Scrutinizing the Security of AES-based Hashing and One-way Functions

Shiyao Chen[3], Jian Guo[3], Eik List[3], Danping Shi[1,2(✉)], and Tianyu Zhang[3(✉)]

[1] State Key Laboratory of Cyberspace Security Defense, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
shidanping@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3] Nanyang Technological University, Singapore
{shiyao.chen,guojian}@ntu.edu.sg, tianyu005@e.ntu.edu.sg, elist@posteo.de

Abstract. AES has cemented its position as the primary symmetric-key primitive for a wide range of cryptographic applications, which motivates the analysis on the concrete security of AES in practical instantiations, for instance, the collision resistance of AES-based hashing, the key commitment security of AES-based authenticated encryption schemes, and the one-wayness of AES-based one-way functions in MPC/ZK protocols. In this work, we further advance the meet-in-the-middle (MITM) attack framework on AES-like constructions. We introduce single-color initial structure (SCIS), which leverages new structural insights to reduce the complexity of neutral word generation, a critical bottleneck in MITM attacks. As a result, we yield a series of improved results on AES over the state-of-the-art, including the first classical one-block collision attack on 7-round AES-MMO/MP, marking the first round advancement in over a decade and matching the best attack round in the quantum setting, as well as the first one-block collision attack on 4-round AES-128-DM, bridging the gap highlighted by Taiyama et al. at Asiacrypt 2024 from a non-differential-based approach. Additionally, we provide a comprehensive list of new results on the security margins of AES-192, AES-256, Rijndael-192, and Rijndael-256 in multiple attack settings.

Keywords: Meet-in-the-Middle Attack · Preimage · Collision · Key Collision · Key Recovery · AES · Rijndael · Hashing · One-way Function · FAEST

1 Introduction

1.1 The Meet-In-The-Middle Attack

The meet-in-the-middle (MITM) attack is one of the most fundamental cryptanalysis technique. Its origins trace back to Diffie and Hellman's seminal attack on Double-DES [32], where they introduced a method to partition the primitive and search space (*e.g.*, key and message space) into two separate parts,

each utilized in a distinct stage of the primitive. Bogdanov and Rechberger later formalized three-subset MITM attacks [20] that described a variant of existing meet-in-the-middle attacks on block ciphers Sasaki and Aoki formally introduced the MITM attacks against hash functions in [68], which is later honored with the prestigious IACR Test-of-Time Award[4]. This direction grew rapidly with many technical innovations, such as splice-and-cut [5], initial structures [6,69], indirect and partial matching [4], probabilistic initial structures [44], bicliques [53] as a formalization of initial structures, and precomputations in the middle [24]. At FSE 2011, Sasaki pioneered the first meet-in-the-middle attacks for preimage attacks on round-reduced AES hashing [66]. Bao *et al.* [8] later improved Sasaki's result by also taking the degrees of freedom from the key space into account.

At Eurocrypt 2020, Bao *et al.* proposed an automatic search framework for MITM preimage attacks on AES-like hash functions [9], which initiated a series of works on automated MITM attacks: Dong *et al.* extended the framework to key-recovery and collision attacks [38]. Bao *et al.* proposed the novel superposition structure with bi-directional costs, and introduced guess-and-determine into the framework [10]. Hua *et al.* then combined guess-and-determine with nonlinearly constrained neutral words in their search for preimage attacks [47]. Schrottenloher and Stevens also explored a simpler modelling on permutations [72] and on block ciphers with lightweight key schedules [73]. At the same time, the automated MITM attack framework was extended to sponge constructions [61,40] and Feistel networks [46]. Recently, Chen *et al.* [27] enriched the MITM framework with linearization, distributed initial structures, and structural similarity.

1.2 Practical Instantiations of AES

AES-based Hashing. A hash function maps inputs with arbitrary lengths to fixed-length hash values and is one of the most fundamental cryptographic primitives. A secure cryptographic hash function must satisfy three fundamental security properties: preimage resistance, second-preimage resistance, and collision resistance. Aside from dedicated hash functions, for instance SHA-2, SHA-3, Ascon, to make use of the common coexistence of cryptographic primitives in resource-constrained systems, a conventional strategy to construct hash function is to instantiate the block cipher with one of the PGV modes [60] as the underlying compression function. In such settings, AES is the primary choice due to its wide implementation, hardware-optimization and long record of withstanding cryptanalysis. AES-128 instantiated with the Matyas-Meyer-Oseas mode is used in the standards of the Zigbee protocol suite [3] and ISO/IEC [51].

AES-based One-way Function Beyond hashing, constructing secure one-way functions (OWFs) based on AES has gained significant attention in the fields of zero-knowledge (ZK) proofs and multi-party computation (MPC). Starting from Ishai *et al.* 's seminal MPC-in-the-head (MPCitH) framework [48], ZK proofs

[4] https://www.iacr.org/testoftime/

could be constructed efficiently (in terms of circuit size) from secure symmetric-key one-way functions. Since then, many different OWFs [2,35] have been proposed and used to improve the efficiency of ZK proofs. Besides MPCitH, Boyle *et al.* [23] enriched the portfolio of ZK proof techniques with vector oblivious linear evaluation (VOLE) correlations. The VOLE technique, particularly the one used in SoftSpokenOT by Roy [63], was further explored in Baum *et al.* 's subsequent work on VOLE-in-the-Head (VOLEitH) [12]. In their framework, they demonstrated how digital signatures could be constructed efficiently using AES or EM-AES. This direction has gained significant attention since the NIST called for further submissions to diversify its lattice-heavy portfolio of quantum-secure digital signature schemes[5].

Though Picnic [64] pioneered the use of arithmetization-oriented primitives such as LowMC [2] in signatures, the security of those primitives remains far from well-understood compared to that of AES. The fact has motivated many recent signature schemes to use AES-based OWF, examples can be seen from MPCitH-based designs, such as BBQ [64], Banquet [14], Limbo [65], Feta [13], Dubhe [33], and other protocols [35,75]; and VOLEitH-based designs, such as FAEST [11], Phecda [34], as well as other zero-knowledge frameworks like Preon [25]. In those schemes, AES is used as a one-way function $F_k(x)$ to encrypt a random-chosen plaintext x under a secret key k with $y = F_k(x)$ such that the proof becomes k for the public known (x, y). This setting restricts the usage of each key to a single encryption only, so that the adversary can observe only 1 or 2 known plaintext/ciphertext pairs. Moreover, the design of FAEST [11] explores the OWF construction of EM-AES, which instantiates fixed-key AES or Rijndael with the single-key Even-Mansour scheme [41].

1.3 Gaps

As an international standard with broad implementations and various optimizations in hardware/software, AES is the go-to choice when an ideal cipher is assumed in instantiations and plays an irreplaceable role in a wide range of applications. Thus, the detailed security evaluation of AES in practical attack settings that aligned with real-world attack scenarios has been a heated topic in the field of cryptanalysis. We hereby identify a few notable open problems and security challenges from the applications of AES, which are listed below. Note that we focus on the results in the classic setting and the one-block attacks (on hash or one-way functions) in this work.

On the Collision Resistance of AES-based Hashing. At FSE 2009, Mendel *et al.* introduced the rebound attack [57], which has become the dominant technique in constructing collision attacks on AES-like hashing. Lamberger *et al.* at Asiacrypt 2009 [54] as well as Gilbert and Peyrin at FSE 2010 [42] proposed the Super-Sbox technique to rebound attacks and obtained one-block collision attack on 6-round AES-MMO/MP, which has hold the record of longest number of round

[5] https://csrc.nist.gov/Projects/pqc-dig-sig/standardization

on AES-MMO/MP attacked ever since. Both [57] and [42] have been recognized with the prestigious FSE Test-of-Time award[6]. At Eurocrypt 2020, Hosoyamada and Sasaki [45] presented a quantum rebound attack to find one-block collision on 7-round AES-MMO/MP. Other works explore two-block collision and variants of collision attacks in classic and quantum settings by leveraging degree of freedom from the key, to list a few: [71,37,26]. In comparison, the best classical collision attack on one-block AES-MMO/MP has stayed at 6 rounds for more than a decade. We can't help but wonder:

> *Open Question 1: Are there any classical one-block collision attacks on AES-MMO/MP that can attack more than* 6 *rounds?*

On the Key-commitment Security of AES. For block ciphers, key collision attacks find distinct keys that generate the same ciphertext, indicating weaknesses of the primitive [7,19,62,52]. Beyond that, key collision security has a relevant role in the security of higher-level modes. A series of works [1,36,43,55] demonstrated key-replacement attacks on widely deployed authenticated encryption schemes that implicitly relied on key collision resistance although conventional AE security does not provide any such guarantees. The community was quick to react and study the key-commitment security of existing schemes, and to propose countermeasures and more secure schemes [15,28,29] and MACs [16]. However, many proofs of modes idealize the primitive, which may leave a gap to concrete security. Albertini *et al.* [1] outlined that while the fix for AES-GCM could be proven secure with an ideal cipher, with a concrete cipher, the security would reduce to finding a key collision more efficiently than the generic bound in the target-plaintext setting, where the plaintext is fixed to some specific value. Chen *et al.* [29] faced similar problems when analyzing HCTR2.

Recently at Asiacrypt 2024, Taiyama *et al.* [74] initiated the first key collision attacks on round-reduced AES. They proposed a well-furnished automatic tool that exploits bit-wise differential characteristics to perform rebound-type attacks. However, in [74], the fixed-target-plaintext key collision attack on AES-128 (equivalent to one-block collision on AES-128-DM) was limited to 2 rounds. The authors discussed the optimality of this result in [74, Appendix B], quote, "*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.*" This has motivated us to investigate:

> *Open Question 2: Can we find a new way to mount collision attacks on AES-128-DM that extends more than 3 rounds?*

On the Security of AES-based OWFs in ZK/MPC. The application of AES-based OWFs in zero-knowledge (ZK) and multi-party computation (MPC) protocols has necessitated the research on their security in practical attack settings, particularly on their resistance to known-plaintext attacks with only 1 or 2 known plaintext/ciphertext pairs. Until now, the best single-known-plaintext

[6] https://tosc.iacr.org/index.php/ToSC/ToT_Award

(SKP) key recovery attack on AES is retained by the series of work by Bouillaguet *et al.* [22,21] from more than a decade ago. At Crypto 2011 [22], a SKP key recovery attack on 5-round AES-128 was found while no dedicated SKP key recovery attack has been proposed for AES-192 and AES-256. For EM-AES, quoting Baum *et al.* [11], *there are no known shortcut attacks on EM-AES except variants of brute-force key search, also no shortcut attacks on variants with reduced rounds.* Hence, we are interested in both improving and delivering a comprehensive cryptanalytic result on EM-AES and AES in the restricted attack setting of ZK and MPC protocols. This objective is well captured by the original words by Baum *et al.* :

> *Open Question 3: How the EM and non-EM one-way functions compare for AES in terms of security margin? [11, Section 10.3.5]*

1.4 Our Contributions

In this work, we address the three aforementioned open questions using meet-in-the-middle (MITM) attacks. A critical requirement for constructing collision attacks with MITM is the existence of an algorithm to generate neutral words with amortized computational cost $O(1)$ and overall time and memory complexity below the birthday bound. To date, no MITM attacks on AES have been proposed matching this requirement. To this end, we introduce single-color initial structure (SCIS), a technique exploring new structural properties of MITM attacks to enable efficient neutral word generation. The incorporation of SCIS into MITM attacks has led to new collision attacks extending the attack rounds compared to state-of-the-art and improved preimage and key recovery attacks, which cannot be achieved by re-evaluating the existing techniques in literature.

In addition, we survey instantiations of AES, revisit their corresponding attack settings, and formulate the conversion from single-known-plaintext (SKP) key recovery attacks to fixed-target-plaintext key collision attacks (equivalent to one-block collision attack in DM mode). While both SCIS and the proposed conversion are fairly generic and can be applied to other targets, we choose to focus ourselves on AES and Rijndael, and have attained the following results (summarized in Table 1):

1. We find the first one-block collision attack on 7-round AES-MMO/MP in the classical setting, which extends one more round upon the state-of-the-art given by rebound attacks with the Super-Sbox technique [54,42,39] in over a decade. Notably, our result matches the best attack rounds of quantum one-block collision attacks on AES-MMO/MP [45].
2. We find the first one-block collision attack on 4-round AES-128-DM in the classical setting. By employing MITM attacks combined with the SCIS technique and our proposed conversion, we bridges the gap highlighted in [74] that no feasible collision attack on more than 3 rounds of AES-128-DM under the estimation of single differential characteristics from a non-differential-based approach.

3. We identified a SKP key recovery attack on 5-round AES with the same time complexity but reducing memory complexity from 2^{96} to 2^{40} compared with Bouillaguet, Derbez, and Fouque [22] at Crypto 2011. In the same attack setting, we provide new attacks on 7-round AES-192 and 8-round AES-256. We also present improved one-block preimage attack on AES-MMO/MP, which is equivalent to SKP key recovery attack on EM-AES. These results can be extended to single-use AES and EM-AES (defined in Section 2.2), which have wide applications in signature schemes adopting AES-based OWFs, *e.g.*, BBQ, Banquet, FAEST.
4. In view of NIST's recent interest in standardizing wider variant of AES[7], we provide a comprehensive list of new results on Rijndael-192 and Rijndael-256 in all mentioned attack settings.

These results offer new insights into AES-like structures, provide a comprehensive security analysis of various AES instantiations, and demonstrate that both AES and its wide variants remain sufficiently secure against the considered attacks.

1.5 Organization

The remainder of this work is structured as follows. In Section 2, we provide preliminaries on AES, practical instantiations of AES, and MITM attacks. We then describe the MITM conversions among collision, key collision, preimage and key-recovery attacks in Section 3. In Section 4, we introduce the concept of SCIS and the SCIS MITM attack frameworks. Thereupon, we illustrate the attack details of classical collision attack on 7-round AES-MMO/MP, classical collision attack on 4-round AES-128-DM, and improved SKP key recovery attack on 5-round AES-128 in Section 5, Section 6, and Section 7.

2 Preliminaries

2.1 AES and Rijndael

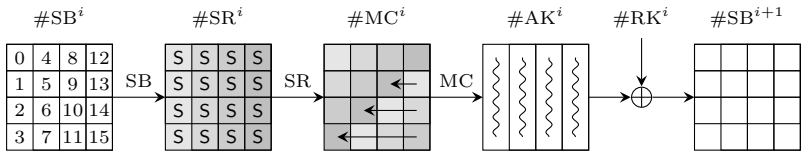


Fig. 1. The round function of AES.

[7] https://csrc.nist.gov/news/2024/nist-proposes-to-standardize-wider-variant-of-aes

Table 1. Summary of results on AES/Rijndael-based hashing and one-way functions. The listed attacks are *classic one-block* attacks unless otherwise specified.

Collision attacks on AES-MMO/MP **and** Rijndael-MMO/MP					
Target	Rounds	Time	Memory	Technique	Reference
AES-128	5/10	2^{56}	2^{4}	Rebound	[57]
	6/10	2^{56}	2^{32}	Super S-box	[54,42]
	6/10	2^{48}	2^{32}	Super S-box	[39]
	7/10	$\mathbf{2^{60}}$	$\mathbf{2^{60}}$	SCIS MITM††	Section 5
Rijndael-192	8/12	2^{92}	2^{92}	SCIS MITM††	Appendix A.1
Rijndael-256	9/14	2^{124}	2^{124}	SCIS MITM††	Appendix A.2
Collision attacks on AES-DM **and** Rijndael-DM**					
Target	Rounds	Time	Memory	Technique	Reference
AES-128	2/10	2^{49}	Negl.	Rebound	[74]
	2/10	**practical**	2^{22}	Rebound	[58]
	3/10‡	2^{60}	2^{52}	Rebound	[74]
	4/10	$\mathbf{2^{60}}$	$\mathbf{2^{60}}$	SCIS MITM††	Section 6
AES-192	5/12	2^{61}	Negl.	Rebound	[74]
	5/12	**practical**	2^{5}	Rebound	[58]
AES-256	6/14	2^{61}	Negl.	Rebound	[74]
	9/14‡	2^{58}	2^{55}	Rebound	[74]
Rijndael-192	4/12	2^{84}	2^{84}	SCIS MITM††	Appendix B.1
Rijndael-256	5/14	2^{124}	2^{124}	SCIS MITM††	Appendix B.2
Single-known-plaintext key recovery attacks on AES **and** Rijndael					
Target	Rounds	Time	Memory	Technique	Reference
AES-128	4*/10	2^{120}	2^{80}	Algebraic*	[22]
	4*/10	$\mathbf{2^{112}}$	$\mathbf{2^{48}}$	MITM	Appendix C.1
	5/10	2^{120}	2^{96}	Algebraic	[22,31]
	5/10	2^{120}	$\mathbf{2^{40}}$	SCIS MITM	Section 7
AES-192	7/12	2^{120} ($2^{184\dagger}$)	2^{72}	MITM	Appendix C.2
AES-256	8/14	2^{120} ($2^{248\dagger}$)	2^{72}	MITM	Appendix C.3
Rijndael-192	6/12	2^{184}	2^{32}	MITM	Appendix C.4
Rijndael-256	7/14	2^{248}	2^{96}	MITM	Appendix C.5
Preimage attacks on AES-MMO/MP **and** Rijndael-MMO/MP**					
Target	Rounds	Time	Memory	Technique	Reference
AES-128	7/10	2^{120}	2^{8}	MITM	[66]
	7/10	$\mathbf{2^{112}}$	2^{32}	MITM	Appendix D.1
Rijndael-192	8/12	2^{176}	2^{16}	MITM	Appendix D.2
Rijndael-256	9/14	2^{248}	2^{16}	MITM	[9]

†† SCIS is the critical technique for the proposed MITM collision attacks.
‡ The attacks find *two-block collisions*.
* Last round with MC operation *i.e.*, $r^{\star}$ denotes r number of full one-round of AES.
* Algebraic Meet-in-the-Middle and Guess-and-Determine automatic search method by Bouillaguet, Derbez, and Fouque [22] at Crypto 2011.
† The complexity refers to key recovery attack on single-use AES-192 and AES-256 in digital signature schemes, as discussed in Section 1.2.
** Equivalent to fixed-target-plaintext key collision on AES and Rijndael.
** Equivalent to single-known-plaintext key recovery on EM-AES and EM-Rijndael

Rijndael is a family of block ciphers designed by Daemen and Rijmen, with block size and key size each to be specified as 128, 192, or 256. In 2001, the NIST selected a subset of Rijndael [30] with a block size of 128 as the Advanced Encryption Standard (AES) [59]. Following the original notations, the intermediate states are organized in a rectangular array with 4 rows, and `Nb` or `Nk` are used to denote the number of columns of a encryption state or a key schedule state. We define the key-block ratio as $\gamma = \texttt{Nk}/\texttt{Nb}$. Hereinafter, we use Rijndael-192 and Rijndael-256 to abbreviate Rijndael variants with $\texttt{Nb} = \texttt{Nk} = 6$ and $\texttt{Nb} = \texttt{Nk} = 8$. The round function of Rijndael is depicted in Figure 1 and illustrated as follows:

- `SubBytes(SB)`: A non-linear byte-wise substitution with an S-box.
- `ShiftRows(SR)`: A cyclic left shift on the i-th row by r_i bytes. When $\texttt{Nb} < 8$, $(r_0, r_1, r_2, r_3) = (0, 1, 2, 3)$; when $\texttt{Nb} = 8$, $(r_0, r_1, r_2, r_3) = (0, 1, 3, 4)$.
- `MixColumns(MC)`: A column-wise left multiplication of a 4×4 constant matrix with Property 1.
- `AddRoundKey(AK)`: A bitwise XOR of the round key $\#\text{RK}^i$ to the state.

Property 1. Knowing any of the $t \geq 4$ bytes out of the input column and the output column of an `MC` operation is sufficient to compute the rest $(8-t)$ bytes.

The key schedule is performed as follows: let w and k be the round keys and the master key each expressed as an array of words. For the j-th column, when $j < \texttt{Nk}$, we have $w[j] = k[j]$; when $j \geq \texttt{Nk}$, we have:

$$w[j] = \begin{cases} w[j-\texttt{Nk}] \oplus \texttt{Rot}(\texttt{SB}(w[j-1])) \oplus c[j/\texttt{Nk}] & j \bmod \texttt{Nk} \equiv 0 \text{ and } \texttt{Nk} < 8, \\ w[j-\texttt{Nk}] \oplus \texttt{SB}(w[j-1]) & j \bmod \texttt{Nk} \equiv 4 \text{ and } \texttt{Nk} = 8, \\ w[j-\texttt{Nk}] \oplus w[j-1] & \text{otherwise,} \end{cases}$$

where `SB` applies the S-box to each byte of the word, and `Rot` rotates the word by one byte, and c represents the array of round constants.

2.2 AES-based Hashing and One-Way Functions

To cater different needs, AES is often instantiated with different provable secure modes under different settings. Here we summarize a few important applications.

AES-based Hashing. Constructing hash functions based on secure block ciphers is a practical approach. The block cipher is first converted into a compression function CF by the instantiation of a PGV mode [60]. The hash function is then defined as an iteration of CF: $\mathcal{H}(M) = h_n$, where $M = m_0||\ldots||m_n$ and $h_{i+1} = \mathsf{CF}(h_i, m_i)$ for $i \in [0, n)$. We refer h_0 as the initial value IV, and h_i with $i > 0$ as chaining values. A selection of PGV modes with formula expressions is illustrated in Figure 2.

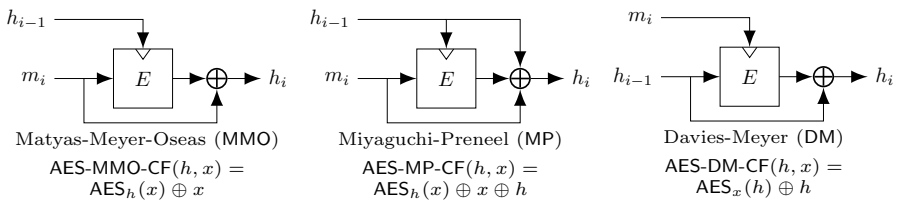


Fig. 2. A selection of PGV modes and corresponding CF formulation

AES-based One-Way Functions. Recent developments in ZK and MPC have demonstrated interests in building schemes using one-way functions that are instantiated from symmetric-key primitives. In this use-case, for a secure OWF $F_k(x)$, its relation $\mathcal{R}$ is defined as $((x, y), k) \in \mathcal{R} \iff F_k(x) = y$, where (x, y) is the input/output pair and k is the secret to hide. Thus, other than the security of the construction and of the protocol, the security relies on the one-wayness of the underlying OWF. As can be seen from proposed schemes in recent years, there are often two practical instantiations of $F_k(x)$ based on AES and its precursor Rijndael to specifies OWF variants for NIST security levels L1, L3 and L5:

The first approach is to use $\beta = \lceil \gamma \rceil$ (γ is defined above as the key-block ratio) blocks of AES as OWF:

- $F_k(x) \coloneqq \mathsf{AES\text{-}128}_k(x)$ for 128-bit security ($\beta = 1$ block);
- $F_k(x) \coloneqq \mathsf{AES\text{-}192}_k(x_0) \| \mathsf{AES\text{-}192}_k(x_1)$ for 192-bit security ($\beta = 2$ blocks);
- $F_k(x) \coloneqq \mathsf{AES\text{-}256}_k(x_0) \| \mathsf{AES\text{-}256}_k(x_1)$ for 256-bit security ($\beta = 2$ blocks).

Hereinafter, we refer this type of OWF construction as single-use AES or Rijndael.

As proposed in [11], the second approach EM-AES leverages the single-key Even-Mansour construction [41]: $\mathsf{SEM}^{\pi}_k(z) \coloneqq \pi(z + k) + k$, where π is realized with: $\pi(z) = \mathsf{AES}_x(z)$. In this setup, the AES key x is fixed and public.

- $F_k(x) \coloneqq \mathsf{AES\text{-}128}_x(k) + k$ for 128-bit security;
- $F_k(x) \coloneqq \mathsf{Rijndael\text{-}192}_x(k) + k$ for 192-bit security;
- $F_k(x) \coloneqq \mathsf{Rijndael\text{-}256}_x(k) + k$ for 256-bit security.

For more details, we refer the readers to the specification of FAEST [11], where these two instantiations have been well summarized and specified.

2.3 The Meet-in-the-Middle Attack Framework and Its Extensions

We provide a high-level overview of MITM attacks in Figure 3 and a list of common notations in all our attack descriptions in Table 2. An MITM attack divides the computations into two independently computable chunks, the forward chunk and the backward chunk. The forward (resp. backward) *neutral word* represents a value that complies with the predefined constants, or satisfies the *constant constraints*, during its propagation. The predefined constants are conventionally refered as *costs of DoF*, or simply *costs*, as they reduce the number of eligible neutral words. The two chunks meet at a matching operation M. The properties

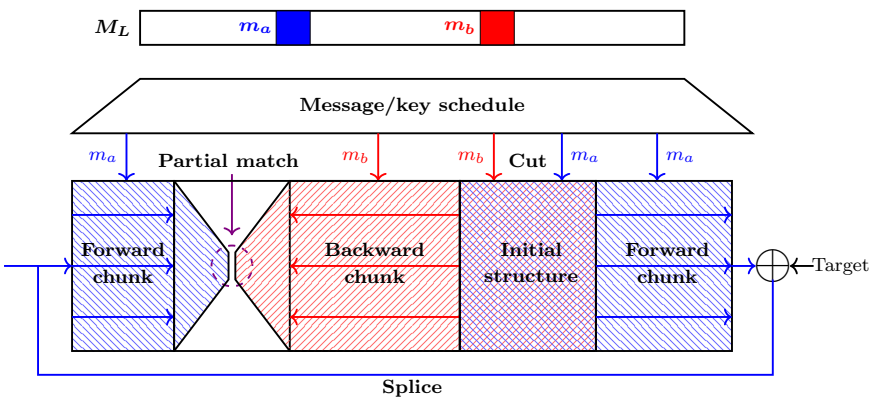


Fig. 3. A high-level overview of MITM attacks [68].

Table 2. Common notations.

DoF	Degree(s) of freedom.
$S^{\texttt{ENC}}/S^{\texttt{KSA}}$	Starting state of encryption/key schedule.
V^+/V^-	The forward/backward neutral word space, which is the set of neutral words that comply with the predefined constants.
$d_{\mathcal{B}}/d_{\mathcal{R}}$	DoF of the forward/backward neutral words, we have $d_{\mathcal{B}} = \log_2 \lvert V^+ \rvert$ and $d_{\mathcal{R}} = \log_2 \lvert V^- \rvert$.
E^+/E^-	The ending state of forward/backward chunk.
m^+/m^-	The forward/backward values at E^+/E^- for matching
M	The matching operation between E^+ and E^-.
$d_{\mathcal{M}}$	The degrees of matching.
T^+/T^-	Lookup tables constructed at E^+/E^-

of M are exploited to match the forward and backward neutral words, which are called *partial-match constraints*. The matched pair of neutral words are referred as a *matched candidate* to check for a full match. Without loss of generality, we assume $d_{\mathcal{B}} \leq d_{\mathcal{R}}$, and an MITM (pseudo-)preimage attack procedure is described as below:

1. Assign arbitrary values to the constants in pre-defined constraints.
2. Compute V^+ and V^- based on the constants.
3. For all $v^+ \in V^+$, compute to E^+ and get m^+, $T^+[m^+] \leftarrow v^+$.
4. For all $v^- \in V^-$, compute to E^- and get m^-, for all $v^+ \in T^+[m^-]$, get matched candidate (v^+, v^-).
5. If (v^+, v^-) is a preimage, return (v^+, v^-). Otherwise, revert to step 1.

The computational complexity of the above attack is evaluated as follows:

$$2^{n-(d_{\mathcal{B}}+d_{\mathcal{R}})} \cdot \left(2^{d_{\mathcal{B}}} + 2^{d_{\mathcal{R}}} + 2^{d_{\mathcal{B}}+d_{\mathcal{R}}-d_{\mathcal{M}}}\right) \simeq 2^{n-\min(d_{\mathcal{B}},d_{\mathcal{R}},d_{\mathcal{M}})}. \quad (1)$$

Converting Preimage Attack into Key Recovery Attack. A variant of the basic MITM attack, known as three-subset MITM attack, was originally pro-

posed by Bogdanov and Rechberger [20] to mount key recovery attacks in two stages: MITM stage and Key Testing stage. In the MITM stage, a block cipher E_K is divided into three chunks as $E_K(P) = H_{K_3\|K_2}(G_{K_1\|K_2\|K_3}(F_{K_1\|K_2}(P))) = C$, where the key space is partitioned into three independent subspaces $K = K_1\|K_2\|K_3$. Then, by using MITM technique, one can independently compute H and F to perform an m-bit partial-match inside G under a fixed K_2, thus reducing the search space from $2^{|K_1|+|K_3|}$ to $2^{|K_1|+|K_3|-m}$ with a time complexity of $2^{|K_1|} + 2^{|K_3|}$; In the Key Testing stage, the surviving key candidates from the MITM stage will be tested using some plaintext/ciphertext pairs, *e.g.*, $\beta = 1$ or 2 AES blocks in above mentioned OWF derived from AES block ciphers directly. Finally, to identify the correct key, the overall time complexity can be evaluated as $2^{|K_2|}(2^{|K_1|} + 2^{|K_3|} + 2^{|K_1|+|K_3|-m})$, where the complexity of the Key Testing stage usually will be negligible with respect to the MITM stage. This method has since been applied to many target ciphers [50,70,49,17,67,38,10].

Converting Preimage Attack into Collision Attack. At FSE 2012, Li, Isobe and Shibutani [56] found that an MITM partial target preimage attack with a t-bit partial match at the last round and a time complexity of 2^l, can be generically converted into a collision attack with a time complexity of $2^{l+(n-t)/2}$. To achieve this, one just need to repeat the partial target preimage attack for $2^{(n-t)/2}$ times with the same t-bit partial target. As the $2^{(n-t)/2}$ candidates share the same t bits hash value, a collision is expected for the rest $(n-t)$-bit state due to the birthday paradox. This conversion then has been applied to Merkle-Damgård hash functions [38,10] and further developed by Dong *et al.* [40] to sponge constructions.

Graphical Representation of MITM Attacks. To visualize MITM attack trails, the cells ■ and ■ are conventionally [66,8,9] used to represent forward and backward neutral bytes, respectively. The linear combination of ■ and ■ is represented by ■, *i.e.*, the superposition state introduced in [10], while ■ represents a constant byte and □ represent an unknown byte.

3 Conversions among Collision, Key Collision, one-block preimage and SKP Key Recovery Attacks

3.1 Revisiting Different Modes and Their Attack Settings

MMO/MP Mode and EM-AES. As summarized in Section 2.2, $\mathsf{EM\text{-}AES}_k(x) = \mathsf{AES\text{-}MMO\text{-}CF}(k, x)$, a single-known-plaintext key recovery attack on EM-AES is equivalent to one-block preimage attack on AES-MMO/MP. In one-block preimage attacks and one-block collision attacks on AES-MMO/MP, the initial value IV is predetermined and fixed. Thus, in these attacks, the adversary can only control *the message* fed to the AES encryption.

DM Mode and Single-use AES. A one-block preimage attack on AES-DM is equivalent to SKP key recovery attacon on AES, and a one-block collision attack on AES-DM is equivalent to fixed-target-plaintext key collision attack on AES. In these attacks, the adversary can only control *the key* fed to the AES encryption.

As introduced in Section 2.2, in single-use AES, the plaintext/ciphertext pair (P, C) will be opened as the public key and fixed. For $\beta > 1$, a SKP key recovery attack on AES can be trivially converted to an attack on β AES blocks by iterating the attack on the first block till one key satisfy all β pairs of (P, C).

3.2 Conversions among Attacks and Implications on MITM Attacks

Previous works have established the generic conversions from preimage attacks to key recovery attacks [20] and to collision attacks [56]. Recently, Taiyama *et al.* [74] highlighted key collision attacks on AES at Asiacrypt 2024. As introduced for *Open Question 2* in Section 1.3, we are motivated to approach key collision attacks from a different angle. In this section, we formulate the generic conversion from SKP key recovery attacks to fixed-target-plaintext key collision attacks, which completes the conversions among collision attacks in MMO/MP mode, fixed-target-plaintext key collision attacks (equivalent to collision attacks in DM mode), SKP key recovery attacks, and one-block preimage attacks as illustrated in Figure 4.

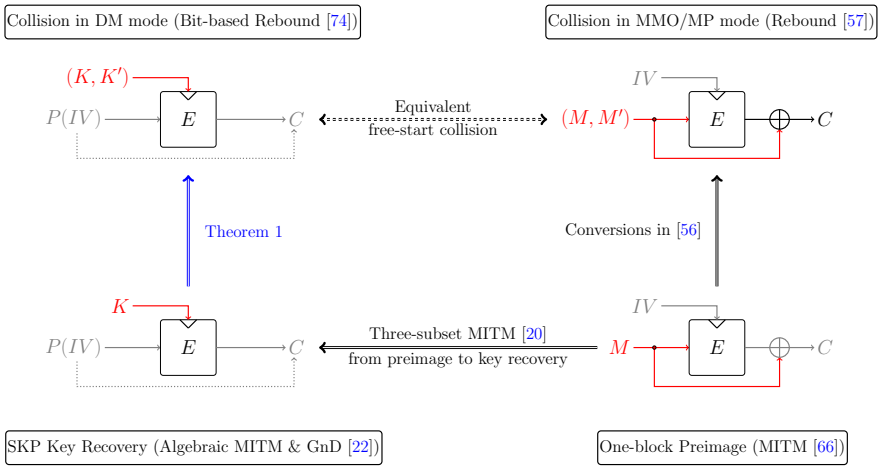


Fig. 4. Conversions among collision attacks in MMO/MP mode, collision attacks in DM mode, SKP key recovery attacks, and one-block preimage attacks

We first define the terminology of a t-bit partial target SKP key recovery attack, followed by the formulation in Theorem 1.

Definition 1 (t-bit partial ciphertext SKP key recovery attack.). *For a block cipher E (with n-bit output length), given a fixed plaintext-ciphertext*

pair (P, C) *and a mask* v *of hamming weight* t*, find a key candidate* k *such that* $(E_k(P) \oplus C) \wedge v = 0_n$*.*

Theorem 1 (Converting t-bit partial ciphertext SKP key recovery attack to fixed-target-plaintext key collision attack). *For a block cipher* E *and given* (P, C)*, if there is an oracle* $\mathcal{A}$ *that can take input of* (v, g) *where* $wt(v) = t$ *and* g *is a constant, and find a candidate* k *such that* $(E_k(P) \oplus C) \wedge v = 0_n$ *with a time complexity of* 2^l*, then there is an algorithm* $\mathcal{B}$ *that is expected to find a fixed-target-plaintext key collision with a complexity of* $2^{l+(n-t)/2}$*.*

Proof. Similar to the conversion from preimage attack to collision attack in [56], $\mathcal{B}$ can be constructed by the following steps:

1. Set t-bit mask v and initialize T to be empty;
2. Call $\mathcal{A}(v, g)$ to obtain values of k;
3. If $T[E_k(P)]$ is empty, $T[E_k(P)] \leftarrow k$, revert to step 2 and call $\mathcal{A}(v, g')$ with an untested g'; Otherwise, output the key collision pair $(k, T[E_k(P)])$.

By iterating the above $2^{(n-t)/2}$ times, a key collision pair is expected to find. □

Necessary Condition for MITM Collision attacks. First, the partial matching of the MITM trail must occur in the last round, such that a candidate that passes the partial match satisfies the partial target (*i.e.*, a distinguished point) [56]. Second, an efficient neutral word generation algorithm must exist for the MITM trail with amortized cost $O(1)$ and overall time and memory complexity below the birthday bound [10]. More details on the second point will be illustrated in Section 4.1.

4 Single-color Initial Structure MITM Attacks

4.1 The Bottleneck of Neutral Word Generation

MITM attacks in literature share a similar structure: the values of the constants are iterated in an outer loop. Within each iteration, the set of blue and red neutral words, values of which must be compatible with the fixed constants, are computed in parallel to the matching point. The term *neutral word generation* refers to the process of obtaining the set of neutral words that satisfies a given set of constants.

At Crypto 2021, Dong *et al.* [38] pointed out that the constraints introduced by fixed constants in MITM attacks can be nonlinear. To address complex cases where these nonlinear constraints are sparsely distributed in many rounds and difficult to solve efficiently, the authors proposed pre-computation for neutral word generation, a table-based technique that enumerates all possible values of the initial space and indexes them according to different constant values before the main procedure. However, pre-computation incurs substantial time complexity that, in most cases, exceeds the birthday bound, thus are incompatible to be used in MITM collision attacks. This limitation was first noted by Bao *et*

al. at Crypto 2022 [10, Remark 4], quote: "*Sometimes, to achieve amortized computational complexity O(1), it is necessary to pre-compute values of neutral bytes for many fixed bytes (enumerated in the outermost loop of the attack) at once. However, for some very complex cases, even aided by automatic tools and considering amortized complexity, it might be difficult to find pre-computation procedures with total complexity lower than the main procedure.*"

As the number of rounds in the attack increases, the complex case is almost inevitable. Hence, there has been no improved collision results on AES proposed with MITM attacks. We have thoroughly re-evaluated and implemented all known techniques from the literature in an effort to improve upon the state-of-the-art collision attacks on AES, but were unsuccessful. This has motivated us to identify and exploit new structures in MITM attacks that can be leveraged to construct more efficient attacks.

4.2 Single-color Initial Structure

In this work, we aim to identify a class of MITM attack trails with a special structure that enables efficient neutral word generation, namely single-color initial structure (SCIS). The core idea is to asymmetrically distribute the constants involved in an MITM attack between the blue and red chunk. As conceptually depicted in Figure 5, without loss of generality, we let the blue chunk be packed with more constraints but contain its propagation to be within a few rounds only. Simultaneously, as we leverage most constraints into the blue chunk, we keep the red chunk relatively free of constants and propagate only red values to the matching point, To the best of our knowledge, such special structure were never systematically exploited in the literature.

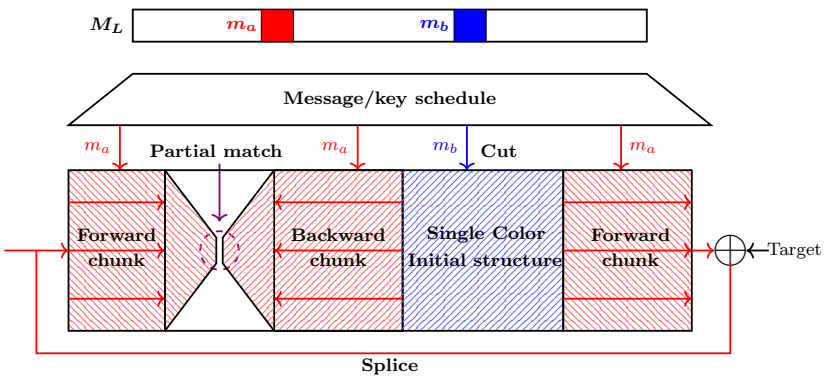


Fig. 5. A high-level overview of single-color initial structure.

The structure alleviates the neutral word generation from several angles. First, the constraints in the blue chunk are concentrated in a (relatively) few

number of rounds. When the nonlinear constraints are densely distributed in a few number of rounds, we can use well-established tools to solve the nonlinear constraints and generate neutral words with amortized cost $O(1)$, for instance the algebraic MITM and guess-and-determine proposed by Bouillaguet, Derbez, and Fouque at Crypto 2011 [22] or the triangulation algorithm proposed by Biryukov, Khovratovich and Nikolic at Crypto 2009 [18]. Ideally, if we are able to solve the nonlinear constraint system of the blue chunk with amortized cost $O(1)$, we are able to generate blue neutral words on-the-fly and bypass pre-computation. Second, as we leverage most constraints into the blue chunk, though the red chunk cover more rounds, its neutral word generation simplifies. The most trivial case is when there are no constraints at all in the red chunk so that the neutral words become an enumeration of the initial word DoFs. Third, the neutral word generation in the blue chunk and the partial matching can be done in parallel, which benefits implementation. In other words, if $v^- \in V^-$ satisfies the partial match, for any $v^+ \in V^+$, (v^-, v^+) is a candidate passes the partial match filter.

4.3 Core SCIS MITM Function

We formalize the integration of the SCIS technique into the MITM attack framework by introducing a core SCIS MITM function f (Algorithm 1) as a general abstraction applicable to preimage, collision, and key recovery attacks.

For a constant in the blue chunk, we heuristically classify it as *enumerating constants* g_{ec} if it is possible to generate neutral words at amortized cost $O(1)$ satisfying with any given value of that constant. The rest constants in the blue chunk are referred as *indexing constants* g_{ic}. In this setting, we can iterate over values of g_{ec} and compute all the blue neutral words complying with g_{ec} on-the-fly, store them in table T_{blue} indexed by the value at g_{ic}. We denote the DoF of g_{ec} as $d_{g_{ec}}$ and g_{ic} as $d_{g_{ic}}$. To optimize memory necessitated in neutral word generation, we should maximize $d_{g_{ec}}$ while minimizing $d_{g_{ic}}$. In other words, we aim to resolve as many constraints as possible. For the solvable part of the constraint system, we first iterate over the constant values, partially solve the system, and then use a hash table to manage the unresolved constraints.

Complexity Analysis of f. In Algorithm 1, Line 1 generates blue neutral words satisfying g_{ec} with amortized cost $O(1)$ and stores in T_{blue}. On average, we expect $2^{d_{\mathcal{B}}}$ entries under each of the $2^{d_{g_{ic}}}$ index in T_{blue}. For all possible values of the indexing constants g_{ic}, Line 4 computes the red neutral word space V^+ to the matching point. The number of iterations of Line 4 is $2^{d_{g_{ic}}+d_{\mathcal{R}}}$. On average, we expect $2^{d_{g_{ic}}+d_{\mathcal{R}}-d_{\mathcal{M}}}$ combinations of (g_{ic}, r) to pass the partial match filter. As the matching is independent from blue values, one call of f generates $2^{d_{g_{ic}}+d_{\mathcal{B}}+d_{\mathcal{R}}-d_{\mathcal{M}}}$ candidates satisfying the partial match, which is also the expected number of iterations of Line 8. The memory complexity of f depends on the size of T_{blue}, which is $2^{d_{\mathcal{B}}+d_{g_{ic}}}$. The computation of one red neutral word to the matching point and the check if the attack is successful can both be done in $O(1)$. We represent the total time complexity of f as T_f, the number of candidates passing the partial match filter as N_f, and the memory complexity of f

Algorithm 1: Core SCIS MITM function $f(g_{ec})$

```
   // Assume we perform the single-color match on red neutral words
1  Generate blue neutral words that comply with g_ec with amortized cost O(1),
     store in T_blue, indexed by its value at g_ic;
2  for g_ic in 2^{d_{g_ic}} values ■ do
3      for r in 2^{d_R} red neutral words ■ do
4          Compute to the matching point;
5          if pass the partial match filter with probability 2^{-d_M} then
6              for b in 2^{d_B} entries of T_blue[g_ic] do
7                  Get candidate (b, r);
8                  Check if a preimage/correct key/collision pair has been found;
```

as M_f. We summarize the complexity analysis of f in Equation (2):

$$\begin{aligned} T_f &= 2^{d_{g_{ic}}} \cdot (2^{d_{\mathcal{B}}} + 2^{d_{\mathcal{R}}} + 2^{d_{\mathcal{B}}+d_{\mathcal{R}}-d_{\mathcal{M}}}) \\ N_f &= 2^{d_{g_{ic}}+d_{\mathcal{B}}+d_{\mathcal{R}}-d_{\mathcal{M}}} \\ M_f &= 2^{d_{g_{ic}}+d_{\mathcal{B}}} \end{aligned} \tag{2}$$

4.4 SCIS MITM Preimage and Key Recovery Attacks

For MITM preimage attacks and key recovery attacks, the checking process (Line 8 of Algorithm 1) is as follows: for all matched (b, r), the attacker recovers the full states of E^+ and E^-, and check if $E^+ = E^-$. As one call of f generates N_f candidates, $2^{n-d_{\mathcal{M}}}/N_f$ iterations of f with different values of g_{ec} expects one preimage, or one correct key. Thus, the total time complexity is:

$$2^{n-d_{\mathcal{M}}}/N_f \cdot T_f \approx 2^{n-\min(d_{\mathcal{B}},d_{\mathcal{R}},d_{\mathcal{M}})}. \tag{3}$$

The memory complexity of a preimage attack of a key recovery attack is M_f.

4.5 SCIS MITM Collision and Key Collision Attacks

For MITM collision attacks, the checking process (Line 8 of Algorithm 1) additionally requires a table T_{dp} initialized to be empty: for all matched (b, r), the attacker recovers the input x and the output h, then lookup in T_{dp} under index h. If the entry is non-empty, a collision pair $(x, T_{dp}[h])$ is found. Otherwise, $T_{dp}[h] \leftarrow x$. One call of f generates N_f candidates that share the same $d_{\mathcal{M}}$ bytes. As we expect one collision pair in $2^{(n-d_{\mathcal{M}})/2}$ candidates, we need to iterate f with $2^{(n-d_{\mathcal{M}})/2}/N_f$ values of g_{ec}. Thus, the total time complexity is:

$$2^{(n-d_{\mathcal{M}})/2}/N_f \cdot T_f \approx 2^{n/2+d_{\mathcal{M}}/2-\min(d_{\mathcal{B}},d_{\mathcal{R}},d_{\mathcal{M}})} \tag{4}$$

A successful attack would require $d_{\mathcal{B}}, d_{\mathcal{R}} > d_{\mathcal{M}}/2$. The memory complexity is dominated by the size of T_{dp}, which is expected to be $2^{(n-d_{\mathcal{M}})/2}$.

Remark 1. The time complexity estimation of an SCIS MITM trail follows the established formula in literature [9,10,38,47] for all mentioned attack settings. However, this estimation relies on the assumption that an efficient neutral word generation is available, which is critical to the validity of the attack. Thus, the core contribution of SCIS lies in identifying new structures that enable efficient neutral word generation, leading to valid and improved attacks.

5 Classical Collision Attack on 7-Round AES-MMO/MP

As shown in Figure 6, we present the first one-block collision attack on 7-round AES-MMO/MP in the classic setting. The round keys are fixed as IV is fixed as constant. For convenience, we denote the constants in the starting point $S^{\texttt{ENC}}$, *i.e.*, $\#\mathrm{SB}^3[0,2,3,6,11,12]$ (resp. $\#\mathrm{MC}^3[0,10,15,14,7,12]$) as $g_0, g_1, g_2, g_3, g_4, g_5$. We also denote the DoF costs in the blue propagation: $\#\mathrm{AK}^4[3,4,9,14]$ as c_0, c_1, c_2, c_3 and $\#\mathrm{MC}^1[1,4,11,14]$ as c_4, c_5, c_6, c_7.

The Core SCIS MITM function. The pseudocode for the core SCIS MITM function f is given in Algorithm 2, which takes the input of a set of values for the enumerating constants g_{ec}, and outputs candidates satisfying the partial target. We begin by fixing $z = c_0, c_1, c_2, c_3$ to zero. Line 1 to Line 15 is a nested MITM procedure which generates 2^{40} values of $\#\mathrm{SB}^3[\blacksquare]$ satisfying the values of enumerating constants $g_{ec} = g_0, g_1, g_2, g_3, g_4, g_5$ and indexes them according to the values of indexing constants $g_{ic} = c_4, c_5, c_6, c_7$ ($d_{g_{ic}} = 32$). Subsequently, with different values of g_{ic}, the red values are computed to the matching point. We set the partial match constraint as $\#\mathrm{AT}^7[4] \oplus M[4] = \texttt{0x00}$[8]. If a red value combined with g_{ic} satisfies the partial match, then every $\#\mathrm{SB}^3[\blacksquare]$ in $T_{\mathrm{blue}}[g_{ic}]$ combined with the red value is a valid candidate satisfying the partial match. Each matched candidate corresponds to a tuple (M, H), where H is the output hash value with its fourth byte $H[4]$ being $\texttt{0x00}$, and M is the preimage of H.

Nested MITM Generation of Blue Neutral Words. We now explain Line 1 to Line 15 of Algorithm 2 (illustrated in Figure 7) in detail, which generates blue neutral words complying with g_{ec} and z. The steps in Figure 7 are in essence an MITM procedure between bytes marked with $a = a_0, a_1, a_2, a_3$ and $b = b_0, b_1, b_2, b_3$: First, 2^{32} values of $x = x_0, x_1, x_2, x_3$ are enumerated (in Figure 7(a)). Then for 2^8 possible values of a_0, $\texttt{id}_0 = \texttt{MC}^{-1}(a_3, 0, 0, a_2)[2]$ is computed and a is stored in $T_a[\texttt{id}_0]$. Similarly, for 2^8 possible values of b_0, $\texttt{id}_1 = \texttt{MC}^{-1}(0, b_3, b_2, 0)[2]$ is computed and matched with a in table T_a with the constraint $\texttt{id}_0 \oplus \texttt{id}_1 = g_1$ (in Figure 7(d)). As the matching filter is 2^{-8}, we obtain $2^{32+8+8-8} = 2^{40}$ blue candidates, store in T_{blue} and index them according to values of indexing constants g_{ic}.

[8] Though in Figure 6, $H[4]$ and $H[14]$ can be both exploited as partial match filter ($d_{\mathcal{M}} = 2$ bytes), we only use $H[4]$ as filter, *i.e.*, the partial target is $H[4] = \#\mathrm{AT}^7[4] \oplus M[4] = \texttt{0x00}$ ($d_{\mathcal{M}} = 1$ byte).

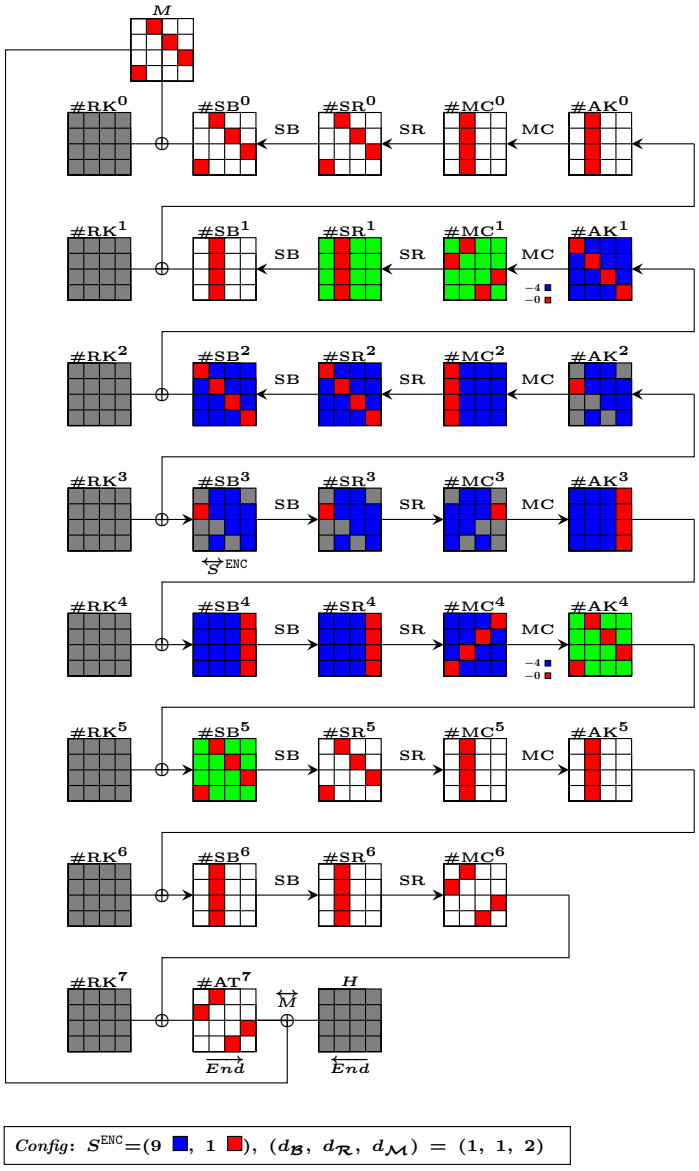


Fig. 6. MITM collision attack on 7-round AES-MMO/MP.

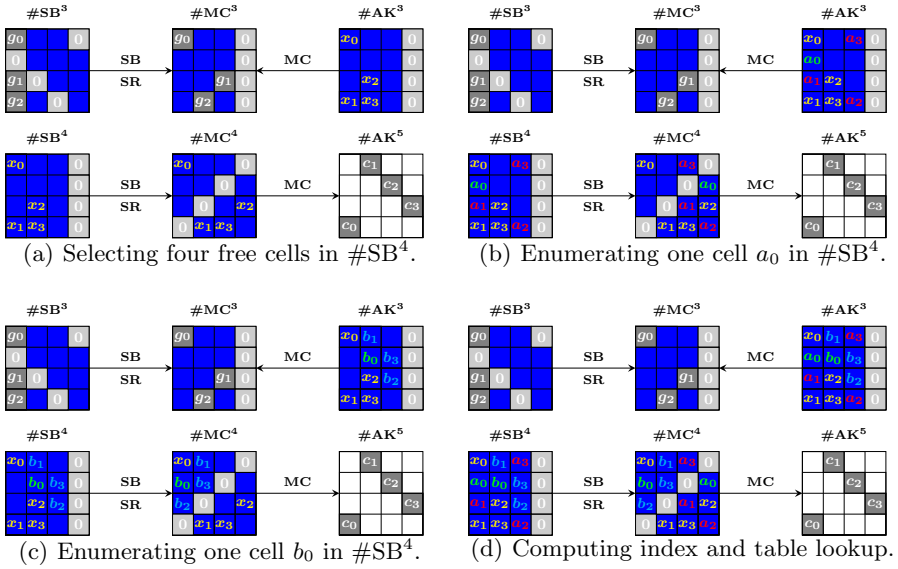
Complexity Analysis. In Algorithm 2, the generation of blue neutral words that comply with g_{ec} and z from Line 1 to Line 15 requires 2^{40} time and 2^8 memory complexity. According to Section 4.5, as we set $d_{\mathcal{M}} = 8$ following Footnote 8, and $d_{\mathcal{B}} = d_{\mathcal{R}} = 8$, we compute the overall time complexity following Equation (4) as $2^{128/2+8/2-8} = 2^{60}$, *i.e.*, enumerating 2^{20} g_{ec} for Algorithm 2.

Algorithm 2: The core MITM Function $f(g_{ec})$ for collision attack on 7-round AES-MMO/MP

Input: A set of constants for enumerating constants $g_{ec} := g_0||g_1||g_2||g_3||g_4||g_5$
Output: 2^{40} candidates stored in T_{dp} or a collision pair

`// Nested MITM to generate blue neutral word space in SCIS`
for 2^{32} *values of* x_0, x_1, x_2, x_3 *in* $\#\mathrm{SB}^4$ **do**
  `// Steps below are shown in Figure 7(b)`
  for 2^8 *values of* a_0 *in* $\#\mathrm{SB}^4$ **do**
    Compute to get a_1 according to the constant g_0 and Property 1;
    Compute to get a_2 according to the constant c_3 and Property 1;
    Compute to get a_3 according to the constant c_2 and Property 1;
    Compute the index value by $\mathtt{id}_0 = \mathtt{MC}^{-1}(a_3, 0, 0, a_2)[2]$;
    Store the value (a_0, a_1, a_2, a_3) under $T_a[\mathtt{id}_0]$;
  `// Steps below are shown in Figure 7(c)`
  for 2^8 *values of* b_0 *in* $\#\mathrm{SB}^4$ **do**
    Compute to get b_1 according to the constant g_2 and Property 1;
    Compute to get b_2 according to the constant c_0 and Property 1;
    Compute to get b_3 according to the constant c_1 and Property 1;
    Compute the index value by $\mathtt{id}_1 = \mathtt{MC}^{-1}(0, b_3, b_2, 0)[2]$;
    `// To match the constant` g_1
    Lookup table by $T_a[\mathtt{id}_1 + g_1]$ to get all possible (a_0, a_1, a_2, a_3);
    Obtain the blue neutral word $\#\mathrm{SB}^3[$■$]$ as shown in Figure 7(d);
    `// There will be` $2^{32+8+8-8} = 2^{40}$ `solutions on average`
    Compute to $g_{ic} = c_4, c_5, c_6, c_7$ and store $\#\mathrm{SB}^3[$■$]$ under $T_{blue}[g_{ic}]$;
`// Partial target candidates for the MITM collision attack`
for 2^{32} *values of* c_4, c_5, c_6, c_7 **do**
  for 2^8 *values of* $\#\mathrm{SB}^3[$■$]$ **do**
    Compute to $\#\mathrm{AT}^7[4]$ and $M[4]$;
    if $\#\mathrm{AT}^7[4] = M[4]$ **then**
      for 2^8 *values in* $T_{blue}[c_4, c_5, c_6, c_7]$ **do**
        `// There will be` $2^{32+8-8+8} = 2^{40}$ `candidates on average`
        Compute the preimage M and hash value H based on full $\#\mathrm{SB}^3$;
        if $T_{dp}[H]$ *is non-empty* **then**
          return collision pair $(T_{dp}[H], M)$;
        else
          $T_{dp}[H] \leftarrow M$;
return T_{dp};

The memory complexity is dominated by T_{dp} for the collision attack, which is $2^{(128-8)/2} = 2^{60}$. Thus, the overall MITM collision attack costs 2^{60} time and 2^{60} memory complexity.



(a) Selecting four free cells in $\#\mathrm{SB}^4$. (b) Enumerating one cell a_0 in $\#\mathrm{SB}^4$.

(c) Enumerating one cell b_0 in $\#\mathrm{SB}^4$. (d) Computing index and table lookup.

Fig. 7. Pre-computation of the blue neutral words given constants in MITM step.

Experiment and Verification. To show the validity of the proposed technique and attack above, we provide an experiment of the 7-round partial target attack on AES-MMO as below according to the MITM trail in Figure 6, which can be further converted into the MITM collision attack given in Algorithm 2. In this experiment, the constants $g_{ec} = g_0, ..., g_5$ in $\#\mathrm{SB}^3$ and $z = c_0, ..., c_3$ in $\#\mathrm{AK}^4$ are fixed. It takes 2^{40} time and 2^8 memory to generate T_{blue} (without storing) with $2^{d_{g_{ic}}} = 2^{32}$ indices. Each index of T_{blue} is associated with about 2^8 blue neutral words. To verify the correctness of our method, we test only one index $g_{ic} = c_4, c_5, c_6, c_7 = 0$ and its associated blue neutral words. As $d_{\mathcal{R}} = d_{\mathcal{M}} = 8$, the number of expected candidates satisfying the partial target attack is about $2^{d_{\mathcal{B}}+d_{\mathcal{R}}-d_{\mathcal{M}}} = 2^{8+8-8} = 2^8$.

Following the above steps, the experiment requires 2^{40} time complexity (evaluated in 7-round AES operations and takes about 11 hours on a single core of 3.6 GHz 10-Core Intel Core i9) and generates about 2^8 candidates (all candidates are verified with the fourth output byte $H[4]$ being `0x00`). The experiment and verification of this 7-round partial target attack on AES-MMO can be found at https://github.com/csy1234/scis_mitm.

6 Classical Collision Attack on 4-Round AES-128-DM

In this section, we provide the first one-block collision attack on 4-round AES-128-DM, which is equivalent to fixed-target-plaintext key collision attack on AES and as shown in Figure 8. Here, the message P is fixed to the constant IV, and

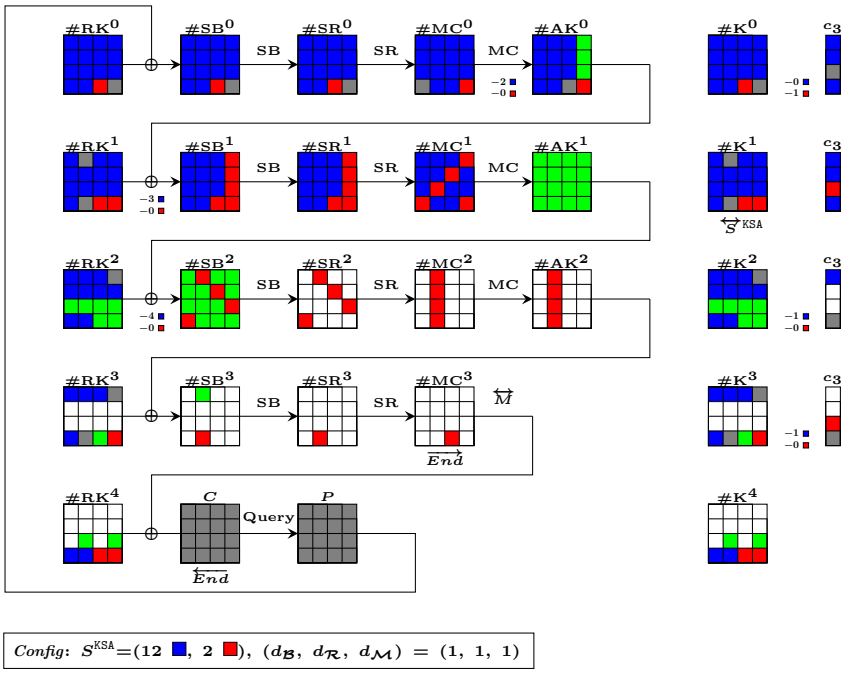


Fig. 8. MITM collision attack on 4-round AES-128-DM.

we let $\#\mathrm{K}^r$ be the updated key state of the key schedule and $\#\mathrm{K}^r = \#\mathrm{RK}^r$ $(0 \leq r \leq 4)$.

The Core SCIS MITM function. The pseudocode for the core SCIS MITM function f is given in Algorithm 3. Line 1 to Line 3 generates and stores 2^{40} solutions of the blue neutral word space that comply with $g_{ec} = \#\mathrm{K}^0[15], \#\mathrm{K}^1[4,7], \#\mathrm{K}^2[12], \#\mathrm{K}^3[12], \#\mathrm{AK}^0[11,15], \#\mathrm{SB}^1[12,13,14]$ in T_{blue}, which are indexed by $g_{ic} = \#\mathrm{SB}^2[3,4,9,14]$. Given any fixed constants g_{ec}, one can recover $\#\mathrm{K}^0[\blacksquare]$ according to $\#\mathrm{SB}^0[7,8,12,13], \#\mathrm{K}^0[14]$ and Table 3, then it has enough information to compute to the cost-related constants g_{ic} corresponding to $\#\mathrm{SB}^2[\blacksquare]$. Subsequently, with different values of g_{ic}, the red values are computed to the matching point. We set the partial match constraint as $\#\mathrm{MC}^3[11] \oplus \#\mathrm{RK}^4[11] = \texttt{0x00}$. If a red value combined with g_{ic} satisfies this partial match, then every blue neutral word in $T_{blue}[g_{ic}]$ combined with the red value is a valid candidate satisfying the partial target. Each matched candidate corresponds to a tuple (K, C), where C is the output value with its eleventh byte $C[11]$ being $\texttt{0x00}$, and K is the corresponding masterkey. Using the proposed conversion in Theorem 1, we are able to construct a fixed-target-plaintext key collision on 4-round AES-128, equivalently a one-block collision on 4-round AES-128-DM.

Algorithm 3: The core MITM Function $f(g)$ for collision attack on 4-round AES-128-DM

Input: $\#\mathrm{K}^0[15], \#\mathrm{K}^1[4,7], \#\mathrm{K}^2[12], \#\mathrm{K}^3[12], \#\mathrm{AK}^0[11,15], \#\mathrm{SB}^1[12,13,14]$
Output: 2^{40} candidates stored in T_{dp} or a collision pair

```
1  for 2^40 values of #SB^0[7,8,12,13], #K^0[14] do
2      Deduce #K^0[■] according to Table 3;
3      Compute to g_ic in #SB^2, store the blue neutral word under T_blue[g_ic];
4  for 2^32 values of g_ic in #SB^2 do
5      for 2^8 values of #K^0[■] do
6          Compute to get #MC^3[11] and #K^4[11];
7          if #MC^3[11] = #K^4[11] then
8              for 2^8 values in T_blue[g_ic] do
9                  Recover masterkey K and compute the ciphertext C;
10                 if T_dp[C] is non-empty then
11                     return collision pair (T_dp[C], K);
12                 else
13                     T_dp[C] ← K;
14 return T_dp;
```

Complexity Analysis. Similarly, in Algorithm 3, the generation of blue neutral words that comply with g_{ec} from Line 1 to Line 3 requires 2^{40} time complexity and constant memory. As we have $d_{\mathcal{B}} = d_{\mathcal{R}} = d_{\mathcal{M}} = 8$, following Equation (4), the overall time complexity is evaluated as $2^{128/2+8/2-8} = 2^{60}$. The memory complexity is dominated by T_{dp}, which is $2^{(128-8)/2} = 2^{60}$. Thus, the attack costs 2^{60} time and 2^{60} memory complexity.

Experiment and Verification. Similarly, we provide an experiment of the 4-round partial target attack on AES-DM as below according to the MITM trail in Figure 8, which can be further converted into the MITM collision attack given in Algorithm 3. In this experiment, the enumerating constants g_{ec} are fixed. It takes 2^{40} time and 2^8 memory to generate T_{blue} (without storing) with $2^{d_{g_{ic}}} = 2^{32}$ indices. Each index of T_{blue} is associated with about 2^8 blue neutral words. To verify the correctness of our method, we test only one index $g_{ic} = 0$ and its associated blue neutral words. As $d_{\mathcal{R}} = d_{\mathcal{M}} = 8$, the number of expected candidates satisfying the partial target is about $2^{d_{\mathcal{B}}+d_{\mathcal{R}}-d_{\mathcal{M}}} = 2^{8+8-8} = 2^8$.

Following the above steps, the experiment requires 2^{40} time complexity (evaluated in 4-round AES operations and takes about 6 hours on a single core of 3.6 GHz 10-Core Intel Core i9) and generates about 2^8 candidates (all candidates are verified with the fourth output byte $C[11]$ being `0x00`). The experiment and verification of this 4-round partial target attack on AES-DM can be found at https://github.com/csy1234/scis_mitm.

Table 3. Steps to recover 14 blue cells of $\#\mathrm{K}^0$ in Figure 8 according to fixed constants $\#\mathrm{K}^0[15], \#\mathrm{K}^1[4,7], \#\mathrm{K}^2[12], \#\mathrm{K}^3[12], \#\mathrm{AK}^0[11,15], \#\mathrm{SB}^1[12,13,14]$ and known (P, C). The identification of such steps can be facilitated by the algebraic methods proposed by Bouillaguet, Derbez, and Fouque [22] at Crypto 2011.

Enumerating 5 bytes $\#\mathrm{SB}^0[7,8,12,13], \#\mathrm{K}^0[14]$
1. $\#\mathrm{SB}^0[15] = \#\mathrm{K}^0[15] \oplus P[15]$
2. $\#\mathrm{K}^3[7] = \#\mathrm{K}^1[7] \oplus \mathtt{S}(\#\mathrm{K}^2[12])$
3. $\#\mathrm{K}^3[8] = \#\mathrm{K}^3[12] \oplus \#\mathrm{K}^2[12]$
4. $\#\mathrm{K}^0[8] = \#\mathrm{SB}^0[8] \oplus P[8]$
5. $\#\mathrm{K}^0[12] = \#\mathrm{SB}^0[12] \oplus P[12]$
6. $\#\mathrm{K}^0[13] = \#\mathrm{SB}^0[13] \oplus P[13]$
7. $\#\mathrm{K}^2[13] \xleftarrow{\mathtt{S}^{-1}} \#\mathrm{K}^0[8] \oplus \#\mathrm{K}^0[12] \oplus \#\mathrm{K}^3[12]$
8. $\#\mathrm{K}^1[12] = \#\mathrm{K}^1[4] \oplus \#\mathrm{K}^0[8,12] \oplus \mathtt{S}(\#\mathrm{K}^2[13])$
9. $\#\mathrm{AK}^0[12] = \#\mathrm{SB}^1[12] \oplus \#\mathrm{K}^1[12]$
10. $\#\mathrm{AK}^0[13] \xleftarrow[\text{relation}]{\mathtt{MC}} (\#\mathrm{AK}^0[15], \#\mathrm{SB}^0[12], \#\mathrm{AK}^0[12])$
11. $\#\mathrm{K}^1[13] = \#\mathrm{AK}^0[13] \oplus \#\mathrm{SB}^1[13]$
12. $\#\mathrm{AK}^0[14] \xleftarrow[\text{relation}]{\mathtt{MC}} (\#\mathrm{AK}^0[15], \#\mathrm{AK}^0[12], \#\mathrm{AK}^0[13])$
13. $\#\mathrm{K}^1[14] = \#\mathrm{AK}^0[14] \oplus \#\mathrm{SB}^1[14]$
14. $\#\mathrm{SB}^0[1,6] \xleftarrow[\text{relation}]{\mathtt{MC}} (\#\mathrm{AK}^0[15], \#\mathrm{AK}^0[12], \#\mathrm{AK}^0[13])$
15. $\#\mathrm{SB}^0[2] \xleftarrow{\mathtt{S}^{-1}} \mathtt{MC}(\mathtt{S}(\#\mathrm{SB}^0[7]), \mathtt{S}(\#\mathrm{SB}^0[8]), \mathtt{S}(\#\mathrm{SB}^0[13]), \#\mathrm{AK}^0[11])$
16. $\#\mathrm{K}^0[7] = \#\mathrm{SB}^0[7] \oplus P[7]$
17. $\#\mathrm{K}^0[3] = \#\mathrm{K}^0[7] \oplus \#\mathrm{K}^1[7] \oplus \mathtt{S}(\#\mathrm{K}^0[12])$
18. $\#\mathrm{SB}^0[3] = P[3] \oplus \#\mathrm{K}^0[3]$
19. $\#\mathrm{K}^3[0] = \#\mathrm{K}^0[12] \oplus \#\mathrm{K}^1[4] \oplus \#\mathrm{K}^2[12] \oplus \mathtt{S}(\#\mathrm{K}^2[13])$
20. $\#\mathrm{K}^0[0] = \#\mathrm{K}^3[0] \oplus \mathtt{S}(\#\mathrm{K}^0[13]) \oplus \mathtt{S}(\#\mathrm{K}^1[13]) \oplus \mathtt{S}(\#\mathrm{K}^2[13])$
21. $\#\mathrm{K}^0[4] = \#\mathrm{K}^0[0] \oplus \mathtt{S}(\#\mathrm{K}^0[13])$
22. $\#\mathrm{K}^2[5] = \#\mathrm{K}^0[13] \oplus \#\mathrm{K}^2[13]$
23. $\#\mathrm{K}^0[5] = \#\mathrm{K}^2[5] \oplus \mathtt{S}(\#\mathrm{K}^1[14])$
24. $\#\mathrm{K}^0[1] = \#\mathrm{SB}^0[1] \oplus P[1]$
25. $\#\mathrm{K}^2[1] = \#\mathrm{K}^0[1] \oplus \mathtt{S}(\#\mathrm{K}^0[14]) \oplus \mathtt{S}(\#\mathrm{K}^1[14])$
26. $\#\mathrm{K}^0[9] = \#\mathrm{K}^2[1] \oplus \#\mathrm{K}^1[13] \oplus \#\mathrm{K}^2[13]$
27. $\#\mathrm{K}^1[10] = \#\mathrm{K}^0[14] \oplus \#\mathrm{K}^1[14]$
28. $\#\mathrm{K}^0[6] = \#\mathrm{SB}^0[6] \oplus P[6]$
29. $\#\mathrm{K}^0[2] = \#\mathrm{SB}^0[2] \oplus P[2]$
30. $\#\mathrm{K}^1[2] = \#\mathrm{K}^0[2] \oplus \mathtt{S}(\#\mathrm{K}^0[15])$
31. $\#\mathrm{K}^1[6] = \#\mathrm{K}^0[6] \oplus \#\mathrm{K}^1[2]$
32. $\#\mathrm{K}^0[10] = \#\mathrm{K}^1[6] \oplus \#\mathrm{K}^1[10]$

7 Improved SKP Key Recovery Attack on 5-Round AES-128

We present the improved single-known-plaintext key recovery attack on 5-round AES-128, which is as shown in Figure 9. Compared with [22,31], we have reduced the memory complexity from 2^{96} to 2^{40} with the SCIS technique.

The Core SCIS MITM function. The pseudocode for the core SCIS MITM function f is given in Algorithm 4. Line 1 to Line 6 generates and stores 2^{40} blue neutral words that satisfy $g_{ec} = \#\mathrm{K}^0[0,2,3,4,11], \#\mathrm{K}^2[12,14], \#\mathrm{SB}^1[6,11,12]$, indexed by $g_{ic} = \#\mathrm{SB}^1[1], \#\mathrm{MC}^3[1,4], \#\mathrm{MC}^2[6]$. Subsequently, for different values of g_{ic}, the red values are computed to the matching point and then filtered by the partial match constraint with a probability of 2^{-8}. The candidates that

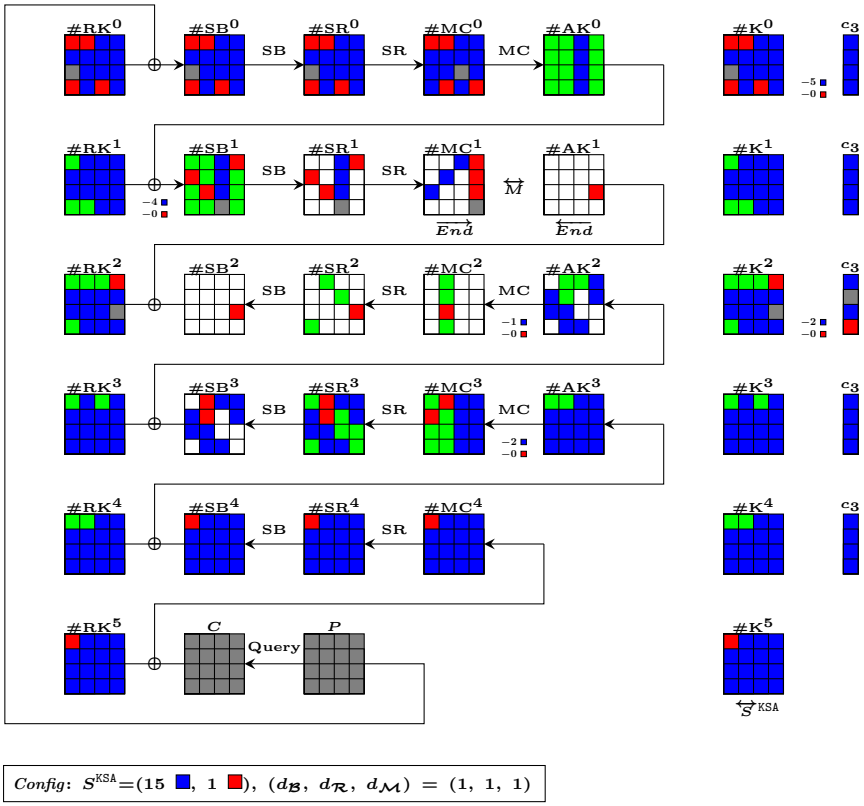


Fig. 9. Improved SKP key recovery attack on 5-round AES-128.

pass the partial match are used to recover the masterkey $\#\mathrm{K}^0$, which is then checked if it is the correct key for the given pair of plaintext/ciphertext (P, C).

Nested MITM Generation of Blue Neutral Words. For Line 1 to Line 6 in Algorithm 4, it is essentially an MITM procedure between two sets of key bytes: $a = \#\mathrm{K}^1[14], \#\mathrm{K}^2[13], \#\mathrm{K}^3[14], \#\mathrm{K}^4[13]$ and $b = \#\mathrm{K}^0[7, 12, 13, 14], \#\mathrm{K}^1[15]$. With the fixed constants, the values of $m = \#\mathrm{K}^0[5, 13], \#\mathrm{K}^3[6], \mathtt{S}(\#\mathrm{K}^2[13]) \oplus \mathtt{S}(\#\mathrm{K}^4[13])$, independently computable from a and from b, are used for matching. First, 2^{32} values of a is enumerated. For each a, the partial values of m^a are computed, then a is stored in $T_a[m^a]$. We then iterate all 2^{40} values of b, compute m^b following Table 4 and match in T_a. As a result, we obtain $2^{32+40-32} = 2^{40}$ blue candidates.

Complexity Analysis. In Algorithm 4, the generation of blue candidates from Line 1 to Line 6 requires 2^{40} time and 2^{40} memory complexity. As we have $d_{g_{ic}} = 32$ and $d_{\mathcal{B}} = d_{\mathcal{R}} = d_{\mathcal{M}} = 8$, we compute $M_f = 2^{32+8} = 2^{40}$ following Equation (2). We then evaluate the overall time complexity as $2^{128-8} = 2^{120}$

Algorithm 4: The core MITM Function $f(g)$ for SKP key recovery attack on 5-round AES-128

Input: A set of constants $\#\mathrm{K}^0[0,2,3,4,11], \#\mathrm{K}^2[12,14], \#\mathrm{SB}^1[6,11,12]$
Output: The master key $\#\mathrm{K}^0$ satisfying the given plaintext/ciphertext
`// Initialization phase (table` T_a`)`
for 2^{32} *values of* $\#\mathrm{K}^1[14], \#\mathrm{K}^2[13], \#\mathrm{K}^3[14], \#\mathrm{K}^4[13]$ **do**
  // $\#\mathrm{K}^0[5] = \#\mathrm{K}^2[13] \oplus \#\mathrm{K}^4[13] \oplus \mathtt{S}(\#\mathrm{K}^1[14]) \oplus \mathtt{S}(\#\mathrm{K}^3[14])$
  // $\#\mathrm{K}^0[13] = \#\mathrm{K}^4[13] \oplus \mathtt{S}(\#\mathrm{K}^3[14])$
  // $\#\mathrm{K}^3[6] = \#\mathrm{K}^1[14] \oplus \#\mathrm{K}^3[14]$
  Compute to $m^a = \#\mathrm{K}^0[5,13], \#\mathrm{K}^3[6], \mathtt{S}(\#\mathrm{K}^2[13]) \oplus \mathtt{S}(\#\mathrm{K}^4[13])$;
  Store the value of $(\#\mathrm{K}^1[14], \#\mathrm{K}^2[13], \#\mathrm{K}^3[14], \#\mathrm{K}^4[13])$ under $T_a[m^a]$;
for 2^{40} *values of* $\#\mathrm{K}^0[7,12,13,14], \#\mathrm{K}^1[15]$ **do**
  Deduce $\#\mathrm{K}^0[$■$]$ according to Table 4 and T_a;
  Compute to $g_{ic} = \#\mathrm{SB}^1[1], \#\mathrm{MC}^3[1,4], \#\mathrm{MC}^2[6]$, store the value under $T_{blue}[g_{ic}]$;
for 2^{32} *values of* $\#\mathrm{SB}^1[1], \#\mathrm{MC}^3[1,4], \#\mathrm{MC}^2[6]$ **do**
  for 2^8 *values of* $\#\mathrm{K}^5[$■$]$ **do**
    Compute to $\#\mathrm{MC}^1[12,13,14,15]$ and $\#\mathrm{AK}^1[14]$;
    if $\mathtt{MC}(\#\mathrm{MC}^1[12,13,14,15])[2] = \#\mathrm{AK}^1[14]$ **then**
      for 2^8 *values in* $T_{blue}[\#\mathrm{SB}^1[1], \#\mathrm{MC}^3[1,4], \#\mathrm{MC}^2[6]]$ **do**
        Check the full match based on blue and red neutral words;
        if *A full match is found* **then**
          return $\#\mathrm{K}^0$;

following Equation (3). Thus, the attack costs 2^{120} time and 2^{40} memory complexity.

Experiment and Verification. An experiment of this 5-round key recovery partial match attack on AES-128 is provided as below according to the MITM trail in Figure 9, which can be further converted into the MITM key recovery attack given in Algorithm 4. In this experiment, the enumerating constants g_{ec} are fixed. It takes 2^{40} time and 2^{32} memory to generate T_{blue} (without storing), which contains about 2^{40} neutral words. To verify the correctness of our method, we fix the red byte in $\#\mathrm{K}^5$ and check if the 2^{40} blue neutral words can match with the fixed red bytes. As $d_{\mathcal{M}} = 8$, the number of expected candidates satisfying the partial match is about $2^{40-d_{\mathcal{M}}} = 2^{40-8} = 2^{32}$.

Following the above steps, the experiment requires 2^{40} time complexity (evaluated in 5-round AES operations and takes about 73 hours on a single core of 2.44 GHz AMD EPYC 7763 64-Core Processor) and 2^{32} memory complexity (costs about 64 GB memory due to two tables of size 2^{32} storing 64-bit value, *i.e.*, $2 \cdot 2^{32} \cdot 64/8/1024^3 = 64$ GB), which generates about 2^{32} candidates. The experiment and verification of this 5-round key recovery partial match attack on AES-128 can be found at `https://github.com/csy1234/scis_mitm`.

Table 4. Steps to recover 11 blue cells of $\#\mathrm{K}^0$ in Figure 9 according to fixed constants $\#\mathrm{K}^0[0,2,3,4,11], \#\mathrm{K}^2[12,14], \#\mathrm{SB}^1[6,11,12]$ and known (P, C). The identification of such steps can be facilitated by the algebraic methods in [22].

Enumerating 5 bytes $\#\mathrm{K}^0[7,12,13,14], \#\mathrm{K}^1[15]$
1. $\#\mathrm{K}^0[15] = \#\mathrm{K}^0[7] \oplus \#\mathrm{K}^0[11] \oplus \#\mathrm{K}^0[3] \oplus \#\mathrm{K}^1[15] \oplus \mathtt{S}(\#\mathrm{K}^0[12])$
2. $\#\mathrm{SB}^0[2] = \#\mathrm{K}^0[2] \oplus P[2]$
3. $\#\mathrm{SB}^0[7,13] = \#\mathrm{K}^0[7,13] \oplus P[7,13]$
4. $\#\mathrm{K}^1[11] = \#\mathrm{K}^0[15] \oplus \#\mathrm{K}^1[15]$
5. $\#\mathrm{AK}^0[11] = \#\mathrm{SB}^1[11] \oplus \#\mathrm{K}^1[11]$
6. $\#\mathrm{SB}^0[8] \xleftarrow{\mathtt{S}^{-1}} \mathtt{MC}(\#\mathrm{AK}^0[11], \mathtt{S}(\#\mathrm{SB}^0[2]), \mathtt{S}(\#\mathrm{SB}^0[7]), \mathtt{S}(\#\mathrm{SB}^0[13]))$
7. $\#\mathrm{K}^0[8] = \#\mathrm{SB}^0[8] \oplus P[8]$
8. $\#\mathrm{SB}^0[3,4] = \#\mathrm{K}^0[3,4] \oplus P[3,4]$
9. $\#\mathrm{SB}^0[14] = \#\mathrm{K}^0[14] \oplus P[14]$
10. $\#\mathrm{K}^2[2] = \#\mathrm{K}^0[2] \oplus \mathtt{S}(\#\mathrm{K}^1[15]) \oplus \mathtt{S}(\#\mathrm{K}^0[15])$
11. $\#\mathrm{K}^2[6] = \#\mathrm{K}^0[14] \oplus \#\mathrm{K}^2[14]$
12. $\#\mathrm{K}^1[6] = \#\mathrm{K}^2[2] \oplus \#\mathrm{K}^2[6]$
13. $\#\mathrm{AK}^0[6] = \#\mathrm{SB}^1[6] \oplus \#\mathrm{K}^1[6]$
14. $\#\mathrm{SB}^0[9] \xleftarrow{\mathtt{S}^{-1}} \mathtt{MC}(\#\mathrm{AK}^0[6], \mathtt{S}(\#\mathrm{SB}^0[3]), \mathtt{S}(\#\mathrm{SB}^0[4]), \mathtt{S}(\#\mathrm{SB}^0[14]))$
15. $\#\mathrm{K}^0[9] = \#\mathrm{SB}^0[9] \oplus P[9]$
16. $\#\mathrm{K}^2[6] = \#\mathrm{K}^0[14] \oplus \#\mathrm{K}^2[14]$
17. $\#\mathrm{K}^0[6] = \#\mathrm{K}^2[6] \oplus \mathtt{S}(\#\mathrm{K}^1[15])$
18. $\#\mathrm{SB}^0[6] = \#\mathrm{K}^0[6] \oplus P[6]$
19. $\#\mathrm{SB}^0[11] = \#\mathrm{K}^0[11] \oplus P[11]$
20. $\#\mathrm{SB}^0[12] = \#\mathrm{K}^0[12] \oplus P[12]$
21. $\#\mathrm{K}^1[12] = \#\mathrm{K}^0[0] \oplus \#\mathrm{K}^0[4] \oplus \#\mathrm{K}^0[8] \oplus \#\mathrm{K}^0[12] \oplus \mathtt{S}(\#\mathrm{K}^0[13])$
22. $\#\mathrm{AK}^0[12] = \#\mathrm{SB}^1[12] \oplus \#\mathrm{K}^1[12]$
23. $\#\mathrm{SB}^0[1] \xleftarrow{\mathtt{S}^{-1}} \mathtt{MC}(\#\mathrm{AK}^0[12], \mathtt{S}(\#\mathrm{SB}^0[6]), \mathtt{S}(\#\mathrm{SB}^0[11]), \mathtt{S}(\#\mathrm{SB}^0[12]))$
24. $\#\mathrm{K}^0[1] = \#\mathrm{SB}^0[1] \oplus P[1]$
25. $\#\mathrm{K}^1[13] \xleftarrow{\mathtt{S}^{-1}} \#\mathrm{K}^0[12] \oplus \#\mathrm{K}^0[4] \oplus \#\mathrm{K}^2[12]$
26. $\#\mathrm{K}^3[13] = \#\mathrm{K}^0[9] \oplus \#\mathrm{K}^0[13] \oplus \mathtt{S}(\#\mathrm{K}^2[14])$
27. $\#\mathrm{K}^2[15] = \#\mathrm{K}^0[3] \oplus \#\mathrm{K}^0[11] \oplus \#\mathrm{K}^1[15] \oplus \mathtt{S}(\#\mathrm{K}^0[12]) \oplus \mathtt{S}(\#\mathrm{K}^1[12])$
28. $\#\mathrm{K}^0[5] = \#\mathrm{K}^0[1] \oplus \#\mathrm{K}^1[13] \oplus \#\mathrm{K}^3[13] \oplus \mathtt{S}(\#\mathrm{K}^0[14]) \oplus \mathtt{S}(\#\mathrm{K}^2[14])$
29. $\#\mathrm{K}^3[6] = \#\mathrm{K}^0[2] \oplus \#\mathrm{K}^0[14] \oplus \#\mathrm{K}^2[14] \oplus \mathtt{S}(\#\mathrm{K}^0[15]) \oplus \mathtt{S}(\#\mathrm{K}^1[15]) \oplus \mathtt{S}(\#\mathrm{K}^2[15])$
Compute index value $m^b = \mathtt{id}_0 \| \mathtt{id}_1 \| \mathtt{id}_2 \| \mathtt{id}_3$ with
$\mathtt{id}_0 = \#\mathrm{K}^0[5]$
$\mathtt{id}_1 = \#\mathrm{K}^0[13]$
$\mathtt{id}_2 = \#\mathrm{K}^3[6]$
$\mathtt{id}_3 = \#\mathrm{K}^0[0] \oplus \mathtt{S}(\#\mathrm{K}^0[13]) \oplus \mathtt{S}(\#\mathrm{K}^1[13]) \oplus \mathtt{S}(\#\mathrm{K}^3[13]) = \mathtt{S}(\#\mathrm{K}^2[13]) \oplus \mathtt{S}(\#\mathrm{K}^4[13])$
Lookup table T_a in Algorithm 4 to obtain $\#\mathrm{K}^1[14], \#\mathrm{K}^2[13], \#\mathrm{K}^3[14], \#\mathrm{K}^4[13]$
30. $\#\mathrm{K}^3[15] = \#\mathrm{K}^0[11] \oplus \#\mathrm{K}^0[15] \oplus \mathtt{S}(\#\mathrm{K}^2[12])$
31. $\#\mathrm{K}^4[14] = \#\mathrm{K}^0[14] \oplus \mathtt{S}(\#\mathrm{K}^3[15])$
32. $\#\mathrm{K}^3[12] = \#\mathrm{K}^0[4] \oplus \#\mathrm{K}^1[12] \oplus \#\mathrm{K}^1[13] \oplus \#\mathrm{K}^3[13] \oplus \#\mathrm{K}^4[13]$
33. $\#\mathrm{K}^4[15] = \#\mathrm{K}^0[11] \oplus \#\mathrm{K}^3[15] \oplus \mathtt{S}(\#\mathrm{K}^2[12]) \oplus \mathtt{S}(\#\mathrm{K}^3[12])$
34. $\#\mathrm{K}^5[2] = \#\mathrm{K}^1[14] \oplus \#\mathrm{K}^2[14] \oplus \#\mathrm{K}^3[14] \oplus \mathtt{S}(\#\mathrm{K}^4[15])$
35. $\#\mathrm{K}^5[10] = \#\mathrm{K}^2[14] \oplus \#\mathrm{K}^3[14] \oplus \#\mathrm{K}^5[2]$
36. $\#\mathrm{K}^0[10] = \#\mathrm{K}^2[14] \oplus \#\mathrm{K}^5[10] \oplus \#\mathrm{K}^5[2] \oplus \#\mathrm{K}^4[14] \oplus \mathtt{S}(\#\mathrm{K}^2[15]) \oplus \mathtt{S}(\#\mathrm{K}^3[15])$

8 Conclusion

This work proposes the single-color initial structure (SCIS) technique to identify meet-in-the-middle (MITM) attack trails that enable efficient neutral word generation and low-memory attacks. With its help, we have attained the following results. First and foremost, we initiate the first classical one-block collision attack on 7-round AES-MMO/MP. This result marks the first advancement in attack

rounds in over a decade and matches the attack rounds in the quantum setting. For key collision attacks, we extend the attack on AES-128 to 4 rounds, which bridges the gap in Taiyama *et al.*'s claim that no feasible key collision attacks on more than 3 rounds under the estimation of single differential characteristics. For SKP key recovery attacks, we present an improved attack on 5-round AES-128, new attacks on 7-round AES-192 and 8-round AES-256 and improved attack on 7-round EM-AES-128. Last but not least, aligning with NIST's interest in standardizing wider variants of AES, we provide comprehensive results on Rijndael-192 and Rijndael-256. These results provide new insights into AES-like structures and show that both AES and its wide variants are sufficiently secure against the considered attacks.

Acknowledgements. We would like to thank all anonymous reviewers for their comments. We would also like to thank Wenjie Nan for fruitful discussions that improved the quality of this work. This research is supported by the National Key R&D Program of China (Grant No. 2024YFB4504900), the National Natural Science Foundation of China (Grants No. 62422214, 62472421, 62172410), the National Cryptologic Science Foundation of China (Grant No. 2025NCSF01012), the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDB0690200), the Youth Innovation Promotion Association of Chinese Academy of Sciences, the International Partnership Program of Chinese Academy of Sciences (Grant No. 205GJHZ2024005FN), the National Research Foundation, Singapore and Infocomm Media Development Authority under its Trust Tech Funding Initiative, the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) project no. 509754807, the Ministry of Education in Singapore under Grant RG102/24. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore and Infocomm Media Development Authority.

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.) USENIX S&P. pp. 3291–3308. USENIX Association (2022), https://www.usenix.org/conference/usenixsecurity22/presentation/albertini
2. Albrecht, M.R., Rechberger, C., Schneider, T., Tiessen, T., Zohner, M.: Ciphers for MPC and FHE. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 430–454. Springer (2015). https://doi.org/10.1007/978-3-662-46800-5_17, https://doi.org/10.1007/978-3-662-46800-5_17
3. Alliance, Z.: zigbee Specification Revision 22 1.0. Tech. rep., ZigBee Alliance (April 19 2017)
4. Aoki, K., Guo, J., Matusiewicz, K., Sasaki, Y., Wang, L.: Preimages for Step-Reduced SHA-2. In: Matsui, M. (ed.) ASIACRYPT. Lecture Notes in Computer Science, vol. 5912, pp. 578–597. Springer (2009), https://doi.org/10.1007/978-3-642-10366-7_34
5. Aoki, K., Sasaki, Y.: Preimage Attacks on One-Block MD4, 63-Step MD5 and More. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) SAC. Lecture Notes in Computer Science, vol. 5381, pp. 103–119. Springer (2008), https://doi.org/10.1007/978-3-642-04159-4_7
6. Aoki, K., Sasaki, Y.: Meet-in-the-Middle Preimage Attacks Against Reduced SHA-0 and SHA-1. In: Halevi, S. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 5677, pp. 70–89. Springer (2009), https://doi.org/10.1007/978-3-642-03356-8_5
7. Aumasson, J., Jr., J.N., Sepehrdad, P.: Cryptanalysis of the ISDB Scrambling Algorithm (MULTI2). In: Dunkelman, O. (ed.) FSE. Lecture Notes in Computer Science, vol. 5665, pp. 296–307. Springer (2009), https://doi.org/10.1007/978-3-642-03317-9_18
8. Bao, Z., Ding, L., Guo, J., Wang, H., Zhang, W.: Improved meet-in-the-middle preimage attacks against AES hashing modes. IACR Trans. Symmetric Cryptol. **2019**(4), 318–347 (2019), https://doi.org/10.13154/tosc.v2019.i4.318-347
9. Bao, Z., Dong, X., Guo, J., Li, Z., Shi, D., Sun, S., Wang, X.: Automatic Search of Meet-in-the-Middle Preimage Attacks on AES-like Hashing. In: EUROCRYPT I. pp. 771–804 (2021), https://doi.org/10.1007/978-3-030-77870-5_27
10. Bao, Z., Guo, J., Shi, D., Tu, Y.: Superposition Meet-in-the-Middle Attacks: Updates on Fundamental Security of AES-like Hashing. In: CRYPTO I. pp. 64–93 (2022), https://doi.org/10.1007/978-3-031-15802-5_3
11. Baum, C., Braun, L., de Saint Guilhem, C.D., Klooß, M., Majenz, C., Mukherjee, S., Orsini, E., Ramacher, S., Rechberger, C., Roy, L., Scholl, P.: Faest: algorithm specifications. Tech. rep., National Institute of Standards and Technology (2023), https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/FAEST-spec-web.pdf
12. Baum, C., Braun, L., de Saint Guilhem, C.D., Klooß, M., Orsini, E., Roy, L., Scholl, P.: Publicly Verifiable Zero-Knowledge and Post-Quantum Signatures from VOLE-in-the-Head. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO V. Lecture Notes in Computer Science, vol. 14085, pp. 581–615. Springer (2023), https://doi.org/10.1007/978-3-031-38554-4_19
13. Baum, C., Jadoul, R., Orsini, E., Scholl, P., Smart, N.P.: Feta: Efficient threshold designated-verifier zero-knowledge proofs. In: Yin, H., Stavrou, A., Cremers, C.,

Shi, E. (eds.) Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022. pp. 293–306. ACM (2022), https://doi.org/10.1145/3548606.3559354
14. Baum, C., de Saint Guilhem, C.D., Kales, D., Orsini, E., Scholl, P., Zaverucha, G.: Banquet: Short and fast signatures from AES. In: Garay, J.A. (ed.) Public-Key Cryptography - PKC 2021 - 24th IACR International Conference on Practice and Theory of Public Key Cryptography, Virtual Event, May 10-13, 2021, Proceedings, Part I. Lecture Notes in Computer Science, vol. 12710, pp. 266–297. Springer (2021), https://doi.org/10.1007/978-3-030-75245-3_11
15. Bellare, M., Hoang, V.T.: Succinctly-Committing Authenticated Encryption. In: Reyzin, L., Stebila, D. (eds.) CRYPTO IV. Lecture Notes in Computer Science, vol. 14923, pp. 305–339. Springer (2024), https://doi.org/10.1007/978-3-031-68385-5_10
16. Bhaumik, R., Chakraborty, B., Choi, W., Dutta, A., Govinden, J., Shen, Y.: The Committing Security of MACs with Applications to Generic Composition. In: Reyzin, L., Stebila, D. (eds.) CRYPTO IV. Lecture Notes in Computer Science, vol. 14923, pp. 425–462. Springer (2024), https://doi.org/10.1007/978-3-031-68385-5_14
17. Biham, E., Dunkelman, O., Keller, N., Shamir, A.: New attacks on IDEA with at least 6 rounds. J. Cryptol. **28**(2), 209–239 (2015), https://doi.org/10.1007/s00145-013-9162-9
18. 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)
19. Biryukov, A., Nikolic, I.: Colliding Keys for SC2000-256. In: Joux, A., Youssef, A.M. (eds.) SAC. Lecture Notes in Computer Science, vol. 8781, pp. 77–91. Springer (2014), https://doi.org/10.1007/978-3-319-13051-4_5
20. Bogdanov, A., Rechberger, C.: A 3-Subset Meet-in-the-Middle Attack: Cryptanalysis of the Lightweight Block Cipher KTANTAN. In: SAC. pp. 229–240 (2010), https://doi.org/10.1007/978-3-642-19574-7_16
21. Bouillaguet, C., Derbez, P., Dunkelman, O., Fouque, P., Keller, N., Rijmen, V.: Low-data complexity attacks on AES. IEEE Trans. Inf. Theory **58**(11), 7002–7017 (2012), https://doi.org/10.1109/TIT.2012.2207880
22. Bouillaguet, C., Derbez, P., Fouque, P.: Automatic search of attacks on round-reduced AES and applications. In: Rogaway, P. (ed.) Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings. Lecture Notes in Computer Science, vol. 6841, pp. 169–187. Springer (2011), https://doi.org/10.1007/978-3-642-22792-9_10
23. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y.: Compressing Vector OLE. In: Lie, D., Mannan, M., Backes, M., Wang, X. (eds.) CCS. pp. 896–912. ACM (2018), https://doi.org/10.1145/3243734.3243868
24. Canteaut, A., Naya-Plasencia, M., Vayssière, B.: Sieve-in-the-Middle: Improved MITM Attacks. In: Canetti, R., Garay, J.A. (eds.) CRYPTO I. Lecture Notes in Computer Science, vol. 8042, pp. 222–240. Springer (2013), https://doi.org/10.1007/978-3-642-40041-4_13
25. Chen, M.S., Chen, Y.S., Cheng, C.M., Fu, S., Hong, W.C., Hsiang, J.H., Hu, S.T., Kuo, P.C., Lee, W.B., Liu, F.H., et al.: Preon: zk-snark based signature scheme (2023), https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/Preon-spec-web.pdf

26. Chen, S., Dong, X., Guo, J., Zhang, T.: Chosen-prefix collisions on aes-like hashing. IACR Trans. Symmetric Cryptol. **2024**(4), 64–96 (2024), https://doi.org/10.46586/tosc.v2024.i4.64-96
27. Chen, S., Guo, J., List, E., Shi, D., Zhang, T.: Diving Deep into the Preimage Security of AES-Like Hashing. In: Joye, M., Leander, G. (eds.) EUROCRYPT I. Lecture Notes in Computer Science, vol. 14651, pp. 398–426. Springer (2024), https://doi.org/10.1007/978-3-031-58716-0_14
28. Chen, Y.L., Davidson, M., Dworkin, M., Kang, J., Kelsey, J., Sasaki, Y., Turan, M.S., Chang, D., Mouha, N., Thompson, A.: Proposal of Requirements for an Accordion Mode: Discussion Draft for the NIST Accordion Mode Workshop 2024. Tech. rep., US National Institute for Standards in Technology (2024), https://csrc.nist.gov/pubs/other/2024/04/10/proposal-of-requirements-for-an-accordion-mode-dis/iprd
29. Chen, Y.L., Flórez-Gutiérrez, A., Inoue, A., Ito, R., Iwata, T., Minematsu, K., Mouha, N., Naito, Y., Sibleyras, F., Todo, Y.: Key Committing Security of AEZ and More. IACR Trans. Symmetric Cryptol. **2023**(4), 452–488 (2023), https://doi.org/10.46586/tosc.v2023.i4.452-488
30. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Information Security and Cryptography, Springer (2002), https://doi.org/10.1007/978-3-662-04722-4
31. Derbez, P.: Meet-in-the-middle attacks on AES. Ph.D. thesis, Ecole Normale Supérieure de Paris-ENS Paris (2013)
32. Diffie, W., Hellman, M.E.: Special feature exhaustive cryptanalysis of the NBS data encryption standard. IEEE Computer **10**(6), 74–84 (1977), https://doi.org/10.1109/C-M.1977.217750
33. Ding, C., Huang, Y.: Dubhe: Succinct zero-knowledge proofs for standard AES and related applications. In: Calandrino, J.A., Troncoso, C. (eds.) 32nd USENIX Security Symposium, USENIX Security 2023, Anaheim, CA, USA, August 9-11, 2023. pp. 4373–4390. USENIX Association (2023), https://www.usenix.org/conference/usenixsecurity23/presentation/ding-changchang
34. Ding, C., Huang, Y.: Phecda: Post-Quantum Transparent zkSNARKs from Improved Polynomial Commitment and VOLE-in-the-Head with Application in Publicly Verifiable AES . In: 2025 IEEE Symposium on Security and Privacy (SP). pp. 55–55. IEEE Computer Society, Los Alamitos, CA, USA (May 2025), https://doi.ieeecomputersociety.org/10.1109/SP61157.2025.00055
35. Dobraunig, C., Kales, D., Rechberger, C., Schofnegger, M., Zaverucha, G.: Shorter signatures based on tailor-made minimalist symmetric-key crypto. In: Yin, H., Stavrou, A., Cremers, C., Shi, E. (eds.) Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022. pp. 843–857. ACM (2022), https://doi.org/10.1145/3548606.3559353
36. Dodis, Y., Grubbs, P., Ristenpart, T., Woodage, J.: Fast Message Franking: From Invisible Salamanders to Encryptment. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO I. Lecture Notes in Computer Science, vol. 10991, pp. 155–186. Springer (2018), https://doi.org/10.1007/978-3-319-96884-1_6
37. Dong, X., Guo, J., Li, S., Pham, P.: Triangulating Rebound Attack on AES-like Hashing. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO I. Lecture Notes in Computer Science, vol. 13507, pp. 94–124. Springer (2022), https://doi.org/10.1007/978-3-031-15802-5_4

38. Dong, X., Hua, J., Sun, S., Li, Z., Wang, X., Hu, L.: Meet-in-the-Middle Attacks Revisited: Key-Recovery, Collision, and Preimage Attacks. In: CRYPTO III. pp. 278–308 (2021), https://doi.org/10.1007/978-3-030-84252-9_10
39. Dong, X., Li, S., Pham, P.: Chosen-key distinguishing attacks on full aes-192, aes-256, kiasu-bc, and more. IACR Cryptol. ePrint Arch. p. 1095 (2023), https://eprint.iacr.org/2023/1095
40. Dong, X., Zhao, B., Qin, L., Hou, Q., Zhang, S., Wang, X.: Generic mitm attack frameworks on sponge constructions. In: Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part IV. pp. 3–37 (2024), https://doi.org/10.1007/978-3-031-68385-5_1
41. Dunkelman, O., Keller, N., Shamir, A.: Minimalism in cryptography: The even-mansour scheme revisited. In: Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings. pp. 336–354 (2012), https://doi.org/10.1007/978-3-642-29011-4_21
42. Gilbert, H., Peyrin, T.: Super-Sbox Cryptanalysis: Improved Attacks for AES-Like Permutations. In: FSE. pp. 365–383 (2010), https://doi.org/10.1007/978-3-642-13858-4_21
43. Grubbs, P., Lu, J., Ristenpart, T.: Message Franking via Committing Authenticated Encryption. In: Katz, J., Shacham, H. (eds.) CRYPTO III. Lecture Notes in Computer Science, vol. 10403, pp. 66–97. Springer (2017), https://doi.org/10.1007/978-3-319-63697-9_3
44. Guo, J., Ling, S., Rechberger, C., Wang, H.: Advanced Meet-in-the-Middle Preimage Attacks: First Results on Full Tiger, and Improved Results on MD4 and SHA-2. In: ASIACRYPT. pp. 56–75 (2010), https://doi.org/10.1007/978-3-642-17373-8_4
45. Hosoyamada, A., Sasaki, Y.: Finding Hash Collisions with Quantum Computers by Using Differential Trails with Smaller Probability than Birthday Bound. In: EUROCRYPT II. pp. 249–279 (2020), https://doi.org/10.1007/978-3-030-45724-2_9
46. Hou, Q., Dong, X., Qin, L., Zhang, G., Wang, X.: Automated Meet-in-the-Middle Attack Goes to Feistel. In: Guo, J., Steinfeld, R. (eds.) ASIACRYPT III. Lecture Notes in Computer Science, vol. 14440, pp. 370–404. Springer (2023), https://doi.org/10.1007/978-981-99-8727-6_13
47. Hua, J., Dong, X., Sun, S., Zhang, Z., Hu, L., Wang, X.: Improved MITM cryptanalysis on streebog. IACR Trans. Symmetric Cryptol. **2022**(2), 63–91 (2022), https://doi.org/10.46586/tosc.v2022.i2.63-91
48. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-Knowledge Proofs from Secure Multiparty Computation. SIAM J. Comput. **39**(3), 1121–1152 (2009), https://doi.org/10.1137/080725398
49. Isobe, T.: A single-key attack on the full GOST block cipher. J. Cryptol. **26**(1), 172–189 (2013), https://doi.org/10.1007/s00145-012-9118-5
50. Isobe, T., Shibutani, K.: Security analysis of the lightweight block ciphers xtea, LED and piccolo. In: Information Security and Privacy - 17th Australasian Conference, ACISP 2012, Wollongong, NSW, Australia, July 9-11, 2012. Proceedings. pp. 71–86 (2012), https://doi.org/10.1007/978-3-642-31448-3_6
51. ISO/IEC: ISO/IEC 10118-2: 2018. IT Security techniques - Hash-functions - Part 2: Hash-functions using an n-bit block cipher (2010)

52. Kelsey, J., Schneier, B., Wagner, D.A.: Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES. In: Koblitz, N. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 1109, pp. 237–251. Springer (1996), https://doi.org/10.1007/3-540-68697-5_19
53. Khovratovich, D., Rechberger, C., Savelieva, A.: Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 Family. In: Canteaut, A. (ed.) FSE. Lecture Notes in Computer Science, vol. 7549, pp. 244–263. Springer (2012), https://doi.org/10.1007/978-3-642-34047-5_15
54. Lamberger, M., Mendel, F., Rechberger, C., Rijmen, V., Schläffer, M.: Rebound Distinguishers: Results on the Full Whirlpool Compression Function. In: ASIACRYPT. pp. 126–143 (2009), https://doi.org/10.1007/978-3-642-10366-7_8
55. Len, J., Grubbs, P., Ristenpart, T.: Partitioning Oracle Attacks. In: Bailey, M.D., Greenstadt, R. (eds.) USENIX S&P. pp. 195–212. USENIX Association (2021), https://www.usenix.org/conference/usenixsecurity21/presentation/len
56. Li, J., Isobe, T., Shibutani, K.: Converting meet-in-the-middle preimage attack into pseudo collision attack: Application to SHA-2. In: Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers. pp. 264–286 (2012), https://doi.org/10.1007/978-3-642-34047-5_16
57. Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl. In: Dunkelman, O. (ed.) FSE. Lecture Notes in Computer Science, vol. 5665, pp. 260–276. Springer (2009), https://doi.org/10.1007/978-3-642-03317-9_16
58. Ni, J., Li, Y., Liu, F., Wang, G.: Practical key collision on AES and kiasu-bc. IACR Cryptol. ePrint Arch. p. 462 (2025), https://eprint.iacr.org/2025/462
59. NIST: Advanced Encryption Standard (AES). Federal Information Processing Standards (NIST FIPS), National Institute of Standards and Technology (2001)
60. Preneel, B., Govaerts, R., Vandewalle, J.: Hash Functions Based on Block Ciphers: A Synthetic Approach. In: CRYPTO. pp. 368–378 (1993), https://doi.org/10.1007/3-540-48329-2_31
61. Qin, L., Hua, J., Dong, X., Yan, H., Wang, X.: Meet-in-the-Middle Preimage Attacks on Sponge-Based Hashing. In: EUROCRYPT IV. pp. 158–188 (2023), https://doi.org/10.1007/978-3-031-30634-1_6
62. Robshaw, M.: A cryptographic review of Cipherunicorn-A (14 December 2001), https://www.cryptrec.go.jp/exreport/cryptrec-ex-1031-2001.pdf
63. Roy, L.: Softspokenot: Quieter OT extension from small-field silent VOLE in the minicrypt model. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13507, pp. 657–687. Springer (2022), https://doi.org/10.1007/978-3-031-15802-5_23
64. de Saint Guilhem, C.D., Meyer, L.D., Orsini, E., Smart, N.P.: BBQ: using AES in picnic signatures. In: Paterson, K.G., Stebila, D. (eds.) Selected Areas in Cryptography - SAC 2019 - 26th International Conference, Waterloo, ON, Canada, August 12-16, 2019, Revised Selected Papers. Lecture Notes in Computer Science, vol. 11959, pp. 669–692. Springer (2019), https://doi.org/10.1007/978-3-030-38471-5_27
65. de Saint Guilhem, C.D., Orsini, E., Tanguy, T.: Limbo: Efficient zero-knowledge mpcith-based arguments. In: Kim, Y., Kim, J., Vigna, G., Shi, E. (eds.) CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security,

Virtual Event, Republic of Korea, November 15 - 19, 2021. pp. 3022–3036. ACM (2021), https://doi.org/10.1145/3460120.3484595
66. Sasaki, Y.: Meet-in-the-Middle Preimage Attacks on AES Hashing Modes and an Application to Whirlpool. In: Joux, A. (ed.) FSE. Lecture Notes in Computer Science, vol. 6733, pp. 378–396. Springer (2011), https://doi.org/10.1007/978-3-642-21702-9_22
67. Sasaki, Y.: Integer linear programming for three-subset meet-in-the-middle attacks: Application to GIFT. In: Advances in Information and Computer Security - 13th International Workshop on Security, IWSEC 2018, Sendai, Japan, September 3-5, 2018, Proceedings. pp. 227–243 (2018), https://doi.org/10.1007/978-3-319-97916-8_15
68. Sasaki, Y., Aoki, K.: Preimage Attacks on 3, 4, and 5-Pass HAVAL. In: Pieprzyk, J. (ed.) ASIACRYPT. Lecture Notes in Computer Science, vol. 5350, pp. 253–271. Springer (2008), https://doi.org/10.1007/978-3-540-89255-7_16
69. Sasaki, Y., Aoki, K.: Finding Preimages in Full MD5 Faster Than Exhaustive Search. In: Joux, A. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 5479, pp. 134–152. Springer (2009), https://doi.org/10.1007/978-3-642-01001-9_8
70. Sasaki, Y., Wang, L., Sakai, Y., Sakiyama, K., Ohta, K.: Three-subset meet-in-the-middle attack on reduced XTEA. In: Progress in Cryptology - AFRICACRYPT 2012 - 5th International Conference on Cryptology in Africa, Ifrance, Morocco, July 10-12, 2012. Proceedings. pp. 138–154 (2012), https://doi.org/10.1007/978-3-642-31410-0_9
71. Sasaki, Y., Wang, L., Wu, S., Wu, W.: Investigating Fundamental Security Requirements on Whirlpool: Improved Preimage and Collision Attacks. In: ASIACRYPT. pp. 562–579 (2012), https://doi.org/10.1007/978-3-642-34961-4_34
72. Schrottenloher, A., Stevens, M.: Simplified MITM Modeling for Permutations: New (Quantum) Attacks. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO III. Lecture Notes in Computer Science, vol. 13509, pp. 717–747. Springer (2022), https://doi.org/10.1007/978-3-031-15982-4_24
73. Schrottenloher, A., Stevens, M.: Simplified Modeling of MITM Attacks for Block Ciphers: New (Quantum) Attacks. IACR Trans. Symmetric Cryptol. **2023**(3), 146–183 (2023), https://doi.org/10.46586/tosc.v2023.i3.146-183
74. Taiyama, K., Sakamoto, K., Ito, R., Taka, K., Isobe, T.: Key Collisions on AES and Its Applications. In: Chung, K., Sasaki, Y. (eds.) ASIACRYPT VII. Lecture Notes in Computer Science, vol. 15490, pp. 267–300. Springer (2024), https://doi.org/10.1007/978-981-96-0941-3_9
75. Takahashi, A., Zaverucha, G.: Verifiable encryption from mpc-in-the-head. IACR Commun. Cryptol. **1**(1), 3 (2024), https://doi.org/10.62056/a3wa3zl7s

Supplementary Material

Appendix A MITM Collision Attacks on Rijndael-MMO/MP

A.1 New MITM Collision Attack on 8-Round Rijndael-192-MMO/MP

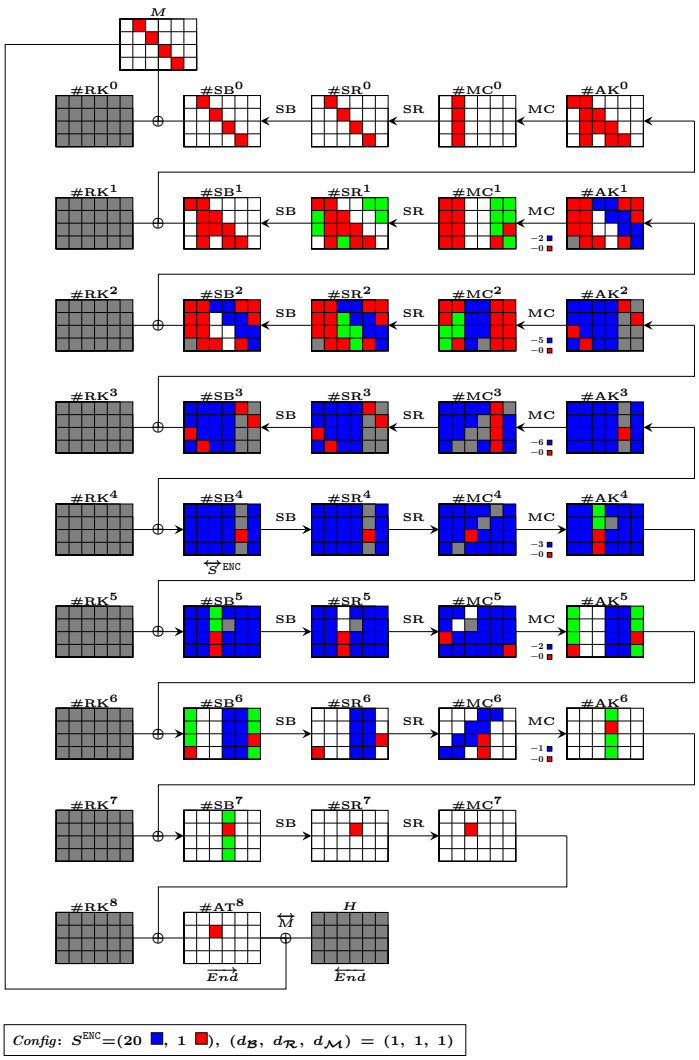


Fig. 10. Configuration of MITM collision attack on 8-round Rijndael-192-MMO/MP (the cost-related constants at $\#\mathrm{MC}^1[19, 22]$, $\#\mathrm{AK}^5[22]$, $\#\mathrm{AK}^6[13]$ are chosen as indexing constants g_{ic} and the rest constants are enumerating constants g_{ec}).

A.2 New MITM Collision Attack on 9-Round Rijndael-256-MMO/MP

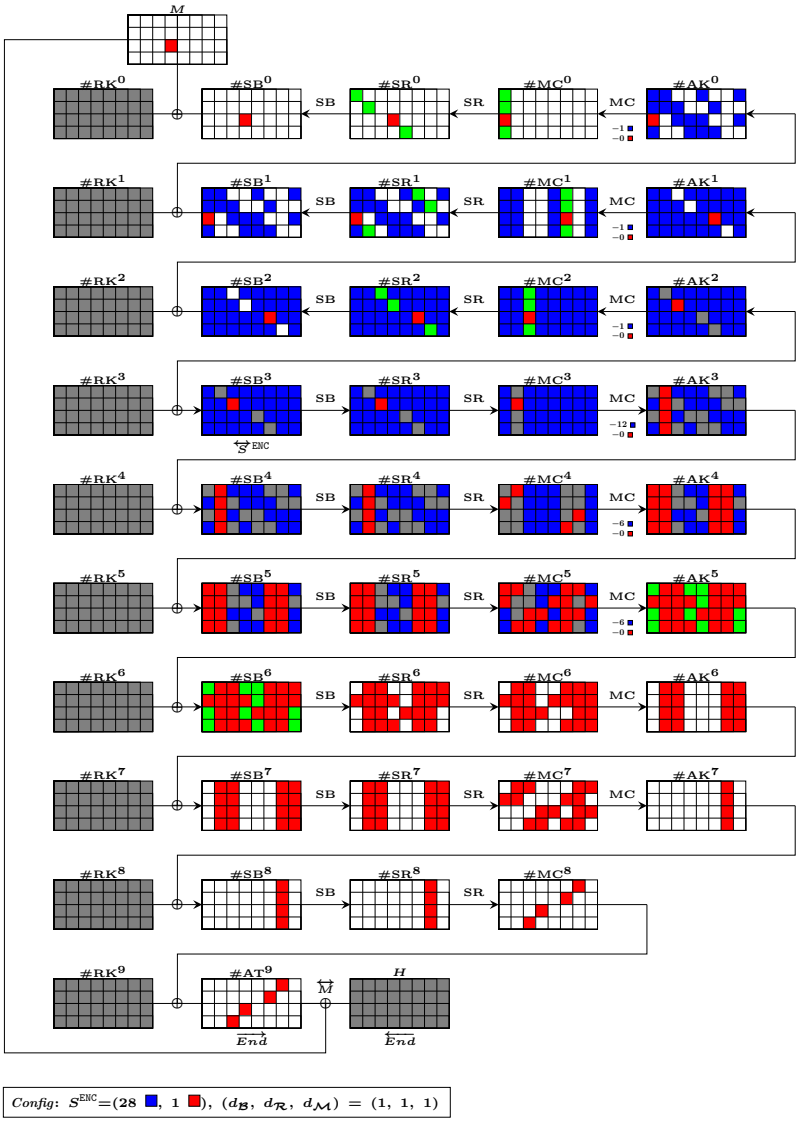


Fig. 11. Configuration of MITM collision attack on 9-round Rijndael-256-MMO/MP (the cost-related constants at $\#\mathrm{MC}^0[2]$, $\#\mathrm{MC}^1[22]$, $\#\mathrm{MC}^2[10]$ are chosen as indexing constants g_{ic} and the rest constants are enumerating constants g_{ec}).

Appendix B MITM Key Collision Attacks on AES-DM

B.1 New MITM Key Collision Attack on 4-Round Rijndael-192-DM

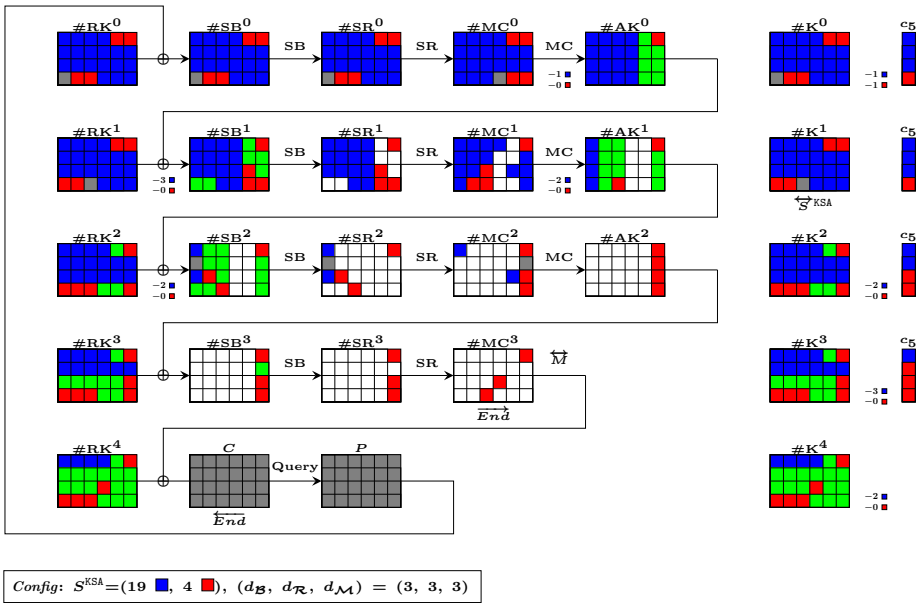


Fig. 12. Configuration of new key collision attack on 4-Round Rijndael-192-DM (the cost-related constants at $\#\mathrm{SB}^2[1,6]$ are chosen as indexing constants g_{ic} and the rest constants are enumerating constants g_{ec}).

B.2 New MITM Key Collision Attack on 5-Round Rijndael-256-DM

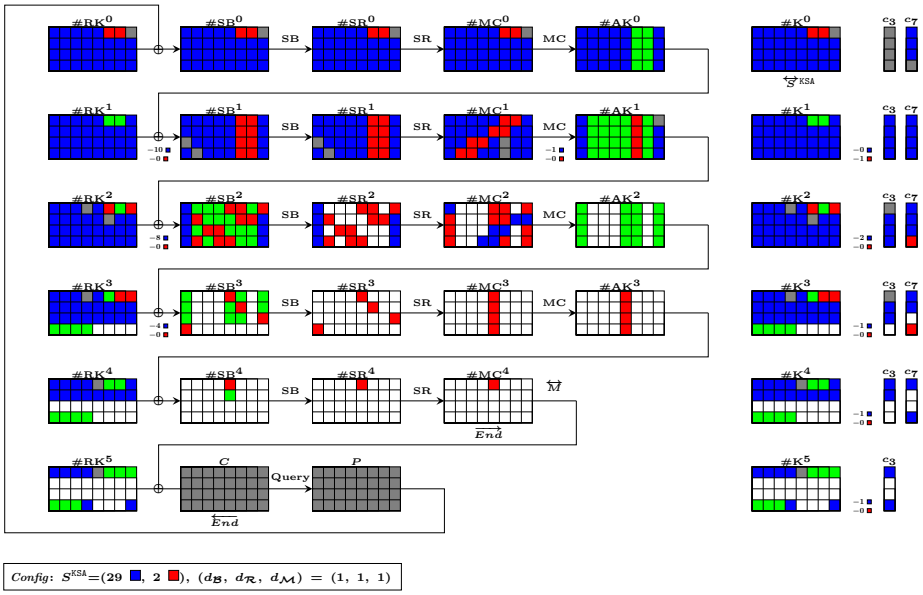


Fig. 13. Configuration of new key collision attack on 5-Round Rijndael-256-DM (the cost-related constants at $\#\mathrm{SB}^2[5, 7, 10, 14, 15, 25]$, $\#\mathrm{SB}^3[3, 16, 21, 30]$ are chosen as indexing constants g_{ic} and the rest constants are enumerating constants g_{ec}).

Appendix C SKP Key Recovery Attacks on AES and Rijndael

C.1 Improved SKP Key Recovery Attack on 4*-Round AES-128

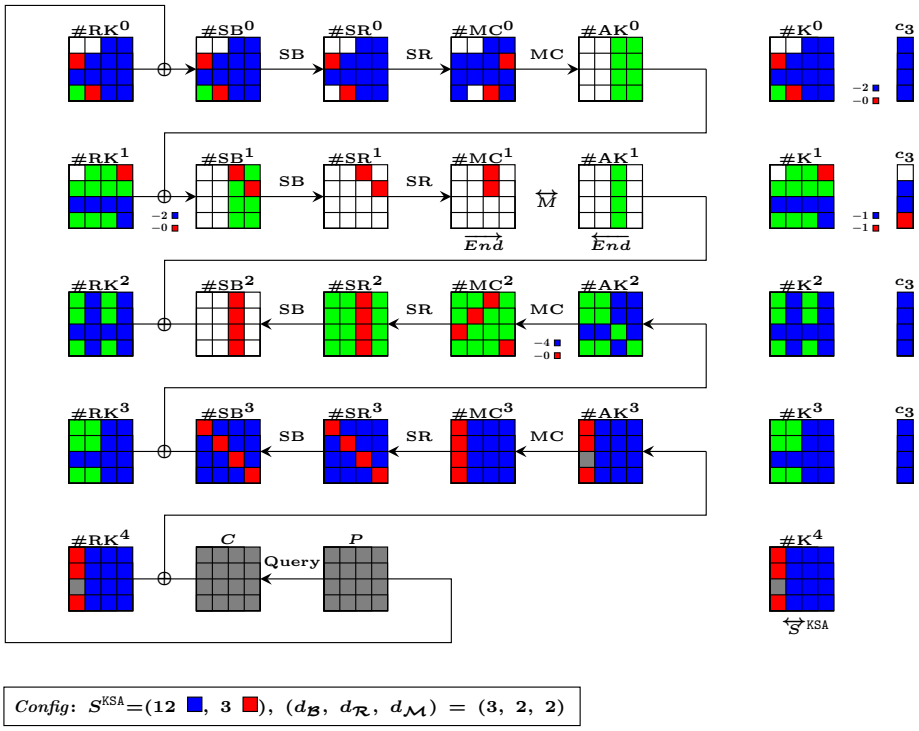


Fig. 14. Configuration of improved SKP key recovery attack on 4 full rounds AES-128 (as $d_{\mathcal{R}} = d_{\mathcal{M}} = 2$, we can consume one more byte blue DoF, *i.e.*, choosing one more blue byte in $\#\mathrm{K}^4$ as the enumerating constant and thus let $d_{\mathcal{B}} = 2$, then the memory complexity is dominated by the blue neutral words with 2^{48}, which are indexed by $\#\mathrm{MC}^2[2, 5, 8, 15]$).

C.2 New SKP Key Recovery Attack on 7-Round AES-192.

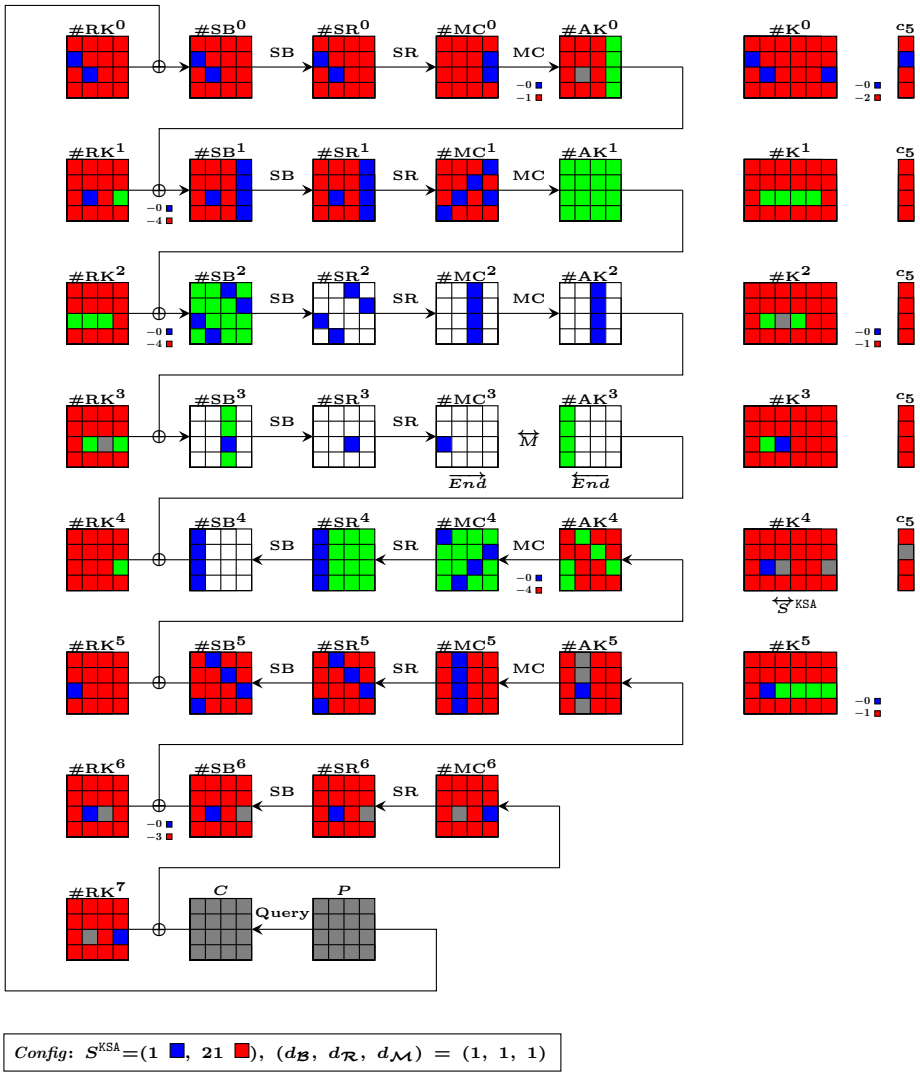
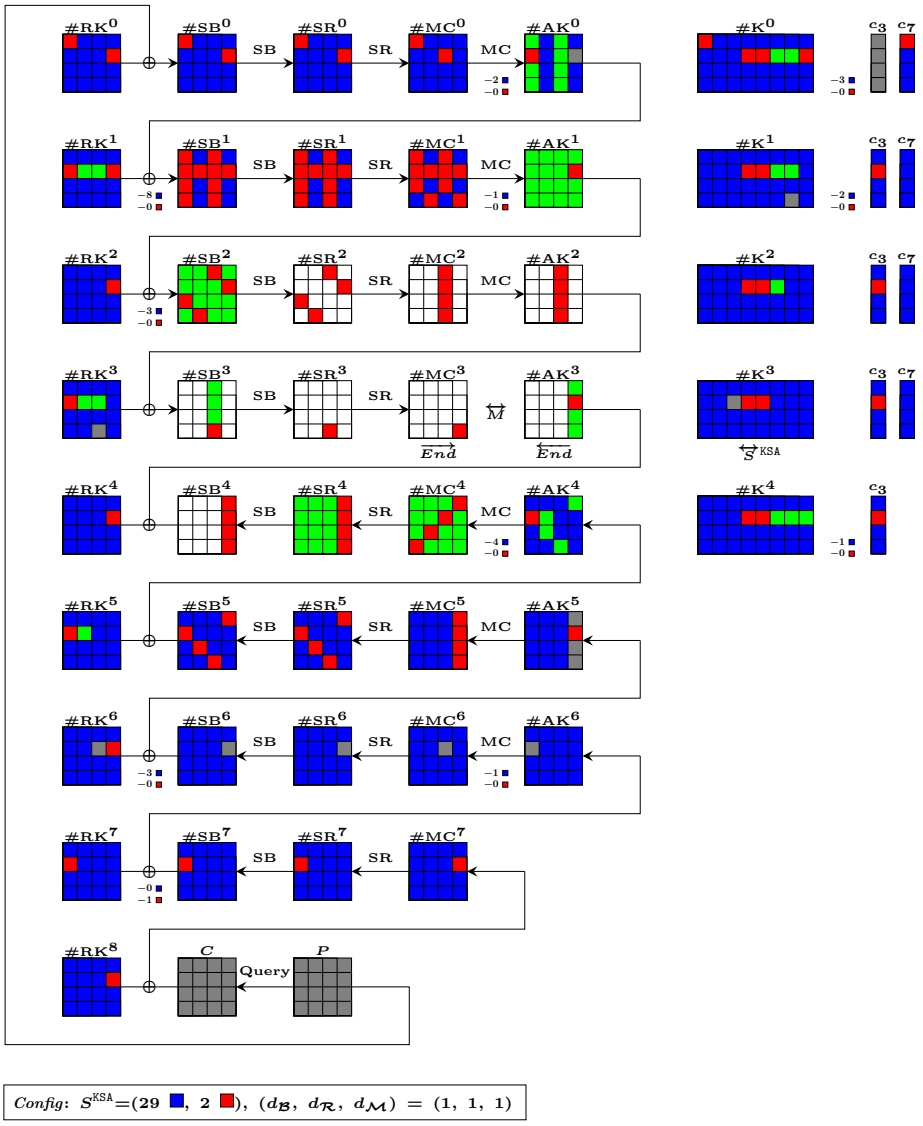


Fig. 15. Configuration of new SKP key recovery attack on 7 rounds AES-192 (the cost-related constants at $\#\mathrm{SB}^2[8]$, $\#\mathrm{AK}^5[4,5,7]$, $\#\mathrm{MC}^4[0,7,10,13]$ are chosen as indexing constants g_{ic} to store blue neutral words and the rest constants are enumerating constants g_{ec}).

C.3 New SKP Key Recovery Attack on 8-Round AES-256.



Config: S^{KSA}=(29 ■, 2 ■), $(d_{\mathcal{B}}, d_{\mathcal{R}}, d_{\mathcal{M}}) = (1, 1, 1)$

Fig. 16. Configuration of new SKP key recovery attack on 8 rounds AES-256 (the memory complexity is dominated by the blue neutral words with 2^{72}, which are indexed by $\#\text{MC}^4[3, 6, 9, 12]$, $\#\text{AK}^5[12, 14, 15]$, $\#\text{MC}^6[10]$).

C.4 New SKP Key Recovery Attack on 6-Round Rijndael-192.

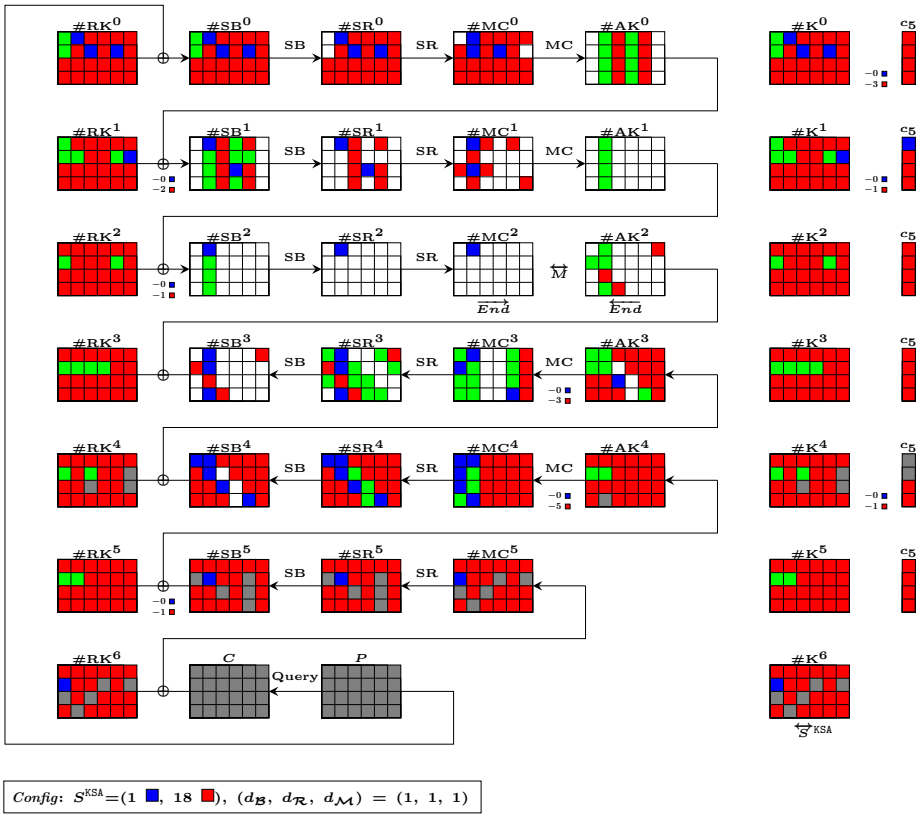


Fig. 17. Configuration of new SKP key recovery attack on 6 rounds Rijndael-192 (the memory complexity is dominated by the red neutral words with 2^{32}, which are indexed by $\#\mathrm{SB}^2[4]$, $\#\mathrm{MC}^3[1, 19]$).

C.5 New SKP Key Recovery Attack on 7-Round Rijndael-256.

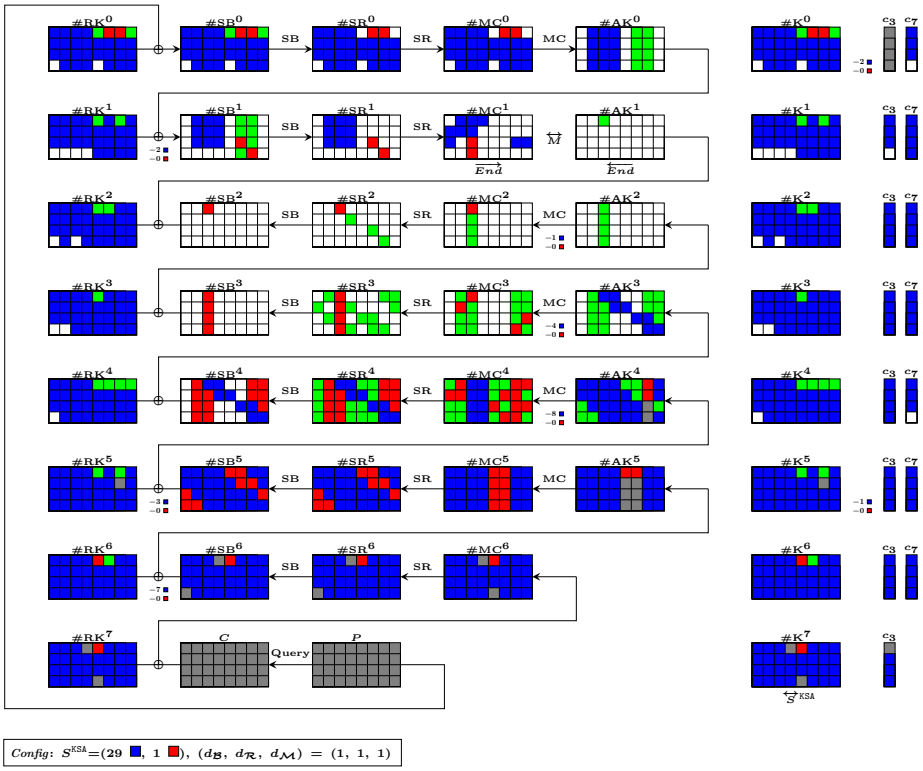


Fig. 18. Configuration of new SKP key recovery attack on 7 rounds Rijndael-256 (the memory complexity is dominated by the blue neutral words with 2^{96}, which are indexed by $\#\mathrm{MC}^2[8]$, $\#\mathrm{MC}^3[5, 8, 27, 30]$, $\#\mathrm{MC}^4[1, 4, 5, 18, 28, 30]$).

Appendix D Fixed-Key MITM Preimage Attacks on AES/Rijndael-EM

D.1 Improved Fixed-Key Preimage Attack on 7-Round AES-EM

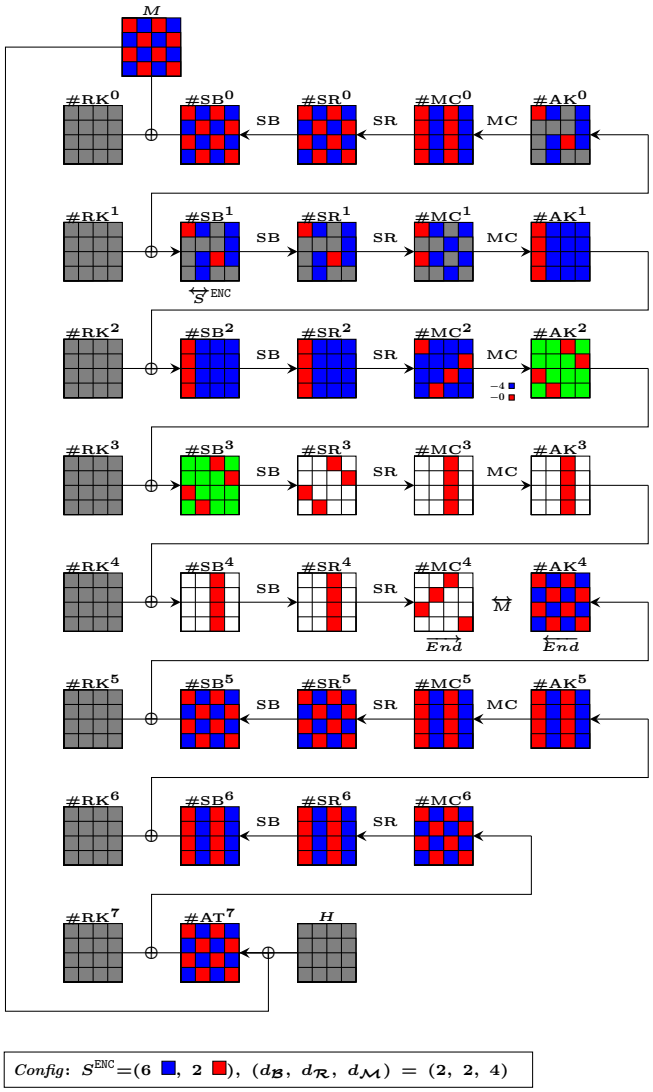


Fig. 19. Configuration of improved MITM fixed-key preimage attack on 7 rounds AES-EM (the memory complexity is dominated by the blue neutral words with 2^{32}, which are indexed by $\#\mathrm{MC}^1[1,3]$).

D.2 New Fixed-Key Preimage Attack on 8-Round Rijndael-192-EM

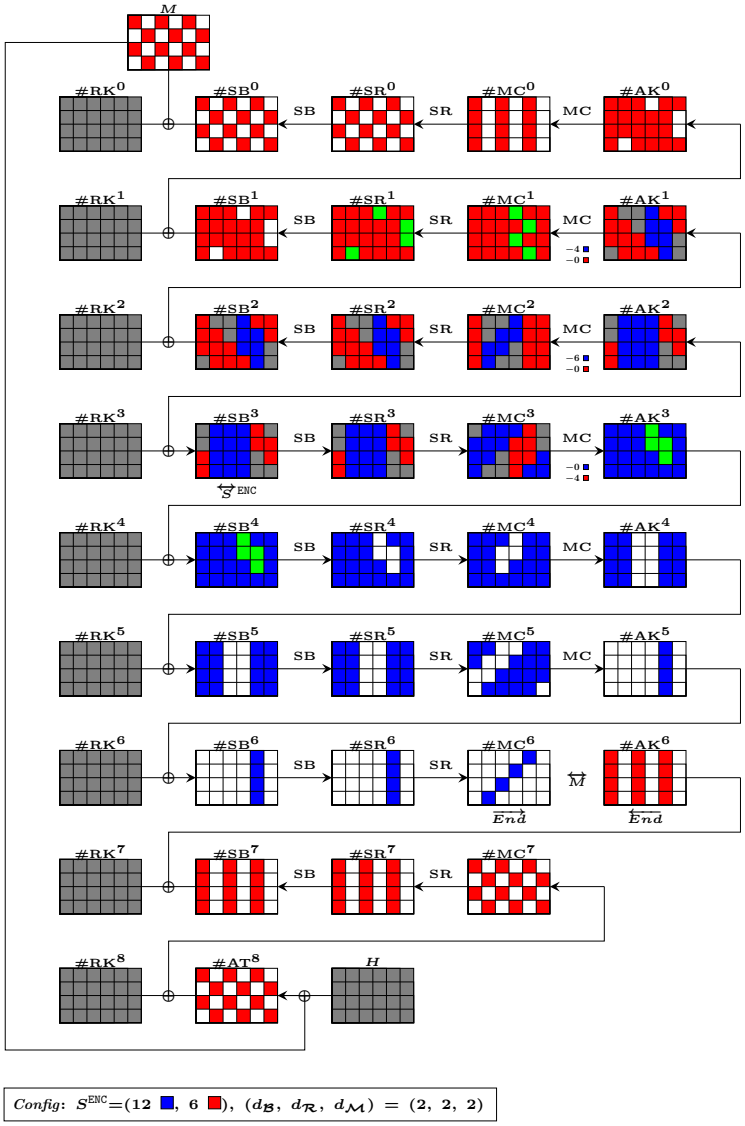


Fig. 20. Configuration of new MITM fixed-key preimage attack on 8 rounds Rijndael-192-EM (the memory complexity is dominated by the blue neutral words with 2^{16} in $\#\mathrm{AK}^1$).