Practical Key Collision on AES and Kiasu-BC

Jianqiang Ni[1*], Yingxin Li[1*], Fukang Liu[2] and Gaoli Wang[1(✉)]

[1] Shanghai Key Laboratory of Trustworthy Computing, School of Cryptology, Software Engineering Institute, East China Normal University, Shanghai, China
jianqiangni0213@163.com,liyx1140@163.com,glwang@sei.ecnu.edu.cn
[2] Institute of Science Tokyo, Tokyo, Japan,
liu.f.ad@m.titech.ac.jp

Abstract. The key collision attack was proposed as an open problem in key-committing security in Authenticated Encryption (AE) schemes like `AES-GCM` and `ChaCha20Poly1305`. In ASIACRYPT 2024, Taiyama et al. introduce a novel type of key collision—target-plaintext key collision (`TPKC`) for `AES`. Depending on whether the plaintext is fixed, `TPKC` can be divided into `fixed-TPKC` and `free-TPKC`, which can be directly converted into collision attacks and semi-free-start collision attacks on the Davies-Meyer (`DM`) hashing mode.

In this paper, we propose a new rebound attack framework leveraging a time-memory tradeoff strategy, enabling practical key collision attacks with optimized complexity. We also present an improved automatic method for finding *rebound-friendly* differential characteristics by controlling the probabilities in the inbound and outbound phases, allowing the identified characteristics to be directly used in *rebound-based* key collision attacks. Through our analysis, we demonstrate that the 2-round `AES-128` `fixed-TPKC` attack proposed by Taiyama et al. is a `free-TPKC` attack in fact, while `fixed-TPKC` attacks are considerably more challenging than `free-TPKC` attacks. By integrating our improved automatic method with a new rebound attack framework, we successfully identify a new differential characteristic for the 2-round `AES-128` `fixed-TPKC` attack and develope the first practical `fixed-TPKC` attack against 2-round `AES-128`. Additionally, we present practical `fixed-TPKC` attacks against 5-round `AES-192` and 3-round `Kiasu-BC`, along with a practical `free-TPKC` attack against 6-round `Kiasu-BC`. Furthermore, we reduce time complexities for `free-TPKC` and `fixed-TPKC` attacks on other `AES` variants.

Keywords: Key collision · Rebound-based attack · AES · SAT · DM hashing mode · Kiasu-BC

1

1 Introduction

For a long time, collision attacks have been a central focus in the cryptanalysis of hash functions, as they highlight potential vulnerabilities in the design of cryptographic algorithms. These attacks aim to find two different messages that yield the same hash value.

For block ciphers and stream ciphers, there also exists a type of collision attack, referred to as a key collision. The key collision in block ciphers can be defined as finding two different master keys that produce the same ciphertext when encrypting the same plaintext. In stream ciphers, it can be defined as finding two different master keys that generate the same key stream. Although key collision research has been initiated long ago, it has not been as popular as key recovery attacks or collision attacks. Early studies in key collision

[1] * These authors contributed equally to this work.

attacks include Kelsey et al.'s identification of trivial colliding keys [KSW96] for the Tiny Encryption Algorithm (TEA), and Aumasson et al.'s analysis of the ISDB Scrambling Algorithm MULTI2 [AJS09]. Matsui further analyzed the stream cipher `RC4` [Mat09], showing that two different keys could produce the same key stream. Moreover, Biryukov and Nikolic exploited the weaknesses in the key scheduling algorithm to discover colliding keys for `SC2000-256`[BN14].

At USENIX Security 2022, Albertini et al. [ADG+22] proposed the *padding fix* method to ensure key commitment security in Authenticated Encryption (AE) schemes like `AES-GCM` and `ChaCha20Poly1305`. This method simply prepends a constant block of all zeros to the plaintext and encrypts the padded plaintext as usual. During decryption, the presence of a leading block of zeros is checked to verify that the correct key was used. These key committing AE schemes are deemed "collision-resistant" because it is computationally challenging to find two different keys that either produce the same ciphertext when encrypting the same plaintext, or, yield two different plaintexts when decrypting the same ciphertext. Recently, the key commitment security has been further investigated in various authenticated encryption algorithms, such as Ascon [NSS23], Rocca [TTI24b, TTI24a], and some `AES`-based AEAD schemes [DFI+24]. In Albertini et al.'s analysis of the padding fix method, they posed an open problem:

"In particular, the padding fix with `AES-GCM` assumes an ideal cipher, which raises the following interesting question: Is it possible to find two keys k_1 and k_2 such that $AES_{k_1}(0) = AES_{k_2}(0)$ in fewer than approximately 2^{64} trials ?"

In ASIACRYPT 2024, Taiyama et al. [TSI+24a] explore this open question posed by Albertini et al. [ADG+22]. They define the concept of a target-plaintext key collision (`TPKC`) and categorize it based on whether the plaintext is predetermined into two types: fixed-target-plaintext key collision (`fixed-TPKC`) and free-target-plaintext key collision (`free-TPKC`). These two types of key collisions can be directly translated into collision attacks and semi-free-start collision attacks on Davies-Meyer (`DM`) hashing mode with `AES`. Consequently, `fixed-TPKC` attacks require simultaneous constraints on both the key and a specific plaintext, whereas `free-TPKC` allows adversaries to freely choose target plaintexts, making fixed-target-plaintext key collision attacks harder to achieve than free-target-plaintext key collision attacks.

Rebound-based Attack. The rebound attack is a generic analysis method introduced by Mendel et al. [MRST09] at FSE 2009, aimed at finding collision message pairs for `AES`-like hash functions. The rebound attack takes advantage of the high degrees of freedom (`DoF`) in the large state of cryptographic algorithms to efficiently identify state pairs that satisfy a truncated differential characteristic, thereby finding input pairs that meet the entire differential characteristic. This method divides the algorithm into two phases: the inbound and outbound phases. In the inbound phase, `DoF` are utilized to deterministically establish part of the differential characteristic, while the remaining part in the outbound phase is completed probabilistically.

At ASIACRYPT 2009, Lamberger et al. [LMR+09] introduced multiple inbound phases, utilizing key `DoF` to connect them. Subsequently, Gilbert and Peyrin [GP10], as well as Lamberger et al. [LMR+09], proposed the super S-box technique, extending the inbound phase to cover two rounds. To further optimize memory complexity, Sasaki et al. [SLW+10] leveraged the properties of MDS matrices at ASIACRYPT 2010 to propose non-full-active super S-boxes. Rebound attacks have been applied to many hash functions [JNP12, DDKS12, DGPW12, MRS14, KNR14]. At EUROCRYPT 2020, Hosoyamada and Sasaki [HS20] presented the first quantum collision attack on `AES-MMO`. At CRYPTO 2022, Dong et al. [DGLP22] combined the triangulation algorithm [KBN09] with the rebound attack to propose the *triangulating rebound attack*, constructing a super-inbound phase that enabled a 7-round semi-free-start attack and an 8-round quantum collision attack on

AES-128-MMO/MP.

Rebound attacks encompass a variety of applications, extending beyond collision attacks on hash functions to include the construction of limited birthday distinguishers [GP10, IPS13] and more [JNP13, DLP23]. We refer to this class of attack methods that incorporate the rebound attack concept as **rebound-based attacks**. Previous rebound-based attacks primarily used truncated differential, Taiyama et al. conducted rebound-based TPKC attacks on AES by searching bit-oriented differential characteristics. They converted these bit-oriented characteristics into a graphical format and applied depth-first search to generate a DoF tree, which optimally guides the selection and sequencing of inbound and outbound vertices.

1.1 Our Contribution

In this work, we propose an improved automatic search method to find *rebound-friendly* differential characteristics, which can be directly applied to AES TPKC attacks.

Improved Automatic Method for Finding Target-plaintext Key Collision Differential Characteristic. In this study, we employ a *Boolean satisfiability problem* (SAT)-based automatic method to search for bit-oriented differential characteristics, which are then applied in rebound-based key collision attacks. Unlike the approach proposed by Taiyama et al. [TSI+24a], we specifically aim to find a rebound-friendly differential characteristic suitable for a rebound-based attack. This characteristic is defined by having a lower differential probability in the inbound phase, along with high DoF. Additionally, it maintains a higher differential probability in the outbound phase, making it well-suited for launching rebound-based attacks. By adding relevant constraints to the traditional SAT model for finding high-probability differentials, we can discover rebound-friendly differential characteristics. Once the DoF in the inbound phase are determined, this differential characteristic can be directly used in rebound-based key collision attacks.

A New Rebound Attack Framework for Key Collision Attacks. In the traditional rebound attack framework [TSI+24a] for key collision attacks, attackers typically use the S-box of the round function as the inbound phase to derive round key, and then immediately proceed to the outbound phase. Inspired by the Super S-box technique [LMR+09, GP10], we introduce a time-memory tradeoff strategy and propose a new rebound attack framework. The new rebound attack framework is divided into two phases: the offline phase and the online phase. During the offline phase, we extend the inbound phase to cover more rounds of key schedule or rounds of the round function. This allows us to precompute starting points that satisfy the differential characteristics of the inbound phase and store them in a precomputation table. In the online phase, we leverage the precomputed table from the offline phase along with the remaining starting points to find colliding key pairs that satisfy the whole differential characteristics. Since the starting points satisfy partial differential characteristics are precomputed during the inbound phase, the differential probability in the outbound phase of the new rebound attack framework is increased compared to that of the traditional rebound attack framework. As the time complexity of rebound attacks primarily depends on the differential probability in the outbound phase, this approach optimizes the time complexity of traditional rebound attack frameworks. Through this new rebound attack framework, we successfully conduct additional practical key collision attacks, with the following results:

1. AES-128. In [TSI+24a], Taiyama et al. claim to propose a 2-round AES-128 fixed-TPKC attack with a time complexity of 2^{49}. However, we discover that their attack is actually a free-TPKC attack and the attack results were not practical. Subsequently, using the improved automatic search method, we identify a new

rebound-friendly differential characteristic for 2-round AES-128, where the outbound phase probability reaches 2^{-40}. By combining our proposed new rebound attack framework, we develop a new 2-round AES-128 fixed-TPKC attack and successfully find practical key collision pairs, which are listed in Table 2. We also improve the results of the 5-round AES-128 free-TPKC attack, reducing its time complexity.

2. AES-192 and AES-256. For AES-192 and AES-256, we identify new differential characteristics that improve both the 5-round AES-192 fixed-TPKC attack and the 7-round AES-192 free-TPKC attack, as well as the 6-round AES-256 fixed-TPKC attack. Using the new rebound attack framework, we also identify the practical key collision pairs for the 5-round AES-192 fixed-TPKC attack.

3. Kiasu-BC. Assuming the tweak can be chosen by the attacker, we utilize the DoF provided by the tweak to propose a fixed-TPKC attack on 3-round Kiasu-BC and a free-TPKC attack on 6-round Kiasu-BC, both of which successfully find practical key collision pairs.

The key collision attacks are directly applied to the DM hashing mode of AES and Kiasu-BC, and a summary of our results is provided in Table 1.

Table 1: Summary of (semi-free-start) collision attacks for AES in DM hashing mode.

Target	Attack	Round	Time	Memory	Reference
AES-128-DM	Collision*	2	2^{49}	-	[TSI+24b]
	Collision	**2**	**practical**	2^{22}	Section 4.2.1
	Two-block collision	3	2^{60}	2^{52}	[TSI+24b]
	Semi-free-start	5	2^{57}	-	[TSI+24b]
	Semi-free-start	**5**	$\mathbf{2^{54}}$	-	Appendix A
AES-192-DM	Collision	5	2^{61}	-	[TSI+24b]
	Collision	**5**	**practical**	2^{5}	Section 4.2.3
	Semi-free-start	7	2^{62}	-	[TSI+24b]
	Semi-free-start	**7**	$\mathbf{2^{56}}$	-	Appendix B
AES-256-DM	Collision†	6	2^{61}	-	[TSI+24b]
	Collision	**6**	$\mathbf{2^{60}}$	-	Appendix C
	Semi-free-start	9	2^{30}	-	[TSI+24b]
Kiasu-BC	Collision	**3**	**practical**	2^{10}	Section 4.3.2
	Semi-free-start	**6**	**practical**	-	Section 4.3.3

∗It is actually a semi-free start collision attack on AES-128-DM.
† The differential characteristic is incorrect.

1.2 Outline

The rest of the paper is organized as follows: Section 2 introduces some definitions related to AES encryption and AES-based hash functions, and reviews rebound attacks and key collision attacks. Section 3 describes our improved automated method for searching rebound-friendly differential characteristics, which is then applied to fixed-TPKC and free-TPKC attacks on AES and Kiasu-BC in Section 4. Finally, Section 5 concludes the paper.

2 Preliminaries

2.1 AES Encryption and AES-based Hash functions

AES Encryption. AES [DR06] is a block cipher that encrypts data in 128-bit blocks. It supports key lengths of 128 bits, 192 bits, or 256 bits, with the number of rounds depending on the key length (10 rounds for AES-128, 12 rounds for AES-192, and 14 rounds for AES-256). Its internal state can be viewed as a 4×4 byte array. Each round of AES consists of four transformations, as shown in Figure 1. Here, $x_i^{(j)}, 1 \leq j \leq 4$ represents the j-th column of the state x_i.

- SubBytes(SB): Applies an 8-bit S-box in parallel to each byte.
- ShiftRows(SR): Performs a cyclic left shift of the i-th row by i positions, for $i = 0, 1, 2, 3$.
- MixColumns(MC): Multiplies each column by a 4×4 Maximum Distance Separable (MDS) matrix.
- AddRoundKey(ARK): XORs the 128-bit round key with the intermediate state.

Before the initial round of encryption, an additional operation called the XOR whitening key is applied. Additionally, in the final round of encryption, the MC operation is excluded.

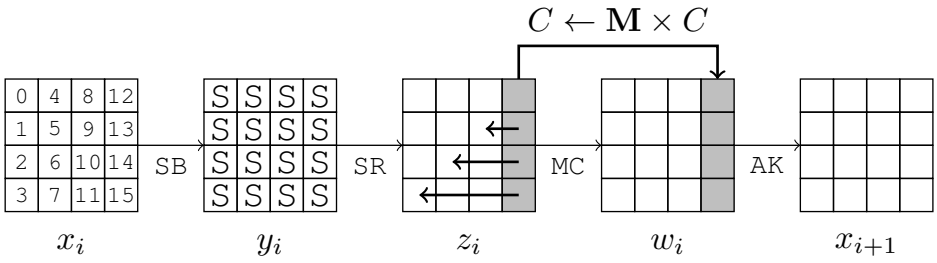
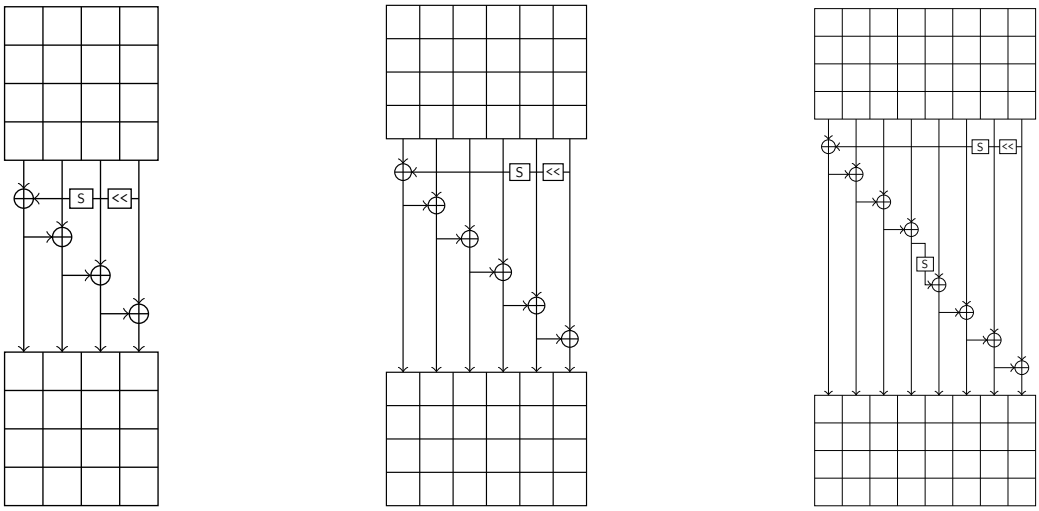


Figure 1: The round function of AES.

In the AES key schedules, N represents the length of the key in 32-bit words, which is 4 words for AES-128, 6 words for AES-192, and 8 words for AES-256, as illustrated in Figure 2. The master key is denoted as $K_0, K_1, \ldots, K_{N-1}$ in 32-bit words. The number of round keys, N_r, required for each AES variant is 11 for AES-128, 13 for AES-192, and 15 for AES-256. The expanded key is represented by $W_0, W_1, \ldots, W_{4N_r-1}$ in 32-bit words. To define the transformations, RotWord(RW) is used as a one-byte left circular shift, expressed as $\mathtt{RW}([b_0, b_1, b_2, b_3]) = [b_1, b_2, b_3, b_0]$. SubWord(SW) is defined as the application of the AES S-box to each of the four bytes of a word, represented as $\mathtt{SW}([b_0, b_1, b_2, b_3]) = [\mathtt{SB}(b_0), \mathtt{SB}(b_1), \mathtt{SB}(b_2), \mathtt{SB}(b_3)]$. For $i = 0 \ldots 4N_r - 1$, the expanded key words W_i are defined as follows:

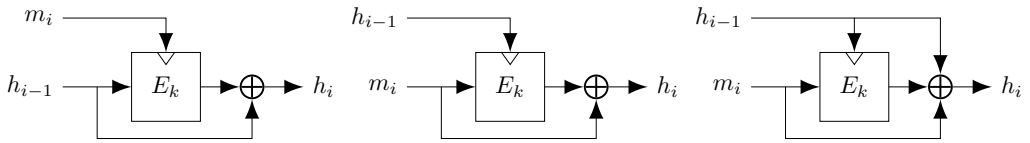
$$W_i = \begin{cases} K_i & \text{if } i < N, \\ W_{i-N} \oplus \mathtt{SW}(\mathtt{RW}(W_{i-1})) \oplus \mathrm{rcon}_{i/N} & \text{if } i \geq N \text{ and } i \equiv 0 \pmod{N}, \\ W_{i-N} \oplus \mathtt{SW}(W_{i-1}) & \text{if } i \geq N, N > 6, \text{ and } i \equiv 4 \pmod{N}, \\ W_{i-N} \oplus W_{i-1} & \text{otherwise.} \end{cases}$$



(a) `AES-128` (b) `AES-192` (c) `AES-256`

Figure 2: Key schedules for the three versions of `AES` [Jea16]. The functions `RW` and `SW` are represented by $\ll$ and S, respectively. The addition of the round-dependent constant, which is integral to the `AES` key schedule, is not depicted here; for detailed information, please refer to [DR06].

`AES`-based Hash functions. Classic hash functions, such as the MD-SHA hash family [Riv92, Pub12], are constructed by combining compression functions (CF) with the Merkle-Damgård construction [Mer89, Dam89]. Similarly, `AES`-based hash functions use compression functions that can be constructed with `AES` round functions in hashing modes such as `DM`, `MMO`, and `MP` [PGV93, MvOV96], as illustrated in Figure 3. Integrating these compression functions into the Merkle-Damgård construction results in `AES`-based Hash functions.



(a) Davies-Meyer (`DM`) (b) Matyas-Meyer-Oseas (`MMO`) (c) Miyaguchi-Preneel (`MP`)

Figure 3: Commonly used hashing modes.

2.2 Rebound Attack

The rebound attack [MRST09], introduced by Mendel et al. at FSE 2009, is an effective method for analyzing `AES`-like hash functions. This attack consists of two phases: the inbound phase and the outbound phase. During the attack, the internal structure of the hash function, whether based on block ciphers or permutations, is divided into three parts: $F = F_{fw} \circ F_{in} \circ F_{bw}$, the overall framework as illustrated in Figure 4.

- **Inbound Phase.** In this phase, the attacker typically aims to find a differential characteristic that maintains a low internal probability. By employing the meet-in-the-middle technique, the attacker seeks to obtain as many data pairs as possible that satisfy the inbound differential characteristic, referred to as starting points. The

maximum number of starting points achievable during the inbound phase is known as the degree of freedom (DoF). These starting points will be used in the outbound phase to facilitate the attack.

- **Outbound Phase.** In this phase, the attacker generally needs to control the probability of the differential characteristic to be greater than that of the birthday paradox. By utilizing the matched pairs obtained in the inbound phase, the attacker performs forward and backward calculations to derive a pair of values that satisfy the differential characteristic requirements of the outbound phase. Ultimately, this process leads to the desired collision pairs.

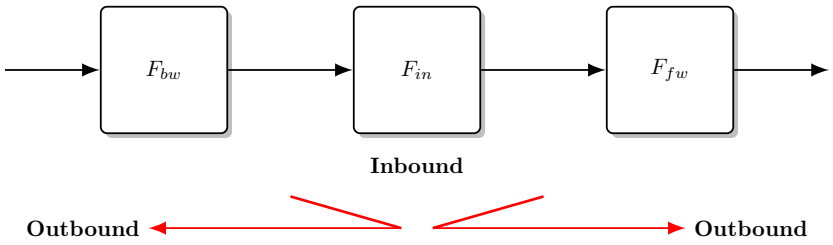


Figure 4: The over framework of rebound attack.

At CRYPTO 2022, Dong [DGLP22] et al. introduced the *Triangulating Rebound Attack* by combining the triangulation algorithm with the rebound attack. They constructed a super-inbound phase to enable the inbound phase to cover more rounds. The sources of DoF for the rebound attack were categorized into five cases:

D1 Truncated differentials of the internal rounds of the Super-Inbound.

D2 Input/output differences and values of the active S-boxes in the state.

D3 Values of the inactive S-boxes in the state.

D4 Differences of key bytes.

D5 Values of key bytes.

In collision or semi-free-start collision attacks on MMO/MP hashing modes, differences can be introduced through the plaintext of the block cipher and may be canceled during the message feed-forward operation, as shown in Figure 3 (b), (c). Therefore, attackers can leverage truncated differential characteristics and utilize the DoF from D1, D2, or D5 to find collision pairs (notably, the DoF D4 and D5 can be more generally represented as "Differences of IV bytes" and "Values of IV bytes").

However, in collision or semi-free-start collision attacks on DM hashing modes or in key collision attacks, differences can only be introduced through the keys of the block cipher, as the message feed-forward operation does not contribute. As a result, attackers must find a differential characteristic where differences cancel themselves out in the output. In this scenario, using truncated differential characteristics for rebound attacks becomes challenging. Instead, it is preferable to identify a bit-wise differential characteristic and exploit the DoF from D2, D3, and D5 within the inbound phase to launch collision or semi-free-start collision attacks in the DM hashing mode.

2.3 Key Collision and DM Hashing Mode Collision

In the upcoming ASIACRYPT 2024, Taiyama et al. [TSI+24a], building on the open problem proposed by Albertini et al. [ADG+22], introduce the concept of target-plaintext key collision (TPKC), defined as follows:

Definition 1 (Target-Plaintext Key Collision[TSI+24a])**.** It refers to two different keys that produce the same ciphertext when encrypting a specific plaintext.

Based on whether the plaintext is fixed in advance, the key collision attack scenario can be divided into fixed-target-plaintext key collision (`fixed-TPKC`) and free-target-plaintext key collision (`free-TPKC`). In `fixed-TPKC`, the plaintext is predetermined, and the goal is to find two different keys that produce the same ciphertext when encrypting this fixed plaintext. In `free-TPKC`, the objective is to find different keys along with corresponding plaintexts such that encryption with these key-plaintext pairs yields the same ciphertext.

Problem 1 (Fixed-Target-Plaintext Key Collision[TSI+24a])**.** *Given a single target plaintext, find a key pair that generates the same ciphertext.*

Problem 2 (Free-Target-Plaintext Key Collision[TSI+24a])**.** *Find a key pair and a corresponding single plaintext that generates the same ciphertext.*

Consider a hash function H constructed using the `DM` hashing mode, as shown in Figure 3 (a), where E_k represents AES encryption. To launch a standard single-block collision attack, the attacker needs to find a message pair (m, m') such that $\texttt{AES}_m(IV) = \texttt{AES}_{m'}(IV)$, where IV is the fixed initial value. To perform a single-block semi-free-start collision, the attacker must find a pair (p, m) and (p, m') such that $\texttt{AES}_m(p) = \texttt{AES}_{m'}(p)$, with $p \neq IV$. To carry out a single-block free-start collision, the attacker needs to find a pair (p, m) and (p', m') such that $\texttt{AES}_m(p) = \texttt{AES}_{m'}(p')$, with $p \neq p'$. It is evident that the objective of a `fixed-TPKC` aligns with that of a single-block collision attack in `DM` hashing mode, while the objective of a `free-TPKC` matches that of a single-block semi-free-start collision attack in `DM` hashing mode.

Remark 1. It should be noted that the hash collision attacks mentioned subsequently refer to single-block collisions unless explicitly stated as two-block collisions.

3 Improved Automatic Search Method for Differential Characteristics in Key Collision

In this section, we propose an improved SAT-based automatic search method for searching *rebound-friendly* differential characteristics. The main idea of our method is to achieve more precise control over the probabilities in the inbound and outbound phases of a differential characteristic, thereby identifying characteristics that are more effective for attacks.

In essence, we enhance the model for searching fixed-probability bit-oriented differential characteristics by introducing constraints to regulate the inbound phase probability, ensuring that the outbound phase probability exceeds the birthday bound. This enables the discovery of rebound-friendly differential characteristics. Our experiments show that adding more constraints significantly reduces the search time for differential characteristics. We will first introduce the process of finding fixed-probability bit-oriented differential characteristics, followed by the method of incorporating additional constraints to identify rebound-friendly differential characteristics.

3.1 SAT-based Method to Search for Bit-Oriented Key Collision Differential Characteristic

The primary challenge is how to transform the search for differential characteristics into the *conjunctive normal form* (CNF) such that the off-the-shelf solvers can solve it. We follow the modelling approach of Sun et al. [SWW21, SW23]. In the following, we briefly introduce the operations in `AES` that need to be modelled and the formulation of the objective function to search for a differential characteristic with a specified probability.

Modelling for S-box of AES. Since we aim to control specific differential probabilities, the S-box model must represent the exact probability propagation. The probabilities of possible differential propagations $\Delta_i \to \Delta_o$ for the AES S-box can take values from the set $\{2^{-7}, 2^{-6}, 1\}$. Therefore, for each S-box, we introduce two binary variables $(\mathtt{w}_7, \mathtt{w}_6)$ to represent the possible differential path probabilities. The complete set of possible differential propagations for each S-box is given by:

$$\mathcal{P}_{Sbox} = \left\{ \Delta_i \parallel \Delta_o \parallel \mathtt{w}_7 \parallel \mathtt{w}_6 \;\middle|\; \begin{array}{l} \Delta_i, \Delta_o \in \mathbb{F}_8^2,\ \mathtt{w}_7, \mathtt{w}_6 \in \mathbb{F}_2, \\ \mathtt{w}_7 \parallel \mathtt{w}_6 = \begin{cases} 0 \parallel 0, & \Pr_s(\Delta_i, \Delta_o) = 1, \\ 1 \parallel 0, & \Pr_s(\Delta_i, \Delta_o) = 2^{-7}, \\ 0 \parallel 1, & \Pr_s(\Delta_i, \Delta_o) = 2^{-6}. \end{cases} \end{array} \right\}$$

To convert all the possible differential propagation sets into CNF, we introduce a boolean function as follows:

$$f(\Delta_i \parallel \Delta_o \parallel \mathtt{w}_7 \parallel \mathtt{w}_6) = \begin{cases} 1, & \text{if } \Delta_i \parallel \Delta_o \parallel \mathtt{w}_7 \parallel \mathtt{w}_6 \in \mathcal{P}_{sbox}, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

A Boolean function $f(x_1, x_2, \ldots, x_n)$ maps n-dimensional binary inputs $(x_1, x_2, \ldots, x_n)$ to a binary output $\{0, 1\}$. Any Boolean function can be represented in CNF which is a conjunction (logical AND) of one or more clauses, where each clause is a disjunction (logical OR) of literals. A literal is a variable or its negation, denoted as x_i or $\neg x_i$, respectively. Formally, a CNF formula ϕ is written as:

$$\phi = C_1 \wedge C_2 \wedge \cdots \wedge C_k,$$

where each clause C_j is expressed as:

$$C_j = (l_{j_1} \vee l_{j_2} \vee \cdots \vee l_{j_m}),$$

and l_{j_i} is a literal.

The Boolean function 1 is then translated into CNF clauses. To obtain a simplified CNF representation, we use the Espresso logic minimizer[2]. After minimization, each AES S-box requires 8292 clauses to fully represent the differential propagation rules. We define the function $\mathtt{addSBclauses}(\Delta_i, \Delta_o, \mathtt{w}_7, \mathtt{w}_6)$ to return the 8,292 clauses.

By introducing these two variables, referred to as weight variables, to control the probabilities of weights 7 and 6 independently, the probability weight of the S-box can be determined by $7 \cdot \mathtt{w}_7 + 6 \cdot \mathtt{w}_6$. Suppose we want to control the propagation probability of a particular S-box to be 2^{-7} or 2^{-6}. In that case, we can set one variable to 1 and the other variable to 0, which makes it more convenient to manage the probabilities of multiple S-boxes.

Modelling for Linear Operations of AES. In the AES round function, apart from the SB operation, the remaining operations are linear. Among these, the SR operation does not introduce new clauses, so we only need to model the MC and ARK operations. Both ARK and MC operations can essentially be transformed into XOR operation modelling.

Let the input differences of the XOR operation be $\Delta_a, \Delta_b \in \mathbb{F}_2$, and the output difference be $\Delta_c \in \mathbb{F}_2$. The differential propagation is valid if and only if the values of Δ_a, Δ_b, and Δ_c satisfy all the following XOR clauses.

$$\begin{cases} (\neg\Delta_a \vee \Delta_b \vee \Delta_c) = 1, \\ (\Delta_a \vee \neg\Delta_b \vee \Delta_c) = 1, \\ (\Delta_a \vee \Delta_b \vee \neg\Delta_c) = 1, \\ (\neg\Delta_a \vee \neg\Delta_b \vee \neg\Delta_c) = 1. \end{cases} \tag{2}$$

[2]https://github.com/classabbyamp/espresso-logic

We define the function $\texttt{addXORclauses}(\Delta_a, \Delta_b, \Delta_c)$ to return the CNF clauses 2.

In the MC operation of AES, we utilize the primitive representation [SLR+15] to express the MDS matrix M, which is composed of elements 0x01,0x02,0x03 in $\mathbb{F}_{2^8}$. The primitive representation of the matrix, denoted as M^{pr}, is a 32×32 binary matrix. Each element in M (e.g.,0x01,0x02,0x03) is replaced by its corresponding binary representation in M^{pr}.

$$M = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix}$$

$$\texttt{0x01} = \begin{pmatrix} 1&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \end{pmatrix}, \texttt{0x02} = \begin{pmatrix} 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \\ 1&0&0&0&0&0&0&0 \end{pmatrix}, \texttt{0x03} = \begin{pmatrix} 1&1&0&0&0&0&0&0 \\ 0&1&1&0&0&0&0&0 \\ 0&0&1&1&0&0&0&0 \\ 1&0&0&1&1&0&0&0 \\ 1&0&0&0&1&1&0&0 \\ 0&0&0&0&0&1&1&0 \\ 1&0&0&0&0&0&1&1 \\ 1&0&0&0&0&0&0&1 \end{pmatrix}.$$

Given an input difference $\Delta_a \in \mathbb{F}_2^{32}$ and an output difference $\Delta_b \in \mathbb{F}_2^{32}$, the MC operation can be expressed as: $\Delta_b = M^{\text{pr}}\Delta_a$. To model this operation, we analyze the nonzero positions in each row of M^{pr} and apply a t-XOR operation to the corresponding positions of Δ_a. Taking the first row of M^{pr} as an example, the leftmost position is labeled as 0, and the nonzero indices in this row are given by $T = \{1, 8, 9, 16, 24\}$. Therefore, we can model the computation of $\Delta_b[0]$ as:

$$\Delta_b[0] = \bigoplus_{i \in T} \Delta_a[i]$$

Observation shows that the value of t is either 5 or 7, meaning that only 5-XOR and 7-XOR operations are required for modelling. The specific clauses and XOR operations follow the same principle but are omitted here for brevity. Finally, we define the function $\texttt{addMCclause}(\Delta_a, \Delta_b)$, which returns the CNF clauses for the MC operation.

Modelling for Objective Function. To more precisely control the probability of differential characteristics, our aim is to set the objective function to control the number of S-boxes in the differential characteristic that contributes probabilities of 2^{-7} and 2^{-6}. Thus, we can define the objective function as

$$\begin{cases} \sum_{0 \le r \le \mathtt{R}-1, 0 \le i \le 15} \mathtt{w}_7[r][i] + \sum_{0 \le r \le \mathtt{R}_k-1, 0 \le i \le 3} \mathtt{wk}_7[r][i] = NS_7, \\ \sum_{0 \le r \le \mathtt{R}-1, 0 \le i \le 15} \mathtt{w}_6[r][i] + \sum_{0 \le r \le \mathtt{R}_k-1, 0 \le i \le 3} \mathtt{wk}_6[r][i] = NS_6, \end{cases} \tag{3}$$

Where R represents the number of rounds for which we want to search for the differential characteristic, and $\mathtt{R}_k$ represents the number of rounds in the key schedule, $0 \le i \le 15$ indicates that the AES state consists of 16 bytes. In the r-th round, $\mathtt{w}_7[r][i]$ and $\mathtt{w}_6[r][i]$ are two binary variables controlling the probability of the i-th S-box, while $\mathtt{wk}_7[r][i]$ and $\mathtt{wk}_6[r][i]$ are the corresponding binary variables for the key schedule. Additionally, NS_7 and NS_6 represent the number of S-boxes in the differential characteristic where the probabilities are set to 2^{-7} and 2^{-6}, respectively.

To convert the objective function (3) into CNF clauses, we employ the sequential encoding method [Sin05], which is adept at transforming inequalities involving summations of boolean variables into constraints suitable for SAT solvers. Specifically, the sequential encoding method can be used to convert the inequality $\sum_{i=0}^{l} x_i \le k$ into CNF clauses,

where x_i are boolean variables and k is a non-negative integer. We add additional constraints to convert it into $\sum_{i=0}^{l} x_i = k$ for our purposes.

$$\begin{cases} \neg x_0 \vee s_{0,0} = 1, \\ x_0 \vee \neg s_{0,0} = 1, \\ \neg s_{0,j} = 1 \quad \text{for } 1 \le j \le k-1, \\ \neg x_i \vee s_{i,0} = 1, \\ \neg s_{i-1,0} \vee s_{i,0} = 1, \\ x_i \vee s_{i-1,0} \vee \neg s_{i,0} = 1, \\ \left.\begin{array}{l} \neg x_i \vee \neg s_{i-1,j-1} \vee s_{i,j} = 1, \\ \neg s_{i-1,j} \vee s_{i,j} = 1, \\ x_i \vee s_{i-1,j} \vee \neg s_{i,j} = 1, \\ s_{i-1,j-1} \vee s_{i-1,j} \vee \neg s_{i,j} = 1, \end{array}\right\} \text{for } 1 \le j \le k-1 \\ \neg x_i \vee \neg s_{i-1,k-1} = 1, \\ \neg x_{n-1} \vee \neg s_{n-2,k-1} = 1, \end{cases} \text{for } 1 \le i \le n-2.$$

where $s_{i,j} \in \mathbb{F}_2$ is an auxiliary variable. Let the $\texttt{addOFclauses}(\texttt{R}, \texttt{R}_k, \texttt{w}_7, \texttt{w}_6, \texttt{w}k_7, \texttt{w}k_6, NS_7, NS_6)$ function return the CNF clauses corresponding to the objective function (3).

Modelling for Finding a Fixed-probability Key Collision Differential characteristic. To search for key collision differential characteristics, it is necessary to control the differences at the plaintext and ciphertext positions to be zero, allowing only the master key input to introduce differences. Therefore, constraints must be added to ensure that the differential bits for both plaintext and ciphertext are zero across all positions. The constraints are formalized as follows:

$$\begin{cases} \texttt{plaintext}[i][j] = 0 & \text{for } 0 \le i \le 15, 0 \le j \le 7, \\ \texttt{ciphertext}[i][j] = 0 & \text{for } 0 \le i \le 15, 0 \le j \le 7. \end{cases} \tag{4}$$

Here, $\texttt{plaintext}[i][j]$ and $\texttt{ciphertext}[i][j]$ represent the differential bit at the j-th bit of the i-th byte of the plaintext and ciphertext, respectively. We define the function $\texttt{addKCclauses}(\texttt{plaintext}, \texttt{ciphertext})$ to return the CNF clauses corresponding to the key collision constraints (4).

We aim to construct an SAT model for AES by defining variables $\texttt{dx}_r, \texttt{dy}_r, \texttt{dz}_r$ to represent the 128-bit state differences before the SB operation, after the SB operation, and after the MC operation in the r-th round of AES, respectively. Variables $\texttt{w}_7$ and $\texttt{w}_6$ represent the probabilities of S-boxes in the AES round function, while sdk and wk represent the key state differences after the SW operation and the probabilities of S-boxes in the key schedule, respectively. Using the described functions as before, we construct an SAT model $\mathcal{M}$ to compute a fixed-probability differential characteristic for AES, as outlined in the Algorithm 1. The function $\texttt{findFPKCcharacteristic}(\texttt{R}, NS_7, NS_6, \texttt{version})$ is defined to return this SAT model.

3.2 How to Search Rebound-Friendly Differential Characteristics

In order to search for rebound-friendly differential characteristics, we need to control for a lower differential probability p_{in} in the inbound phase, a higher differential probability p_{out} in the outbound phase, and the DoF greater than or equal to $1/p_{out}$ in the inbound phase. In this section, we will build upon the fixed-probability differential characteristic search in Section 3.1 and discuss how to add constraints to find rebound-friendly differential characteristics. The constraints for finding differential characteristics in fixed-TPKC attacks, those for finding free-TPKC differential characteristics, and the constraints for different AES versions are all distinct, and we will discuss them in detail.

Algorithm 1: SAT-based method for searching fixed-probability key collision differential characteristics in AES

Input: The number of rounds R, the number of S-boxes NS_7 and NS_6 with different probabilities 2^{-7} and 2^{-6}, the AES version.

Output: A SAT model for searching differential characteristic with probability $p = 2^{-7 \cdot \mathtt{w}_7 - 6 \cdot \mathtt{w}_6}$.

Initialize an empty SAT model $\mathcal{M}$;
if *version=128* **then**
  $\mathtt{R}_k \leftarrow \mathtt{R}$;
if *version=192* **then**
  $\mathtt{R}_k \leftarrow \lceil (\mathtt{R}+1) \cdot \frac{2}{3} - 1 \rceil$;
if *version=256* **then**
  $\mathtt{temp} \leftarrow \lceil (\mathtt{R}+1) \cdot \frac{2}{3} - 1 \rceil$, $\mathtt{R}_k \leftarrow \mathtt{temp} + \lceil \frac{\mathtt{temp}}{2} \rceil$;
$\mathcal{M}.\mathtt{var} \leftarrow \{\mathtt{plaintext}[i][j] \in \{0,1\} : 0 \le r \le \mathtt{R}-1, 0 \le i \le 15, 0 \le j \le 7\}$;
$\mathcal{M}.\mathtt{var} \leftarrow \{\mathtt{dx}_r[i][j] \in \{0,1\} : 0 \le r \le \mathtt{R}-1, 0 \le i \le 15, 0 \le j \le 7\}$;
$\mathcal{M}.\mathtt{var} \leftarrow \{\mathtt{dy}_r[i][j] \in \{0,1\} : 0 \le r \le \mathtt{R}-1, 0 \le i \le 15, 0 \le j \le 8\}$;
$\mathcal{M}.\mathtt{var} \leftarrow \{\mathtt{w}_7[r][i] \in \{0,1\}, \mathtt{w}_6[r][i] : 0 \le r \le \mathtt{R}-1, 0 \le i \le 15\}$;
$\mathcal{M}.\mathtt{var} \leftarrow \{\mathtt{dz}_r[i][j] \in \{0,1\} : 0 \le r \le \mathtt{R}-2, 0 \le i \le 15, 0 \le j \le 8\}$;
$\mathcal{M}.\mathtt{var} \leftarrow \{\mathtt{dk}_r[i][j] \in \{0,1\} : 0 \le r \le \mathtt{R}, 0 \le i \le 15, 0 \le j \le 8\}$;
$\mathcal{M}.\mathtt{var} \leftarrow \{\mathtt{sdk}_r[i][j] \in \{0,1\} : 0 \le r \le \mathtt{R}_k-1, 0 \le i \le 3, 0 \le j \le 8\}$;
$\mathcal{M}.\mathtt{var} \leftarrow \{\mathtt{wk}_7[r][i] \in \{0,1\}, \mathtt{wk}_6[r][i] \in \{0,1\} : 0 \le r \le \mathtt{R}_k-1, 0 \le i \le 3\}$;
$\mathcal{M}.\mathtt{var} \leftarrow \{\mathtt{ciphertext}[i][j] \in \{0,1\} : 0 \le r \le \mathtt{R}-1, 0 \le i \le 15, 0 \le j \le 7\}$;
$\mathcal{M}.\mathtt{clause} \leftarrow \mathtt{addARKclauses}(\mathtt{plaintext}, \mathtt{dk}_0)$ /*Return the CNF clauses for the ARK operation*/;
for $r = 0$ *to* $R-1$ **do**
  for $i = 0$ *to* 15 **do**
    $\mathcal{M}.\mathtt{clause} \leftarrow \mathtt{addSBclauses}(\mathtt{dx}_r[i], \mathtt{dy}_r[i], \mathtt{w}_7[r][i], \mathtt{w}_6[r][i])$;
  if $r < R-1$ **then**
    $\mathcal{M}.\mathtt{clause} \leftarrow \mathtt{addMCclause}(\mathtt{SR}(\mathtt{dz}_r), \mathtt{dk}_{r+1})$;
    $\mathcal{M}.\mathtt{clause} \leftarrow \mathtt{addARKclauses}(\mathtt{dz}_r, \mathtt{dk}_{r+1})$;
  else
    $\mathcal{M}.\mathtt{clause} \leftarrow \mathtt{addARKclauses}(\mathtt{SR}(\mathtt{dy}_r), \mathtt{dk}_{r+1})$;
$\mathcal{M}.\mathtt{clause} \leftarrow \mathtt{addKSclauses}(\mathtt{R}_k, \mathtt{dk}, \mathtt{sdk}, \mathtt{wk}_7, \mathtt{wk}_6, \mathtt{version})$/*Return the CNF clauses for the key schedule*/;
$\mathcal{M}.\mathtt{clause} \leftarrow \mathtt{addKCclauses}(\mathtt{plaintext}, \mathtt{ciphertext})$;
$\mathcal{M}.\mathtt{clause} \leftarrow \mathtt{addOFclauses}(\mathtt{R}, \mathtt{R}_k, \mathtt{w}_7, \mathtt{w}_6, \mathtt{wk}_7, \mathtt{wk}_6, NS_7, NS_6)$;
return $\mathcal{M}$;

3.2.1 Constraints on AES Fixed-target-plaintext Key Collision

AES-128. We start by analyzing AES-128. Figure 5 represents the first round of the 2-round AES-128 fixed-TPKC differential characteristic we identify. In this differential characteristic, the inbound phase (marked with a red dashed line) begins with the SB operation in the first round, where the gray and blue cells represent fixed differences rather than truncated differences.

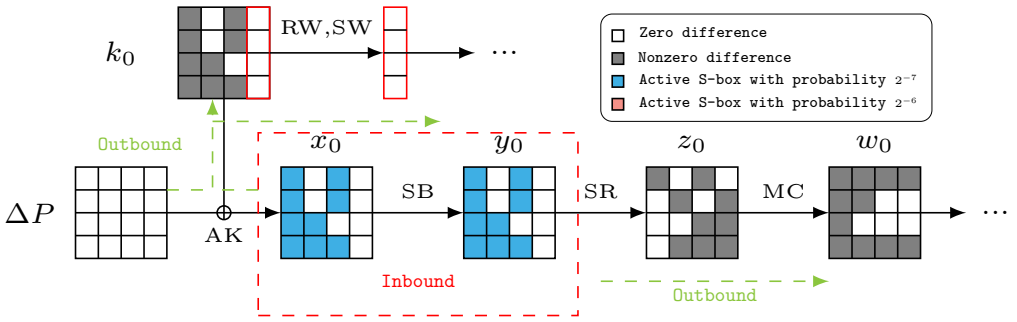


Figure 5: The first round of the 2-round AES-128 fixed-target-plaintext key collision differential characteristic.

In the following, we describe how to build on the fixed-probability differential characteristic search from the previous subsection by adding constraints to enable a direct search for rebound-friendly differential characteristics. In the inbound phase, given the differences Δx_0 and Δy_0, the differential distribution table (DDT) allows us to calculate the value of pairs that satisfy this difference. For an S-box with a probability of 2^{-7}, two pairs satisfy the difference, while for 2^{-6}, there are four pairs. The number of these pairs represents the DoF in rebound-based attacks. Positions with zero difference have 2^8 DoF available. Since the plaintext is fixed in advance in fixed-TPKC attacks, we cannot use the DoF from the plaintext in computing state or key values. By leveraging the DoF provided by the active byte differentials and the values of inactive bytes during the inbound phase, we first compute a pair (x_0, y_0) that satisfies the difference for the SB operation. This pair (x_0, y_0) serves as the starting point in a rebound-based attack. Subsequently, the value of k_0 can be calculated as $k_0 = x_0 \oplus P$, initiating the outbound phase.

From this analysis, we can conclude that the rebound-based key collision attack essentially utilizes the high DoF in the inbound phase's internal state to calculate the value of round keys, and then propagates these key and state values forward and backward to compute colliding key pairs. To precisely control the probability in the inbound phase, we introduce two constraints in the fixed-probability differential characteristic search model.

Suppose we want to search for a differential characteristic with probability p. The DoF available in an AES-128 fixed-TPKC attack is 2^{128}, requiring us to ensure the probability $p \geq 2^{-128}$. Additionally, to guarantee the outbound phase differential probability $p_{out} > 2^{-64}$, making it feasible to find a collision within a complexity less than $2^{n/2}$, we set the inbound phase probability as $p_{in} = p/p_{out}$. Let $w_{in} = -\log_2 p_{in}$ represent the total differential probability weight in the inbound phase. This weight can be decomposed as $w_{in} = 7 \cdot NS_{in_7} + 6 \cdot NS_{in_6}$. We then add the following constraints to control the probability in the inbound phase:

$$\begin{cases} \sum_{0 \leq i \leq 15} \mathtt{w}_7[0][i] = NS_{in_7}, \\ \sum_{0 \leq i \leq 15} \mathtt{w}_6[0][i] = NS_{in_6}, \end{cases} \tag{5}$$

where $\mathtt{w}_7[0][i]$and $\mathtt{w}_6[0][i]$ are binary variables controlling the probability for each S-box during the first round of the SB operation. For convenience in the following description,

let $\mathcal{C}^{\texttt{fixed-TPKC}}_{\texttt{AES-128}_1}$ represent constraint (5).

In order to prevent the consumption of the plaintext's DoF by the first round SW operation in the key schedule, it is crucial to ensure that the input difference for this operation is zero, or equivalently, that the input difference of the SW operation matches the input difference of the first round SB operation. Consequently, the following constraints are imposed:

$$\begin{cases} \texttt{wk}_7[0][i] = 0 \text{ for } 0 \le i \le 3, \\ \texttt{wk}_6[0][i] = 0 \text{ for } 0 \le i \le 3. \end{cases} \text{ or } \begin{cases} \texttt{wk}_7[0][i] = \texttt{w}_7[0][i] \text{ for } 0 \le i \le 3, \\ \texttt{wk}_6[0][i] = \texttt{w}_6[0][i] \text{ for } 0 \le i \le 3. \end{cases} \tag{6}$$

Here, $\texttt{wk}_7[0][i]$ and $\texttt{wk}_6[0][i]$ are binary variables controlling the probability for each S-box in the SW operation. Let $\mathcal{C}^{\texttt{fixed-TPKC}}_{\texttt{AES-128}_2}$ represent constraint (6).

AES-192. As analyzed for AES-128, the differences in the inbound phase are effectively introduced by the master key. Therefore, in the case of fixed-TPKC, the available DoF should match the size of the master key. Figure 6 illustrates the first round of the AES-192 fixed-TPKC differential characteristic. Unlike AES-128, the size of AES-192 master key is 24 bytes, so the inbound phase in the differential characteristic should include the SB operations from the first round and the first two columns of the second-round SB operation. When searching for rebound-friendly differential characteristics, the probability p of the differential characteristic should be at least 2^{-192}. In the first round function, the AK operation applies the round key without passing through the SW operation. Therefore, we do not need to worry about consuming the plaintext's DoF, and there is no need to introduce constraints similar to those in case $\mathcal{C}^{\texttt{fixed-TPKC}}_{\texttt{AES-128}_2}$. Similar to AES-128, we control the probability of the inbound phase by introducing the following constraints:

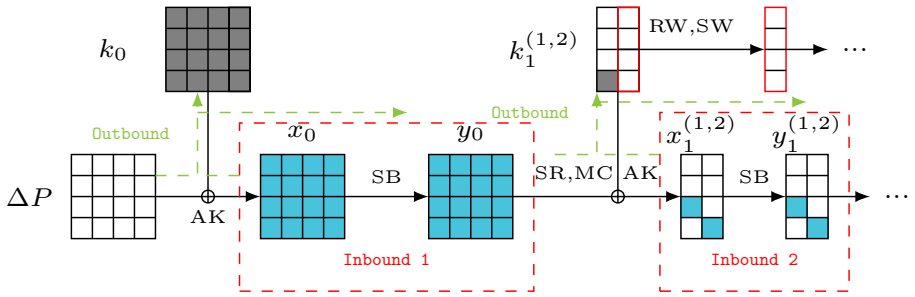


Figure 6: The first round and second round of the 5-round AES-192 fixed-target-plaintext key collision differential characteristic.

$$\begin{cases} \sum_{0\le i\le 15} \texttt{w}_7[0][i] + \sum_{0\le j\le 7} \texttt{w}_7[1][j] = NS_{in_7}, \\ \sum_{0\le i\le 15} \texttt{w}_6[0][i] + \sum_{0\le j\le 7} \texttt{w}_6[1][j] = NS_{in_6}, \end{cases} \tag{7}$$

where $\texttt{w}_7[0][i], \texttt{w}_6[0][i]$ and $\texttt{w}_7[0][j], \texttt{w}_6[0][j]$ are auxiliary variables controlling the probability for each S-box during the first round and the first two columns of the state in the second round of the SB operation. Let $\mathcal{C}^{\texttt{fixed-TPKC}}_{\texttt{AES-192}}$ represent constraint (7).

AES-256. In AES-256, the size of the master key is 32 bytes. Therefore, we can control the differences in the SB operations of the first two rounds. When searching for rebound-friendly differential characteristics, the probability p of the differential characteristic should be at least 2^{-256}. Similar to the previous two AES variants, we control the probability of

the inbound phase by introducing the following constraints:

$$\begin{cases} \sum_{0\le i\le 15} \mathtt{w}_7[0][i] + \sum_{0\le j\le 15} \mathtt{w}_7[1][j] = NS_{in_7}, \\ \sum_{0\le i\le 15} \mathtt{w}_6[0][i] + \sum_{0\le j\le 15} \mathtt{w}_6[1][j] = NS_{in_6}, \end{cases} \tag{8}$$

where $\mathtt{w}_7[0][i], \mathtt{w}_6[0][i]$ and $\mathtt{w}_7[0][j], \mathtt{w}_6[0][j]$ are auxiliary variables controlling the probability for each S-box during the first round and the second round of the SB operation. Let $\mathcal{C}^{\texttt{fixed-TPKC}}_{\texttt{AES-256}}$ represent constraint (8)

3.2.2 Constraints on AES Free-target-plaintext Key Collision

AES-128. As shown in Figure 7, this is the inbound phase of a new fixed-TPKC differential characteristic we identify for AES-128. In a fixed-TPKC, we can use the DoF from the plaintext, so the inbound phase does not need to start at the first round. Instead, we place it in rounds r_i and r_{i+1}, using the DoF from the two rounds of nonlinear transformations to calculate the key $k_{r_{i+1}}$.

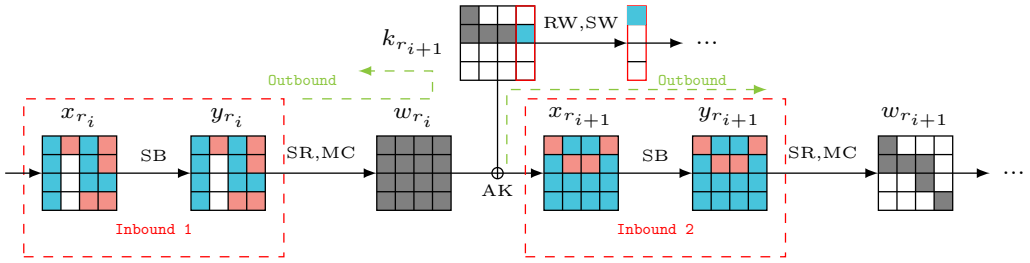


Figure 7: The r_i-th and r_{i+1}-th rounds of the 5-round AES-128 fixed-target-plaintext key collision differential characteristic.

Similar to the inbound phase of the fixed-TPKC attack described above, in the Inbound 1 phase, we use the DDT to calculate pairs that satisfy the known difference $\Delta x_{r_i}, \Delta y_{r_i}$, obtaining values for y_{r_i}. Then, by applying the SR and MC operations forward, we calculate w_{r_i}. In the Inbound 2 phase, we calculate the value of $x_{r_{i+1}}$ by assessing the DDT and obtain $k_{r_{i+1}} = w_{r_i} \oplus x_{r_{i+1}}$. Since the difference in $k_{r_{i+1}}[13]$ is non-zero, it undergoes the SW operation with a probability of 2^{-7}, requiring 2^7 DoF from the Inbound 1 phase to find pairs that satisfy this difference.

Similarly, suppose we want to search for a differential characteristic with probability p. In a free-TPKC attack on AES-128, the available DoF are $2^{128+128} = 2^{256}$, so $p \ge 2^{-256}$. To ensure that the differential probability in the outbound phase, p_{out}, remains greater than 2^{-64}, allowing a collision to be found with a time complexity of less than $2^{n/2}$, we set the inbound phase probability to $p_{in} = p/p_{out}$. Let $w_{in} = -\log_2 p_{in}$ represent the total differential probability weight in the inbound phase, which can be decomposed as $w_{in} = 7 \cdot NS_{in_7} + 6 \cdot NS_{in_6}$. To control the probability in the inbound phase, we add the following constraints:

$$\begin{cases} \sum_{0\le j\le 15} (\mathtt{w}_7[r_i][j] + \mathtt{w}_7[r_{i+1}][j]) = NS_{in_7}, \\ \sum_{0\le j\le 15} (\mathtt{w}_6[r_i][j] + \mathtt{w}_6[r_{i+1}][j]) = NS_{in_6}, \end{cases} \tag{9}$$

where $\mathtt{w}_7[r_i][j]$and $\mathtt{w}_6[r_i][j]$ are auxiliary variables controlling the probability for each S-box during the inbound phase. Let $\mathcal{C}^{\texttt{free-TPKC}}_{\texttt{AES-128}}$ represent constraint (9) **AES-192 and AES-256.** Similar to AES-128, we control the S-box probabilities during the inbound phase

of AES-192 and AES-256. The constraints are defined as follows:

$$\begin{cases} \sum_{0\le j\le 15}(\mathtt{w}_7[r_i][j] + \mathtt{w}_7[r_{i+1}][j]) + \sum_{0\le t\le 7}\mathtt{w}_7[r_{i+2}][t] = NS_{in_7}, \\ \sum_{0\le j\le 15}(\mathtt{w}_6[r_i][j] + \mathtt{w}_6[r_{i+1}][j]) + \sum_{0\le t\le 7}\mathtt{w}_6[r_{i+2}][t] = NS_{in_6}. \end{cases} \tag{10}$$

$$\begin{cases} \sum_{0\le j\le 15}(\mathtt{w}_7[r_i][j] + \mathtt{w}_7[r_{i+1}][j] + \mathtt{w}_7[r_{i+2}][j]) = NS_{in_7}, \\ \sum_{0\le j\le 15}(\mathtt{w}_6[r_i][j] + \mathtt{w}_6[r_{i+1}][j] + \mathtt{w}_6[r_{i+2}][j]) = NS_{in_6}. \end{cases} \tag{11}$$

Here, $\mathtt{w}_7$ and $\mathtt{w}_6$ are auxiliary variables controlling the probability for each S-box during the inbound phase. Let $\mathcal{C}^{\texttt{free-TPKC}}_{\texttt{AES-192}}$ and $\mathcal{C}^{\texttt{free-TPKC}}_{\texttt{AES-256}}$ represent constraints (10) and (11), respectively.

3.2.3 Algorithm for Searching Rebound-Friendly Key Collision Differential Characteristics in AES

To search for rebound-friendly key collision differential characteristics, we add relevant constraints to the fixed-probability differential characteristic search model. Based on the analysis in Sections 3.2.1 and 3.2.2, we convert the inbound phase constraints $\mathcal{C}^{\texttt{attack-type}}_{\texttt{version}}$, determined by the AES version and the type of key collision attack, into CNF clauses. We define this process as the function $\texttt{addICclause}(r_i, \mathtt{w}_7, \mathtt{w}_6, NS_{in_7}, NS_{in_6}, \texttt{version}, \texttt{attack-type})$, which returns the corresponding CNF clauses. Algorithm 2, named findRFKCcharacteristic, is designed to find rebound-friendly key collision differential characteristics for AES, with input parameters $\mathtt{R}, r_i, NS_7, NS_6, NS_{in_7}, NS_{in_6}$, version, and attack-type. If a differential characteristic is found, the algorithm returns a rebound-friendly differential characteristic; otherwise, it returns NULL.

All experiments in this paper were conducted on a CPU with the following specifications: 11th Gen Intel(R) Core(TM) i9-11900 @ 2.50GHz and 32 GB RAM. During the differential characteristic search, 8 cores were used, while a single core was used for computing collision pairs.

Algorithm 2: Searching for rebound-friendly key collision differential characteristics in AES

Input: The number of rounds R, the starting round of the inbound phase r_i, the number of S-boxes with differential probabilities of 2^{-7} and 2^{-6} NS_7, NS_6, the number of S-boxes with different probabilities NS_{in_7} and NS_{in_6} in the inbound phase, the AES version (128,192,256), and the attack type attack-type of key collision attack (fixed-TPKC or free-TPKC).

Output: A rebound-friendly key collision differential characteristic with probability $p = 2^{-7\cdot NS_7 - 6\cdot NS_6}$, where the inbound phase starts at r_i with a probability of $p_{in} = 2^{-7\cdot NS_{in_7} - 6\cdot NS_{in_6}}$, and the outbound phase has a probability of p/p_{in}.

```
Initialize an empty SAT model M;
M ← findFPKCcharacteristic(R, NS_7, NS_6, version);
M.clause ← addICclauses(r_i, w_7, w_6, version, NS_in7, NS_in6, attack-type);
solve the SAT model M;
if the problem is satisfiable then
    dc ← (dx, dy, dz, dk);
    return dc;
return NULL;
```

4 Rebound-based Key Collision Attack for AES and Kiasu-BC

4.1 The New Rebound Attack Framework

The overall framework for rebound-based key collision attack.

1. **Search Rebound-Friendly Differential Characteristic.** Instead of a truncated differential characteristic, identify a *rebound-friendly* differential characteristic. In such a characteristic, the inbound phase has a lower probability with sufficient DoF available for the outbound phase. The outbound phase has a higher probability, facilitating the discovery of colliding key pairs.

2. **Compute Starting Points in the Inbound Phase.** In the inbound phase, generate sufficient starting points based on fixed differentials, using active byte differentials and arbitrary values for inactive bytes.

3. **Calculate Collision Pairs in the Outbound Phase.** Using the starting points obtained from the inbound phase, perform forward and backward computations in the outbound phase to determine colliding key pairs and their corresponding plaintext pairs (for semi-free-start collisions).

A New Rebound Attack Framework for Key Collision Attack. Inspired by the Super S-box technique [LMR+09, GP10] and building upon the traditional rebound attack framework [TSI+24a], we propose a new rebound attack framework incorporating a time-memory tradeoff strategy. This framework divides the attack into two distinct phases: the offline phase and the online phase.

- **Offline Phase**: During this phase, we extend the inbound phase to cover additional rounds of the key schedule or round function. By doing so, we precompute and store starting points that satisfy the differential characteristics of the inbound phase in a precomputation table. This approach allows the inbound phase to cover more rounds. Although memory consumption increases, it correspondingly increases the differential probability of the outbound phase.

- **Online Phase**: In this phase, we leverage the precomputed table to efficiently fulfill the differential requirements of the outbound phase. Since the stored entries already satisfy differential characteristics of the inbound phase, the differential probability in the outbound phase is significantly increased compared to traditional rebound attacks. This enhancement directly reduces the time complexity, as the attack primarily depends on the outbound phase's success probability. By combining the precomputed pairs with the remaining steps, we systematically identify colliding key pairs that satisfy the full differential characteristic.

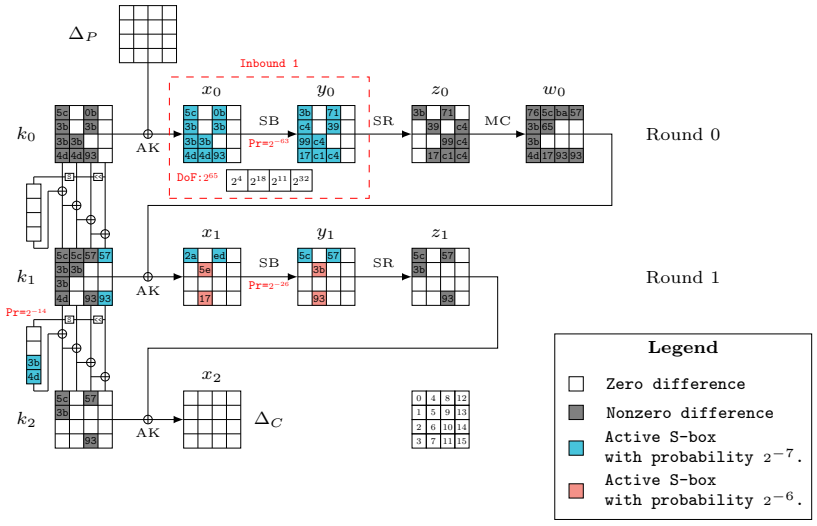
To provide a detailed explanation of the attack process in the new rebound attack framework, we will find the 2-round AES-128 fixed-target-plaintext key collision attack in our new rebound attack framework and traditional rebound attack frameworks. The detailed attack process can be found in Subsection 4.2.1.

4.2 Improved Key Collision Attacks on AES

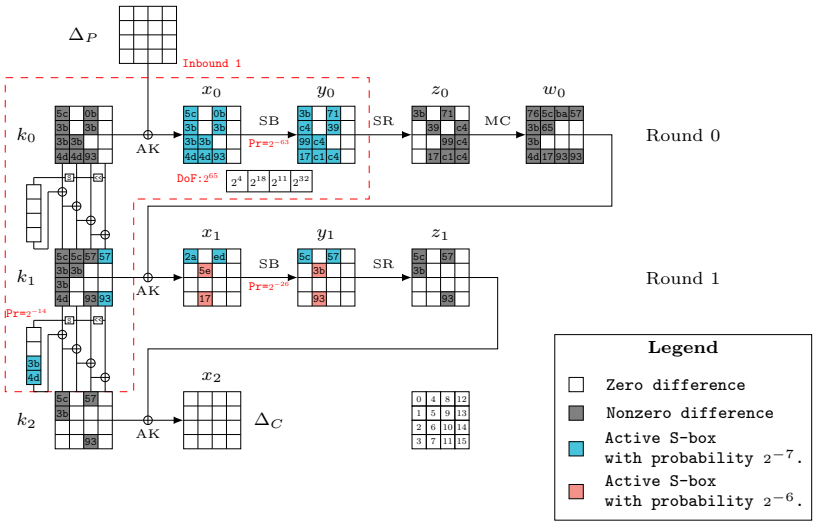
In this section, we first present a practical 2-round AES-128 fixed-TPKC attack using our improved automatic method to discover a new differential characteristic and the new proposed rebound attack framework. Detailed attack procedures for both frameworks are provided for comparison. Subsequently, we demonstrate why the 2-round AES-128 fixed-TPKC attack reported by Taiyama et al. [TSI+24a] is, in fact, a free-TPKC attack.

Additionally, we propose a practical 5-round AES-192 fixed-TPKC attack, further validating the effectiveness of our framework. A detailed comparison is shown in Table 1. All the differential characteristics used in the key collision attacks can be found in Appendix D.

4.2.1 New Fixed-target-plaintext Key Collision Attack on 2-round AES-128



(a) The traditional rebound attack framework.



(b) New rebound attack framework.

Figure 8: New fixed-target-plaintext key collision attack on 2-round AES-128.

Based on the constraints added in the Subsection 3.2.1, in this subsection, we propose a new fixed-TPKC attack on 2-round AES-128. Using the improved automatic search method, we set the input parameters of the findRFcharacteristic function as $(\mathtt{R}, r_i, NS_7, NS_6, NS_{in_7}, NS_{in_6}, \mathtt{version}, \mathtt{attack\text{-}type}) = (2, 0, 13, 2, 9, 0, 128, \mathtt{fixed\text{-}TPKC})$. The new 2-round AES-128 fixed-TPKC differential characteristic is shown in Figure 8.

In the fixed-TPKC attack, where the plaintext is fixed in advance. Given the differences Δx_0 and Δy_0 in the inbound phase, we can determine values for the active bytes by assessing the DDT, while inactive bytes can take arbitrary values. Therefore, in x_0 and y_0, we can

generate up to 2^4 pairs for the first column, 2^{18} pairs for the second column, 2^{11} pairs for the third column, and 2^{32} pairs for the fourth column. This means we can generate up to 2^{65} starting points (x_0, y_0) in the inbound phase, which provides sufficient freedom to launch a `fixed-TPKC` attack.

We first present the attack procedure of the traditional rebound attack framework without employing a time-memory tradeoff strategy.Then we give the attack procedure of our proposed new rebound attack framework utilizing a time-memory tradeoff strategy.

The Traditional Rebound Attack Framework. Corresponding to Figure 8 (a), the process of the traditional rebound attack framework for finding collision key pairs is as follows:

1. Generate values x_0 and y_0 in the Inbound 1 phase that satisfy the differences Δx_0 and Δy_0. Compute $k_0 = P \oplus x_0$ to obtain the value of k_0. Since the `SW` operation in the first round key schedule is inactive, there is no need to consume `DoF` in the plaintext to generate values of pairs that satisfy the `SW` difference.

2. Calculate $w_0 = \texttt{MC} \circ \texttt{SR}(y_0)$. With k_0 known, calculate $k_1 = \texttt{KS}(k_0)$ and $x_1 = w_0 \oplus k_1$. Then compute $y_1 = \texttt{SB}(x_1)$ and $y_1' = \texttt{SB}(x_1 \oplus \Delta x_1)$. If $y_1 \oplus y_1' \neq \Delta y_1$, return to step 1.

3. Calculate $sk_1 = \texttt{SW} \circ \texttt{RW}(k_1[12, 13, 14, 15])$ and $sk_1' = \texttt{SW} \circ \texttt{RW}(k_1[12, 13, 14, 15] \oplus \Delta k_1[12, 13, 14, 15])$. If $sk_1 \oplus sk_1' \neq \Delta sk_1$, return to step 1.

4. After obtaining a valid y_1, a colliding key pair $(k_0, k_0 \oplus \Delta k_0)$ is successfully identified.

Complexity Analysis. Since the probability of the outbound phase is 2^{-40}, we need to repeat the above attack process 2^{40} times. Therefore, the time complexity is equivalent to 2^{40} executions of 2-round `AES-128` encryption, while the memory complexity remains negligible.

New Rebound Attack Framework. Corresponding to Figure 8 (b), the process of the new rebound attack framework for finding collision key pairs is as follows:

- **Offline Phase**: By observing Figure 8 (b), it is noted that although there are no active `SW` operations in the key schedule phase from $k_0 \to k_1$, the bytes $k_1[12, 15]$ become active during the $k_1[12, 15] \to \texttt{SW}(k_2[12, 15])$ phase. The update rules for $k_1[12, 15]$ are:

$$k_1[12] = k_0[0] \oplus k_0[4] \oplus k_0[8] \oplus k_0[12] \oplus \texttt{SW}(k_0[13]) \oplus \texttt{rcon}_0,$$
$$k_1[15] = k_0[3] \oplus k_0[7] \oplus k_0[11] \oplus k_0[15] \oplus \texttt{SW}(k_0[12]).$$

  Since k_0 depends on $k_0 = P \oplus x_0$, to precompute $k_0[0, 3, 4, 7, 8, 12, 13, 15]$, we need to determine $x_0[0, 3, 4, 7, 8, 12, 13, 15]$ in advance. The values of x_0 are influenced by the `SB` operation, resulting in 2^{36} possible values for $x_0[0, 3, 4, 7, 8, 12, 13, 15]$. Using $k_0 = P \oplus x_0$, $k_0[0, 3, 4, 7, 8, 12, 13, 15]$ also has 2^{36} possibilities. Based on the update functions of $k_1[12, 15]$, we precompute all valid values satisfying these constraints and store them in a precomputed table $\texttt{Tab}_{\texttt{pre}}$ of size 2^{γ}.

- **Online Phase:**

  1. Select a valid entry from $\texttt{Tab}_{\texttt{pre}}$, which corresponds to $x_0[0, 3, 4, 7, 8, 12, 13, 15]$. For the remaining unfixed bytes of x_0, randomly select a value that satisfies its differential characteristics. Once x_0 is fixed, k_0 is also fixed via $k_0 = P \oplus x_0$.

2. Calculate $w_0 = \texttt{MC} \circ \texttt{SR}(y_0)$. With k_0 known, compute $k_1 = \texttt{KS}(k_0)$ and $x_1 = w_0 \oplus k_1$. Then derive $y_1 = \texttt{SB}(x_1)$ and $y_1' = \texttt{SB}(x_1 \oplus \Delta x_1)$. If $y_1 \oplus y_1' \neq \Delta y_1$, return to Step 1.
3. After obtaining a valid y_1, a colliding key pair $(k_0, k_0 \oplus \Delta k_0)$ is successfully identified.

Complexity Analysis. In the above steps, since we have precomputed $x_0[0,3,4,7,8,12,13,15]$ to satisfy the probabilistic constraints in the key schedule, we only need to fulfill the outbound phase. Testing reveals that $x_0[0,3,4,7,8,12,13,15]$ has 2^{22} possible values (i.e., $\gamma = 22$). The probability of the outbound phase is 2^{-26}, requiring the attack process to be repeated 2^{26} times. It is worth noting that although we need to calculate 2^{36} possible values for $x_0[0,3,4,7,8,12,13,15]$, the time complexity of this calculation is negligible compared to the `AES` round function, as it only involves two `SW` operations and nine XOR operations. The time spent in the offline phase can be considered negligible relative to the computation time in the online phase. Thus, the time complexity is equivalent to 2^{26} executions of 2-round `AES-128` encryption. The memory complexity involves storing the precomputed values of $x_0[0,3,4,7,8,12,13,15]$ that satisfy the inbound phase differential characteristic, with a size of 2^{22}. To validate this process, we provide a practical 2-round `AES-128` key collision pair in Table 2.

Table 2: Pair of the fixed-target-plaintext key collision attack on 2-round `AES-128`

i	$\texttt{Plaintext}_i$	$\texttt{Key}_i$	$\texttt{Ciphertext}_i$
1	00 00 00 00	60 80 37 00	02 e8 fd 24
	00 00 00 00	de 10 85 28	bc 38 22 1f
	00 00 00 00	d0 de 15 71	b6 95 9b aa
	00 00 00 00	d9 93 7a 22	03 d9 ba 55
2	00 00 00 00	3c 80 3c 00	02 e8 fd 24
	00 00 00 00	e5 10 be 28	bc 38 22 1f
	00 00 00 00	eb e5 15 71	b6 95 9b aa
	00 00 00 00	94 de e9 22	03 d9 ba 55

4.2.2 Taiyama et al.'s Fixed-target-plaintext Key Collision Attack on 2-round `AES-128` [TSI+24a]

As shown in Figure 9, the first-round differential characteristic of the 2-round `AES-128` `fixed-TPKC` found by Taiyama et al. According to the analysis in Section 3.2.1, we can see that this characteristic qualifies only as a `free-TPKC`, as it uses the `DoF` from the IV to satisfy the difference in $k_0[12]$ after nonlinear transformations.

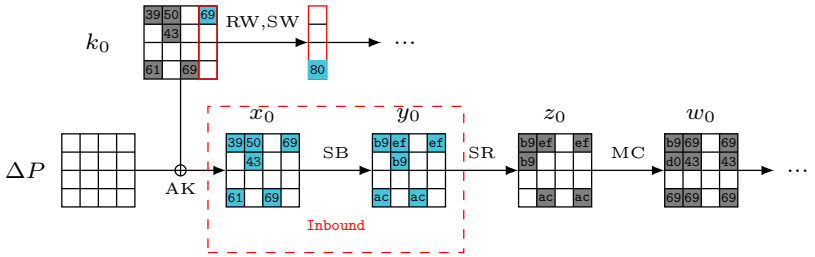


Figure 9: The first-round differential characteristic of the 2-round `AES-128` fixed-target-plaintext key collision by Taiyama et al. [TSI+24a].

In the inbound phase, we can deduce the full state values x_0 and y_0 by evaluating the DDT. We compute the full state values of k_0 using $k_0 = P \oplus x_0$. Since the first-round key undergoes the nonlinear SW operation with an active byte $k_0[12]$, with input-output differentials of (0x69, 0x08), only two pairs $(k_0[12], sk_0[12])$ satisfy this difference: (0x02, 0x77) and (0x6b, 0x7f). We calculate $(x_0, x_0') = (\texttt{0x72}, \texttt{0x1b})$ to satisfy the differentials $\Delta x_0[12], \Delta y_0[12]$. Therefore, to obtain pairs that satisfy the input difference of $k_0[12]$, $P[12]$ must be one of

$$\begin{aligned}
\texttt{0x70} &= \texttt{0x72} \oplus \texttt{0x02},\\
\texttt{0x05} &= \texttt{0x72} \oplus \texttt{0x77},\\
\texttt{0x19} &= \texttt{0x1b} \oplus \texttt{0x02},\\
\texttt{0x6c} &= \texttt{0x1b} \oplus \texttt{0x77}.
\end{aligned}$$

This requirement consumes the DoF of the plaintext, making it a free-TPKC attack. We find a new 2-round AES-128 fixed-TPKC differential characteristic, as shown in Table 8. Using this characteristic, we launch a 2-round AES-128 free-TPKC attack. The time complexity of the attack is 2^{32}.

4.2.3 New Fixed-target-plaintext Key Collision Attack on 5-round AES-192

Using our automatic search method, we identify new AES-192 fixed-TPKC and free-TPKC differential characteristics and launch corresponding key collision attacks on AES-192 using these characteristics. The new 5-round AES-192 fixed-TPKC differential characteristic is shown in Figure 10. The total probability of this differential characteristic is 2^{-185}. Using this new differential characteristic combined with the new rebound attack framework, we launch a practical 5-round AES-192 fixed-TPKC attack. The attack procedure of the new rebound attack framework is as follows:

- **Offline Phase:** To satisfy the differential characteristics $\Delta x_1[8, 13, 15] \xrightarrow{\texttt{SB}} \Delta y_1[8, 13, 15]$ and $\Delta k_2[15] \rightarrow \texttt{SW}(k_2[15])$ in the 5-round AES-192 fixed-TPKC attack, we need to utilize the DoF in the first-round subkey values $k_0, k_1[0-7]$ of AES-192. Based on the key schedule and round function of AES-192, we can determine which bytes of the first-round subkey must be used to satisfy the differential characteristics of the four S-boxes mentioned above. The relationships between these key bytes are as follows:

$$\begin{aligned}
x_1[8] &= k_0[0] \oplus \texttt{SW}(k_1[4]) \oplus w_0[8],\\
x_1[13] &= k_0[1] \oplus \texttt{SW}(k_1[5]) \oplus k_0[5] \oplus w_0[13],\\
x_1[15] &= k_0[3] \oplus \texttt{SW}(k_1[7]) \oplus k_0[7] \oplus w_0[15],\\
k_2[15] &= k_1[7] \oplus k_1[3] \oplus k_0[15] \oplus k_0[11] \oplus k_0[7] \oplus k_0[3] \oplus \texttt{SW}(k_1[7]).
\end{aligned}$$

  To identify all starting points that satisfy the differential characteristics $\Delta x_1[8, 13, 15] \xrightarrow{\texttt{SB}} \Delta y_1[8, 13, 15]$ and $\Delta k_2[15] \rightarrow \Delta\texttt{SW}(k_2[15])$, we need to compute the values of $k_0[0, 1, 3, 5, 7, 11, 15]$, $k_1[3, 4, 5, 7]$, $w_0[13, 15]$. These values can be easily derived from the first-round SB operation and the first two columns of the second-round SB operation, yielding a total of 2^{32} DoF. Since the probability of the differential characteristics $\Delta x_1[8, 13, 15] \xrightarrow{\texttt{SB}} \Delta y_1[8, 13, 15]$ and $\Delta k_2[15] \rightarrow \texttt{SW}(k_2[15])$ is 2^{-27}, the size of the precomputed table $\texttt{Tab}_{\textbf{pre}}$ is $2^{32-27} = 2^5$.

- **Online Phase:** Exploit $\texttt{Tab}_{\textbf{pre}}$ and the remaining DoF in x_0, x_1 to fulfill the differential characteristics $\Delta x_2[3, 15] \xrightarrow{\texttt{SB}} \Delta y_2[3, 15]$, $\Delta x_3[7] \xrightarrow{\texttt{SB}} \Delta y_3[7]$, and $\Delta x_4[3, 15] \xrightarrow{\texttt{SB}} \Delta y_4[3, 15]$. This involves:

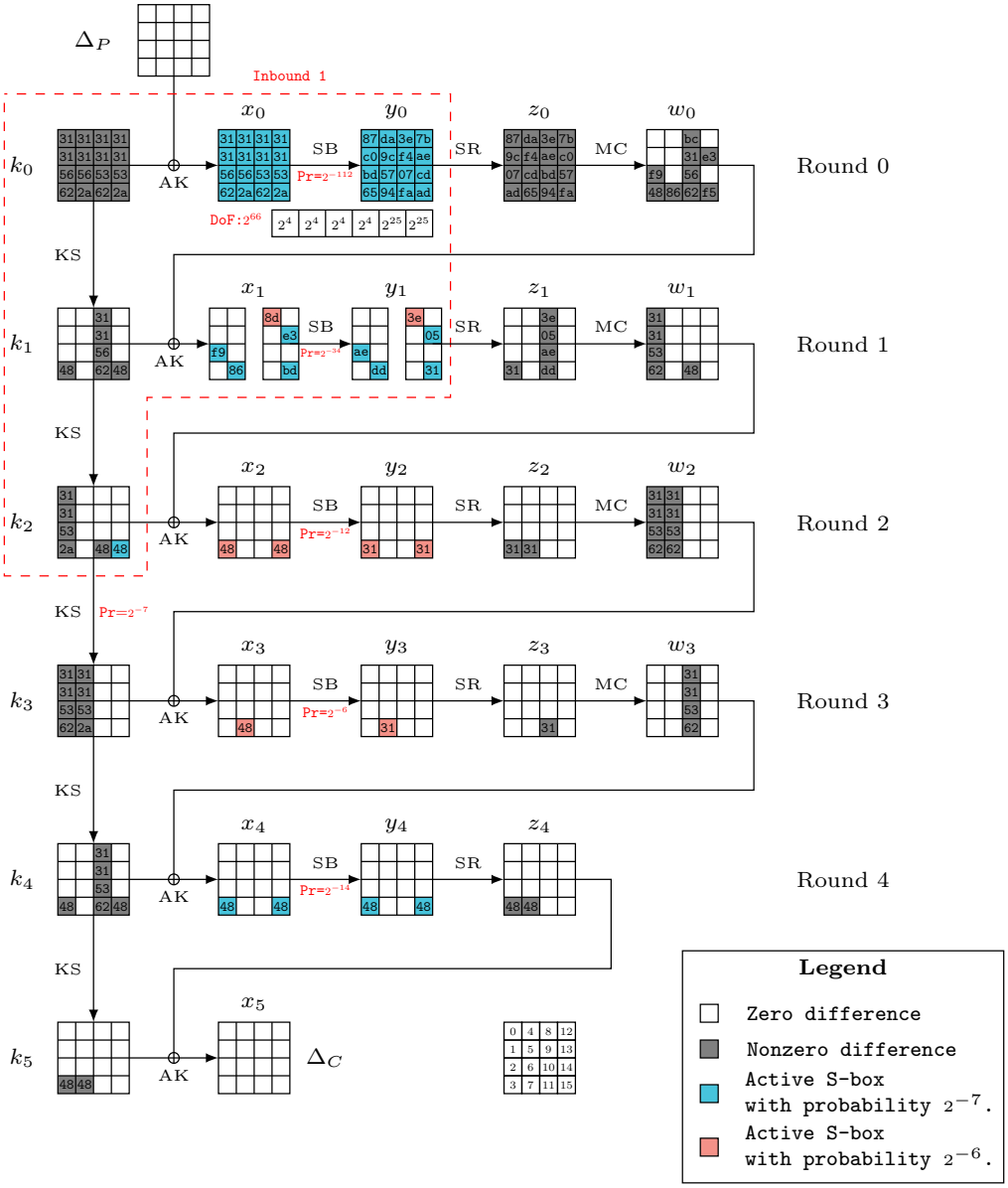


Figure 10: New fixed-target-plaintext key collision attack on 5-round AES-192.

1. Iterating through entries in $\texttt{Tab}_{\textbf{pre}}$ to fix x_0 and partial round keys.
2. Propagating constraints forward through the cipher using the known $x_1[0,1]$ values.
3. Verifying consistency with the required differential characteristics in subsequent rounds.

Complexity Analysis. The probability of satisfying the outbound phase differential characteristics is 2^{-32}, necessitating 2^{32} iterations of the attack process to achieve a collision. Thus, the time complexity is equivalent to 2^{32} executions of 5-round AES-128 encryption. The memory complexity is the size of the precomputed table, which is 2^5. To demonstrate the practical validity of this approach, we provide a practical colliding key pair for 5-round AES-192 in Table 3.

Table 3: Pair of the fixed-target-plaintext key collision on 5-round AES-192

i	$\texttt{Plaintext}_i$	$\texttt{Key}_i$	$\texttt{Ciphertext}_i$
1	00 00 00 00	49 ab ea fb 0e 14	3c 3e 87 e3
	00 00 00 00	36 f8 f0 d9 7f 00	d8 4b 21 4a
	00 00 00 00	59 02 76 fb d7 89	2a ae 98 2c
	00 00 00 00	9c ba a7 a3 a1 28	79 5d 96 96
2	00 00 00 00	78 9a db ca 0e 14	3c 3e 87 e3
	00 00 00 00	07 c9 c1 e8 7f 00	d8 4b 21 4a
	00 00 00 00	0f 54 25 a8 d7 89	2a ae 98 2c
	00 00 00 00	fe 90 c5 89 e9 28	79 5d 96 96

4.3 Key Collision Attacks on Kiasu-BC

4.3.1 Description of Kiasu-BC

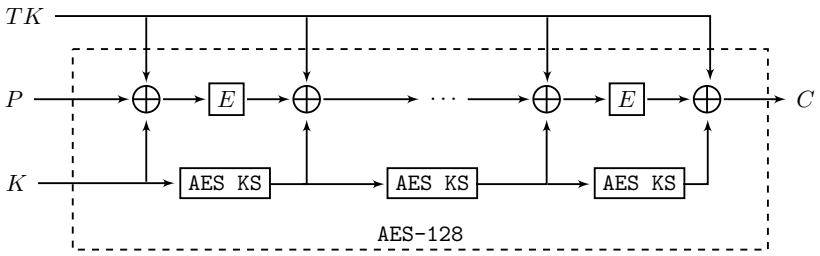


Figure 11: The Kiasu-BC tweakable block cipher based on AES-128

Kiasu-BC is a tweakable block cipher proposed by Jean et al. [JNP14a], designed based on the AES-128, with additional modifications inspired by the TWEAKEY framework [JNP14b] introduced at ASIACRYPT 2014. As illustrated in Figure 11, Kiasu-BC takes three inputs: a 64-bit tweak T, a 128-bit plaintext P, and a 128-bit key K. Different from the AES-128 encryption, Kiasu-BC introduces an XOR operation with the 64-bit tweak T to the first two rows of the AES internal state after ARK in the round function, including after the pre-whitening key addition. Notably, Kiasu-BC does not utilize a tweak schedule, and the same tweak T is applied in every round.

We aim to investigate whether the tweak in Kiasu-BC can serve as a source of DoF for rebound-based attacks, thereby enabling attacks on additional rounds. Since Kiasu-BC does not employ a tweak schedule, we select it as the subject of our study. Assuming

that the attacker can introduce differences in the tweak, we propose both `fixed-TPKC` and `free-TPKC` attacks on `Kiasu-BC`.

4.3.2 Fixed-target-plaintext Key Collision Attack on 3-Round `Kiasu-BC`

In this section, assuming the attacker can independently choose values of the tweak, we leverage the additional `DoF` provided by the tweak to launch a `fixed-TPKC` attack on 3-round `Kiasu-BC`, as illustrated in Figure 12. The overall probability of this differential

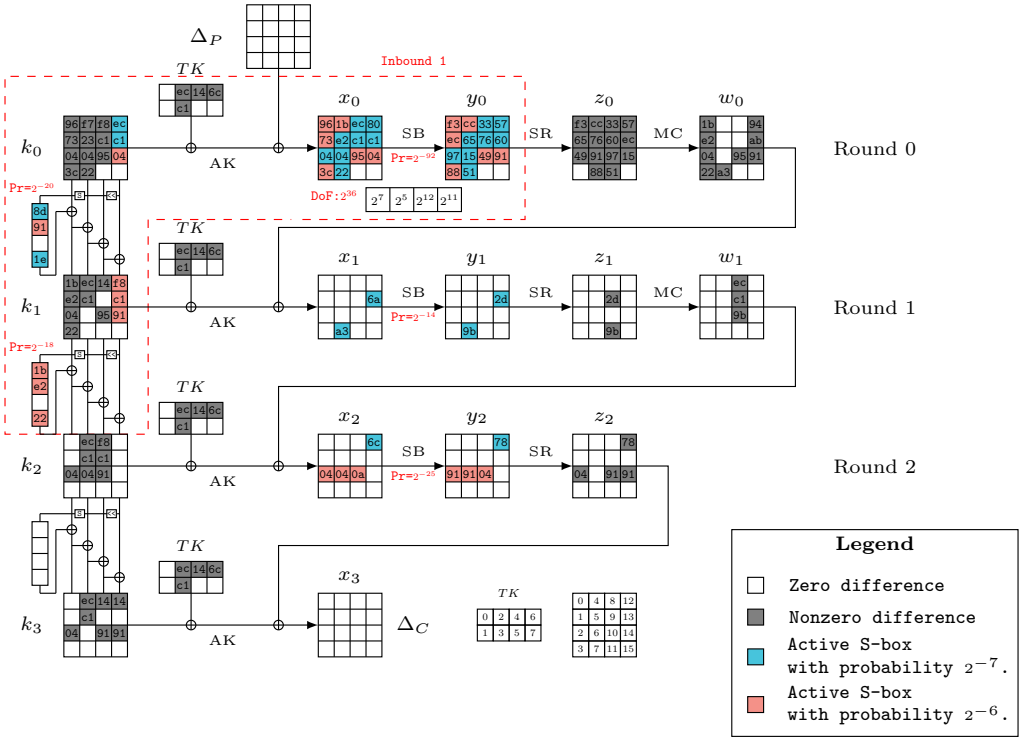


Figure 12: Fixed-target-plaintext key collision attack on 3-round `Kiasu-BC`.

characteristic is 2^{-169}. The first round `SB` operation provides 2^{36} `DoF`, which is insufficient to satisfy the differential characteristics of the outbound phase. Therefore, during the attack, we utilize the `DoF` provided by the tweak TK, which, being 64 bits in size, offers an additional 2^{64} `DoF`. The attack procedure of the new rebound attack framework is as follows:

- **Offline Phase:** Referring to the differential characteristics in Figure 12, it is observed that $\Delta k_0[12, 13, 14] \rightarrow \Delta\texttt{SW}(k_0[12, 13, 14])$ is active, where $\Delta k_0[12, 13] \neq \Delta x_0[12, 13]$ and $\Delta\texttt{SW}(k_0[12, 13]) \neq \Delta y_0[12, 13]$, leading to a contradiction. However, $TK[6, 7]$ effectively resolves this contradiction. Therefore, the differential characteristic $\Delta k_0[12, 13, 14] \rightarrow \Delta\texttt{SW}(k_0[12, 13, 14])$ can be directly satisfied by $TK[6, 7]$. To satisfy the differential characteristic $\Delta k_1[12, 13, 14] \rightarrow \Delta\texttt{SW}(k_1[12, 13, 14])$, we need to utilize the `DoF` of the tweak. Based on the `AES-128` key schedule and round function, the following relationships hold:

$$
\begin{aligned}
k_1[12] &= k_0[12] \oplus k_0[8] \oplus k_0[4] \oplus k_0[0] \oplus \texttt{SW}(k_0[13]) \oplus \texttt{rcon}_0, \\
k_1[13] &= k_0[13] \oplus k_0[9] \oplus k_0[5] \oplus k_0[1] \oplus \texttt{SW}(k_0[14]) \oplus \texttt{rcon}_1, \\
k_1[15] &= k_0[14] \oplus k_0[10] \oplus k_0[6] \oplus k_0[2] \oplus \texttt{SW}(k_0[15]) \oplus \texttt{rcon}_2.
\end{aligned}
$$

Since the DoF of the tweak can be introduced in each round, we do not need to compute all starting points that satisfy the differential characteristic $\Delta k_1[12,13,14] \rightarrow \Delta\mathtt{SW}(k_1[12,13,14])$. Here, we only need to utilize the DoF of $TK[4,5]$ and $x_0[8,9,15]$. Regarding the differential characteristic $\Delta x_1[7,13] \xrightarrow{\mathtt{SB}} \Delta y_1[13,15]$, we can respectively utilize the DoFof $y_0[11]$ and $x_0[7], y_0[3,4,9,14]$ to satisfy this differential characteristic. Since the DoF required by the differential characteristic $\Delta k_0[12,13,14] \rightarrow \Delta\mathtt{SW}(k_0[12,13,14])$ are independent from those needed for $\Delta x_1[7,13] \xrightarrow{\mathtt{SB}} \Delta y_1[13,15]$, they can be computed independently. Thus, we can generate a precomputed table $\mathtt{Tab_{pre}}$ that satisfies the differential characteristics $\Delta k_0[12,13,14] \rightarrow \Delta\mathtt{SW}(k_0[12,13,14])$ and $\Delta x_1[7,13] \xrightarrow{\mathtt{SB}} \Delta y_1[13,15]$, with a size of 2^{10}.

- **Online Phase:** Using the precomputed table $\mathtt{Tab_{pre}}$ and TK, along with the remaining degrees of freedom in x_0, we perform the following steps:
  1. Iterate over the fixed portions of x_0 and TK from the precomputed table, and traverse the remaining degrees of freedom in TK, x_0, and y_0.
  2. Verify consistency with the required differential characteristics in the subsequent rounds.

Complexity Analysis. The time complexity of the offline phase can be considered negligible. Therefore, the time complexity is equivalent to 2^{26} executions of 3-round Kiasu-BC encryption. The memory complexity is the size of the precomputed table, which is 2^{10}. To demonstrate the practical validity of this approach, we provide a practical colliding key pair of 3-round Kiasu-BC in Table 4.

Table 4: Pair of the fixed-target-plaintext key collision on 3-round Kiasu-BC

i	$\mathtt{Plaintext}_i$	$\mathtt{Tweak}_i$	$\mathtt{Key}_i$	$\mathtt{Ciphertext}_i$
1	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	98 c5 b5 f9 08 30 7b 33	0e de bb 59 7b dc 91 59 3f 68 00 c6 0e 52 00 ac	af 2e a5 98 16 38 bf 01 27 e6 e1 eb 80 90 68 95
2	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	98 29 a1 95 08 f1 7b 33	98 29 43 b5 08 ff 50 98 3b 6c 95 c2 32 70 00 ac	af 2e a5 98 16 38 bf 01 27 e6 e1 eb 80 90 68 95

4.3.3 Free-target-plaintext Key Collision Attack on 6-Round Kiasu-BC

The 6-round Kiasu-BC rebound-friendly differential characteristic, as shown in Figure 13. The total probability of this differential characteristic is 2^{-174}. The inbound phase consists of the SB operations in the first and second rounds, with a probability of 2^{-136}, while the remaining portion serves as the outbound phase with a probability of 2^{-38}.

The process of finding a colliding key pair is stated as follows:

1. In Figure 13 Inbound 1, generate (x_2, y_2) that satisfy the differences Δx_2 and Δy_2. In Inbound 2, generate (x_3, y_3) that satisfy the differences Δx_3 and Δy_3. Compute $w_2 = \mathtt{MC} \circ \mathtt{SR}(y_2)$, and $k_3 = x_3 \oplus w_2 \oplus TK$.
2. Compute x_2 forward and y_3 backward, and calculate k_3 both forward and backward. In the SB operations of Rounds 0, 1, and 5, the differential probability is 2^{-38}, which requires consuming 2^{37} DoF from the inbound phase to satisfy these differences. If the conditions are not met, return to step 1.

3. Once a valid (x_0, y_5) is obtained, the resulting `free-TPKC` key pair is $(k_0, k_0 \oplus \Delta k_0)$, with the corresponding plaintext given by $k_0 \oplus x_0$.

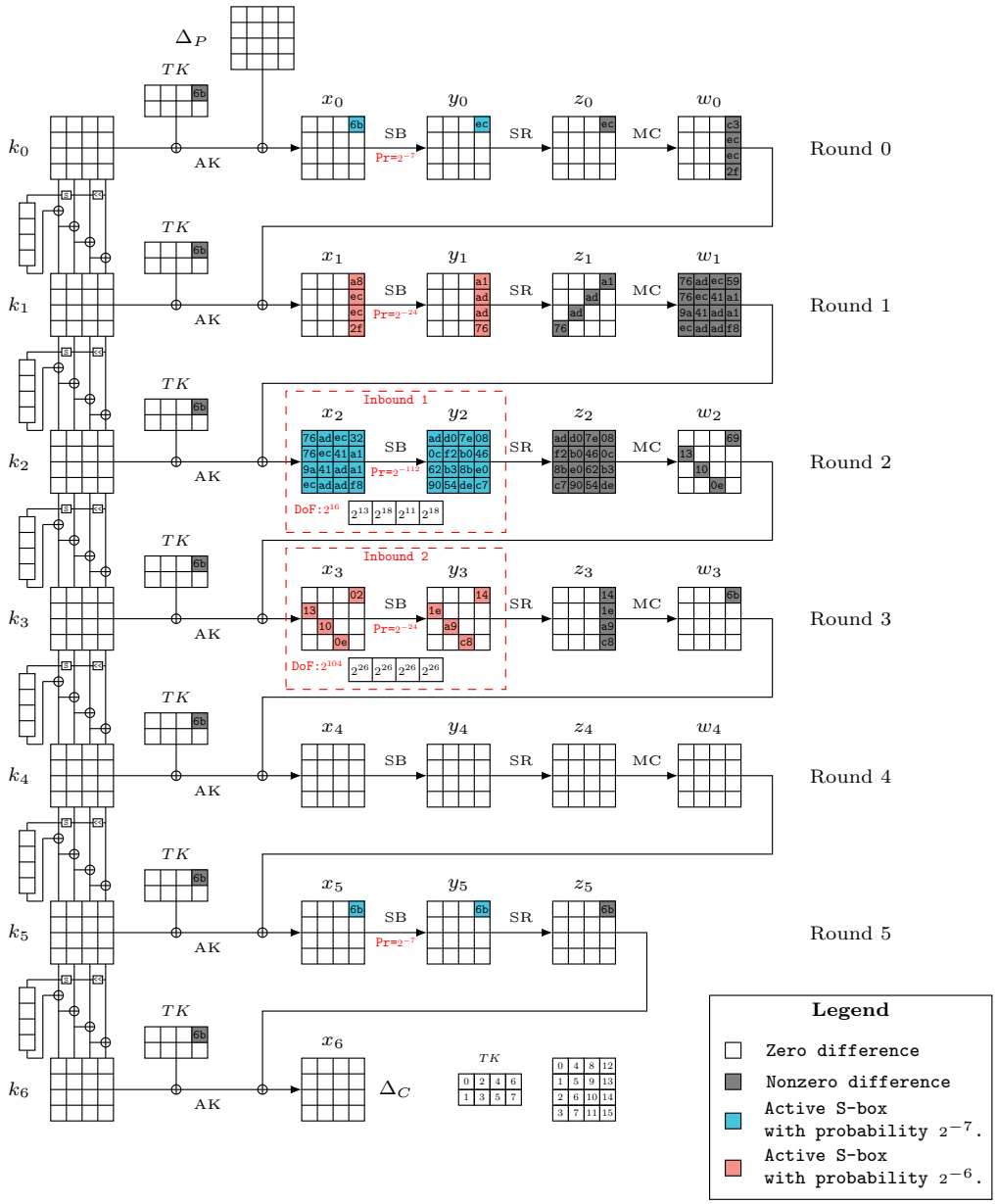


Figure 13: Free-target-plaintext key collision attack on 6-round `Kiasu-BC`

Complexity Analysis. The probability of satisfying the outbound phase differential characteristics is 2^{-37}. Theoretically, it would take 2^{38} iterations of the attack process to find a collision. However, since we can directly utilize the degrees of freedom of $TK[6, 7]$ to satisfy the differential characteristics $\Delta x_1[12, 13] \xrightarrow{\texttt{SB]}} \Delta y_1[12, 13]$, $\Delta x_0[12] \xrightarrow{\texttt{SB]}} \Delta y_0[12]$, or $x_5[12] \xrightarrow{\texttt{SB]}} \Delta y_5[12]$, the overall time complexity is reduced to 2^{25} executions of 6-round `Kiasu-BC` encryption. The memory complexity is negligible. To demonstrate the

effectiveness of this process, we present a practical collision key pair of 6-round Kiasu-BC in Table 5.

Table 5: Pair of the free-target-plaintext key collision on 6-round Kiasu-BC

i	Plaintext$_i$	Tweak$_i$	Key$_i$	Ciphertext$_i$
1	0f b1 53 0f b3 3a bd cd 98 91 1c 90 d4 d2 8b 53	00 00 00 00 00 00 00 00	76 7e 87 a7 7f 58 ea 30 41 5a ef 04 d6 34 be f5	9d 0b ee 3a 5a 4a 81 e9 5b 09 f4 fb 6d d4 24 bd
2	0f b1 53 0f b3 3a bd cd 98 91 1c 90 d4 d2 8b 53	00 00 00 6b 00 00 00 00	76 7e 87 a7 7f 58 ea 30 41 5a ef 04 d6 34 be f5	9d 0b ee 3a 5a 4a 81 e9 5b 09 f4 fb 6d d4 24 bd

5 Conclusion and Future Work

This paper enhances the SAT model for discovering rebound-friendly key collision differential characteristics as proposed in [TSI+24a]. To identify key collision differential characteristics, we control the differential probability in the inbound phase to facilitate the discovery of rebound-friendly key collision differential characteristics. Additionally, we propose a new rebound attack framework for key collision attacks by introducing a time-memory trade-off strategy. Using our improved automatic search method combined with the new rebound attack framework, we are able to find new key collision differential characteristics that can be used to launch practical key collision attacks on different versions of AES, achieving improvements over previous results.

Additionally, we summarize the sources of degrees of freedom in rebound-based attacks. In key collision attacks or collision/semi-free start collision attacks in DM hashing mode, since differences can only be introduced at the keys, it is challenging to initiate collision attacks by leveraging freedom in truncated differentials. Therefore, bit-oriented key collision attacks become the only viable approach, where active byte freedoms in the internal state and freedoms in non-active bytes are utilized to launch attacks. Future work could explore the resistance of key collision attacks on authenticated encryption algorithms and further investigate the application of rebound-based cryptanalysis in finding chosen-key distinguishers.

References

[ADG+22] Ange Albertini, Thai Duong, Shay Gueron, Stefan Kölbl, Atul Luykx, and Sophie Schmieg. How to abuse and fix authenticated encryption without key commitment. In Kevin R. B. Butler and Kurt Thomas, editors, *31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022*, pages 3291–3308. USENIX Association, 2022.

[AJS09] Jean-Philippe Aumasson, Jorge Nakahara Jr., and Pouyan Sepehrdad. Cryptanalysis of the ISDB scrambling algorithm (MULTI2). In Orr Dunkelman, editor, *Fast Software Encryption, 16th International Workshop, FSE 2009, Leuven, Belgium, February 22-25, 2009, Revised Selected Papers*, volume 5665 of *Lecture Notes in Computer Science*, pages 296–307. Springer, 2009.

[BN14] Alex Biryukov and Ivica Nikolic. Colliding keys for SC2000-256. In Antoine Joux and Amr M. Youssef, editors, *Selected Areas in Cryptography - SAC 2014 - 21st International Conference, Montreal, QC, Canada, August 14-15, 2014, Revised Selected Papers*, volume 8781 of *Lecture Notes in Computer Science*, pages 77–91. Springer, 2014.

[Dam89] Ivan Damgård. A design principle for hash functions. In Gilles Brassard, editor, *Advances in Cryptology - CRYPTO 1989, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 416–427. Springer, 1989.

[DDKS12] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Efficient dissection of composite problems, with applications to cryptanalysis, knapsacks, and combinatorial search problems. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*, pages 719–740. Springer, 2012.

[DFI+24] Patrick Derbez, Pierre-Alain Fouque, Takanori Isobe, Mostafizar Rahman, and André Schrottenloher. Key committing attacks against aes-based AEAD schemes. *IACR Trans. Symmetric Cryptol.*, 2024(1):135–157, 2024.

[DGLP22] Xiaoyang Dong, Jian Guo, Shun Li, and Phuong Pham. Triangulating rebound attack on aes-like hashing. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part I*, volume 13507 of *Lecture Notes in Computer Science*, pages 94–124. Springer, 2022.

[DGPW12] Alexandre Duc, Jian Guo, Thomas Peyrin, and Lei Wei. Unaligned rebound attack: Application to keccak. In Anne Canteaut, editor, *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, volume 7549 of *Lecture Notes in Computer Science*, pages 402–421. Springer, 2012.

[DLP23] Xiaoyang Dong, Shun Li, and Phuong Pham. Chosen-key distinguishing attacks on full aes-192, aes-256, kiasu-bc, and more. *IACR Cryptol. ePrint Arch.*, page 1095, 2023.

[DR06] Joan Daemen and Vincent Rijmen. Understanding two-round differentials in AES. In Roberto De Prisco and Moti Yung, editors, *Security and Cryptography for Networks, 5th International Conference, SCN 2006, Maiori, Italy, September 6-8, 2006, Proceedings*, volume 4116 of *Lecture Notes in Computer Science*, pages 78–94. Springer, 2006.

[GP10] Henri Gilbert and Thomas Peyrin. Super-sbox cryptanalysis: Improved attacks for aes-like permutations. In Seokhie Hong and Tetsu Iwata, editors, *Fast Software Encryption, 17th International Workshop, FSE 2010, Seoul, Korea, February 7-10, 2010, Revised Selected Papers*, volume 6147 of *Lecture Notes in Computer Science*, pages 365–383. Springer, 2010.

[HS20] Akinori Hosoyamada and Yu Sasaki. Finding hash collisions with quantum computers by using differential trails with smaller probability than birthday bound. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory*

and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II, volume 12106 of *Lecture Notes in Computer Science*, pages 249–279. Springer, 2020.

[IPS13] Mitsugu Iwamoto, Thomas Peyrin, and Yu Sasaki. Limited-birthday distinguishers for hash functions - collisions beyond the birthday bound can be meaningful. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 504–523. Springer, 2013.

[Jea16] Jérémy Jean. TikZ for Cryptographers. https://www.iacr.org/authors/tikz/, 2016.

[JNP12] Jérémy Jean, María Naya-Plasencia, and Thomas Peyrin. Improved rebound attack on the finalist grøstl. In Anne Canteaut, editor, *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, volume 7549 of *Lecture Notes in Computer Science*, pages 110–126. Springer, 2012.

[JNP13] Jérémy Jean, María Naya-Plasencia, and Thomas Peyrin. Multiple limited-birthday distinguishers and applications. In Tanja Lange, Kristin E. Lauter, and Petr Lisonek, editors, *Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*, volume 8282 of *Lecture Notes in Computer Science*, pages 533–550. Springer, 2013.

[JNP14a] Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. Kiasu v1. additional first-round candidates of caesar compeition, 2014.

[JNP14b] Jérémy Jean, Ivica Nikolic, and Thomas Peyrin. Tweaks and keys for block ciphers: The TWEAKEY framework. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*, volume 8874 of *Lecture Notes in Computer Science*, pages 274–288. Springer, 2014.

[KBN09] Dmitry Khovratovich, Alex Biryukov, and Ivica Nikolic. Speeding up collision search for byte-oriented hash functions. In Marc Fischlin, editor, *Topics in Cryptology - CT-RSA 2009, The Cryptographers' Track at the RSA Conference 2009, San Francisco, CA, USA, April 20-24, 2009. Proceedings*, volume 5473 of *Lecture Notes in Computer Science*, pages 164–181. Springer, 2009.

[KNR14] Dmitry Khovratovich, Ivica Nikolic, and Christian Rechberger. Rotational rebound attacks on reduced skein. *J. Cryptol.*, 27(3):452–479, 2014.

[KSW96] John Kelsey, Bruce Schneier, and David A. Wagner. Key-schedule cryptanalysis of idea, g-des, gost, safer, and triple-des. In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO 1996, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, volume 1109 of *Lecture Notes in Computer Science*, pages 237–251. Springer, 1996.

[LMR+09] Mario Lamberger, Florian Mendel, Christian Rechberger, Vincent Rijmen, and Martin Schläffer. Rebound distinguishers: Results on the full whirlpool

compression function. In Mitsuru Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*, volume 5912 of *Lecture Notes in Computer Science*, pages 126–143. Springer, 2009.

[Mat09] Mitsuru Matsui. Key collisions of the RC4 stream cipher. In Orr Dunkelman, editor, *Fast Software Encryption, 16th International Workshop, FSE 2009, Leuven, Belgium, February 22-25, 2009, Revised Selected Papers*, volume 5665 of *Lecture Notes in Computer Science*, pages 38–50. Springer, 2009.

[Mer89] Ralph C. Merkle. A certified digital signature. In Gilles Brassard, editor, *Advances in Cryptology - CRYPTO 1989, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 218–238. Springer, 1989.

[MRS14] Florian Mendel, Vincent Rijmen, and Martin Schläffer. Collision attack on 5 rounds of grøstl. In Carlos Cid and Christian Rechberger, editors, *Fast Software Encryption - 21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers*, volume 8540 of *Lecture Notes in Computer Science*, pages 509–521. Springer, 2014.

[MRST09] Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. The rebound attack: Cryptanalysis of reduced whirlpool and grøstl. In Orr Dunkelman, editor, *Fast Software Encryption, 16th International Workshop, FSE 2009, Leuven, Belgium, February 22-25, 2009, Revised Selected Papers*, volume 5665 of *Lecture Notes in Computer Science*, pages 260–276. Springer, 2009.

[MvOV96] Alfred Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[NSS23] Yusuke Naito, Yu Sasaki, and Takeshi Sugawara. Committing security of ascon: Cryptanalysis on primitive and proof on mode. *IACR Trans. Symmetric Cryptol.*, 2023(4):420–451, 2023.

[PGV93] Bart Preneel, René Govaerts, and Joos Vandewalle. Hash functions based on block ciphers: A synthetic approach. In Douglas R. Stinson, editor, *Advances in Cryptology - CRYPTO 1993, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, volume 773 of *Lecture Notes in Computer Science*, pages 368–378. Springer, 1993.

[Pub12] Fips Pub. Secure hash standard (shs). *Fips pub*, 180(4), 2012.

[Riv92] Ronald L. Rivest. The MD5 message-digest algorithm. *RFC*, 1321:1–21, 1992.

[Sin05] Carsten Sinz. Towards an optimal CNF encoding of boolean cardinality constraints. In Peter van Beek, editor, *Principles and Practice of Constraint Programming - CP 2005, 11th International Conference, CP 2005, Sitges, Spain, October 1-5, 2005, Proceedings*, volume 3709 of *Lecture Notes in Computer Science*, pages 827–831. Springer, 2005.

[SLR+15] Bing Sun, Zhiqiang Liu, Vincent Rijmen, Ruilin Li, Lei Cheng, Qingju Wang, Hoda Alkhzaimi, and Chao Li. Links among impossible differential, integral and zero correlation linear cryptanalysis. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual*

Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I, volume 9215 of *Lecture Notes in Computer Science*, pages 95–115. Springer, 2015.

[SLW+10] Yu Sasaki, Yang Li, Lei Wang, Kazuo Sakiyama, and Kazuo Ohta. Non-full-active super-sbox analysis: Applications to ECHO and grøstl. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, volume 6477 of *Lecture Notes in Computer Science*, pages 38–55. Springer, 2010.

[SW23] Ling Sun and Meiqin Wang. Sok: Modeling for large s-boxes oriented to differential probabilities and linear correlations. *IACR Trans. Symmetric Cryptol.*, 2023(1):111–151, 2023.

[SWW21] Ling Sun, Wei Wang, and Meiqin Wang. Accelerating the search of differential and linear characteristics with the SAT method. *IACR Trans. Symmetric Cryptol.*, 2021(1):269–315, 2021.

[TSI+24a] Kodai Taiyama, Kosei Sakamoto, Ryoma Ito, Kazuma Taka, and Takanori Isobe. Key collisions on AES and its applications. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology - ASIACRYPT 2024 - 30th International Conference on the Theory and Application of Cryptology and Information Security, Kolkata, India, December 9-13, 2024, Proceedings, Part VII*, volume 15490 of *Lecture Notes in Computer Science*, pages 267–300. Springer, 2024.

[TSI+24b] Kodai Taiyama, Kosei Sakamoto, Ryoma Ito, Kazuma Taka, and Takanori Isobe. Key collisions on AES and its applications. *IACR Cryptol. ePrint Arch.*, page 1508, 2024.

[TTI24a] Ryunosuke Takeuchi, Yosuke Todo, and Tetsu Iwata. Practical committing attacks against rocca-s. Cryptology ePrint Archive, Paper 2024/901, 2024.

[TTI24b] Ryunouchi Takeuchi, Yosuke Todo, and Tetsu Iwata. Key recovery, universal forgery, and committing attacks against revised rocca: How finalization affects security. *IACR Trans. Symmetric Cryptol.*, 2024(2):85–117, 2024.

A New Free-target-plaintext Key Collision Attack 5-round `AES-128`

Using the improved automatic search method, we set parameters $\texttt{R} = 5, r_i = 1, NS_7 = 21, NS_6 = 18, NS_{in_7} = 20, NS_{in_6} = 9$, `version` = `128` and `attack-type` = `free-TPKC` as the input to the `findRFcharacteristic` function, which allowed us to find a new 5-round rebound-friendly differential characteristic, as shown in Figure 14. The total probability of this differential characteristic is 2^{-255}. The inbound phase consists of the `SB` operations in the first and second rounds, with a probability of 2^{-194}, while the remaining portion serves as the outbound phase with a probability of 2^{-61}.

Given the differences Δx_1, Δy_1, Δx_2, and Δy_2 in the inbound phase, we can determine the values of the active bytes by assessing the `DDT`, while inactive bytes can take arbitrary values. Therefore, in Inbound 1, we can generate up to 2^{42} starting points (x_1, y_1), and in Inbound 2, we can generate up to 2^{20} starting points (x_2, y_2).

The process of finding a colliding key pair is stated as follows:

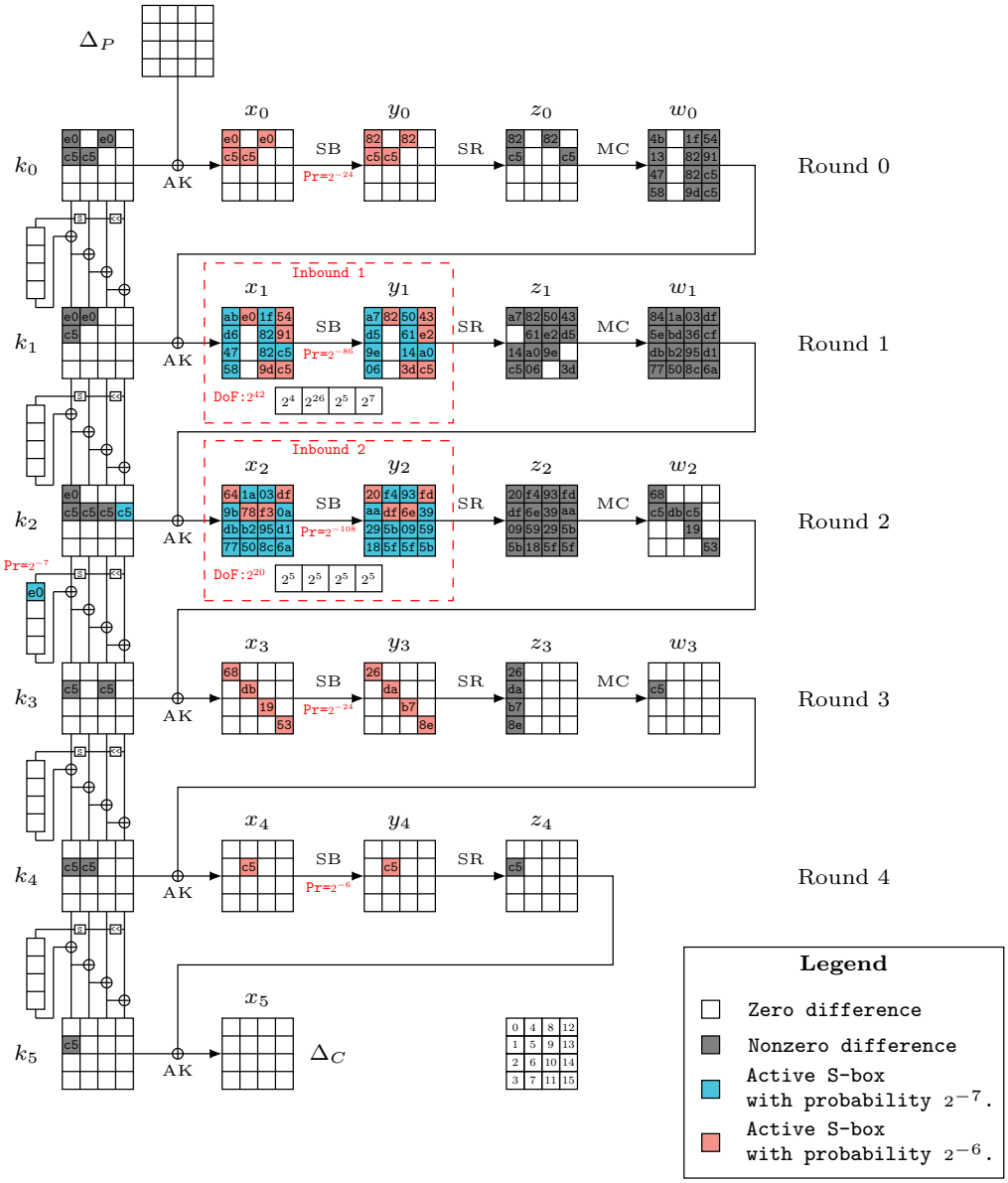


Figure 14: New free-target-plaintext key collision attack on 5-round AES-128.

1. In inbound phase, we can generate (x_1, y_1) that satisfy the differences Δx_1 and Δy_1 in Inbound 1, and generate (x_2, y_2) that match the differences Δx_2 and Δy_2 in Inbound 2. Compute $w_1 = \texttt{MC} \circ \texttt{SR}(y_1)$ and $k_2 = x_2 \oplus w_1$. Since the difference $\Delta k_2[13]$ is non-zero, the SW operation has a differential probability of 2^{-7}, consuming 2^7 DoF from $y_1[6]$ to satisfy this difference.

2. Calculate $w_2 = \texttt{MC} \circ \texttt{SR}(y_2)$, $k_3 = \texttt{KS}(k_2)$, and $x_3 = w_2 \oplus k_3$. Then compute $y_3 = \texttt{SB}(x_3)$ and $y_3' = \texttt{SB}(x_3 \oplus \Delta x_3)$. In the Round 3 SB operation, the differential probability is 2^{-24}, requiring 2^{24} DoF from the inbound phase. If $y_3 \oplus y_3' \neq \Delta y_3$, return to step 1.

3. Once a valid y_3 is obtained, calculate $w_3 = \texttt{MC} \circ \texttt{SR}(y_3)$, $k_4 = \texttt{KS}(k_3)$, $y_4 = \texttt{SB}(k_4 \oplus w_3)$, and $y_4' = \texttt{SB}(k_4 \oplus w_3 \oplus \Delta x_3)$. In the Round 4 SB operation, the differential probability is 2^{-6}, requiring 2^6 DoF from the inbound phase. If $y_4 \oplus y_4' \neq \Delta y_4$, return to step 1.

4. Starting from x_1 in Inbound 1, calculate $k_1 = \texttt{KS}^{-1}(k_2)$, $w_0 = x_1 \oplus k_1$, and $y_0 = \texttt{SR}^{-1} \circ \texttt{MC}^{-1}(w_0)$. In Round 0 SB operation, the differential probability is 2^{-24}, requiring 2^{24} DoF from the inbound phase.

5. Once a valid x_0 is obtained, calculate $k_0 = \texttt{KS}^{-1}(k_1)$. The resulting free-TPKC key pair is $(k_0, k_0 \oplus \Delta k_0)$, with the corresponding plaintext given by $k_0 \oplus x_0$.

In the attack process described above, it can be observed that the complexity of steps 2, 3, and 4 is dominant, requiring approximately 2^{54} partial AES-128 encryption and decryption computations. Therefore, the total time complexity can be approximated as 2^{54} executions of a 5-round AES-128 encryption.

B New Free-target-plaintext Key Collision Attack on 7-round AES-192

The new 7-round AES-192 rebound-friendly differential characteristic, as shown in Figure 15. The total probability of this differential characteristic is 2^{-270}.

Given the differences Δx_2, Δy_2, Δx_3, Δy_3, $\Delta x_4^{(1,2)}, \Delta y_4^{(1,2)}$ in the inbound phase, we can determine the values of the active bytes by assessing the DDT, while inactive bytes can take arbitrary values. Therefore, in Inbound 1, we can generate up to 2^{40} starting points (x_2, y_2), and in Inbound 2, we can generate up to 2^{66} starting points $(x_3, y_3, x_4^{(1,2)}, y_4^{(1,2)})$.

The process of finding a colliding key pair is stated as follows:

1. In Inbound 1, generate (x_2, y_2) that satisfy the differences Δx_2 and Δy_2. In Inbound 2, generate (x_3, y_3) that match the differences Δx_3 and Δy_3, then, based on the differences $\Delta x_4^{(1,2)}, \Delta y_4^{(1,2)}$, generate $(x_4^{(1,2)}, y_4^{(1,2)})$. Compute $w_2 = \texttt{MC} \circ \texttt{SR}(y_2)$, $k_3 = x_3 \oplus w_2$, $w_3 = \texttt{MC} \circ \texttt{SR}(y_3)$ and $k_4^{(1,2)} = w_3^{(1,2)} \oplus x_4^{(1,2)}$.

2. Calculate $(k_4^{(3,4)}, k_5) = \texttt{KS}(k_3, k_4^{(1,2)})$, and $x_4^{(3,4)} = w_3^{(3,4)} \oplus k_4^{(3,4)}$. Then compute $y_4^{(3,4)} = \texttt{SB}(x_4^{(3,4)})$ and $y_4'^{(3,4)} = \texttt{SB}(x_4^{(3,4)} \oplus \Delta x_4^{(3,4)})$. In the Round 4 SB operation of the third and fourth columns, the differential probability is 2^{-12}, requiring 2^{12} DoF from the inbound phase. If $y_4^{(3,4)} \oplus y_4'^{(3,4)} \neq \Delta y_4^{(3,4)}$, return to step 1.

3. Once a valid $y_4^{(3,4)}$ is obtained, calculate $w_4 = \texttt{MC} \circ \texttt{SR}(y_4)$, $x_5 = k_5 \oplus w_4$. Due to the zero differences for the Round 5 SB operation, we proceed with one more round of the AES round function to compute w_5. Next, we calculate k_6 and $k_7^{(1,2)}$ use the key schedule as follows: $k_6, k_7^{(1,2)} = \texttt{KS}(k_4^{(3,4)}, k_5)$,$k_7^{(3,4)} = \texttt{KS}(k_6, k_7^{(1,2)})$. Then, compute $x_6 = w_5 \oplus k_6$. For Round 6, the SB operation's differential probability is 2^{-7}, requiring 2^7 DoF from the inbound phase to satisfy this part of differences. Return to step 1 if this requirement cannot be met.

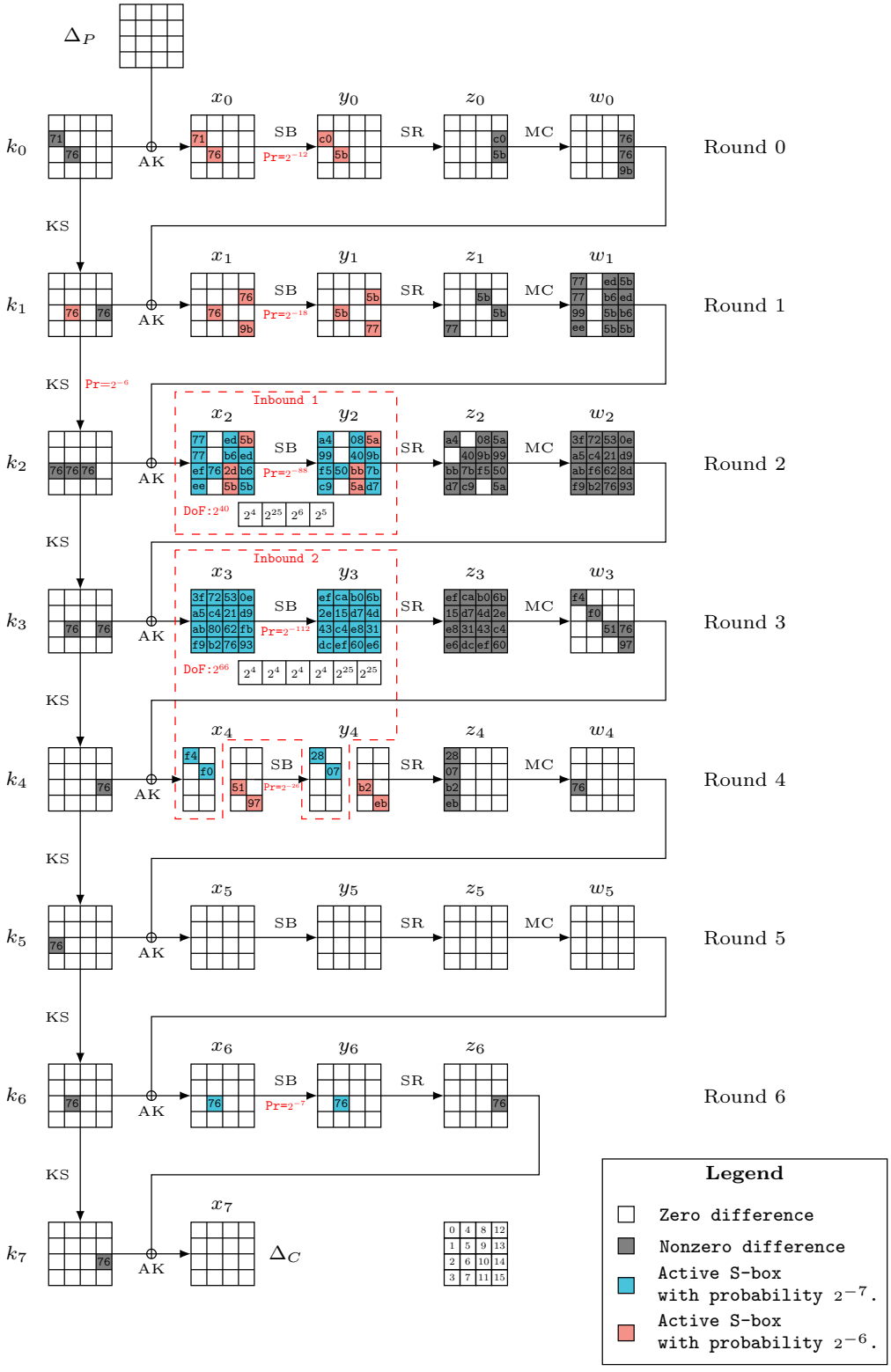


Figure 15: New free-target-plaintext key collision attack on 7-round AES-192.

4. Starting from x_2 in Inbound 1, calculate $(k_1^{(3,4)}, k_2) = \texttt{KS}^{-1}(k_3, k_4^{(1,2)})$, $(k_0, k_1^{(1,2)}) = \texttt{KS}^{-1}(k_1^{(3,4)}, k_2)$, $w_1 = x_2 \oplus k_2$, and $y_1 = \texttt{SR}^{-1} \circ \texttt{MC}^{-1}(w_1)$. In the Round 1 SB operation, the differential probability is 2^{-18}, requiring 2^{18} DoF from the inbound phase. From the differential characteristic in Figure 15, it can be seen that to obtain $k_1[6]$ and $x_0[1, 6]$ that satisfy the differential, we need to consume 2^{20} DoF from the inbound phase. If the state does not satisfy this differential, return to step 1.

5. Once a valid x_0 is obtained, calculate $k_0 = \texttt{KS}^{-1}(k_1)$. The resulting free-TPKC key pair is $(k_0, k_0 \oplus \Delta k_0)$, with the corresponding plaintext given by $k_0 \oplus x_0$.

In the attack process described above, it can be observed that the complexity of steps 2, 3, and 4 is dominant, requiring approximately 2^{57} partial AES-192 encryption and decryption computations. Therefore, the total time complexity can be approximated as 2^{56} executions of a 7-round AES-192 encryption.

C New Fixed-target-plaintext Key Collision Attack on 6-round AES-256

In this section, we discover that the differential characteristic used in the 6-round AES-256 fixed-TPKC attack by Taiyama et al. [TSI+24a] is incorrect. As shown in Table 6, during the SW operation in the first round of the key schedule, the S-box input-output difference pair 0x02 and 0x48 (highlighted in red in the table) does not correspond to any valid input-output pair in the AES S-box.

Although their incorrect differential path might have been a printing error, we still use our improved search method to find a new 6-round AES-256 fixed-TPKC differential characteristic, as shown in Figure 16. The total probability of this differential characteristic is 2^{-228}. The inbound phase includes the SB operations in the first and second rounds, with a probability of 2^{-133}, while the remaining part serves as the outbound phase, with a probability of 2^{-95}. Using this new differential characteristic, we launched a new 6-round AES-256 fixed-TPKC attack with a time complexity approximately equivalent to 2^{60} 6-round AES encryptions.

Table 6: The first-round differential characteristic used in the fixed-target-plaintext key collision attack on 6-round AES-256 found by Taiyama et al. [TSI+24b].

Round	Operation	State differences	Key differences
Plaintext		0x00000000 00000000 00000000 00000000	
0	After AK	0x24161248 00000000 00000000 00b90000	0x24161248 00000000 00000000 00b90000
	After SB	0xd0033d01 00000000 00000000 007c0000	
	After SR	0xd0000000 00000001 007c3d00 00030000	0xb952d069 00000000 00000000 02060402
	After MC	0xbbd0d06b 01010302 b9bf0641 05060303	After SW ∘ RW 0x24161248 00320000

Given the differences Δx_0, Δy_0, Δx_1, and Δy_1 in the inbound phase, we can determine the values of the active bytes by assessing the DDT, while inactive bytes can take arbitrary values. Therefore, in Inbound 1, we can generate up to 2^{79} starting points (x_0, y_0), and in Inbound 2, we can generate up to 2^{44} starting points (x_1, y_1).

The process of finding a colliding key pair is stated as follows:

1. In Inbound 1, generate (x_0, y_0) that satisfy the differences Δx_0 and Δy_0. In Inbound 2, generate (x_1, y_1) that match the differences Δx_1 and Δy_1. Compute $k_0 = P \oplus x_0$ to obtain the value of k_0. Compute $w_0 = \texttt{MC} \circ \texttt{SR}(y_0)$ and $k_1 = x_1 \oplus w_0$. Since the difference $\Delta k_1[12 - 15]$ is non-zero, the SW operation has a differential probability of 2^{-28}, consuming 2^{49} DoF from $y_0[1, 6, 11, 12], x_1[12 - 15]$ to satisfy this difference. Compute k_2 using the AES-256 key schedule, where $(k_2, k_3) = \texttt{KS}(k_0, k_1)$, to obtain the value of k_2. Since the difference $\Delta k_2[12]$ is non-zero, the SW operation has a

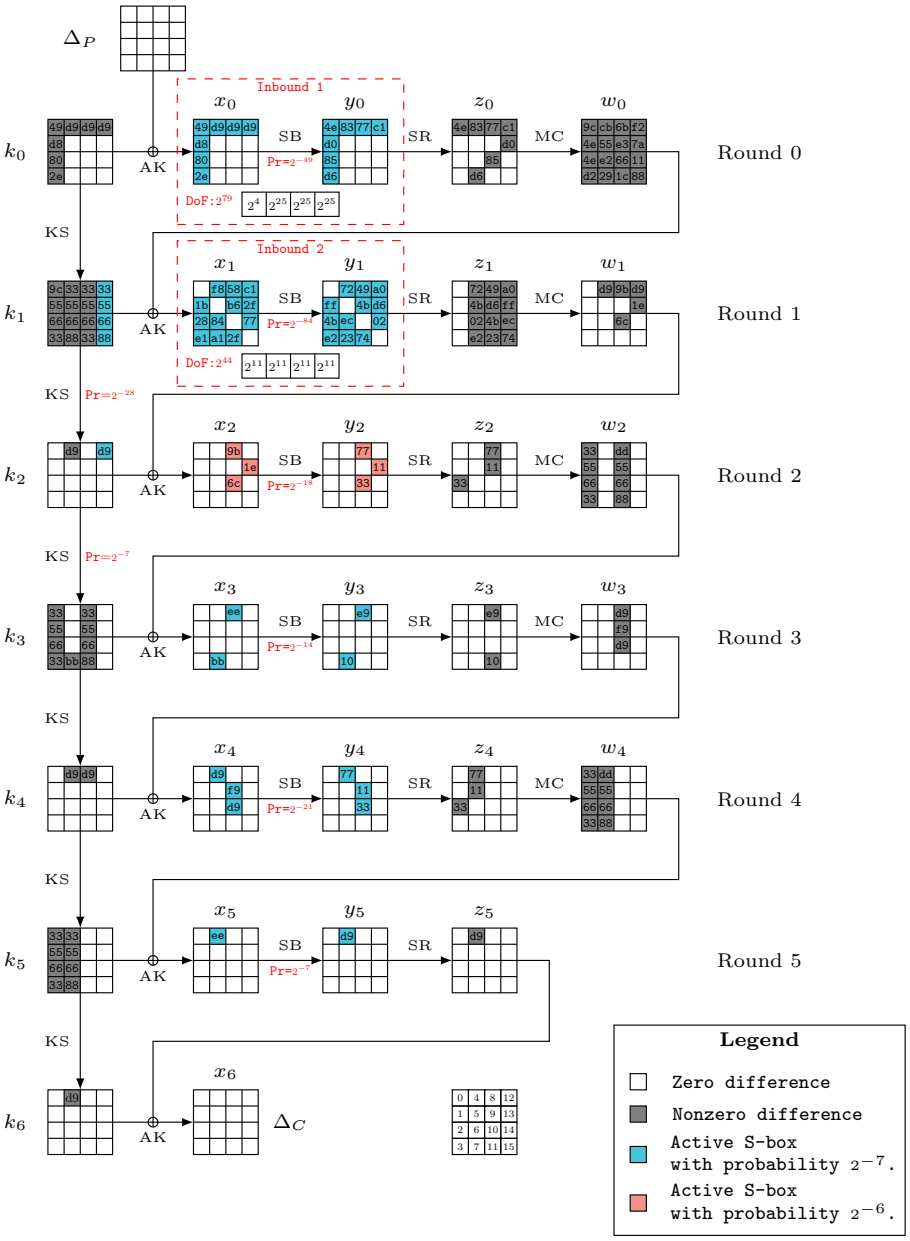


Figure 16: New fixed-target-plaintext key collision attack on 6-round AES-256.

differential probability of 2^{-7}, consuming 2^7 DoF from inbound phase to satisfy this difference.

2. Calculate $w_1 = \texttt{MC} \circ \texttt{SR}(y_1)$, $k_2 = \texttt{KS}(k_0, k_1)$, and $x_2 = w_1 \oplus k_2$. Then compute $y_2 = \texttt{SB}(x_2)$ and $y_2' = \texttt{SB}(x_2 \oplus \Delta x_2)$. In the Round 2 SB operation, the differential probability is 2^{-18}, requiring 2^{18} DoF from the inbound phase. If $y_2 \oplus y_2' \neq \Delta y_2$, return to step 1.

3. Once a valid y_2 is obtained, calculate $w_2 = \texttt{MC} \circ \texttt{SR}(y_2)$, $y_3 = \texttt{SB}(k_3 \oplus w_2)$, and $y_3' = \texttt{SB}(k_3 \oplus w_2 \oplus \Delta x_3)$. In the Round 3 SB operation, the differential probability is 2^{-14}, requiring 2^{14} DoF from the inbound phase. If $y_3 \oplus y_3' \neq \Delta y_3$, return to step 1. Perform the same calculation for Round 4 and Round 5. The S-box differential probabilities for these two rounds are 2^{-21} and 2^{-7}, respectively. Therefore, 2^{28} DoF from the inbound phase are required to satisfy this part of the difference. Return to step 1 if this requirement cannot be met.

4. Once a valid y_5 is obtained. The resulting fixed-TPKC key pair is $(k_0, k_0 \oplus \Delta k_0)$.

In the attack process described above, it can be observed that the complexity of steps 2 and 3 is dominant, requiring approximately 2^{60} partial AES-256 encryption computations. Therefore, the total time complexity can be approximated as 2^{60} executions of a 6-round AES-256 encryption.

D All Differential Characteristics Used in Key Collision Attacks.

Table 7: The differential characteristic used in the fixed-target-plaintext key collision attack on 2-round AES-128.

Round	Operation	State differences	Key differences
	Plaintext	0x00000000 00000000 00000000 00000000	0x5c3b3b4d 00003b4d 0b3b0093 00000000
0	After AK	0x5c3b3b4d 00003b4d 0b3b0093 00000000	0x5c3b3b4d 00003b4d 0b3b0093 00000000
	After SB	0x3bc49917 0000c4c1 713900c4 00000000	
	After SR	0x3b000000 00390017 710099c1 00c4c4c4	After SW ∘ RW 0x00000000
	After MC	0x763b3b4d 5c650017 ba000093 57000093	
1	After AK	0x2a000000 005e0017 ed000000 00000000	0x5c3b3b4d 5c3b0000 57000093 57000093
	After SB	0x5c000000 003b0093 57000000 00000000	
	After SR	0x5c3b0000 00000000 57000093 00000000	After SW ∘ RW 0x00003b4d
	Ciphertext	0x00000000 00000000 00000000 00000000	0x5c3b0000 00000000 57000093 00000000

Table 8: The differential characteristic used in the free-target-plaintext key collision attack on 2-round AES-128.

Round	Operation	State differences	Key differences
	Plaintext	0x00000000 00000000 00000000 00000000	0x4900c000 c500008c 8cc55c00 0000008c
0	After AK	0x4900c000 c500008c 8cc55c00 0000008c	0x4900c000 c500008c 8cc55c00 0000008c
	After SB	0xe700d900 68000091 24348c00 00000068	
	After SR	0xe7008c68 68340000 2400d991 00000000	After SW ∘ RW 0x00009c00
	After MC	0x31005c6e 8c005c8c 00c5258c 00000000	
1	After AK	0x7800006e 00000000 00002500 00c50000	0x49005c00 8c005c8c 00c5008c 00c50000
	After SB	0x8c00008c 00000000 00005c00 00c50000	
	After SR	0x8c005c00 0000008c 00c50000 00000000	After SW ∘ RW 0xc5000000
	Ciphertext	0x00000000 00000000 00000000 00000000	0x8c005c00 0000008c 00c50000 00000000

Table 9: The differential characteristic used in the free-target-plaintext key collision attack on 5-round AES-128.

Round	Operation	State differences	Key differences
	Plaintext	0x00000000 00000000 00000000 00000000	0xe0c50000 00c50000 e0000000 00000000
0	After AK	0xe0c50000 00c50000 e0000000 00000000	0xe0c50000 00c50000 e0000000 00000000
	After SB	0x82c50000 00c50000 82000000 00000000	
	After SR	0x82c50000 00000000 82000000 00c50000	After SW ∘ RW 0x00000000
	After MC	0x4b134758 00000000 1f82829d 5491c5c5	
1	After AK	0xabd64758 e0000000 1f82829d 5491c5c5	0xe0c50000 e0000000 00000000 00000000
	After SB	0xa7d59e06 82000000 5061143d 43e2a0c5	
	After SR	0xa70014c5 8261a006 50e29e00 43d5003d	After SW ∘ RW 0x00000000
	After MC	0x845edb77 1abdb250 0336958c dfcfd16a	
2	After AK	0x649bdb77 1a78b250 03f3958c df0ad16a	0xe0c50000 00c50000 00c50000 00c50000
	After SB	0x20aa2918 f4df5b5f 936e095f fd39595b	
	After SR	0x20df095b f46e5918 9339295f fdaa5b5f	After SW ∘ RW 0xe0000000
	After MC	0x68c50000 00db0000 00c51900 00000053	
3	After AK	0x68000000 00db0000 00001900 00000053	0x00c50000 00000000 00c50000 00000000
	After SB	0x26000000 00da0000 0000b700 0000008e	
	After SR	0x26dab78e 00000000 00000000 00000000	After SW ∘ RW 0x00000000
	After MC	0x00c50000 00000000 00000000 00000000	
4	After AK	0x00000000 00c50000 00000000 00000000	0x00c50000 00c50000 00000000 00000000
	After SB	0x00000000 00c50000 00000000 00000000	
	After SR	0x00c50000 00000000 00000000 00000000	After SW ∘ RW 0x00000000
	Ciphertext	0x00000000 00000000 00000000 00000000	0x00c50000 00000000 00000000 00000000

Table 10: The differential characteristic used in the fixed-target-plaintext key collision attack on 5-round AES-192.

Round	Operation	State differences	Key differences
	Plaintext	0x00000000 00000000 00000000 00000000	
0	After AK	0x31315662 3131562a 31315362 3131532a	0x31315662 3131562a 31315362 3131532a
	After SB	0x87c0bd65 da9c5794 3ef407fa 7baecdad	
	After SR	0x879c07ad daf4cd65 3eaebd94 7bc057fa	
	After MC	0x0000f948 00000086 bc315662 00e300f5	
1	After AK	0x0000f900 00000086 8d000000 00e300bd	0x00000048 00000000 31315662 00000048
	After SB	0x0000ae00 000000dd 3e000000 00050031	
	After SR	0x00000031 00000000 3e05aedd 00000000	0x00000000 after SW ∘ RW 0x00000000
	After MC	0x31315362 00000000 00000048 00000000	
2	After AK	0x00000048 00000000 00000000 00000048	0x3131532a 00000000 00000048 00000048
	After SB	0x00000031 00000000 00000000 00000031	
	After SR	0x00000031 00000031 00000000 00000000	0x00000048 after SW ∘ RW 0x00000500
	After MC	0x31315362 31315362 00000000 00000000	
3	After AK	0x00000000 00000048 00000000 00000000	0x31315362 3131532a 00000000 00000000
	After SB	0x00000000 00000031 00000000 00000000	
	After SR	0x00000000 00000000 00000031 00000000	
	After MC	0x00000000 00000000 31315362 00000000	
4	After AK	0x00000048 00000000 00000000 00000048	0x00000048 00000000 31315362 00000048
	After SB	0x00000048 00000000 00000000 00000048	
	After SR	0x00000048 00000048 00000000 00000000	0x00000000 after SW ∘ RW 0x00000000
	Ciphertext	0x00000000 00000000 00000000 00000000	0x00000048 00000048 00000000 00000000

Table 11: The differential characteristic used in the free-target-plaintext key collision attack on 7-round AES-192.

Round	Operation	State differences	Key differences
	Plaintext	0x00000000 00000000 00000000 00000000	
0	After AK	0x00710000 00007600 00000000 00000000	0x00710000 00007600 00000000 00000000
	After SB	0x00c00000 00005b00 00000000 00000000	
	After SR	0x00000000 00000000 00000000 00c05b00	
	After MC	0x00000000 00000000 00000000 0076769b	
1	After AK	0x00000000 00007600 00000000 0076009b	0x00000000 00007600 00000000 00007600
	After SB	0x00000000 00005b00 00000000 005b0077	
	After SR	0x00000077 00000000 005b0000 00005b00	0x00007600 after SW ∘ RW 0x00710000
	After MC	0x777799ee 00000000 edb65b5b 5bedb65b	
2	After AK	0x7777efee 00007600 edb62d5b 5bedb65b	0x00007600 00007600 00007600 00000000
	After SB	0x9e96bcc9 00003d00 3d0a93ff 5ade1ef1	
	After SR	0x9e0093f1 000a1ec9 3ddebc00 5a963dff	0x00000000 after SW ∘ RW 0x00000000
	After MC	0x45c1abd3 c9ff769d bf458025 d7d5aca0	
3	After AK	0x45c1abd3 c9ff009d bf458025 d7d5daa0	0x00000000 00007600 00000000 00007600
	After SB	0x11a24352 ec9400ef b00d48c8 974d86fa	
	After SR	0x119448fa ec0d8652 b04d43ef 97a200c8	
	After MC	0x37000000 00350000 00005100 0000768b	
4	After AK	0x37000000 00350000 00005100 0000008b	0x00000000 00000000 00000000 00007600
	After SB	0x28000000 00070000 0000b200 000000eb	
	After SR	0x2807b2eb 00000000 00000000 00000000	0x00000000 after SW ∘ RW 0x00000000
	After MC	0x00007600 00000000 00000000 00000000	
5	After AK	0x00000000 00000000 00000000 00000000	0x00007600 00000000 00000000 00000000
	After SB	0x00000000 00000000 00000000 00000000	
	After SR	0x00000000 00000000 00000000 00000000	0x00000000 after SW ∘ RW 0x00000000
	After MC	0x00000000 00000000 00000000 00000000	
6	After AK	0x00000000 00007600 00000000 00000000	0x00000000 00007600 00000000 00000000
	After SB	0x00000000 00007600 00000000 00000000	
	After SR	0x00000000 00000000 00000000 00007600	
	Ciphertext	0x00000000 00000000 00000000 00000000	0x00000000 00000000 00000000 00007600 0x00000000 after SW ∘ RW 0x00000000

Table 12: The differential characteristic used in the fixed-target-plaintext key collision attack on 6-round AES-256.

Round	Operation	State differences	Key differences
	Plaintext	0x00000000 00000000 00000000 00000000	
0	After AK	0x49d8802e d9000000 d9000000 d9000000	0x49d8802e d9000000 d9000000 d9000000
	After SB	0x4ed085d6 83000000 77000000 c1000000	
	After SR	0x4e000000 830000d6 77008500 c1d00000	
	After MC	0x9c4e4ed2 cb55e229 6be3661c f27a1188	
1	After AK	0x001b28e1 f80084a1 58b6002f c12f7700	0x9c556633 33556688 33556633 33556688
	After SB	0x00ff4be2 7200ec23 494b0074 a0d60200	
	After SR	0x00000000 724b02e2 49d64b23 a0ffec74	0x33556688 after SW ∘ RW 0x49d8802e
	After MC	0x00000000 d9000000 9b006c00 d91e0000	0xd9000000 after SW 0xaf000000
2	After AK	0x00000000 00000000 9b006c00 001e0000	0x00000000 d9000000 00000000 d9000000
	After SB	0x00000000 00000000 77003300 00110000	
	After SR	0x00003300 00000000 77110000 00000000	
	After MC	0x33556633 00000000 dd556688 00000000	
3	After AK	0x00000000 000000bb ee000000 00000000	0x33556633 000000bb 33556688 00000000
	After SB	0x00000000 00000010 e9000000 00000000	
	After SR	0x00000000 00000000 e9000010 00000000	0x00000000 after SW ∘ RW 0x00000000
	After MC	0x00000000 00000000 d9f9d900 00000000	0x00000000 after SW 0x00000000
4	After AK	0x00000000 d9000000 00f9d900 00000000	0x00000000 d9000000 d9000000 00000000
	After SB	0x00000000 77000000 00113300 00000000	
	After SR	0x00003300 77110000 00000000 00000000	
	After MC	0x33556633 dd556688 00000000 00000000	
5	After AK	0x00000000 ee000000 00000000 00000000	0x33556633 33556688 00000000 00000000
	After SB	0x00000000 d9000000 00000000 00000000	
	After SR	0x00000000 d9000000 00000000 00000000	0x00000000 after SW ∘ RW 0x00000000
	Ciphertext	0x00000000 00000000 00000000 00000000	0x00000000 d9000000 00000000 00000000