^1_{\ell,1}$ and $\alpha^1_{\ell,2}$ are non-zero and

$$\mathsf{MC}^{-1} \cdot (\alpha^1_{\ell,0}, \alpha^1_{\ell,1}, \alpha^1_{\ell,2}, 0)^T = (*, *, 0, 0)^T \tag{11}$$

for $0 \le \ell \le d-1$; and the target byte $\beta^4_{*,12}$, for which $|\Gamma| \le 2^{152}$. The subkeys involved are

$$K_{\text{out}} = \{K'^5_4,\ K'^6_1,\ K'^6_4,\ K'^6_{11},\ K'^6_{14}\} \cup K'^7, \qquad |K_{\text{out}}| = 168,$$

and $K_{\text{in}} = \{K^0_0, K^0_5\}$ with $|K_{\text{in}}| = 16$. Owing to the AES-192 key schedule, once all bytes of $K'^7$ are known, all bytes of $K^7$ are known, from which $K^6_0, K^6_1, \ldots, K^6_7$ and $K^5_{12}, \ldots, K^5_{15}$, and hence $K'^5_{12}$, $K'^6_3$, $K'^6_6$, are deduced; that is, only $|K_{\text{in}} \cup K_{\text{out}}| = 160$ key bits need to be guessed. Since $\log_2 |\Gamma| = 152 < 160 = |K_{\text{in}} \cup K_{\text{out}}|$, the candidate-table framework offers no advantage and we describe only the target-table variant.

*Plaintext-grouping.* Propagating the nonzero components of $\boldsymbol{\alpha}^1$ backward through one round yields $\nu = 16$ active plaintext bits, namely the bytes $P_0, P_5$; let $\mathbb{P}$ be the plaintext set that is free exactly on these bytes ($|\mathbb{P}| = 2^{16}$). Fix $P^0 \in \mathbb{P}$ and build the plaintext-grouping table:

$$\mathcal{P}_{K_{\text{in}}} = \left\{ P^1 \in \mathbb{P} \;\middle|\; \mathsf{MC} \cdot \begin{pmatrix} S(P^0_0 \oplus K^0_0) \oplus S(P^1_0 \oplus K^0_0) \\ S(P^0_5 \oplus K^0_5) \oplus S(P^1_5 \oplus K^0_5) \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} * \\ * \\ * \\ 0 \end{pmatrix} \right\}. \tag{12}$$

For each $K_{\text{in}}$, the lemma pins down exactly one such $(d+1)$-tuple in $\mathbb{P}$.

- **Target-table variant.** Fixing $d = 24$ gives the sieve fraction $2^{152-24\cdot 8} = 2^{-40}$.
  1) $\Gamma$-*precomputing.* Storing the precomputed $\Gamma$ takes $2^{152} \cdot (24 \cdot 8 + 5 \cdot 152)/128 = 2^{154.90}$ blocks.
  2) *Online sieve.* For each candidate $K_{\text{in}}$, retrieve the unique plaintext tuple $X_0^0 = P^0, X_1^0, \ldots, X_d^0$ from $\mathcal{P}_{K_{\text{in}}}$ that realizes $\boldsymbol{\alpha}^1$ via a key-indexed lookup, and obtain their ciphertexts $(Y_0^7, \ldots, Y_d^7)$. For each of the $2^{144}$ candidates $K_{\text{out}}$, partially decrypt the last 3 rounds of the ciphertexts, compute the candidate target $d$-difference $\boldsymbol{\beta}_{*,12}^4$, and discard the candidate unless it belongs to $\Gamma$.
  3) *Exhaustive search.* What the sieve leaves is $2^{160} \cdot 2^{t-d\eta} = 2^{160-40} = 2^{120}$ candidates for $K_{\text{in}} \cup K_{\text{out}}$; the remaining $|K| - |K_{\text{in}} \cup K_{\text{out}}| = 32$ key bits are then enumerated for each of them, recovering the full key.

  *Complexity.* Only $\mathcal{DC} = 2^{\nu} = 2^{16}$ plaintexts are needed. The dominant memory term is the distinguisher table, $\mathcal{MC} \approx 2^{154.90}$ blocks; the plaintext-grouping table costs $|\Theta| \cdot 2^{|K_{\text{in}}|} \cdot \nu/n$ blocks, which is negligible. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = (24+1) \cdot 2^{160} \cdot \frac{1+4+16}{16 \cdot 8} + 2^{160} \approx 2^{162.35},$$

  with the final enumeration of the key bits costing $2^{120+32} = 2^{152}$. Thus $\mathcal{TC} \approx 2^{162.35}$ counts of 7-round encryptions, with the sieve as the leading term.

### G. LED-128

Within one step of LED (containing four rounds), no key material is involved, so a full step can be regarded as a keyless super Sbox. A prescribed $d$-difference therefore has at most $2^{64}$ preimages that yield this $d$-difference; this is the key observation behind the *large-Sbox strategy*, which makes the preimages forming $\Gamma$ cheap to enumerate. The distinguishers below rest on this property.

- **Low-complexity 7.5-round distinguishers.** Similar to Joltik-BC, LED-128 utilizes a 4-bit Sbox. For constructing long-round distinguishers, the set of all possible $d$-differences is large; hence the output target nibble cannot be fixed to a single nibble, otherwise a valid distinguisher cannot be obtained. We therefore select the target cells corresponding to the two cells located on the anti-diagonal. Since no key material is involved within a single step, we adopt *the large-Sbox strategy* to construct the distinguishers. In total, activating a single nibble of the input $d$-difference and selecting the two target nibbles on an anti-diagonal of the output $d$-difference gives $16 \times 24 \times C_{15}^d$ IP transitions. Below is an example.

  **Distinguisher 9.** *Let $\boldsymbol{\alpha}^0$ denote the input $d$-difference, where only the components $\alpha_{\ell,0}^0$ are non-zero for $0 \le \ell \le d-1$. Let $\boldsymbol{\alpha}'^8$ denote the output $d$-difference without the final MC operation, and let $\Gamma$ be the set of all possible tuples $(\alpha_{0,0}'^8 \| \alpha_{0,13}'^8, \ldots, \alpha_{d-1,0}'^8 \| \alpha_{d-1,13}'^8)$. As Figure 16 shows, $|\Gamma| \le 2^{108}$. Setting $d = 14$ lets us distinguish the 8-round LED-128 from a random permutation with success probability $1 - 2^{108-14\cdot 8} = 1 - 2^{-4}$.*
  *$2^{108} \cdot (14+1) \cdot 2/(6 \cdot 16) + 2^{108} \cdot 2 \approx 2^{109.21}$ counts of 8-round encryptions, and the table is stored in $2^{108} \cdot (14 \cdot 8 + 5 \cdot 108)/64 \approx 2^{111.35}$ blocks.*
- **Low-data 7.5-round distinguishers.** When the target cells correspond to the four nibbles lying on the anti-diagonal, by adopting *the large-Sbox strategy* we obtain $16 \times 4 \times C_{15}^d$ low-data IP transitions in total. A concrete instance follows.

  **Distinguisher 10.** *Let $\boldsymbol{\alpha}^0$ denote the input $d$-difference, where only the components $\alpha_{\ell,0}^0$ are non-zero for $0 \le \ell \le d-1$. Let $\boldsymbol{\alpha}'^8$ denote the output $d$-difference excluding the final MC operation, and let $\Gamma$ be the set of all possible tuples*

$$\big(\alpha_{0,0}'^8 \| \alpha_{0,7}'^8 \| \alpha_{0,10}'^8 \| \alpha_{0,13}'^8, \ldots, \alpha_{d-1,0}'^8 \| \alpha_{d-1,7}'^8 \| \alpha_{d-1,10}'^8 \| \alpha_{d-1,13}'^8\big).$$

  *Then, $|\Gamma| \le 2^{116}$. With $d = 8$ we distinguish 8-round LED-128 from a random permutation with success probability $1 - 2^{116-8\cdot 16} = 1 - 2^{-12}$. Storage for the table is $2^{116} \cdot (8 \cdot 16 + 5 \cdot 116)/64 \approx 2^{119.47}$ blocks, while construction consumes $2^{116} \cdot (8+1) \cdot 4/(6 \cdot 16) + 2^{116} \cdot 2 \approx 2^{117.25}$ counts of 8-round encryptions.*

### H. ChiLow-32

In this part, we present our results for ChiLow-32. We first introduce the IP distinguisher under the CC setting, followed by the IP distinguisher and attack under the CP setting.

#### (1) IP Distinguishers in the CC setting

To construct longer distinguishers, we employ *the subtlety-aware strategy* to analyze the detailed properties of $\mathcal{X}_{32}$ as follows. Here and below, $x_{<<l}$ denotes the left shift of $x$ by $l$ bits.

**Property 1.** *Let $d \in \{10, 16\}$, and let $\boldsymbol{\alpha}$ be an input $d$-difference of $\mathcal{X}_{32}$ with $\boldsymbol{\alpha}_\ell = (\ell+1)_{<<26}$ for all $0 \le \ell \le d-1$. Then $\mathcal{X}_{32}$ yields exactly $2^6$ output $d$-differences when $d = 10$, and exactly $2^7$ when $d = 16$.*

Property 1 can be proven via exhaustive search.

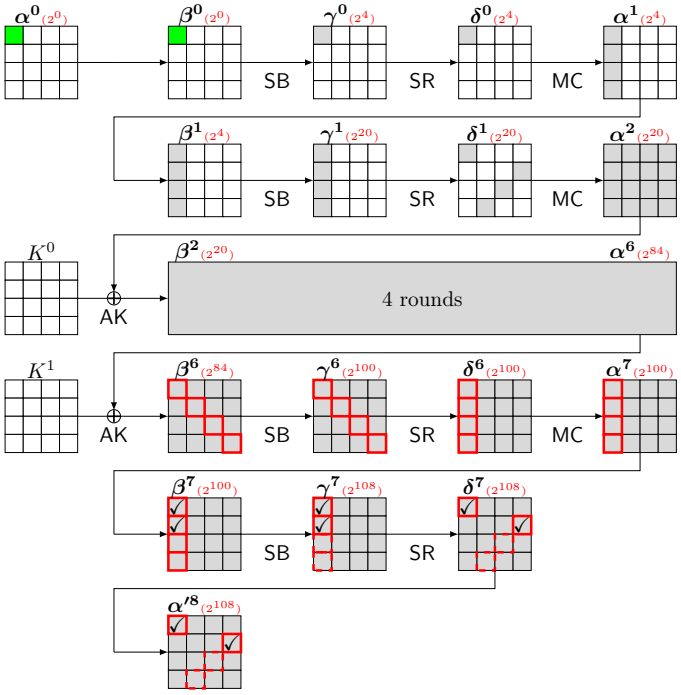


Fig. 20: The 7.5-round IP transition for LED-128. The low-data distinguishers differ only in the selection of the target nibbles (marked by the dashed line).

**Property 2.** *Let $\pi_{13}$ denote the projection of each difference-component onto its leftmost 13 bits. For every input d-difference $\boldsymbol{\alpha}$, the projected output d-differences $\pi_{13}(\boldsymbol{\beta})$ of $\mathcal{X}_{32}$ take at most $2^{15}$ distinct values.*

*Proof.* When $\chi_{15}$ is regarded as an Sbox, any input $d$-difference yields at most $2^{15}$ distinct output $d$-differences. Furthermore, the linear function $\lambda$ within $\mathcal{X}_{32}$ does not affect the leftmost 13 bits of the state. For any input $d$-difference the output $d$-differences restricted to the leftmost 13 bits therefore number at most $2^{15}$. □

- **The 4.5-round distinguisher of $\mathsf{D}_{32}$.** By adopting *the subtlety-aware strategy* and the property of the nonlinear layer, we present the following 4.5-round distinguisher for ChiLow-32.

**Distinguisher 11.** *Let $\boldsymbol{\alpha}^0$ denote the input 10-difference, where $\boldsymbol{\alpha}_\ell = (\ell+1)_{<<26}$ for $0 \le \ell \le 9$. Let $\boldsymbol{\beta}^4$ denote the output 10-difference, and let $\Gamma$ be the set of all possible output 10-differences targeting the leftmost 13 bits. As Figure 21 shows, $|\Gamma| \le 2^{117}$. We can distinguish the 5-round $D_{32}$ from a random permutation with success probability $1-2^{117-10\cdot 13} = 1-2^{-13}$.*
*We allocate $2^{117}\cdot(10\cdot 13+5\cdot 117)/32 \approx 2^{121.48}$ blocks to the table and pay $2^{117}\cdot(10+1)\cdot 1/(2\cdot 5)+2^{117}\cdot 2 \approx 2^{118.64}$ counts of 5-round encryptions for building it.*

*(2) IP Distinguishers in the CP setting*

**Property 3.** *For $(\mathcal{X}_{32})^{-1}$, there are 43264 possible output 4-differences for the input 4-difference $(1_{\ll 18}, 1_{\ll 17}, 3_{\ll 17}, 1_{\ll 16})$; and there are 907264 possible output 7-differences for the input 7-difference $(1_{\ll 18}, 1_{\ll 17}, 3_{\ll 17}, 1_{\ll 16}, 5_{\ll 16}, 3_{\ll 16}, 7_{\ll 16})$.*

To obtain all output $d$-differences corresponding to an input $d$-difference through $(\mathcal{X}_{32})^{-1}$, a straightforward approach is to treat $\mathcal{X}_{32}$ as a 32-bit Sbox, precompute its inverse $(\mathcal{X}_{32})^{-1}$, and then evaluate the $d$-differences. However, this approach incurs prohibitive memory overhead. Instead, we exploit the internal structure of $\mathcal{X}_{32}$ to devise a memory-efficient computation method.

As Figure 22 shows, an input $d$-difference $\boldsymbol{\psi}$ can propagate to an output difference $\boldsymbol{\varpi}$ through $(\mathcal{X}_{32})^{-1}$ if and only if there exist two $d$-differences $\boldsymbol{\psi}^0$ and $\boldsymbol{\psi}^1$ such that $\boldsymbol{\psi}^1 = \lambda(\boldsymbol{\varpi})$, $\boldsymbol{\psi}^0 = \boldsymbol{\psi} \oplus \boldsymbol{\psi}^1$, and the $d$-difference $\boldsymbol{\psi}^0$ can propagate to $\boldsymbol{\varpi}$ through $\big((\chi_{15})^{-1}, (\chi_{17})^{-1}\big)$. Here, $\boldsymbol{\psi} \oplus \boldsymbol{\psi}^1$ denotes the component-wise XOR of $\boldsymbol{\psi}$ and $\boldsymbol{\psi}^1$, and $\lambda(\boldsymbol{\varpi})$ denotes the application of $\lambda$ to each component of $\boldsymbol{\varpi}$. Since the function $\lambda$ affects bits $13, 14, 15, 16$ of the output of $\mathcal{X}_{32}$, we select the input $d$-difference such that $\boldsymbol{\psi}_\ell$ is active on at most bits $13, 14, 15, 16$. Accordingly, $\boldsymbol{\psi}^0_\ell$ is also active on at most bits $15, 16, 17, 18$. Using this property, we compute all possible output 4-differences for the input 4-difference $\boldsymbol{\psi} = (1_{\ll 18}, 1_{\ll 17}, 3_{\ll 17}, 1_{\ll 16})$.

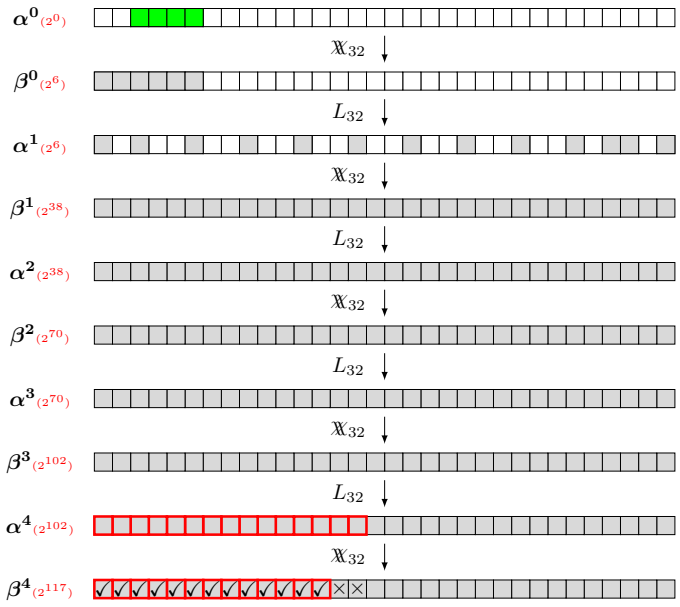


Fig. 21: The 4.5-round distinguisher of $\mathsf{D}_{32}$.

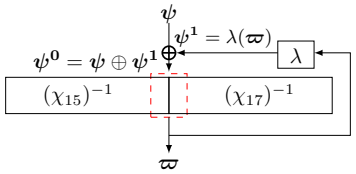


Fig. 22: Illustration of the $d$-difference propagation of $(\mathbb{X}_{32})^{-1}$

1) We treat $\chi_{15}$ as a 15-bit Sbox and precompute the truth table of $(\chi_{15})^{-1}$. We then iterate over all values of $(\psi^0_{0,13}, \psi^0_{0,14}, \psi^0_{1,13}, \psi^0_{1,14}, \psi^0_{2,13}, \psi^0_{2,14}, \psi^0_{3,13}, \psi^0_{3,14})$. Using $(\chi_{15})^{-1}$, we derive all possible tuples $(\varpi_{0,13}, \varpi_{0,14}, \varpi_{1,13}, \varpi_{1,14}, \varpi_{2,13}, \varpi_{2,14}, \varpi_{3,13}, \varpi_{3,14})$, where no constraints are imposed on $\varpi_{\ell,i}$ for $0 \le \ell \le 3$ and $0 \le i \le 12$.
2) We treat $\chi_{17}$ as a 17-bit Sbox and precompute the truth table of $(\chi_{17})^{-1}$. We then iterate over all values of $(\psi^0_{0,15}, \psi^0_{0,16}, \psi^0_{1,15}, \psi^0_{1,16}, \psi^0_{2,15}, \psi^0_{2,16}, \psi^0_{3,15}, \psi^0_{3,16})$. Using $(\chi_{17})^{-1}$, we derive all possible tuples $(\varpi_{0,15}, \varpi_{0,16}, \varpi_{1,15}, \varpi_{1,16}, \varpi_{2,15}, \varpi_{2,16}, \varpi_{3,15}, \varpi_{3,16})$, where no constraints are imposed on $\varpi_{\ell,i}$ for $0 \le \ell \le 3$ and $17 \le i \le 31$.
3) For all tuples $(\psi^0_{\ell,13}, \psi^0_{\ell,14}, \psi^0_{\ell,15}, \psi^0_{\ell,16}, \varpi_{\ell,13}, \varpi_{\ell,14}, \varpi_{\ell,15}, \varpi_{\ell,16})$ with $0 \le \ell \le 3$, we compute $\boldsymbol{\psi}^1$ and sieve out impossible values according to $\boldsymbol{\psi}$.
4) For the remaining tuples $(\psi^0_{\ell,13}, \psi^0_{\ell,14}, \psi^0_{\ell,15}, \psi^0_{\ell,16}, \varpi_{\ell,13}, \varpi_{\ell,14}, \varpi_{\ell,15}, \varpi_{\ell,16})$ with $0 \le \ell \le 3$, we collect all possible $\boldsymbol{\varpi}$, which are exactly the output $d$-differences.

This leaves 7 tuples $(\psi^0_{\ell,15}, \psi^0_{\ell,16}, \psi^0_{\ell,17}, \psi^0_{\ell,18}, \varpi_{\ell,15}, \varpi_{\ell,16}, \varpi_{\ell,17}, \varpi_{\ell,18})$ for $0 \le \ell \le 3$, and $43264 \approx 2^{15.40}$ values of $\boldsymbol{\varpi}$, which enables us to construct the following distinguishers.

- **The 4-round distinguisher of $\mathsf{D}_{32}$.** By adopting *the subtlety-aware strategy* and the property of the nonlinear layer, we present the following 4-round CP distinguisher for ChiLow-32.

**Distinguisher 12.** *Let $\boldsymbol{\alpha}^0$ denote the input 4-difference, where $\boldsymbol{\alpha} = (1_{\ll 18}, 1_{\ll 17}, 3_{\ll 17}, 1_{\ll 16})$. Let $\boldsymbol{\alpha}^4$ denote the output 4-difference. As Figure 23 shows, $|\Gamma| \le 2^{111.40}$. We can distinguish the 4-round $D_{32}$ from a random permutation with success probability $1-2^{111.40-4\cdot 32} = 1-2^{-16.6}$. Construction of the table costs $2^{111.40}\cdot(4+1)\cdot 1/(2\cdot 4)+2^{111.40}\cdot 2 \approx 2^{112.79}$ counts of 4-round encryptions and occupies $2^{111.40}\cdot(4\cdot 32+5\cdot 112)/32 \approx 2^{115.83}$ blocks.*

*(3) IP Key-Recovery Attacks in the CP setting*

We use the input 7-difference $\boldsymbol{\alpha}^0 = (1_{\ll 18}, 1_{\ll 17}, 3_{\ll 17}, 1_{\ll 16}, 5_{\ll 16}, 3_{\ll 16}, 7_{\ll 16})$ to mount the full-round attack. We first calculate all output 7-differences $\boldsymbol{\varpi}$ through $(\mathbb{X}_{32})^{-1}$ for the input 7-difference $\boldsymbol{\psi} = (1_{\ll 18}, 1_{\ll 17}, 3_{\ll 17}, 1_{\ll 16}, 5_{\ll 16}, 3_{\ll 16}, 7_{\ll 16})$ as follows.

1) Using the method for constructing Distinguisher 12, we obtain all tuples $(\psi^0_{\ell,13}, \psi^0_{\ell,14}, \psi^0_{\ell,15}, \psi^0_{\ell,16}, \varpi_{\ell,13}, \varpi_{\ell,14}, \varpi_{\ell,15}, \varpi_{\ell,16})$ with $0 \le \ell \le 3$ from the propagation of the 4-difference, and all tuples $(\psi^0_{\ell,13}, \psi^0_{\ell,14}, \psi^0_{\ell,15}, \psi^0_{\ell,16}, \varpi_{\ell,13}, \varpi_{\ell,14}, \varpi_{\ell,15}, \varpi_{\ell,16})$ with $4 \le \ell \le 6$ from the propagation of the 3-difference.

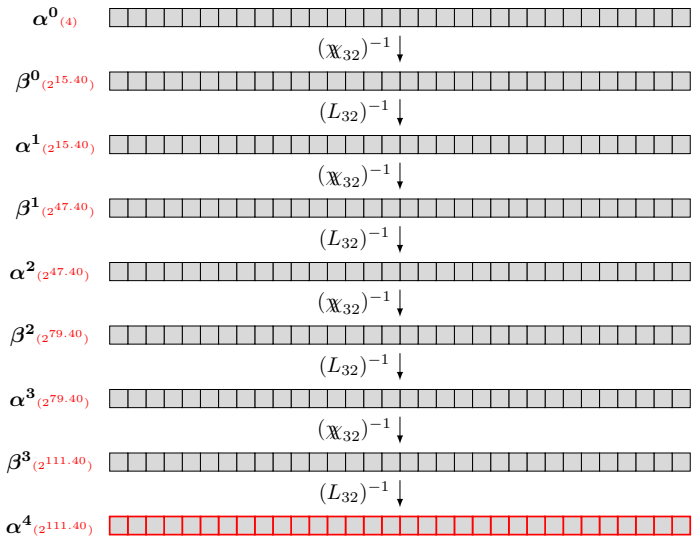


Fig. 23: The 4-round CP IP transition of ChiLow-32

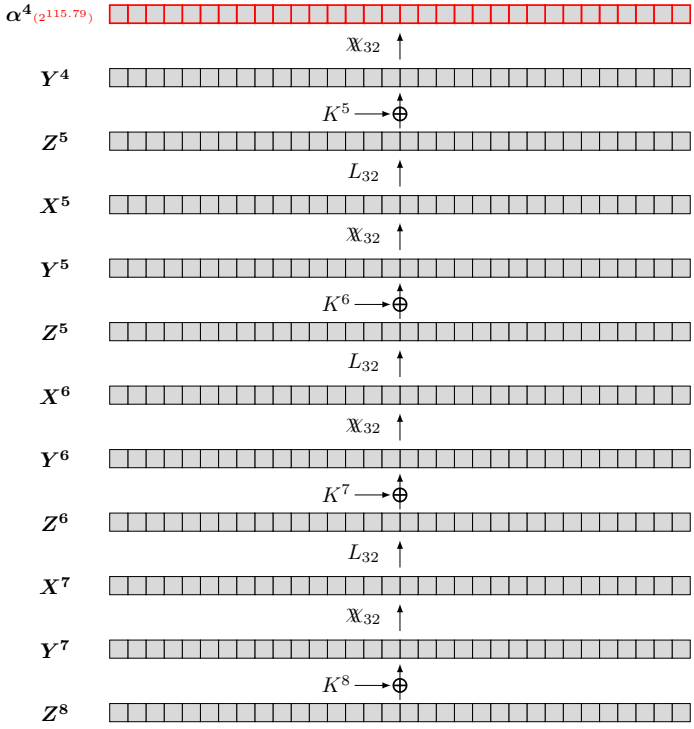


Fig. 24: The full-round CP IP attack of ChiLow-32

2) We merge these tuples to form the 7-difference, and sieve out invalid assignments under the constraint that the 7-difference propagates through $(\chi_{15})^{-1}$ and $(\chi_{17})^{-1}$.
3) For the remaining tuples $(\psi^0_{\ell,13}, \psi^0_{\ell,14}, \psi^0_{\ell,15}, \psi^0_{\ell,16}, \varpi_{\ell,13}, \varpi_{\ell,14}, \varpi_{\ell,15}, \varpi_{\ell,16})$ with $0 \le \ell \le 7$, we collect all feasible $\boldsymbol{\varpi}$, which are exactly the output 7-differences.

This leaves 896 tuples $(\psi^0_{\ell,15}, \psi^0_{\ell,16}, \psi^0_{\ell,17}, \psi^0_{\ell,18}, \varpi_{\ell,15}, \varpi_{\ell,16}, \varpi_{\ell,17}, \varpi_{\ell,18})$ for $0 \le \ell \le 6$, and $907264 \approx 2^{19.79}$ values of $\boldsymbol{\varpi}$. Thus, for the input 7-difference $\boldsymbol{\alpha}^0 = (1_{\ll 18}, 1_{\ll 17}, 3_{\ll 17}, 1_{\ll 16}, 5_{\ll 16}, 3_{\ll 16}, 7_{\ll 16})$, there are at most $2^{115.79}$ possible values of $\boldsymbol{\alpha}^4$.

- **The full-round attack.** We exploit the 4-round IP transition for key recovery to construct a full-round IP attack on ChiLow-32 by appending 4 rounds, as illustrated in Figure 24. To reduce the computational complexity, we adopt the partition-guess-filter technique. We first employ a greedy algorithm to determine the ordering of target bit positions. Specifically, we iteratively select target bit positions: for the $i$-th bit position, given that the first $i-1$ positions have

TABLE III: The guessed key bits for full-round attack on ChiLow-32 in CP setting

| Step | Bit Position | Guessed Key | Guessed Key Bits |
|---|---|---|---|
| 1 | $13,15,16,17,18,19,20,21,22,$ $23,24,25,26,27,28,29,30,31$ | $K_0^5, K_{13}^5, K_{14}^5, K_{15}^5, K_{16}^5, K_{17}^5, K_{18}^5, K_{19}^5, K_{20}^5, K_{21}^5, K_{22}^5,$ $K_{23}^5, K_{24}^5, K_{25}^5, K_{26}^5, K_{27}^5, K_{28}^5, K_{29}^5, K_{30}^5, K_{31}^5, K^6, K^7, K^8$ | 116 |
| 2 | $1,2,3,4,5,6,$ $7,8,9,10,11,12$ | $K_1^5, K_2^5, K_3^5, K_4^5, K_5^5, K_6^5,$ $K_7^5, K_8^5, K_9^5, K_{10}^5, K_{11}^5, K_{12}^5$ | 12 |

already been fixed, we choose the position that requires the minimal number of additional guessed key bits. Then, we use output bit positions $0,1,2,3,4,5,6,7,8,9,10,11,12,13,15,16,17,18$ to sieve keys, since they yield the minimal number of guessed key bits under this greedy strategy, and use the remaining bit positions to further recover keys. We choose $d=7$; the attack procedure is described as follows.

1) $\Gamma$-*precomputing.* For the input 7-difference, we construct table $\Gamma_0$ to store all feasible output $d$-differences of $\boldsymbol{\alpha}^4$ restricted to bit positions $0,1,2,3,4,5,6,7,8,9,10,11,12,13,15,16,17,18$, and construct table $\Gamma_1$ to store all feasible output $d$-differences of $\boldsymbol{\alpha}^4$ over all bit positions. For $\Gamma_0$, the memory complexity is $2^{115.79}\cdot(7\cdot18+5\cdot115.79)/32 = 2^{120.25}$ blocks. For $\Gamma_1$, the memory complexity is $2^{115.79}\cdot(7\cdot32+5\cdot115.79)/32 = 2^{120.44}$ blocks.
2) *Online sieve.* As Table III shows, we perform the key recovery in 2 steps.
   a) At Step 1, we guess 116 key bits, partially decrypt the 8 ciphertexts, and check whether the corresponding $d$-differences belong to $\Gamma_0$. After filtering, $2^{116}\times 2^{115.79-7\cdot18} = 2^{105.79}$ key candidates remain.
   b) At Step 2, we additionally guess 12 key bits, partially decrypt the 8 ciphertexts, and check whether the corresponding $d$-differences belong to $\Gamma_1$. In total, we enumerate $2^{105.79+12}$ key-bit assignments. After filtering, at most $2^{117.79}\times 2^{-7\cdot14} = 2^{19.79}$ key candidates remain, according to Eq. (3).
3) *Exhaustive search.* Since at most $2^{19.79}$ candidates of $K^8$ remain, we exhaustively search over $2^{19.79+96}$ master keys via the key schedule to recover the full key.

*Complexity.* The attack uses 8 chosen plaintexts, so $\mathcal{DC}=8$. The distinguisher table dominates the memory footprint, $\mathcal{MC} = 2^{120.25}+2^{120.44}\approx 2^{121.35}$ blocks. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = ((7+1)\cdot 2^{116}\cdot\frac{4}{8}+2^{115.79}) + ((7+1)\cdot 2^{117.79}\cdot\frac{4}{8}+2^{115.79}) \approx 2^{120.29},$$

and the final key search contributes $2^{115.79}$. The total time complexity is $\mathcal{TC} = 2^{120.29}+2^{115.79}\approx 2^{120.35}$ counts of full-round encryptions.

## V. The Known-Plaintext Distinguishers and Attacks

We first propose a KP attack on full-round ChiLow-32 that needs 107 plaintexts, below the 160 of the best known full-round KP attack [21]. We then construct low-data KP distinguishers for Deoxys-BC-384, Deoxys-BC-256, AES-256, and AES-192. The results are summarized in Table II.

In the known-plaintext setting, the adversary is given arbitrary plaintexts rather than chosen ones. In the chosen-plaintext setting the active cell positions of the input $d$-difference are fixed by the chosen structure; here the input $d$-difference is no longer constrained by any active-cell condition, so any $d+1$ plaintexts are admissible and the components of the resulting input $d$-difference may take arbitrary values. Consequently, the target output $d$-difference set cannot be precomputed offline as described in Section III and Section IV: the IP transition must be computed online from the input $d$-difference derived from the known plaintexts.

### A. ChiLow-32

#### (1) IP Distinguishers

- **The 3-round distinguishers.** A 3-round KP distinguisher is provided below.

**Distinguisher 13.** *Let $\boldsymbol{\alpha}^0$ denote the input $d$-difference induced by the $d+1$ plaintexts, whose components are not constrained and can be arbitrary values. Let $\boldsymbol{\alpha}^3$ denote the output $d$-difference, and let $\Gamma$ be the set of all possible tuples $(\alpha_0^3,\ldots,\alpha_{d-1}^3)$. As Figure 25 shows, $|\Gamma|\le 2^{96}$. With $d=4$ the 3-round ChiLow-32 is distinguished from a random permutation with success probability $1-2^{96-32\cdot4} = 1-2^{-32}$. Building the table requires $2^{96}\cdot(4\cdot32+5\cdot96)/32\approx 2^{100.25}$ blocks of storage and $2^{96}\cdot(4+1)\cdot1/3+2^{96}\cdot2\approx 2^{97.87}$ counts of 3-round encryption.*

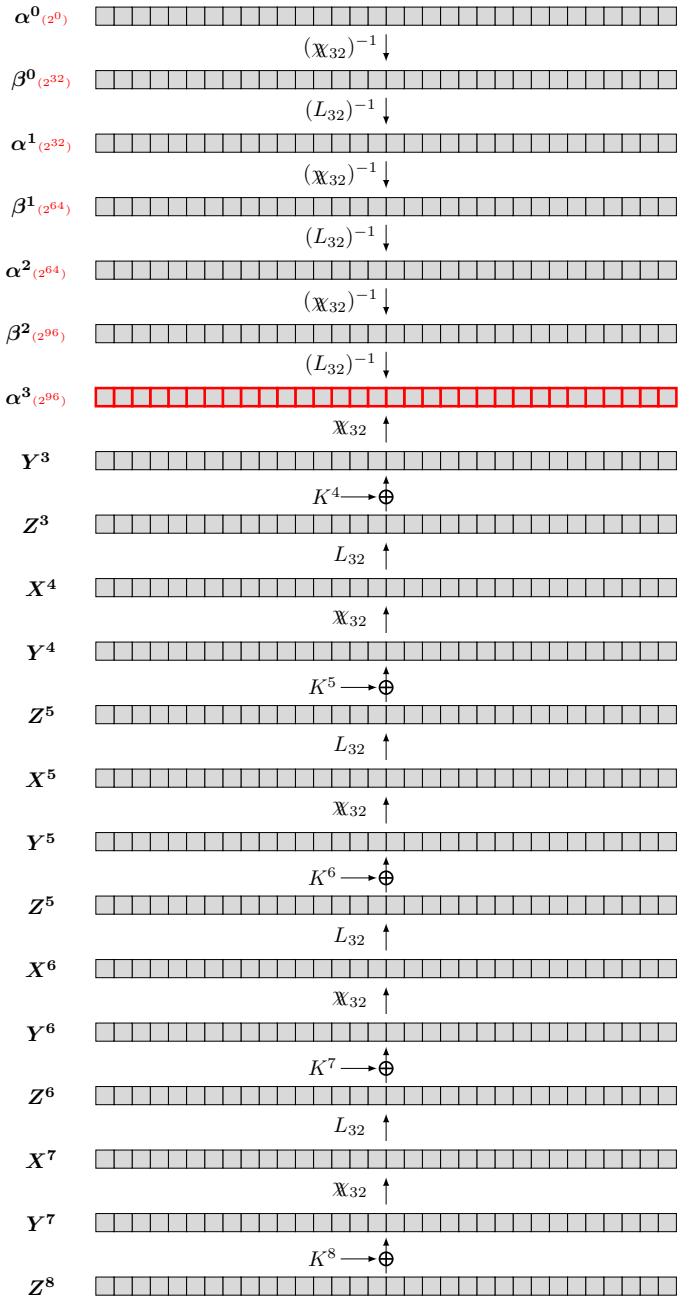


Fig. 25: Full-round KP IP attack on ChiLow-32

TABLE IV: The guessed key bits for full-round attack on ChiLow-32 in KP setting

| Step | Bit Position | Guessed Key | Guessed Key Bits |
|---|---|---|---|
| 1 | 15 | $K^4_{13}, K^4_{16}, K^4_{17}, K^5_0, K^5_1, K^5_2, K^5_4, K^5_5, K^5_6, K^5_7, K^5_8, K^5_9, K^5_{20}, K^5_{21}$ $K^5_{22}, K^5_{23}, K^5_{24}, K^5_{25}, K^5_{26}, K^5_{27}, K^5_{28}, K^5_{29}, K^5_{30}, K^6, K^7, K^8$ | 119 |
| 2 | 5 | $K^4_5, K^4_6, K^4_7, K^5_3, K^5_{11}, K^5_{12}, K^5_{13}, K^5_{15}, K^5_{18}, K^5_{19}$ | 10 |

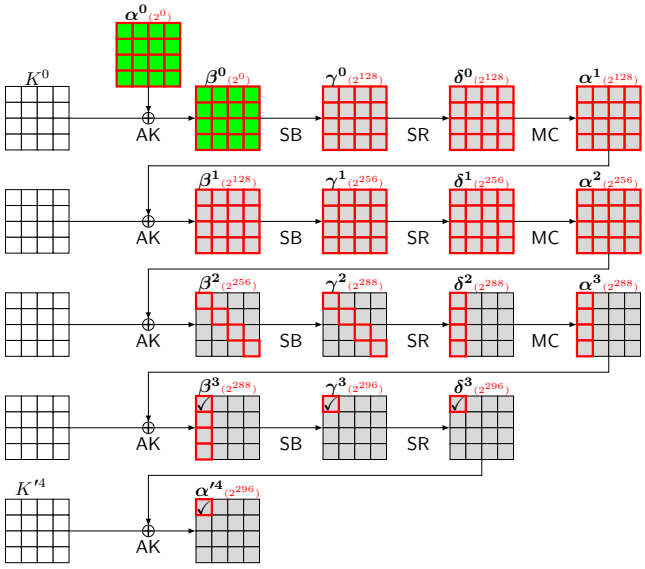


Fig. 26: The 3.5-round low-complexity KP IP transition for Deoxys-BC-384

*(2) IP Key-Recovery Attacks*

- **The full-round attack.** We exploit the 3-round IP transition for key recovery to construct a full-round IP attack on ChiLow-32 appending 5 rounds, shown in Figure 25. To reduce the computational complexity, we adopt the partition-guess-filter technique. We first employ a greedy algorithm to determine the ordering of target bit positions. Specifically, we iteratively select target bit positions: for the $i$-th bit position, given that the first $i-1$ positions have already been fixed, we choose the position that requires the minimal number of additional guessed key bits. The results are summarized in Table IV, and the output bit positions 15 and 5 are used to recover the key. We choose $d = 106$; the attack procedure is described as follows.

1) $\Gamma$-*computing.* For the input 106-difference, we construct table $\Gamma_0$ to store all possible output $d$-differences of $\boldsymbol{\alpha}^3$ restricted to bit position 15, and construct table $\Gamma_1$ to store all possible output $d$-differences of $\boldsymbol{\alpha}^3$ restricted to bit position 5. For $\Gamma_0$, the memory complexity is $2^{96} \cdot (106 \cdot 1 + 5 \cdot 96)/32 \approx 2^{100.19}$ blocks. For $\Gamma_1$, the memory complexity is $2^{96} \cdot (106 \cdot 2 + 5 \cdot 96)/32 \approx 2^{100.43}$ blocks. Constructing $\Gamma_0$ and $\Gamma_1$ costs $2 \cdot 2^{96}$ each.
2) *Online sieve.* As Table III shows, we perform the key recovery in 2 steps.
   a) At Step 1, we guess 119 key bits, partially decrypt the 106 ciphertexts, and check whether the corresponding $d$-differences belong to $\Gamma_0$. After filtering, $2^{119} \times 2^{96-106 \cdot 1} = 2^{109}$ key candidates remain.
   b) At Step 2, we additionally guess 10 key bits, partially decrypt the 106 ciphertexts, and check whether the corresponding $d$-differences belong to $\Gamma_1$. In total, we enumerate $2^{109+10}$ key candidates. After filtering, at most $2^{119} \times 2^{-106} = 2^{13}$ key candidates remain, according to Eq. (3).
3) *Exhaustive search.* Since at most $2^{13}$ candidates of $K^8$ remain, we exhaustively search over $2^{13+96}$ master keys via the key schedule to recover the full key.

*Complexity.* The attack relies on 107 known plaintexts, so $\mathcal{DC} = 107$. Memory is governed by the distinguisher table, $\mathcal{MC} = 2^{100.19} + 2^{100.43} \approx 2^{101.32}$ blocks. The sieve costs

$$\mathcal{TC}_{\text{sieve}} = 2 \cdot 2 \cdot 2^{96} + ((106+1) \cdot 2^{119} \cdot \frac{5}{8} + 2^{119}) + ((106+1) \cdot 2^{119} \cdot \frac{5}{8} + 2^{119}) \approx 2^{126.08},$$

while the exhaustive search adds $2^{109}$. Hence $\mathcal{TC} \approx 2^{126.08}$ counts of full-round encryptions, the sieve being the dominant term.

**Note.** ChiLow-32 supports 128-bit master keys. In this attack, we treat the round keys independently and exploit 129 bits of round-key material. Even so, our key recovery remains within the 128-bit security level, which demonstrates the power of the partition-guess-filter technique.

### *B. Deoxys-BC-384*

- **Low-complexity 3.5-round distinguishers.** Varying the positions of the output target byte gives 16 distinguishers in total. We give a concrete instance below.

**Distinguisher 14.** *Let $\boldsymbol{\alpha}^0$ denote the input $d$-difference induced by the $d+1$ plaintexts, whose components are not constrained and can be arbitrary values. Let $\boldsymbol{\alpha}'^4$ denote the output $d$-difference without the final MC operation, and let*

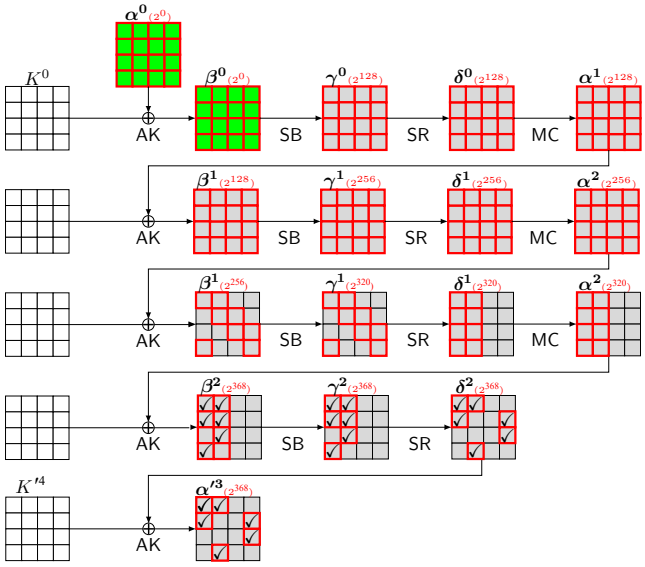


Fig. 27: The 3.5-round low-data KP IP transition for Deoxys-BC-384

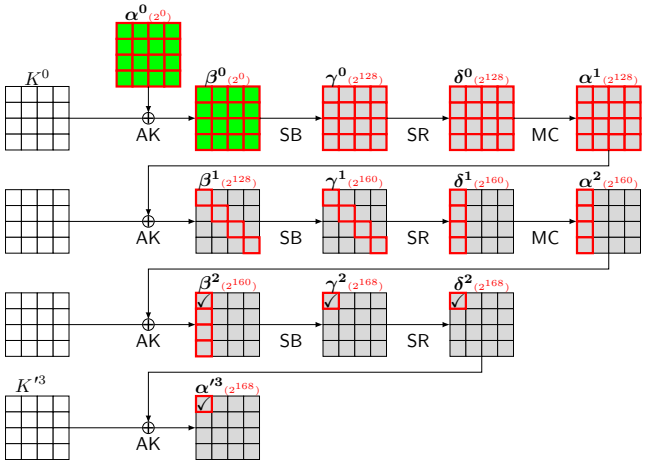


Fig. 28: The 2.5-round low-complexity KP IP transition for AES-256 and Deoxys-BC-256

$\Gamma$ *be the set of all possible tuples* $(\alpha'^4_{0,0},\ldots,\alpha'^4_{d-1,0})$. *As Figure 26 shows,* $|\Gamma| \le 2^{296}$. $d = 38$ *yields a distinguisher that tells apart the 4-round Deoxys-BC-384 from a random permutation with success probability* $1 - 2^{296-38\cdot 8} = 1 - 2^{-8}$. *The table occupies* $2^{296}\cdot(38\cdot 8+5\cdot 296)/128 \approx 2^{299.80}$ *blocks; its construction takes* $2^{296}\cdot(38+1)\cdot 1/(4\cdot 16)+2^{296}\cdot 2 \approx 2^{297.38}$ *counts of 4-round encryption.*

- **Low-data 3.5-round distinguishers.** As in the low-complexity case, the input $d$-difference can be an arbitrary value, and any $d+1$ plaintexts suffice. When the target cells correspond to six bytes located on two distinct anti-diagonals, with three bytes on each, we obtain 96 low-data distinguishers in total. One such distinguisher is displayed below.

  **Distinguisher 15.** *Let* $\boldsymbol{\alpha}^0$ *denote the input d-difference induced by the* $d+1$ *plaintexts; as above, its components can be arbitrary values. Let* $\boldsymbol{\alpha}'^4$ *denote the output d-difference excluding the final MC operation, and let* $\Gamma$ *be the set of all possible tuples*

$$\left(\alpha'^4_{0,0}\|\alpha'^4_{0,1}\|\alpha'^4_{0,4}\|\alpha'^4_{0,7}\|\alpha'^4_{0,13}\|\alpha'^4_{0,14},\ldots,\alpha'^4_{d-1,0}\|\alpha'^4_{d-1,1}\|\alpha'^4_{d-1,4}\|\alpha'^4_{d-1,7}\|\alpha'^4_{d-1,13}\|\alpha'^4_{d-1,14}\right).$$

*As Figure 27 shows, we can infer that* $|\Gamma| \le 2^{368}$. *Setting* $d = 8$, *i.e., any 9 plaintexts, lets us distinguish the 4-round Deoxys-BC-384 from a random permutation with success probability* $1 - 2^{368-8\cdot 48} = 1 - 2^{-16}$. $2^{368}\cdot(8+1)\cdot 6/(4\cdot 16)+2^{368}\cdot 2 \approx 2^{369.50}$ *counts of 4-round encryption, and the table is stored in* $2^{368}\cdot(8\cdot 48+5\cdot 368)/128 \approx 2^{372.12}$ *blocks.*

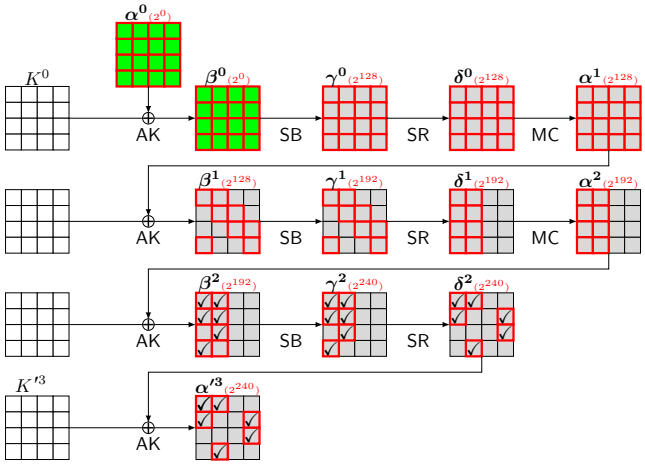


Fig. 29: The 2.5-round low-data KP IP transition for AES-256 and Deoxys-BC-256

### C. AES-192, AES-256 and Deoxys-BC-256

- **Low-complexity 2.5-round distinguishers.** Varying the positions of output target 1-bytes gives 16 distinguishers in total. Below is an example.

  **Distinguisher 16.** *Let $\boldsymbol{\alpha}^0$ denote the input $d$-difference corresponding to the $(d+1)$ plaintexts. Let $\boldsymbol{\alpha}'^3$ denote the output $d$-difference without the final MC operation, and let $\Gamma$ be the set of all possible tuples $(\alpha'^3_{0,0},\ldots,\alpha'^3_{d-1,0})$. As Figure 28 shows, $|\Gamma| \le 2^{168}$. Setting $d = 22$ lets us distinguish 3-round AES-192, AES-256 and Deoxys-BC-256 from a random permutation with success probability $1 - 2^{168-22\cdot 8} = 1 - 2^{-8}$.*
  *Storage for the table is $2^{168}\cdot(22\cdot 8 + 5\cdot 168)/128 \approx 2^{170.99}$ blocks, while construction consumes $2^{168}\cdot(22+1)\cdot 1/(3\cdot 16) + 2^{168}\cdot 2 \approx 2^{169.31}$ counts of 3-round encryptions.*

- **Low-data 2.5-round distinguishers.** For AES-256 and Deoxys-BC-256, target cells corresponding to six bytes on two anti-diagonals (three each) yield 96 low-data distinguishers. A concrete instance follows.

  **Distinguisher 17.** *Let $\boldsymbol{\alpha}^0$ denote the input $d$-difference corresponding to the $(d+1)$ plaintexts. Let $\boldsymbol{\alpha}'^3$ denote the output $d$-difference excluding the final MC operation, and let $\Gamma$ be the set of all possible tuples*

  $$(\alpha'^3_{0,0}\|\alpha'^3_{0,1}\|\alpha'^3_{0,4}\|\alpha'^3_{0,7}\|\alpha'^3_{0,13}\|\alpha'^3_{0,14},\ldots,\alpha'^3_{d-1,0}\|\alpha'^3_{d-1,1}\|\alpha'^3_{d-1,4}\|\alpha'^3_{d-1,7}\|\alpha'^3_{d-1,13}\|\alpha'^3_{d-1,14}).$$

  *Then, as shown in Figure 29, $|\Gamma| \le 2^{240}$. $d = 6$—that is, 7 plaintexts—yields a distinguisher telling apart 3-round AES-256 and Deoxys-BC-256 from a random permutation with success probability $1 - 2^{240-6\cdot 48} = 1 - 2^{-48}$.*
  *We allocate $2^{240}\cdot(6\cdot 48 + 5\cdot 240)/128 \approx 2^{243.54}$ blocks to the table and pay $2^{240}\cdot(6+1)\cdot 6/(3\cdot 16) + 2^{240}\cdot 2 \approx 2^{241.52}$ counts of 3-round encryptions for building it.*

  For AES-192, the target cells corresponding to the three bytes lying on the anti-diagonal yield 12 low-data distinguishers. We give a concrete instance below.

  **Distinguisher 18.** *Let $\boldsymbol{\alpha}^0$ denote the input $d$-difference corresponding to the $(d+1)$ plaintexts. Let $\boldsymbol{\alpha}'^3$ denote the output $d$-difference excluding the final MC operation, and let $\Gamma'$ be the set of all possible tuples*

  $$(\alpha'^3_{0,0}\|\alpha'^3_{0,7}\|\alpha'^3_{0,10},\ldots,\alpha'^3_{d-1,0}\|\alpha'^3_{d-1,7}\|\alpha'^3_{d-1,10}).$$

  *Thus, $|\Gamma'| \le 2^{184}$. Setting $d = 8$, i.e., any 9 plaintexts, lets us distinguish 3-round AES-192 from a random permutation with success probability $1 - 2^{184-8\cdot 24} = 1 - 2^{-8}$.*
  *Construction of the table costs $2^{184}\cdot(8+1)\cdot 3/(3\cdot 16) + 2^{184}\cdot 2 \approx 2^{185.37}$ counts of 3-round encryptions and occupies $2^{184}\cdot(8\cdot 24 + 5\cdot 184)/128 = 2^{187.12}$ blocks.*

## VI. Conclusion

This paper studied the impossible polytopic attack (IP attack) for block ciphers whose key size exceeds their block size. We formalized the bound-estimation-based IP transition underlying Tiessen's original work and proposed a general framework for its construction, together with two strategies—the large-Sbox strategy and the subtlety-aware strategy—that reduce the construction cost. We further introduced two dedicated key-recovery techniques: the plaintext-grouping technique extends key recovery to both sides of the distinguisher and thus enables attacks on more rounds, while the partition-guess-filter technique allows the entire master key to be guessed during the key-recovery phase and substantially lowers the attack complexity. Combining these techniques yields the first key-recovery framework for IP

attack, in two variants, the target-table and the candidate-table frameworks. As applications, new distinguishers and attacks are derived for AES, Deoxys-BC, Joltik-BC, LED-128, and ChiLow-32 in the single-key setting. In particular, the 10-round attacks on Deoxys-BC-384 and Joltik-BC-192 surpass the best known impossible differential attacks by one round, and two full-round IP attacks on ChiLow-32 in the CP setting and KP setting require only 8 and 107 plaintexts, lower than the 64 and 160 needed by the best known full-round attack, respectively. The results support treating the IP attack as a tool of its own rather than a variant of ID cryptanalysis: it builds longer distinguishers, admits a complete key-recovery procedure on both sides of the distinguisher, and operates in a low-data regime in which ID attacks are typically far more expensive in data. The two frameworks sit at opposite ends of the memory-time trade-off. The uniform cost accounting adopted throughout keeps the reported complexities comparable, term by term, with those of other attack families.

## Conflict of Interest

None of the authors have a conflict of interest to disclose.

## References


[1] J. Daemen and V. Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*, ser. Information Security and Cryptography. Springer, 2002. I, II-B1

[2] J. Jean, I. Nikolić, T. Peyrin, and Y. Seurin, "Deoxys v1.41," CAESAR Competition submission, Round 3, oct 2016. [Online]. Available: https://competitions.cr.yp.to/round3/deoxysv141.pdf I, II-B2

[3] Y. Belkheyar, P. Derbez, S. Ghosh, G. Leander, S. Mella, L. Perrin, S. Rasoolzadeh, L. Stennes, S. Sun, G. Van Assche, and D. Vizár, "ChiLow and ChiChi: New constructions for code encryption," in *Advances in Cryptology – EUROCRYPT 2025*, ser. LNCS, S. Fehr and P. Fouque, Eds. Springer, 2025, pp. 212–243. I, I, II, II-B5

[4] Institute of Commercial Cryptography Standards, China, "Call for Comments on Submission Requirements for Block Cipher Algorithms (Draft) and Evaluation Criteria for Block Cipher Algorithms," https://www.niccs.org.cn/niccs/Notice/pc/content/content_2095703042235641856.html, 2026, issued on September 4, 2026. I

[5] T. Tiessen, "Polytopic cryptanalysis," in *Advances in Cryptology – EUROCRYPT 2016*, ser. LNCS, M. Fischlin and J. Coron, Eds. Springer, 2016, pp. 214–239. I, I, 1, 2, 1, II-A, II-A, III-A, III-A, III-A, 1, III-B, 2

[6] E. Biham, A. Biryukov, and A. Shamir, "Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials," in *Advances in Cryptology – EUROCRYPT '99*, ser. LNCS, J. Stern, Ed., vol. 1592. Springer, 1999, pp. 12–23. I

[7] L. Grassi, C. Rechberger, and S. Rønjom, "A new structural-differential property of 5-round AES," in *Advances in Cryptology – EUROCRYPT 2017*, ser. LNCS, vol. 10211. Springer, 2017, pp. 289–317. I

[8] S. Rønjom, N. G. Bardeh, and T. Helleseth, "Yoyo tricks with AES," in *Advances in Cryptology – ASIACRYPT 2017*, ser. LNCS, vol. 10624. Springer, 2017, pp. 217–243. I

[9] N. G. Bardeh and S. Rønjom, "The exchange attack: How to distinguish six rounds of AES with chosen plaintexts," in *Advances in Cryptology – ASIACRYPT 2019*, ser. LNCS, vol. 11923. Springer, 2019, pp. 347–370. I

[10] X. Hu, Y. Li, L. Jiao, S. Tian, and M. Wang, "Mind the propagation of states - new automatic search tool for impossible differentials and impossible polytopic transitions," in *Advances in Cryptology – ASIACRYPT 2020*, ser. LNCS, S. Moriai and H. Wang, Eds., vol. 12491. Springer, 2020, pp. 415–445. I

[11] X. Hu, L. Jiao, Y. Li, S. Tian, Z. Liu, M. Wang, and D. Feng, "Revisit the propagation of states: New construction theory and search method for impossible differentials and impossible polytopic transitions," *IEEE Transactions on Information Theory*, vol. 72, no. 1, pp. 742–764, 2026. I

[12] H. Demirci and A. A. Selçuk, "A meet-in-the-middle attack on 8-round AES," in *Fast Software Encryption – FSE 2008*, ser. LNCS, vol. 5086, 2008, pp. 116–126. (iv)

[13] H. Hadipour, S. Sadeghi, and M. Eichlseder, "Finding the impossible: Automated search for full impossible-differential, zero-correlation, and integral attacks," in *Advances in Cryptology – EUROCRYPT 2023*, ser. LNCS, C. Hazay and M. Stam, Eds., vol. 14007. Springer, 2023, pp. 128–157. I

[14] E. Biham and N. Keller, "Cryptanalysis of reduced variants of Rijndael," Official Public Comment for Round 2 of the Advanced Encryption Standard Development Effort, 2000, available at https://csrc.nist.gov/encryption/aes/round2/conf3/papers/35-ebiham.pdf. I

[15] G. Leurent and C. Pernot, "New representations of the AES key schedule," in *Advances in Cryptology – EUROCRYPT 2021*, ser. LNCS, vol. 12696, 2021, pp. 54–84. I

[16] S. Peng, A. Khalesi, Z. Ahmadian, H. Hadipour, J. He, K. Hu, Z. Niu, S. Rasoolzadeh, and M. Wang, "Cube and integral attacks on ChiLow-32," *IACR Transactions on Symmetric Cryptology*, vol. 2026, no. 1, pp. 253–292, 2026. I, IV

[17] A. Bar-On, O. Dunkelman, N. Keller, E. Ronen, and A. Shamir, "Improved key recovery attacks on reduced-round AES with practical data and memory complexities," in *Advances in Cryptology – CRYPTO 2018*, ser. LNCS, H. Shacham and A. Boldyreva, Eds., vol. 10992. Springer, 2018, pp. 185–212. I

[18] P. Derbez, P. Fouque, and J. Jean, "Improved key recovery attacks on reduced-round AES in the single-key setting," in *Advances in Cryptology – EUROCRYPT 2013*, ser. LNCS, T. Johansson and P. Q. Nguyen, Eds., vol. 7881. Springer, 2013, pp. 371–387. I
[19] O. Dunkelman, N. Keller, and A. Shamir, "Improved single-key attacks on 8-round AES-192 and AES-256," in *Advances in Cryptology – ASIACRYPT 2010*, ser. LNCS, M. Abe, Ed., vol. 6477. Springer, 2010, pp. 158–176. I
[20] F. Yang and T. Tian, "Revisiting mixture-differential cryptanalysis on AES: new techniques and results," *Designs, Codes and Cryptography*, vol. 94, no. 7, p. 147, 2026. I
[21] E. Lambooij, P. Neumann, M. Verbauwhede, S. Wang, and T. Zhang, "Meet-in-the-middle attacks on full ChiLow," in *Advances in Cryptology – CRYPTO 2026*, ser. LNCS, N. Heninger and M. Rosulek, Eds., vol. 16805. Springer, 2026, pp. 353–373. I, I, II, V
[22] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, "Joltik v1.1," Submission to the CAESAR Competition, 2015. I, II-B3
[23] J. Guo, T. Peyrin, A. Poschmann, and M. Robshaw, "LED: A light block cipher," in *Cryptographic Hardware and Embedded Systems – CHES 2011*, ser. LNCS, vol. 6917. Springer, 2011, pp. 326–341. I, II-B4
[24] J. Jean, I. Nikolić, and T. Peyrin, "Tweaks and keys for block ciphers: the TWEAKEY framework," in *Advances in Cryptology – ASIACRYPT 2014*, ser. LNCS, vol. 8874. Springer, 2014, pp. 274–288. II-B2
[25] M. L. Fredman, J. Komlós, and E. Szemerédi, "Storing a sparse table with $O(1)$ worst case access time," *Journal of the ACM*, vol. 31, no. 3, pp. 538–544, 1984. II-C

## Appendix

### A. The impossible differential attacks on Deoxys-BC-384 and Joltik-BC-192

As illustrated in Figure 30, the 9-round attack of Deoxys-BC-384 is constructed by prepending 3 rounds and appending 2 rounds to the 4-round impossible differential. The detailed attack steps are given below.

1) Consider a plaintext structure where only the 15th byte is inactive. By preparing $q$ such structures, we obtain $q \cdot 2^{239}$ plaintext–ciphertext pairs.
2) Perform partial encryption of the first round.
   a) Guess $K_0^0$, $K_5^0$, $K_{10}^0$, $K_{15}^0$, verify the differences on $\alpha_0^1$, $\alpha_3^1$, and discard invalid plaintext-ciphertext pairs.
   b) For the remaining plaintext-ciphertext pairs, guess $K_1^0$, $K_6^0$, $K_{11}^0$, $K_{12}^0$, verify the differences on $\alpha_{14}^1$, $\alpha_{15}^1$, and discard invalid plaintext-ciphertext pairs.
   c) For the surviving plaintext-ciphertext pairs, guess $K_2^0$, $K_7^0$, $K_8^0$, $K_{13}^0$, verify the differences on $\alpha_9^1$, $\alpha_{10}^1$, and discard invalid plaintext-ciphertext pairs.
   d) For each surviving plaintext-ciphertext pair, guess $K_3^0$, $K_4^0$, $K_9^0$, $K_{14}^0$, verify the differences on $\alpha_4^1$, $\alpha_5^1$, and discard invalid plaintext-ciphertext pairs.

   The time complexity of this step is $q2^{239} \cdot 2^{-6\cdot 8} \cdot 2^{16\cdot 8} \cdot (2 \cdot 4)/(9 \cdot 16) = \frac{1}{9} q2^{318}$ 9-round encryptions; the surviving number of plaintext–ciphertext pairs is $q2^{239} \cdot 2^{-8\cdot 8} = q2^{175}$.
3) Perform partial encryption of the second round.
   a) For the surviving plaintext-ciphertext pairs, guess $K_1^1$, $K_6^1$, $K_{11}^1$, $K_{12}^1$, verify the differences on $\alpha_{13}^2$, $\alpha_{14}^2$, $\alpha_{15}^2$, and discard invalid plaintext-ciphertext pairs.
   b) For each surviving plaintext-ciphertext pair, guess $K_2^1$, $K_7^1$, $K_8^1$, $K_{13}^1$, verify the differences on $\alpha_8^1$, $\alpha_9^1$, $\alpha_{10}^1$, and discard invalid plaintext-ciphertext pairs.

   The time complexity of this step is $q2^{175} \cdot 2^{-3\cdot 8} \cdot 2^{24\cdot 8} \cdot (2 \cdot 4)/(9 \cdot 16) = \frac{1}{9} q2^{343}$ 9-round encryptions; this leaves $q2^{175} \cdot 2^{-6\cdot 8} = q2^{127}$.
4) Perform partial encryption of the third round. For the surviving plaintext-ciphertext pairs, guess $K_{11}^2$, $K_{12}^2$, verify the differences on $\alpha_{14}^3$, and discard invalid plaintext-ciphertext pairs. The time complexity of this step is $q2^{127} \cdot 2^{26\cdot 8} \cdot (2 \cdot 2)/(9 \cdot 16) = \frac{1}{9} q2^{333}$ 9-round encryptions; the remaining plaintext–ciphertext pairs number $q2^{127} \cdot 2^{-8} = q2^{119}$.
5) Perform partial decryption of the last round. Note that, since $K_2^0$, $K_{13}^1$ and $K_{12}^2$ are known, $K_{13}^9$ is known according to the tweakey schedule.
   a) For the remaining plaintext-ciphertext pairs, guess $K_1'^9, K_4'^9, K_{11}'^9$ and $K_{14}'^9$, verify the differences on $\alpha_5'^8, \alpha_6'^8, \alpha_7'^8$, and discard invalid plaintext-ciphertext pairs.
   b) For each surviving plaintext-ciphertext pair, guess $K_2'^9, K_5'^9, K_8'^9$ and $K_{15}'^9$, verify the differences on $\alpha_8'^8, \alpha_9'^8, \alpha_{10}'^8$, and discard invalid plaintext-ciphertext pairs.
   c) For the surviving plaintext-ciphertext pairs, guess $K_3'^9, K_6'^9, K_9'^9$ and $K_{12}'^9$, verify the differences on $\alpha_{12}'^8, \alpha_{13}'^8, \alpha_{15}'^8$, and discard invalid plaintext-ciphertext pairs.

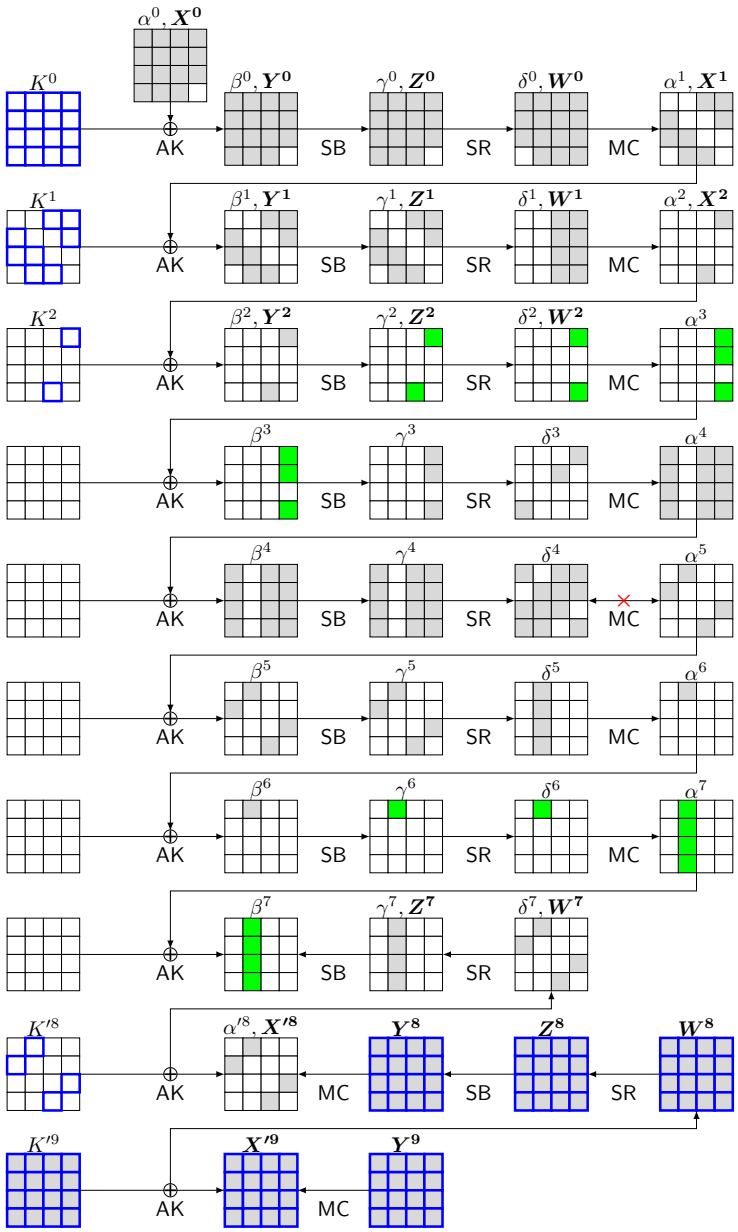


Fig. 30: The 9-round impossible differential attack of Deoxys-BC-384

d) Since

$$\begin{pmatrix} K'^9_{12} \\ K'^9_{13} \\ K'^9_{14} \\ K'^9_{15} \end{pmatrix} = M^{-1} \cdot \begin{pmatrix} K^9_{12} \\ K^9_{13} \\ K^9_{14} \\ K^9_{15} \end{pmatrix}$$

and $K'^9_{12}, K'^9_{14}, K'^9_{15}$ and $K^9_{13}$ are known, $K'^9_{13}$ can be deduced, where $M$ is the MDS matrix of AES. For each surviving plaintext-ciphertext pair, guess $K'^9_0, K'^9_7$ and $K'^9_{10}$, verify the differences on $\alpha'^8_0, \alpha'^8_2, \alpha'^8_3$, and discard invalid plaintext-ciphertext pairs.

The time complexity of this step is $q2^{119}2^{-9\cdot 8} \cdot 2^{41\cdot 8} \cdot (2 \cdot 4)/(9 \cdot 16) = \frac{1}{9}q2^{374}$ 9-round encryptions; the surviving number of plaintext–ciphertext pairs is $q2^{119} \cdot 2^{-12\cdot 8} = q2^{23}$.

6) Perform partial decryption of the second-last round. Once $\gamma^6_4$ is enumerated, the values of $\beta^7_4, \beta^7_5, \beta^7_6, \beta^7_7$ are uniquely determined. This is a consequence of the linear relations among the tweakey-schedule words recorded in Section II-B2: the four values are affine functions of $\gamma^6_4$, and the round that produces them is linear in the differences, so no additional guessing is required.

a) For the remaining plaintext-ciphertext pairs, guess $K_1'^8$, verify the differences on $\beta_5^7$, and discard invalid plaintext-ciphertext pairs.
b) For the remaining plaintext-ciphertext pairs, guess $K_4'^8$, verify the differences on $\beta_4^7$, and discard invalid plaintext-ciphertext pairs.
c) For the remaining plaintext-ciphertext pairs, guess $K_{11}'^8$, verify the differences on $\beta_7^7$, and discard invalid plaintext-ciphertext pairs.
d) For the remaining plaintext-ciphertext pairs, guess $K_{14}'^8$, verify the differences on $\beta_6^7$, and discard invalid plaintext-ciphertext pairs.

The time complexity of this step is $2^8 \cdot q2^{23} \cdot 2^{-24} \cdot 2^{45 \cdot 8} \cdot (2 \cdot 1)/(9 \cdot 16) = \frac{1}{9} q2^{264}$ 9-round encryptions; this leaves $2^8 \cdot q2^{23} \cdot 2^{-4 \cdot 8} = q2^{-1}$.

7) We perform an exhaustive search over all remaining key candidates to recover the correct key.

With $q = 2^4$, the probability that the key survives before the exhaustive search is $p = (1-2^{-24})^{q \cdot 2^{23}} = e^{-2^3} \approx 2^{-11.54}$. Thus, the complexity of exhaustive key search is less than $\frac{1}{9} q \cdot 2^{374}$. Overall, the data complexity is $2^{124}$, the memory complexity is $4 \cdot 2^{2 \cdot 120 - 1 + 4} = 2^{245}$ blocks, and the time complexity is $\frac{1}{9} \cdot 2^{374+4} \approx 2^{374.83}$.

The 9-round impossible differential attack on Joltik-BC-192 is very similar to that for Deoxys-BC-384. Set $q = 2^4$. The data complexity is $2^{64}$, the memory complexity is $4 \cdot 2^{2 \cdot 60 - 1 + 4} = 2^{125}$ blocks, and the time complexity is $\frac{1}{9} \cdot 2^{186+4} \approx 2^{186.83}$.

### *B. The impossible differential attacks on Deoxys-BC-256 and Joltik-BC-128*

As illustrated in Figure 31, an 8-round attack of Deoxys-BC-256 is constructed by prepending 2 rounds and appending 2 rounds to the 4-round impossible differential. The detailed attack steps are given below.

1) Consider a plaintext structure where the 1st, 2nd, 6th, 7th, 8th, 11th, 12th, and 13th bytes are active. By preparing $q$ such structures, we obtain $q \cdot 2^{127}$ plaintext–ciphertext pairs.
2) Perform partial encryption of the first round.
   a) For the surviving plaintext-ciphertext pairs, guess $K_1^0$, $K_6^0$, $K_{11}^0$, $K_{12}^0$, verify the differences on $\alpha_{13}^1$, $\alpha_{14}^1$, $\alpha_{15}^1$, and discard invalid plaintext-ciphertext pairs.
   b) For each surviving plaintext-ciphertext pair, guess $K_2^0$, $K_7^0$, $K_8^0$, $K_{13}^0$, verify the differences on $\alpha_8^1$, $\alpha_9^1$, $\alpha_{10}^1$, and discard invalid plaintext-ciphertext pairs.

   The time complexity of this step is $q2^{127} \cdot 2^{-3 \cdot 8} \cdot 2^{8 \cdot 8} \cdot (2 \cdot 4)/(8 \cdot 16) = q2^{163}$ 8-round encryptions; the remaining plaintext–ciphertext pairs number $q2^{127} \cdot 2^{-6 \cdot 8} = q2^{79}$.
3) Perform partial encryption of the third round. For the surviving plaintext-ciphertext pairs, guess $K_{11}^1$, $K_{12}^1$, verify the differences on $\alpha_{14}^2$, and discard invalid plaintext-ciphertext pairs. The time complexity of this step is $q2^{79} \cdot 2^{10 \cdot 8} \cdot (2 \cdot 2)/(8 \cdot 16) = q2^{154}$ 8-round encryptions; the surviving number of plaintext–ciphertext pairs is $q2^{79} \cdot 2^{-8} = q2^{71}$.
4) Perform partial decryption of the last round. Note that, since $K_{13}^0$ and $K_{12}^1$ are known, $K_{13}^8$ is known according to the tweakey schedule.
   a) For the remaining plaintext-ciphertext pairs, guess $K_1'^8, K_4'^8, K_{11}'^8$ and $K_{14}'^8$, verify the differences on $\alpha_5'^7, \alpha_6'^7, \alpha_7'^7$, and discard invalid plaintext-ciphertext pairs.
   b) For each surviving plaintext-ciphertext pair, guess $K_2'^8, K_5'^8, K_8'^8$ and $K_{15}'^8$, verify the differences on $\alpha_8'^7, \alpha_9'^7, \alpha_{10}'^7$, and discard invalid plaintext-ciphertext pairs.
   c) For the surviving plaintext-ciphertext pairs, guess $K_3'^8, K_6'^8, K_9'^8$ and $K_{12}'^8$, verify the differences on $\alpha_{12}'^7, \alpha_{13}'^7, \alpha_{15}'^7$, and discard invalid plaintext-ciphertext pairs.
   d) Since

$$\begin{pmatrix} K_{12}'^8 \\ K_{13}'^8 \\ K_{14}'^8 \\ K_{15}'^8 \end{pmatrix} = M^{-1} \cdot \begin{pmatrix} K_{12}^8 \\ K_{13}^8 \\ K_{14}^8 \\ K_{15}^8 \end{pmatrix}$$

   and $K_{12}'^8, K_{14}'^8, K_{15}'^8$ and $K_{13}^8$ are known, $K_{13}'^8$ can be deduced, where $M$ is the MDS matrix of AES. For each surviving plaintext-ciphertext pair, guess $K_0'^8, K_7'^8$ and $K_{10}'^8$, verify the differences on $\alpha_0'^7, \alpha_2'^7, \alpha_3'^7$, and discard invalid plaintext-ciphertext pairs.

   The time complexity of this step is $q2^{71}2^{-9 \cdot 8} \cdot 2^{25 \cdot 8} \cdot (2 \cdot 4)/(8 \cdot 16) = q2^{195}$ 8-round encryptions; this leaves $q2^{71} \cdot 2^{-12 \cdot 8} = q2^{-25}$.
5) Perform partial decryption of the second-last round. Once $\gamma_4^5$ is enumerated, the values of $\beta_4^6, \beta_5^6, \beta_6^6, \beta_7^6$ are uniquely determined. Again by the linear relations of Section II-B2, the four values are affine functions of $\gamma_4^5$, so no additional guessing is involved.
   a) For the remaining plaintext-ciphertext pairs, guess $K_1'^7$, verify the differences on $\beta_5^6$, and discard invalid plaintext-ciphertext pairs.

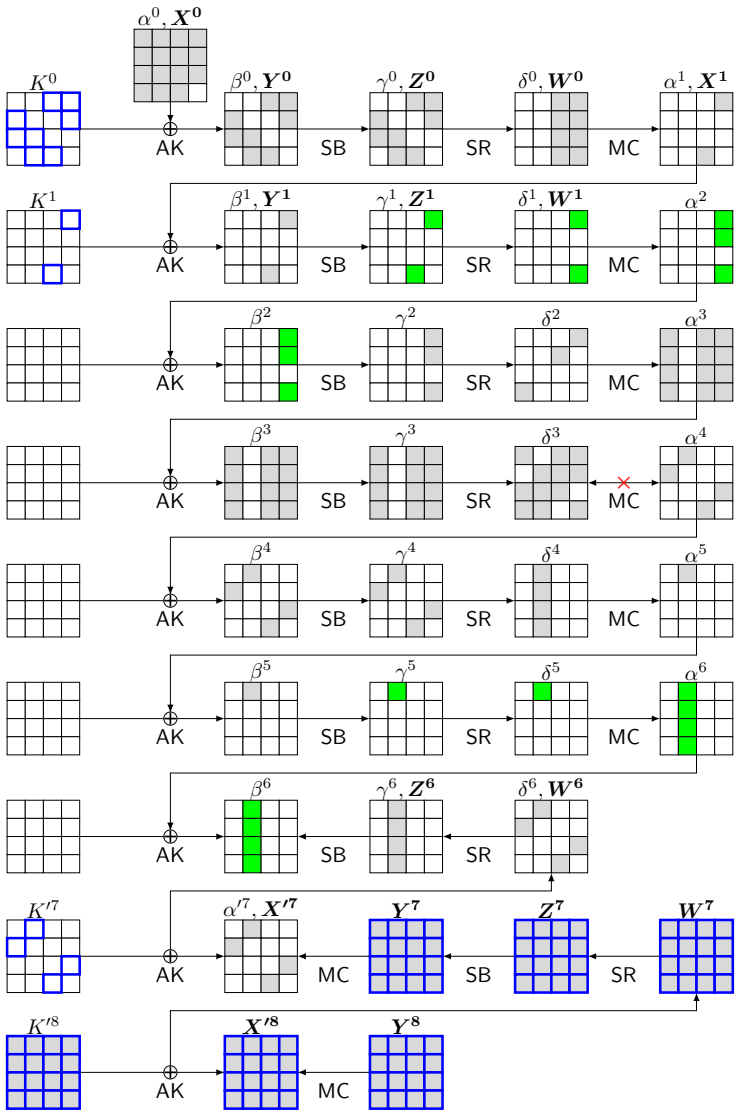


Fig. 31: The 8-round impossible differential attack of Deoxys-BC-256

b) For the remaining plaintext-ciphertext pairs, guess $K'^7_4$, verify the differences on $\beta^6_4$, and discard invalid plaintext-ciphertext pairs.
c) For the remaining plaintext-ciphertext pairs, guess $K'^7_{11}$, verify the differences on $\beta^6_7$, and discard invalid plaintext-ciphertext pairs.
d) For the remaining plaintext-ciphertext pairs, guess $K'^7_{14}$, verify the differences on $\beta^6_6$, and discard invalid plaintext-ciphertext pairs.

The time complexity of this step is $2^8 \cdot q2^{-25} \cdot 2^{-3 \cdot 8} \cdot 2^{29 \cdot 8} \cdot (2 \cdot 1)/(8 \cdot 16) = q2^{185}$ 8-round encryptions; the remaining plaintext–ciphertext pairs number $2^8 \cdot q2^{-25} \cdot 2^{-4 \cdot 8} = q2^{-49}$.

6) We perform an exhaustive search over all remaining key candidates to recover the correct key.

Choosing $q = 2^{53}$, the probability that the key survives before the exhaustive search is $p = (1 - 2^{-24})^{q \cdot 2^{-25}} = e^{-2^4} \approx 2^{-23.08}$. Thus, the complexity of exhaustive key search is less than $q2^{195}$. Overall, the data complexity is $2^{53+64} = 2^{117}$, the memory complexity is $4 \cdot 2^{2 \cdot 64 - 1 + 53} = 2^{182}$ blocks, and the time complexity is $q2^{195} = 2^{248}$.

The 8-round impossible differential attack on Joltik-BC-128 is very similar to that for Deoxys-BC-256. Set $q = 2^{28}$. The probability that the key survives before the exhaustive search is $p = (1 - 2^{-12})^{q \cdot 2^{-13}} = e^{-2^3} \approx 2^{-11.54}$. Thus, the complexity of exhaustive key search is less than $q2^{95}$. The data complexity is $2^{60}$, the memory complexity is $4 \cdot 2^{2 \cdot 32 - 1 + 4} = 2^{125}$ blocks, and the time complexity is $2^{95+28} = 2^{123}$.