Key Collisions on AES and Its Applications

Kodai Taiyama[1], Kosei Sakamoto[2], Ryoma Ito[3], Kazuma Taka[1], and Takanori Isobe[1,3]

[1] University of Hyogo, Kobe, Japan
ad23x032@guh.u-hyogo.ac.jp, takanori.isobe@ai.u-hyogo.ac.jp
[2] Mitsubishi Electric Corporation, Kamakura, Japan
k.sakamoto0728@gmail.com
[3] NICT, Koganei, Japan.
itorym@nict.go.jp

Abstract. In this paper, we explore a new type of key collisions called target-plaintext key collisions of AES, which emerge as an open problem in the key committing security and are directly converted into single-block collision attacks on Davies-Meyer (DM) hashing mode. For this key collision, a ciphertext collision is uniquely observed when a specific plaintext is encrypted under two distinct keys. We introduce an efficient automatic search tool designed to find target-plaintext key collisions. This tool exploits bit-wise behaviors of differential characteristics and dependencies among operations and internal variables of both data processing and key scheduling parts. This allows us to hierarchically perform rebound-type attacks to identify key collisions. As a result, we demonstrate single-block collision attacks on 2/5/6-round AES-128/192/256-DM and semi-free-start collision attacks on 5/7/9-round AES-128/192/256-DM, respectively. To validate our attacks, we provide an example of fixed-target-plaintext key collision/semi-free-start collisions on 9-round AES-256-DM. Furthermore, by exploiting a specific class of free-start collisions with our tool, we present two-block collision attacks on 3/9-round AES-128/256-DM, respectively.

Keywords: AES, Davies-Meyer hashing mode, collision, rebound attacks

1 Introduction

1.1 Background

In block ciphers, a key collision is defined as two distinct keys that produce identical subkeys through the key scheduling function. When such colliding keys are used, any plaintext can be encrypted into the same ciphertext. Such key collisions are known for several ciphers. For instance, Robshaw [38] revealed that the block cipher CIPHERUNICORN-A, one of the CRYPTREC candidates, has colliding keys. Kelsey et al. [31] found trivial colliding keys for the Tiny Encryption Algorithm (TEA) block cipher. Furthermore, Aumasson et al. [2] showed

that the ISDB Scrambling Algorithm, MULTI2, also allows such keys. Biryukov and Nikolic [10] found colliding keys of SC2000-256 by exploiting the weakness of the key scheduling function. For stream ciphers, Matsui [35] investigated the behavior of colliding key pairs for the stream cipher RC4, in which two distinct keys generate the same key stream.

Recently, the importance of a new variant of key collisions has been demonstrated in the domain of the key committing security. Albertini et al. [1] revealed that standard AE (Authenticated Encryption) schemes such as AES-GCM and ChaCha20-Poly1305 lack this type of security and introduce a simple countermeasure, referred to as the padding fix. This method involves prepending an ℓ-bit string of 0's, denoted as X, to the message M for each encryption, resulting in $Enc(K, N, A, X||M)$, and check for the presence of X at the start of the message after decryption; decryption fails if X is not present. This countermeasure leads to the following open problem [1].

> "In particular, the padding fix with AES-GCM assumes an ideal cipher, and therefore raises the following interesting problem: Is it possible to find two keys k_1 and k_2 such that $AES_{k_1}(0) = AES_{k_2}(0)$ in less than 2^{64} trials. If the key size is larger than the block size, then such a pair of keys must exist. While there has been some work on the chosen-key setting [21] or using AES in a hashing mode [40], we are not aware of any results on this specific problem."

Compared to existing key collisions [2,10,31,35,38], particularly the subkey collision in the key scheduling function, a ciphertext collision is uniquely observed when a specific plaintext is encrypted under two distinct keys. In this paper, we refer to this as a target-plaintext key collision. Such collisions can be further categorized into two variants based on whether the target plaintext is predetermined: fixed-target-plaintext key collision and free-target-plaintext key collision. These key collisions can be directly converted into single-block collisions or semi-free-start collisions on the Davies-Meyer (DM) hashing mode with AES. This is due to the fact that target-plaintext key collisions are equivalent to the scenarios of collision attacks on DM mode where only message difference (which is a key part in AES) causes a collision when a fixed input chaining value (which is a part of plaintext in AES) is given.

Despite its significance, to the best of our knowledge, this type of attack has not yet been investigated for AES over the past 20 years. In fact, there are no results on collision and semi-free-start collision attacks on DM mode with AES over the significant number of rounds. On the other hand, there are numerous results of free-start collisions on DM hashing with AES [8,30,36], as well as collisions on Matyas-Meyer-Oseas (MMO) and Miyaguchi-Preneel (MP) modes [17,19,19,20]. In these attacks, the adversary can also inject differences into the plaintext part in AES to cause a collision, unlike DM mode. Once differential characteristics for collisions in MMO and MP are obtained, the values that fulfill differential characteristics are efficiently found by the rebound attacks [37], which consists of an inbound phase and an outbound phase. In the inbound phase, the differential

characteristics are realized deterministically by exploding degrees of freedom of the internal state. In the outbound phase, it is fulfilled in a probabilistic manner. For more information, existing tools and their limitations for a key-collision search on AES are discussed in Appendix A.

1.2 Difficulties for Finding Target-Plaintext Key Collisions

Here, we discuss the technical obstacles for finding target-plaintext key collisions and (semi-free) collision attacks on DM mode with AES by existing approaches.

- First of all, as far as we know, no differential characteristics of AES where a difference is inserted into only key, not into plaintext, leading to a ciphertext collision, have been demonstrated in the literature. Compared to the search for best related-key differential characteristics for AES [9, 15, 21, 25], this search requires strict conditions such that key differences should be canceled out by themselves without the help of plaintext differences. Consequently, the weights of target differential characteristics become quite high, resulting in time-consuming tasks. Therefore, finding these characteristics within a practical time appears to be very challenging so far.

- Even if differential characteristics for key collisions are identified, efficiently mounting rebound attacks in DM remains non-trivial. This difficulty arises because the core techniques of rebound attacks, e.g. super S-box [26, 29, 33], non-full-active super S-box [19, 20, 41], and super inbound [17] techniques, face limitations in controlling plaintext values. These techniques rely on exploiting the degrees of freedom (DoF) in the plaintext of an underlying block cipher, which is a message domain that the adversary can manipulate in MMO and MP, while in DM, this is a chaining value domain. Additionally, the strict conditions of differential characteristics for key collisions make it exceedingly difficult to find differential characteristics that align well with these techniques during the inbound phase. Particularly in DM, differential characteristics in the key scheduling function should be carefully considered in rebound attacks at the same time.

1.3 Our Contribution

In this paper, we introduce an efficient automatic search tool designed to find target-plaintext key collisions of AES, and then apply it to AES-128/192/256 and its DM hashing mode.

New Tool for Finding Key Collision on AES. To address the limitations of existing approaches, we first utilize *bit-wise* differential characteristics, which lead to a ciphertext collision resulting solely from key differences, by SAT method [44] with state-of-the-art SAT solver and encoding methods. In contrast, existing collision attacks on AES utilize truncated differential characteristics [8, 17, 19, 19, 20, 30, 36]. Bit-wise differential characteristics enable us to

exploit the bit-level relationship between the DoF of the inbound phase and the differential probability of the outbound phase in rebound attacks.

Specifically, we convert bit-wise differential characteristics into a graphical expression to leverage the DoF information of each vertex, dependencies between each vertex, and grouping based on these relationships. We then use a *depth-first search* in graph theory to generate a DoF tree, illustrating the optimal strategy for selecting inbound vertices, which are vertices categorized in the inbound phase, and determining the sequence for identifying outbound vertices, which are vertices categorized in outbound phase, from a given inbound vertex.

This DoF tree allows us to mount new rebound-type attacks, which *hierarchically* perform inbound and outbound phases across several groups categorized by vertex dependencies. As a result, our tool can efficiently mount rebound attacks for very complex differential characteristics, including those of the key scheduling function under the strict conditions of key collisions.

Application Results. We present fixed-target-plaintext key collisions on 2/5/6-round AES-128/192/256 and free-target-plaintext key collisions on 5/7/9-round AES-128/192/256. These are directly converted into single-block collision attacks on 2/5/6-round AES-128/192/256-DM and semi-free-start collision attacks on 5/7/9-round AES-128/192/256-DM, respectively. Table 1 summarizes our application results. To show the validity of our attacks, we provide an example of the fixed-target-plaintext key collision on AES-256 or semi-free-start collisions on 9-round AES-256-DM. Furthermore, by exploiting a specific class of free-start collisions, we present two-block collision attacks on 3/9-round AES-128/256-DM, respectively. As far as we know, these are the first results of collisions and semi-free start collisions of AES-DM over a significant number of rounds.

2 Preliminaries

2.1 Description of AES

AES [13] is a block cipher that supports a block size of 128 bits. It has three variants called AES-128, AES-192, and AES-256, depending on the combination of a key size of K_{len} bits and the number of rounds N_r. More specifically, $(K_{len}, N_r) = (128, 10)$, $(192, 12)$, and $(256, 14)$ for AES-128, AES-192, and AES-256, respectively. The internal state can be viewed as a 4×4 array of bytes.

Round Function. The round function consists of the following four operations:

- SubBytes (SB) is a parallel execution of 8-bit S-boxes. Due to page limitation, we do not give the specific table of the S-box.
- ShiftRows (SR) is a row-wise shuffle operation. Specifically, the i-th row of the state is cyclically shifted by i-bytes to the left.
- MixColumns (MC) is a column-wise 4×4 matrix multiplication over the finite field $\mathbb{F}_2^8$ with the irreducible polynomial $x^8 + x^4 + x^3 + x + 1$.

Table 1: Summary of our application results.

Target	Attack	Round	Time	Memory	Ref.
AES-128-DM	Collision	2	2^{49}	Negligible	Appendix C.1
	Collision*	3	2^{60}	2^{52}	Appendix F
	Semi-free-start	5	2^{57}	Negligible	Appendix C.2
	Free-start	5	2^{56}	2^{32}	[36]
	Free-start	6	2^{32}	2^{16}	[30]
AES-192-DM	Collision	5	2^{61}	Negligible	Appendix C.3
	Semi-free-start	7	2^{62}	Negligible	Appendix C.4
AES-256-DM	Collision	6	2^{61}	Negligible	Sect. 5.1
	Collision*	9	2^{58}	2^{55}	Sect. 6.3
	Semi-free-start	9	2^{30}	Negligible	Sect. 5.2
	q pseudo-collision†	14 (full)	$q \cdot 2^{67}$	Negligible	[8]

* It is a two-block collision. † The q pseudo-collision attack is synonymous with the q free-start collision attack. Kim et al. [32] claimed that this attack [8] is insufficient for a free-start collision attack on AES-256-DM, as it is inferior to the birthday attack in terms of the generic notion of collision attacks.

– AddRoundKey (AK) is the application of the round key. The 128-bit round key is XORed to the internal state.

The round function of AES is denoted by $f = \mathsf{AK} \circ \mathsf{MC} \circ \mathsf{SR} \circ \mathsf{SB}$; the i-th round internal state before the SB, SR, MC, and AK operations are denoted by x_i, y_i, z_i, and w_i, respectively; and the i-th round key is denoted by k_i, as depicted in Fig. 1. After an initial state w_0 is initialized with the initial round key k_0 by the AK operation, the internal state is updated by iterating the round function N_r times. Note that the MC operation is omitted for the final round.

Key Schedules. The key schedule algorithm takes the master key K and performs the key expansion function to generate the round keys k_i for $0 \le i \le N_r$. The resulting round keys consist of a linear array of 4-byte words, denoted by v_j with the range of $0 \le j < 4 \cdot (N_r + 1)$; namely, $k_i[j \bmod 4] = v_{4i+j}$.

The key expansion function consists of two steps. The first step is to initialize v_j for $0 \le j < N_k$ with K, where $N_k = 4$, 6, and 8 for AES-128, AES-192, and AES-256, respectively. The second step is to compute the remaining round keys, i.e., v_i for $N_k \le j < 4 \cdot (N_r + 1)$, as the following procedure (see Fig. 2):

$$v_j = \begin{cases} v_{j-N_k} \oplus \mathsf{SW}(\mathsf{RW}(v_{j-1})) \oplus \mathsf{RC}(i/N_k) & \text{if } j \equiv 0 \bmod N_k, \\ v_{j-N_k} \oplus \mathsf{SW}(v_{j-1}) & \text{if } N_k = 8 \text{ and } j \equiv 4 \bmod N_k, \\ v_{j-N_k} \oplus v_{i-1} & \text{otherwise,} \end{cases}$$

where SW is the SubWord function that takes a 4-byte input word and applies the SB operation to each of four bytes to produce a 4-byte output word, RW is the RotWord function that takes a 4-byte input word $[u_0, u_1, u_2, u_3]$, performs

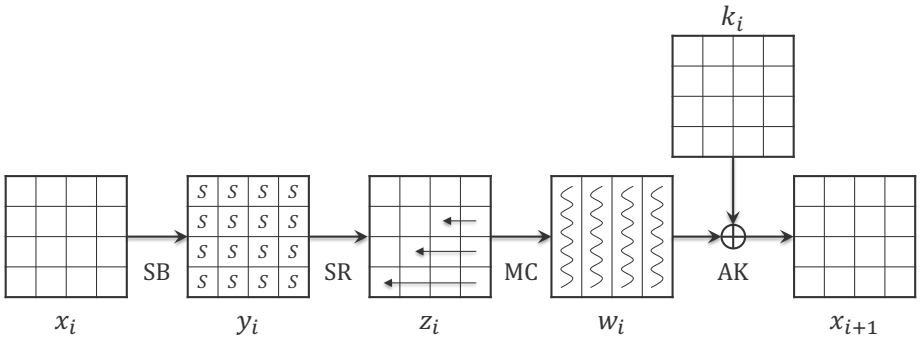


Fig. 1: The round function of AES.

the cyclic shift operation, and returns a 4-byte output word $[u_1, u_2, u_3, u_0]$, and $\mathsf{RC}(i/N_k)$ is the round constant.

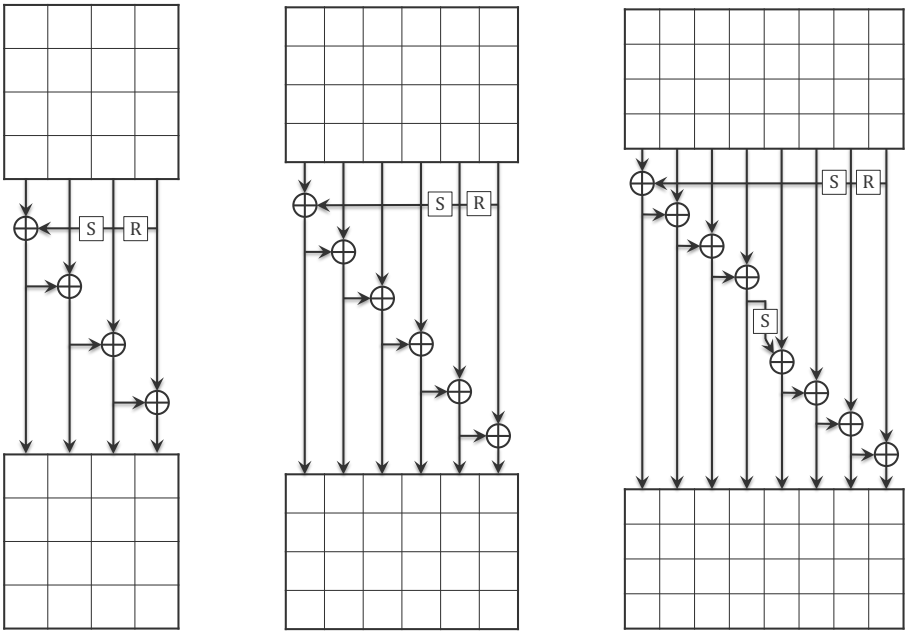
2.2 Rebound Attack

The rebound attack, proposed by Mendel et al. at FSE 2009 [37], is a generic tool for cryptanalysis of hash functions, especially on AES-like hashing. The basic idea of the attack is to obtain a specific differential characteristic in the underlying primitive (e.g., a block cipher or a cryptographic permutation) of the target hash function. More specifically, the rebound attack consists of an inbound phase and an outbound phase by decomposing the target primitive E into three parts so that $E = E_{fw} \circ E_{in} \circ E_{bw}$, as depicted in Fig. 3.

- **The inbound phase** aims to find a differential characteristic with a lower probability in E_{in} of the target primitive. To achieve this, an attacker must carefully control an input/output differential pair of the non-linear layers in E_{in} and determine an input/output differential pair in E_{in} that maximizes the differential probability in the following outbound phase. Searching for such a pair enables us to obtain many available solutions, which are the starting points for the outbound phase. It becomes the degree of freedom in the inbound phase.
- **The outbound phase** aims to obtain a valid differential characteristic in both forward and backward direction through E_{fw} and E_{bw} to find a desired collision. If the obtained differential characteristic has a sufficient probability of violating the security of collision attacks, the rebound attack can be considered successful. Otherwise, the attacker repeats the inbound phase to obtain more starting points for the outbound phase.

2.3 Collision Attacks and Its Variant

Given a hash function H, a collision is to identify message pair (m, m') satisfies $H(IV, m) = H(IV, m')$, where the initial vector IV is a fixed initial value. Let



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

Fig. 2: Key schedules of AES-128, AES-192, and AES-256. The SubWord and RotWord functions are denoted by S and R, respectively. Note that the round constant operation is omitted.

v be the chaining value that is equal to the output of the previous block. A semi-free-start collision is to find a pair (v, m) and (v, m'), such that $H(v, m) = H(v, m')$, where $(v \neq IV)$. A free-start collision is to find a pair (v, m_{i-1}) and (v', m'_{i-1}), so that $H(v', m) = H(v', m')$, where $(v \neq v')$. When the hash function H is built by iterating the compression function (CF) with the Merkle-Damgård construction, we can similarly define the semi-free-start and free-start collision attacks on the compression function.

3 Key Collision

In block ciphers, a key collision is defined as two distinct keys that produce identical subkeys through the key scheduling function. When such colliding keys are used, any plaintext can be encrypted into the same ciphertext. The existence of such keys is known for a few ciphers. For instance, Robshaw [38] has shown that the CRYPTREC candidate, the block cipher CIPHERUNICORN-A, has colliding keys. Kelsey et al. [31] have found colliding keys for the Tiny Encryption Algorithm (TEA) block cipher. Furthermore, Aumasson et al. [2] have discovered that the ISDB Scrambling Algorithm, the cipher MULTI2, allows such keys as well.

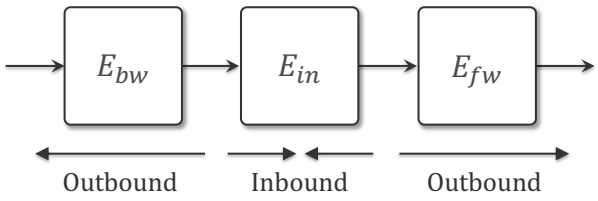


Fig. 3: A schematic view of the rebound attack.

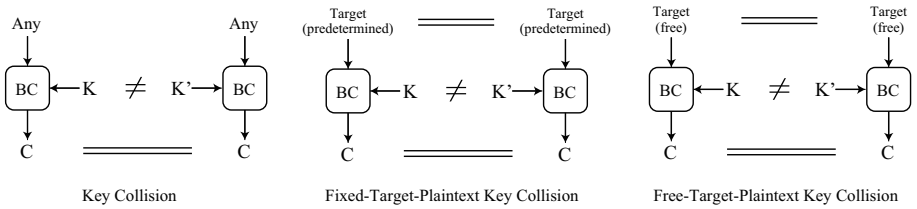


Fig. 4: Variants of key collisions.

Biryukov and Nikolic [10] have found colliding keys of SC2000-256 by exploiting the weakness of the key scheduling function. For stream ciphers, Matsui [35] has investigated the behavior of colliding key pairs for the stream cipher RC4, in which, two distinct keys generate the same key stream.

3.1 New Variants of Key Collision

In this paper, we introduce new variants of key collisions, termed *target-plaintext key collision*, defined as follows.

Definition 1 (Target-Plaintext Key Collision) *It is two distinct keys that generate the same ciphertext for a single target plaintext.*

Compared to existing key collisions, particularly the subkey collision in the key scheduling function, a ciphertext collision occurs exclusively with a specific plaintext under two distinct keys. Identifying such a collision can be classified into two different problems depending on whether a single target plaintext is predetermined or not, illustrated in Fig. 4.

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

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

In Problem 1, given a predetermined target plaintext, the adversary must identify two distinct keys that yield the same ciphertext. In contrast, Problem 2

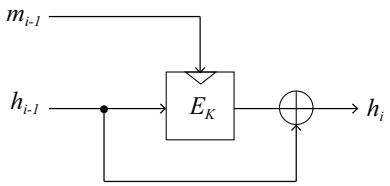


Fig. 5: A schematic view of the Davies-Meyer (DM) hashing mode.

allows the adversary to choose the target plaintext and find colliding key pairs freely. Clearly, Problem 1 is more challenging than Problem 2.

The time complexity for solving these problems by generic attack (assuming an underlying block cipher is an ideal cipher) depends on the size of the ciphertext. Specifically, for an n-bit ciphertext, such pairs can be found within a time complexity of $2^{n/2}$, owing to the birthday paradox.

3.2 Applications of Target-Plaintext Key Collisions

We explore the implications and significance of target-plaintext key collisions in theoretical and practical domains with several applications.

Collision Attack on DM Hashing Mode. The DM hashing mode is used to construct a cryptographic hash function from a block cipher. In this mode, as shown in Fig. 5, the key in the block cipher is treated as the message input for the hash function, and the plaintext is the initial vector or chaining value.

A fixed-target-plaintext key collision, where two keys produce the same ciphertext for a predetermined plaintext, corresponds to a single-block collision in DM mode. This implies that two different messages (keys) result in the same hash output (ciphertext) for a given initial state (plaintext). Similarly, a free-target-plaintext key collision, where the adversary can choose the plaintext and find two keys that produce the same ciphertext, correlates with a semi-free-start collision in DM mode. In this scenario, the attacker has the freedom to select the initial state (plaintext) and find two different messages (keys) that lead to the same hash output (ciphertext). This flexibility makes the semi-free-start collision less constrained and potentially attacks on more rounds than a single-block collision.

Open Problem of Padding Fix for Key Committing Security. Recently, the key committing security has garnered significant attention. In this line of research, Albertini et al. [1] revealed that standard AE schemes such as AES-GCM and ChaCha20-Poly1305 lack this type of security. In their paper, the authors introduce a simple countermeasure, referred to as the padding fix. This method involves prepending an ℓ-bit string of 0's, denoted as X, to the message M for each encryption, resulting in $Enc(K, N, A, X||M)$, and check for the presence

of X at the start of the message after decryption; decryption fails if X is not present. This countermeasure leads to the following open problem [1].

> "In particular, the padding fix with AES-GCM assumes an ideal cipher, and therefore raises the following interesting problem: Is it possible to find two keys k_1 and k_2 such that $AES_{k_1}(0) = AES_{k_2}(0)$ in less than 2^{64} trials. If the key size is larger than the block size, then such a pair of keys must exist. While there has been some work on the chosen-key setting [21] or using AES in a hashing mode [40], we are not aware of any results on this specific problem."

This issue is equivalent to the task of identifying target-plaintext key collisions in AES. To the best of our knowledge, this type of attack has not yet been investigated for AES over the past 20 years.

Embedded Device Keys by Malicious Factory. Regarding other applications of target-plaintext key collisions, consider the scenario of a malicious factory setting in which a device-specific key is embedded at an untrustworthy factory. This situation is akin to the context of a hardware trojan [5], with Original Equipment Manufacturing (OEM) serving as an illustrative example.

In such cases, there exists a significant risk when the fixed key is utilized for generating session keys via key derivation functions based on block ciphers for each device [12]. If the factory puts in specific key pairs meant for key collisions in two distinct devices, it may result in identical session keys for certain inputs. Consequently, an adversary possessing one device might be able to decrypt messages intended for another.

Moreover, the adversary could feasibly impersonate another device in systems where devices authenticate their identities by responding to a challenge computed using a block cipher with a device-fixed key. This is because their responses, which are generated using colliding key pairs, would be identical.

4 Automatic Tools for Key Collision on AES

In this section, we propose a new automatic tool designed to efficiently identify key collisions. This tool is comprised of six steps, and we describe them step by step. The overview of our attack procedure is shown in Alg. 1. Our rebound-type attack consists of the inbound and outbound phases as well as the standard rebound attack, but they have complex structures. Therefore, we call the internal states classified to the inbound phase (resp. outbound phase) as the inbound vertex (resp. outbound vertex).

Step 1: Searching for Differential Characteristics for Key Collisions. We employ the SAT-based automatic search method proposed by Sun et al. [44] to find good differential characteristics applied to our rebound-type attack. We

Algorithm 1: The Proposed Rebound-type Attack

```
   Data: b, rounds, LT
   Result: T_comp
1  cnf ← GenCNF(b, rounds);
2  path ← SAT(cnf) ;                                   // Step1
3  G ← GenGraph(path) ;                                // Step2
4  starts ← GetStarts(rounds) ;                        // Step3
5  forall S in starts do
6  |  DoF_tree ← GenTree(S, G) ;                       // Step4 and Step5
7  |  T_comp ← CalcTime(DoF_tree) ;                    // Step6
```

only give an overview of this method and our environment for evaluation. For more information about the modeling method, please refer to [44].

In the SAT-based approach, we express a differential propagation in a primitive and its weight caused by non-linear operations into CNF (Conjunctive Normal Form). For linear operations, such as an XOR and a permutation, their CNFs are shown in the previous works [42–44]. In contrast, for a non-linear operation, such as an S-box, we can derive their CNF expression by some algorithms used to simplify the Boolean function. Then, we employ Espresso logic minimizer[4] to derive CNF of an S-box. In addition to linear and non-linear operations, it is necessary to model the output such that the output difference is zero in order to discover a differential characteristic that leads to a collision. After converting differential propagation and its weight into CNF expression, we set the target weight k by Boolean cardinality constraints expressed in CNF and give the created CNF to a SAT solver. If an SAT solver returns "SAT", we can find differential characteristics with a weight of $\leq k$. Otherwise, we increase k and repeat this procedure until a SAT solver returns "SAT". In this work, we utilize ParKissat-RS[5] and totalizer as a SAT solver and Boolean cardinality constraints, respectively. GenCNF() in Alg. 1 accepts the number of rounds, $rounds$, that we attempt to find the key collision as the input and outputs a CNF, cnf, to find a differential characteristic. Then, SAT() accepts the CNF generated by GenCNF() and outputs the found differential characteristic, $path$, with its probability. In the standard situation where we attempt to identify optimal differential characteristics, we solve SAT models with a target weight k in ascending order. In contrast, we do not need to find optimal differential characteristics but good differential characteristics for our rebound-type attack.

In our attack, the probability of an underlying differential characteristic is constrained by the maximum DoF in each attack for successful attacks. A fixed-target-plaintext key collision can utilize the DoF of the key, while the free-target-plaintext key collision can utilize the DoF of both the key and plaintext. Thus, fixed-target-plaintext key collision requires a differential characteris-

[4] https://ptolemy.berkeley.edu/projects/embedded/pubs/downloads/espresso/index.htm

[5] https://github.com/shaowei-cai-group/ParKissat-RS

tic with probability of more than $2^{-128}/2^{-192}/2^{-256}$ on AES-128/192/256 while it is $2^{-256}/2^{-320}/2^{-384}$ on AES-128/192/256 for the free-target-plaintext collision, respectively.

Step 2: Converting Differential Characteristics into Graphical Expression. Since our rebound-type attack consists of the complex structure of the inbound and outbound vertices, we need to efficiently calculate the available DoF for fulfilling the differential characteristics found in Step 1. To this end, we first convert the differential characteristics into the graphical expression, including a set of vertices and edges, and utilize *depth-first search* to calculate the available DoF. GenGraph() in Alg. 1 accepts the differential characteristic found in Step 1 as the input and generates the corresponding graph.

Let $G = (V, E)$ be the graph of the differential characteristic where V and E denote a set of vertices and edges, respectively. V and E are determined as following rules:

Degree of the Graph. We first determine how the degree of bit-wise we convert differential characteristics into a set of vertices and edges. It significantly influences the efficiency of the later steps in our tool; the smaller the degree will be, the more complex constructing the attack will be. In this paper, we generate the graph in 32-bit wise for AES. Therefore, we treat a 32-bit word as a single unit on AES.

A set of vertices V. V is a set of vertices v, representing a unit of internal states that we can independently determine the values. Each vertex holds information about its weight and the available DoF.

A set of edges E. E is a set of edges e, representing vertices to which a particular vertex is connected. For example, if the vertex v_α is connected to the vertices v_β, and v_γ, we hold the edge $e_{v_\alpha}[v_\beta, v_\gamma]$ in E. We can interact with the available DoF in each connected vertex via the edge.

After creating the graph of the differential characteristic, we classify each SB and SW layer depending on in which v they belong. It can allow us to add information about the total probability in each vertex, helping us determine which vertices will be set to inbound or outbound vertices. It should be mentioned that the shape of the graph is determined by a specification of a primitive and the number of analysed rounds while available DoF in each vertex corresponds to a differential characteristic.

Example. For a better understanding, we take Fig. 6 as an example, which shows the round function and key scheduling function of the 2-round AES. First, as shown in Fig. 6, we convert the internal state into a set of vertices in 32-bit wise as follows:

$$V = \{v_{IV[i]}, v_{s^1[i]}, v_{MC^1[i]}, v_{s^2[i]}, v_{SR^2[i]}, v_{OUT[i]}, v_{K^0[i]}, v_{K^1[i]}, v_{K^2[i]}\},$$

where $i \in \{0, 1, 2, 3\}$, each of which can independently determine their values.

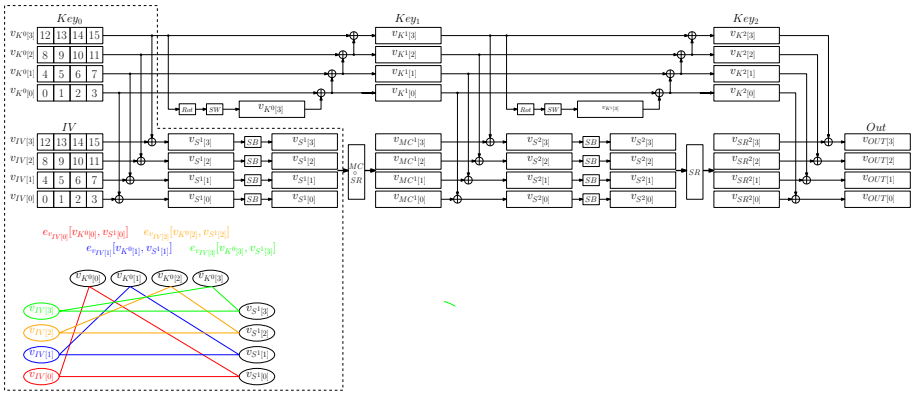


Fig. 6: Example of the graphical expression of round functions of AES.

From a set of vertices, we can obtain a set of edges corresponding to the vertices $v_{IV[i]}$ and $v_{K^0[i]}$ where $i \in \{0,1,2,3\}$ as follows:

$$E = \{e_{v_{IV[i]}}[v_{K^0[i]}, v_{S^1[i]}], e_{v_{K^0[0]}}[v_{IV[0]}, v_{S^1[0]}, v_{K^0[3]}, v_{K^1[0]}], e_{v_{K^0[1]}}[v_{IV[1]}, v_{S^1[1]}, v_{K^1[0]}, v_{K^1[1]}],$$
$$e_{v_{K^0[2]}}[v_{IV[2]}, v_{S^1[2]}, v_{K^1[1]}, v_{K^1[2]}], e_{v_{K^0[3]}}[v_{IV[3]}, v_{S^1[3]}, v_{K^0[0]}, v_{K^1[2]}, v_{K^1[3]}]\},$$

where $i \in \{0,1,2,3\}$, each of which connects vertices. We can obtain the remaining edges in the same manner. For simplicity, we depict edges in a part of the 2-round AES in Fig. 6.

Step 3: Determining the Starting Points. In the inbound phase of rebound attacks, we can determine the values following the differential characteristics by leveraging the DoF in the internal state. Specifically, by properly choosing values of internal state, we ensure that the probability of differential transition through an SB layer is one. This operation is executed for multiple vertices, depending on the available DoF. We call such a set of vertices as the starting points and vertices in the starting points as *inbound vertices*. The total number of inbound vertices is determined by the maximum available DoF in the target primitive and setting. For example, to find the fixed-target-plaintext key collision of the r-round AES-128, we can choose four vertices including SB or SW layers as the inbound vertices because the maximum available DoF is 2^{128}. In that case, since each round includes a total of 5 vertices, including SB and SW, the number of combinations for inbound vertices is $\binom{5r}{4}$, which will be too expensive to evaluate all combinations as r becomes large. Therefore, we restrict the number of rounds that include inbound vertices to the minimum in our evaluation.

We conduct the depth-first search from the inbound vertices to efficiently obtain the value pairs following the differential characteristics in the later steps. Hereafter, we call a vertex not belonging to the starting points as an *outbound vertex*. Once we determine the starting point, the *outbound vertices*, which consists of all the vertices not belonging to the starting point, is also determined

simultaneously. GetStarts() in Alg. 1 accepts the number of rounds that we attempt to find the key collisions as the input and outputs all combinations of starting points.

Step 4: Calculating the Degrees of Freedom in the Starting Points. After determining the starting point, we calculate DoF derived from the given individual inbound vertices. This calculation is equivalent to determining the number of valid values within the starting point that fulfills the corresponding differential characteristics.

Available Degrees of Freedom. Suppose that the total size of inbound vertices in the given starting point is b bits, and the probability of its differential propagation is 2^{-P_1}, we can obtain the DoF of 2^{b-P_1}. It should be emphasized that we can independently calculate the DoF derived from each inbound vertex. Therefore, the time complexity in a starting point will not be 2^{b-P_1} but can be much smaller than it. Inbound() in Alg. 2 accepts an starting point as the input and outputs this starting point with its DoF if the starting point is valid. Note that any choice of starting points does not always lead to the valid collision attacks, meaning that some choices of starting point cannot bring the colliding pairs due to their structures. In that case, Inbound() outputs `NULL`.

Example. Suppose that we choose the vertex $v_{S^1[0]}$ in Fig. 6 as one of inbound vertices and its probability is 2^{-14}, the available DoF derived from $v_{S^1[0]}$ is estimated $2^{18} (= 2^{32-14})$. Note that we must derive the DoF from all those states if there are multiple states over the non-linear operation in an inbound vertex.

Step 5: Finding Value Pairs Fulfilling the Outbound Vertices. After determining the starting point, we proceed to derive the value pairs fulfilling the entire differential characteristics for key collisions. To derive such pairs efficiently, we categorize inbound and outbound vertices using the *depth-first search*. During this depth-first search, we classify them based on their ability to calculate the values independently using the available DoF. This categorization allows us to derive the value pairs independently in each category, thereby minimizing the time complexity. For a better understanding, we give a simple example of how to categorize the inbound and outbound vertices.

Categorizing Vertices by Depth-first Search. Fig. 7 illustrates the overview of the depth-first search with a simple example. In this example, we have a set of vertices $V = \{v_{in_1}, v_{in_2}, v_{in_3}, v_{out_1}, v_{out_2}, v_{out_3}\}$ where $(v_{in_1}, v_{in_2}, v_{in_3})$ and $(v_{out_1}, v_{out_2}, v_{out_3})$ are inbound and outbound vertices, respectively. Besides, according to Fig. 7, we also have a set of edges

$$E = \{e_{v_{in_1}}[v_{in_2}, v_{out1}], e_{v_{in_2}}[v_{in_1}, v_{in_3}, v_{out_1}, v_{out_2}], e_{v_{in_3}}[v_{in_2}, v_{in_3}]$$
$$e_{v_{out_1}}[v_{in_1}, v_{in_2}, v_{out_2}, v_{out_3}], e_{v_{out_2}}[v_{in_2}, v_{in_3}, v_{out_1}, v_{out_3}], e_{v_{out_3}}[v_{out_1}, v_{out_2}]\}$$

Algorithm 2: GenTree

```
Data: S, G
Result: DoF_tree
if Inbound(S, G) = NULL then // Step4
    break;
else
    DoF_tree ← Grouping(S, G) // Step5
```

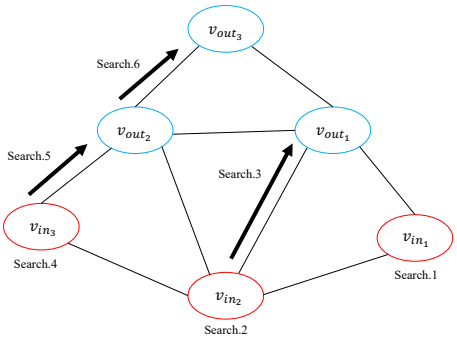


Fig. 7: The illustration of the depth-first search to categorize outbound vertices.

Then, we attempt to categorize outbound vertices based on the inbound vertices.

This search starts from all inbound vertices in the starting point. The procedure of our depth-first search is as follows:

Search 1. Start the depth-first search from v_{in_1}. According to $e_{v_{in_1}}[v_{in_2}, v_{out1}]$, we know that v_{in_1} is connected to v_{in_2} and v_{out_1}, but value pairs of both vertices have not been determined yet. Therefore, only the value pairs of in_1 can be determined in the first invocation of the depth-first search, and the corresponding vertices from v_{in_1} are only v_{in_1}.

Search 2. Start the depth-first search from v_{in_2}. According to $e_{v_{in_2}}[v_{in_1}, v_{in_3}, v_{out_1}, v_{out_2}]$, we know that v_{in_2} is connected to v_{in_1} and the value pairs of v_{in_1} have been already determined in Search 1. In other words, we have already conducted the inbound phase for v_{in_1} (or the outbound phase for the outbound vertices). However, the corresponding vertex of v_{in_2} is only v_{in_2} because we can determine the value pairs of v_{in_2} independently.

Search 3. After Search 2, we know that value pairs of v_{out_1} can be determined because the value pairs of v_{in_1} and v_{in_2} have been already determined. Besides, according to $e_{v_{out_1}}[v_{in_1}, v_{in_2}, v_{out_2}, v_{out_3}]$, v_{out_1} is also connected to v_{out_2} and v_{out_3}, but we cannot determine their value pairs yet. Therefore, the corresponding vertices of v_{out_1} are v_{in_1} and v_{in_2}.

Search 4. Start the depth-first search from v_{in_3}. For the same reason as Searches 1 and 2, the corresponding vertex of v_{in_3} is only v_{in_3}.

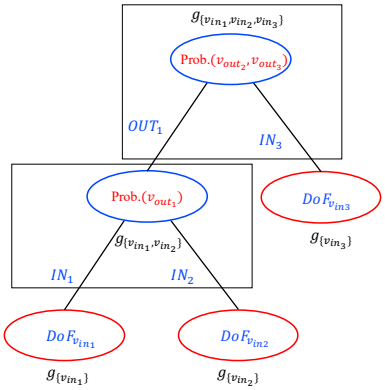


Fig. 8: Overview of the DoF tree. Prob.(v_{out_1}) and Prob.(v_{out_2}, v_{out_3}) denote the probability of the outbound groups $g_{\{v_{in_1},v_{in_2}\}}$ and $g_{\{v_{in_1},v_{in_2},v_{in_3}\}}$, respectively. $(DoF_{v_{in_1}}, DoF_{v_{in_2}}, DoF_{v_{in_3}})$ denote the available DoF of $(v_{in_1}, v_{in_2}, v_{in_3})$, respectively. $(IN_1, IN_2, IN_3, OUT_1)$ denote the DoF used to find the value pairs of the connected outbound groups.

Search 5. After Search 4, we know that v_{out_2} can be determined. The value pairs of v_{in_3} can be determined independently, but the value pairs of v_{in_2} have already been determined depending on the value pairs of v_{in_1} to obtain the value pairs of v_{out_1}. Therefore, the value pairs of v_{out_2} must correspond to the value pairs of v_{in_1}, and the corresponding vertices of v_{out_2} are v_{in_1}, v_{in_2}, and v_{in_3}.

Search 6. We determine the value pairs of the remaining vertex v_{out_3}. According to $e_{v_{out_3}}[v_{out_1}, v_{out_2}]$, v_{out_3} is connected to v_{out_1} and v_{out_2}, and their corresponding vertices contain v_{in_1}, v_{in_2}, and v_{in_3}. Therefore, the corresponding vertices of v_{out_3} are also v_{in_1}, v_{in_2}, and v_{in_3}.

Then, we obtain Table 2 that shows which inbound vertices each vertex corresponds to. After categorizing them, we generate a tree construction of vertices, called a *DoF tree*, which allows us to calculate the value pairs of each vertex with a minimum time complexity. Fig. 8 shows the DoF tree based on Table 2. In Fig. 8, each surrounded vertex by the black square can be calculated from their value pairs independently. For $g_{\{v_{in_1},v_{in_2}\}}$, we can use $DoF_{v_{in_1}}$ and $DoF_{v_{in_2}}$ to find the value pairs of $g_{\{v_{in_1},v_{in_2},v_{in_3}\}}$ while we can also calculate the value pairs of $g_{\{v_{in_1},v_{in_2},v_{in_3}\}}$ using $DoF_{v_{in_3}}$ at the same time. If $\text{Prob.}(v_{out_2}, v_{out_3})^{-1} > DoF_{v_{in3}}$, we use $DoF_{v_{in_1}}$ and $DoF_{v_{in_2}}$ via $g_{\{v_{in_1},v_{in_2}\}}$. Hence, in that case, we can partly calculate the value pairs of $g_{\{v_{in_1},v_{in_2},v_{in_3}\}}$ independently. The detailed attack complexity will be calculated in Step 6. GROUPING() in Alg. 2 accepts the starting point and the graph expression of the differential characteristic as the input and outputs the DoF tree.

Table 2: Categorizing outbound vertices. The outbound vertices are highlighted in red.

Step	Start points	Corresponding inbound vertices	Groups
1	v_{in_1}	$\{v_{in_1}\}$	$g_{\{v_{in_1}\}}$
2	v_{in_2}	$\{v_{in_2}\}$	$g_{\{v_{in_2}\}}$
3	v_{out_1}	$\{v_{in_1}, v_{in_2}\}$	$g_{\{v_{in_1}, v_{in_2}\}}$
4	v_{in_3}	$\{v_{in_3}\}$	$g_{\{v_{in_3}\}}$
5	v_{out_2}	$\{v_{in_1}, v_{in_2}, v_{in_3}\}$	$g_{\{v_{in_1}, v_{in_2}, v_{in_3}\}}$
6	v_{out_3}	$\{v_{in_1}, v_{in_2}, v_{in_3}\}$	$g_{\{v_{in_1}, v_{in_2}, v_{in_3}\}}$

Step 6: Estimating Attack Complexity. We check whether the total available DoF is enough to find the value pairs fulfilling the differential characteristic and its time complexity. We use MILP (*Mixed-Integer Linear Programming*) to check those and minimize the time complexity. During an MILP modeling, we assign DoF derived from each outbound group, highlighted red color in Table 2, as the linear constraints. For a better understanding, we give all constraints as the product of the probability and DoF, but we can express them by the linear inequalities by the weight and give them to an MILP solver. Let DoF_{OUT_i}, DoF_{C_j}, and Prob.(OUT_i) be DoF derived from outbound group OUT_i, the DoF used to calculate the value pairs in each connected group c_j, and the probability of outbound group OUT_i, respectively. The DoF derived from the outbound group OUT_i is calculated as follows:

$$DoF_{OUT_i} = \left(\prod_{j=1}^{n} DoF_{C_j} \right) \cdot \text{Prob.}(OUT_i), \tag{1}$$

where n denotes the number of connected groups to OUT_i. We assign Eq. (1) for all outbound groups as the linear constraints in an MILP model. For inbound groups, the black colored groups in Table 2, DoF used to calculate the value pairs of the connected outbound groups must be smaller than DoF derived from this inbound group. Thus, we assign such constraints as follows:

$$IN_k \leq DoF_{IN_k}, \tag{2}$$

where IN_k and DoF_{IN_k} denote the DoF used to calculate the value pairs of the connected outbound groups and DoF derived from this inbound group IN_k, respectively. We assign Eq. (2) for all inbound groups as the linear constraints in an MILP model.

Besides, for each outbound group, we need to ensure that the time complexity is smaller than the birthday bound as follows:

$$T^i_{max} \geq \prod_{j=1}^{m} IN_j \cdot \prod_{k=1}^{n} OUT_k \tag{3}$$

where T^i_{max}, m, and n denote the objective variables, the number of outbound groups, and the number of inbound groups connected to the target outbound group, respectively. We assign Eq. (3) for all outbound groups as the linear constraints in an MILP model.

Then, we need to set the objective function, which minimizes the time complexity. In the field of symmetric-key cryptography, we often assign one objective function, such as the number of active S-boxes and the total weight in primitives, and minimize it, equivalent to solving the minimization problem. In contrast, the time complexity of our attack is dominated by the maximum T^i_{max}, but there is no way to know which T^i_{max} will be maximum in the standard MILP model. Therefore, we solve the MIN-MAX problem, a class of problems where the objective is to minimize the maximum value of a set of variables and functions, instead of the minimization problem. Hence, we assign a set of all T^i_{max} as the objective function as follows:

$$(T^1_{max}, T^2_{max}, \ldots, T^m_{max}). \tag{4}$$

Then, we minimize the maximum variables in Eq. (4) by an MILP solver. In this work, we use `SageMath`[6] as an MILP solver. Our attack is successful if the maximum value in Eq. (4) will be smaller than the birthday bound. Otherwise, we conduct the same procedure for another DoF tree. CalcTime() in Alg. 1 accepts the DoF tree as the input and outputs the time complexity of this attack.

5 Key Collisions on AES-128/192/256

In this section, we show fixed-target-plaintext key collisions on 2/5/6-round AES-128/192/256 and free-target-plaintext key collisions on 5/7/9-round AES-128/192/256, which are found by our automatic tool provided in Sect. 4. Here, we only show the attacks on AES-256, and the attacks on AES-128/192 are shown in Appendix C. Besides, the optimality of the fixed-target-plaintext key collision attack on AES-128 with respect to the number of attacked rounds is discussed in Appendix B.

Notations. We use the following notations for our attacks. For $i \in \{1, \ldots, 12\}$, let in_i be the inbound vertices; let IN_i be the assigned degrees of freedom in the attack; let IN_{M_i} be the available degrees of freedom from a given differential characteristic; and let $P(in_i)$ be a differential probability of in_i in the corresponding SB operation.

Moreover, the internal states of AES are treated here as a column-wise array of 4-byte words, with columns numbered from the left. For example, $x_i[0]$ and $x_i[3]$ are represented as 4-byte words in the leftmost and rightmost columns in the i-th round internal state before the SB operation, respectively. In the same manner, the i-th round internal state before the AK operation is represented as

[6] https://www.sagemath.org

$w_i[3]$ and $w_i[0]$, respectively, and the i-th round key is denoted by k_i. Besides, all inbound vertices are written in the same color since it is clear that their values are obtained independently.

5.1 Fixed-Target-Plaintext Key Collision on 6-round AES-256

Fig. 26 in Appendix D illustrates an underlying differential characteristic for a fixed-target-plaintext key collision for 6-round AES-256 with a probability of 2^{-179}. Assuming that the 1st and 2nd rounds in the data processing part are an inbound phase (i.e., $\{in_1, \ldots, in_8\} = \{x_1[3], \ldots, x_1[0], x_2[3], \ldots, x_2[0]\}$), and the remaining part, including the key scheduling part, is an outbound phase, the probability of inbound and outbound phases are 2^{-118} and 2^{-61}, respectively. After applying our tool, we can obtain the DoF tree as shown in Fig. 27 in Appendix D for collision attacks. By using this tree, we can construct a fixed-target-plaintext key collision on 6-round AES-256.

Degree of Freedom in the Inbound Phase. In our attacks, we exploit the degrees of freedom of each inbound vertex as follows: $IN_1 = 2^{17}$, $IN_2 = 2^{17}$, $IN_3 = 2^{17}$, $IN_4 = 2^0$, $IN_5 = 2^{11}$, $IN_6 = 2^{11}$, $IN_7 = 2^{11}$, and $IN_8 = 2^{12}$.

By using our automatic tool, we have obtained the differential probability of each inbound vertex as follows: $P(in_1) = 2^{-7}$, $P(in_2) = 2^{-0}$, $P(in_3) = 2^{-0}$, $P(in_4) = 2^{-28}$, $P(in_5) = 2^{-21}$, $P(in_6) = 2^{-21}$, $P(in_7) = 2^{-21}$, and $P(in_8) = 2^{-20}$. Based on these results, the maximum degrees of freedom in each vertex are estimated as $IN_{M_1} = 2^{25(=32-7)}$, $IN_{M_2} = 2^{32(=32-0)}$, $IN_{M_3} = 2^{32(=32-0)}$, $IN_{M_4} = 2^{4(=32-28)}$, $IN_{M_5} = 2^{11(=32-21)}$, $IN_{M_6} = 2^{11(=32-21)}$, $IN_{M_7} = 2^{11(=32-21)}$, and $IN_{M_8} = 2^{12(=32-20)}$. Thus, the degrees of freedom of each inbound vertex, which is required for the attack, are sufficiently available.

Attack Procedures.

1. Start with $2^{51(=17+17+17+0)}$ sets of $\{in_1, \ldots, in_4\} = \{x_1[3], \ldots, x_1[0]\}$ and the fixed initial value sets of $\{w_0[3], \ldots, w_0[0]\}$ (red part in Fig. 9); then, obtain 2^{51} sets of $\{w_1[0], \ldots, w_1[3], k_0[0], \ldots, k_0[3]\}$ (blue, purple, green, and orange parts in Fig. 9).
2. Prepare 2^{11} values of $in_5 = x_2[3]$ (red part in Fig. 9); then, obtain $2^{62(=51+11)}$ sets of $\{k_1[3], k_2[0], \ldots, k_2[3], k_3[0]\}$ (turquoise part in Fig. 9). As the differential probability of the corresponding two SW functions in the key schedule part is $2^{-35(=-28-7)}$, there exist $2^{27(=62-35)}$ values that fulfill the differential characteristic (turquoise part in Fig. 9).
3. Prepare $2^{22(11+11)}$ sets of $\{in_6, in_7\} = \{x_2[2], x_2[1]\}$ (red part in Fig. 9); then, obtain $2^{49(=27+22)}$ sets of $\{k_1[2], k_1[1]\}$ (yellow and brown parts in Fig. 9).
4. Prepare 2^{12} values of $in_8 = x_2[0]$ (red part in Fig. 9); then, obtain $2^{61(=49+12)}$ values of the remaining data processing and key scheduling parts (gray part in Fig. 9). As the differential probability of the remaining parts is 2^{-61}, there exists $1 = 2^{0(=61-61)}$ value that fulfills the differential characteristic (gray part in Fig. 9).

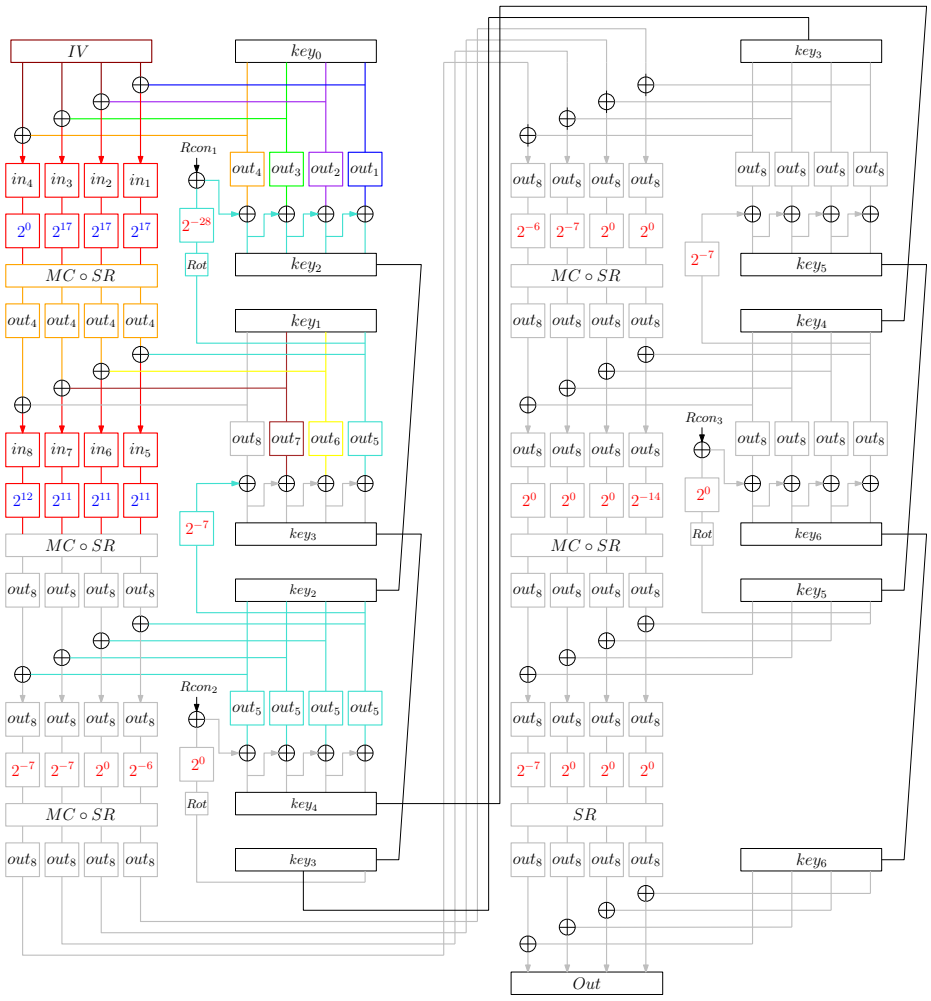


Fig. 9: Fixed-target-plaintext collision attack on 6-round AES-256.

Attack Complexity. Following the above attack procedures, the attack complexity for Step 2 appears to be dominant, which requires approximately 2^{62} computations of the 1-round AES-256 key schedule. However, the attack complexity for Step 4 requires approximately 2^{61} computations of the partial (at least 4-round) AES-256 encryption; thus, from the perspective of a fair complexity estimation, it can be considered that the attack complexity for Step 4 is dominant. Therefore, total complexity is bounded by 2^{61} computations of 6-round AES-256.

As in the actual execution of the attack, it is not necessary to store a complete set of values at each outbound vertices. For example, in Step 3, rather than storing $2^{49(=27+22)}$ sets of $\{k_1[2], k_1[1]\}$, we can proceed to evaluate the next outbound vertices for a single value of the previous outbound vertices. Thus,

our attacks in this section and Appendix C can be done with negligible memory. This is confirmed by our experiment for the semi-free start collision attack on 9-round AES-256.

5.2 Free-Target-Plaintext Key Collision on 9-round AES-256

Fig. 28 in Appendix E illustrates an underlying differential characteristic for a free-target-plaintext key collision for 9-round AES-256 with a probability of 2^{-193}. Assuming that the 7th, 8th, and 9th rounds in the data processing part are an inbound phase (i.e., $\{in_1, \ldots, in_{12}\} = \{x_7[3], \ldots, x_7[0], x_8[3], \ldots, x_8[0], x_9[3], \ldots, x_9[0]\}$), and the remaining part, including the key scheduling part, is an outbound phase, the probability of inbound and outbound phases are 2^{-102} and 2^{-91}, respectively. After applying our tool, we can obtain the DoF tree as shown in Fig. 29 in Appendix E for collision attacks. By using this tree, we can construct a free-target-plaintext key collision on 9-round AES-256.

Degree of Freedom in the Inbound Phase. In our attacks, we exploit the degrees of freedom of each inbound vertex as follows: $IN_1 = 2^1$, $IN_2 = 2^0$, $IN_3 = 2^0$, $IN_4 = 2^0$, $IN_5 = 2^{26}$, $IN_6 = 2^0$, $IN_7 = 2^0$, $IN_8 = 2^9$, $IN_9 = 2^{19}$, $IN_{10} = 2^6$, $IN_{11} = 2^{11}$ and $IN_{12} = 2^{19}$.

By using our automatic tool, we have obtained the differential probability of each inbound vertex as follows: $P(in_1) = 2^{-6}$, $P(in_2) = 2^{-6}$, $P(in_3) = 2^{-6}$, $P(in_4) = 2^{-6}$, $P(in_5) = 2^{-6}$, $P(in_6) = 2^{-7}$, $P(in_7) = 2^{-6}$, $P(in_8) = 2^{-7}$, $P(in_9) = 2^{-13}$, $P(in_{10}) = 2^{-13}$, $P(in_{11}) = 2^{-13}$, and $P(in_{12}) = 2^{-13}$. Based on these results, the maximum degrees of freedom in each vertex are estimated as $IN_{M_1} = 2^{26(=32-6)}$, $IN_{M_2} = 2^{26(=32-6)}$, $IN_{M_3} = 2^{26(=32-6)}$, $IN_{M_4} = 2^{26(=32-6)}$, $IN_{M_5} = 2^{26(=32-6)}$, $IN_{M_6} = 2^{25(=32-7)}$, $IN_{M_7} = 2^{26(=32-6)}$, $IN_{M_8} = 2^{25(=32-7)}$, $IN_{M_9} = 2^{19(=32-13)}$, $IN_{M_{10}} = 2^{19(=32-13)}$, $IN_{M_{11}} = 2^{19(=32-13)}$ and $IN_{M_{12}} = 2^{19(=32-13)}$. Thus, the degrees of freedom of each inbound vertex, which is required for the attack, are sufficiently available.

Attack Procedures.

1. Start with $2^{1(=1+0+0+0)}$ sets of $\{in_1, \ldots, in_4\} = \{x_7[0], \ldots, x_7[3]\}$ (red part in Fig. 10); then, obtain 2^1 sets of $\{w_7[3], \ldots, w_7[0]\}$ (blue part in Fig. 10).
2. Prepare 2^{26} values of $in_5 = x_8[3]$ (red part in Fig. 10); then, obtain $2^{27(=26+1)}$ values of $k_7[3]$ (purple part in Fig. 10). As the differential probability of the corresponding SW function in the key schedule part is 2^{-27}, there exist $2^{0(=27-27)}$ values that fulfill the differential characteristic (purple part in Fig. 10).
3. Prepare 2^0 values of $in_6 = x_8[2]$ (red part in Fig. 10); then, obtain $2^{0(=0+0)}$ values of $k_7[2]$ (green part in Fig. 10). As the differential probability of the corresponding key schedule part is 2^0, there exist $2^{(=0-0)}$ values that fulfill the differential characteristic (green part in Fig. 10).

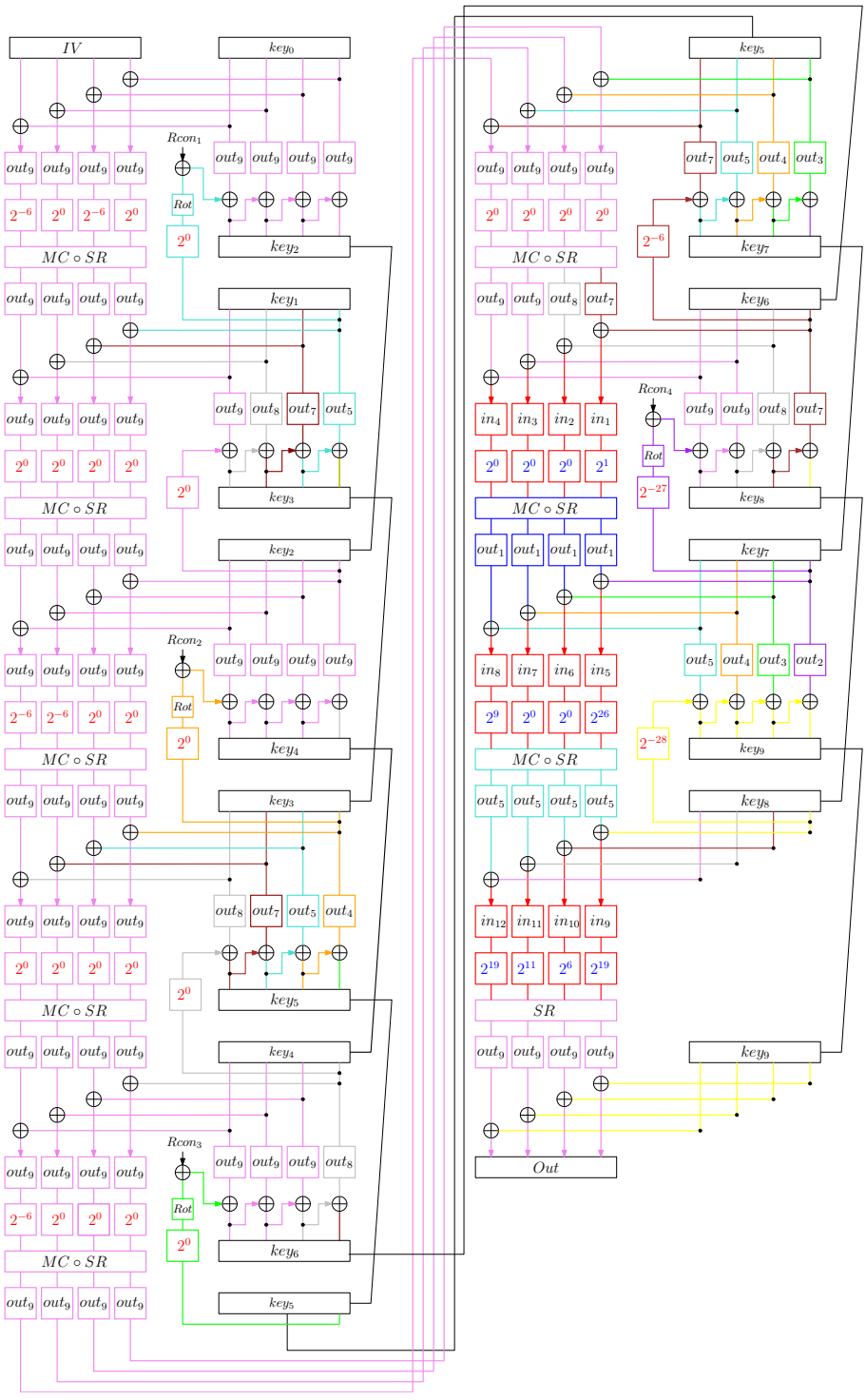


Fig. 10: Free-target-plaintext collision attack on 9-round AES-256.

4. Prepare 2^0 values of $in_7 = x_8[1]$ (red part in Fig. 10); then, obtain $2^{0(=0+0)}$ values of $k_7[1]$ (orange part in Fig. 10). As the differential probability of the corresponding key schedule part is 2^0, there exist $2^{0(=0-0)}$ values that fulfill the differential characteristic (orange part in Fig. 10).
5. Prepare 2^9 values of $in_8 = x_8[0]$ (red part in Fig. 10); then, obtain $2^{9(=9+0)}$ values of $k_7[0]$ (turquoise part in Fig. 10). As the differential probability of the corresponding key schedule part is 2^0, there exists $1 = 2^{9(=9-0)}$ value that fulfills the differential characteristic (turquoise part in Fig. 10).
6. Prepare 2^{19} values of $in_9 = x_9[3]$ (red part in Fig. 10); then, obtain $2^{28(=19+9)}$ values of $k_8[3]$ (yellow part in Fig. 10). As the differential probability of the corresponding key schedule part is 2^{-28}, there exist $2^{0(=28-28)}$ values that fulfill the differential characteristic (yellow part in Fig. 10).
7. Prepare 2^6 values of $in_{10} = x_9[2]$ (red part in Fig. 10); then, obtain $2^{6(=0+6)}$ values of $k_8[2]$ (brown part in Fig. 10). As the differential probability of the corresponding key schedule part is 2^{-6}, there exist $2^{0(=6-6)}$ values that fulfill the differential characteristic (brown part in Fig. 10).
8. Prepare 2^{11} values of $in_{11} = x_9[1]$ (red part in Fig. 10); then, obtain $2^{11(=0+11)}$ values of $k_8[1]$ (gray part in Fig. 10). As the differential probability of the corresponding key schedule part is 2^0, there exist $2^{11(=11-0)}$ values that fulfill the differential characteristic (gray part in Fig. 10).
9. Prepare 2^{19} values of $in_{12} = x_9[0]$ (red part in Fig. 10); then, obtain $2^{30(=11+19)}$ values of $k_8[0]$ (violet part in Fig. 10). As the differential probability of the remaining parts is 2^{-30}, there exists $1 = 2^{0(=30-30)}$ value that fulfills the differential characteristic (violet part in Fig. 10).

Attack Complexity. Following the above attack procedures, it is obvious that the attack complexity for Step 9 is dominant, which requires approximately 2^{30} computations of the partial AES-256 encryption. Therefore, total complexity is bounded by 2^{30} computations of 9-round AES-256.

Experimental Verification. We experimentally verify the validity of the proposed attack, especially of the free-target-plaintext collision attack on 9-round AES-256. Our experiment was executed on AMD Ryzen Threadripper™PRO 5995WX @2.7GHz (64C/128T) with 512GB RAM; then, it was completed in 12 hours. As a result, we have practically found the value at which such a key collision occurs. Table 3 shows an example case of a free-target-plaintext key collision for 9-round AES-256 (see Table 4 in Appendix E for more details), and the values with differences in the keys are shown in red.

6 Application to AES-DM

In this section, we present single-block (semi-free-start) collision attacks on AES-DM, which are based on the fixed/free-target-plaintext key collision attacks on AES provided in Sect. 5. Moreover, we show two-block collision attacks on AES-128-DM and AES-256-DM, also found by our automatic tool in Sect. 4. Here, we

Table 3: An example input and output values of a free-target-plaintext key collision for 9-round AES-256. It is directly converted into a semi-free start collision attack on 9-round AES-256-DM by adding the feedword operation.

i	$Plaintext_i$	Key_i	$Ciphertext_i$
1	83 66 63 dc b1 bc 61 82 30 38 ab f7 14 c3 d4 6a	ca 45 20 ea 26 11 ac 9c 30 3c c2 06 7c 39 55 e2 7e 2f d9 46 84 1f b2 3e 96 2a 82 ef 21 00 57 6c	7f ea d8 40 c0 59 30 d5 11 29 07 d0 39 08 5a 65
2	83 66 63 dc b1 bc 61 82 30 38 ab f7 14 c3 d4 6a	35 45 20 ea 26 11 ac 9c cf 3c c2 06 7c 39 55 e2 94 5a ac d9 84 1f b2 3e 7c 5f f7 70 21 00 57 6c	7f ea d8 40 c0 59 30 d5 11 29 07 d0 39 08 5a 65

only show the attack on AES-256-DM, and the attack on AES-128-DM is shown in Appendix F.

Notations. Unlike the notations in Sect. 5, the internal states of AES are treated here as a byte-wise array. Then, we denote the i-th round of the state array at the m-th row from the left and the n-th column from the top by $x_i[4m+n]$ for $m, n \in \{0, 1, 2, 3\}$. For example, $x_i[6]$ is represented as the i-th round state array at the 1st row from the left and the 2nd column from the top.

6.1 Single-block (Semi-Free-Start) Collision Attacks on AES-DM

As discussed in Sect. 3.2, fixed-target-plaintext key collisions and free-target-plaintext key collisions on AES-128/192/256 are naturally converted into one-block collision and semi-free collision attacks on AES-DM, respectively. Thus, we can construct collision attacks on 2/5/6 round AES-DM-128/192/256 and semi-free-start collision attacks on 5/7/9 round AES-DM-128/192/256, respectively.

6.2 How to Find Two-block Collision Attack on AES-DM

We show that a class of single-block free-start collision attacks on AES-DM is converted into 2-block collision attacks. Specifically, in the second block, we prepare a specific class of free-start collision attacks in which an input chaining values could have any difference Δh_i, but its value h_i is predetermined, while in the setting of the standard free-start collision, both of value and difference of input chaining values are freely chosen by the adversary. We call this type of collision *free-differential-start collision*, defined as follows.

Definition 2 (Free-Differential-Start Collision) *Given a compression function CF, it finds a pair (v, m) and (v', m'), so that $CF(v, m) = CF(v', m')$, where v is a fixed value and $(v \neq v')$.*

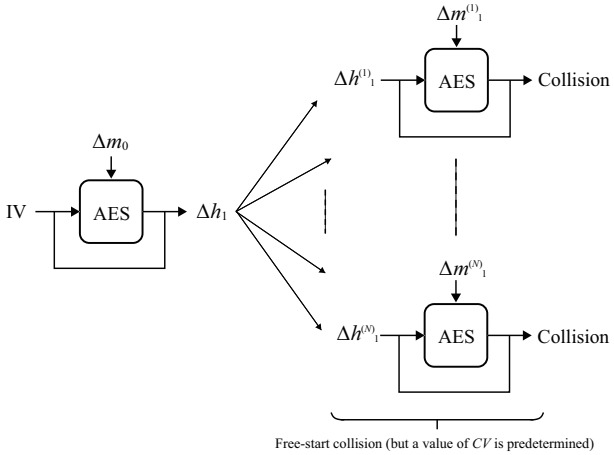


Fig. 11: Overview of the proposed two-block collision attack.

Basic Idea. Suppose that a free-differential-start collision in the second block can be found with a time complexity of $T_2 < 2^{64}$. Furthermore, we assume that such collisions can be obtained, along with N different input differences of $\Delta h_1^{(1)}, \Delta h_1^{(2)}, \ldots, \Delta h_1^{(N)}$ with corresponding message differences of $\Delta m_1^{(1)}, \Delta m_1^{(2)}$, $\ldots, \Delta m_1^{(N)}$, and each of them can be found with the same complexity of $T_2 < 2^{64}$. It is expressed as follows:

$$\Delta CF(\Delta h_1^{(1)}, \Delta m_1^{(1)}) = \Delta CF(\Delta h_1^{(2)}, \Delta m_1^{(2)}) = \cdots = \Delta CF(\Delta h_1^{(N)}, \Delta m_1^{(N)}) = 0.$$

We first compute the first block by randomly choosing a pair of input messages m_0. Suppose we obtain a pair of h_1 having a difference that is equal to one of N patterns of $\Delta h_1^{(1)}, \Delta h_1^{(2)}, \ldots, \Delta h_1^{(N)}$. In that case, we mount a free-differential-start collision attack on the second block where a pair of input h_1 is fixed to the value identified in the search in the first block, and Δh_1 corresponds to one of N candidates. Consequently, this approach enables us to connect between the first and second blocks, thereby finding a collision in a two-block AES-DM as shown in Fig. 11.

Attack Complexity. The probability that Δh_1 coincides with one of N specified patterns of $\Delta h_1^{(1)}, \Delta h_1^{(2)}, \ldots, \Delta h_1^{(N)}$ is estimated to be $N/2^{128}$. Therefore, upon collecting $2^{128}/N$ pairs of h_1, we can find such a pair with high probability. Due to the birthday paradox, $2^{128}/N$ pairs can be prepared from $\sqrt{2^{128}/N} = 2^{64}/N^{\frac{1}{2}}$ values of h_i. Thus, the total complexity is estimated as

$$2^{64}/N^{\frac{1}{2}} \text{ one block comp.} + T_2 \text{ one block comp.}$$

For example, if $N = 4$ and $T_2 = 2^{62}$, the time complexity is estimated as

$$2^{63} \text{ one block comp.} + 2^{62} \text{ one block comp.} < 2^{63} \text{ two block comp.}$$

The memory requirements of $2^{64}/N^{\frac{1}{2}}$ in the first block.

6.3 Two-Block Collision Attacks on 9-round AES-256-DM

Using our automatic tool, we can develop a 9-round free-differential-start collision in the second block with a time complexity of 2^{58} as shown in Fig. 12. To convert it into a two-block collision attack, we need to construct multiple attacks with N distinct input differences Δh_i and the same time complexity.

How to Find N Distinct Inputs. We can easily obtain such attacks by exploiting the differential characteristics in Fig. 13 (see Fig. 33 in Appendix G for more details). It is well known that given a fixed input and output difference of Δx and Δy, the probability of $(\Delta y = Sbox(\Delta x))$ is approximately $1/2$ where $Sbox()$ is an operation of S-box of AES [37].

This property indicates that there are about 128 distinct differences of $\Delta x_0[7]$, which result in $\Delta y_0[7] = 0$x33 through S-box. It means that $\Delta h_1[7](= \Delta x_0[7] \oplus \Delta k_0[7])$ also has 128 possible values, which lead to differential characteristics of y_0.

$\Delta h_1[7]$ is forwarded to the output, and $\Delta y_8[7]$ is computed as $\Delta y_8[7] = \Delta k_9[7] \oplus \Delta h_1[7]$. Once $\Delta h_1[7]$ is chosen out of 128 candidates, the corresponding $\Delta y_8[7]$ is determined. The probability that $(\Delta y_8[7] = Sbox(\Delta x_8[7] = 0\text{xfd}))$ is $1/2$. Thus, there exists $128/2 = 64$ possible $\Delta h_1[7]$, which follow the characteristics for collision with time complexity of 2^{58}. As $\Delta h_1[5]$ and $\Delta h_1[6]$ also have 64 possible candidates, respectively, with the same reason, in total, there are $N = (64)^3 = 2^{18}$ distinct inputs with the same time complexity.

Attack Complexity. For $N = 2^{18}$ and $T_2 = 2^{58}$, the time complexity for 2-block collision attack is estimated as

$$2^{55} \text{ one block comp.} + 2^{58} \text{ one block comp.} < 2^{58} \text{ two block comp.}$$

The memory requirements of 2^{55} in the first block.

7 Conclusion

In this paper, we investigated the new type of key collisions called target-plaintext key collisions of AES, which arise as an open problem in the key committing security and are directly converted into collision attacks on Davies-Meyer (DM) hashing mode. This key collision is such that a ciphertext collision is uniquely observed when a specific plaintext is encrypted under two distinct keys. We introduced an efficient automatic search tool designed to find target-plaintext key collisions. As a result, we demonstrated collision attacks on 2/5/6-round AES-128/192/256-DM and semi-free-start collision attacks on 5/7/9-round AES-128/192/256-DM, respectively. Furthermore, by exploiting a specific class of

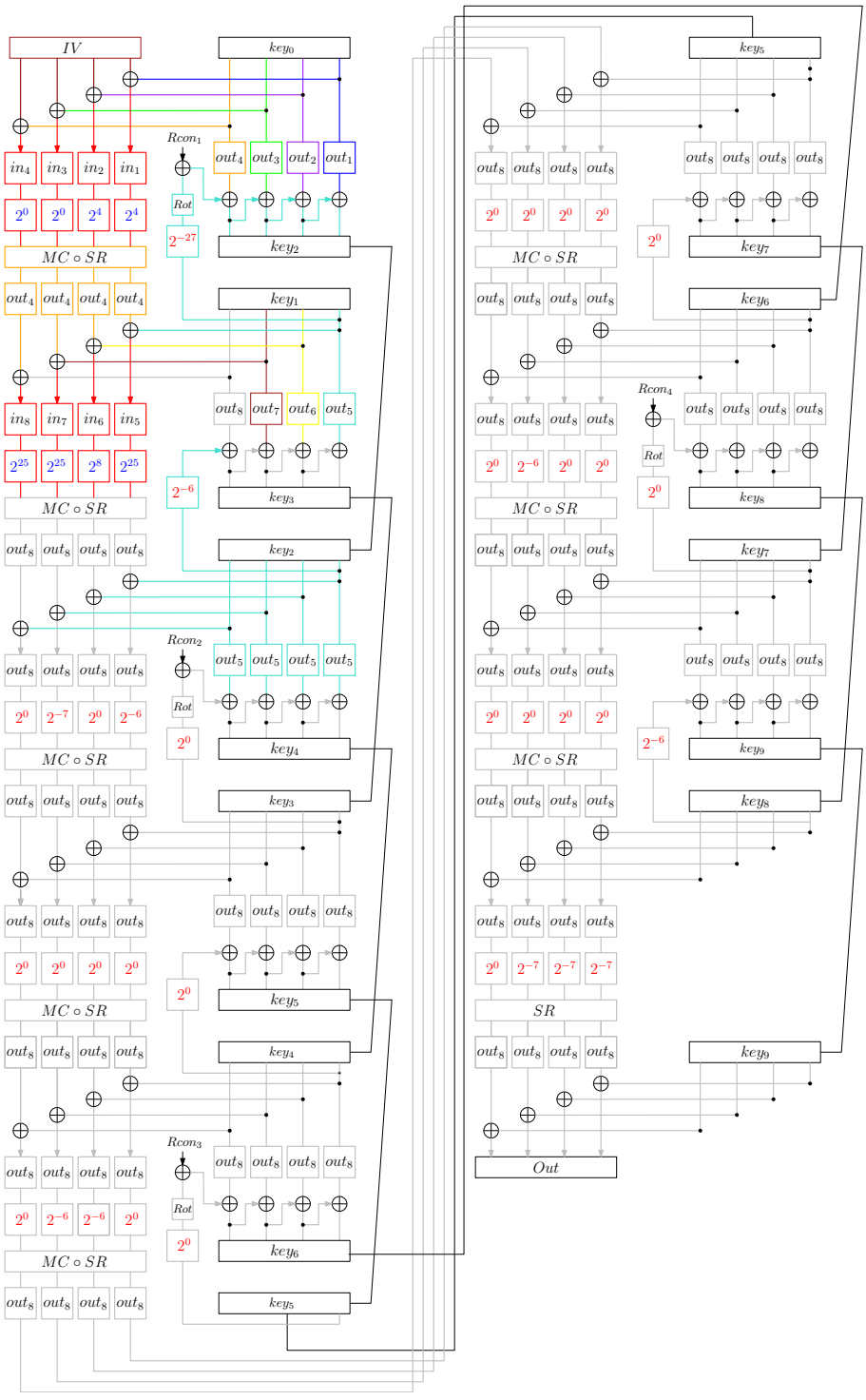


Fig. 12: Free-differential-start collision on 9-round AES-256-DM.

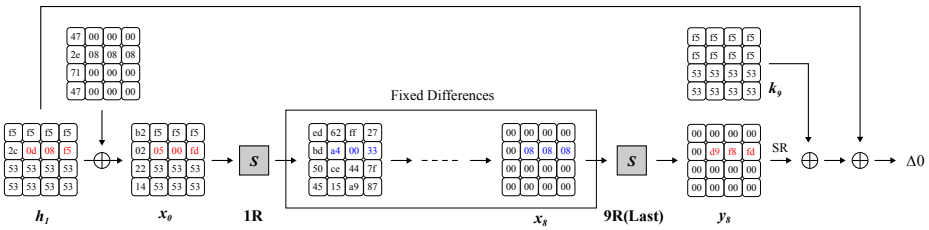


Fig. 13: Differential characteristic for the free-differential-start collision on 9-round AES-256-DM.

free-start collisions, we present collision attacks on 3/9-round AES-128/256-DM, respectively.

For further directions, it would be interesting to optimize how to determine the best choice of inbound vertices. Also, efficiently identifying invalid starting points would be promising to improve the efficiency of our method.

Acknowledgements

This result is obtained from the commissioned research (JPJ012368C05801) by the National Institute of Information and Communications Technology (NICT), Japan. This work was also supported by JSPS KAKENHI Grant Number JP24H00696.

References

1. Albertini, A., Duong, T., Gueron, S., Kölbl, S., Luykx, A., Schmieg, S.: How to abuse and fix authenticated encryption without key commitment. In: Butler, K.R.B., Thomas, K. (eds.) 31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022. pp. 3291–3308. USENIX Association (2022), https://www.usenix.org/conference/usenixsecurity22/presentation/albertini
2. Aumasson, J., Jr., J.N., Sepehrdad, P.: Cryptanalysis of the ISDB scrambling algorithm (MULTI2). In: Dunkelman, O. (ed.) Fast Software Encryption, 16th International Workshop, FSE 2009, Leuven, Belgium, February 22-25, 2009, Revised Selected Papers. Lecture Notes in Computer Science, vol. 5665, pp. 296–307. Springer (2009). https://doi.org/10.1007/978-3-642-03317-9_18, https://doi.org/10.1007/978-3-642-03317-9_18
3. Baek, S., Cho, S., Kim, J.: Quantum cryptanalysis of the full aes-256-based davies-meyer, hirose and MJH hash functions. Quantum Inf. Process. **21**(4), 163 (2022)
4. Barreto, P.S.L.M., Rijmen, V.: The whirlpool secure hash function. In: Submitted to NESSIE (updated) (2003)
5. Becker, G.T., Regazzoni, F., Paar, C., Burleson, W.P.: Stealthy dopant-level hardware trojans: extended version. J. Cryptogr. Eng. **4**(1), 19–31 (2014). https://doi.org/10.1007/S13389-013-0068-0, https://doi.org/10.1007/s13389-013-0068-0

6. Benadjila, R., Billet, O., Gilbert, H., Macario-Rat, G., Peyrin, T., Robshaw, M., Seurin, Y.: Sha-3 proposal: Echo. In: Submission to NIST (updated) (2009)
7. Biryukov, A., Khovratovich, D.: Related-key cryptanalysis of the full AES-192 and AES-256. In: ASIACRYPT. Lecture Notes in Computer Science, vol. 5912, pp. 1–18. Springer (2009)
8. Biryukov, A., Khovratovich, D., Nikolic, I.: Distinguisher and related-key attack on the full AES-256. In: CRYPTO. Lecture Notes in Computer Science, vol. 5677, pp. 231–249. Springer (2009)
9. Biryukov, A., Nikolic, I.: Automatic search for related-key differential characteristics in byte-oriented block ciphers: Application to aes, camellia, khazad and others. In: EUROCRYPT. Lecture Notes in Computer Science, vol. 6110, pp. 322–344. Springer (2010)
10. Biryukov, A., Nikolic, I.: Colliding keys for SC2000-256. In: Joux, A., Youssef, A.M. (eds.) Selected Areas in Cryptography - SAC 2014 - 21st International Conference, Montreal, QC, Canada, August 14-15, 2014, Revised Selected Papers. Lecture Notes in Computer Science, vol. 8781, pp. 77–91. Springer (2014). https://doi.org/10.1007/978-3-319-13051-4_5, https://doi.org/10.1007/978-3-319-13051-4_5
11. Boura, C., Derbez, P., Funk, M.: Related-key differential analysis of the AES. IACR Trans. Symmetric Cryptol. **2023**(4), 215–243 (2023)
12. Chen, L.: Recommendation for key derivation using pseudorandom functions. NIST SP 800-108r1 (2022)
13. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Information Security and Cryptography, Springer (2002)
14. Delaune, S., Derbez, P., Vavrille, M.: Catching the fastest boomerangs application to SKINNY. IACR Trans. Symmetric Cryptol. **2020**(4), 104–129 (2020)
15. Derbez, P., Euler, M., Fouque, P., Nguyen, P.H.: Revisiting related-key boomerang attacks on AES using computer-aided tool. In: ASIACRYPT (3). Lecture Notes in Computer Science, vol. 13793, pp. 68–88. Springer (2022)
16. Derbez, P., Fouque, P., Jean, J.: Faster chosen-key distinguishers on reduced-round AES. In: INDOCRYPT. Lecture Notes in Computer Science, vol. 7668, pp. 225–243. Springer (2012)
17. Dong, X., Guo, J., Li, S., Pham, P.: Triangulating rebound attack on aes-like hashing. In: CRYPTO (1). Lecture Notes in Computer Science, vol. 13507, pp. 94–124. Springer (2022)
18. Dong, X., Qin, L., Sun, S., Wang, X.: Key guessing strategies for linear key-schedule algorithms in rectangle attacks. In: EUROCRYPT (3). Lecture Notes in Computer Science, vol. 13277, pp. 3–33. Springer (2022)
19. Dong, X., Sun, S., Shi, D., Gao, F., Wang, X., Hu, L.: Quantum collision attacks on aes-like hashing with low quantum random access memories. In: ASIACRYPT (2). Lecture Notes in Computer Science, vol. 12492, pp. 727–757. Springer (2020)
20. Dong, X., Zhang, Z., Sun, S., Wei, C., Wang, X., Hu, L.: Automatic classical and quantum rebound attacks on aes-like hashing by exploiting related-key differentials. In: ASIACRYPT (1). Lecture Notes in Computer Science, vol. 13090, pp. 241–271. Springer (2021)
21. Fouque, P., Jean, J., Peyrin, T.: Structural evaluation of AES and chosen-key distinguisher of 9-round AES-128. In: CRYPTO (1). Lecture Notes in Computer Science, vol. 8042, pp. 183–203. Springer (2013)
22. Gauravaram, P., Knudsen, L.R., Matusiewicz, K., Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: Grøstl - a SHA-3 candidate. In: Symmetric Cryptography. Dagstuhl Seminar Proceedings, vol. 09031. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany (2009)

23. Gérault, D., Lafourcade, P., Minier, M., Solnon, C.: Revisiting AES related-key differential attacks with constraint programming. Inf. Process. Lett. **139**, 24–29 (2018)
24. Gérault, D., Lafourcade, P., Minier, M., Solnon, C.: Computing AES related-key differential characteristics with constraint programming. Artif. Intell. **278** (2020)
25. Gérault, D., Minier, M., Solnon, C.: Constraint programming models for chosen key differential cryptanalysis. In: CP. Lecture Notes in Computer Science, vol. 9892, pp. 584–601. Springer (2016)
26. Gilbert, H., Peyrin, T.: Super-sbox cryptanalysis: Improved attacks for aes-like permutations. In: FSE. Lecture Notes in Computer Science, vol. 6147, pp. 365–383. Springer (2010)
27. Grassi, L., Leander, G., Rechberger, C., Tezcan, C., Wiemer, F.: Weak-key distinguishers for AES. In: SAC. Lecture Notes in Computer Science, vol. 12804, pp. 141–170. Springer (2020)
28. Hadipour, H., Bagheri, N., Song, L.: Improved rectangle attacks on SKINNY and CRAFT. IACR Trans. Symmetric Cryptol. **2021**(2), 140–198 (2021)
29. Hosoyamada, A., Sasaki, Y.: Finding hash collisions with quantum computers by using differential trails with smaller probability than birthday bound. In: EUROCRYPT (2). Lecture Notes in Computer Science, vol. 12106, pp. 249–279. Springer (2020)
30. Jean, J., Naya-Plasencia, M., Peyrin, T.: Multiple limited-birthday distinguishers and applications. In: Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 8282, pp. 533–550. Springer (2013)
31. Kelsey, J., Schneier, B., Wagner, D.A.: Key-schedule cryptanalysis of idea, g-des, gost, safer, and triple-des. In: Koblitz, N. (ed.) Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings. Lecture Notes in Computer Science, vol. 1109, pp. 237–251. Springer (1996). https://doi.org/10.1007/3-540-68697-5_19, https://doi.org/10.1007/3-540-68697-5_19
32. Kim, H., Park, M., Cho, J., Kim, J., Kim, J.: Weaknesses of some lightweight blockciphers suitable for iot systems and their applications in hash modes. Peer-to-Peer Netw. Appl. **13**(2), 489–513 (2020)
33. Lamberger, M., Mendel, F., Rechberger, C., Rijmen, V., Schläffer, M.: Rebound distinguishers: Results on the full whirlpool compression function. In: ASIACRYPT. Lecture Notes in Computer Science, vol. 5912, pp. 126–143. Springer (2009)
34. Lin, L., Wu, W., Zheng, Y.: Improved automatic search tool for related-key differential characteristics on byte-oriented block ciphers. In: ISC. Lecture Notes in Computer Science, vol. 10599, pp. 58–76. Springer (2017)
35. Matsui, M.: Key collisions of the RC4 stream cipher. In: Dunkelman, O. (ed.) Fast Software Encryption, 16th International Workshop, FSE 2009, Leuven, Belgium, February 22-25, 2009, Revised Selected Papers. Lecture Notes in Computer Science, vol. 5665, pp. 38–50. Springer (2009). https://doi.org/10.1007/978-3-642-03317-9_3, https://doi.org/10.1007/978-3-642-03317-9_3
36. Mendel, F., Peyrin, T., Rechberger, C., Schläffer, M.: Improved cryptanalysis of the reduced grøstl compression function, ECHO permutation and AES block cipher. In: Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 5867, pp. 16–35. Springer (2009)
37. Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: The rebound attack: Cryptanalysis of reduced whirlpool and grøstl. In: FSE. Lecture Notes in Computer Science, vol. 5665, pp. 260–276. Springer (2009)

38. Robshaw, M.: A cryptographic review of cipherunicorn-a. CRYPTRECT Technical report (2001)
39. Rouquette, L., Gérault, D., Minier, M., Solnon, C.: And rijndael?: Automatic related-key differential analysis of rijndael. In: AFRICACRYPT. Lecture Notes in Computer Science, vol. 13503, pp. 150–175. Springer Nature Switzerland (2022)
40. Sasaki, Y.: Meet-in-the-middle preimage attacks on AES hashing modes and an application to whirlpool. In: Joux, A. (ed.) Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers. Lecture Notes in Computer Science, vol. 6733, pp. 378–396. Springer (2011). https://doi.org/10.1007/978-3-642-21702-9_22, https://doi.org/10.1007/978-3-642-21702-9_22
41. Sasaki, Y., Li, Y., Wang, L., Sakiyama, K., Ohta, K.: Non-full-active super-sbox analysis: Applications to ECHO and grøstl. In: ASIACRYPT. Lecture Notes in Computer Science, vol. 6477, pp. 38–55. Springer (2010)
42. Sun, L., Wang, M.: Sok: Modeling for large s-boxes oriented to differential probabilities and linear correlations. IACR Trans. Symmetric Cryptol. **2023**(1), 111–151 (2023)
43. Sun, L., Wang, W., Wang, M.: More Accurate Differential Properties of LED64 and Midori64. IACR Trans. Symmetric Cryptol. **2018**(3), 93–123 (2018)
44. Sun, L., Wang, W., Wang, M.: Accelerating the Search of Differential and Linear Characteristics with the SAT Method. IACR Cryptol. ePrint Arch. p. 213 (2021)
45. Sun, S., Gérault, D., Lafourcade, P., Yang, Q., Todo, Y., Qiao, K., Hu, L.: Analysis of aes, skinny, and others with constraint programming. IACR Trans. Symmetric Cryptol. **2017**(1), 281–306 (2017)

A Existing Tools for Key Collision Search on AES

As mentioned in Sect. 1.1, no studies have been reported to date that have addressed the task of identifying target-plaintext key collisions in AES. For this, there are no existing tools yet that address this task directly. However, some existing tools have the potential to solve this task indirectly. For example, search tools for related-key differential characteristics on AES, collision attacks on AES-DM, and collision attacks on other AES-like hashing modes (e.g., MMO and MP modes). In this section, we outline these existing tools and then discuss the difficulty of applying them to finding the key collision on AES.

A.1 Existing Tools and Their Applications on AES-like Schemes

Related-key Differential Characteristics Search on AES. Based on Definition 1, the problems of searching for target-plaintext key collisions can be considered equivalent to the problem of searching for related-key differential characteristics where both the input and output differences are zero and the key difference is non-zero. Search tools for related-key differential characteristics on AES can be roughly divided into two categories: search tools implemented from scratch in general-purpose programming languages [7, 9, 21, 27, 34, 36] and automatic search tools based on optimization problems (e.g., MILP, SAT/SMT, CP) [11, 15, 23–25, 39, 45]. In the following, we mainly focus on the second category.

The first automatic search tool for finding related-key differential characteristics on AES was proposed by Gérault, Minier, and Solnon at CP 2016 [25]. Their tool is based on the CP-based approach, and their proposed algorithm consists of two steps. The first step is to search for all truncated characteristics with the lowest number of active S-boxes. The second step aims to decide whether each characteristic is valid or not. If it is valid, this step continually aims to find the actual cell values that maximize the differential probability. Although most of the proposed CP modeling methods rely on the existing techniques [9, 21], by adding new constraints, the proposed tool succeeded in obtaining tight characteristics faster and less memory than the existing techniques. The proposed tool also succeeded in proving that an optimal characteristic claimed in the existing studies [9, 21] is not optimal by providing a better solution. Since then, automatic search tools for finding tight related-key differential characteristics have developed based on their CP-based approach [11, 23, 24, 39, 45].

As another direction, there is a MILP-based tool proposed by Derbez et al. at ASIACRYPT 2022 [15]. Their tool is based on the existing MILP modeling techniques [14, 18, 28], but these techniques highly rely on the linearity of the key scheduling function in the target cipher; thus, they cannot be applied to the analysis of AES directly. Then, the authors in [15] tweaked the existing techniques to find related-key boomerang characteristics for AES-192 and AES-256. Since then, the latest automatic search tools have been provided by Boura, Derbez, and Funk at ToSC 2023(4) [11]. Both the existing CP-based and MILP-based approaches [15, 39] have a potential issue of requiring many additional variables

and constraints to avoid obtaining invalid truncated differential characteristics. Then, the authors in [11] tackled this issue and proposed two types of search tools to obtain a correct solution at high speed: one is based on a new MILP-based approach, and the other is a new ad-hoc algorithm based on dynamic programming. It can be seen from Table 2 in [11] that the new MILP-based approach is the fastest for finding optimal characteristics on AES-128 and AES-192, whereas the new dynamic programming-based approach is the fastest for finding optimal characteristics on AES-256.

Collision Attacks on AES-DM. As far as we know, no studies have been reported on an automatic search tool based on optimization problems for collision attacks on AES-DM. Then, we outline here the existing collision attacks on AES-DM from the following two approaches: one is an approach based on applying a known-key or chosen-key distinguisher to a free-strat collision attack [30,36], and the other is an approach based on a notion of differential q-multicollision [3,8,32].

At SAC 2009, Mendel et al. [36] improved the existing rebound attack [37], and the improved method was applied to Grøstl-256, ECHO, and AES. Focusing on AES, they provided the 7-round known-key distinguisher with a time complexity of 2^{24} and showed that the provided distinguisher can be applied to a free-start collision attack on the 5-round AES-DM with a time complexity of 2^{56}. This free-start collision attack is based on the fact that both the first and the sixth rounds in the 7-round distinguisher have the same active byte positions. At SAC 2013, Jean, Naya-Plasencia, and Peyrin [30] showed that the chosen-key distinguisher provided in [16] can be applied to a free-start collision attack on the 6-round AES-DM with a time complexity of 2^{32}, which is the same way as Mendel et al.'s method [36].

At CRYPTO 2009, Biryukov, Khovratovich, and Nikolic [8] provided the full-round chosen-key distinguisher on AES-256 based on a new notion of a differential q-multicollision. Then, by exploiting the provided distinguisher, they demonstrated that q pseudo-collision (i.e., q free-start collision) attacks on the full-round AES-256-DM can be done with a time complexity of $q \cdot 2^{67}$. In 2020, Kim et al. [32] claimed that the chosen-key distinguisher provided in [8] is insufficient for a free-start collision attack on AES-256-DM, as it is inferior to the birthday attack in terms of the generic notion of collision attacks. Then, to show a valid attack on AES-256-DM, they provided a new chosen-key distinguisher on the 12-round AES-256 and demonstrated a free-start collision attack on the 12-round AES-256-DM with a time complexity of $2^{61.3}$ by exploiting the provided distinguisher. In the latest study, Beck, Cho, and Kim [3] presented a quantum free-start collision attack on the full-round AES-256-DM, which is based on the chosen-key distinguisher provided in [8]. However, we identify that these attacks [3,32] are invalid for the following reasons. These attacks rely on 2-block collision strategies, wherein, a 1-block free-start collision in the second block is connected to chaining values which are computed from IV (Initial Value) of the hash function in the first block. In these attacks, triangulation algorithm [8], which exploits the degree of freedom of parts of key and plaintext in an under-

lying block cipher, is used for finding a free-start collision in the second block. However, in the situation of the 2-block collision attack, as input chaining values (namely plaintext part) to the second block are predetermined, there is no degree of freedom in these inputs (plaintext parts). Thus, a freestart collision in the second block cannot be constructed with a complexity of less than 2^{64}.

Collision Attacks on Other AES-like Hashing Modes. One of the powerful collision attacks on other AES-like hashing modes is a rebound attack [37], including its extended versions based on super S-box [26, 29, 33], non-full-active super S-box [19, 20, 41], and super inbound [17] techniques. Then, we outline here the existing studies [17, 19, 20, 29] used an automatic search tool based on optimization problems.

At EUROCRYPT 2020, Hosoyamada and Sasaki [29] proposed a dedicated quantum collision attack on AES-MMO, AES-MP, and Whirlpool, which is based on the classical rebound attack [37]. They applied the super S-boxes technique to the inbound phase in the proposed attack. Note that they used the MILP-based tool only to verify the optimality of the differential trail; thus, the automatic search tool was not used to automate the rebound attack.

At ASIACRYPT 2020, Dong et al. [19] proposed classical and quantum (semi-free-start) collision attacks on AES-MMO, AES-MP, and Grøstl, which is also based on the rebound attack. They applied the non-full-active super S-boxes technique to the inbound phase in the proposed attack and used the MILP-based tool to search for a valid truncated differential characteristic with non-full-active super S-boxes. At ASIACRYPT 2021, Dong et al. [20] proposed classical and quantum (free-start) collision attacks on Whirlpool, Saturnin-hash, SKINNY-MMO, and SKINNY-MP. This is similar to Dong et al.'s attack [19], but they focused on searching for a free-start collision on the target ciphers. For this, they introduced the MILP-based tool to determine a related-key truncated differential characteristic with non-full-active super S-boxes, which were optimized for rebound attacks. At CRYPTO 2022, Dong et al. [17] first proposed a new technique called super inbound for rebound attacks by generalizing the super S-boxes and non-full-active super S-boxes techniques. Then, they applied the super inbound technique to the inbound phase in the rebound attack and used the CP-based tool to search for a valid truncated differential characteristic with a super-inbound property.

A.2 Limitations on Existing Results and Tools for Key Collisions

Several existing methods described in Appendix A.1 have the potential to apply to finding the key collision on AES. In this subsection, we discuss the difficulty of applying them to finding the key collision on AES.

Related-key Differential Characteristics. The natural way to find the key collision is to utilize the automatic search tools for differential characteristics in the related-key model, i.e., searching the related-key differential characteristics

with the differences in both the plaintext and ciphertext fixed to zero. The most straightforward approach is to collect and encrypt the plaintext pairs following a differential characteristic with a probability of greater than 2^{-64}, which can find the free-target-plaintext key collision. However, according to [39], optimal related-key differential characteristics of AES become less than the probability of 2^{-64} over 4, 7, and 9 rounds for the 128-, 192-, and 256-bit key variants, respectively. Considering the differences in the plaintext and ciphertext fixed to zero, the number of rounds that we can construct differential characteristics for the key collision could decrease more. Hence, this approach does not seem suitable for finding the key collision. However, we can generally control the internal state in the chosen-key setting, leading to a more sophisticated technique as described next.

Rebound Attacks. The rebound attack is one of such sophisticated techniques applying to hash functions [37]. For the hash functions based on AES-like permutations, such as ECHO [6], Whirlpool [4], and Grøstl [22], *the super S-box technique* [26, 29, 33] is often employed to efficiently collect the starting point of the inbound phase, called the degrees of freedom (DoF). The concept of the super S-box technique is to collect a lot of value pairs in the input and output of super S-boxes and check the validation by the meet-in-the-middle technique. In that case, we can no longer find the fixed-target-plaintext key collision because value pairs in the input of super S-boxes can not be pre-determined, i.e., all key collisions that we can find by this technique are categorized to the free-target-plaintext key collision. Hence, we focus solely on the free-target-plaintext key collision by the rebound attack with the super S-box technique, equivalent to the semi-free-start collision on AES-DM.

To the best of our knowledge, there are no studies published on the semi-free-start collision on AES-DM by the rebound attack, despite their significance. This fact already implies its difficulty, but we take a closer look at it here. First of all, no differential characteristics of AES where a difference is inserted into only key, leading to a ciphertext collision has not been demonstrated in the literature. Compared to the search for best related-key differential characteristics for AES [9, 15, 21, 25], this search requires strict conditions such that key differences should be canceled out by themselves without the help of plaintext differences. Consequently, the weights of target differential characteristics become quite high, resulting in time-consuming tasks. Therefore, finding these characteristics within a practical time appears to be very challenging so far. Besides, even when differential characteristics for key collision are identified, rebound attacks [37] and triangle attacks [8], which efficiently find the values which fulfill differential characteristics, are not well-suited for solving target-plaintext key collisions.

Experiments. To briefly examine their capacities for finding key collisions on AES, we attempt to explore the truncated differential characteristics with well-known super S-box structures. Our evaluation is simple: putting well-known super S-box structures into arbitrary rounds as the inbound phase and then

counting the number of active S-boxes in the outbound phase and key scheduling with differences in both plaintext and ciphertext fixed to zero. As we can find one value pair by one super S-box operation on average, the DoF obtained by the super S-box technique is limited up to 2^{64}, that is, the number of active S-boxes must be less than 11 ($2^{-6\times11} < 2^{-64} < 2^{-6\times10}$).We give the overview of our evaluation in Fig. 14.

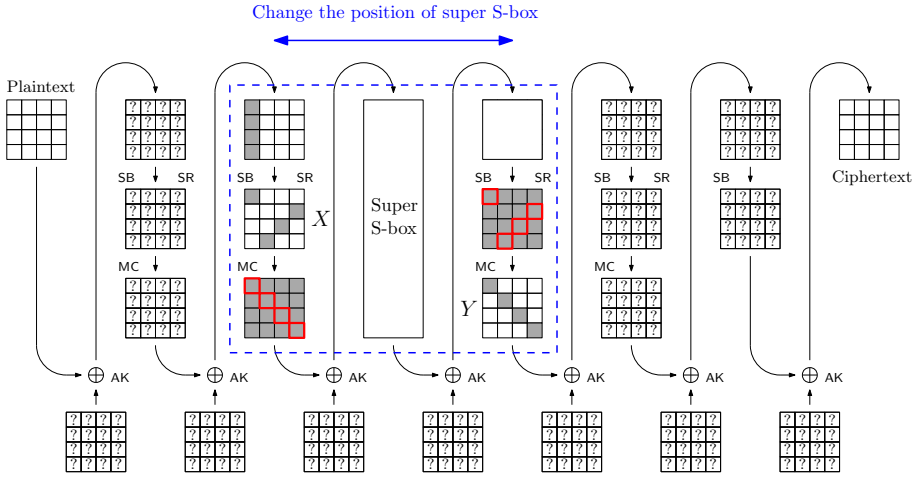


Fig. 14: The overview of our evaluation with an example of the 6-round AES. We check the popular four input and output patterns of the super S-box in X and Y. Our evaluation does not consider the linear equations from the round function and key scheduling shown in [11], namely, not to remove invalid truncated differential characteristics.

As a result of our evaluation, the number of active S-boxes outnumbers 11 ($2^{-6\times11} < 2^{-64}$) over 4, 5, and 4 rounds in AES-128, AES-192, and AES-256, respectively, implying that it can be difficult to find key collisions by these techniques. There are several novel techniques for extending the inbound phase including the non-active super S-box and the super inbound techniques [17, 41] for collision attacks on MMO and MP mode. However, we believe that their direct applications in finding key collisions pose significant difficulties. These difficulties arise from the fact that key differences are inserted in the inbound phase, and key differences should be canceled out by themselves, without plaintext and feedforward differences, unlike MMO and MP. Thus, it is difficult to construct specific differential characteristics that are well-suited for these techniques in the inbound phase. Furthermore, even if we succeed in extending the number of rounds in the inbound phase using these techniques, it is essential to consider the probability of causing a collision in the final sub-key addition, which requires additional time complexity.

Since our evaluation is straightforward, it would be notable that our experimental results do not show the impossibility of applying the super S-box technique but how difficult it is.

B Optimality for Fixed-Target-Plaintext Key Collision Attacks on 2-Round AES-128

Some might consider that our approach can achieve a very small number of rounds on AES-128 for fixed-target plaintext key collision attacks. To demonstrate the optimality of the 2-round attack, we derive tight bounds of the differential characteristic probability for key collisions on AES-128, where a difference is inserted into only key, not into plaintext, leading to a ciphertext collision.

As a result, the probability drops below 2^{-128} after 3 rounds. It means that in the fixed-target-plaintext scenario, no key collision pairs are guaranteed after 3 rounds for a given target plaintext, even when considering the entire 128-bit key space. This confirms that more than a 3-round attack is infeasible, even if an efficient rebound attack is applied, under the estimation of single differential characteristics.

C Key Collisions on AES-128/192

In this section, we show fixed/free-target-plaintext key collisions on AES-128/192, which are found by our automatic tool provided in Sect. 4.

Notations. We use the following notations for our attacks. For $i \in \{1, \ldots, 12\}$, let in_i be the inbound vertices; let IN_i be the assigned degrees of freedom in the attack; let IN_{M_i} be the available degrees of freedom from a given differential characteristic; and let $P(in_i)$ be a differential probability of in_i in the corresponding SB operation.

Moreover, the internal states of AES are treated here as a column-wise array of 4-byte words, with columns numbered from the left. For example, $x_i[0]$ and $x_i[3]$ are represented as 4-byte words in the leftmost and rightmost columns in the i-th round internal state before the SB operation, respectively.

C.1 Fixed-Target-Plaintext Key Collision on 2-round AES-128

Fig. 15a illustrates an underlying differential characteristic for a fixed-target-plaintext key collision for 2-round AES-128 with a probability of 2^{-98}. Assuming that the 1st round in the data processing part is an inbound phase (i.e., $\{in_1, \ldots, in_4\} = \{x_1[3], \ldots, x_1[0]\}$), and the remaining part, including the key scheduling part, is an outbound phase, the probability of inbound and outbound phases are 2^{-42} and 2^{-56}, respectively. After applying our tool, we can obtain the DoF tree as shown in Fig. 15b for collision attacks. By using this tree, we can construct a fixed-target-plaintext key collision on 2-round AES-128.

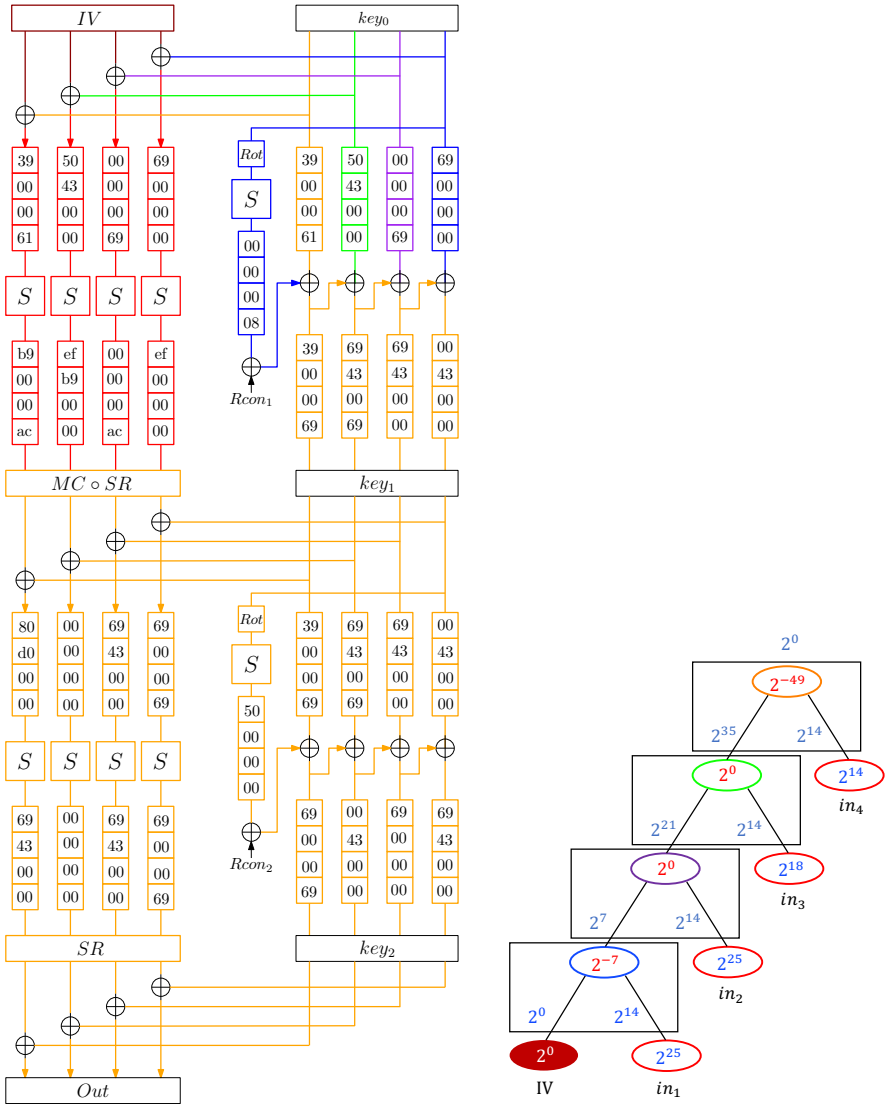
Degree of Freedom in the Inbound Phase. In our attacks, we exploit the degrees of freedom of each inbound vertex as follows: $IN_1 = 2^{14}$, $IN_2 = 2^{14}$, $IN_3 = 2^{14}$, and $IN_4 = 2^{14}$.

By using our automatic tool, we have obtained the differential probability of each inbound vertex as follows: $P(in_1) = 2^{-7}$, $P(in_2) = 2^{-7}$, $P(in_3) = 2^{-14}$, and $P(in_4) = 2^{-14}$. Based on these results, the maximum degrees of freedom in each vertex are estimated as $IN_{M_1} = 2^{25(=32-7)}$, $IN_{M_2} = 2^{25(=32-7)}$, $IN_{M_3} = 2^{18(=32-14)}$, and $IN_{M_4} = 2^{18(=32-14)}$. Thus, the degrees of freedom of each inbound vertex, which is required for the attack, are sufficiently available.

Attack Procedures.

1. Start with 2^{14} values of $in_1 = x_1[3]$ and the fixed initial value of $w_0[3]$ (red part in Fig. 16); then, obtain 2^{14} values of $k_0[3]$ (blue part in Fig. 16). As the differential probability of the SW function in the first round of the key schedule part is 2^{-7}, there exist $2^{7(=14-7)}$ values that fulfill the differential characteristic.
2. Prepare $2^{28(14+14)}$ sets of $\{in_2, in_3\} = \{x_1[2], x_1[1]\}$ and the fixed initial value sets of $\{w_0[2], w_0[1]\}$ (red part in Fig. 16); then, obtain $2^{35(=7+28)}$ sets of $\{k_0[2], k_0[1]\}$ (purple and green parts in Fig. 16).
3. Prepare 2^{14} values of $in_4 = x_1[0]$ and the fixed initial value of $w_0[0]$ (red part in Fig. 16); then, obtain $2^{49(=35+14)}$ values of the remaining data processing and key scheduling parts (orange parts in Fig. 16). As the differential probability of the remaining parts is 2^{-49}, there exists $1 = 2^{0(=49-49)}$ value that fulfills the differential characteristic (orange parts in Fig. 16).

Attack Complexity. Following the above attack procedures, it is obvious that the attack complexity for Step 3 is dominant, which requires approximately 2^{49} computations of the partial AES-128 encryption. Therefore, total complexity is bounded by 2^{49} computations of 2-round AES-128.



(a) Differential characteristic. (b) DoF tree.

Fig. 15: Fixed-target-plaintext collision attack on 2-round AES-128.

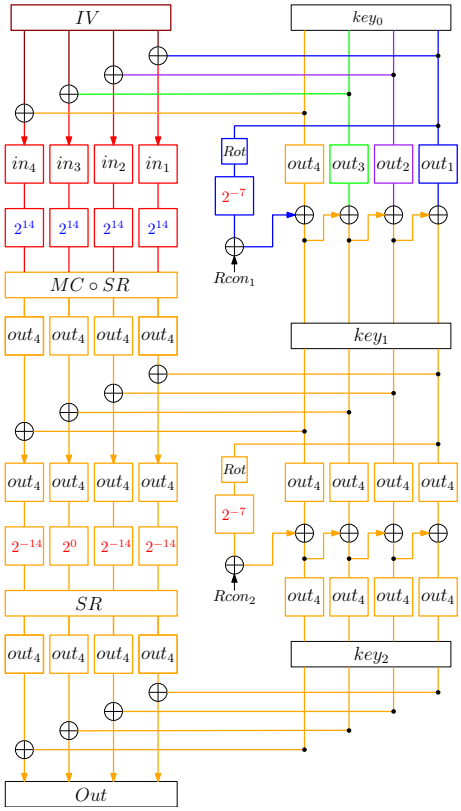


Fig. 16: Fixed-target-plaintext collision attack on 2-round AES-128.

C.2 Free-Target-Plaintext Key Collision on 5-round AES-128

Fig. 17 illustrates an underlying differential characteristic for a free-target-plaintext key collision for 5-round AES-128 with a probability of 2^{-251}. Assuming that the 2nd and 3rd rounds in the data processing part are an inbound phase (i.e., $\{in_1, \ldots, in_8\} = \{x_2[0], \ldots, x_2[3], x_3[3], \ldots, x_3[0]\}$), and the remaining part, including the key scheduling part, is an outbound phase, the probability of inbound and outbound phases are 2^{-187} and 2^{-64}, respectively. After applying our tool, we can obtain the DoF tree as shown in Fig. 18 for collision attacks. By using this tree, we can construct a free-target-plaintext key collision on 5-round AES-128.

Degree of Freedom in the Inbound Phase. In our attacks, we exploit the degrees of freedom of each inbound vertex as follows: $IN_1 = 2^{11}$, $IN_2 = 2^{20}$, $IN_3 = 2^4$, $IN_4 = 2^4$, $IN_5 = 2^4$, $IN_6 = 2^5$, $IN_7 = 2^{11}$ and $IN_8 = 2^5$.

By using our automatic tool, we have obtained the differential probability of each inbound vertex as follows: $P(in_1) = 2^{-21}$, $P(in_2) = 2^{-7}$, $P(in_3) = 2^{-28}$, $P(in_4) = 2^{-28}$, $P(in_5) = 2^{-28}$, $P(in_6) = 2^{-27}$, $P(in_7) = 2^{-21}$, and $P(in_8) = 2^{-27}$. Based on these results, the maximum degrees of freedom in each vertex are estimated as $IN_{M_1} = 2^{11(=32-21)}$, $IN_{M_2} = 2^{25(=32-7)}$, $IN_{M_3} = 2^{4(=32-28)}$, $IN_{M_4} = 2^{4(=32-28)}$, $IN_{M_5} = 2^{4(=32-28)}$, $IN_{M_6} = 2^{5(=32-27)}$, $IN_{M_7} = 2^{11(=32-21)}$ and $IN_{M_8} = 2^{5(=32-27)}$. Thus, the degrees of freedom of each inbound vertex, which is required for the attack, are sufficiently available.

Attack Procedures.

1. Start with $2^{39(=11+20+4+4)}$ sets of $\{in_1, \ldots, in_4\} = \{x_2[0], \ldots, x_2[3]\}$ (red part in Fig. 19); then, obtain 2^{39} sets of $\{w_2[0], \ldots, w_2[3]\}$ (blue part in Fig. 19).
2. Prepare 2^4 values of $in_5 = x_3[3]$ (red part in Fig. 19); then, obtain $2^{43(=39+4)}$ values of $k_2[3]$ (purple part in Fig. 19). As the differential probability of the SW function in the third round of the key schedule part is 2^{-7}, there exist $2^{36(=43-7)}$ values that fulfill the differential characteristic (purple part in Fig. 19).
3. Prepare 2^5 values of $in_6 = x_3[2]$ (red part in Fig. 19); then, obtain $2^{41(=36+5)}$ values of $k_2[2]$ (green part in Fig. 19). As the differential probability of the SW function in the second round of the key schedule part is 2^0, there exist $2^{41(=41-0)}$ values that fulfill the differential characteristic; then, obtain $2^{41(=41-0)}$ sets of $\{w_1[3], k_1[3]\}$ (green part in Fig. 19).
4. Prepare 2^{11} values of $in_7 = x_3[1]$ (red part in Fig. 19); then, obtain $2^{52(=41+11)}$ sets of $\{k_1[2], k_2[1]\}$ (orange part in Fig. 19). As the differential probability of the SW function in the second round of the key schedule part is 2^0, there exist $2^{52(=52-0)}$ values that fulfill the differential characteristic; then, obtain $2^{52(=52-0)}$ sets of $\{w_1[2], k_0[4]\}$ (orange part in Fig. 19).
5. Prepare 2^5 values of $in_8 = x_3[0]$ (red part in Fig. 19); then, obtain $2^{57(=52+5)}$ values of the remaining data processing and key scheduling parts (turquoise part in Fig. 19). As the differential probability of the remaining parts is 2^{-57},

there exists $1 = 2^{0(=57-57)}$ value that fulfills the differential characteristic (turquoise part in Fig. 19).

Attack Complexity. Following the above attack procedures, it is obvious that the attack complexity for Step 5 is dominant, which requires approximately 2^{57} computations of the partial AES-128 encryption. Therefore, total complexity is bounded by 2^{57} computations of 5-round AES-128.

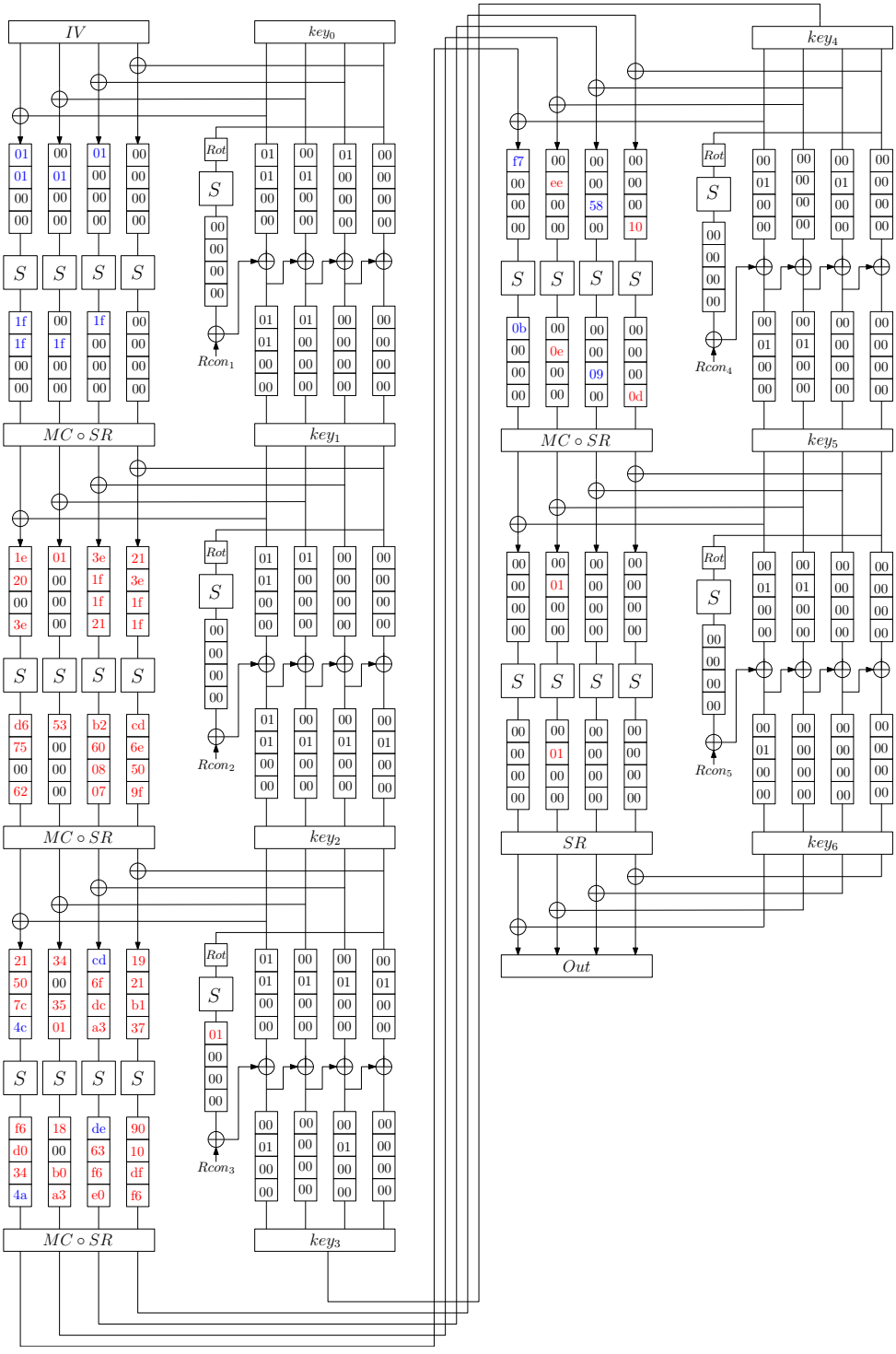


Fig. 17: Differential characteristic for a free-target-plaintext collision attack on 5-round AES-128.

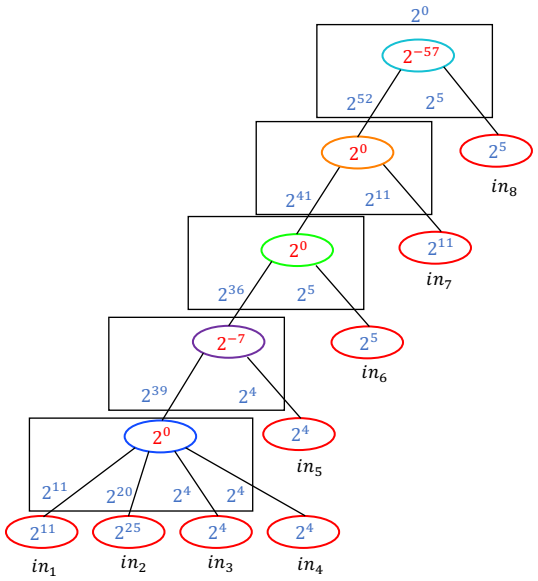


Fig. 18: DoF tree for a free-target-plaintext collision attack on 5-round AES-128.

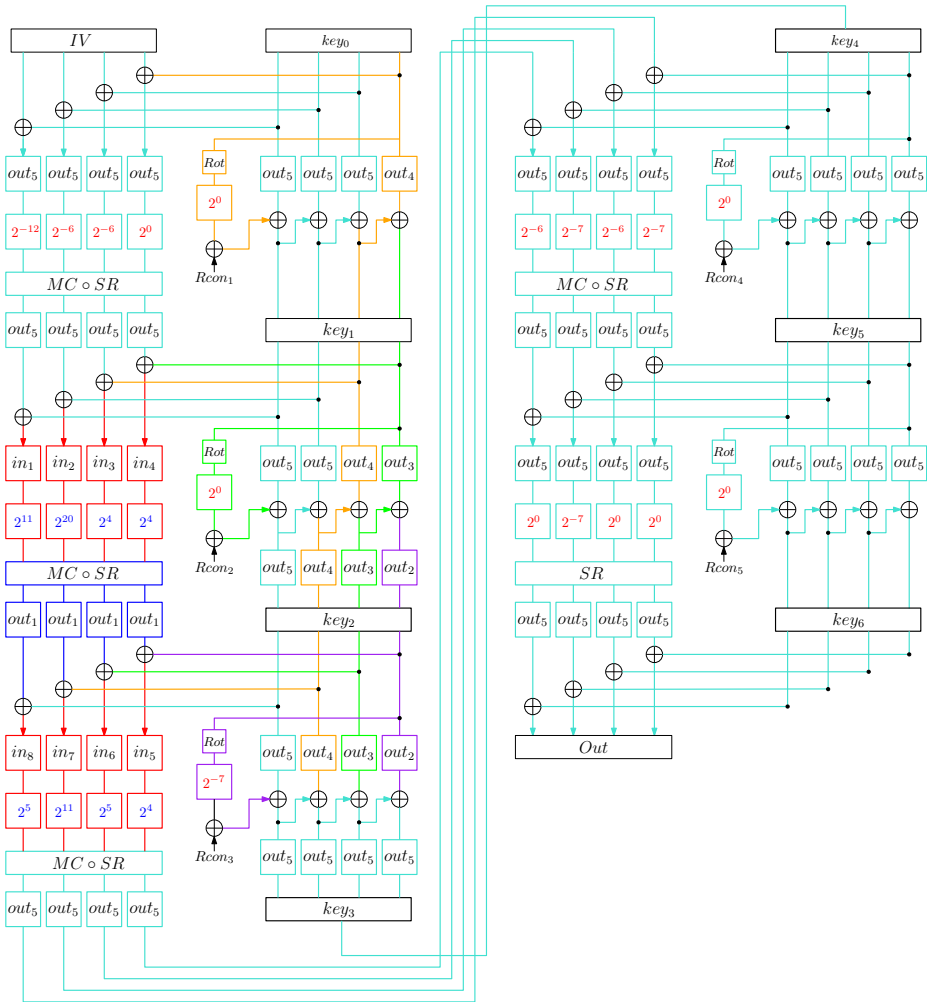


Fig. 19: Free-target-plaintext collision attack on 5-round AES-128.

C.3 Fixed-Target-Plaintext Key Collision on 5-round AES-192

Fig. 20 illustrates an underlying differential characteristic for a fixed-target-plaintext key collision for 5-round AES-192 with a probability of 2^{-186}. Assuming that the 1st round and part of the 2nd round in the data processing part are an inbound phase (i.e., $\{in_1, \ldots, in_6\} = \{x_1[0], \ldots, x_1[3], x_2[0], x_2[1]\}$), and the remaining part, including the key scheduling part, is an outbound phase, the probability of inbound and outbound phases are 2^{-125} and 2^{-61}, respectively. After applying our tool, we can obtain the DoF tree as shown in Fig. 21 for collision attacks. By using this tree, we can construct a fixed-target-plaintext key collision on 5-round AES-192.

Degree of Freedom in the Inbound Phase. In our attacks, we exploit the degrees of freedom of each inbound vertex as follows: $IN_1 = 2^3$, $IN_2 = 2^3$, $IN_3 = 2^3$, $IN_4 = 2^2$, $IN_5 = 2^{25}$, and $IN_6 = 2^{25}$.

By using our automatic tool, we have obtained the differential probability of each inbound vertex as follows: $P(in_1) = 2^{-28}$, $P(in_2) = 2^{-28}$, $P(in_3) = 2^{-27}$, $P(in_4) = 2^{-28}$, $P(in_5) = 2^{-7}$, and $P(in_6) = 2^{-7}$. Based on these results, the maximum degrees of freedom in each vertex are estimated as $IN_{M_1} = 2^{4(=32-28)}$, $IN_{M_2} = 2^{4(=32-28)}$, $IN_{M_3} = 2^{5(=32-27)}$, $IN_{M_4} = 2^{4(=32-28)}$, $IN_{M_5} = 2^{25(=32-7)}$, and $IN_{M_6} = 2^{25(=32-7)}$. Thus, the degrees of freedom of each inbound vertex, which is required for the attack, are sufficiently available.

Attack Procedures.

1. Start with $2^{11(=3+3+3+2)}$ sets of $\{in_1, \ldots, in_4\} = \{x_1[0], \ldots, x_1[3]\}$ and the fixed initial value sets of $\{w_0[0], \ldots, w_0[3]\}$ (red part in Fig. 22); then, obtain 2^{11} sets of $\{w_1[0], \ldots, w_1[3], k_0[0], \ldots, k_0[3]\}$ (blue, purple, green, and orange parts in Fig. 22).
2. Prepare 2^{25} values of $in_5 = x_2[0]$ (red part in Fig. 22); then, obtain $2^{36(=11+25)}$ values of $k_1[0]$ (turquoise part in Fig. 22).
3. Prepare 2^{25} values of $in_6 = x_2[1]$ (red part in Fig. 22); then, obtain $2^{61(=36+25)}$ values of the remaining data processing and key scheduling parts (yellow parts in Fig. 22). As the differential probability of the remaining parts is 2^{-61}, there exists $1 = 2^{0(=61-61)}$ value that fulfills the differential characteristic (yellow parts in Fig. 22).

Attack Complexity. Following the above attack procedures, it is obvious that the attack complexity for Step 3 is dominant, which requires approximately 2^{61} computations of the partial AES-192 encryption. Therefore, total complexity is bounded by 2^{61} computations of 5-round AES-192.

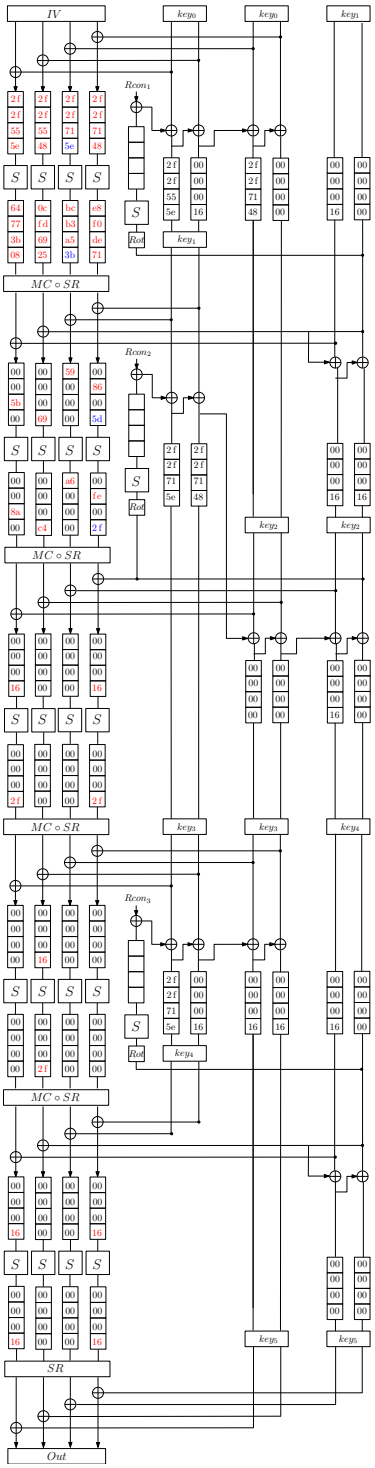


Fig. 20: Differential characteristic for a fixed-target-plaintext collision attack on 5-round AES-192.

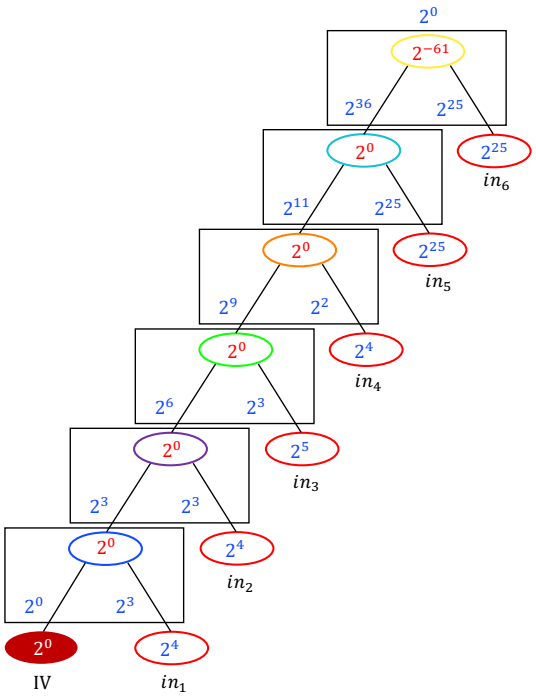


Fig. 21: DoF tree for a fixed-target-plaintext collision attack on 5-round AES-192.

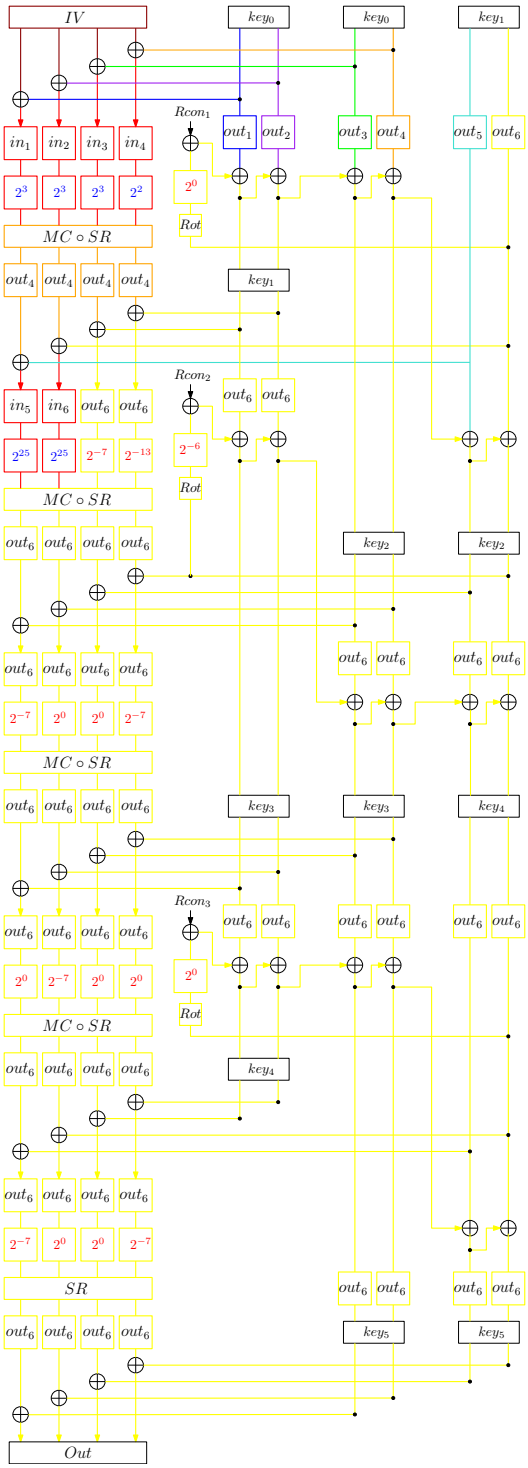


Fig. 22: Fixed-target-plaintext collision attack on 5-round AES-192.

C.4 Free-Target-Plaintext Key Collision on 7-round AES-192.

Fig. 23 illustrates an underlying differential characteristic for a free-target-plaintext key collision for 7-round AES-192 with a probability of 2^{-248}. Assuming that the 3rd and 4th rounds and part of the 5th round in the data processing part are an inbound phase (i.e., $\{in_1, \ldots, in_{10}\} = \{x_3[0], \ldots, x_3[3], x_4[0], \ldots, x_4[3], x_5[0], x_5[1]\}$), and the remaining part, including the key scheduling part, is an outbound phase, the probability of inbound and outbound phases are 2^{-186} and 2^{-62}, respectively. After applying our tool, we can obtain the DoF tree as shown in Fig. 24 for collision attacks. By using this tree, we can construct a free-target-plaintext key collision on 7-round AES-192.

Degree of Freedom in the Inbound Phase. In our attacks, we exploit the degrees of freedom of each inbound vertex as follows: $IN_1 = 2^0$, $IN_2 = 2^0$, $IN_3 = 2^0$, $IN_4 = 2^0$, $IN_5 = 2^0$, $IN_6 = 2^2$, $IN_7 = 2^5$, $IN_8 = 2^4$, $IN_9 = 2^{26}$ and $IN_{10} = 2^{25}$.

By using our automatic tool, we have obtained the differential probability of each inbound vertex as follows: $P(in_1) = 2^{-6}$, $P(in_2) = 2^{-7}$, $P(in_3) = 2^{-28}$, $P(in_4) = 2^{-21}$, $P(in_5) = 2^{-28}$, $P(in_6) = 2^{-28}$, $P(in_7) = 2^{-27}$, $P(in_8) = 2^{-28}$, $P(in_9) = 2^{-6}$, and $P(in_{10}) = 2^{-7}$. Based on these results, the maximum degrees of freedom in each vertex are estimated as $IN_{M_1} = 2^{26(=32-6)}$, $IN_{M_2} = 2^{25(=32-7)}$, $IN_{M_3} = 2^{4(=32-28)}$, $IN_{M_4} = 2^{11(=32-21)}$, $IN_{M_5} = 2^{4(=32-28)}$, $IN_{M_6} = 2^{4(=32-28)}$, $IN_{M_7} = 2^{5(=32-27)}$, $IN_{M_8} = 2^{4(=32-28)}$, $IN_{M_9} = 2^{26(=32-6)}$ and $IN_{M_10} = 2^{25(=32-7)}$. Thus, the degrees of freedom of each inbound vertex, which is required for the attack, are sufficiently available.

Attack Procedures.

1. Start with $2^{0(=0+0+0+0)}$ sets of $\{in_1, \ldots, in_4\} = \{x_3[0], \ldots, x_3[3]\}$ (red part in Fig. 25); then, obtain 2^0 sets of $\{w_3[0], \ldots, w_3[3]\}$ (blue part in Fig. 25).
2. Prepare $2^{11(=0+2+5+4)}$ sets of $\{in_5, \ldots, in_8\} = \{x_4[0], \ldots, x_4[3]\}$ (red part in Fig. 25); then, obtain $2^{11(=0+11)}$ sets of $\{w_4[0], \ldots, w_4[3], k_3[0], \ldots, k_3[3]\}$ (purple, green, orange, and turquoise parts in Fig. 25).
3. Prepare 2^{26} values of $in_9 = x_5[0]$ (red part in Fig. 25); then, obtain $2^{37(=11+26)}$ values of $k_4[0]$ (yellow part in Fig. 25).
4. Prepare 2^{25} values of $in_{10} = x_5[1]$ (red part in Fig. 25); then, obtain $2^{62(=37+25)}$ values of the remaining data processing and key scheduling parts (brown parts in Fig. 25). As the differential probability of the remaining parts is 2^{-62}, there exists $1 = 2^{0(=62-62)}$ value that fulfills the differential characteristic (brown parts in Fig. 25).

Attack Complexity. Following the above attack procedures, it is obvious that the attack complexity for Step 4 is dominant, which requires approximately 2^{62} computations of the partial AES-192 encryption. Therefore, total complexity is bounded by 2^{62} computations of 7-round AES-192.

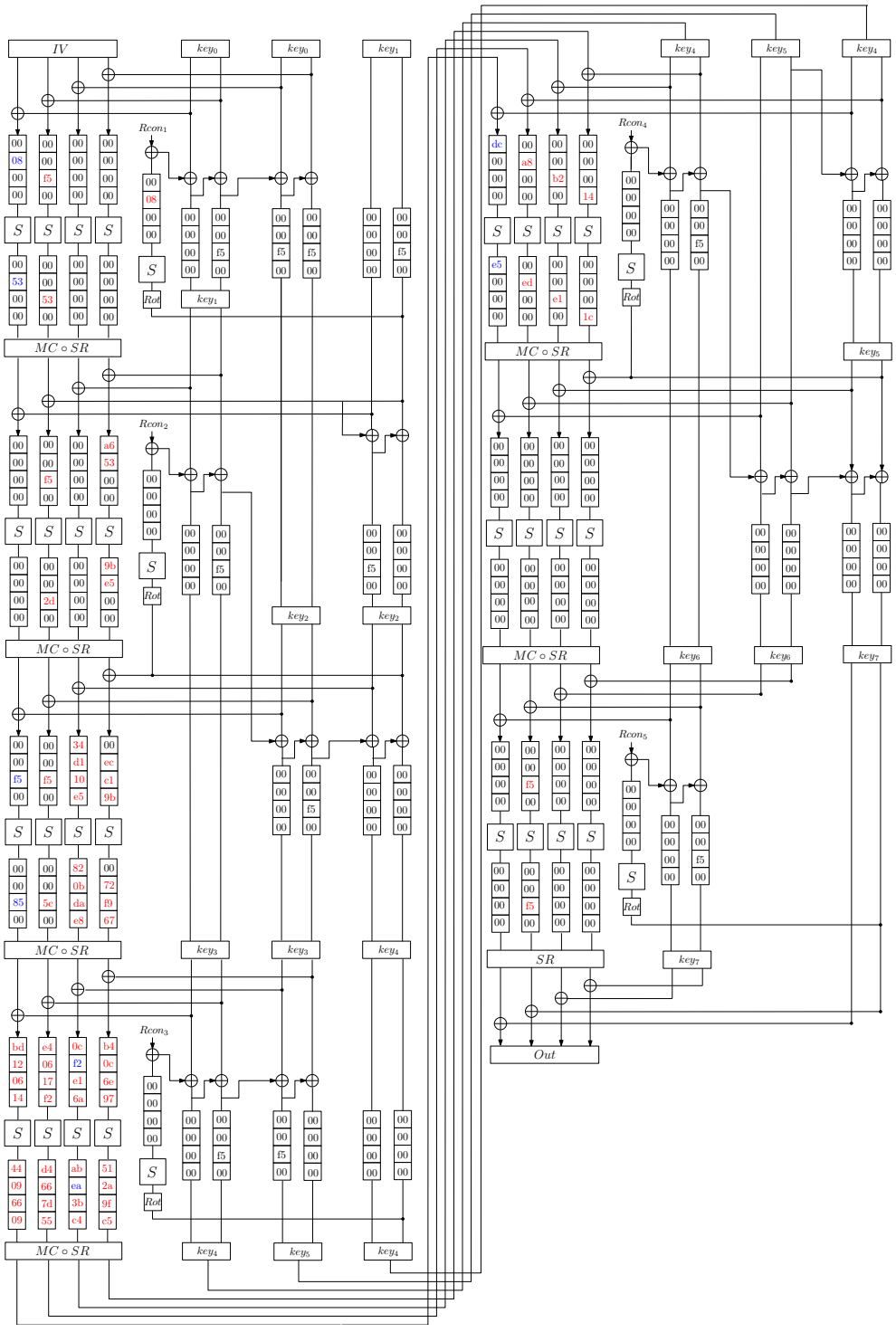


Fig. 23: Differential characteristic for a free-target-plaintext collision attack on 7-round AES-192.

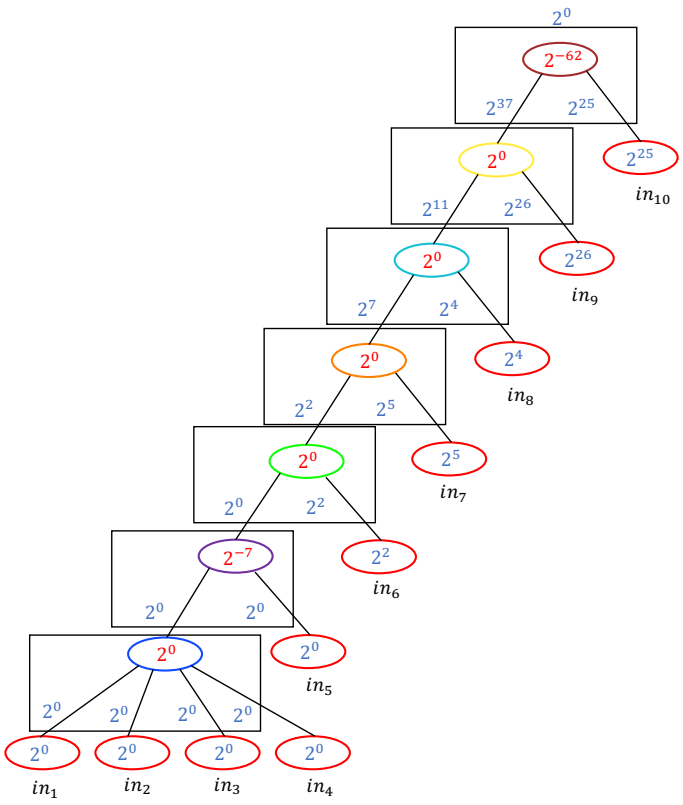


Fig. 24: DoF tree for a free-target-plaintext collision attack on 7-round AES-192.

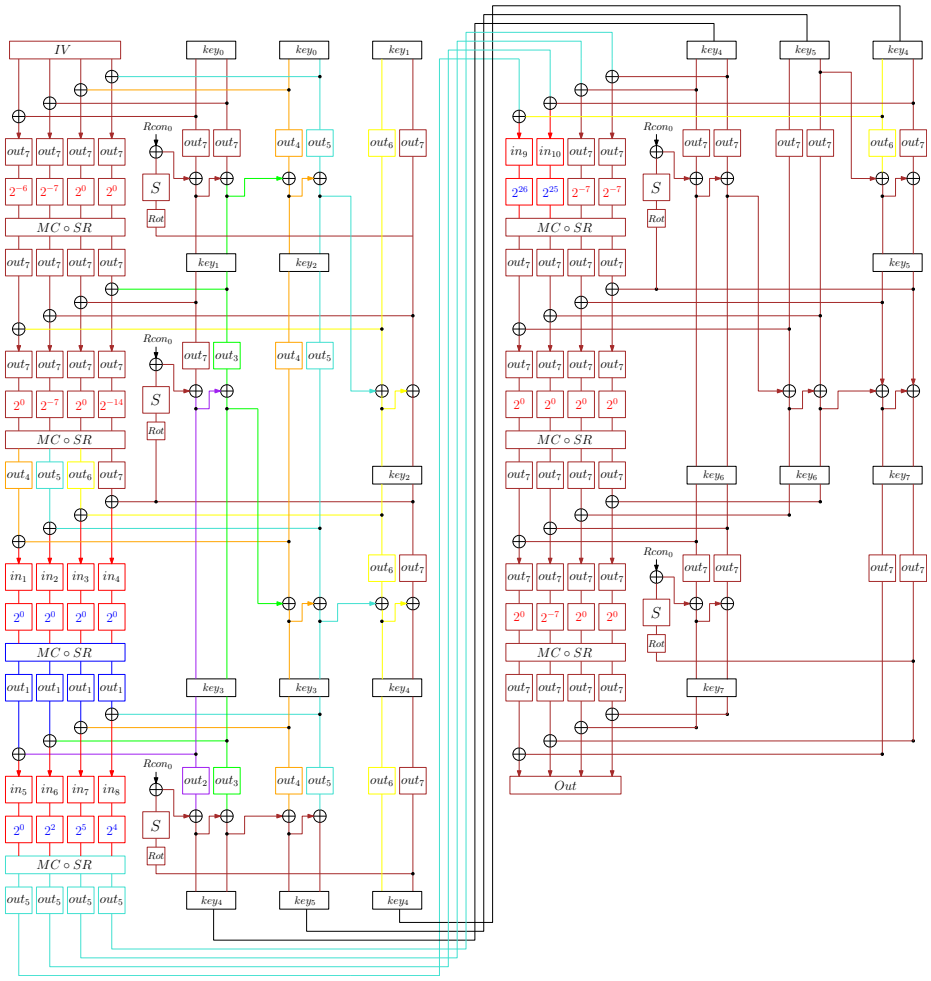


Fig. 25: Free-target-plaintext collision attack on 7-round AES-192.

D Supplemental Materials for Fixed-target-plaintext Collision Attack on 6-round AES-256

This section provides supplemental materials for the fixed-target-plaintext collision attack on 6-round AES-256, explained in Sect. 5.1. Fig. 26 illustrates an underlying differential characteristic for the fixed-target-plaintext key collision for 6-round AES-256 with a probability of 2^{-179}. Fig. 27 shows the DoF tree for the attack, which has been obtained by our automatic tool.

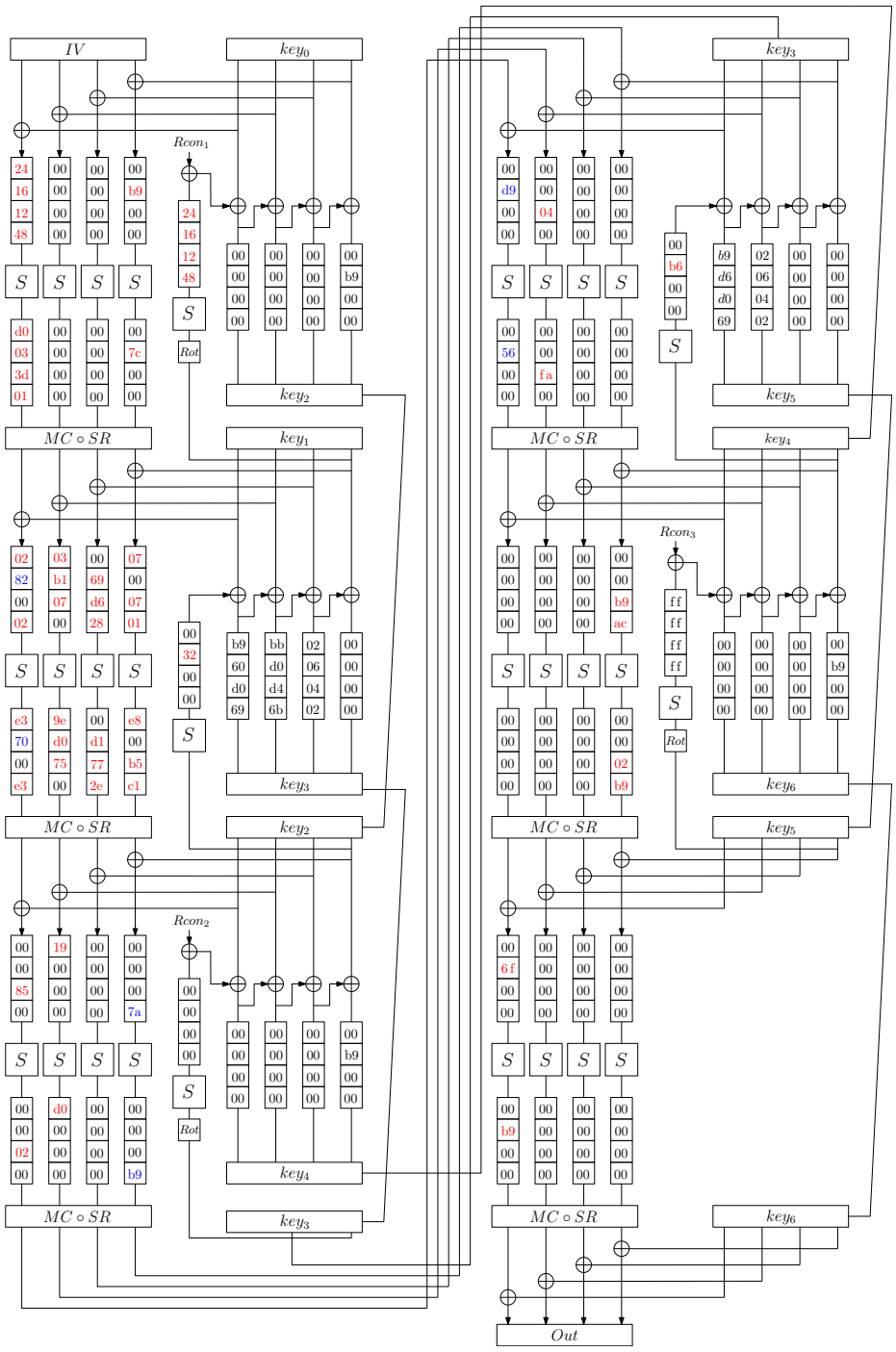


Fig. 26: Differential characteristic for a fixed-target-plaintext collision attack on 6-round AES-256.

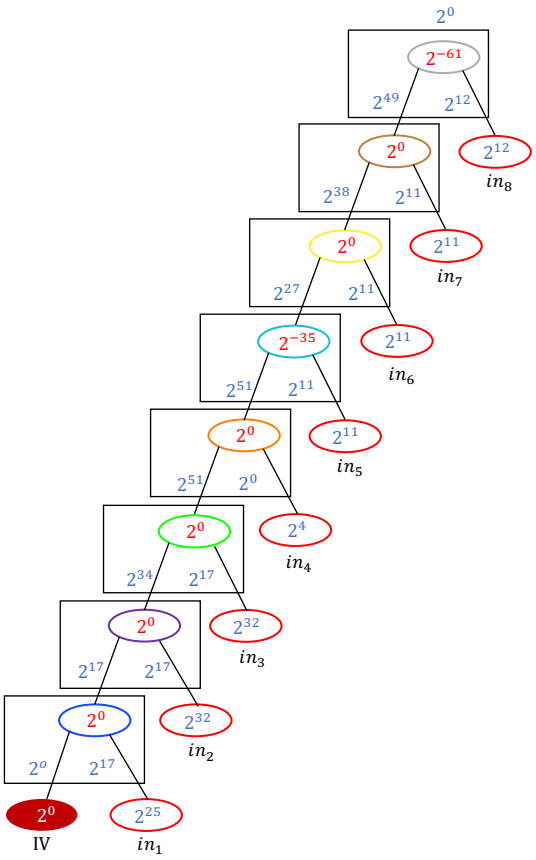


Fig. 27: DoF tree for a fixed-target-plaintext collision attack on 6-round AES-256.

E Supplemental Materials for Free-target-plaintext Collision Attack on 9-round AES-256

This section provides supplemental materials for the free-target-plaintext collision attack on 9-round AES-256, explained in Sect. 5.2. Fig. 28 illustrates an underlying differential characteristic for a free-target-plaintext key collision for 8-round AES-256 with a probability of 2^{-193}. Fig. 29 shows the DoF tree for the attack, which has been obtained by our automatic tool. Moreover, Table 4 shows an example trail of a free-target-plaintext key collision for 9-round AES-256, provided in Sect. 5.2.

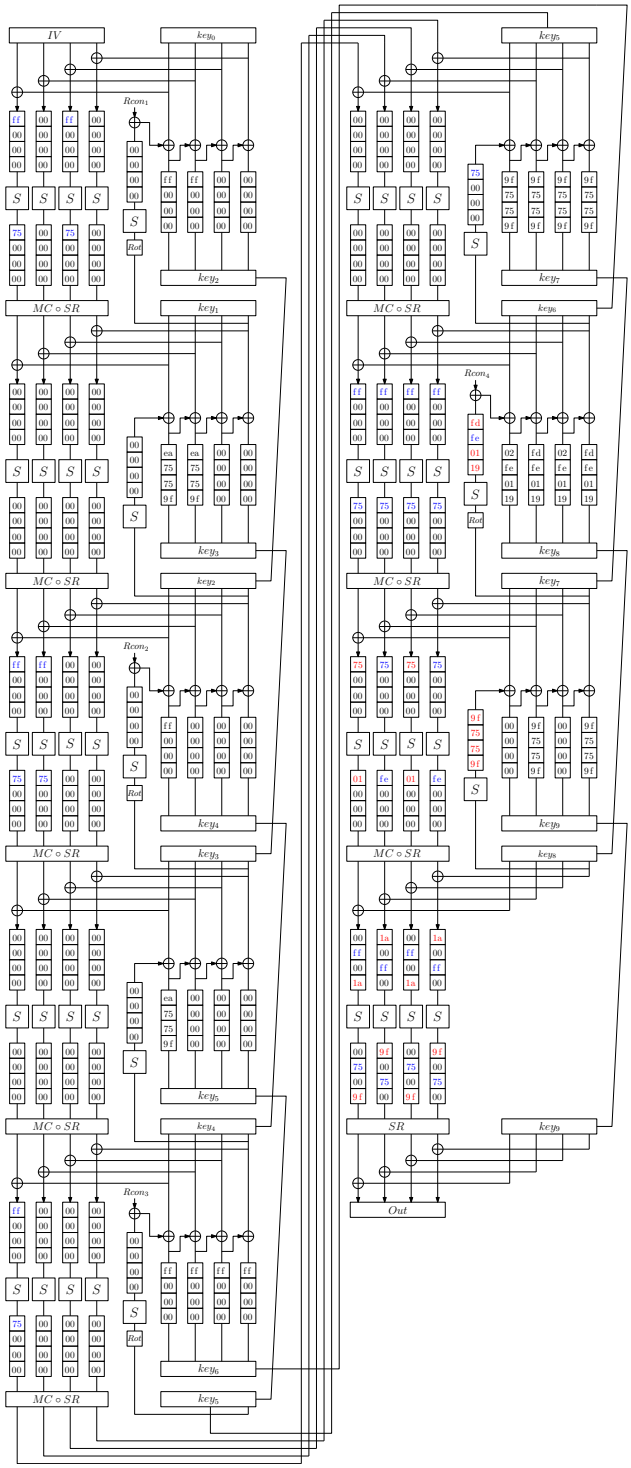


Fig. 28: Differential characteristic for a free-target-plaintext collision attack on 9-round AES-256.

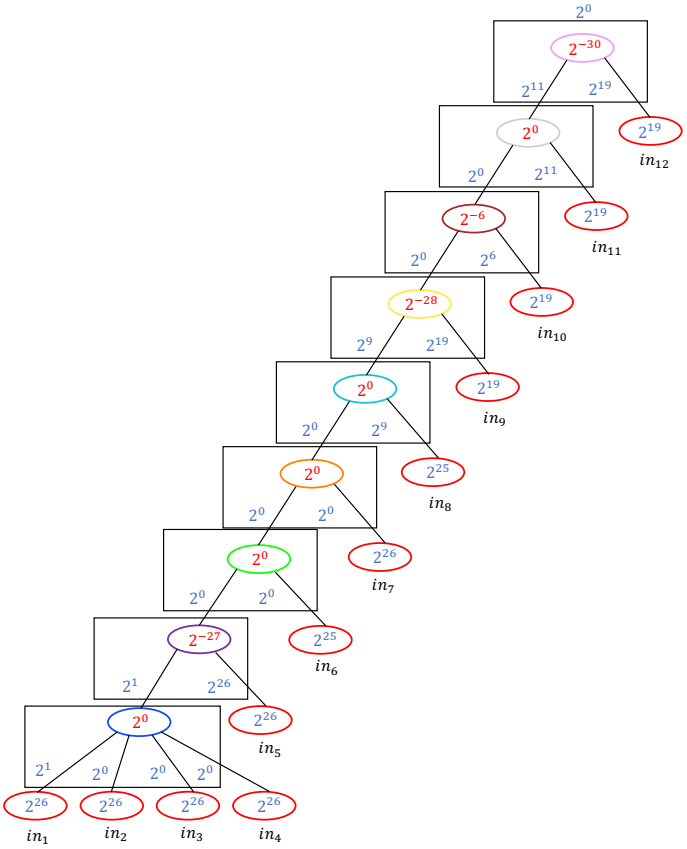


Fig. 29: DoF tree for a free-target-plaintext collision attack on 9-round AES-256.

Table 4: An example trail of a free-target-plaintext key collision for 9-round AES-256.

i	$Plaintext_1$	Key_1	i	$Plaintext_2$	Key_2
0	83 66 63 dc b1 bc 61 82 30 38 ab f7 14 c3 d4 6a	ca 45 20 ea 26 11 ac 9c 30 3c c2 06 7c 39 55 e2 7e 2f d9 46 84 1f b2 3e 96 2a 82 ef 21 00 57 6c	0	83 66 63 dc b1 bc 61 82 30 38 ab f7 14 c3 d4 6a	35 45 20 ea 26 11 ac 9c cf 3c c2 06 7c 39 55 e2 94 5a ac d9 84 1f b2 3e 7c 5f f7 70 21 00 57 6c

i	After SB	After MC	SubKey	i	After SB	After MC	SubKey
1	3b 26 1a 05 88 95 bd 72 63 f2 f9 a1 45 2d 0c c4	ef de 10 b2 0f 66 6d 77 d9 65 ec 76 fc 74 fa 0d	7e 2f d9 46 84 1f b2 3e 96 2a 82 ef 21 00 57 6c	1	4e 26 1a 05 88 95 bd 72 16 f2 f9 a1 45 2d 0c c4	05 ab 65 2d 0f 66 6d 77 33 10 99 e9 fc 74 fa 0d	94 5a ac d9 84 1f b2 3e 7c 5f f7 70 21 00 57 6c
2	81 a1 dd bf 3d b6 9e 3b 84 84 9f ee c1 92 95 ef	a8 a3 38 74 c7 35 52 33 58 fc fa ae 11 cf 6e a0	a8 1e 70 17 8e 0f dc 8b be 33 1e 8d c2 0a 4b 6f	2	81 a1 dd bf 3d b6 9e 3b 84 84 9f ee c1 92 95 ef	a8 a3 38 74 c7 35 52 33 58 fc fa ae 11 cf 6e a0	57 1e 70 17 71 0f dc 8b be 33 1e 8d c2 0a 4b 6f
3	63 7a 52 fb 3b 80 19 6c 8e 8a 69 26 66 a6 3f 8a	be 49 b4 43 37 8e d9 15 c8 43 38 a5 7d 9f 44 85	5b 48 6a ee df 57 d8 d0 49 7d 5a 3f 68 7d 0d 53	3	16 7a 52 fb 4e 80 19 6c 8e 8a 69 26 66 a6 3f 8a	54 3c c1 dc dd fb ac 8a c8 43 38 a5 7d 9f 44 85	b1 3d 1f 71 35 22 ad 4f 49 7d 5a 3f 68 7d 0d 53
4	d9 7c 1d 95 9b 35 7c a6 0c b2 aa b8 59 98 3b f6	aa a0 a2 18 4e 3c fb 0e 10 a6 5f c6 f2 9d 0e 80	55 c9 9d 52 db c6 41 d9 65 f5 5f 54 a7 ff 14 3b	4	d9 7c 1d 95 9b 35 7c a6 0c b2 aa b8 59 98 3b f6	aa a0 a2 18 4e 3c fb 0e 10 a6 5f c6 f2 9d 0e 80	aa c9 9d 52 db c6 41 d9 65 f5 5f 54 a7 ff 14 3b
5	16 f9 75 d6 2a 2d f4 0e 9d ed 63 4f fc aa a2 ea	d2 03 d8 bb 0c c0 f9 86 bf 43 cf 7f 48 5d 27 8c	07 5e 90 0c d8 09 48 dc 91 74 12 e3 f9 09 1f b0	5	63 f9 75 d6 2a 2d f4 0e 9d ed 63 4f fc aa a2 ea	38 76 ad 24 0c c0 f9 86 bf 43 cf 7f 48 5d 27 8c	ed 2b e5 93 d8 09 48 dc 91 74 12 e3 f9 09 1f b0
6	03 4c 52 a9 48 dd c8 be 31 9a c1 de c8 20 07 eb	50 11 61 d4 8b c7 3c 0c ee 39 6c 46 49 cd 76 60	50 09 7a cb 8b cf 3b 12 ee 3a 64 46 49 c5 70 7d	6	03 4c 52 a9 48 dd c8 be 31 9a c1 de c8 20 07 eb	50 11 61 d4 8b c7 3c 0c ee 39 6c 46 49 cd 76 60	af 09 7a cb 74 cf 3b 12 11 3a 64 46 b6 c5 70 7d
7	63 ad af c0 63 30 c5 72 63 7b 30 63 63 30 6f a4	02 f7 c4 f6 e4 e4 9d 2a 4b 9b 80 de 8c 15 fa 0b	3c f8 c1 f3 e4 f1 89 2f 75 85 9b cc 8c 8c 84 7c	7	16 ad af c0 16 30 c5 72 16 7b 30 63 16 30 6f a4	e8 82 b1 69 0e 91 e8 b5 a1 ee f5 41 66 60 8f 94	a3 8d b4 6c 7b 84 fc b0 ea f0 ee 53 13 f9 f1 e3
8	b2 76 6b 6b 63 59 fa 6b b2 72 af c9 63 ee f3 f5	ce 1f aa ca c8 e2 51 f2 56 a3 37 9e 6f 53 ba a0	3c 56 6a af b7 99 51 bd 59 a3 35 fb 10 66 45 86	8	b3 76 6b 6b 9d 59 fa 6b b3 72 af c9 9d ee f3 f5	cc 1e ab c9 2f 1c af eb 54 a2 36 9d 88 ad 44 b9	3e a8 6b b6 4a 67 50 a4 5b 5d 34 e2 ed 98 44 9f
9	89 3b ba 4d d2 21 63 84 76 63 77 4d d2 96 16 f7	- - - - - - - - - - - - - - - -	f6 cb af b7 12 3a 26 98 67 bf bd 54 eb 33 39 28	9	89 4e ba d2 4d 21 16 84 76 16 77 d2 4d 96 63 f7	- - - - - - - - - - - - - - - -	f6 cb af b7 8d 4f 53 07 67 bf bd 54 74 46 4c b7

$Ciphertext_1$	$Ciphertext2$
7f ea d8 40 c0 59 30 d5 11 29 07 d0 39 08 5a 65	7f ea d8 40 c0 59 30 d5 11 29 07 d0 39 08 5a 65

F Two-Block Collision Attacks on 3-round AES-128-DM

Our automatic tool enables us to develop a 3-round free-differential-start collision in the second block with a time complexity of 2^{60} as shown in Fig. 30.

Notations. Unlike the notations in Sect. 5 and Appendix C, the internal states of AES are treated here as a byte-wise array. Then, we denote the i-th round of the state array at the m-th row from the left and the n-th column from the top by $x_i[4m+n]$ for $m, n \in \{0, 1, 2, 3\}$. For example, $x_i[6]$ is represented as the i-th round state array at the 1st row from the left and the 2nd column from the top.

How to Find N Distinct Inputs. We can easily obtain such attacks by exploiting the differential characteristics in Fig. 31 (see Fig. 32 for more details). It is well known that given a fixed input and output difference of Δx and Δy, the probability of $(\Delta y = Sbox(\Delta x))$ is approximately $1/2$ where $Sbox()$ is an operation of S-box of AES [37].

This property indicates that there are about 128 distinct differences of $\Delta x_0[4]$, which result in $\Delta y_0[4] = 0\text{x}65$ through S-box. It means that $\Delta h_1[4] (= \Delta x_0[4] \oplus \Delta k_0[4])$ also has 128 possible values, which lead to differential characteristics of y_0.

$\Delta h_1[4]$ is forwarded to the output, and $\Delta y_2[4]$ is computed as $\Delta y_2[4] = \Delta k_3[4] \oplus \Delta h_1[4]$. Once $\Delta h_1[4]$ is chosen out of 128 candidates, the corresponding $\Delta y_2[4]$ is determined. The probability that $(\Delta y_2[4] = Sbox(\Delta x_2[4] = 0\text{x}42))$ is $1/2$. Thus, there exists $128/2 = 64$ possible $\Delta h_1[4]$, which follow the characteristics for collision with time complexity of 2^{60}. As $\Delta h_1[5]$, $h_1[9]$ and $\Delta h_1[13]$ also have 64 possible candidates, respectively, with the same reason, in total, there are $N = (64)^4 = 2^{24}$ distinct inputs with the same time complexity.

Attack Complexity. For $N = 2^{24}$ and $T_2 = 2^{60}$, the time complexity for 2-block collision attack is estimated as

$$2^{52} \text{ one block comp.} + 2^{60} \text{ one block comp.} < 2^{60} \text{ two block comp.}$$

The memory requirements of 2^{52} in the first block.

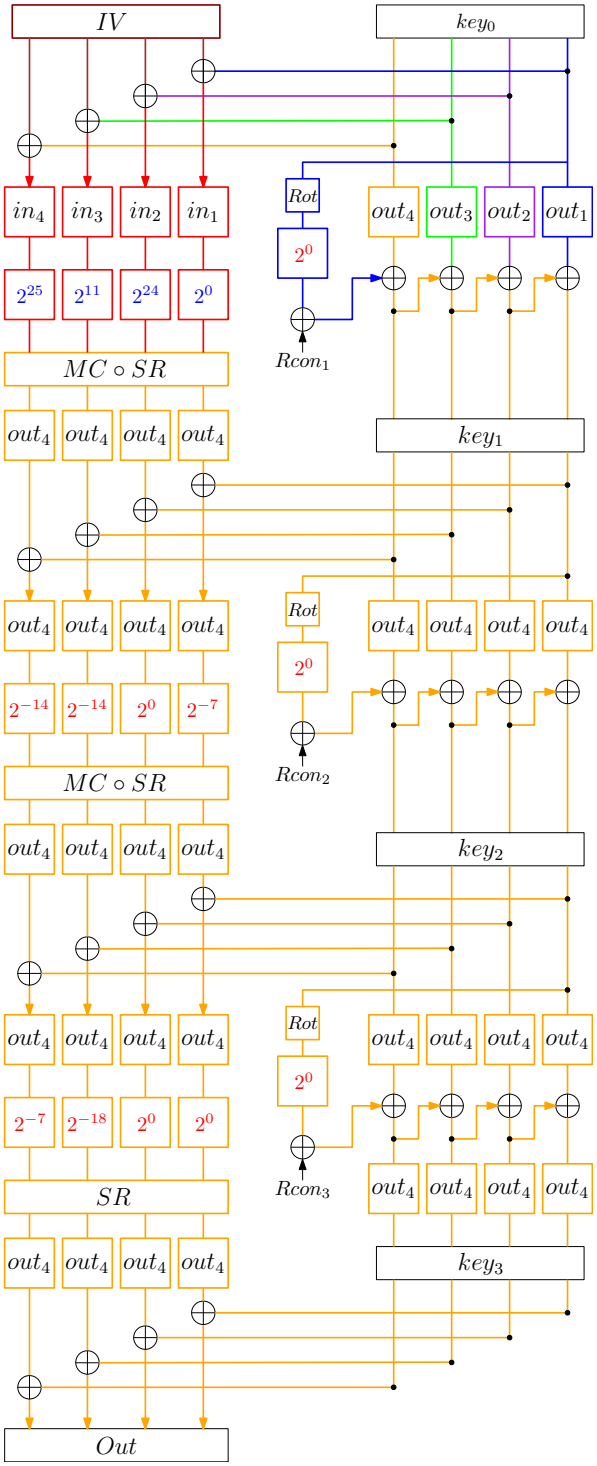


Fig. 30: Free-differential-start collision on 3-round AES-128-DM.

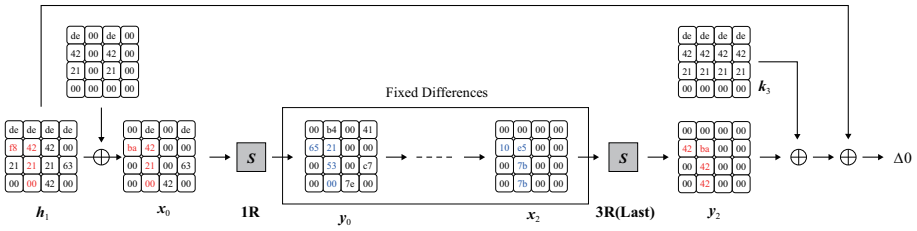


Fig. 31: Differential characteristic for a free-differential-start collision on 3-round AES-128-DM.

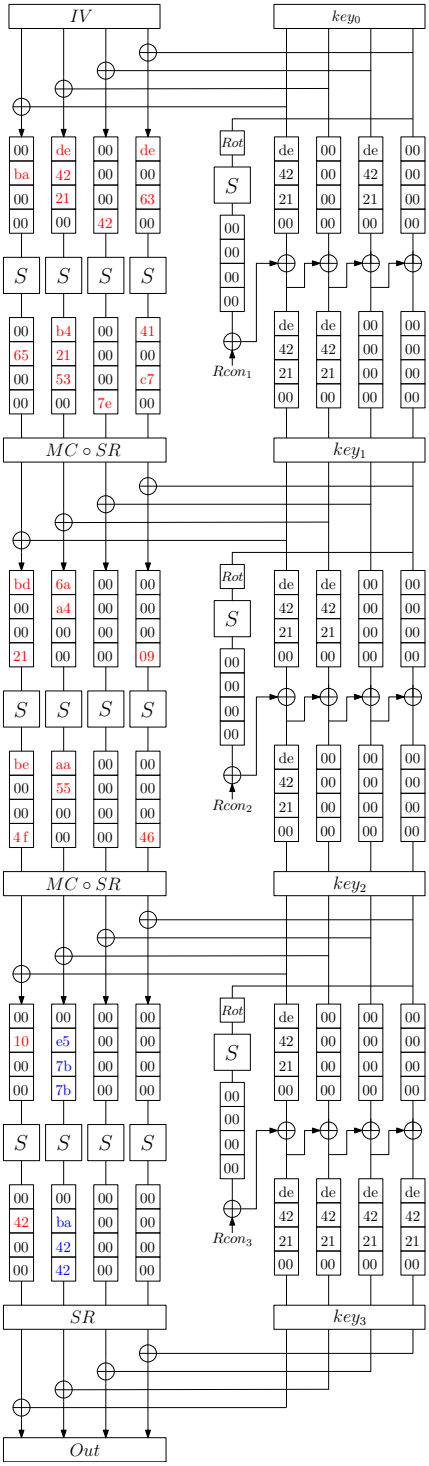


Fig. 32: Details on a differential characteristic for a free-differential-start collision on 3-round AES-128-DM.

G Supplemental Materials for Two-Block Collision Attacks on 9-round AES-256-DM

This section provides supplemental materials for the two-block collision attack on 9-round AES-256, explained in Sect. 6.3. Fig. 33 illustrates an underlying differential characteristic for the two-block collision for 9-round AES-256 with a probability of 2^{-58}.

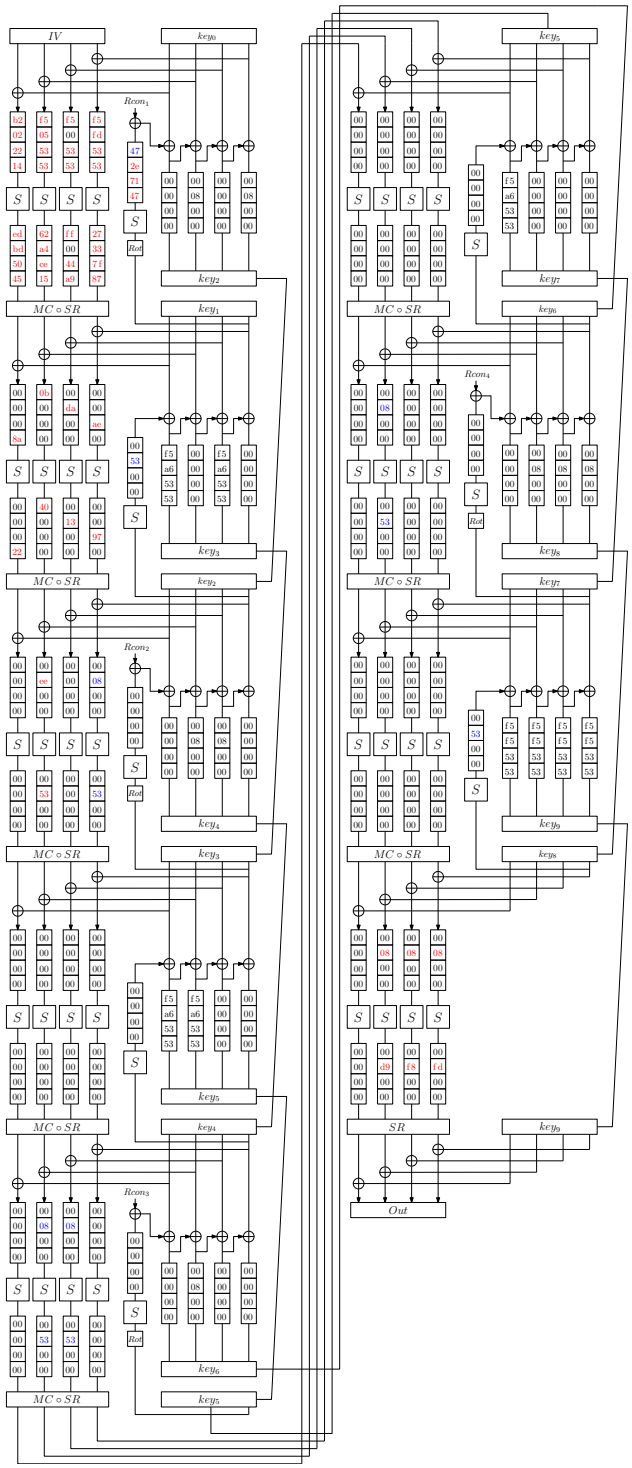


Fig. 33: Differential characteristic for a free-differential-start collision on 9-round AES-256-DM.