Triangulating Meet-in-the-Middle Attack (Full Version)

Boxin Zhao[1], Qingliang Hou[3], Lingyue Qin[2,1,4], and Xiaoyang Dong[2,1,4](✉)

[1] Zhongguancun Laboratory, Beijing, P.R.China
zhaobx@mail.zgclab.edu.cn
[2] Tsinghua University, Beijing, P.R.China
{qinly,xiaoyangdong}@tsinghua.edu.cn
[3] School of Cyber Science and Technology, Shandong University, Qingdao, P.R.China
qinglianghou@mail.sdu.edu.cn
[4] State Key Laboratory of Cryptography and Digital Economy Security, Tsinghua University, Beijing, P.R.China

Abstract. To penetrate more rounds with Meet-in-the-Middle (MitM) attack, the neutral words are usually subject to some linear constraints, *e.g.*, Sasaki and Aoki's *initial structure* technique. At CRYPTO 2021, Dong *et al.* found the neutral words can be nonlinearly constrained. They introduced a table-based method to precompute and store the solution space of the neutral words, which led to a huge memory complexity. In this paper, we find some nonlinearly constrained neutral words can be solved efficiently by Khovratovich *et al.*'s triangulation algorithm (TA). Furthermore, motivated by the structured Gaussian elimination paradigm developed by LaMacchia *et al.* [37] and Bender *et al.* [6], we improve the TA to deal with the case when there are still many unprocessed equations, but no variable exists in only one equation (the original TA will terminate). Then, we introduce the new MitM attack based on our improved TA, called *triangulating MitM attack*.

As applications, the memory complexities of the single-plaintext key-recovery attacks on 4-/5-round AES-128 are significantly reduced from 2^{80} to the practical 2^{24} or from 2^{96} to 2^{40}. Besides, a series of new one/two-plaintext attacks are proposed for reduced AES-192/-256 and Rijndael-EM (basic primitives of NIST PQC candidate FAEST). A partial key-recovery experiment is conducted on 4-round AES-128 to verify the correctness of our technique. For AES-256-DM, the memory complexity of the 10-round preimage attack is reduced from 2^{56} to 2^{8}, thus an experiment is also implemented. Without our technique, the impractical memories 2^{80} or 2^{56} of previous attacks in the precomputation phase will always prevent any kind of (partial) experimental simulations.

In the full version, we extend our techniques to Sponge functions. We figure out some memory efficient attacks, *e.g.*, reducing the memories of Qin *et al.*'s 4-round attack on Keccak[1024] and Dong *et al.*'s 3-round attack on Xoodyak-XOF from 2^{108} to 2^{52}, or from 2^{118} to the current 2^{37}. Besides, the first 3-round collision attack on Xoodyak-XOF with 128-bit target is given, while previous MitM approaches are always worse than the birthday bound due to the high memory. The memories of the MitM attacks on 3-/4-round Ascon are reduced from $2^{24}/2^{34}$ to $2^{14}/2^{14}$, which have been partially implemented to verify the attacks.

Keywords: AES · Triangulating MitM · Key-recovery · Hash Function · Triangulation Algorithm

1 Introduction

The Rijndael block cipher [17] was designed by Daemen and Rijmen in 1997 and accepted by NIST as the AES (Advanced Encryption Standard) standard in October 2000. Today, it is probably the most widely used block cipher. In 2024, NIST announced the 2nd round candidates for the contest of additional digital signature schemes for the NIST PQC, including FAEST [5]. In FAEST, the secret signing key is an AES key, while the public verification key is one plaintext-ciphertext pair for FAEST based on AES-128, and two plaintext-ciphertext pairs for FAEST based on AES-192 and AES-256. The plaintext-ciphertext pairs are obtained by encrypting some random messages with AES under the signing key. Therefore, the security of FAEST is reduced to the security of AES with one or two known plaintext-ciphertext pairs. Therefore, it is important to study the security of AES in this scenario. In fact, attacks on AES with data complexity restricted to only a few known or chosen plaintexts have been studied extensively [11,10,19,54,47,4,28]. Among these attacks, the single plaintext-ciphertext attacks are based on the Meet-in-the-Middle (MitM) approach or Guess-and-Determine [11].

The MitM attack proposed by Diffie and Hellman in 1977 is a time-memory trade-off cryptanalysis of symmetric-key primitives [21]. Currently, the MitM attack has been successfully applied to block ciphers and hash functions with more sophisticated techniques, such as the internal state guessing [29], splice-and-cut [1], initial structure [50], bicliques [8,35], 3-subset MitM [9], (indirect) partial matching [1,50], guess-and-determine [51,33], sieve-in-the-middle [13], match-box [31], dissection [23], non-linear initial structure [32], MitM in differential view [36,30], and differential MitM [12], etc. Automating MitM attack was first reported in CRYPTO 2011 and 2016 [11,20], which present attacks on AES with low data complexity, or even a single plaintext-ciphertext pair. In 2018, Sasaki [48] first tried to automate MitM with Mixed Integer Linear Programming (MILP). At EUROCRYPT 2021, Bao *et al.* [2] built a fully automated MitM preimage attack using MILP on AES-like hashing. Later, the automated MitM models were improved with more techniques by Dong *et al.* [26], Bao *et al.* [3], and Chen *et al.* [14], or further developed for the sponge functions by Schrottenloher and Stevens [52,53], Qin *et al.* [45], and Dong *et al.* [27].

In the MitM attacks, as shown in Figure 1, the iterative round-based computation of the compression function or block cipher is divided at a certain round (starting point) into two chunks. The two chunks are computed independently and end at a common matching point. In both chunks, the computation involves different message words, denoted by N^+ and N^- respectively. So one chunk computes all possible values of the involved message words N^+ independently of the message words N^- involved in the other chunk. The different words N^+ and N^- are called the *neutral words*. At EUROCRYPT 2009, Sasaki and Aoki proposed the *initial structure* (IS) technique with the purpose of skipping several rounds at the beginning of two chunks to enhance the MitM attack [50]. As shown in Figure 1, the two chunks are in the opposite direction, and only a few consecutive starting rounds are overlapped, which form the so-called IS. Although the two sets of neutral words N^+ and N^- appear simultaneously at these rounds, they are only involved in the computation of one chunk each. This is achieved by assigning some linear constraints to the values of neutral words of one chunk, such that different values lead to constant impact on the computation of the opposite chunk. The constrained space of the values of the neutral words is derived by solving the linear systems via Gaussian elimination. At CRYPTO

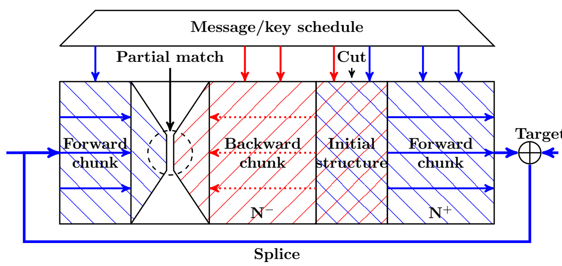

Fig. 1: The Splice-and-cut MitM Attack [1]

2021, Dong *et al.* [26] noticed that in many potential MitM characteristics, the two sets of neutral words N^+ and N^- are constrained by some nonlinear constraints, such that different values lead to a constant impact on the computation of the opposite chunk only if the nonlinear system holds. Therefore, Dong *et al.* presented a table-based technique to obtain the constrained space of the values of the neutral words. The drawback of Dong *et al.*'s approach is that it requires a huge amount of memory to prepare two hash tables, and many attacks based on this method need huge memory, *e.g.*, [26,45].

Our Contribution. At CT-RSA 2009, Khovratovich, Biryukov, and Nikolic presented the triangulation algorithm (TA) [34] to solve the nonlinear system. Combining TA and rebound attack [43], Dong *et al.* proposed the triangulating rebound attack [25].

In this paper, motivated by the structured Gaussian elimination paradigm developed by LaMacchia *et al.* [37] and Bender *et al.* [6], we improve the triangulation algorithm to deal with the case where there are still many unprocessed equations but no variable exists in only one equation (the original TA will terminate immediately). Moreover, with the help of the improved triangulation algorithm, we find a memory-efficient approach to derive the value space of the nonlinear constrained neutral words for MitM, thereby solving the memory problem of Dong *et al.*'s MitM attacks [26]. We name this new method as the *triangulating MitM attack*. With this advanced method in hand, we achieve the following results:

- **Improved key-recovery attacks on AES with single/two plaintext-ciphertext pairs:** This setting directly impacts the security of NIST PQC candidate FAEST [5], where only one plaintext-ciphertext pair acts as the public key in FAEST-128 (based on AES-128), and two plaintext-ciphertext pairs act as the public key in FAEST-192/-256 (based on AES-192/-256). The encryption key of AES acts as secret signing key in FAEST. Our goal is to recover the secret signing key with the public key, *i.e.*, single/two plaintext-ciphertext pairs. Once the secret signing key is recovered, a forgery attack on FAEST is found. The cryptanalysis records on up to 5-round AES in this setting are kept by Bouillaguet, Derbez, and Fouque from CRYPTO 2011 [11] and Derbez's PhD thesis [19]. We break their 10+ year record by significantly reducing the memory complexities of both the 4-round and 5-round attacks on AES-128 by a factor of 2^{56}, *i.e.*, from 2^{80} to the practical 2^{24} and from 2^{96} to 2^{40}. Due to the practical memory, the new 4-round attack has been practically verified by a 4-byte partial key-recovery experiment in Sect. 4.2. With the help of Leurent and Pernot's new representation of AES's key schedule [39], we also improve both the time and memory complexities of the 4-round key-recovery attack on AES-128 and also propose the attacks on 6-round AES-192 and 7-round AES-256 with two plaintext-ciphertext pairs.
- **Key-recovery attacks on Rijndael-EM with one plaintext-ciphertext pair:** The high-performance versions of FAEST are based on Rijndael-EM [5]. We first convert the key-recovery attacks on Rijndael-EM into the preimage attacks on its hashing mode. By applying the triangulating MitM attack, we find the preimage attacks and then convert them back to key-recovery attacks with one plaintext-ciphertext pair on 7-/8-/9-round Rijndael-EM-128/192/256, respectively.
- **DM Hashing mode with AES-256:** The memory complexity of the preimage attack on 10-round AES-256-DM is reduced from the impractical 2^{56} [26] to the practical 2^{8}. Therefore, an experiment is performed to find a 40-bit partial target preimage to verify our technique in Sect. 4.6. Without our improvement, the impractical memory of size 2^{56} in the precomputation will prevent any (partial) experiments.
- **Applications to the MitM attacks on Sponge functions:**
  - We improve the 4-round preimage attacks on Keccak[1024] and also give the first MitM preimage attack on 4-round Keccak[768]. Compared to Qin *et al.*'s 4-round attack on Keccak[1024] [46], the memory complexity is significantly reduced from 2^{108} to 2^{52} with the same time complexity.
  - For Xoodyak-XOF, we significantly reduce the memory of Dong *et al.*'s 3-round preimage attack [27] from 2^{118} to the current 2^{37}. Then, the first 3-round collision attack on Xoodyak-XOF with 128-bit target is introduced. Previously, Dong *et al.*'s MitM attack [27] needs to prepare a huge hash

table of size 2^{118} before the MitM phase, which is already worse than the birthday bound 2^{64}, and cannot be converted into collision attack.

- We also improve the attacks on reduced Ascon-XOF with 128-bit target, Subterranean 2.0, and Gimli-XOF with 128-bit target. The experiments of the partial attacks on 3-/4-round Ascon-XOF are conducted in Supplementary Material H to verify the technique.

All our codes including the experiments on 4-round AES-128, 10-round AES-256-DM, and 3-/4-round Ascon-XOF are given at

https://github.com/boxindev/Triangulation-MitM

The summary of key recovery attacks on AES and Rijndael-EM is given in Table 1. The summary of the results on the hash functions is given in Table 2.

Table 1: Key-recovery attacks on AES and Rijndael-EM with low data. KP: known plaintext; CP: Chosen plaintext; ACC: Adaptive chosen plaintext and ciphertext.

Target	Methods	Rounds	Data	Time	Memory	Generic	Ref.
AES-128	MitM	$3^\dagger/10$	1KP	2^{96}	2^{72}	2^{128}	[11]
	MitM	$4^\dagger/10$	1KP	2^{120}	2^{80}	2^{128}	[11]
	MitM	$4^\dagger/10$	1KP	2^{120}	$\mathbf{2^{24}}$	2^{128}	Sect. 4.2
	MitM	$4^\dagger/10$	1KP	$\mathbf{2^{112}}$	$\mathbf{2^{56}}$	2^{128}	Sect. 4.3
	MitM	5/10	1KP	2^{120}	2^{96}	2^{128}	[19]
	MitM	5/10	1KP	2^{120}	$\mathbf{2^{40}}$	2^{128}	Sect. 4.1
	MitM	$4^\dagger/10$	2CP	2^{80}	2^{80}	2^{128}	[11]
	MitM	$5^\dagger/10$	8CP	2^{64}	2^{56}	2^{128}	[19]
	Partial Sum	5/10	2^8CP	2^{40}	small	2^{128}	[55]
	R-Boomerang	5/10	2^9ACC	2^{23}	2^9	2^{128}	[28]
	Yoyo	5/10	2^{11}ACC	2^{31}	small	2^{128}	[47]
AES-192	MitM	6/12	2KP	2^{176}	2^{72}	2^{192}	Sect. 4.4
	MitM	6/12	2^8CP	$2^{109.6}$	$2^{109.6}$	2^{192}	[19]
	Multiple-of-8	7/12	2^{26}CP	$2^{146.3}$	2^{40}	2^{192}	[4]
AES-256	MitM	7/14	2KP	2^{248}	2^{72}	2^{256}	Sect. 4.5
	MitM	6/14	2^8CP	2^{122}	2^{113}	2^{256}	[19]
	MitM	7/14	2^8CP	2^{170}	2^{186}	2^{256}	[19]
	MitM	7/14	2^{26}CP	2^{146}	2^{40}	2^{256}	[4]
Rijndael-EM-128	MitM	7/10	1KP	2^{112}	2^{32}	2^{128}	Sect. 5.1
Rijndael-EM-192	MitM	8/12	1KP	2^{176}	2^{16}	2^{192}	Sect. 5.2
Rijndael-EM-256	MitM	9/14	1KP	2^{248}	2^8	2^{256}	Sect. 5.3

†: The attacks cover x full rounds of AES.

2 Preliminaries

2.1 AES and Rijndael

AES-128/192/256 [17] is a 128-bit block cipher with a 128/192/256-bit key, respectively. In contrast, the block length of Rijndael [17] can be 128/192/256 bits. The state is treated as a $4 \times N_{col}$ ($N_{col} = 4, 6, 8$) two-dimensional array of bytes. The i-th ($i \geq 0$, $\mathsf{MC}^{(-1)} = P$) round of Rijndael round function (Figure 2) typically consists of the following operations:

Table 2: A Summary of the attacks on Hash functions.

Target	Attacks	Methods	Rounds	Time	Memory	Generic	Ref.
Keccak[768]	Preimage	Rotational	4/24	2^{378}	-	2^{384}	[44]
		Linear Structure	4/24	2^{375}	-		[42]
		Algebraic	4/24	2^{374}	2^{224}		[22]
		MitM	4/24	2^{367}	$\mathbf{2^{157}}$		Sect. I.3
Keccak[1024]	Preimage	Rotational	4/24	2^{506}	-	2^{512}	[44]
		MitM	4/24	2^{504}	2^{108}		[46]
		MitM	4/24	2^{504}	$\mathbf{2^{52}}$		Sect. I.1
		Algebraic†	4/24	2^{502}	2^{482}		[22]
		MitM	4/24	2^{500}	$\mathbf{2^{118}}$		Sect. I.2
Ascon-XOF	Preimage	MitM	3/12	2^{120}	2^{39}	2^{128}	[46]
		MitM	3/12	2^{114}	2^{24}		[18]
		MitM	3/12	2^{114}	$\mathbf{2^{14}}$		Sect. H.1
		MitM	4/12	2^{124}	2^{54}		[46]
		MitM	4/12	2^{124}	2^{34}		[18]
		MitM	4/12	2^{124}	$\mathbf{2^{14}}$		Sect. H.2
AES-256	Preimage	MitM	9/14	2^{120}	2^{8}	2^{128}	[2]
		MitM	10/14	2^{120}	2^{56}		[26]
		MitM	10/14	2^{120}	$\mathbf{2^{8}}$		Sect. 4.6
Xoodyak-XOF 128-bit Tag	Preimage	MitM	3/12	2^{125}	2^{97}	2^{128}	[45]
		MitM	3/12	2^{121}	2^{118}		[27]
		MitM	3/12	2^{121}	$\mathbf{2^{37}}$		Sect. G.1
	Collision	MitM	3/12	$\mathbf{2^{60.5}}$	$\mathbf{2^{60.5}}$	2^{64}	Sect. G.2
Gimli-XOF	Preimage	MitM	9/24	2^{104}	2^{70}	2^{128}	[41]
		MitM	10/24	2^{125}	2^{64}		Sect. J
Subterranean 2.0	Preimage	MitM	Full	2^{160}	2^{100}	2^{224}	[27]
		MitM	Full	2^{152}	2^{91}		Sect. K

- AddRoundKey (AK): XOR a round key $\mathsf{RK}^{(i)}$ into the state $\mathsf{MC}^{(i-1)}$ to produce $\mathsf{A}^{(i)}$. The key schedule for AES is given in Figure 15, 16, 17 in Supplementary Material B.
- SubBytes (SB): Substitute each cell of $\mathsf{A}^{(i)}$ according to an S-box to get $\mathsf{SB}^{(i)}$.
- ShiftRows (SR): For $N_{col} = 4, 6$, rotate the ith row of $\mathsf{SB}^{(i)}$ to the left by i bytes ($i = 0, 1, 2, 3$). For $N_{col} = 8$, rotate the $0, 1, 2, 3$rd row to the left by $0, 1, 3, 4$ bytes, respectively.
- MixColumns (MC): Update each column of $\mathsf{SR}^{(i)}$ by left-multiplying an MDS matrix shown in Eq. (25) in Supplementary Material B to get $\mathsf{MC}^{(i)}$.

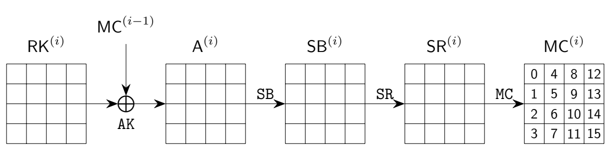

Fig. 2: One Round of AES

AES in FAEST [5]. FAEST is a 2nd-round candidate of NIST PQC - Additional Digital Signature Schemes. FAEST's one-way function is defined using AES and

Rijndael. Taking the FAEST-128 as an example, which is based on AES-128, the plaintext-ciphertext pair $(P, C = \text{AES-128}(k, P))$ is used as the public key of the signature scheme (verification key) and encryption key k is used as the secret key (signing key). If an adversary can recover the encryption key k given only a single plaintext-ciphertext pair (P, C) of AES-128, *i.e.*, the public key of the signature scheme, then he can compute the secret signing key k. This allows him to forge a signature by following exactly the honest prover protocol with the recovered signing key k. This demonstrates that a key recovery attack with one data complexity on AES-128 leads to a signature forgery on FAEST. In FAEST-192/-256 based on AES-192/-256, the size of k is larger than the block size, and FAEST-192/-256 uses two (P, C) pairs as the public key (verification key). Because one (P, C) pair with only 128-bit information cannot prove a 192 or 256-bit knowledge of the secret signing key k. Therefore, the key-recovery attack on AES-192/-256 with two plaintext-ciphertext pairs matters for FAEST-192/-256. FAEST additionally uses Rijndael in Even-Mansour (EM) mode, *i.e.*, FAEST-EM, where Rijndael block cipher is used as a permutation in EM mode. The original key of Rijndael block cipher is published as part of the public key (along with one plaintext-ciphertext), and the new block cipher Rijndael-EM's key is the secret signing key. Therefore, the original key of Rijndael block cipher is a known constant when performing the key-recovery attack on Rijndael-EM.

2.2 Preliminaries of Basic Meet-in-the-Middle Attack

The following notations will be used in the MitM framework.

Symbol	Description
$\mathcal{E}$	The encryption process.
$\mathcal{K}$	The key schedule process.
$I^{\mathcal{E}}/I^{\mathcal{K}}$	Starting state for encryption, key schedule, respectively.
E^+/E^-	Ending state for the forward/backward computation.
$\mathcal{B}^{\mathcal{E}}/\mathcal{B}^{\mathcal{K}}$	Blue cells ■ in starting state $I^{\mathcal{E}}/I^{\mathcal{K}}$.
$\mathcal{R}^{\mathcal{E}}/\mathcal{R}^{\mathcal{K}}$	Red cells ■ in starting state $I^{\mathcal{E}}/I^{\mathcal{K}}$.
$\mathcal{G}^{\mathcal{E}}/\mathcal{G}^{\mathcal{K}}$	Gray cells ■ in starting state $I^{\mathcal{E}}/I^{\mathcal{K}}$.
$\mathcal{M}^+/\mathcal{M}^-$	Matching cells in the ending state E^+/E^-.
$\lambda^+ = \lvert\mathcal{B}^{\mathcal{E}}\rvert + \lvert\mathcal{B}^{\mathcal{K}}\rvert$	The initial degree of freedom for the forward computation.
$\lambda^- = \lvert\mathcal{R}^{\mathcal{E}}\rvert + \lvert\mathcal{R}^{\mathcal{K}}\rvert$	The initial degree of freedom for the backward computation.
π^+/π^-	Certain constraints on the starting state.

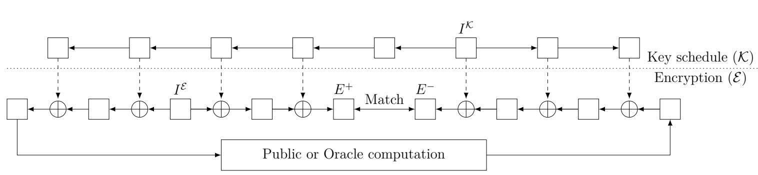

Fig. 3: A high-level overview of the MitM attacks [26]

At CRYPTO 2021, Dong *et al.* [26] gave a formal description of the MitM attack as shown in Figure 3. Assume that the states involved in the encryption ($\mathcal{E}$) and key schedule ($\mathcal{K}$) contain n and $\bar{n}$ w-bit cells, respectively. The public or oracle computation in Figure 3 can be a simple exclusive-or of a given target value for preimage attacks, or an oracle of block cipher for key-recovery attacks.

Dong *et al.* [26] specified several states for MitM in Figure 3, *i.e.*, two starting states $I^{\mathcal{E}}, I^{\mathcal{K}}$, the ending state E^+ for the forward computation (the computation path starting from $(I^{\mathcal{E}}, I^{\mathcal{K}})$ leading to E^+), and similarly the ending state E^- for the backward computation. The cells of $(I^{\mathcal{E}}, I^{\mathcal{K}})$ are partitioned into different subsets with different meanings. Let $\mathcal{B}^{\mathcal{E}}$, $\mathcal{B}^{\mathcal{K}}$, $\mathcal{R}^{\mathcal{E}}$, $\mathcal{R}^{\mathcal{K}}$, $\mathcal{M}^+$, and $\mathcal{M}^-$ be some ordered subsets of $\mathcal{N} = \{0, 1, \cdots, n-1\}$ or $\overline{\mathcal{N}} = \{0, 1, \cdots, \bar{n}-1\}$ such that $\mathcal{B}^{\mathcal{E}} \cap \mathcal{R}^{\mathcal{E}} = \emptyset$, $\mathcal{B}^{\mathcal{K}} \cap \mathcal{R}^{\mathcal{K}} = \emptyset$, $\mathcal{G}^{\mathcal{E}} = \mathcal{N} - \mathcal{B}^{\mathcal{E}} \cup \mathcal{R}^{\mathcal{E}}$ and $\mathcal{G}^{\mathcal{K}} = \overline{\mathcal{N}} - \mathcal{B}^{\mathcal{K}} \cup \mathcal{R}^{\mathcal{K}}$. The index sets are used to reference the cells of the states, *e.g.*, for a 16-cell state I and $\mathcal{B}^{\mathcal{E}} = [0, 1, 3]$, we have $I[\mathcal{B}^{\mathcal{E}}] = I[0, 1, 3] = (I[0], I[1], I[3])$.

The cells $(I^{\mathcal{E}}[\mathcal{B}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{B}^{\mathcal{K}}])$, visualized as ■ cells, are called neutral words of the forward computation, and the cells $(I^{\mathcal{E}}[\mathcal{R}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{R}^{\mathcal{K}}])$, visualized as ■ cells, are called neutral words of the backward computation. The initial degrees of freedom for the forward and backward computation are defined as $\lambda^+ = |\mathcal{B}^{\mathcal{E}}| + |\mathcal{B}^{\mathcal{K}}|$ and $\lambda^- = |\mathcal{R}^{\mathcal{E}}| + |\mathcal{R}^{\mathcal{K}}|$ respectively, that is, the numbers of ■ cells and ■ cells in the starting states. $I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}]$ and $I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}]$ are visualized as ■ cells. Define ℓ^+ functions $\boldsymbol{\pi}^+ = (\pi_1^+, \cdots, \pi_{\ell^+}^+)$ whose values can be computed with the knowledge of the ■ cells $(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}])$ and ■ cells $(I^{\mathcal{E}}[\mathcal{B}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{B}^{\mathcal{K}}])$ in the starting states, where

$$\pi_i^+ : \mathbb{F}_2^{w \cdot (|\mathcal{G}^{\mathcal{E}}| + |\mathcal{G}^{\mathcal{K}}| + |\mathcal{B}^{\mathcal{E}}| + |\mathcal{B}^{\mathcal{K}}|)} \rightarrow \mathbb{F}_2^w$$

is a function mapping $(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], I^{\mathcal{E}}[\mathcal{B}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{B}^{\mathcal{K}}])$ to a w-bit word. Similarly, we define a sequence of ℓ^- functions $\boldsymbol{\pi}^- = (\pi_1^-, \cdots, \pi_{\ell^-}^-)$ whose values can be computed with the knowledge of the ■ cells $(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}])$ and ■ cells $(I^{\mathcal{E}}[\mathcal{R}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{R}^{\mathcal{K}}])$. $\boldsymbol{\pi}^+$ and $\boldsymbol{\pi}^-$ will be used to represent certain constraints on the neutral words of the forward and backward computations, respectively, as given in Property 1.

Property 1. For any fixed $\mathfrak{c}^+ = (a_1, \cdots, a_{\ell^+}) \in \mathbb{F}_2^{w \cdot \ell^+}$ *and* $\mathfrak{c}^- = (b_1, \cdots, b_{\ell^-}) \in \mathbb{F}_2^{w \cdot \ell^-}$*, when the cells* $(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}])$ *are fixed to an arbitrary constant, the neutral words fulfill the following systems:*

$$\begin{cases} \pi_1^+(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], I^{\mathcal{E}}[\mathcal{B}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{B}^{\mathcal{K}}]) = a_1 \\ \pi_2^+(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], I^{\mathcal{E}}[\mathcal{B}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{B}^{\mathcal{K}}]) = a_2 \\ \cdots\ \cdots \\ \pi_{\ell^+}^+(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], I^{\mathcal{E}}[\mathcal{B}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{B}^{\mathcal{K}}]) = a_{\ell^+} \end{cases} \quad (1) \qquad \begin{cases} \pi_1^-(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], I^{\mathcal{E}}[\mathcal{R}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{R}^{\mathcal{K}}]) = b_1 \\ \pi_2^-(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], I^{\mathcal{E}}[\mathcal{R}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{R}^{\mathcal{K}}]) = b_2 \\ \cdots\ \cdots \\ \pi_{\ell^-}^-(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], I^{\mathcal{E}}[\mathcal{R}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{R}^{\mathcal{K}}]) = b_{\ell^-} \end{cases} \quad (2)$$

Then $E^+[\mathcal{M}^+]$ *can be derived from neutral words* $(I^{\mathcal{E}}[\mathcal{B}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{B}^{\mathcal{K}}])$ *and* $E^-[\mathcal{M}^-]$ *can be derived from neutral words* $(I^{\mathcal{E}}[\mathcal{R}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{R}^{\mathcal{K}}])$*, independently.*

For any given $(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}])$ and $\mathfrak{c}^+ = (a_1, \cdots, a_{\ell^+})$, the solution space of $(I^{\mathcal{E}}[\mathcal{B}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{B}^{\mathcal{K}}])$ induced by Eq. (1) is denoted by

$$\mathbb{B}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathfrak{c}^+).$$

Since there are $\lambda^+ = |\mathcal{B}^{\mathcal{E}}| + |\mathcal{B}^{\mathcal{K}}|$ w-bit variables and ℓ^+ equations, we expect $2^{w \cdot (\lambda^+ - \ell^+)}$ solutions, and we call $\mathrm{DoF}^+ = \lambda^+ - \ell^+$ the *degree of freedom (DoF)* *for the forward computation*. Similarly, the solution space of $(I^{\mathcal{E}}[\mathcal{R}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{R}^{\mathcal{K}}])$ induced by Eq. (2) is denoted by $\mathbb{R}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathfrak{c}^-)$, whose size is $2^{w\cdot(\lambda^- - \ell^-)}$. We call $\text{DoF}^- = \lambda^- - \ell^-$ the *DoF for the backward computation*. Assume the computation connecting $E^+[\mathcal{M}^+]$ and $E^-[\mathcal{M}^-]$ forms an m-cell filter, which is denoted as the degree of matching ($\text{DoM} = m$). The MitM attack is Algorithm 1. To find a preimage of h-cell target, the complexity of Algorithm 1 is about

$$(2^w)^{h-\min\{\text{DoF}^+,\text{DoF}^-,\text{DoM}\}} + \mathcal{T}_{pre}, \tag{3}$$

where $\mathcal{T}_{pre}$ is the time complexity to precompute $\mathbb{B}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathfrak{c}^+)$ and $\mathbb{R}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathfrak{c}^-)$ in Line 2.

Algorithm 1: The MitM Attack

1 Assign $(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}])$ and $\mathfrak{c}^+$, and $\mathfrak{c}^-$ to some constants.
2 Solve Eq. (1) and (2) to obtain $\mathbb{B}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathfrak{c}^+)$ and $\mathbb{R}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathfrak{c}^-)$.
3 For values in $\mathbb{B}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathfrak{c}^+)$, compute $E^+[\mathcal{M}^+]$ and insert it into L
4 For values in $\mathbb{R}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathfrak{c}^-)$, compute $E^-[\mathcal{M}^-]$ to match L
5 In case of partial-matching exists in the above step, for the surviving pairs, check for a full-state match. In case none of them are fully matched, repeat the procedure by changing the values of fixed bytes till finding a full match.

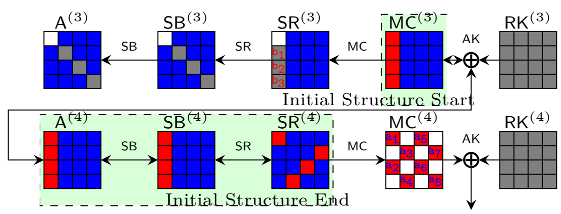

Fig. 4: Sasaki and Aoki's initial structure of MitM attack on AES

Sasaki and Aoki's Initial Structure [50]. In Line 2 of Algorithm 1, we have to solve Eq. (1) and (2). In most previous MitM attacks, the system of equations is linear and easy to solve [49,2]. At EUROCRYPT 2009, Sasaki and Aoki formalized this linear case as the *initial structure* technique [50]. Take the *initial structure* of Sasaki's 7-round MitM attack [49] on AES as an example (also given in Figure 14), which covers from state $\mathsf{MC}^{(3)}$ to $\mathsf{SR}^{(4)}$ in Figure 4, where the red/blue neutral words satisfy some linear equation system, *i.e.*, the cancellations are linear. For example, Eq. (4) are the linear computations for red neutral words,

$$\begin{bmatrix} b_1 = 9 \cdot \mathsf{MC}^{(3)}[0] \oplus e \cdot \mathsf{MC}^{(3)}[1] \oplus b \cdot \mathsf{MC}^{(3)}[2] \oplus d \cdot \mathsf{MC}^{(3)}[3] \\ b_2 = d \cdot \mathsf{MC}^{(3)}[0] \oplus 9 \cdot \mathsf{MC}^{(3)}[1] \oplus e \cdot \mathsf{MC}^{(3)}[2] \oplus b \cdot \mathsf{MC}^{(3)}[3] \\ b_3 = b \cdot \mathsf{MC}^{(3)}[0] \oplus d \cdot \mathsf{MC}^{(3)}[1] \oplus 9 \cdot \mathsf{MC}^{(3)}[2] \oplus e \cdot \mathsf{MC}^{(3)}[3] \end{bmatrix}, \tag{4}$$

where $\mathfrak{c}^- = (b_1, b_2, b_3)$, $\ell^- = 3$. After satisfying the cancellations in Eq. (4), the value space of the red neutral words $\mathsf{MC}^{(3)}[0-3]$ is reduced from $2^{w\cdot\lambda^-} = 2^{8\times 4}$ to $2^{w\cdot(\lambda^- - \ell^-)} = 2^8$, and the value of $\mathsf{MC}^{(3)}[0-3]$ from the space of size 2^8 has a constant impact on the backward computation. The value space of neutral words is easily obtained by solving the linear system Eq. (4). Therefore, the time complexity $\mathcal{T}_{pre}$ in Eq. (3) is usually ignored [49,2].

Dong *et al.*'s Nonlinear Constrained Neutral Words [26]. As noticed by Dong *et al.* [26], the Eq. (1) and (2) of many interesting MitM characteristics are nonlinear systems in practice, and there is no efficient method to solve them. Therefore, Dong *et al.* presented a table-based technique in Algorithm 2 which can be applied in attacks relying on such MitM characteristics without solving the equations. The major drawback of Dong *et al.*'s approach is that it would require a huge amount of memory to prepare two hash tables V and U, and many attacks based on this method need huge memory, *e.g.*, [26,45].

Algorithm 2: Computing the solution spaces of the neutral words

Input: $(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}]) \in \mathbb{F}_2^{w\cdot(|\mathcal{G}^{\mathcal{E}}|+|\mathcal{G}^{\mathcal{K}}|)}$
Output: V, U

$V \leftarrow [\,]$, $U \leftarrow [\,]$
for $(I^{\mathcal{E}}[\mathcal{B}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{B}^{\mathcal{K}}]) \in \mathbb{F}_2^{w\cdot(|\mathcal{B}^{\mathcal{E}}|+|\mathcal{B}^{\mathcal{K}}|)}$ **do**
  $\mathbf{v} \leftarrow \boldsymbol{\pi}^+(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], I^{\mathcal{E}}[\mathcal{B}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{B}^{\mathcal{K}}])$ by Eq. (1)
  Insert $(I^{\mathcal{E}}[\mathcal{B}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{B}^{\mathcal{K}}])$ into V at index $\mathbf{v}$
end
for $(I^{\mathcal{E}}[\mathcal{R}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{R}^{\mathcal{K}}]) \in \mathbb{F}_2^{w\cdot(|\mathcal{R}^{\mathcal{E}}|+|\mathcal{R}^{\mathcal{K}}|)}$ **do**
  $\mathbf{u} \leftarrow \boldsymbol{\pi}^-(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], I^{\mathcal{E}}[\mathcal{R}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{R}^{\mathcal{K}}])$ by Eq. (2)
  Insert $(I^{\mathcal{E}}[\mathcal{R}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{R}^{\mathcal{K}}])$ into U at index $\mathbf{u}$
end

2.3 Triangulation Algorithm (TA)

The triangulation algorithm (TA) was introduced by Khovratovich, Biryukov, and Nikolic [34] at CT-RSA 2009, which is a Gaussian-like elimination process to solve the nonlinear system. The algorithm expresses all transformations as equations that link the internal variables. Variables refer to bits or bytes/-words depending on the trail. In the triangulation algorithm, *free variables* form the basis of the nonlinear system, which are to be assigned with arbitrary values. The variables that can be determined by the free variables are called *dependent variables*. The idea is to build a set of *dependent variables* that includes many variables. The more such variables we have among the dependent variables, the more conditions are satisfied at no cost. The heart of the triangulation algorithm is to search for dependent variables. The formal process can be described as follows.

1. Given the system of equations with fixed predefined values as constants.
2. Label all variables and equations as unprocessed. Initially, all variables and equations are marked as unprocessed, meaning they have not yet been simplified or solved.
3. Identify a variable that appears in only one unprocessed equation. Label both the variable and the corresponding equation as processed. If there is no such variable, label all the unprocessed equations as processed, exit.
4. Repeat Step 3 if there are still unprocessed equations.
5. If all equations have been processed, mark all the remaining unprocessed variables as free variables.
6. Assign random values to free variables and compute the remaining variables.

For example, Eq. (5) is a nonlinear system of 7 byte-variables $s, t, u, v, x, y, z \in \mathbb{F}_2^8$, and F, G, H, and L are bijective functions. After applying TA, we get Eq. (6), where x and s are free variables and by varying them we deduce the values of the other 5 dependent variables.

$$
\begin{cases}
F(x \oplus s) \oplus v = 0, \\
G(x \oplus u) \oplus s \oplus L(y \oplus z) = 0, \\
v \oplus G(u \oplus s) = 0, \\
H(z \oplus s \oplus v) \oplus t = 0, \\
u \oplus H(t \oplus x) = 0,
\end{cases} \tag{5}
$$

$$
\begin{cases}
L(y \oplus z) \oplus G(u \oplus x) \oplus s = 0, \\
z \oplus H^{-1}(t) \oplus v \oplus s = 0, \\
t \oplus H^{-1}(u) \oplus x = 0, \\
u \oplus G^{-1}(v) \oplus s = 0, \\
v \oplus F(x \oplus s) = 0.
\end{cases} \tag{6}
$$

The TA algorithm is used by Khovratovich *et al.* to speed up the collision search on AES hashing mode [34]. They described the hash function as a system of equations with S-boxes, and added equations to force the message and chaining value to obey their differential characteristic inside the function. Solving these equations will produce a collision. At CRYPTO 2022, Dong *et al.* combined the TA and rebound attack [43] to propose the triangulating rebound attack [25], where the TA is used to solve certain nonlinear system to connect multiple inbound phases efficiently. Therefore, TA was mainly exploited in differential attacks previously, and in this paper we will exploit TA in MitM attacks.

3 Triangulating MitM Attack Framework

3.1 Limitations of Khovratovich *et al.*'s TA.

The previous triangulation algorithm faces a significant limitation in Step 3 to Step 5 in Sect. 2.3 when there are still many unprocessed equations, but no variable exists in only one equation. For example, if there is another byte-equation

$$P(s \oplus v \oplus t) \oplus z = 0, \tag{7}$$

then together with Eq. (5), only one dependent variable y can be obtained. Similar to [34], consider the equation system as a *matrix of dependencies*, where the rows correspond to equations, and the columns to variables. In Eq. (8), the matrix before TA represents the system combining Eq. (5) and the additional Eq. (7). When applying Khovratovich *et al.*'s TA given in Sect. 2.3, after determining one dependent variable y, the remaining six variables in the remaining 5 equations would be directly marked as free variables since there is no variable that appears in only one unprocessed equation, and the TA terminates. We move the 5 equations to the top of the right matrix of Eq. (8) and mark them in cyan. In this case, Khovratovich *et al.*'s TA can not reduce the system and eliminate potential free variables further. Then, we have to randomly assign values for the six free variables $s, t, u, v, x, z \in \mathbb{F}_2^8$ and check if the 5 byte-equations (the first 5 rows in cyan) are satisfied, whose probability is 2^{-40}. Once satisfied, y is deduced to satisfy the last equation. The time complexity is around 2^{40}.

$$\text{Before TA:} \left(\begin{array}{ccccccc} s & t & u & v & x & y & z \\ \hline 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right) \xrightarrow[\text{Khovratovich } et\ al.\text{'s TA}]{\text{after extract } y} \left(\begin{array}{c|cccccc} y & s & t & u & v & x & z \\ \hline 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ \hline 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ & \multicolumn{6}{c}{free\ variables} \end{array}\right) \tag{8}$$

3.2 Improved Triangulation Algorithm with Structured Gaussian Elimination

Structured Gaussian elimination (SGE). At 1990 and 1999, LaMacchia *et al.* [37] and Bender *et al.* [6] proposed the structured Gaussian elimination (SGE) paradigm, which solves the large and sparse linear system efficiently. Consider the linear system of equations of the form $M\mathbf{x} = \mathbf{0}$, where M is the coefficient matrix of the linear system. The SGE steps of LaMacchia *et al.* [37] can be summarized roughly as follows:

1. Delete all the columns that have a single non-zero coefficient and the rows in which those columns have non-zero coefficients (this step is similar to step 3 of Khovratovich *et al.*'s TA).
2. For any row which has only a single non-zero coefficient, subtract appropriate multiples of that row from all other rows that have non-zero coefficients on that column so as to make those coefficients 0. This step can reduce the matrix without introducing new non-zero coefficients for other rows.
   However, for nonlinear system, this step usually does not help. E.g. in Eq. (8), the matrix is different from the coefficient matrix of the linear system. In Eq. (8), the non-zero entry of the matrix means the variable exists in the corresponding nonlinear equation, *i.e.,* the variable may exist in multiple linear or nonlinear terms in that equation. Therefore, one cannot apply similar step to reduce the rows for nonlinear system.

3. Delete some rows which have the largest number of non-zero elements. Apply this step when steps 1 and 2 are not possible.
   We apply this step when the Khovratovich *et al.*'s TA cannot proceed.

Improved TA with the idea of SGE. The improvements happen to Step 3 to Step 5 of Khovratovich *et al.*'s TA by a similar approach of the SGD [37,6], *i.e.,* when we are stuck and cannot determine any new dependent variable, greedily remove the biggest equation that have the largest number of non-zero element (that we will have to satisfy stochastically) and until we can make progress. Specifically, we introduce a new rule to process the system (highlighted in italics), and the modified algorithm proceeds as follows.

1. **Construct the system of equations:** Given the system of equations, fix the predefined values to constants.
2. **Initialize all variables and equations as unprocessed:** Mark all variables and all equations as unprocessed.
3. **Find the variable involved in only one unprocessed equation:**
   (a) Search for a variable that appears in only one unprocessed equation. If such a variable exists, mark the equation and the variable as processed.
   (b) *If no such variable can be found, perform the following steps:*
      i. *Count the number of variables present in each unprocessed equation.*
      ii. *Identify the unprocessed equations that contain the largest number of variables.*
      iii. *Remove one of the equations in (ii) from the system and mark it as processed. This reduces the scale of the remaining system.*
4. **Repeat Step 3 until all equations have been processed:** *Continue searching for variables involved in a single equation or removing equations with the maximum number of variables until no unprocessed equations exist.*
5. **Assign free variables:** After all equations are processed, mark all remaining unprocessed variables as free.
6. **Solve the system:** Assign random values to the free variables. Using these values, compute the remaining variables by substituting them back into the processed equations.

This enhancement ensures that the system is further simplified even when no variable appears in a single equation. By strategically removing the equation with the largest number of variables, we reduce the remaining system and maximize the opportunities for variable elimination. At last, fewer variables are marked as free, leading to a more efficient solution process.

Now let's continue to consider the example in Eq. (8), the whole process is illustrated in Eq. (9). After extracting y, instead of immediately marking the remaining variables as free, we analyze the number of variables included in each remaining unprocessed equation and prioritize the equations with the largest number of variables (4-th and 6-th row in the first matrix of Eq. (9)), which are highlighted in **bold**. Label one of them as processed and remove it from the equation system, *i.e.* move this equation to the top of the second matrix of Eq. (9) and highlight it in cyan, continue to process the remaining 4 unprocessed equations (the last 4 rows) to extract dependent variables z, t, u, v sequentially.

$$
\begin{pmatrix}
y & s & t & u & v & x & z \\ \hline
1 & 1 & 0 & 1 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 1 & 1 & 0 & 0 \\
\mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} \\
0 & 0 & 1 & 1 & 0 & 1 & 0 \\
\mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1}
\end{pmatrix}
\xrightarrow[6th\ row]{remove}
\begin{pmatrix}
y & s & t & u & v & x & z \\ \hline
0 & 1 & 1 & 0 & 1 & 0 & 1 \\ \hline
1 & 1 & 0 & 1 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 1 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 1 & 0
\end{pmatrix}
\xrightarrow[z]{extract}
\begin{pmatrix}
y & z & s & t & u & v & x \\ \hline
0 & 1 & 1 & 1 & 0 & 1 & 0 \\ \hline
1 & 1 & 1 & 0 & 1 & 0 & 1 \\
0 & 1 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 1
\end{pmatrix}
\xrightarrow[t]{extract}
$$

$$
\begin{pmatrix}
y & z & t & s & u & v & x \\ \hline
0 & 1 & 1 & 1 & 0 & 1 & 0 \\ \hline
1 & 1 & 0 & 1 & 1 & 0 & 1 \\
0 & 1 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 0
\end{pmatrix}
\xrightarrow[u,v]{extract}
\begin{pmatrix}
y & z & t & u & v & s & x \\ \hline
0 & 1 & 1 & 0 & 1 & 1 & 0 \\ \hline
1 & 1 & 0 & 1 & 0 & 1 & 1 \\
0 & 1 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1
\end{pmatrix}
\xrightarrow{Finally}
\begin{pmatrix}
y & z & t & u & v & x & s \\ \hline
0 & 1 & 1 & 0 & 1 & 0 & 1 \\ \hline
1 & 1 & 0 & 1 & 0 & 1 & 1 \\
0 & 1 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1
\end{pmatrix}_{\qquad free} . \tag{9}
$$

As a result, we determine 5 dependent variables (y, z, t, u, v) and 2 free variables x, s. Then we randomly assign values for the 2 free variables $x, s \in \mathbb{F}_2^8$ and deduce the values of v, u, t, z, y in turn. And then, check if the first equation marked in cyan in Eq. (9) is satisfied, whose probability is 2^{-8}. The total time complexity to solve the nonlinear system is 2^8, which is significantly smaller than the time 2^{40} by Khovratovich *et al.*'s TA.

3.3 Triangulating MitM Attack: Solving Nonlinear Constrained Neutral Words with the New TA

When applying our improved TA to the MitM attack, we combine the nonlinear system solving by the improved TA and a memory-aided precomputation to compute the solution space of the neutral words efficiently. Taking Eq. (8) as an example and supposing the system Eq. (2) is Eq. (8), *i.e.*, Eq. (10), where the 7-byte variables $s, t, u, v, x, y, z \in \mathbb{F}_2^8$ are the $\lambda^- = 7$ cells $(I^{\mathcal{E}}[\mathcal{R}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{R}^{\mathcal{K}}])$ in the starting states of the MitM path. Given global constants $(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}])$, Eq. (2) becomes the left system of Eq. (10), *i.e.*, $\ell^- = 6$.

$$
\begin{cases}
\pi_1^-(s, \quad\ v, x, \qquad) = b_1 \\
\pi_2^-(s, \ u, \quad x, y, z\,) = b_2 \\
\pi_3^-(s, \ u, v, \qquad\quad) = b_3 \\
\pi_4^-(s, t, \ v, \qquad z\,) = b_4 \\
\pi_5^-(\ \ t, u, \quad x, \qquad) = b_5 \\
\pi_6^-(s, t, \ v, \qquad z\,) = b_6
\end{cases}
\xrightarrow{\text{New TA}}
\begin{cases}
\begin{array}{l|l|l|l|l|l|l|l}
\pi_4^-(& z, & t, & & v, & & s &) = b_4 \\ \hline
\pi_2^-(y, & z, & & u, & & x, & s &) = b_2 \\
\pi_6^-(& z, & t, & & v, & & s &) = b_6 \\
\pi_5^-(& & t, & u, & & x, & &) = b_5 \\
\pi_3^-(& & & u, & v, & & s &) = b_3 \\
\pi_1^-(& & & & v, & x, & s &) = b_1
\end{array}
\end{cases} \tag{10}
$$

Following Dong *et al.*'s method in Algorithm 2, we have to traverse 7-byte variables and compute $\mathbf{u} \leftarrow (b_1, b_2, \cdots, b_6) \in \mathbb{F}_2^{48}$ and store the 7-byte string (s, t, u, v, x, y, z) into a hash table U at index $\mathbf{u}$, which needs a huge memory of about $2^{56} \cdot 7$ bytes.

Based on our new TA, we give a new Algorithm 3 to solve the nonlinear constrained neutral words. After applying the new TA, we get the right system of Eq. (10), where x, s are two free variables and v, u, t, z, y are 5 dependent variables. Given b_2, b_6, b_5, b_3, b_1 and x, s, the dependent variable v, u, t, z, y are successively determined by the evaluation the lower 5 nonlinear equations in

Algorithm 3: Computing the value space of the neutral words with New TA and a memory-aided precomputation

```
   Input: (I^E[G^E], I^K[G^K]) ∈ F_2^{w·(|G^E|+|G^K|)}
1  for (b_2, b_6, b_5, b_3, b_1) ∈ F_2^{8×5} do
2      V ← [ ], U ← [ ]
3      for (x, s) ∈ F_2^{8×2} do
4          Compute v from π_1^-()
5          Compute u from π_3^-()
6          Compute t from π_5^-()
7          Compute z from π_6^-()
8          Compute y from π_2^-()
9          Compute u = b_4 by equations marked by cyan
10         Store U[u] ← (x, s, v, u, t, z, y)
11     end
12     Similarly, we can prepare V
13     Then, under each index i, j, compute the values from U[i] backward, and
         independently, compute the values from V[j] forward, and filter the
         states by the matching point.
14 end
```

Eq. (10). After all variables are determined, compute b_4 by $\pi_4^-()$ as the index $\mathbf{u}$. In Algorithm 3, the 2^8 solution spaces of $\mathbb{R}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathfrak{c}^-)$, with $|\mathbb{R}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathfrak{c}^-)| = 2^{8\cdot(\lambda^- - \ell^-)} = 2^8$, are stored in U in Line 10. Therefore, only the lower 5 nonlinear equations in the right system of Eq. (10) are actually solved, whose solutions are all stored in U under different index $\mathbf{u}$. We call this method to prepare the solution space of neutral words as a combination of improved TA with memory-aided precomputation. At last, the size to store U is about $2^{16} \cdot 7$ bytes. Compared to Dong *et al.*'s method, the memory is significantly reduced from $2^{56} \cdot 7$ bytes to $2^{16} \cdot 7$ bytes.

3.4 Automatic Triangulating MitM

The full search framework of our attacks consists of two steps:

1. The first step is to use the existing MILP models for MitM attacks on AES and other primitives to find massive MitM paths.
   E.g., for AES we use Dong *et al.*'s model [26] to search potential MitM paths, and more than 1000 MitM paths are found for 5-round AES-128.
2. The second step is to apply the improved TA to each MitM path to solve the systems of the nonlinear constrained neutral words (e.g., Eq. (2) and (1)) and recognize the good MitM path with improved memory complexity.

Comparison with the guess-and-determine approach in [11]. At CRYPTO 2011, Bouillaguet, Derbez, and Fouque [11] introduced a powerful automatic tool for searching guess-and-determine and MitM attacks on byte-oriented symmetric primitives by programming techniques such as knowledge propagation and some pruning techniques. The tool automatically and exhaustively searches all possible sets of "free variables" to find a good one, leading to the exploration of a large search tree. The complexity of the exhaustive search is inherently exponential and exploring the whole space might not be feasible.

Our improved TA developed from the structured Gaussian elimination [37,6] does not explore the full space. Therefore, for certain nonlinear byte-equation systems, our algorithm may output weaker solutions than Bouillaguet *et al.*'s tool. However, as shown in our search framework, we first automatically find massive MitM paths and then solve many nonlinear systems for those MitM paths. Hence, the method used to solve nonlinear systems should be very efficient. The time complexity of our improved TA is linear with the number of equations, which is very suitable for our search framework. The guess-and-determine algorithm in [11] exhausted all possible solutions, but it can be slow because so many nonlinear systems should be solved. Moreover, our improved TA is also efficient when the system is huge (e.g., 299 nonlinear equations with 316 variables in Supplementary Material I.3), where the algorithm in [11] may not output solutions in a reasonable time.

Furthermore, in our triangulating MitM attacks, we are not solving the full nonlinear system to get the solution space of the neutral words. As explained in Sect. 3.3, we actually combine the improved TA with the memory-aided precomputation to compute the solution space of neutral words. This core idea is well explained in our 5-round attack on AES-128 in Sect. 4.1. For example, in Eq. (13)-(f), our triangulating MitM first automatically selects a system of 5 byte-equations and solves it for each value of the 5-byte value $(\widehat{\mathsf{A}}^{(2)}[4], \widehat{\mathsf{RK}}^{(2)}[12,13], \widehat{\mathsf{A}}^{(1)}[3,4])$. Then, store all the solutions in a hash table under the index of 4-byte value $\mathbf{u} = (\widehat{\mathsf{SR}}^{(3)}[1,4], \widehat{\mathsf{RK}}^{(5)}[0], \widehat{\mathsf{A}}^{(1)}[14])$ (see Line 7-9 in Algorithm 4). Therefore, all the solutions of the 5 byte-equations are stored under different index of the table U and no solutions are filter out, since they are all useful in the following MitM procedures. Our improved TA is very suitable for the memory-aided precomputation, since it directly identifies these 5 byte-equations.

4 Attacks on Reduced AES with One/Two Plaintexts

4.1 Single-Plaintext Key-Recovery Attack on 5-round AES-128

The 5-round MitM characteristic is shown in Figure 5, where green cells 🟩 mean linear combinations of 🟥 and 🟦. In the MitM path, the starting state $\mathsf{RK}^{(0)}$, whose bytes are denoted as k_0 to k_{15}, contains $\lambda^+ = 4$ 🟦 bytes and $\lambda^- = 12$ 🟥 bytes. In the computation from $\mathsf{RK}^{(0)}$ to $\mathsf{RK}^{(5)}$, from $\mathsf{A}^{(0)}$ to $\mathsf{SR}^{(2)}$, and from $\mathsf{SR}^{(4)}$ to $\mathsf{MC}^{(2)}$, the consumed degrees of freedom (DoFs) of 🟦 and 🟥 are $\ell^+ = 3$ and $\ell^- = 9$ bytes, respectively. Therefore, $\mathrm{DoF}^+ = 1$, $\mathrm{DoF}^- = 3$, and there is DoM $= 1$ matching byte in round 2. The 9 consumed DoFs of 🟥 on $\mathsf{A}^{(1)}[3]$, $\mathsf{A}^{(1)}[4]$,

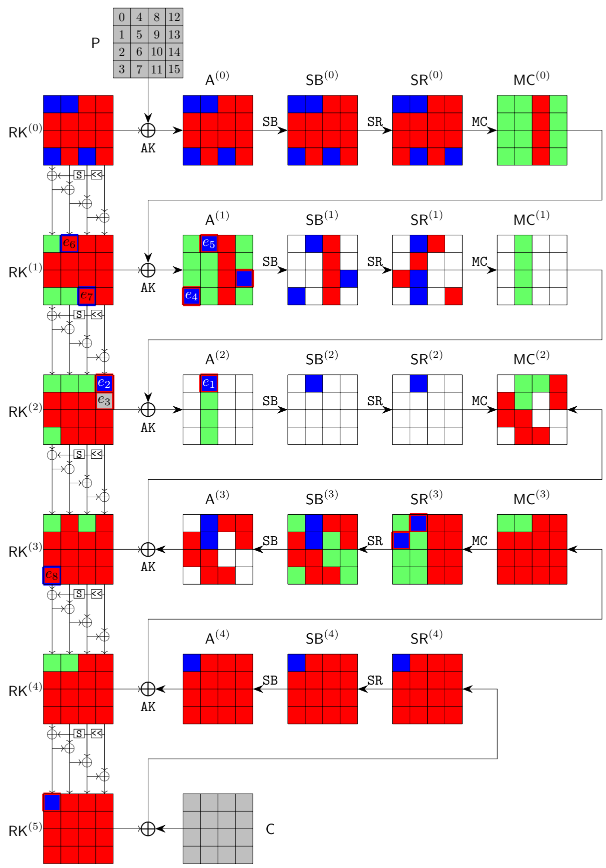

Fig. 5: 5-round attack on AES-128

$\mathsf{A}^{(1)}[14]$, $\mathsf{RK}^{(2)}[12]$, $\mathsf{RK}^{(2)}[13]$, $\mathsf{A}^{(2)}[4]$, $\mathsf{SR}^{(3)}[1]$, $\mathsf{SR}^{(3)}[4]$, and $\mathsf{RK}^{(5)}[0]$ marked by $\blacksquare/\blacksquare$ in Figure 5 are a system on the byte-variables of $\mathsf{RK}^{(0)}$ in Eq. (11).

$$\begin{cases}
\mathsf{A}^{(1)}[3] = S(k_5) \oplus S(k_{10}) \oplus \mathtt{2} \cdot S(k_{15}) \oplus S(k_{12}) \oplus \mathtt{3} \cdot S(k_0) \oplus k_3 \\
\mathsf{A}^{(1)}[4] = \mathtt{3} \cdot S(k_9) \oplus S(k_{14}) \oplus S(k_{13}) \oplus \mathtt{2} \cdot S(k_4) \oplus k_4 \oplus S(k_3) \oplus k_0 \\
\mathsf{A}^{(1)}[14] = S(k_{12}) \oplus S(k_1) \oplus \mathtt{2} \cdot S(k_6) \oplus k_{14} \oplus k_{10} \oplus k_6 \oplus k_2 \oplus S(k_{15}) \oplus \mathtt{3} \cdot S(k_{11}) \\
\mathsf{A}^{(2)}[4] = \mathtt{3} \cdot S(S(k_8) \oplus \mathtt{2} \cdot S(k_{13}) \oplus \mathtt{3} \cdot S(k_2) \oplus S(k_7) \oplus k_9 \oplus k_5 \oplus k_1 \oplus S(k_{14})) \oplus \\
\qquad S(k_{13} \oplus k_9 \oplus k_5 \oplus k_1 \oplus S(k_{14})) \oplus k_4 \oplus S(\mathsf{A}^{(1)}[3]) \oplus \mathtt{2} \cdot S(\mathsf{A}^{(1)}[4]) \oplus S(\mathsf{A}^{(1)}[14]) \\
\mathsf{RK}^{(2)}[12] = k_{12} \oplus S(k_{13} \oplus k_9 \oplus k_5 \oplus k_1 \oplus S(k_{14})) \oplus k_4 \\
\mathsf{RK}^{(2)}[13] = k_{13} \oplus k_5 \oplus S(k_{14} \oplus k_{10} \oplus k_6 \oplus k_2 \oplus S(k_{15})) \\
\mathsf{SR}^{(3)}[1] = \mathtt{9} \cdot (S(\mathsf{RK}^{(0)}[13]) \oplus S(\mathsf{RK}^{(1)}[13]) \oplus S(\mathsf{RK}^{(2)}[13]) \oplus S(\mathsf{RK}^{(3)}[13])) \oplus \\
\qquad \mathtt{e} \cdot (\mathsf{MC}^{(3)}[1]) \oplus \mathtt{b} \cdot (\mathsf{MC}^{(3)}[2]) \oplus \mathtt{d} \cdot (\mathsf{MC}^{(3)}[3]) \oplus \mathtt{9} \cdot (\mathsf{RK}^{(0)}[0] \oplus \mathsf{A}^{(4)}[0]) \\
\mathsf{SR}^{(3)}[4] = \mathtt{e} \cdot (S(\mathsf{RK}^{(0)}[13]) \oplus S(\mathsf{RK}^{(1)}[13]) \oplus S(\mathsf{RK}^{(2)}[13]) \oplus S(\mathsf{RK}^{(3)}[13]) \oplus \mathsf{RK}^{(3)}[4] \oplus \mathsf{A}^{(4)}[4]) \\
\qquad \oplus \mathtt{b} \cdot (\mathsf{MC}^{(3)}[5]) \oplus \mathtt{d} \cdot (\mathsf{MC}^{(3)}[6]) \oplus \mathtt{9} \cdot (\mathsf{MC}^{(3)}[7]) \oplus \mathtt{e} \cdot (\mathsf{RK}^{(0)}[0]) \\
\mathsf{RK}^{(5)}[0] = S(\mathsf{RK}^{(0)}[13]) \oplus S(\mathsf{RK}^{(1)}[13]) \oplus S(\mathsf{RK}^{(2)}[13]) \oplus S(\mathsf{RK}^{(3)}[13]) \oplus S(\mathsf{RK}^{(4)}[13]) \oplus \mathsf{RK}^{(0)}[0]
\end{cases} \tag{11}$$

Excluding the parts of Eq. (11) related to blue bytes, we get Eq. (12) which is only related to the red bytes, where the boxed parts are deleted[5].

$$\begin{cases}
\widehat{\mathsf{A}}^{(1)}[3] = S(k_5) \oplus S(k_{10}) \oplus \mathtt{2} \cdot S(k_{15}) \oplus S(k_{12}) \boxed{\oplus \mathtt{3} \cdot S(k_0) \oplus k_3} \\
\widehat{\mathsf{A}}^{(1)}[4] = \mathtt{3} \cdot S(k_9) \oplus S(k_{14}) \oplus S(k_{13}) \boxed{\oplus \mathtt{2} \cdot S(k_4) \oplus k_4 \oplus S(k_3) \oplus k_0} \\
\widehat{\mathsf{A}}^{(1)}[14] = S(k_{12}) \oplus S(k_1) \oplus \mathtt{2} \cdot S(k_6) \oplus k_{14} \oplus k_{10} \oplus k_6 \oplus k_2 \oplus S(k_{15}) \oplus \boxed{\mathtt{3} \cdot S(k_{11})} \\
\widehat{\mathsf{A}}^{(2)}[4] = \mathtt{3} \cdot S(S(k_8) \oplus \mathtt{2} \cdot S(k_{13}) \oplus \mathtt{3} \cdot S(k_2) \oplus S(k_7) \oplus k_9 \oplus k_5 \oplus k_1 \oplus S(k_{14})) \oplus S(k_{13} \oplus k_9 \oplus k_5 \\
\qquad \oplus k_1 \oplus S(k_{14})) \boxed{\oplus k_4 \oplus S(\widehat{\mathsf{A}}^{(1)}[3] \oplus B_1) \oplus \mathtt{2} \cdot S(\widehat{\mathsf{A}}^{(1)}[4] \oplus B_2) \oplus S(\widehat{\mathsf{A}}^{(1)}[14] \oplus B_3)} \\
\widehat{\mathsf{RK}}^{(2)}[12] = k_{12} \oplus S(k_{13} \oplus k_9 \oplus k_5 \oplus k_1 \oplus S(k_{14})) \boxed{\oplus k_4} \\
\widehat{\mathsf{RK}}^{(2)}[13] = k_{13} \oplus k_5 \oplus S(k_{14} \oplus k_{10} \oplus k_6 \oplus k_2 \oplus S(k_{15})) \\
\widehat{\mathsf{SR}}^{(3)}[1] = \mathtt{9} \cdot (S(\mathsf{RK}^{(0)}[13]) \oplus S(\mathsf{RK}^{(1)}[13]) \oplus S(\widehat{\mathsf{RK}}^{(2)}[13]) \oplus S(\mathsf{RK}^{(3)}[13])) \oplus \\
\qquad \mathtt{e} \cdot (\mathsf{MC}^{(3)}[1]) \oplus \mathtt{b} \cdot (\mathsf{MC}^{(3)}[2]) \oplus \mathtt{d} \cdot (\mathsf{MC}^{(3)}[3]) \boxed{\oplus \mathtt{9} \cdot (\mathsf{RK}^{(0)}[0] \oplus \mathsf{A}^{(4)}[0])} \\
\widehat{\mathsf{SR}}^{(3)}[4] = \mathtt{e} \cdot (S(\mathsf{RK}^{(0)}[13]) \oplus S(\mathsf{RK}^{(1)}[13]) \oplus S(\widehat{\mathsf{RK}}^{(2)}[13]) \oplus S(\mathsf{RK}^{(3)}[13]) \oplus \mathsf{RK}^{(3)}[4] \oplus \mathsf{A}^{(4)}[4]) \\
\qquad \oplus \mathtt{b} \cdot (\mathsf{MC}^{(3)}[5]) \oplus \mathtt{d} \cdot (\mathsf{MC}^{(3)}[6]) \oplus \mathtt{9} \cdot (\mathsf{MC}^{(3)}[7]) \boxed{\oplus \mathtt{e} \cdot (\mathsf{RK}^{(0)}[0])} \\
\widehat{\mathsf{RK}}^{(5)}[0] = S(\mathsf{RK}^{(0)}[13]) \oplus S(\mathsf{RK}^{(1)}[13]) \oplus S(\widehat{\mathsf{RK}}^{(2)}[13]) \oplus S(\mathsf{RK}^{(3)}[13]) \oplus S(\mathsf{RK}^{(4)}[13]) \boxed{\oplus \mathsf{RK}^{(0)}[0]}
\end{cases} \tag{12}$$

where $B_1 = \mathtt{3} \cdot S(k_0) \oplus k_3,\ B_2 = \mathtt{2} \cdot S(k_4) \oplus k_4 \oplus S(k_3) \oplus k_0,\ B_3 = \mathtt{3} \cdot S(k_{11})$. The Eq. (12) is first expressed as the matrix (a) in Eq. (13), where the rows correspond to the equations and the columns to variables. Apply our new TA:

1. At the beginning, in matrix (a), no variable appears in only one unprocessed equation. Count the number of variables present in each unprocessed equation; there are 3 unprocessed equations that contain 12 variables, which are highlighted in **bold**.
2. Remove the 3 bold rows and label them as processed by moving them to the top of the matrix highlighted in cyan. We get the matrix (b).

[5] In our MitM attack (see Line 14 to 27 of Algorithm 4), we need the 9 bytes $(\mathsf{A}^{(1)}[3], \mathsf{A}^{(1)}[4], \cdots)$ on the left side of Eq. (11) (the bytes marked in red border in Figure 5) to depend only on the blue/gray bytes. Therefore, we specify the red parts in Eq. (12) as global constants $(\widehat{\mathsf{A}}^{(1)}[3], \widehat{\mathsf{A}}^{(1)}[4], \cdots)$, so that the red bytes have a constant effect on the 9 bytes $(\mathsf{A}^{(1)}[3], \mathsf{A}^{(1)}[4], \cdots)$.

3. Process the last 6 rows of (b) with TA and extract a dependent variable k_7 marked by green. We get the matrix (c).
4. Process the last 5 rows of (c). No variable appears in only one unprocessed equation, we count and remove the unprocessed equation that contains the most number of variables, *i.e.*, row $\widehat{\mathsf{A}}^{(1)}[14]$ and get the matrix (d).
5. Process the last 4 rows of matrix (d) and extract k_1, k_2, k_5, k_9 sequentially to get the matrix (f).

$$
\left(\begin{array}{r|cccccccccccc}
 & k_1 & k_2 & k_5 & k_6 & k_7 & k_8 & k_9 & k_{10} & k_{12} & k_{13} & k_{14} & k_{15} \\
\hline
\widehat{\mathsf{A}}^{(1)}[3] & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
\widehat{\mathsf{A}}^{(1)}[4] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[14] & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
\widehat{\mathsf{A}}^{(2)}[4] & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(2)}[12] & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(2)}[13] & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[1] & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\
\widehat{\mathsf{SR}}^{(3)}[4] & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\
\widehat{\mathsf{RK}}^{(5)}[0] & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1}
\end{array}\right)
\quad (a)
$$

$$
\left(\begin{array}{r|cccccccccccc}
 & k_1 & k_2 & k_5 & k_6 & k_7 & k_8 & k_9 & k_{10} & k_{12} & k_{13} & k_{14} & k_{15} \\
\hline
\widehat{\mathsf{SR}}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{RK}}^{(5)}[0] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
\widehat{\mathsf{A}}^{(1)}[3] & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
\widehat{\mathsf{A}}^{(1)}[4] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[14] & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
\widehat{\mathsf{A}}^{(2)}[4] & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(2)}[12] & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(2)}[13] & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1
\end{array}\right)
\quad (b)
$$

$$
\left(\begin{array}{r|cccccccccccc}
 & k_7 & k_1 & k_2 & k_5 & k_6 & k_8 & k_9 & k_{10} & k_{12} & k_{13} & k_{14} & k_{15} \\
\hline
\widehat{\mathsf{SR}}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{RK}}^{(5)}[0] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
\widehat{\mathsf{A}}^{(2)}[4] & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[3] & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
\widehat{\mathsf{A}}^{(1)}[4] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[14] & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{1} \\
\widehat{\mathsf{RK}}^{(2)}[12] & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(2)}[13] & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1
\end{array}\right)
\quad (c)
$$

$$
\left(\begin{array}{r|cccccccccccc}
 & k_7 & k_1 & k_2 & k_5 & k_6 & k_8 & k_9 & k_{10} & k_{12} & k_{13} & k_{14} & k_{15} \\
\hline
\widehat{\mathsf{SR}}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{RK}}^{(5)}[0] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{A}}^{(1)}[14] & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
\hline
\widehat{\mathsf{A}}^{(2)}[4] & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[3] & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
\widehat{\mathsf{A}}^{(1)}[4] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(2)}[12] & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(2)}[13] & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1
\end{array}\right)
\quad (d)
\tag{13}
$$

$$
\left(\begin{array}{r|cccccccccccc}
 & k_7 & k_1 & k_2 & k_5 & k_6 & k_8 & k_9 & k_{10} & k_{12} & k_{13} & k_{14} & k_{15} \\
\hline
\widehat{\mathsf{SR}}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{RK}}^{(5)}[0] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{A}}^{(1)}[14] & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
\hline
\widehat{\mathsf{A}}^{(2)}[4] & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(2)}[12] & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[3] & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
\widehat{\mathsf{A}}^{(1)}[4] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(2)}[13] & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1
\end{array}\right)
\quad (e)
$$

$$
\left(\begin{array}{r|ccccc|ccccccc}
 & k_7 & k_1 & k_2 & k_5 & k_9 & k_6 & k_8 & k_{10} & k_{12} & k_{13} & k_{14} & k_{15} \\
\hline
\widehat{\mathsf{SR}}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{RK}}^{(5)}[0] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{A}}^{(1)}[14] & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
\hline
\widehat{\mathsf{A}}^{(2)}[4] & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(2)}[12] & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(2)}[13] & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\
\widehat{\mathsf{A}}^{(1)}[3] & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
\widehat{\mathsf{A}}^{(1)}[4] & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0
\end{array}\right)_{\text{free variables}}
\quad (f)
$$

Finally, we extract 5 dependent variables $\mathsf{RK}^{(0)}[7,1,2,5,9] = (k_7, k_1, k_2, k_5, k_9)$ marked in green in Eq. (13)-(f) from the rows of $\widehat{\mathsf{A}}^{(2)}[4]$, $\widehat{\mathsf{RK}}^{(2)}[12]$, $\widehat{\mathsf{RK}}^{(2)}[13]$, $\widehat{\mathsf{A}}^{(1)}[3]$, and $\widehat{\mathsf{A}}^{(1)}[4]$. The others are 7 free variables $\mathsf{RK}^{(0)}[6,8,10,12,13,14,15] = (k_6, k_8, k_{10}, k_{12}, k_{13}, k_{14}, k_{15})$. The values $(e_1, e_2, e_3, e_4, e_5) \in \mathbb{F}_2^{40}$ are assigned to the expressions for the red bytes of $(\widehat{\mathsf{A}}^{(2)}[4]$, $\widehat{\mathsf{RK}}^{(2)}[12]$, $\widehat{\mathsf{RK}}^{(2)}[13]$, $\widehat{\mathsf{A}}^{(1)}[3]$, $\widehat{\mathsf{A}}^{(1)}[4])$, then given the values of the seven free variables, the dependent variables $(k_9, k_5, k_2, k_1, k_7)$ are deduced in sequence. Thereafter, the values of $\widehat{\mathsf{SR}}^{(3)}[1]$, $\widehat{\mathsf{SR}}^{(3)}[4]$, $\widehat{\mathsf{RK}}^{(5)}[0]$ and $\widehat{\mathsf{A}}^{(1)}[14]$ are deduced directly.

In the 3 consumed DoFs of blue byte ■ on $\mathsf{RK}^{(1)}[4]$, $\mathsf{RK}^{(1)}[11]$, and $\mathsf{RK}^{(3)}[3]$, the expressions are

$$\begin{cases} \mathsf{RK}^{(1)}[4] = k_4 \oplus k_0 \oplus S(k_{13}) \\ \mathsf{RK}^{(1)}[11] = k_{11} \oplus k_3 \oplus k_7 \oplus S(k_{12}) \\ \mathsf{RK}^{(3)}[3] = k_3 \oplus S(k_4 \oplus e_2) \oplus S(k_4 \oplus k_0 \oplus S(k_{13}) \oplus k_8 \oplus k_{12}) \oplus S(k_{12}) \end{cases}. \tag{14}$$

After assigning the following formulas as constants (e_6, e_7, e_8),

$$\begin{cases} k_4 \oplus k_0 = e_6 \\ k_{11} \oplus k_3 = e_7 \\ k_3 \oplus S(k_4 \oplus e_2) = e_8 \end{cases}, \tag{15}$$

the bytes $\mathsf{RK}^{(1)}[4]$, $\mathsf{RK}^{(1)}[11]$, $\mathsf{RK}^{(3)}[3]$ will be ■, *i.e.*, only determined by the red cells. By applying the TA to Eq. (15), 1 free variable k_0 is obtained, the other 3 variables $\mathsf{RK}^{(0)}[3, 4, 11] = (k_3, k_4, k_{11})$ are deduced directly for any value of the free variable k_0. The 5-round MitM attack is given in Algorithm 4.

Analysis of Algorithm 4. In Line 12 to 27, $2^{\zeta+24+16+32+8+8} = 2^{128}$ states should be tested to recover the 128-bit key; therefore, $\zeta = 40$. According to Eq. (3), $\mathcal{T}_{pre}$ corresponds to the time complexity of Line 7-9, which is about $2^{\zeta+24+16+40} = 2^{120}$ 1-round AES. Therefore, the first part of Eq. (3) dominates the overall complexity, which is about $2^{128-8\cdot\min\{\mathrm{DoF}^+,\mathrm{DoF}^-,\mathrm{DoM}\}} = 2^{120}$ 5-round AES. The memory complexity to store U is about 2^{40}.

4.2 Practical-Memory Key-Recovery Attack on 4-full-round AES-128

The attack leverages the new representation of AES's key schedule by Leurent and Pernot [39]. They introduced a new basis $\mathsf{S}^{(i)}$ to derive the round keys, *i.e.*, $\mathsf{RK}^{(i)} = C_0 \cdot \mathsf{S}^{(i)}$ as shown in Figure 6, where C_0 is a 16×16 binary matrix given Eq. (26) in Supplementary Material B.

The MitM path is shown in Figure 6. The starting state $\mathsf{S}^{(2)}$, whose bytes are denoted by k_0 to k_{15}, contains $\lambda^+ = 1$ ■ byte and $\lambda^- = 6$ ■ bytes. The consumed DoFs of ■ and ■ are $\ell^+ = 0$ and $\ell^- = 5$ bytes, respectively. The $\ell^- = 5$ constraints (marked by ■/■) form a system of 5 nonlinear equations in Eq. (17) (deleting the boxed parts). Therefore, $\mathrm{DoF}^+ = 1$, $\mathrm{DoF}^- = 1$, and there is $\mathrm{DoM} = 1$ matching byte in round 1. Using new TA, we get 3 free variables $\mathsf{S}^{(2)}[7, 10, 13] = (k_7, k_{10}, k_{13})$ and 3 dependent variables $\mathsf{S}^{(2)}[0, 1, 4] = (k_0, k_1, k_4)$. The matrices before and after the improved TA are shown in Eq. (16). The steps for the MitM attack are given in Algorithm 5. The time complexity is about

Algorithm 4: Key-recovery attack on 5-round AES-128 with 1 (P, C)

for 2^{ζ} *values of* $(e_1, e_2, e_3, e_4, e_5) \in \mathbb{F}_2^{40}$ **do**
    for $(e_6, e_7, e_8) \in \mathbb{F}_2^{24}$ **do**
        for $\mathsf{RK}^{(0)}[14, 15] \in \mathbb{F}_2^{16}$ **do**
            $U \leftarrow [\ ]$
            $(\widehat{\mathsf{A}}^{(1)}[4], \widehat{\mathsf{RK}}^{(2)}[12], \widehat{\mathsf{RK}}^{(2)}[13], \widehat{\mathsf{A}}^{(1)}[3], \widehat{\mathsf{A}}^{(2)}[4]) \leftarrow (e_1, e_2, e_3, e_4, e_5)$
            for $\mathsf{RK}^{(0)}[6, 8, 10, 12, 13] \in \mathbb{F}_2^{40}$ **do**
                Compute $\mathsf{RK}^{(0)}[7, 1, 2, 5, 9] = (k_7, k_1, k_2, k_5, k_9)$ by Eq. (13)-(f)
                Compute $\mathbf{u} = (\widehat{\mathsf{SR}}^{(3)}[1], \widehat{\mathsf{SR}}^{(3)}[4], \widehat{\mathsf{RK}}^{(5)}[0], \widehat{\mathsf{A}}^{(1)}[14]) \in \mathbb{F}_2^{32}$
                $U[\mathbf{u}] \leftarrow \mathsf{RK}^{(0)}[1, 2, 5-10, 12-15] \in \mathbb{F}_2^{8\times 12}$
                `/* The nonlinear system solving and memory-aided precomputation are combined to get the solution space of the neutral words. There are about` 2^8 `elements in` $U[\mathbf{u}]$ `for each` $\mathbf{u}$`. */`
            end
            for $\mathbf{u} \in \mathbb{F}_2^{32}$ **do**
                $L \leftarrow [\ ]$
                for $\mathsf{RK}^{(0)}[0] = k_0 \in \mathbb{F}_2^{8}$ **do**
                    Compute $\mathsf{RK}^{(0)}[3, 4, 11] = (k_3, k_4, k_{11})$ by Eq. (15)
                    Compute the 1-byte matching point $v = \mathsf{SR}^{(2)}[4] \oplus e \cdot \mathsf{MC}^{(2)}[4] \oplus b \cdot \mathsf{MC}^{(2)}[5]$
                    $L[v] \leftarrow (k_0, k_3, k_4, k_{11})$
                end
                for *values in* $U[\mathbf{u}]$ **do**
                    Compute the 1-byte matching point $v' = e \cdot \mathsf{MC}^{(2)}[4] \oplus b \cdot \mathsf{MC}^{(2)}[5] \oplus d \cdot \mathsf{MC}^{(2)}[6] \oplus 9 \cdot \mathsf{MC}^{(2)}[7]$
                    for *values in* $L[v']$ **do**
                        if $E(Key = \mathsf{RK}^{(0)}, P) = C$ **then**
                            return $\mathsf{RK}^{(0)}$
                        end
                    end
                end
            end
        end
    end
end

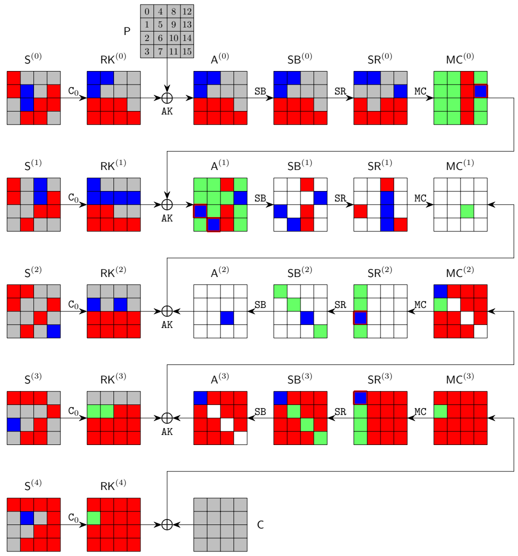

Fig. 6: 4-round attack on AES-128

$$
\begin{array}{c}
\left(\begin{array}{r|cccccc}
 & k_0 & k_1 & k_4 & k_7 & k_{10} & k_{13} \\
\hline
\widehat{\mathsf{MC}}^{(0)}[13] & 0 & 0 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{A}}^{(1)}[2] & 0 & 1 & 1 & 1 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[7] & 1 & 0 & 0 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[0] & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(2)}[2] & 1 & 1 & 1 & 1 & 1 & 1
\end{array}\right) \\
(a) \\
\left(\begin{array}{rccc|ccc}
 & k_0 & k_1 & k_4 & k_7 & k_{10} & k_{13} \\
\hline
\widehat{\mathsf{SR}}^{(3)}[0] & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(2)}[2] & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
\widehat{\mathsf{A}}^{(1)}[7] & 1 & 0 & 0 & 1 & 1 & 1 \\
\widehat{\mathsf{A}}^{(1)}[2] & 0 & 1 & 1 & 1 & 1 & 0 \\
\widehat{\mathsf{MC}}^{(0)}[13] & 0 & 0 & 1 & 1 & 1 & 1
\end{array}\right) \\
(b)
\end{array}
\tag{16}
$$

$2^{128-8\cdot\min\{\mathrm{DoF}^+,\mathrm{DoF}^-,\mathrm{DoM}\}} = 2^{120}$. The memory is 2^{24} to store U.

$$
\begin{cases}
\widehat{\mathsf{MC}}^{(0)}[13] = 3 \cdot S(k_{14} \oplus k_4 \oplus S(k_7)) \oplus S(k_{10} \oplus k_{13}) \boxed{\oplus 2 \cdot \mathsf{SR}^{(0)}[13] \oplus \mathsf{SR}^{(0)}[12]} \\
\widehat{\mathsf{A}}^{(1)}[2] = 2 \cdot S(k_{14} \oplus k_1) \oplus 3 \cdot S(k_{10}) \oplus k_4 \oplus S(k_7) \oplus k_1 \boxed{\oplus \mathsf{SR}^{(0)}[0] \oplus k_{11} \oplus k_{14} \oplus \mathsf{SR}^{(0)}[1]} \\
\widehat{\mathsf{A}}^{(1)}[7] = 2 \cdot S(k_{10} \oplus k_{13} \oplus k_0 \oplus S(k_3) \oplus k_7 \oplus S(k_6)) \oplus k_7 \oplus k_{13} \oplus \boxed{3 \cdot (\mathsf{SR}^{(0)}[4]) \oplus \mathsf{SR}^{(0)}[5] \oplus \mathsf{SR}^{(0)}[6]} \\
\widehat{\mathsf{SR}}^{(2)}[2] = \mathtt{9} \cdot (\mathsf{A}^{(3)}[1] \oplus \mathsf{S}^{(3)}[5] \oplus \mathsf{S}^{(3)}[8] \oplus \mathsf{S}^{(3)}[15]) \oplus \mathtt{e} \cdot \mathsf{MC}^{(2)}[2] \oplus \mathtt{b} \cdot \mathsf{MC}^{(2)}[3] \boxed{\oplus \mathtt{9} \cdot \mathsf{S}^{(3)}[2] \oplus \mathtt{d} \cdot \mathsf{MC}^{(2)}[0]} \\
\widehat{\mathsf{SR}}^{(3)}[0] = \mathtt{e} \cdot \mathsf{RK}^{(4)}[0] \oplus \mathtt{b} \cdot (\mathsf{S}^{(4)}[2] \oplus \cdot \mathsf{S}^{(4)}[8] \oplus \mathsf{S}^{(4)}[15]) \oplus \mathtt{d} \cdot \mathsf{RK}^{(4)}[2] \oplus \mathtt{9} \cdot \mathsf{RK}^{(4)}[3] \boxed{\oplus \mathtt{b} \cdot \mathsf{S}^{(4)}[5]}
\end{cases}
\tag{17}
$$

Partial Experiment of the Key-Recovery Attack. We give an experiment of a 4-byte partial key-recovery attack as follows:

1. Data collection: encrypt the plaintext $P = 0$ with key $\mathsf{S}^{(2)} = 0$ to get the ciphertext C.
2. Given $(P = 0, C)$ and 12-byte key information to recover the other 4-byte key $\mathsf{S}^{(2)}[7, 10, 13, 15]$. If the recovered $\mathsf{S}^{(2)}[7, 10, 13, 15] = 0$, the attack succeeds. The 12-byte key information includes 9 ■ bytes $\mathsf{S}^{(2)}[2, 3, 5, 6, 8, 9, 11, 12, 14] = 0$ and 3-byte key relations of $(\widehat{\mathsf{MC}}^{(0)}[13], \widehat{\mathsf{A}}^{(1)}[2, 7]) = (e_1, e_2, e_3)$ on the 6 red key bytes of $\mathsf{S}^{(2)}$, *i.e.*, assign (e_1, e_2, e_3) to the last 3 expressions of Eq. (16)-(b). From (P, C) and $\mathsf{S}^{(2)} = 0$, pre-compute the 3-byte $(e_1, e_2, e_3) = (\mathtt{0x75}, \mathtt{0x00}, \mathtt{0xc6})$.
   In our experiment, $(P = 0, C)$ and 12-byte key information $(\mathsf{S}^{(2)}[2, 3, 5, 6, 8, 9, 11, 12, 14] = 0$, $(e_1, e_2, e_3) = (\mathtt{0x75}, \mathtt{0x00}, \mathtt{0xc6}))$ are given, the goal is

to recover $\mathsf{S}^{(2)}[7, 10, 13, 15] = 0$. In brute-force search, the time will be 2^{32}. With the given information, we actually conduct the experiment from Line 3 to Line 20 according to Algorithm 5. The time is 2^{24} with 2^{24} memory[6].

We successfully recover the 4-byte partial key $\mathsf{S}^{(2)}[7, 10, 13, 15] = 0$, and the source codes and results are available via

https://github.com/boxindev/Triangulation-MitM

We implemented the experiment on a computer with an i9-13900KF CPU and 32GB of memory. It takes about 200 seconds, while the brute force needs 2^{32} evaluations of 4-round AES-128, which takes about 3200 seconds in the same platform using the same code for AES.

Algorithm 5: Practical-memory attack on 4-full-round AES-128

1 **for** $\mathsf{S}^{(2)}[2, 3, 5, 6, 8, 9, 11, 12, 14] \in \mathbb{F}_2^{8\times 9}$ *and* $(e_1, e_2, e_3) \in \mathbb{F}_2^{24}$ **do**
2 　$(\widehat{\mathsf{MC}}^{(0)}[13], \widehat{\mathsf{A}}^{(1)}[2, 7]) \leftarrow (e_1, e_2, e_3)$, $U \leftarrow [\ ]$
3 　**for** $\mathsf{S}^{(2)}[7, 10, 13] \in \mathbb{F}_2^{24}$ **do**
4 　　Compute $\mathsf{S}^{(2)}[0, 1, 4]$ by Eq. (16)-(b) and $\mathbf{u} = (\widehat{\mathsf{SR}}^{(2)}[2], \widehat{\mathsf{SR}}^{(3)}[0]) \in \mathbb{F}_2^{16}$
5 　　$U[\mathbf{u}] \leftarrow \mathsf{S}^{(2)}[0, 1, 4, 7, 10, 13] \in \mathbb{F}_2^{8\times 7}$
6 　**end**
7 　**for** $\mathbf{u} \in \mathbb{F}_2^{16}$ **do**
8 　　$L \leftarrow [\ ]$
9 　　**for** $\mathsf{S}^{(2)}[15] \in \mathbb{F}_2^{8}$ **do**
10 　　　Compute the 1-byte matching point v, $L[v] \leftarrow \mathsf{S}^{(2)}[15]$
11 　　**end**
12 　　**for** *values in* $U[\mathbf{u}]$ **do**
13 　　　Compute the 1-byte matching point v'
14 　　　**for** *values in* $L[v']$ **do**
15 　　　　**if** $E(Key = \mathsf{S}^{(2)}, P) = C$ **then**
16 　　　　　return $\mathsf{S}^{(2)}$
17 　　　　**end**
18 　　　**end**
19 　　**end**
20 　**end**
21 **end**

4.3 New Attack on 4-full-round AES-128 with 2^{112} Complexity

As shown in Figure 7, the starting state $\mathsf{S}^{(3)} = (k_0, k_1, \cdots, k_{15})$ contains $\lambda^+ = 2$ ■ bytes (*i.e.*, k_{10} and k_{14}) and $\lambda^- = 14$ ■ bytes. The consumed DoFs of ■

[6] In our experiment, we use the data structure "map<uint32_t, vector<vector<uint8_t>>>" to store U, which needs about 300 MB memory.

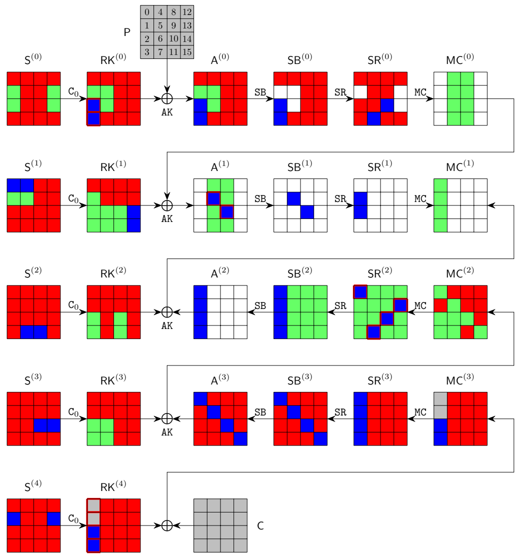

Fig. 7: New 4-round attack on AES-128

$$
\begin{pmatrix}
\begin{array}{r|cccccccccccccc}
 & k_0 & k_1 & k_2 & k_3 & k_4 & k_5 & k_6 & k_7 & k_8 & k_9 & k_{11} & k_{12} & k_{13} & k_{15} \\
\hline
\widehat{\mathsf{RK}}^{(0)}[2] & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(0)}[3] & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(4)}[0] & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\
\widehat{\mathsf{RK}}^{(4)}[1] & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
\widehat{\mathsf{RK}}^{(4)}[2] & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\widehat{\mathsf{RK}}^{(4)}[3] & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[5] & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
\widehat{\mathsf{A}}^{(1)}[10] & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(2)}[0] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(2)}[7] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(2)}[10] & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(2)}[13] & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{array}
\end{pmatrix}
$$

(a)

$$
\begin{pmatrix}
\begin{array}{r|ccccccc|ccccccc}
 & k_{15} & k_2 & k_8 & k_9 & k_{12} & k_{13} & k_{16} & k_1 & k_3 & k_4 & k_5 & k_6 & k_7 & k_{11} \\
\hline
\widehat{\mathsf{SR}}^{(2)}[0] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(2)}[7] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(2)}[10] & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(2)}[13] & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
\widehat{\mathsf{A}}^{(1)}[10] & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\
\hline
\widehat{\mathsf{RK}}^{(4)}[1] & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
\widehat{\mathsf{A}}^{(1)}[5] & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
\widehat{\mathsf{RK}}^{(4)}[0] & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
\widehat{\mathsf{RK}}^{(0)}[3] & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(4)}[3] & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\widehat{\mathsf{RK}}^{(0)}[2] & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\
\widehat{\mathsf{RK}}^{(4)}[2] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1
\end{array}
\end{pmatrix}
\quad (18)
$$

(b)

and ■ are $\ell^+ = 0$ and $\ell^- = 12$ bytes, respectively. The $\ell^- = 12$ constraints (marked by ■/■ in Figure 7) form a system of 12 equations in Eq. (19). Therefore, $\text{DoF}^+ = 2$, $\text{DoF}^- = 2$, and $\text{DoM} = 2$. Using new TA, we get 7 free variables $\mathsf{S}^{(3)}[1, 3, 4, 5, 6, 7, 11]$ and 7 dependent variables $\mathsf{S}^{(3)}[0, 2, 8, 9, 12, 13, 15]$. The matrices before and after TA are given in Eq. (18). The steps for the MitM attack are given in Algorithm 6. The time complexity is about $2^{128-8\cdot\min\{\text{DoF}^+,\text{DoF}^-,\text{DoM}\}} = 2^{112}$. The memory is 2^{56} to store U.

$$
\begin{cases}
\widehat{\mathsf{RK}}^{(0)}[2] = k_1 \oplus S(k_{13}) \oplus k_4 \oplus S(k_7) \oplus k_{11} \boxed{\oplus S(k_{10}) \oplus k_{14}} \\
\widehat{\mathsf{RK}}^{(0)}[3] = k_{13} \oplus k_0 \oplus S(k_3) \oplus k_7 \oplus S(k_6) \oplus S(k_9) \boxed{\oplus k_{10}} \\
\widehat{\mathsf{RK}}^{(4)}[0] = k_{12} \oplus k_3 \oplus k_6 \oplus k_9 \oplus S(k_8) \\
\widehat{\mathsf{RK}}^{(4)}[1] = k_{15} \oplus k_2 \oplus k_5 \oplus S(k_4) \oplus k_8 \\
\widehat{\mathsf{RK}}^{(4)}[2] = k_1 \oplus S(k_0) \oplus k_4 \oplus k_{11} \boxed{\oplus k_{14}} \\
\widehat{\mathsf{RK}}^{(4)}[3] = k_{13} \oplus S(k_{12}) \oplus k_0 \oplus k_7 \boxed{\oplus k_{10}} \\
\widehat{\mathsf{A}}^{(1)}[5] = S(k_3 \oplus S(k_2) \oplus k_9) \oplus 2 \cdot S(k_5 \oplus k_8 \oplus S(k_{11})) \oplus 3 \cdot S(k_1) \oplus k_8 \oplus S(k_{11}) \\
\qquad \oplus k_2 \boxed{\oplus \mathsf{SR}^{(0)}[7]} \\
\widehat{\mathsf{A}}^{(1)}[10] = S(k_{12} \oplus S(k_{15}) \oplus k_9) \oplus S(k_5) \oplus 3 \cdot S(k_{13} \oplus k_7 \oplus S(k_6)) \oplus k_1 \\
\qquad \boxed{\oplus 2 \cdot \mathsf{SR}^{(0)}[10] \oplus k_{14}} \\
\widehat{\mathsf{SR}}^{(2)}[0] = \mathtt{e} \cdot \mathsf{RK}^{(3)}[0] \oplus \mathtt{b} \cdot (\mathsf{RK}^{(3)}[1] \oplus \mathsf{A}^{(3)}[1]) \oplus \mathtt{d} \cdot (k_1 \oplus k_4 \oplus k_{11} \oplus \mathsf{A}^{(3)}[2]) \oplus 9 \cdot (k_0 \\
\qquad \oplus k_7 \oplus k_{13} \oplus \mathsf{A}^{(3)}[3]) \boxed{\oplus \mathtt{e} \cdot \mathsf{A}^{(3)}[0] \oplus \mathtt{d} \cdot k_{14} \oplus 9 \cdot k_{10}} \\
\widehat{\mathsf{SR}}^{(2)}[7] = \mathtt{b} \cdot (\mathsf{RK}^{(3)}[4] \oplus \mathsf{A}^{(3)}[4]) \oplus \mathtt{d} \cdot \mathsf{RK}^{(3)}[5] \oplus 9 \cdot (k_4 \oplus \mathsf{A}^{(3)}[6]) \oplus \mathtt{e} \cdot (k_0 \oplus \mathsf{A}^{(3)}[7]) \\
\qquad \boxed{\oplus \mathtt{d} \cdot \mathsf{A}^{(3)}[5] \oplus 9 \cdot k_{14} \oplus \mathtt{e} \cdot k_{10}} \\
\widehat{\mathsf{SR}}^{(2)}[10] = \mathtt{d} \cdot \mathsf{MC}^{(2)}[8] \oplus 9 \cdot \mathsf{MC}^{(2)}[9] \oplus \mathtt{e} \cdot \mathsf{RK}^{(3)}[10] \oplus \mathtt{b} \cdot \mathsf{MC}^{(2)}[11] \boxed{\oplus \mathtt{e} \cdot \mathsf{A}^{(3)}[10]} \\
\widehat{\mathsf{SR}}^{(2)}[13] = 9 \cdot \mathsf{MC}^{(2)}[12] \oplus \mathtt{e} \cdot \mathsf{MC}^{(2)}[13] \oplus \mathtt{b} \cdot \mathsf{MC}^{(2)}[14] \oplus \mathtt{d} \cdot \mathsf{RK}^{(3)}[15] \boxed{\oplus \mathtt{d} \cdot \mathsf{A}^{(3)}[15]}
\end{cases}
\tag{19}
$$

Algorithm 6: New Attack on 4-full-round AES-128 with 2^{112} time

for $(e_1, e_2, e_3, e_4, e_5, e_6, e_7) \in \mathbb{F}_2^{56}$ **do**
  $(\widehat{\mathsf{A}}^{(1)}[5], \widehat{\mathsf{RK}}^{(0)}[2,3], \widehat{\mathsf{RK}}^{(4)}[0,1,2,3]) \leftarrow (e_1, e_2, e_3, e_4, e_5, e_6, e_7)$, $U \leftarrow [\,]$
  for $\mathsf{S}^{(3)}[1,3,4,5,6,7,11] \in \mathbb{F}_2^{56}$ **do**
    Compute $\mathsf{S}^{(3)}[0,2,8,9,12,13,16]$ by Eq. (18)-(b)
    Compute $\mathbf{u} = (\widehat{\mathsf{A}}^{(1)}[10], \widehat{\mathsf{SR}}^{(2)}[0,7,10,13]) \in \mathbb{F}_2^{40}$
    $U[\mathbf{u}] \leftarrow \mathsf{S}^{(3)}[0-9, 11-13, 15] \in \mathbb{F}_2^{8\times 14}$
  end
  for $\mathbf{u} \in \mathbb{F}_2^{40}$ **do**
    $L \leftarrow [\,]$
    for $\mathsf{S}^{(3)}[10,14] \in \mathbb{F}_2^{16}$ **do**
      Compute the 2-byte matching point v, $L[v] \leftarrow \mathsf{S}^{(3)}[10,14]$
    end
    for *values in* $U[\mathbf{u}]$ **do**
      Compute the 2-byte matching point v' **for** *values in* $L[v']$ **do**
        if $E(Key = \mathsf{S}^{(3)}, P) = C$ **then**
          return $\mathsf{S}^{(3)}$
        end
      end
    end
  end
end

4.4 Two-Plaintext Key-Recovery Attack on 6-round AES-192

The MitM attack on 6-round AES-192 needs two plaintext-ciphertext pairs. One plaintext-ciphertext pair is used in the MitM phase, and the other pair is used to identify the unique correct 192-bit key. Similar situation happens to the 7-round attack on AES-256 in Sect. 4.5.

The 6-round MitM path is given in Figure 8, where the starting state $\mathsf{S}^{(3)}$, whose bytes are denoted as $k_0, k_1, \cdots, k_{23}$, contains $\lambda^+ = 2$ ■ bytes and $\lambda^- = 21$ ■ bytes. The consumed degrees of freedom (DoFs) of ■ and ■ are $\ell^+ = 0$ and $\ell^- = 19$ bytes, respectively. Therefore, $\mathrm{DoF}^+ = 2$, $\mathrm{DoF}^- = 2$, and there are $\mathrm{DoM} = 2$ matching bytes in round 2. We get 19 equations as Eq. (20) (deleting the boxed parts) on the red bytes of $\mathsf{S}^{(3)}$ for the 19 consumed DoFs of ■ marked by ■/■ in Figure 8.

Using the new TA, we get 9 free variables $\mathsf{S}^{(3)}[0, 3, 4, 13, 14, 15, 16, 21, 23]$ and 12 dependent variables $\mathsf{S}^{(3)}[1, 2, 5, 6, 7, 11, 12, 17, 18, 19, 20, 22]$. The matrices after TA are given in Eq. (21). The 6-round MitM attack is given in Algorithm 10 in Supplementary Material C. The total time complexity is about $2^{192-8\cdot\min\{\mathrm{DoF}^+,\mathrm{DoF}^-,\mathrm{DoM}\}} = 2^{176}$. The memory is 2^{72} to store U.

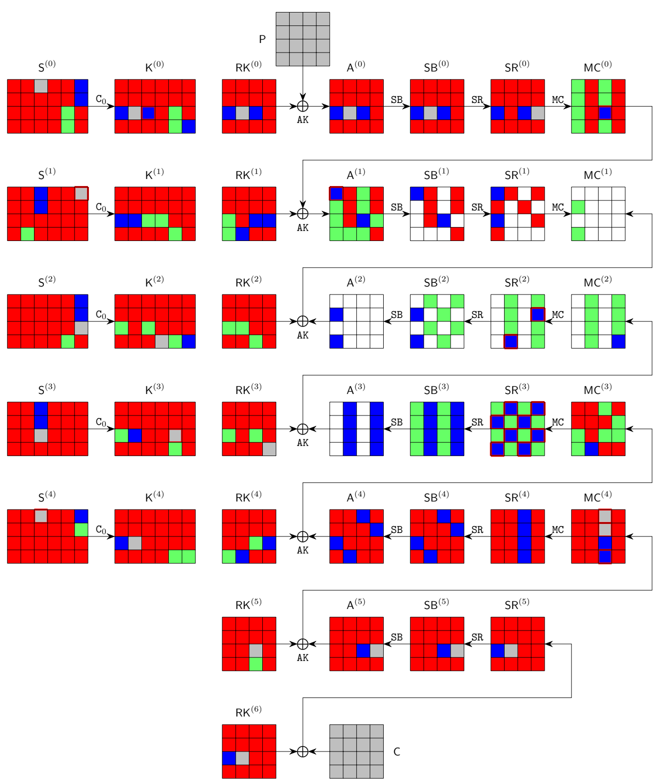

Fig. 8: The 6-round attack on AES-192

$$
\begin{cases}
\widehat{\mathsf{S}}^{(1)}[20] = k_{20} \oplus S(k_{21} \oplus k_{22}); \widehat{\mathsf{S}}^{(4)}[8] = k_{20} \oplus S(k_{21}); \widehat{\mathsf{K}}^{(0)}[10] = k_{19} \oplus k_{20} \oplus k_{7} \boxed{\oplus S(k_9)}; \widehat{\mathsf{K}}^{(3)}[18] = k_6 \oplus k_{18} \\
\widehat{\mathsf{MC}}^{(0)}[10] = S(k_{13} \oplus k_{14} \oplus k_1 \oplus S(k_3)) \oplus S(k_4 \oplus k_5) \oplus 3 \cdot S(k_{11} \oplus S(k_0 \oplus k_1)) \boxed{\oplus 2 \cdot \mathsf{SR}^{(0)}[10]} \\
\widehat{\mathsf{A}}^{(1)}[0] = 2 \cdot S(k_{14} \oplus S(k_{15} \oplus k_{16}) \oplus k_2 \oplus S(k_3) \oplus S(k_3 \oplus k_4)) \oplus 3 \cdot S(k_5 \oplus S(k_6 \oplus k_7)) \\
\qquad \oplus S(k_{10} \oplus k_{11}) \oplus k_{12} \oplus k_{13} \oplus k_0 \oplus k_2 \boxed{\oplus \mathsf{SR}^{(0)}[2]} \\
\widehat{\mathsf{MC}}^{(4)}[8] = S^{-1}(k_{13} \oplus k_{14} \oplus S(k_{15}) \oplus k_1) \oplus k_0 \oplus k_{12}; \quad \widehat{\mathsf{MC}}^{(4)}[9] = S^{-1}(k_5 \oplus S(k_6)) \oplus k_3 \oplus k_{15} \\
\widehat{\mathsf{MC}}^{(4)}[11] = S^{-1}(k_{10} \oplus k_{11} \oplus S(k_0)) \oplus k_{21} \boxed{\oplus k_9} \\
\widehat{\mathsf{SR}}^{(3)}[1] = \mathtt{9} \cdot \mathsf{MC}^{(3)}[0] \oplus \mathtt{e} \cdot \mathsf{MC}^{(3)}[0] \oplus \mathtt{b} \cdot \mathsf{RK}^{(4)}[2] \oplus \mathtt{d} \cdot (k_{21} \oplus k_{22} \oplus \mathsf{A}^{(4)}[3]) \boxed{\oplus \mathtt{b} \cdot \mathsf{A}^{(4)}[2] \oplus \mathtt{d} \cdot k_9} \\
\widehat{\mathsf{SR}}^{(3)}[3] = \mathtt{b} \cdot \mathsf{MC}^{(3)}[0] \oplus \mathtt{d} \cdot \mathsf{MC}^{(3)}[1] \oplus \mathtt{9} \cdot \mathsf{RK}^{(4)}[2] \oplus \mathtt{e} \cdot (k_{21} \oplus k_{22} \oplus \mathsf{A}^{(4)}[3]) \boxed{\oplus \mathtt{9} \cdot \mathsf{A}^{(4)}[2] \oplus \mathtt{e} \cdot k_9} \\
\widehat{\mathsf{SR}}^{(3)}[4] = \mathtt{e} \cdot \mathsf{MC}^{(3)}[4] \oplus \mathtt{b} \cdot \mathsf{MC}^{(3)}[5] \oplus \mathtt{d} \cdot \mathsf{MC}^{(3)}[6] \oplus \mathtt{9} \cdot \mathsf{MC}^{(3)}[7] \\
\widehat{\mathsf{SR}}^{(3)}[6] = \mathtt{d} \cdot \mathsf{MC}^{(3)}[4] \oplus \mathtt{9} \cdot \mathsf{MC}^{(3)}[5] \oplus \mathtt{e} \cdot \mathsf{MC}^{(3)}[6] \oplus \mathtt{b} \cdot \mathsf{MC}^{(3)}[7] \\
\widehat{\mathsf{SR}}^{(3)}[9] = \mathtt{9} \cdot \mathsf{MC}^{(4)}[8] \oplus \mathtt{e} \cdot \mathsf{MC}^{(3)}[9] \oplus \mathtt{b} \cdot (k_{20} \oplus \mathsf{A}^{(4)}[10]) \oplus \mathtt{d} \cdot \mathsf{MC}^{(3)}[11] \boxed{\oplus \mathtt{9} \cdot \mathsf{A}^{(4)}[8] \oplus \mathtt{b} \cdot k_8} \\
\widehat{\mathsf{SR}}^{(3)}[11] = \mathtt{b} \cdot \mathsf{MC}^{(4)}[8] \oplus \mathtt{d} \cdot \mathsf{MC}^{(3)}[9] \oplus \mathtt{9} \cdot (k_{20} \oplus \mathsf{A}^{(4)}[10]) \oplus \mathtt{e} \cdot \mathsf{MC}^{(3)}[11] \boxed{\oplus \mathtt{b} \cdot \mathsf{A}^{(4)}[8] \oplus \mathtt{9} \cdot k_8} \\
\widehat{\mathsf{SR}}^{(3)}[12] = \mathtt{e} \cdot \mathsf{MC}^{(3)}[12] \oplus \mathtt{b} \cdot \mathsf{RK}^{(4)}[13] \oplus \mathtt{d} \cdot \mathsf{A}^{(4)}[14] \oplus \mathtt{9} \cdot \mathsf{MC}^{(3)}[15] \boxed{\oplus \mathtt{b} \cdot \mathsf{A}^{(4)}[13] \oplus \mathtt{d} \cdot \mathsf{RK}^{(4)}[14]} \\
\widehat{\mathsf{SR}}^{(3)}[14] = \mathtt{d} \cdot \mathsf{MC}^{(3)}[12] \oplus \mathtt{9} \cdot \mathsf{RK}^{(4)}[13] \oplus \mathtt{e} \cdot \mathsf{A}^{(4)}[14] \oplus \mathtt{b} \cdot \mathsf{MC}^{(3)}[15] \boxed{\oplus \mathtt{9} \cdot \mathsf{A}^{(4)}[13] \oplus \mathtt{e} \cdot \mathsf{RK}^{(4)}[14]} \\
\widehat{\mathsf{SR}}^{(2)}[7] = \mathtt{b} \cdot k_{14} \oplus \mathtt{d} \cdot k_5 \oplus \mathtt{9} \cdot k_{20} \oplus \mathtt{e} \cdot k_{11} \boxed{\oplus \mathtt{b} \cdot \mathsf{A}^{(3)}[4] \oplus \mathtt{d} \cdot \mathsf{A}^{(3)}[5] \oplus \mathtt{9} \cdot \mathsf{A}^{(3)}[6] \oplus \mathtt{e} \cdot \mathsf{A}^{(3)}[7]} \\
\widehat{\mathsf{SR}}^{(2)}[13] = \mathtt{9} \cdot k_{13} \oplus \mathtt{e} \cdot k_4 \oplus \mathtt{b} \cdot k_{19} \boxed{\oplus \mathtt{9} \cdot \mathsf{A}^{(3)}[12] \oplus \mathtt{e} \cdot \mathsf{A}^{(3)}[13] \oplus \mathtt{b} \cdot \mathsf{A}^{(3)}[14] \oplus \mathtt{d} \cdot (\mathsf{A}^{(3)}[15] \oplus k_{10})}
\end{cases}
\tag{20}
$$

$$
\left(
\begin{array}{r|cccccccccccc|ccccccccc}
 & k_{18} & k_{17} & k_{22} & k_2 & k_6 & k_7 & k_{12} & k_{19} & k_1 & k_5 & k_{11} & k_{20} & k_0 & k_3 & k_4 & k_{13} & k_{14} & k_{15} & k_{16} & k_{21} & k_{23} \\
\hline
\widehat{\mathsf{SR}}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[3] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[6] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[9] & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[11] & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[12] & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
\widehat{\mathsf{K}}^{(3)}[18] & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{SR}}^{(3)}[14] & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{S}}^{(1)}[20] & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[0] & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(4)}[9] & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\widehat{\mathsf{K}}^{(0)}[10] & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(4)}[8] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
\widehat{\mathsf{SR}}^{(2)}[13] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(0)}[10] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{SR}}^{(2)}[7] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(4)}[11] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\widehat{\mathsf{S}}^{(4)}[8] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{array}
\right)
\tag{21}
$$

4.5 Two-Plaintext Key-Recovery Attack on 7-round AES-256

Figure 19 in Supplementary Material D is the 7-round MitM path, where the starting state $\mathsf{S}^{(1)}$ contains $\lambda^{+} = 1$ ■ bytes and $\lambda^{-} = 23$ ■ bytes. The consumed DoFs of ■ and ■ are $\ell^{+} = 0$ and $\ell^{-} = 22$ bytes, respectively. Therefore, $\mathrm{DoF}^{+} = 1$, $\mathrm{DoF}^{-} = 1$, and there is $\mathrm{DoM} = 1$ matching byte in round 3. Compute 22 expressions, given in Eq. (29) (by deleting the boxed parts) in Supplementary Material D, for the 22 consumed DoFs of ■ (marked by ■/■ in Figure 19) on the red bytes of $\mathsf{S}^{(1)}$. Using the new TA, we get 9 free variables

$\mathsf{S}^{(1)}[0,6,14,18,20,22,27,28,30]$ and 14 dependent variables $\mathsf{S}^{(1)}[1,2,3,4,5,7,9,10,11,16,17,21,29,31]$. The matrices after TA is given in Eq. (22). The steps for the 7-round MitM attack are given in Algorithm 11 in Supplementary Material D. The total time complexity is about $2^{256-8\cdot\min\{\mathrm{DoF}^+,\mathrm{DoF}^-,\mathrm{DoM}\}} = 2^{248}$. The memory is 2^{72} to store U.

$$\begin{array}{rc}
 & \begin{array}{cccccccccccccc|ccccccccc} k_5 & k_{21} & k_{17} & k_{11} & k_{16} & k_{10} & k_{31} & k_9 & k_{29} & k_1 & k_3 & k_4 & k_7 & k_2 & k_0 & k_6 & k_{14} & k_{18} & k_{20} & k_{22} & k_{27} & k_{28} & k_{30} \end{array} \\
\begin{array}{r}
\widehat{\mathsf{SR}}^{(4)}[1] \\ \widehat{\mathsf{SR}}^{(4)}[4] \\ \widehat{\mathsf{SR}}^{(4)}[11] \\ \widehat{\mathsf{SR}}^{(4)}[14] \\ \widehat{\mathsf{MC}}^{(1)}[4] \\ \widehat{\mathsf{MC}}^{(1)}[14] \\ \widehat{\mathsf{A}}^{(2)}[9] \\ \widehat{\mathsf{MC}}^{(0)}[14] \\
\widehat{\mathsf{MC}}^{(5)}[9] \\ \widehat{\mathsf{MC}}^{(5)}[11] \\ \widehat{\mathsf{MC}}^{(5)}[8] \\ \widehat{\mathsf{A}}^{(2)}[3] \\ \widehat{\mathsf{K}}^{(0)}[1] \\ \widehat{\mathsf{A}}^{(1)}[2] \\ \widehat{\mathsf{A}}^{(1)}[13] \\ \widehat{\mathsf{K}}^{(3)}[18] \\ \widehat{\mathsf{A}}^{(1)}[10] \\ \widehat{\mathsf{K}}^{(3)}[10] \\ \widehat{\mathsf{A}}^{(1)}[6] \\ \widehat{\mathsf{A}}^{(2)}[8] \\ \widehat{\mathsf{A}}^{(1)}[12] \\ \widehat{\mathsf{MC}}^{(0)}[15]
\end{array} &
\left(\begin{array}{cccccccccccccc|ccccccccc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
\hline
1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0
\end{array}\right)
\end{array} \tag{22}$$

4.6 Improved Preimage Attack on 10-round AES-256-DM

In addition to the key-recovery attacks, we also significantly reduce the memory complexity of Dong *et al.*'s 10-round preimage attack on DM hashing mode with AES-256 from the impractical 2^{56} [26] to the practical 2^8, with the same time complexity. The compression function of AES-256-DM is $h_i = \mathsf{AES\text{-}256}_{m_{i-1}}(h_{i-1}) \oplus h_{i-1}$, where h_i is the 128-bit chaining variable and the 256-bit message block m_{i-1} acts as the encryption key of AES-256. Given a 128-bit target h_i, the preimage attack is to generate a preimage (m_{i-1}, h_{i-1}) satisfying the target with time complexity lower than 2^{128}.

We reuse the 10-round MitM characteristic in [26] as shown in Figure 20 in Supplementary Material E. In the MitM path, the starting states are $\mathsf{A}^{(4)}$ and $\mathsf{S}^{(3)}$. The 1-byte matching occurs in the MC operation in round 8. In $\mathsf{S}^{(3)}$, there are 19 ■ cells, 1 ■ cell and 12 ■ cells. In $\mathsf{A}^{(4)}$, there are 8 ■ cells, 8 ■ cells. Hence, the total initial DoFs are $\lambda^+ = 19 + 8 = 27$ cells for ■ cells, and $\lambda^- = 1 + 8 = 9$ for ■ cells. The consumed DoFs of ■ and DoFs of ■ are $\ell^+ = 26$ and $\ell^- = 8$ bytes, respectively. Therefore, $\mathrm{DoF}^+ = 1$, $\mathrm{DoF}^- = 1$, and there is $\mathrm{DoM} = 1$ matching byte.

We obtain the 18 equations on ■ bytes of $\mathsf{S}^{(3)}$ for the consumed DoFs of blue bytes in $\mathsf{S}^{(1)}[13,14]$, $\mathsf{S}^{(2)}[12,13,14,15]$, $\mathsf{S}^{(4)}[1]$, $K^{(2)}[0,3,4,5,9]$, $\mathsf{SR}^{(1)}[3,6,9,12]$ and $\mathsf{SR}^{(9)}[6,12]$, where the expressions of $\mathsf{SR}^{(1)}[3,6,9,12]$ are obtained by $\mathsf{MC}^{-1}(\mathsf{RK}^{(1)})$, and assign $a_i (0 \le i < 18)$ to them. Using the new TA, we get 4 free variables $\mathsf{S}^{(3)}[8,16,18,24]$ and 15 dependent variables $\mathsf{S}^{(3)}[0,1,2,3,4,5,6,7,9,10,11,17,27,$

28, 31]. The matrices before and after TA are shown in Eq. (31) in Supplementary Material E, where the bytes of $\mathsf{S}^{(3)}$ are denoted as $k_0, k_1, \cdots, k_{31}$.

Since we try to find a 128-bit preimage, the 128-bit encryption data path and 256-bit key-schedule path provide enough degrees of freedom, we do not need to traverse all the cells to find the 128-bit preimage. The steps for the 10-round MitM attack are given in Algorithm 12 in Supplementary Material E. The total time complexity is about 2^{120}. The memory is 2^8 to store U.

Experiment of Preimage Attack. We conduct an experiment of the 5-byte partial target preimage attack by fixing the 5 target bytes $T[0, 1, 2, 6, 12] = 0$. According to Algorithm 12, we first fix all gray bytes ■ in $\mathsf{S}^{(3)}$ as 0 in Line 1, and 15 a_i as 0 in Line 2. Traverse 4 free variables ■ $\mathsf{S}^{(3)}[8, 16, 18, 24]$ and deduce the other 3 (a_8, a_{16}, a_{18}), only store the bytes that satisfy $a_8 = a_{16} = a_{18} = 0$, where about 2^8 values of blue bytes can be obtained and need 2^8 memory.

In Line 10 to 22, set all the ■ bytes in $\mathsf{MC}^{(3)}, \mathsf{MC}^{(4)}$ except $\mathsf{MC}^{(4)}[0]$ to 0 and traverse 3-byte $\mathsf{A}^{(5)}[2, 6]$ and $\mathsf{MC}^{(4)}[0]$, where 2^{32} time complexity is needed to get a preimage of the 5-byte partial target $T[0, 1, 2, 6, 12] = 0$. Obviously, to find a preimage of a 5-byte target, a brute-force search takes $2^{8 \cdot 5 = 40}$ time. The source codes and results are available via

https://github.com/boxindev/Triangulation-MitM

We deploy the experiment on a computer with an i9-13900KF CPU and 32GB memory. In each experiment, the time of precomputation from Line 4 to 8 is about 500 seconds. Then, the process from Line 10 to 22 takes about 35000 seconds, and 4 preimages are produced with $T[0, 1, 2, 6, 12] = 0$, which are listed in Table 3 in Supplementary Material E. The brute force search for 40-bit target preimage needs 2^{40} evaluations of 10-round AES-256, which takes about 5888000 seconds using the same AES-256 code in the same platform.

Our 4-round attack on AES-128 with 2^{24} memory in Sect. 4.2 is about 16 times faster than brute force, but the 10-round attack with 2^8 memory is about 168 times faster than brute force. The reason behind may be that accessing a larger memory needs more time in practical implementations. This phenomenon also proves that the attacks we proposed with significantly reduced memory complexities are very important in practical attacks.

5 Single-Plaintext Key-Recovery on Reduced Rijndael-EM

FAEST [5] additionally uses Rijndael in the Even-Mansour (EM) mode shown in Figure 9, *i.e.*, Rijndael-EM-128/-192/-256, where Rijndael with block sizes of 128/192/256 bits acts as the ideal permutations. Given a plaintext-ciphertext pair (P, C), suppose $X = P \oplus k$, then Rijndael-EM in Figure 9 can be transformed into Figure 10, *i.e.*, Rijndael $(X) \oplus X = P \oplus C$. Therefore, the key-recovery problem turns into the preimage attack on DM-like hashing mode, *i.e.*, given the target $P \oplus C$ and find the preimage X. Then, find the key $k = X \oplus P$.

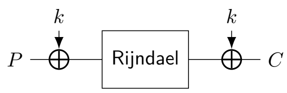

Fig. 9: Rijndael-EM

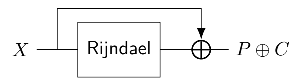

Fig. 10: Equivalent Form

5.1 Key-Recovery Attack on 7-round Rijndael-EM-128

The 7-round MitM characteristic is shown in Figure 11. The starting states $\mathsf{A}^{(1)}$ contains $\lambda^+ = 6$ bytes and $\lambda^- = 2$ bytes. In the computation from $\mathsf{A}^{(1)}$ to $\mathsf{SR}^{(2)}$ and $\mathsf{MC}^{(2)}$, the consumed DoFs of and are $\ell^+ = 4$ and $\ell^- = 0$ bytes, respectively. Therefore, $\mathrm{DoF}^+ = 2,\ \mathrm{DoF}^- = 2$, and there are $\mathrm{DoM} = 4$ matching bytes in round 2. The steps for the 7-round MitM attack are given in Algorithm 7. The time complexity is about $2^{128-8\cdot\min\{\mathrm{DoF}^+,\mathrm{DoF}^-,\mathrm{DoM}\}} = 2^{112}$. The memory is 2^{32} to store U.

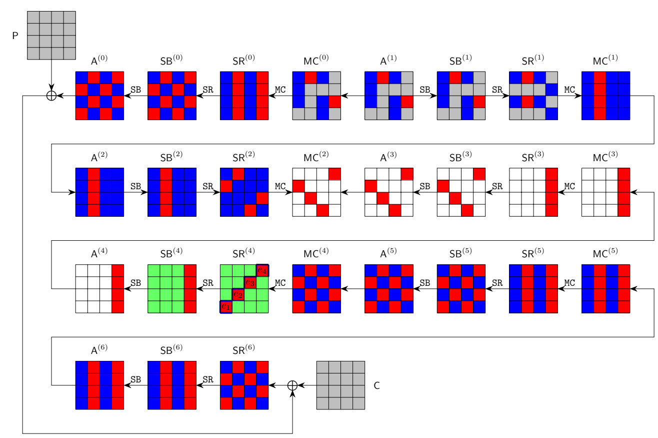

Fig. 11: The 7-round attack on Rijndael-EM-128

5.2 Key-Recovery Attack on 8-round Rijndael-EM-192

The 8-round MitM characteristic is shown in Figure 12. The starting state $\mathsf{A}^{(1)}$ contains $\lambda^+ = 4$ bytes and $\lambda^- = 8$ bytes. In the computation from $\mathsf{A}^{(1)}$ to $\mathsf{SR}^{(4)}$ and $\mathsf{MC}^{(4)}$, the consumed DoFs of and are $\ell^+ = 2$ and $\ell^- = 6$ bytes, respectively. Therefore, $\mathrm{DoF}^+ = 2,\ \mathrm{DoF}^- = 2$, and there are $\mathrm{DoM} = 2$ matching bytes.

The steps for the 8-round MitM attack are given in Algorithm 8. The time complexity is about $2^{192-8\cdot\min\{\mathrm{DoF}^+,\mathrm{DoF}^-,\mathrm{DoM}\}} = 2^{176}$. The memory is 2^{16} to store the table L.

Algorithm 7: Attack on 7-round Rijndael-EM-128

for $\mathsf{A}^{(1)}[5,6,7,12,13,15] \in \mathbb{F}_2^{48}$ **do**
for $(e_1, e_2, e_3, e_4) \in \mathbb{F}_2^{32}$ **do**
$U \leftarrow [\ ]$
for $\mathsf{MC}^{(4)}[0,5,8,13] \in \mathbb{F}_2^{32}$ **do**
Compute $\mathsf{MC}^{(4)}[2,7,10,15]$ according to (e_1, e_2, e_3, e_4)
/* e.g., $\mathtt{b} \cdot \mathsf{MC}^{(4)}[0] \oplus \mathtt{9} \cdot \mathsf{MC}^{(4)}[2] = e_1$ */
Compute $\mathbf{u} = \mathsf{MC}^{(0)}[3,9] \in \mathbb{F}_2^{16}$
$U[\mathbf{u}] \leftarrow \mathsf{MC}^{(4)}[0,2,5,7,8,10,13,15]$
end
for $\mathbf{u} \in \mathbb{F}_2^{16}$ **do**
$L \leftarrow [\ ]$
for $\mathsf{A}^{(1)}[4,14] \in \mathbb{F}_2^{16}$ **do**
Compute the 4-byte matching point v, $L[v] \leftarrow \mathsf{A}^{(1)}[4,14]$
end
for *values in* $U[\mathbf{u}]$ **do**
Compute the 4-byte matching point v'
for *values in* $L[v']$ **do**
Check if the target $P \oplus C$ is satisfied
end
end
end
end
end

Algorithm 8: Attack on 8-round Rijndael-EM-192

for $\mathsf{A}^{(1)}[5,6,7,10,11,12,15,16,17,20,21,22] \in \mathbb{F}_2^{96}$ **do**
for $\mathsf{SR}^{(0)}[1,2] \in \mathbb{F}_2^{16}$ **do**
for $\mathsf{MC}^{(1)}[4,7,8,9] \in \mathbb{F}_2^{32}$ **do**
for $(e_1, e_2) \in \mathbb{F}_2^{16}$ **do**
$L \leftarrow [\ ]$
for $\mathsf{A}^{(1)}[0,1] \in \mathbb{F}_2^{16}$ **do**
Compute $\mathsf{A}^{(1)}[2,3]$ according to $\mathsf{SR}^{(0)}[1,2]$
Compute the 2-byte matching point v
$L[v] \leftarrow A^{(1)}[0,1,2,3]$
end
for $\mathsf{SR}^{(2)}[1,22] \in \mathbb{F}_2^{16}$ **do**
Compute $\mathsf{SR}^{(2)}[2,23]$ according to (e_1, e_2)
/* e.g., $\mathtt{3} \cdot \mathsf{SR}^{(2)}[1] \oplus \mathsf{SR}^{(2)}[2] = e_1$ */
Compute the 2-byte matching point v'
for *values in* $L[v']$ **do**
Check if the target $P \oplus C$ is satisfied
end
end
end
end
end
end

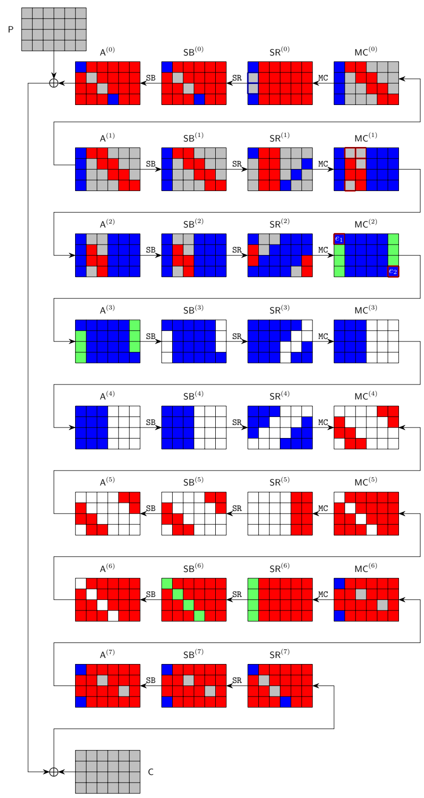

Fig. 12: 8-round Rijndael-EM-192

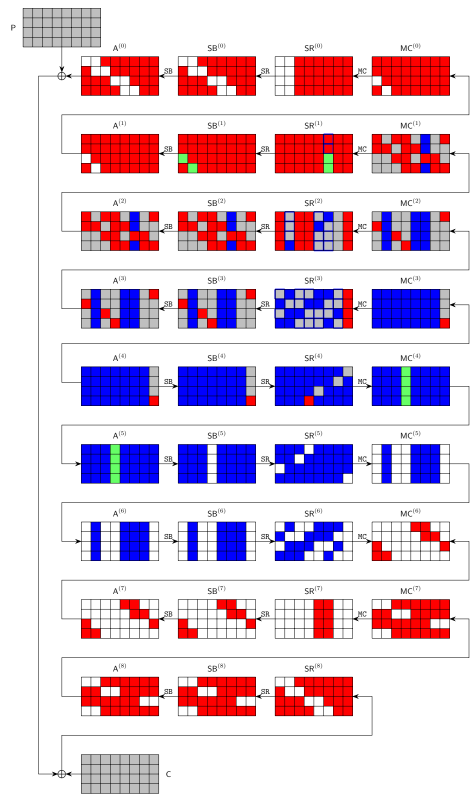

Fig. 13: 9-round Rijndael-EM-256

5.3 Key-Recovery Attack on 9-round Rijndael-EM-256

The 9-round MitM characteristic is shown in Figure 13. The starting state $\mathsf{A}^{(4)}$ contains $\lambda^+ = 28$ ■ bytes and $\lambda^- = 1$ ■ bytes. In the computation from $\mathsf{A}^{(4)}$ to $\mathsf{SR}^{(6)}$ and $\mathsf{MC}^{(6)}$, the consumed degrees of freedom (DoFs) of ■ and ■ are $\ell^+ = 27$ and $\ell^- = 0$ bytes, respectively. Therefore, $\mathrm{DoF}^+ = 1,\ \mathrm{DoF}^- = 1$, and there is $\mathrm{DoM} = 1$ matching byte.

The steps for the 9-round MitM attack are given in Algorithm 9. The time complexity is about $2^{256-8\cdot\min\{\mathrm{DoF}^+,\mathrm{DoF}^-,\mathrm{DoM}\}} = 2^{248}$. The memory is 2^8 to store table L.

Algorithm 9: Attack on 9-round Rijndael-EM-256

---

for $\mathsf{A}^{(4)}[28, 29, 30] \in \mathbb{F}_2^{24}$ **do**
    for $\mathsf{SR}^{(1)}[20, 21] \in \mathbb{F}_2^{16}$ **do**
        for $\mathsf{SR}^{(2)}[4, 5, 6, 16, 18, 19, 21, 22, 23] \in \mathbb{F}_2^{72}$ **do**
            for $\mathsf{SR}^{(3)}[0, 2, 5, 8, 9, 11, 12, 14, 15, 18, 19, 21, 22, 24, 25, 27] \in \mathbb{F}_2^{128}$ **do**
                $L \leftarrow [\,]$
                for $\mathsf{MC}^{(1)}[20] \in \mathbb{F}_2^{8}$ **do**
                    Compute $\mathsf{MC}^{(1)}[21, 23]$ according to $\mathsf{SR}^{(1)}[20, 21]$
                    Compute forward and backward to the 1-byte matching point v
                    $L[v] \leftarrow \mathsf{MC}^{(1)}[20, 21, 23]$
                end
                for $\mathsf{A}^{(4)}[31] \in \mathbb{F}_2^{8}$ **do**
                    Compute forward and backward to the 2-byte matching point v'
                    for *values in* $L[v']$ **do**
                        Check if $E(Key = \mathsf{A}^{(0)} \oplus P, P) = C$
                    end
                end
            end
        end
    end
end

---

6 Further Applications and Conclusion

This paper introduces the *triangulating Meet-in-the-Middle attack* to reduce the memory complexity when considering MitM paths with nonlinearly constrained neutral words. We achieve this goal by leveraging and improving the Triangulation Algorithm (TA) of Khovratovich *et al.* to solve nonlinear systems of the MitM efficiently.

For AES, we reduce the memory complexities of the 4-/5-round single-plaintext key-recovery attacks on AES-128 and propose new attacks on 6-round AES-192 and 7-round AES-256. For Rijndael-EM, we convert key-recovery attack into preimage attack on its hashing mode and give new key recovery attacks Rijndael-EM-128/192/256 with a single plaintext-ciphertext pair.

For sponge functions like Keccak[1024], Keccak[768], Xoodyak-XOF, Ascon-XOF, Gimli-XOF, and Subterranean 2.0, we either reduce the memory complexity of existing attacks or introduce new attacks with better time complexity in Supplementary Material G, H, I, J, and K.

Acknowledgements. We would like to thank Yu Sasaki and the anonymous reviewers from CRYPTO 2025 for their excellent guidance in improving the paper. This work is supported by the National Key R&D Program of China (2024YFA1013000), the Natural Science Foundation of China (62272257, 62302250), the Young Elite Scientists Sponsorship Program by CAST (2023QNRC001), and the Zhongguancun Laboratory.

References

1. Kazumaro Aoki and Yu Sasaki. Preimage attacks on one-block MD4, 63-step MD5 and more. In *SAC 2008*, volume 5381, pages 103–119. Springer, 2008.
2. Zhenzhen Bao, Xiaoyang Dong, Jian Guo, Zheng Li, Danping Shi, Siwei Sun, and Xiaoyun Wang. Automatic search of meet-in-the-middle preimage attacks on AES-like hashing. In *EUROCRYPT 2021, Part I*, volume 12696, pages 771–804.
3. Zhenzhen Bao, Jian Guo, Danping Shi, and Yi Tu. Superposition meet-in-the-middle attacks: Updates on fundamental security of AES-like hashing. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Proceedings, Part I*, volume 13507 of *LNCS*, pages 64–93. Springer, 2022.
4. Achiya Bar-On, Orr Dunkelman, Nathan Keller, Eyal Ronen, and Adi Shamir. Improved key recovery attacks on reduced-round AES with practical data and memory complexities. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 185–212. Springer, 2018.
5. Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Christian Majenz, Shibam Mukherjee, Emmanuela Orsini, Sebastian Ramacher, Christian Rechberger, Lawrence Roy, et al. Faest: algorithm specifications. Technical report, Technical report, National Institute of Standards and Technology, 2023.
6. Edward A. Bender and E. Rodney Canfield. An approximate probabilistic model for structured gaussian elimination. *J. Algorithms*, 31(2):271–290, 1999.
7. Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Keccak sponge function family main document. *Submission to NIST*, 3(30):320–337, 2009.
8. Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biclique cryptanalysis of the full AES. In *ASIACRYPT 2011, Proceedings*, pages 344–371.
9. Andrey Bogdanov and Christian Rechberger. A 3-subset meet-in-the-middle attack: Cryptanalysis of the lightweight block cipher KTANTAN. In *SAC 2010*, volume 6544, pages 229–240. Springer, 2010.
10. Charles Bouillaguet, Patrick Derbez, Orr Dunkelman, Pierre-Alain Fouque, Nathan Keller, and Vincent Rijmen. Low-data complexity attacks on aes. *IEEE transactions on information theory*, 58(11):7002–7017, 2012.
11. Charles Bouillaguet, Patrick Derbez, and Pierre-Alain Fouque. Automatic search of attacks on round-reduced AES and applications. In Phillip Rogaway, editor, *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture Notes in Computer Science*, pages 169–187. Springer, 2011.

12. Christina Boura, Nicolas David, Patrick Derbez, Gregor Leander, and María Naya-Plasencia. Differential meet-in-the-middle cryptanalysis. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Proceedings, Part III*, volume 14083 of *Lecture Notes in Computer Science*, pages 240–272. Springer, 2023.
13. Anne Canteaut, María Naya-Plasencia, and Bastien Vayssière. Sieve-in-the-middle: Improved MITM attacks. In *CRYPTO 2013, Proceedings, Part I*, pages 222–240.
14. Shiyao Chen, Jian Guo, Eik List, Danping Shi, and Tianyu Zhang. Diving deep into the preimage security of aes-like hashing. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part I*, volume 14651 of *Lecture Notes in Computer Science*, pages 398–426. Springer, 2024.
15. Joan Daemen, Seth Hoffert, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. Xoodyak, a lightweight cryptographic scheme. *IACR Trans. Symmetric Cryptol.*, 2020(S1):60–87, 2020.
16. Joan Daemen, Pedro Maat Costa Massolino, Alireza Mehrdad, and Yann Rotella. The subterranean 2.0 cipher suite. *IACR Trans. Symmetric Cryptol.*, 2020(S1):262–294, 2020.
17. Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.
18. Mathieu Degré, Patrick Derbez, Lucie Lahaye, and André Schrottenloher. New models for the cryptanalysis of ASCON. *IACR Cryptol. ePrint Arch.*, page 298, 2024.
19. Patrick Derbez. *Meet-in-the-middle attacks on AES*. PhD thesis, Ecole Normale Supérieure de Paris-ENS Paris, 2013.
20. Patrick Derbez and Pierre-Alain Fouque. Automatic search of meet-in-the-middle and impossible differential attacks. In *CRYPTO 2016, Proceedings, Part II*, volume 9815, pages 157–184. Springer, 2016.
21. Whitfield Diffie and Martin E. Hellman. Special feature exhaustive cryptanalysis of the NBS data encryption standard. *Computer*, 10(6):74–84, 1977.
22. Itai Dinur. Cryptanalytic applications of the polynomial method for solving multivariate equation systems over GF(2). In *EUROCRYPT 2021, Proceedings, Part I*, volume 12696, pages 374–403. Springer, 2021.
23. Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Efficient dissection of composite problems, with applications to cryptanalysis, knapsacks, and combinatorial search problems. In *CRYPTO 2012*, volume 7417, pages 719–740.
24. Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1.2: Lightweight authenticated encryption and hashing. *J. Cryptol.*, 34(3):33, 2021.
25. Xiaoyang Dong, Jian Guo, Shun Li, and Phuong Pham. Triangulating rebound attack on aes-like hashing. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part I*, volume 13507 of *Lecture Notes in Computer Science*, pages 94–124. Springer, 2022.
26. Xiaoyang Dong, Jialiang Hua, Siwei Sun, Zheng Li, Xiaoyun Wang, and Lei Hu. Meet-in-the-middle attacks revisited: Key-recovery, collision, and preimage attacks. In *CRYPTO 2021, Proceedings, Part III*, volume 12827, pages 278–308. Springer.
27. Xiaoyang Dong, Boxin Zhao, Lingyue Qin, Qingliang Hou, Shun Zhang, and Xiaoyun Wang. Generic mitm attack frameworks on sponge constructions. In Leonid

Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part IV*, volume 14923 of *Lecture Notes in Computer Science*, pages 3–37. Springer, 2024.
28. Orr Dunkelman, Nathan Keller, Eyal Ronen, and Adi Shamir. The retracing boomerang attack, with application to reduced-round AES. *J. Cryptol.*, 37(3):32, 2024.
29. Orr Dunkelman, Gautham Sekar, and Bart Preneel. Improved meet-in-the-middle attacks on reduced-round DES. In *INDOCRYPT 2007, Proceedings*, volume 4859, pages 86–100. Springer, 2007.
30. Thomas Espitau, Pierre-Alain Fouque, and Pierre Karpman. Higher-order differential meet-in-the-middle preimage attacks on SHA-1 and BLAKE. In *CRYPTO 2015, Proceedings, Part I*, volume 9215, pages 683–701. Springer, 2015.
31. Thomas Fuhr and Brice Minaud. Match box meet-in-the-middle attack against KATAN. In *FSE 2014*, pages 61–81, 2014.
32. Jian Guo, San Ling, Christian Rechberger, and Huaxiong Wang. Advanced meet-in-the-middle preimage attacks: First results on full Tiger, and improved results on MD4 and SHA-2. In *ASIACRYPT 2010, Proceedings*, volume 6477, pages 56–75.
33. Hosein Hadipour and Maria Eichlseder. Autoguess: A tool for finding guess-and-determine attacks and key bridges. In Giuseppe Ateniese and Daniele Venturi, editors, *Applied Cryptography and Network Security - 20th International Conference, ACNS 2022, Rome, Italy, June 20-23, 2022, Proceedings*, volume 13269 of *Lecture Notes in Computer Science*, pages 230–250. Springer, 2022.
34. Dmitry Khovratovich, Alex Biryukov, and Ivica Nikolic. Speeding up collision search for byte-oriented hash functions. In Marc Fischlin, editor, *Topics in Cryptology - CT-RSA 2009, The Cryptographers' Track at the RSA Conference 2009, San Francisco, CA, USA, April 20-24, 2009. Proceedings*, volume 5473 of *Lecture Notes in Computer Science*, pages 164–181. Springer, 2009.
35. Dmitry Khovratovich, Gaëtan Leurent, and Christian Rechberger. Narrow-bicliques: Cryptanalysis of full IDEA. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012, Proceedings*, volume 7237, pages 392–410.
36. Simon Knellwolf and Dmitry Khovratovich. New preimage attacks against reduced SHA-1. In *CRYPTO 2012, Proceedings*, volume 7417, pages 367–383.
37. Brian A. LaMacchia and Andrew M. Odlyzko. Solving large sparse linear systems over finite fields. In Alfred Menezes and Scott A. Vanstone, editors, *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, volume 537 of *Lecture Notes in Computer Science*, pages 109–133. Springer, 1990.
38. Charlotte Lefevre and Bart Mennink. Tight preimage resistance of the sponge construction. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part IV*, volume 13510 of *Lecture Notes in Computer Science*, pages 185–204. Springer, 2022.
39. Gaëtan Leurent and Clara Pernot. New Representations of the AES Key Schedule. Cryptology ePrint Archive, Report 2020/1253, 2020.
40. Gaëtan Leurent and Clara Pernot. New representations of the AES key schedule. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021,*

Proceedings, Part I, volume 12696 of *Lecture Notes in Computer Science*, pages 54–84. Springer, 2021.
41. Fukang Liu, Takanori Isobe, and Willi Meier. Exploiting weak diffusion of gimli: Improved distinguishers and preimage attacks. *IACR Trans. Symmetric Cryptol.*, 2021(1):185–216, 2021.
42. Fukang Liu, Takanori Isobe, Willi Meier, and Zhonghao Yang. Algebraic attacks on round-reduced Keccak. In *ACISP 2021, Proceedings*, volume 13083, pages 91–110.
43. Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. The rebound attack: Cryptanalysis of reduced whirlpool and grøstl. In *FSE 2009, 2009*, pages 260–276, 2009.
44. Pawel Morawiecki, Josef Pieprzyk, and Marian Srebrny. Rotational cryptanalysis of round-reduced Keccak. In *FSE 2013,*, volume 8424, pages 241–262.
45. Lingyue Qin, Jialiang Hua, Xiaoyang Dong, Hailun Yan, and Xiaoyun Wang. Meet-in-the-middle preimage attacks on sponge-based hashing. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part IV*, volume 14007 of *Lecture Notes in Computer Science*, pages 158–188. Springer, 2023.
46. Lingyue Qin, Jialiang Hua, Xiaoyang Dong, Hailun Yan, and Xiaoyun Wang. Meet-in-the-middle preimage attacks on sponge-based hashing. In *EUROCRYPT 2023, Proceedings, Part IV*, volume 14007, pages 158–188. Springer, 2023.
47. Sondre Rønjom, Navid Ghaedi Bardeh, and Tor Helleseth. Yoyo tricks with AES. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I*, volume 10624 of *Lecture Notes in Computer Science*, pages 217–243. Springer, 2017.
48. Yu Sasaki. Integer linear programming for three-subset meet-in-the-middle attacks: Application to GIFT. In *IWSEC 2018*, volume 11049, pages 227–243.
49. Yu Sasaki. Meet-in-the-middle preimage attacks on AES hashing modes and an application to whirlpool. In Antoine Joux, editor, *Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers*, volume 6733 of *Lecture Notes in Computer Science*, pages 378–396. Springer, 2011.
50. Yu Sasaki and Kazumaro Aoki. Finding preimages in full MD5 faster than exhaustive search. In *EUROCRYPT 2009, Proceedings*, volume 5479, pages 134–152.
51. Yu Sasaki, Lei Wang, Shuang Wu, and Wenling Wu. Investigating fundamental security requirements on whirlpool: Improved preimage and collision attacks. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *Lecture Notes in Computer Science*, pages 562–579. Springer, 2012.
52. André Schrottenloher and Marc Stevens. Simplified MITM modeling for permutations: New (quantum) attacks. In *CRYPTO 2022, Proceedings, Part III*, volume 13509, pages 717–747. Springer, 2022.
53. André Schrottenloher and Marc Stevens. Simplified modeling of MITM attacks for block ciphers: new (quantum) attacks. *IACR Cryptol. ePrint Arch.*, page 816, 2023.
54. Tyge Tiessen. Polytopic cryptanalysis. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International*

Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I, volume 9665 of *Lecture Notes in Computer Science*, pages 214–239. Springer, 2016.

55. Michael Tunstall. Improved "partial sums"-based square attack on AES. In Pierangela Samarati, Wenjing Lou, and Jianying Zhou, editors, *SECRYPT 2012 - Proceedings of the International Conference on Security and Cryptography, Rome, Italy, 24-27 July, 2012, SECRYPT is part of ICETE - The International Joint Conference on e-Business and Telecommunications*, pages 25–34. SciTePress, 2012.

Supplementary Material

A Sasaki's 7-round MitM attack on AES

Sasaki presented a Meet-in-the-Middle (MitM) preimage attack targeting 7-round AES hashing in [49]. The method leverages chunk separation, forward-backward computation, and partial matches to achieve efficient preimage recovery.

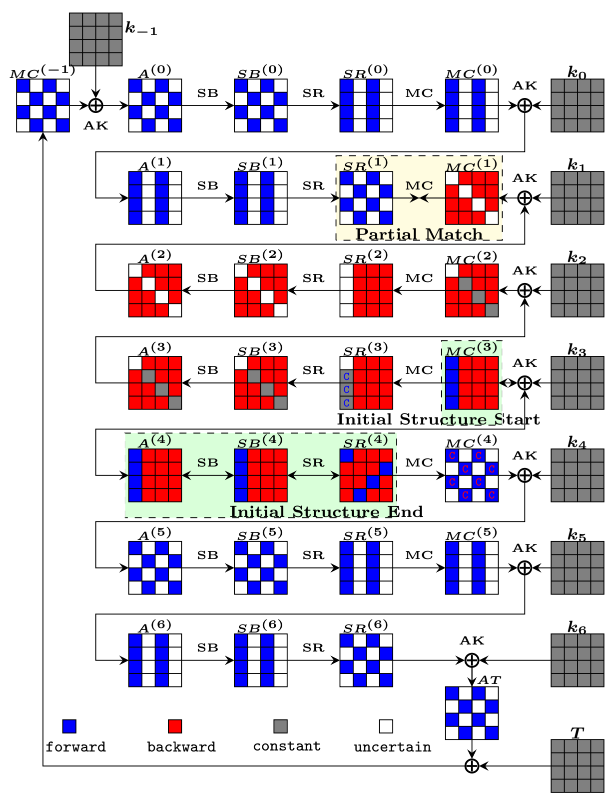

Fig. 14: The MitM preimage attack on 7-round AES-hashing.

Internal State Representation and Operations

Forward and Backward Chunk Separation As shown in Figure 14, the MitM attack divides the computations into two chunks, starts at the state $\mathsf{MC}^{(3)}$, traverses through $\mathsf{A}^{(4)}$, $\mathsf{SB}^{(4)}$, and stops at $\mathsf{SR}^{(4)}$.

The $\mathsf{MC}^{(3)}[0-3]$ are chosen as neutral bytes (marked by blue) for the forward chunk and the $\mathsf{SR}^{(4)}[1-6,8,9,11,12,14,15]$ (marked by red) are chosen as neutral bytes for the backward chunk. The intersection of the forward and backward computations occurs at states $\mathsf{SR}^{(1)}$ and $\mathsf{MC}^{(1)}$, where a partial match is verified.

Constraints on the Initial Structure The constraints are applied to the neutral bytes in the forward computation to ensure the two chunks operate independently. Specifically, three constraints are added to the bytes $\mathsf{MC}^{(3)}[0-3]$ to prevent interference with the backward propagation.

The constants for $\mathsf{SR}^{(3)}_{\{1,2,3\}}$ are precomputed as follows:

$$\begin{bmatrix} \mathsf{c}_0 = 9\cdot\mathsf{MC}^{(3)}[0]\oplus e\cdot\mathsf{MC}^{(3)}[1]\oplus b\cdot\mathsf{MC}^{(3)}[2]\oplus d\cdot\mathsf{MC}^{(3)}[3] \\ \mathsf{c}_1 = d\cdot\mathsf{MC}^{(3)}[0]\oplus 9\cdot\mathsf{MC}^{(3)}[1]\oplus e\cdot\mathsf{MC}^{(3)}[2]\oplus b\cdot\mathsf{MC}^{(3)}[3] \\ \mathsf{c}_2 = b\cdot\mathsf{MC}^{(3)}[0]\oplus d\cdot\mathsf{MC}^{(3)}[1]\oplus 9\cdot\mathsf{MC}^{(3)}[2]\oplus e\cdot\mathsf{MC}^{(3)}[3] \end{bmatrix}. \tag{23}$$

There are 2^8 possibilities for the values of $\mathsf{MC}^{(3)}[0-3]$ given the constraints. Similarly, the backward chunk imposes 8 additional constraints on the bytes $\mathsf{MC}^{(4)}[0,2,5,7,8,10,13,15]$.

Partial Matching Through MixColumns The MitM approach relies on the MixColumns operation to validate matches. Since each column in $\mathsf{SR}^{(1)}$ and $\mathsf{MC}^{(1)}$ contains five known bytes, one match is derived per column.

For example, the first column match involves deriving $\mathsf{SR}^{(1)}[0,2]$ in the forward direction and $\mathsf{MC}^{(1)}[1,2,3]$ in the backward computation. The relationship can be expressed as:

$$\begin{aligned} & d\cdot SR^{(1)}[0]\oplus e\cdot SR^{(1)}[2] \\ =& d\cdot(b\cdot\mathsf{MC}^{(1)}[1]\oplus d\cdot\mathsf{MC}^{(1)}[2]\oplus 9\cdot\mathsf{MC}^{(1)}[3])\oplus e\cdot(9\cdot\mathsf{MC}^{(1)}[1] \\ & \oplus e\cdot\mathsf{MC}^{(1)}[2]\oplus b\cdot\mathsf{MC}^{(1)}[3]). \end{aligned} \tag{24}$$

Forward and Backward Computations

1. **Forward Computation:** The neutral bytes from $\mathsf{MC}^{(3)}$ propagate forward to $\mathsf{SR}^{(1)}$. By traversing all 2^8 possibilities of $\mathsf{MC}^{(3)}[0-3]$, the resulting values of $\mathsf{SR}^{(1)}$ are stored in table L_1, by the value of $\mathsf{SR}^{(1)}$ as the left part of Eq.(24) ($i.e.$ $d\cdot SR^{(1)}[0]\oplus e\cdot SR^{(1)}[2]$).
2. **Backward Computation:** Similarly, the red neutral bytes from $\mathsf{SR}^{(4)}$ propagate backward to $\mathsf{MC}^{(1)}$. These values are stored in table L_2, indexed by $\mathsf{MC}^{(1)}$ as the right part of Eq. (24).
3. **Partial Matching:** The two tables L_1 and L_2 are compared to identify 32-bit partial matches based on the indices, allowing the preimage to be recovered efficiently.

B New Representation of AES's Key Schedule

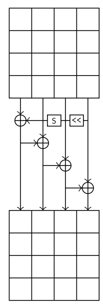

Fig. 15: AES-128

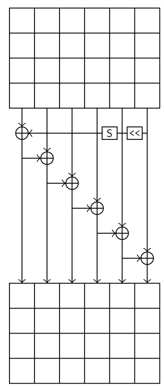

Fig. 16: AES-192

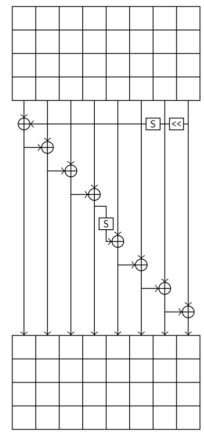

Fig. 17: AES-256

$$\mathsf{MC} = \begin{bmatrix} \mathsf{2} & \mathsf{3} & \mathsf{1} & \mathsf{1} \\ \mathsf{1} & \mathsf{2} & \mathsf{3} & \mathsf{1} \\ \mathsf{1} & \mathsf{1} & \mathsf{2} & \mathsf{3} \\ \mathsf{3} & \mathsf{1} & \mathsf{1} & \mathsf{2} \end{bmatrix} \qquad \mathsf{MC}^{-1} = \begin{bmatrix} \mathsf{e} & \mathsf{b} & \mathsf{d} & \mathsf{9} \\ \mathsf{9} & \mathsf{e} & \mathsf{b} & \mathsf{d} \\ \mathsf{d} & \mathsf{9} & \mathsf{e} & \mathsf{b} \\ \mathsf{b} & \mathsf{d} & \mathsf{9} & \mathsf{e} \end{bmatrix} \tag{25}$$

At EUROCRYPT 2021, Leurent and Pernot introduced the new representation of AES's key schedule [40]. For AES-128, denote the 16-byte round key by $\mathsf{RK}^{(i)} = (a_0, a_1, \cdots, a_{15})$. Leurent *et al.* derived a new basis $\mathsf{S}^{(i)} = (s_0, s_1, \cdots, s_{15})$. The relations between $\mathsf{RK}^{(i)}$ and $\mathsf{S}^{(i)}$ are $\mathsf{S}^{(i)} = A \cdot (\mathsf{RK}^{(i)})^\dagger$ and $\mathsf{RK}^{(i)} = C_0 \cdot (\mathsf{S}^{(i)})^\dagger$, where A is a matrix in Eq. (26) and $C_0 = A^{-1}$. Denote the state of next round by $\mathsf{S}^{(i)} = (s'_0, s'_1, \cdots, s'_{15})$, the update process is illustrated in Figure 18, and the details listed in Eq. (27). For more details of AES-192/256, please refer to [40].

$$\mathrm{A} = \begin{pmatrix}
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0
\end{pmatrix}
C_0 = \begin{pmatrix}
0&0&0&1&0&0&1&0&0&1&0&0&1&0&0&0\\
0&0&1&0&0&1&0&0&1&0&0&0&0&0&0&1\\
0&1&0&0&1&0&0&0&0&0&0&1&0&0&1&0\\
1&0&0&0&0&0&0&1&0&0&1&0&0&1&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0\\
0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0\\
1&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1\\
0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0\\
1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{pmatrix}
(\mathsf{S}^{(i)})^\dagger = \begin{pmatrix} s_0\\ s_1\\ s_2\\ s_3\\ s_4\\ s_5\\ s_6\\ s_7\\ s_8\\ s_9\\ s_{10}\\ s_{11}\\ s_{12}\\ s_{13}\\ s_{14}\\ s_{15} \end{pmatrix} \tag{26}$$

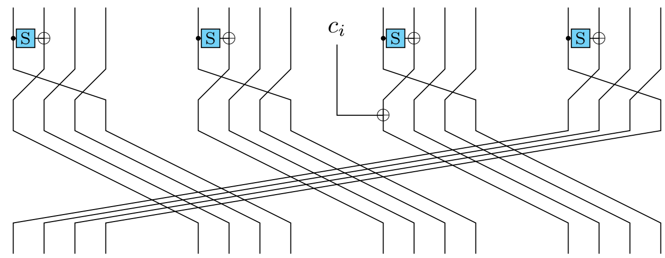

Fig. 18: New Representations of AES-128's key schedule [40]

$$
\begin{cases}
s'_0 = s_{13} \oplus S(s_{12}) & s'_8 = s_5 \oplus S(s_4) \\
s'_1 = s_{14} & s'_9 = s_6 \\
s'_2 = s_{15} & s'_{10} = s_7 \\
s'_3 = s_{12} & s'_{11} = s_4 \\
s'_4 = s_1 \oplus S(s_0) & s'_{12} = s_9 \oplus S(s_8) \oplus c_i \\
s'_5 = s_2 & s'_{13} = s_{10} \\
s'_6 = s_3 & s'_{14} = s_{11} \\
s'_7 = s_0 & s'_{15} = s_8
\end{cases}
\tag{27}
$$

C Key-Recovery Attack on 6-round AES-192 with Two Plaintext-Ciphertext Pairs

The 6-round MitM characteristic is shown in Figure 8. Denote the bytes of the starting state $\mathsf{S}^{(3)}$ by $k_0, k_1, \cdots, k_{23}$. The 19 expressions on the red bytes of $\mathsf{S}^{(3)}$ for the 19 consumed DoFs of ■ are shown as Eq. (20) (deleting the boxed parts). The matrices before and after TA are shown in Eq. (28). The steps for the 6-round MitM attack are given in Algorithm 10.

$$
\left(
\begin{array}{r|ccccccccccccccccccccc}
 & k_0 & k_1 & k_2 & k_3 & k_4 & k_5 & k_6 & k_7 & k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} & k_{17} & k_{18} & k_{19} & k_{20} & k_{21} & k_{22} & k_{23} \\
\hline
\widehat{\mathsf{S}}^{(1)}[20] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
\widehat{\mathsf{S}}^{(4)}[8] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
\widehat{\mathsf{K}}^{(0)}[10] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
\widehat{\mathsf{K}}^{(3)}[18] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(0)}[10] & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{A}}^{(1)}[0] & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(4)}[8] & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(4)}[9] & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(4)}[11] & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
\widehat{\mathsf{SR}}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[3] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[6] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[9] & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[11] & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[12] & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[14] & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(2)}[7] & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\widehat{\mathsf{SR}}^{(2)}[13] & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\end{array}
\right)
$$

(a)

$$
\left(
\begin{array}{r|cccccccccccc|ccccccccc}
 & k_{18} & k_{17} & k_{22} & k_2 & k_6 & k_7 & k_{12} & k_{19} & k_1 & k_5 & k_{11} & k_{20} & k_0 & k_3 & k_4 & k_{13} & k_{14} & k_{15} & k_{16} & k_{21} & k_{23} \\
\hline
\widehat{\mathsf{SR}}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[3] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[6] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[9] & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[11] & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(3)}[12] & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
\widehat{\mathsf{K}}^{(3)}[18] & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{SR}}^{(3)}[14] & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{S}}^{(1)}[20] & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[0] & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(4)}[9] & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\widehat{\mathsf{K}}^{(0)}[10] & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(4)}[8] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
\widehat{\mathsf{SR}}^{(2)}[13] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(0)}[10] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{SR}}^{(2)}[7] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(4)}[11] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\widehat{\mathsf{S}}^{(4)}[8] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{array}
\right)
\tag{28}
$$

(b)

D Key-Recovery Attack on 7-round AES-256 with Two Plaintext-Ciphertext Pairs

The 7-round MitM characteristic is shown in Figure 19. The bytes of the starting state $\mathsf{S}^{(1)}$ are denoted as $k_0, k_1, \cdots, k_{31}$. The 22 expressions on the red bytes of $\mathsf{S}^{(1)}$ for the 22 consumed DoFs of ■ are shown as Eq. (29) (deleting the boxed parts). Using TA, we can get 9 free variables $\mathsf{S}^{(1)}[0, 6, 14, 18, 20, 22, 27, 28, 30]$ and 14 dependent variables $\mathsf{S}^{(1)}[1, 2, 3, 4, 5, 7, 9, 10, 11, 16, 17, 21, 29, 31]$, the matrices before and after TA are shown in Eq. (30). The steps for the 7-round MitM attack are given in Algorithm 11.

Algorithm 10: Attack on 6-round $\mathsf{AES\text{-}192}$ with 2 (P, C)

for 2^{ζ} *values of* $\mathsf{S}^{(3)}[10] \in \mathbb{F}_2^8$ **do**
    for $(e_1, e_2, \cdots, e_{12}) \in \mathbb{F}_2^{96}$ **do**
        $U \leftarrow [\,]$
        for $\mathsf{S}^{(3)}[0, 3, 4, 13, 14, 15, 16, 21, 23] \in \mathbb{F}_2^{72}$ **do**
            Assign $(e_1, e_2, \cdots, e_{12})$ to the last 12 expressions in Eq. (21) (b)
            Deduce $\mathsf{S}^{(3)}[1, 2, 5, 6, 7, 11, 12, 17, 18, 19, 20, 22]$
            Compute $\mathbf{u} = \widehat{\mathsf{SR}}^{(3)}[1, 3, 4, 6, 9, 11, 12] \in \mathbb{F}_2^{8\times 7}$
            $U[\mathbf{u}] \leftarrow \mathsf{S}^{(3)}[v_{\mathcal{R}}]$ `/*` $v_{\mathcal{R}}$ `represents all 21 red indexes */`
        end
        for $\mathbf{u} \in \mathbb{F}_2^{56}$ **do**
            $L \leftarrow [\,]$
            for $\mathsf{S}^{(3)}[8, 9] \in \mathbb{F}_2^{16}$ **do**
                Compute the 2-byte matching point v, $L[v] \leftarrow \mathsf{S}^{(3)}[8, 9]$
            end
            for *values in* $U[\mathbf{u}]$ **do**
                Compute the 2-byte matching point v'
                for *values in* $L[v']$ **do**
                    if $E(Key = \mathsf{S}^{(3)}, P_1) = C_1$ *and* $E(Key = \mathsf{S}^{(3)}, P_2) = C_2$ **then**
                        return $\mathsf{S}^{(3)}$
                    end
                end
            end
        end
    end
end

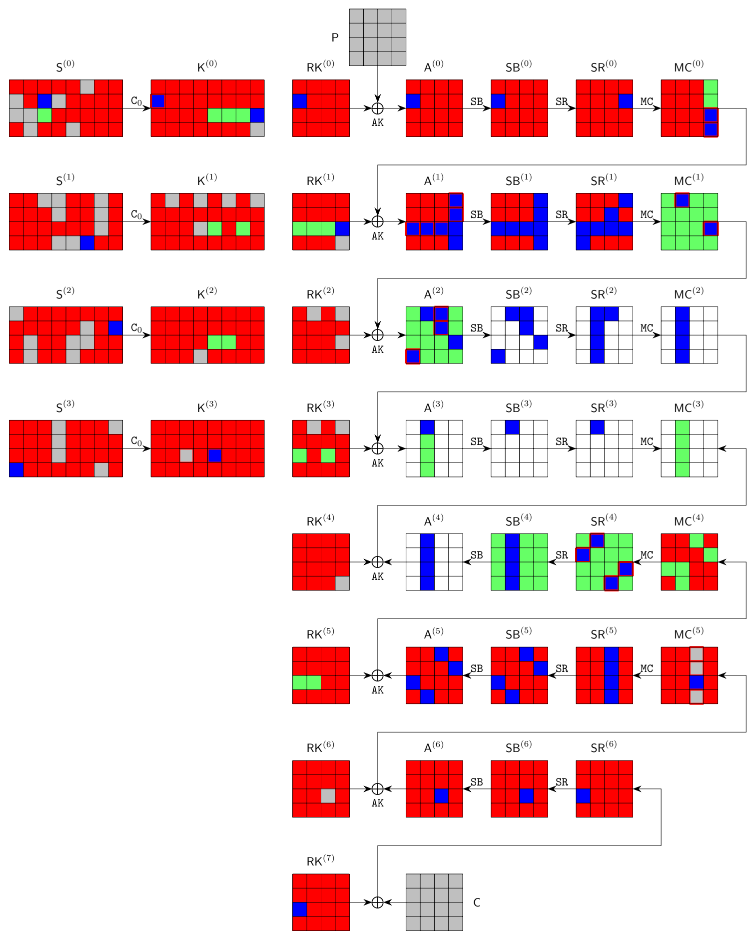

Fig. 19: The 7-round attack on AES-256

$$
\begin{cases}
\widehat{\mathsf{K}}^{(0)}[1] = k_{10} \oplus k_{16} \oplus k_{30} \oplus k_4 \boxed{\oplus S(k_{23})} \\
\widehat{\mathsf{K}}^{(3)}[10] = k_{28} \oplus S(k_{27} \oplus S(k_{26} \oplus S(k_{25}))) \oplus k_2 \oplus S(k_1) \\
\widehat{\mathsf{K}}^{(3)}[18] = k_{29} \oplus S(k_{28} \oplus S(k_{27} \oplus S(k_{26} \oplus S(k_{25})))) \oplus k_3 \oplus S(k_2 \oplus S(k_1)) \oplus k_9 \boxed{\oplus k_{23}} \\
\widehat{\mathsf{MC}}^{(0)}[14] = S(k_6) \oplus 2 \cdot S(k_{22} \oplus k_2) \oplus 3 \cdot S(k_{14} \oplus k_{20}) \boxed{\oplus \mathsf{SR}^{(0)}[13]} \\
\widehat{\mathsf{MC}}^{(0)}[15] = 3 \cdot S(k_6) \oplus S(k_{22} \oplus k_2) \oplus 2 \cdot S(k_{14} \oplus k_{20}) \boxed{\oplus \mathsf{SR}^{(0)}[13]} \\
\widehat{\mathsf{A}}^{(1)}[2] = S(k_{12} \oplus k_{18} \oplus k_{24} \oplus S(k_{31}) \oplus k_6) \oplus S(k_{10} \oplus k_{30}) \oplus 2 \cdot S(k_{22} \oplus k_{28}) \oplus 3 \cdot S(k_{14}) \\
\qquad\qquad \oplus k_9 \oplus k_{29} \oplus k_3 \boxed{\oplus S(k_8) \oplus k_{23}} \\
\widehat{\mathsf{A}}^{(1)}[6] = S(k_{18} \oplus k_6) \oplus S(k_{30} \oplus k_4) \oplus 2 \cdot S(k_{22}) \oplus 3 \cdot S(k_{14} \oplus k_{20} \oplus k_{26} \oplus k_0 \oplus S(k_7)) \\
\qquad\qquad \oplus k_3 \boxed{\oplus k_{23}} \\
\widehat{\mathsf{A}}^{(1)}[10] = S(k_{12} \oplus k_6) \oplus S(k_{30}) \oplus 2 \cdot S(k_8 \oplus S(k_{15}) \oplus k_{22} \oplus k_{28} \oplus k_2) \oplus 3 \cdot S(k_{14} \oplus k_{26}) \\
\qquad\qquad \oplus k_{29} \boxed{\oplus k_{23}} \\
\widehat{\mathsf{A}}^{(1)}[12] = 2 \cdot S(k_6) \oplus S(k_{22} \oplus k_2) \oplus S(k_{14} \oplus k_{20}) \oplus k_7 \boxed{\oplus 3 \cdot \mathsf{SR}^{(0)}[13]} \\
\widehat{\mathsf{A}}^{(1)}[13] = S(k_6) \oplus 3 \cdot S(k_{22} \oplus k_2) \oplus S(k_{14} \oplus k_{20}) \oplus k_{31} \boxed{\oplus 2 \cdot \mathsf{SR}^{(0)}[13]} \\
\widehat{\mathsf{MC}}^{(1)}[4] = 2 \cdot \mathsf{SR}^{(1)}[4] \oplus 3 \cdot \mathsf{SR}^{(1)}[5] \oplus \mathsf{SR}^{(1)}[7] \boxed{\oplus \mathsf{SR}^{(1)}[6]} \\
\widehat{\mathsf{MC}}^{(1)}[14] = \mathsf{SR}^{(1)}[13] \oplus 3 \cdot \mathsf{SR}^{(1)}[6] \boxed{\oplus \mathsf{SR}^{(1)}[12] \oplus 2 \cdot \mathsf{SR}^{(1)}[14]} \\
\widehat{\mathsf{A}}^{(2)}[3] = 3 \cdot \mathsf{SR}^{(1)}[0] \oplus \mathsf{SR}^{(1)}[1] \oplus \mathsf{RK}^{(2)}[3] \boxed{\oplus \mathsf{SR}^{(1)}[2] \oplus 2 \cdot \mathsf{SR}^{(1)}[3]} \\
\widehat{\mathsf{A}}^{(2)}[8] = 2 \cdot \mathsf{SR}^{(1)}[8] \oplus \mathsf{SR}^{(1)}[11] \oplus \mathsf{RK}^{(2)}[8] \boxed{\oplus 3 \cdot \mathsf{SR}^{(1)}[9] \oplus \mathsf{SR}^{(1)}[10]} \\
\widehat{\mathsf{A}}^{(2)}[9] = \mathsf{SR}^{(1)}[8] \oplus \mathsf{SR}^{(1)}[11] \oplus \mathsf{RK}^{(2)}[9] \boxed{\oplus 2 \cdot \mathsf{SR}^{(1)}[9] \oplus 3 \cdot \mathsf{SR}^{(1)}[10]} \\
\widehat{\mathsf{MC}}^{(5)}[8] = S^{-1}(k_{19} \oplus S(k_{18} \oplus S(k_{17})) \oplus k_{13} \oplus S(k_{12} \oplus S(k_{11} \oplus S(k_{10} \oplus S(k_9))))) \oplus k_{18} \\
\qquad\qquad \oplus S(k_{17}) \oplus S(k_{11} \oplus S(k_{10} \oplus S(k_9))) \boxed{\oplus k_{12}} \\
\widehat{\mathsf{MC}}^{(5)}[9] = S^{-1}(k_{17} \oplus k_5 \oplus S(k_4 \oplus S(k_3 \oplus S(k_2 \oplus S(k_1))))) \oplus k_4 \oplus S(k_3 \oplus S(k_2 \oplus S(k_1))) \\
\qquad\qquad \oplus k_{10} \oplus S(k_9) \\
\widehat{\mathsf{MC}}^{(5)}[11] = S^{-1}(k_{21} \oplus S(k_{20} \oplus S(k_{19} \oplus S(k_{18} \oplus S(k_{17}))))) \oplus k_{20} \oplus S(k_{19} \oplus S(k_{18} \oplus S(k_{17}))) \\
\qquad\qquad \boxed{\oplus k_{26} \oplus S(k_{25})} \\
\widehat{\mathsf{SR}}^{(4)}[1] = \mathtt{9} \cdot \mathsf{MC}^{(4)}[0] \oplus \mathtt{e} \cdot \mathsf{MC}^{(4)}[1] \oplus \mathtt{b} \cdot (\mathsf{S}^{(2)}[3] \oplus \mathsf{S}^{(2)}[9] \oplus \mathsf{S}^{(2)}[23]) \oplus \mathtt{d} \cdot \mathsf{MC}^{(4)}[3] \\
\qquad\qquad \boxed{\oplus \mathtt{b} \cdot (\mathsf{S}^{(2)}[29] \oplus \mathsf{A}^{(5)}[2])} \\
\widehat{\mathsf{SR}}^{(4)}[4] = \mathtt{e} \cdot \mathsf{MC}^{(4)}[4] \oplus \mathtt{b} \cdot \mathsf{MC}^{(4)}[5] \oplus \mathtt{d} \cdot (\mathsf{S}^{(2)}[9] \oplus \mathsf{A}^{(5)}[6]) \oplus \mathtt{9} \cdot \mathsf{RK}^{(2)}[29] \boxed{\oplus \mathtt{b} \cdot (\mathsf{S}^{(2)}[29] \oplus \mathsf{A}^{(5)}[7])} \\
\widehat{\mathsf{SR}}^{(4)}[11] = \mathtt{d} \cdot \mathsf{RK}^{(5)}[8] \oplus \mathtt{9} \cdot \mathsf{MC}^{(4)}[9] \oplus \mathtt{e} \cdot \mathsf{MC}^{(4)}[10] \oplus \mathtt{b} \cdot \mathsf{MC}^{(4)}[11] \boxed{\oplus \mathtt{d} \cdot \mathsf{A}^{(5)}[8]} \\
\widehat{\mathsf{SR}}^{(4)}[14] = \mathtt{b} \cdot \mathsf{MC}^{(4)}[12] \oplus \mathtt{d} \cdot \mathsf{RK}^{(5)}[13] \oplus \mathtt{9} \cdot \mathsf{MC}^{(4)}[14] \oplus \mathtt{e} \cdot \mathsf{MC}^{(4)}[15] \boxed{\oplus \mathtt{d} \cdot \mathsf{A}^{(5)}[13]}
\end{cases}
\tag{29}
$$

$$
\begin{gathered}
\left(\begin{array}{r|ccccccccccccccccccccccc}
 & k_0 & k_1 & k_2 & k_3 & k_4 & k_5 & k_6 & k_7 & k_9 & k_{10} & k_{11} & k_{14} & k_{16} & k_{17} & k_{18} & k_{20} & k_{21} & k_{22} & k_{27} & k_{28} & k_{29} & k_{30} & k_{31} \\
\hline
\widehat{\mathsf{K}}^{(0)}[1] & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\widehat{\mathsf{K}}^{(3)}[10] & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
\widehat{\mathsf{K}}^{(3)}[18] & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(0)}[14] & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(0)}[15] & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{A}}^{(1)}[2] & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{A}}^{(1)}[6] & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[10] & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[12] & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{A}}^{(1)}[13] & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
\widehat{\mathsf{MC}}^{(1)}[4] & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
\widehat{\mathsf{MC}}^{(1)}[14] & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\
\widehat{\mathsf{A}}^{(2)}[3] & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\
\widehat{\mathsf{A}}^{(2)}[8] & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\
\widehat{\mathsf{A}}^{(2)}[9] & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\
\widehat{\mathsf{MC}}^{(5)}[8] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(5)}[9] & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(5)}[11] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{SR}}^{(4)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(4)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(4)}[11] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(4)}[14] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{array}\right) \\
(a) \\
\left(\begin{array}{r|cccccccccccccc|ccccccccc}
 & k_5 & k_{21} & k_{17} & k_{11} & k_{16} & k_{10} & k_{31} & k_9 & k_{29} & k_1 & k_3 & k_4 & k_7 & k_2 & k_0 & k_6 & k_{14} & k_{18} & k_{20} & k_{22} & k_{27} & k_{28} & k_{30} \\
\hline
\widehat{\mathsf{SR}}^{(4)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(4)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(4)}[11] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{SR}}^{(4)}[14] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{MC}}^{(1)}[4] & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{MC}}^{(1)}[14] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\
\widehat{\mathsf{A}}^{(2)}[9] & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{MC}}^{(0)}[14] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
\hline
\widehat{\mathsf{MC}}^{(5)}[9] & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(5)}[11] & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(5)}[8] & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
\widehat{\mathsf{A}}^{(2)}[3] & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
\widehat{\mathsf{K}}^{(0)}[1] & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\widehat{\mathsf{A}}^{(1)}[2] & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\
\widehat{\mathsf{A}}^{(1)}[13] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
\widehat{\mathsf{K}}^{(3)}[18] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[10] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\
\widehat{\mathsf{K}}^{(3)}[10] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
\widehat{\mathsf{A}}^{(1)}[6] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\
\widehat{\mathsf{A}}^{(2)}[8] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\widehat{\mathsf{A}}^{(1)}[12] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
\widehat{\mathsf{MC}}^{(0)}[15] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0
\end{array}\right) \\
(b)
\end{gathered}
\tag{30}
$$

E Experiments of the Preimage Attack on Hashing Modes with 10-round AES-256

We reuse the 10-round MitM characteristic in [26] as shown in Figure 20. Obtain the 18 equations on ■ bytes in $\mathsf{S}^{(3)}$ for the cancellations of blue in $\mathsf{S}^{(1)}[13,14]$, $\mathsf{S}^{(2)}[12,13,14,15]$, $\mathsf{S}^{(4)}[1]$, $K^{(2)}[0,3,4,5,9]$, $\mathsf{SR}^{(1)}[3,6,9,12]$ and $\mathsf{SR}^{(9)}[6,12]$. The matrices before and after the new TA are shown in Eq. (31). The steps for the 10-round MitM attack are given in Algorithm 12. The total time complexity is about 2^{120}. The memory is 2^8 to store U.

Algorithm 11: Attack on 7-round AES-256 with 2 (P, C)

```
for 2^ζ values of S^(1)[8,12,13,15,19,24,25,26] ∈ F_2^64 do
    for (e_1, e_2, ···, e_14) ∈ F_2^112 do
        U ← [ ]
        for S^(1)[0,6,14,18,20,22,27,28,30] ∈ F_2^72 do
            Assign (e_1, e_2, ···, e_14) to the last 14 expressions in Eq. (30) (b)
            Deduce S^(1)[1,2,3,4,5,7,9,10,11,16,17,21,29,31]
            Compute 8-byte
              u = (\hat{SR}^(4)[1,4,11,14], \hat{MC}^(1)[4,14], \hat{A}^(2)[9], \hat{MC}^(0)[14]) ∈ F_2^64
            U[u] ← S^(1)[v_R] /* v_R represents all 23 red indexes */
        end
        for u ∈ F_2^64 do
            L ← [ ]
            for S^(1)[23] ∈ F_2^8 do
                Compute forward and backward to the 1-byte matching point v
                L[v] ← S^(1)[23]
            end
            for values in U[u] do
                Compute forward and backward to the 1-byte matching point v'
                for values in L[v'] do
                    if E(Key = S^(1), P_1) = C_1 and E(Key = S^(1), P_2) = C_2 then
                        return S^(1)
                    end
                end
            end
        end
    end
end
```

We experiment with the 5-byte partial target preimage attack by fixing the 5 target bytes $T[0,1,2,6,12] = 0$. The 4 preimages are produced with $T[0,1,2,6,12] = 0$, which are listed in Table 3.

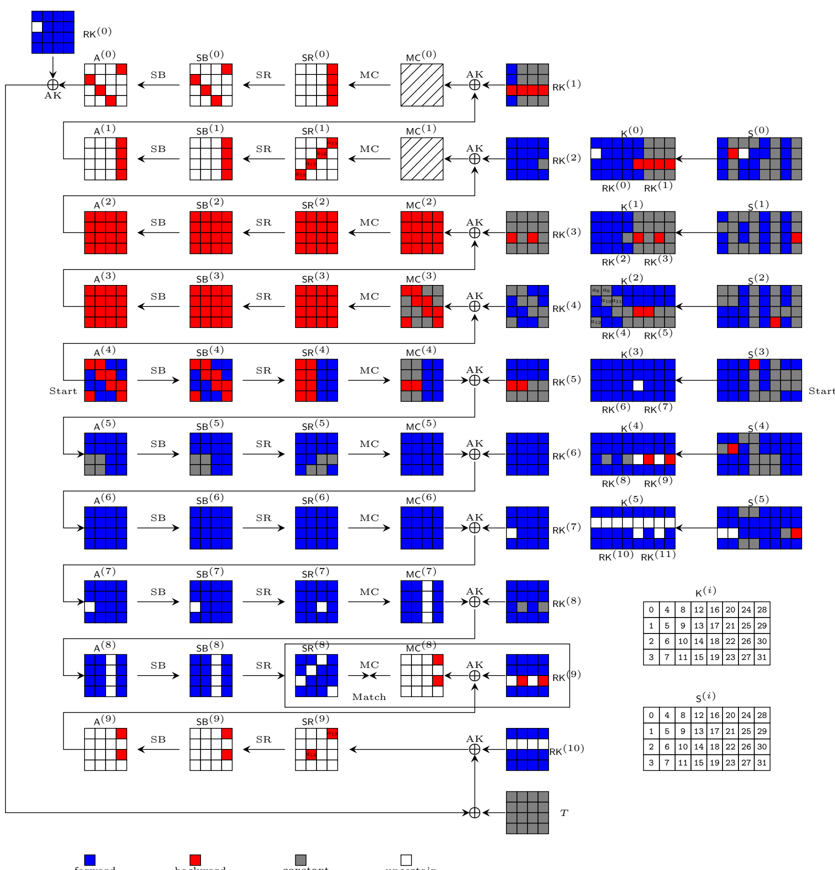

Fig. 20: The MitM attack on 10-round AES-256

	k_0	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}	k_{11}	k_{16}	k_{17}	k_{18}	k_{24}	k_{27}	k_{28}	k_{31}
a_1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
a_2	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
a_3	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
a_4	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
a_5	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a_6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
a_7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
a_8	0	0	0	1	0	0	0	0	0	0	1	0	0	1	0	1	0	0	1
a_9	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0
a_{10}	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0
a_{11}	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0
a_{12}	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0
a_{13}	0	0	0	0	0	1	0	0	0	1	1	0	0	1	0	0	0	0	0
a_{14}	0	1	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0
a_{15}	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0
a_{16}	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0
a_{17}	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	0	0	1	1
a_{18}	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1

(a)

	k_6	k_2	k_3	k_7	k_{11}	k_{31}	k_0	k_4	k_{28}	k_1	k_5	k_9	k_{17}	k_{10}	k_{27}	k_8	k_{16}	k_{24}	k_{18}
a_{18}	0	1	1	0	1	1	1	0	1	1	0	1	1	1	1	1	1	1	1
a_{16}	0	0	0	0	1	0	0	0	0	0	0	1	1	1	0	1	1	0	1
a_8	0	0	1	0	0	1	0	0	0	0	0	0	1	1	0	0	0	1	0
a_3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a_{17}	0	1	1	0	0	1	1	0	1	1	0	0	1	0	0	0	1	0	1
a_4	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a_{14}	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0
a_{12}	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1
a_6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
a_5	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
a_1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0
a_7	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0
a_2	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
a_{13}	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0
a_{10}	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
a_{15}	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1
a_9	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0
a_{11}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

(b)

(31)

F Details of Specifications on Keccak, Ascon and Xoodyak Hash functions

F.1 The Keccak Hash Function

The Keccak [7] family has a $b = 1600$-bit state and different sizes of capacity and rate. For example, Keccak[1024] has a 512-bit digest and a capacity $c = 2 \times 512$. The 1600-bit state can be viewed as a $5 \times 5 \times 64$ array of bits. Denote $A^{(r)}_{\{x,y,z\}}$ as the bit located at the x-th column, y-th row and z-th lane in the round r ($r \geq 0$), where $0 \leq x \leq 4,\ 0 \leq y \leq 4,\ 0 \leq z \leq 63$. For simplicity, all the coordinates are considered modulo 5 for x and y, and modulo 64 for z in the following paper. The inner permutation consists of 24 rounds, and each round has five operations $\iota \circ \chi \circ \pi \circ \rho \circ \theta$. The internal states of round r are denoted as $A^{(r)} \xrightarrow{\theta} \theta^{(r)} \xrightarrow{\rho} \rho^{(r)} \xrightarrow{\pi} \pi^{(r)} \xrightarrow{\chi} \chi^{(r)} \xrightarrow{\iota} A^{(r+1)}$. We list the five operations in each

Algorithm 12: Preimage Attack on 10-round AES-256-DM

Assign 0 to all ■ in $\mathsf{S}^{(3)}$, $\mathsf{S}^{(3)}[13, 14, 15, 19, 20, 21, 22, 23, 25, 26, 29, 30] \leftarrow 0$.
$(a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_9, a_{10}, a_{11}, a_{12}, a_{13}, a_{14}, a_{15}, a_{17}) \leftarrow 0$
$U \leftarrow [\ ]$
for $\mathsf{S}^{(3)}[8, 16, 18, 24] \in \mathbb{F}_2^{32}$ **do**
    Compute $\mathsf{S}^{(3)}[0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 17, 27, 28, 31]$
    Compute $a_8,\ a_{16},\ a_{18}$(denoted as $\mathbf{u} \in \mathbb{F}_2^{24}$)
    If $a_8 = 0,\ a_{16} = 0$ and $a_{18} = 0$, store the 19-byte ■ of $\mathsf{S}^{(3)}$ in $U[\mathbf{u}]$.
end
`/* In the following, we always fix` $a_8 = 0,\ a_{16} = 0$ `and` $a_{18} = 0$ `*/`
for 2^{112} *values of* ■ *in* $\mathsf{MC}^{(3)}, \mathsf{MC}^{(4)}$ *and* $\mathsf{A}^{(5)}[2, 6]$ **do**
    $L \leftarrow [\ ]$
    for $\mathsf{S}^{(3)}[12] \in \mathbb{F}_2^8$ **do**
        Compute backward to the 1-byte matching point v
        $L[v] \leftarrow \mathsf{S}^{(3)}[12]$
    end
    for *values in* $U[0]$ **do**
        Compute forward to the 1-byte matching point v'
        for *values in* $L[v']$ **do**
            Check the full preimage.
        end
    end
end

Rounds	$\mathsf{S}^{(3)}$	$\mathsf{A}^{(4)}$	Target
10	0x40, 0x09, 0x01, 0x7a, 0x2d, 0x00, 0x50, 0x53, 0x90, 0x60, 0x2d, 0x00, 0x70, 0x00, 0x00, 0x00, 0x30, 0x04, 0x50, 0x00, 0x8e, 0x00, 0x00, 0x00, 0x11, 0x82, 0x23, 0x00, 0x00, 0x00, 0x2d, 0xd8	0xb9, 0x7e, 0xf2, 0xba, 0x89, 0x36, 0x01, 0x23, 0x20, 0x6c, 0x59, 0x8e, 0x50, 0x2d, 0x70, 0x43	0x00, 0x00, 0x00, 0xe9, 0x2d, 0xcf, 0x00, 0x56, 0xde, 0xfa, 0xac, 0xe6, 0x00, 0x7c, 0x3b, 0x83
	0xc7, 0xc6, 0xb4, 0xff, 0xe7, 0x00, 0xb3, 0x6d, 0x5a, 0xbe, 0xe7, 0x00, 0x51, 0x00, 0x00, 0x00, 0xa9, 0xd3, 0xb3, 0x00, 0x4e, 0x00, 0x00, 0x00, 0x73, 0x8f, 0xb5, 0x00, 0x00, 0x00, 0xe7, 0x94	0x57, 0x2d, 0x8d, 0xc3, 0x50, 0x57, 0xb4, 0xb5, 0xe2, 0x75, 0xe3, 0x4e, 0xb3, 0xe7, 0x28, 0x9c	0x00, 0x00, 0x00, 0xb7, 0xb6, 0x05, 0x00, 0x21, 0xda, 0x1e, 0x49, 0x31, 0x00, 0xbd, 0x0d, 0xef
	0xe0, 0xe1, 0xf8, 0x6e, 0xd3, 0x00, 0x34, 0x18, 0x79, 0xb6, 0xd3, 0x00, 0x6c, 0x00, 0x00, 0x00, 0xa0, 0xe0, 0x34, 0x00, 0x8e, 0x00, 0x00, 0x00, 0x5f, 0xcf, 0xc3, 0x00, 0x00, 0x00, 0xd3, 0x66	0xcd, 0x10, 0xe2, 0x6e, 0x40, 0xd8, 0xf8, 0xc3, 0x58, 0x19, 0x1a, 0x8e, 0x34, 0xd3, 0xbb, 0x11	0x00, 0x00, 0x00, 0x5b, 0x4a, 0x26, 0x00, 0x95, 0x94, 0xf4, 0x2d, 0x08, 0x00, 0x48, 0x16, 0x2d
	0xc0, 0xba, 0xf4, 0x17, 0x12, 0x00, 0x5a, 0xbe, 0x91, 0x81, 0x12, 0x00, 0x1a, 0x00, 0x00, 0x00, 0xd5, 0x03, 0x5a, 0x00, 0xf3, 0x00, 0x00, 0x00, 0xc7, 0xc6, 0x9d, 0x00, 0x00, 0x00, 0x12, 0xc9	0xbf, 0xa4, 0x06, 0x44, 0x87, 0x32, 0xf4, 0x9d, 0x40, 0xde, 0x3c, 0xf3, 0x5a, 0x12, 0x39, 0xf7	0x00, 0x00, 0x00, 0x57, 0xe8, 0x01, 0x00, 0xb7, 0x94, 0x6d, 0xcc, 0xe3, 0x00, 0x9f, 0xfb, 0xa7

Table 3: Preimages of 10-round AES-256

round as follows:

$$
\begin{aligned}
&\theta : \theta^{(r)}_{\{x,y,z\}} = A^{(r)}_{\{x,y,z\}} \oplus D^{(r)}_{\{x,z\}}, D^{(r)}_{\{x,z\}} = C^{(r)}_{\{x-1,z\}} \oplus C^{(r)}_{\{x+1,z-1\}}, C^{(r)}_{\{x,z\}} = \sum\nolimits_{y'=0}^{4} A^{(r)}_{\{x,y',z\}},\\
&\rho : \rho^{(r)}_{\{x,y,z\}} = \theta^{(r)}_{\{x,y,z-\gamma[x,y]\}},\\
&\pi : \pi^{(r)}_{\{y,2x+3y,z\}} = \rho^{(r)}_{\{x,y,z\}},\\
&\chi : \chi^{(r)}_{\{x,y,z\}} = \pi^{(r)}_{\{x,y,z\}} \oplus (\pi^{(r)}_{\{x+1,y,z\}} \oplus 1) \cdot \pi^{(r)}_{\{x+2,y,z\}},\\
&\iota : A^{(r+1)} = \chi^{(r)} \oplus RC_r, RC_r \text{ is round-dependent constant,}
\end{aligned}
\tag{32}
$$

where the θ operation is divided into three steps. The rotation constants $\gamma[x, y]$s are given in Table 4.

This paper focuses on Keccak[1024] and Keccak[768], as well as SHA3-512 and SHA3-384. For Keccak, the message is padded with "10*1", which is a single bit 1 followed by the minimum number of 0s and followed by a single bit 1. For SHA3, the message is padded with "0110*1".

	$x = 0$	$x = 1$	$x = 2$	$x = 3$	$x = 4$
$y = 0$	0	1	62	28	27
$y = 1$	36	44	6	55	20
$y = 2$	3	10	43	25	39
$y = 3$	41	45	15	21	8
$y = 4$	18	2	61	56	14

Table 4: The offset $\gamma[x, y]$ in the ρ operation forKeccak

F.2 Ascon-Hash and Ascon-XOF

The Ascon family [24] includes the hash functions Ascon-Hash and Ascon-Hasha as well as the extendable output functions Ascon-XOF and Ascon-XOFa with sponge-based modes of operations.

Ascon Permutation. The inner permutation applies 12 round functions to a 320-bit state. The state A is split into five 64-bit words, and denote $A^{(r)}_{\{x,y\}}$ to be the x-th (column) bit of the y-th (row) 64-bit word, where $0 \leq y \leq 4$, $0 \leq x \leq 63$. The round function consists of three operations p_C, p_S, and p_L. Denote the internal states of round r as $A^{(r)} \xrightarrow{p_S \circ p_C} S^{(r)} \xrightarrow{p_L} A^{(r+1)}$.

- **Addition of Constants** p_C**:** $A^{(r)}_{\{*,2\}} = A^{(r)}_{\{*,2\}} \oplus RC_r$.
- **Substitution Layer** p_S**:** For each x, this step updates the columns $A^{(r)}_{\{x,*\}}$ using the 5-bit Sbox. Assume the S-box maps $(a_0, a_1, a_2, a_3, a_4) \in \mathbb{F}_2^5$ to

$(b_0, b_1, b_2, b_3, b_4) \in \mathbb{F}_2^5$, where a_0 is the most significant bit. The algebraic normal form (ANF) of the Sbox is as follows:

$$\begin{cases} b_0 = a_4a_1 + a_3 + a_2a_1 + a_2 + a_1a_0 + a_1 + a_0 \\ b_1 = a_4 + a_3a_2 + a_3a_1 + a_3 + a_2a_1 + a_2 + a_1 + a_0 \\ b_2 = a_4a_3 + a_4 + a_2 + a_1 + 1 \\ b_3 = a_4a_0 + a_4 + a_3a_0 + a_3 + a_2 + a_1 + a_0 \\ b_4 = a_4a_1 + a_4 + a_3 + a_1a_0 + a_1 \end{cases} \tag{33}$$

– **Linear Diffusion Layer p_L:**

$$\begin{aligned} A_{\{*,0\}}^{(r+1)} &\leftarrow S_{\{*,0\}}^{(r)} \oplus (S_{\{*,0\}}^{(r)} \ggg 19) \oplus (S_{\{*,0\}}^{(r)} \ggg 28), \\ A_{\{*,1\}}^{(r+1)} &\leftarrow S_{\{*,1\}}^{(r)} \oplus (S_{\{*,1\}}^{(r)} \ggg 61) \oplus (S_{\{*,1\}}^{(r)} \ggg 39), \\ A_{\{*,2\}}^{(r+1)} &\leftarrow S_{\{*,2\}}^{(r)} \oplus (S_{\{*,2\}}^{(r)} \ggg 1) \oplus (S_{\{*,2\}}^{(r)} \ggg 6), \\ A_{\{*,3\}}^{(r+1)} &\leftarrow S_{\{*,3\}}^{(r)} \oplus (S_{\{*,3\}}^{(r)} \ggg 10) \oplus (S_{\{*,3\}}^{(r)} \ggg 17), \\ A_{\{*,4\}}^{(r+1)} &\leftarrow S_{\{*,4\}}^{(r)} \oplus (S_{\{*,4\}}^{(r)} \ggg 7) \oplus (S_{\{*,4\}}^{(r)} \ggg 41). \end{aligned}$$

Ascon-Hash and Ascon-XOF. The state A is composed of the outer part with 64 bits $A_{\{*,0\}}$ and the inner part 256 bits $A_{\{*,i\}}$ $(i = 1, 2, 3, 4)$. For Ascon-Hash, the output size is 256 bits, and the security claim is 2^{128}. For Ascon-XOF, the output can have arbitrary length and the security claim against preimage attack is $\min(2^{128}, 2^l)$, where l is the output length.

F.3 Xoodyak and Xoodoo Permutation

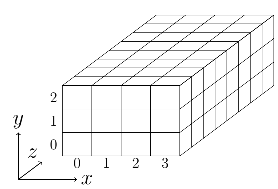

Fig. 21: Toy version of the Xoodoo state. The order in y is opposite to Keccak

Internally, Xoodyak makes use of the Xoodoo permutation [15], whose state (shown in Figure 21) bit denoted by $A_{\{x,y,z\}}^{(r)}$ is located at the x-th column, y-th row and z-th lane in the round r, where $0 \le x \le 3,\ 0 \le y \le 2,\ 0 \le z \le 31$. For Xoodoo, all the coordinates are considered modulo 4 for x, modulo 3 for y, and modulo 32 for z. The permutation consists of the iteration of a round function $R = \rho_{\text{east}} \circ \chi \circ \iota \circ \rho_{\text{west}} \circ \theta$. The number of rounds is a parameter, which is 12 in Xoodyak. Denote the internal states of the round r as

$$A^{(r)} \xrightarrow{\theta} \theta^{(r)} \xrightarrow{\rho_{\text{west}}} \rho^{(r)} \xrightarrow{\iota} \iota^{(r)} \xrightarrow{\chi} \chi^{(r)} \xrightarrow{\rho_{\text{east}}} A^{(r+1)}.$$

$$\begin{aligned}
\theta: \ & \theta^{(r)}_{\{x,y,z\}} = A^{(r)}_{\{x,y,z\}} \oplus \sum\nolimits_{y'=0}^{2} (A^{(r)}_{\{x-1,y',z-5\}} \oplus A^{(r)}_{\{x-1,y',z-14\}}),\\
\rho_{\text{west}}: \ & \rho^{(r)}_{\{x,0,z\}} = \theta^{(r)}_{\{x,0,z\}},\ \rho^{(r)}_{\{x,1,z\}} = \theta^{(r)}_{\{x-1,1,z\}},\ \rho^{(r)}_{\{x,2,z\}} = \theta^{(r)}_{\{x,2,z-11\}},\\
\iota: \ & \iota^{(r)}_{\{0,0,z\}} = \rho^{(r)}_{\{0,0,z\}} \oplus RC_r, \text{ where } RC_r \text{ is round-dependent constant,} \qquad (34)\\
\chi: \ & \chi^{(r)}_{\{x,y,z\}} = \iota^{(r)}_{\{x,y,z\}} \oplus (\iota^{(r)}_{\{x,y+1,z\}} \oplus 1) \cdot \iota^{(r)}_{\{x,y+2,z\}},\\
\rho_{\text{east}}: \ & A^{(r+1)}_{\{x,0,z\}} = \chi^{(r)}_{\{x,0,z\}},\ A^{(r+1)}_{\{x,1,z\}} = \chi^{(r)}_{\{x,1,z-1\}},\ A^{(r+1)}_{\{x,2,z\}} = \chi^{(r)}_{\{x-2,2,z-8\}}.
\end{aligned}$$

Xoodyak can serve as a XOF, i.e. Xoodyak-XOF, which offers arbitrary output length l. The preimage resistance is $\min(2^{128}, 2^l)$. We target on Xoodyak-XOF with output of 128-bit hash value and 128-bit absorbed message block.

G MITM Attacks on 3-round Xoodyak-XOF

The specification of Xoodyak [15] (one of the finalists of NIST LWC) is given in F.3. The Xoodyak-XOF offers an arbitrary output length l and the preimage resistance is $\min(2^{128}, 2^l)$. We target `Xoodyak-XOF` with a 128-bit digest against the preimage attack and collision attack.

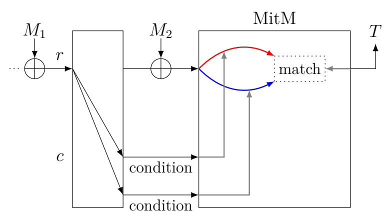

Fig. 22: MitM Preimage Attack [45]

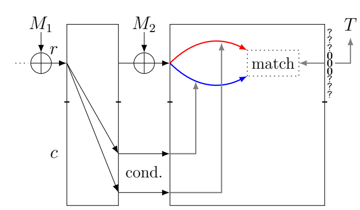

Fig. 23: MitM Collision Attack [27]

G.1 MitM Preimage Attack on 3-round Xoodyak-XOF

Following Qin *et al.*'s framework [45] of MitM preimage attack on sponge function in Figure 22, we launch a new 3-round MitM preimage attack on Xoodyak-XOF with a new 3-round MitM path given in Figure 24.

The starting state $\mathsf{A}^{(0)}$ contains $\lambda^+ = 7$ ■ bits and $\lambda^- = 67$ ■ bits. In the computation from $\mathsf{A}^{(0)}$ to $\iota^{(2)}$, the consumed degrees of freedom (DoFs) of ■ and ■ are $l^+ = 0$ and $l^- = 60$ bits, respectively. Therefore, $\text{DoF}^+ = 7$, $\text{DoF}^- = 7$, and there is $\text{DoM} = 7$ matching bits as in Eq. (35) with the deterministic relations of $\iota^{(2)}$:

$$\begin{aligned}
&\chi^{(2)}_{\{1,2,8\}} = \iota^{(2)}_{\{1,2,8\}} \oplus (\iota^{(2)}_{\{1,0,8\}} \oplus 1) \cdot \iota^{(2)}_{\{1,1,8\}},\ \chi^{(2)}_{\{1,2,10\}} = \iota^{(2)}_{\{1,2,10\}} \oplus (\iota^{(2)}_{\{1,0,10\}} \oplus 1) \cdot \iota^{(2)}_{\{1,1,10\}},\\
&\chi^{(2)}_{\{2,2,10\}} = \iota^{(2)}_{\{2,2,10\}} \oplus (\iota^{(2)}_{\{2,0,10\}} \oplus 1) \cdot \iota^{(2)}_{\{2,1,10\}},\ \chi^{(2)}_{\{1,2,17\}} = \iota^{(2)}_{\{1,2,17\}} \oplus (\iota^{(2)}_{\{1,0,17\}} \oplus 1) \cdot \iota^{(2)}_{\{1,1,17\}},\\
&\chi^{(2)}_{\{1,2,26\}} = \iota^{(2)}_{\{1,2,26\}} \oplus (\iota^{(2)}_{\{1,0,26\}} \oplus 1) \cdot \iota^{(2)}_{\{1,1,26\}},\ \chi^{(2)}_{\{1,2,31\}} = \iota^{(2)}_{\{1,2,31\}} \oplus (\iota^{(2)}_{\{1,0,31\}} \oplus 1) \cdot \iota^{(2)}_{\{1,1,31\}},\\
&\chi^{(2)}_{\{2,2,31\}} = \iota^{(2)}_{\{2,2,31\}} \oplus (\iota^{(2)}_{\{2,0,31\}} \oplus 1) \cdot \iota^{(2)}_{\{2,1,31\}}.
\end{aligned} \qquad (35)$$

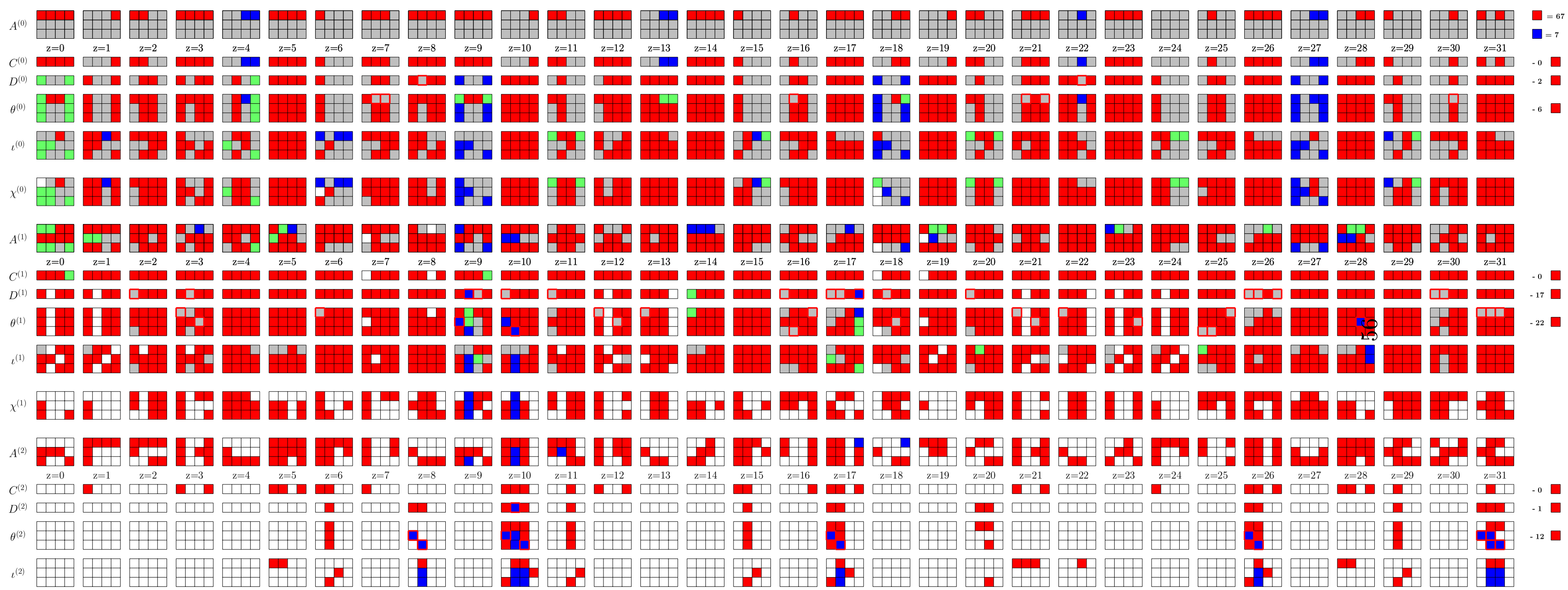

Fig. 24: The improved MitM preimage attack on 3-round Xoodyak-XOF

As shown in Figure 22, we use two message blocks (M_1, M_2) to perform the preimage attack, where M_2 has two padding bits. In the 2nd block, 70 conditions on $\iota^{(0)}$ listed in Table 5 should be satisfied before the MitM process, which form a system of linear equations involving 256 bits of the inner part of the 2nd block and $128 - 7 - 67 - 2 = 52$ gray bits of M_2. We calculate the rank of the coefficient matrix of the linear system related to the 52 bits of M_2, and the result is 32. In other words, through certain linear transformations, there are $70 - 32 = 38$ equations determined solely by the bits of the inner part. Consequently, randomly given an M_1 (thus 256-bit inner part of the 2nd block is determined), the probability of finding a solution of this system is expected to be 2^{-38}. Then, for a proper 256-bit inner part, there are $2^{52-32} = 2^{20}$ solutions of M_2 that make all the conditions hold.

$$\iota^{(0)}_{\{2,0,0\}} = 0; \iota^{(0)}_{\{2,0,1\}} = 0; \iota^{(0)}_{\{2,0,6\}} = 0; \iota^{(0)}_{\{2,0,15\}} = 0; \iota^{(0)}_{\{2,0,24\}} = 0; \iota^{(0)}_{\{3,1,0\}} = 0; \iota^{(0)}_{\{3,1,4\}} = 0;$$
$$\iota^{(0)}_{\{3,1,9\}} = 0; \iota^{(0)}_{\{3,1,18\}} = 0; \iota^{(0)}_{\{3,1,22\}} = 0; \iota^{(0)}_{\{3,1,27\}} = 0; \iota^{(0)}_{\{0,0,2\}} = 0; \iota^{(0)}_{\{0,0,11\}} = 0; \iota^{(0)}_{\{0,0,16\}} = 0;$$
$$\iota^{(0)}_{\{0,0,20\}} = 0; \iota^{(0)}_{\{0,0,25\}} = 0; \iota^{(0)}_{\{0,0,29\}} = 0; \iota^{(0)}_{\{1,2,3\}} = 0; \iota^{(0)}_{\{1,2,6\}} = 0; \iota^{(0)}_{\{1,2,9\}} = 0; \iota^{(0)}_{\{1,2,18\}} = 0;$$
$$\iota^{(0)}_{\{1,2,27\}} = 0; \iota^{(0)}_{\{2,1,3\}} = 0; \iota^{(0)}_{\{2,1,8\}} = 0; \iota^{(0)}_{\{0,2,4\}} = 0; \iota^{(0)}_{\{3,0,6\}} = 0; \iota^{(0)}_{\{3,0,11\}} = 0; \iota^{(0)}_{\{3,0,15\}} = 0;$$
$$\iota^{(0)}_{\{3,0,20\}} = 0; \iota^{(0)}_{\{3,0,24\}} = 0; \iota^{(0)}_{\{3,0,29\}} = 0; \iota^{(0)}_{\{1,1,12\}} = 0; \iota^{(0)}_{\{1,1,29\}} = 0; \iota^{(0)}_{\{0,1,15\}} = 0; \iota^{(0)}_{\{1,0,30\}} = 0;$$
$$\iota^{(0)}_{\{1,0,0\}} = 1; \iota^{(0)}_{\{1,0,3\}} = 1; \iota^{(0)}_{\{1,0,6\}} = 1; \iota^{(0)}_{\{1,0,9\}} = 1; \iota^{(0)}_{\{1,0,18\}} = 1; \iota^{(0)}_{\{1,0,27\}} = 1; \iota^{(0)}_{\{2,1,0\}} = 1;$$
$$\iota^{(0)}_{\{2,1,1\}} = 1; \iota^{(0)}_{\{2,1,6\}} = 1; \iota^{(0)}_{\{2,1,15\}} = 1; \iota^{(0)}_{\{2,1,24\}} = 1; \iota^{(0)}_{\{2,1,27\}} = 1; \iota^{(0)}_{\{3,2,0\}} = 1; \iota^{(0)}_{\{3,2,4\}} = 1;$$
$$\iota^{(0)}_{\{3,2,9\}} = 1; \iota^{(0)}_{\{3,2,18\}} = 1; \iota^{(0)}_{\{3,2,27\}} = 1; \iota^{(0)}_{\{0,1,2\}} = 1; \iota^{(0)}_{\{0,1,7\}} = 1; \iota^{(0)}_{\{0,1,11\}} = 1; \iota^{(0)}_{\{0,1,16\}} = 1;$$
$$\iota^{(0)}_{\{0,1,20\}} = 1; \iota^{(0)}_{\{0,1,29\}} = 1; \iota^{(0)}_{\{2,2,3\}} = 1; \iota^{(0)}_{\{2,2,8\}} = 1; \iota^{(0)}_{\{0,0,4\}} = 1; \iota^{(0)}_{\{3,1,6\}} = 1; \iota^{(0)}_{\{3,1,11\}} = 1;$$
$$\iota^{(0)}_{\{3,1,15\}} = 1; \iota^{(0)}_{\{3,1,20\}} = 1; \iota^{(0)}_{\{3,1,24\}} = 1; \iota^{(0)}_{\{3,1,29\}} = 1; \iota^{(0)}_{\{1,2,12\}} = 1; \iota^{(0)}_{\{2,0,22\}} = 1; \iota^{(0)}_{\{1,1,30\}} = 1;$$

Table 5: Bit Conditions in 3-round Attack on Xoodyak-XOF

Using the new TA, we get 37 free variables and 30 dependent variables. The TA matrices are shown in MATRIX/Xoodyak_3r.txt at https://anonymous.4open.science/r/Triangulation-MitM-7373. The 3-round MitM attack is given in Algorithm 13. In Line 17 to 28, a space of $2^{7+7} = 2^{14}$ is searched. Assuming 2^{ζ_1} M_1 are needed, to search a 128-bit preimage, we have to search a space of $2^{\zeta_1-38+\zeta_2+30+30+7+7} = 2^{128}$, we set $\zeta_1 = 72$ and $\zeta_2 = 20$. The complexity is about $2^{128-\min\{\mathrm{DoF}^+,\mathrm{DoF}^-,\mathrm{DoM}\}} = 2^{121}$ time and 2^{37} memory.

G.2 MitM Collision Attack on 3-round Xoodyak-XOF

The characteristic in Figure 24 can also be used to conduct a collision attack for Xoodyak-XOF following Figure 23. In the attack, set the 7 expressions of Eq. (35) as 0. We need to find $2^{(128-7)/2} = 2^{60.5}$ preimages satisfying the 7-bit 0 target, then we expect to get a collision on average for the other 121-bit target.

The detailed attack is given in Algorithm 14. In Line 20 to 33, we get $2^{7+7-7} = 2^7$ preimages, so that we need to repeat $2^{(128-7)/2-7} = 2^{53.5}$ times to build $2^{60.5}$ preimages. Assume 2^{ζ_1} possible values of M_1 are required and 2^{ζ_2} out of 2^{20} solutions of M_2 are required. We set $\zeta_1 = 38$ and $\zeta_2 = 0$. The time and memory complexities are both $2^{60.5}$.

Algorithm 13: Preimage Attack on 3-round Xoodyak-XOF

```
for 2^{ζ_1} values of M_1    /* ζ_1 = 72 */
do
    Compute the inner part of the 2nd block
    Solve the system of 70 linear equations
    if the equations have solutions   /* with probability of 2^{-38} */
    then
        for 2^{ζ_2} solutions of M_2    /* ζ_2 = 20 ≤ 20 */
        do
            U ← [ ]
            for (e_1, e_2, ··· , e_30) ∈ F_2^30 do
                for 37 free variables ■ in A^(0) ∈ 2^37 do
                    Assign (e_1, e_2, ··· , e_30) to 30 expressions
                    Deduce the 30 dependent variables
                    Compute forward to other 30-bit ■/■ u ∈ F_2^30
                    U[u] ← A^(0)[v_R]/* v_R represents 67 red indexes */
                end
                for u ∈ F_2^30 do
                    L ← [ ]
                    for A^(0)[v_B] ∈ 2^7/* v_B represents 7 blue indexes */
                    do
                        Compute forward to the 7-bit matching point v
                        L[v] ← A^(0)[v_B]
                    end
                    for values in U[u] do
                        Compute forward the 7-bit matching point v'
                        for values in L[v'] do
                            if it leads to the given hash value then
                                Output the preimage
                            end
                        end
                    end
                end
            end
        end
    end
end
```

Algorithm 14: The MitM Collision Attack on 3-round Xoodyak-Xof with 128-bit Tag

```
Fix the 7-bit matching point to 0
for $2^{\zeta_1}$ values of $M_1$   /* $\zeta = 38$ */
do
    Compute the inner part of the 2nd block
    Solve the system of 70 linear equations
    if the equations have solutions   /* with probability of $2^{-38}$ */
    then
        for $2^{\zeta_2}$ solutions of $M_2$   /* $\zeta_2 = 0 \leq 20$ */
        do
            $U \leftarrow [\ ]$
            for $2^{23.5}$ values of $(e_1, e_2, \cdots, e_{30}) \in \mathbb{F}_2^{30}$ do
                for 37 free variables in $\mathsf{A}^{(0)} \in 2^{37}$ do
                    Assign $(e_1, e_2, \cdots, e_{30})$ to 30 expressions
                    Deduce the 30 dependent variables
                    Compute forward to other 30-bit / $\mathbf{u} \in \mathbb{F}_2^{30}$
                    $U[\mathbf{u}] \leftarrow A^{(0)}[v_{\mathcal{R}}]$ /* $v_{\mathcal{R}}$ represents all 67 red indexes */
                end
                for $\mathbf{u} \in \mathbb{F}_2^{30}$ do
                    $L \leftarrow [\ ]$
                    for $\mathsf{A}^{(0)}[v_{\mathcal{B}}] \in 2^7$ /* $v_{\mathcal{B}}$ represents all 7 blue indexes */
                    do
                        Compute forward to the 7-bit matching point $v$
                        $L[v] \leftarrow \mathsf{A}^{(0)}[v_{\mathcal{B}}]$
                    end
                    for values in $U[\mathbf{u}]$ do
                        Compute forward the 7-bit matching point $v'$
                        for values in $L[v']$ do
                            Compute the 128-bit target $h$ and store the $(M_1, M_2, h)$ in $L_1$ indexed by $h$
                            if the size of $L_1$ is $2^{(128-7)/2} = 2^{60.5}$ then
                                Check $L_1$ and return $(M_1, M_2)$ and $(M_1', M_2')$ with the same $h$
                            end
                        end
                    end
                end
            end
        end
    end
end
```

H MITM Preimage Attacks on Ascon-XOF

The description of Ascon-XOF are given in Supplementary Material F.2. The 320-bit state A of Ascon is divided into five 64-bit words, and denote $\mathsf{A}^{(r)}_{\{x,y\}}$ to be the x-th (column) bit of the y-th (row) 64-bit word, where $0 \le y \le 4$, $0 \le x \le 63$.

We explore the symmetry in the x-axis to speed up the search by cutting the full 64-bit word into 32-bit word. Therefore, the linear operation works modular 32 instead of 64. For example, the linear operation in the second row changes from the original $A^{(r+1)}_{\{*,1\}} \leftarrow S^{(r)}_{\{*,1\}} \oplus (S^{(r)}_{\{*,1\}} \ggg 61) \oplus (S^{(r)}_{\{*,1\}} \ggg 39)$ into the current $A^{(r+1)}_{\{*,1\}} \leftarrow S^{(r)}_{\{*,1\}} \oplus (S^{(r)}_{\{*,1\}} \ggg 29) \oplus (S^{(r)}_{\{*,1\}} \ggg 7)$.

H.1 3-round Preimage Attack on Ascon-XOF

The 3-round MitM characteristic is shown in Figure 25. The starting state $\mathsf{A}^{(0)}$ in the full MitM path contains $\lambda^+ = 14$ 🟦 bits and $\lambda^- = 24$ 🟥 bits. There are 48 conditions on ◼ bits of $\mathsf{A}^{(0)}$. In the computation from $\mathsf{A}^{(0)}$ to $\mathsf{A}^{(2)}$, the consumed DoFs of 🟦 and 🟥 are $l^+ = 0$ and $l^- = 10$ bits, respectively. Therefore, $\mathrm{DoF}^+ = 14$, $\mathrm{DoF}^- = 14$, and there are $\mathrm{DoM} = 14$ matching bits.

Using new TA to system of 10 constraints on red 🟥 bits, we get 14 free variables and 10 dependent variables, the matrices before and after TA are shown in Eq. (37)[7]. The 3-round attack is given in Algorithm 15. The total time complexity is about $2^{128-\min\{\mathrm{DoF}^+,\mathrm{DoF}^-,\mathrm{DoM}\}} = 2^{114}$ time and 2^{14} memory.

$$\begin{cases}
e_1 = A^{(2)}_{\{26,1\}} & = A^{(0)}_{\{29,0\}}\left(A^{(0)}_{\{29,0\}} \oplus A^{(0)}_{\{32,0\}} \oplus A^{(0)}_{\{54,0\}}\right) \oplus A^{(0)}_{\{7,0\}} \oplus A^{(0)}_{\{22,0\}} \oplus A^{(0)}_{\{32,0\}} \\
e_2 = A^{(2)}_{\{58,1\}} & = A^{(0)}_{\{61,0\}}\left(A^{(0)}_{\{0,0\}} \oplus A^{(0)}_{\{22,0\}} \oplus A^{(0)}_{\{61,0\}}\right) \oplus A^{(0)}_{\{0,0\}} \oplus A^{(0)}_{\{39,0\}} \oplus A_{\{54,0\}} \\
e_3 = A^{(2)}_{\{2,1\}} & = A^{(0)}_{\{25,0\}} \oplus A^{(0)}_{\{27,0\}} \oplus A^{(0)}_{\{38,0\}} \oplus A^{(0)}_{\{47,0\}} \oplus A^{(0)}_{\{50,0\}} \oplus 1 \\
e_4 = A^{(2)}_{\{34,1\}} & = A^{(0)}_{\{6,0\}} \oplus A^{(0)}_{\{15,0\}} \oplus A^{(0)}_{\{18,0\}} \oplus A^{(0)}_{\{57,0\}} \oplus A^{(0)}_{\{59,0\}} \oplus 1 \\
e_5 = A^{(2)}_{\{12,1\}} & = A^{(0)}_{\{15,0\}} \oplus A^{(0)}_{\{18,0\}} \oplus A^{(0)}_{\{37,0\}} \oplus A^{(0)}_{\{60,0\}} \\
e_6 = A^{(2)}_{\{44,1\}} & = A^{(0)}_{\{5,0\}} \oplus A^{(0)}_{\{28,0\}} \oplus A^{(0)}_{\{47,0\}} \oplus A^{(0)}_{\{50,0\}} \\
e_7 = A^{(2)}_{\{6,1\}} & = A^{(0)}_{\{6,0\}} A^{(0)}_{\{60,0\}} \oplus A^{(0)}_{\{6,0\}} \oplus A^{(0)}_{\{32,0\}} \oplus A^{(0)}_{\{54,0\}} \\
e_8 = A^{(2)}_{\{38,1\}} & = A^{(0)}_{\{28,0\}} A^{(0)}_{\{38,0\}} \oplus A^{(0)}_{\{0,0\}} \oplus A^{(0)}_{\{22,0\}} \oplus A^{(0)}_{\{38,0\}} \\
e_9 = A^{(2)}_{\{16,1\}} & = A^{(0)}_{\{0,0\}} \oplus A^{(0)}_{\{22,0\}} \oplus A^{(0)}_{\{61,0\}} \oplus 1 \\
e_{10} = A^{(2)}_{\{48,1\}} & = A^{(0)}_{\{29,0\}} \oplus A^{(0)}_{\{32,0\}} \oplus A^{(0)}_{\{54,0\}} \oplus 1
\end{cases} \tag{36}$$

[7] Note that we explore the symmetry in the x-axis of Ascon by cutting the full 64-bit word into 32-bit word. Therefore, there are two systems of constraints as Eq. (37) totally.

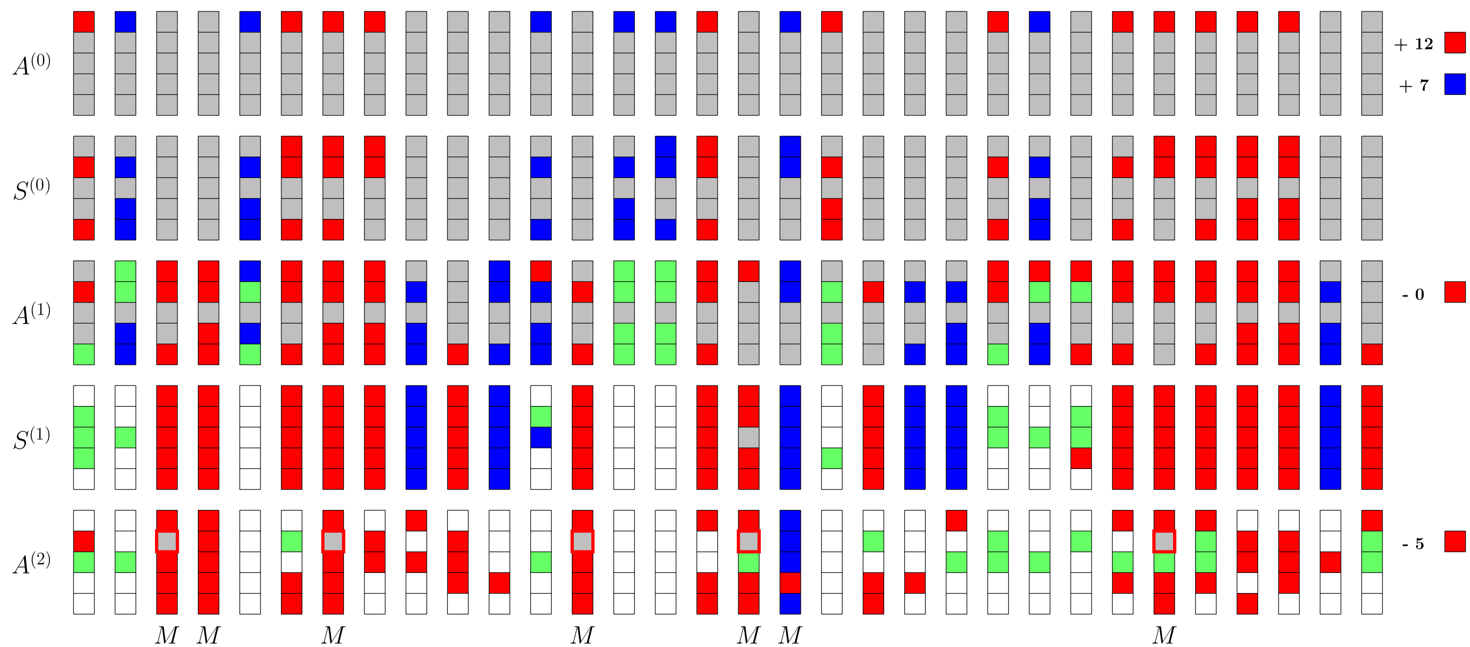

Fig. 25: The 3-round Preimage attack on Ascon-XOF. Since we explore the symmetry in the x-axis to speed up the search by cutting the full 64-bit word into 32-bit word, only 32-bit state version of Ascon-XOF is displayed here.

$$
\left(\begin{array}{c|cccccccccccc}
 & v_0 & v_5 & v_6 & v_7 & v_{15} & v_{18} & v_{22} & v_{25} & v_{26} & v_{27} & v_{28} & v_{29} \\
\hline
A^{(2)}_{\{2,1\}} & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\
A^{(2)}_{\{6,1\}} & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
A^{(2)}_{\{12,1\}} & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
A^{(2)}_{\{16,1\}} & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
A^{(2)}_{\{26,1\}} & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1
\end{array}\right)_{(a)}
\left(\begin{array}{c|ccccc|ccccccc}
 & v_7 & v_{25} & v_5 & v_6 & v_{22} & v_0 & v_{15} & v_{18} & v_{26} & v_{27} & v_{28} & v_{29} \\
\hline
A^{(2)}_{\{26,1\}} & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\
A^{(2)}_{\{2,1\}} & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
A^{(2)}_{\{12,1\}} & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
A^{(2)}_{\{6,1\}} & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\
A^{(2)}_{\{16,1\}} & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1
\end{array}\right)_{(b)} \quad (37)
$$

Algorithm 15: Preimage Attack on 3-round Ascon-XOF

Compute $S^{(3)}_{\{*,0\}} = p_L^{-1}(T)$ `/*` T `is the hash value */`
for 2^{ζ} *values of* (M_1, M_2) `/*` $\zeta = 114$ `*/`
do
    Compute the inner part of the 3rd block
    if *the 48 conditions are satisfied* `/* with probability of` 2^{-48} `*/`
    then
        for 2^{24} *values of* ■ *bits in* $\mathsf{A}^{(0)}$ **do**
            for $(e_1, e_2, \cdots, e_{10}) \in \mathbb{F}_2^{10}$ **do**
                $L \leftarrow [\,]$
                for $\mathsf{A}^{(0)}[v_{\mathcal{B}}] \in 2^{14}$ `/*` $v_{\mathcal{B}}$ `represents all 14 blue indexes */`
                do
                    Compute forward to the 14-bit matching point v
                    $L[v] \leftarrow \mathsf{A}^{(0)}[v_{\mathcal{B}}]$
                end
                for *14 free variables* ■ *in* $\mathsf{A}^{(0)} \in 2^{14}$ **do**
                    Assign $(e_1, e_2, \cdots, e_{10})$ to the last 10 expressions in Eq. (37) (b)
                    `/* Due to symmetry, the complete matrix in Eq.` (37) `(b) contains 10 rows and 24 columns. */`
                    Deduce 10 dependent variables
                    Compute forward to the 14-bit matching point v'
                    for *values in* $L[v']$ **do**
                        Check if T is satisfied
                    end
                end
            end
        end
    end
end

Partial Experiment of the Preimage Attack on 3-round Ascon-XOF. To verify the correctness of the bit-wise triangulating MitM attack, we give an experiment of a 32-bit partial target preimage attack. Fix the 14-bit target $S^{(2)}_{\{i,0\}}$ ($i \in [2, 3, 6, 12, 16, 17, 26, 34, 35, 38, 44, 48, 49, 58]$) and another 18-bit target $S^{(2)}_{\{j,0\}}$ ($j \in [0, 1, 4, 5, 7, 8, 9, 10, 11, 13, 14, 15, 18, 19, 20, 21, 22, 23]$) as zero. Totally, the 32-bit target is fixed as zero and the goal is to find the preimages of the 32-bit 0 target. The procedures are as follows:

1. Set the 256-bit inner part in $A^{(0)}$ as fixed values, which satisfy the 48-bit condition, *i.e.*, $A^{(0)}_{\{*,1\}} = \texttt{0xc8142340c8142340}$, $A^{(0)}_{\{*,3\}} = \texttt{0x8713427087134270}$, and $A^{(0)}_{\{*,2\}} = A^{(0)}_{\{*,4\}} = \texttt{0x0}$. In order to simplify the expressions for TA, we set 7 ■ bits $A^{(0)}_{\{i,0\}} = 1$ for $i \in [12, 16, 19, 30, 44, 48, 51]$, and the remaining 17 ■ bits to be 0. The simplified equations are given in Eq. (36).
2. In our experiment, we traverse the values of the first 8-bit expressions $(e_1, e_2, \cdots, e_8)$ in Eq. (36) and fix $e_9 = e_{10} = 0$ in Alg. 15, *i.e.*, $2^{8+14+14} = 2^{36}$ states are tested with MitM approach and $2^{36-32} = 2^4$ preimages are expected to find.

We finally found $2^{4.3}$ preimages satisfying the 32-bit all-zero target, which is close to the theoretical expectation. The results are given in Table 6. The experiment is run on a computer with an i9-13900KF CPU and 32 GB of memory in seconds. The source codes are available via https://anonymous.4open.science/r/Triangulation-MitM-7373.

Round	First row of preimage $\left(A^{(0)}_{\{*,0\}}\right)$	32-bit target $S^{(2)}_{\{i,0\}}$ ($i \in [0-23, 26, 34, 35, 38, 44, 48, 49, 58]$)
$r = 3$	8d0fd306451f902e	00000000
	c61b91524318b242	00000000
	8018936e800ab33e	00000000
	4a1dd34ec81a926a	00000000
	4119b25e461cb142	00000000
	cf0bb302471ed376	00000000
	cb0cb346871b931e	00000000
	c10db24a4318f256	00000000
	c80ab1028e1d9002	00000000
	cf1f924ac11f912e	00000000

Table 6: 32-bit Partial Target Preimage Examples of 3-round Ascon-Xof

H.2 4-round Preimage Attack on Ascon-XOF

The 4-round MitM characteristic is shown in Figure 26. The starting state $\mathsf{A}^{(0)}$ in the full MitM path contains $\lambda^+ = 4$ ■ bits and $\lambda^- = 34$ ■ bits. In the computation from $\mathsf{A}^{(0)}$ to $\mathsf{A}^{(3)}$, the consumed degrees of freedom (DoFs) of ■ and ■ are $l^+ = 0$ and 26 bits, respectively. Additionally, there are 4 consumed DoFs of ■ to make

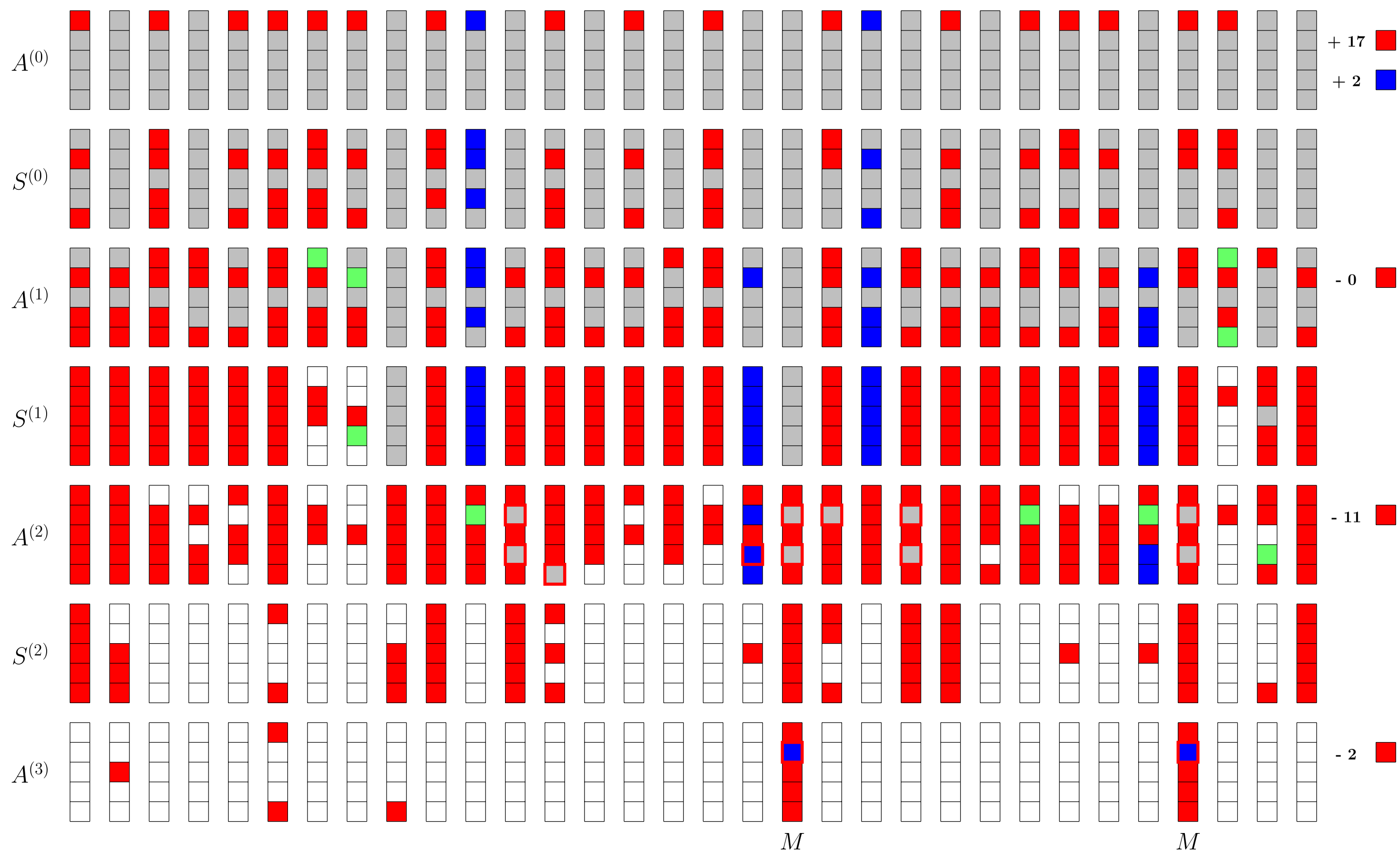

Fig. 26: The 4-round Preimage attack on Ascon-XOF. Since we explore the symmetry in the x-axis to speed up the search by cutting the full 64-bit word into 32-bit word, only 32-bit state version of Ascon-XOF is displayed here.

$a_0 + a_2 + a_4$ become ■ for matching points, so that $l^- = 26 + 4 = 30$. Therefore, $\mathrm{DoF}^+ = 4$, $\mathrm{DoF}^- = 4$, and there are $\mathrm{DoM} = 4$ matching bits.

Using the new TA, we can get 14 free variables and 20 dependent variables, the matrices before and after TA are shown in Eq. (38). The 4-round attack is given in Algorithm 16. The total time complexity is about $2^{128-\min\{\mathrm{DoF}^+,\mathrm{DoF}^-,\mathrm{DoM}\}} = 2^{124}$ time and 2^{14} memory.

$$
\begin{array}{c}
\left(\begin{array}{c|ccccccccccccccccc}
 & v_0 & v_2 & v_4 & v_5 & v_6 & v_7 & v_9 & v_{12} & v_{14} & v_{16} & v_{19} & v_{22} & v_{24} & v_{25} & v_{26} & v_{28} & v_{29} \\
\hline
A^{(2)}_{\{11,1\}} & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
A^{(2)}_{\{18,1\}} & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
A^{(2)}_{\{19,1\}} & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\
A^{(2)}_{\{21,1\}} & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
A^{(2)}_{\{28,1\}} & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\
A^{(2)}_{\{11,3\}} & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
A^{(2)}_{\{17,3\}} & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
A^{(2)}_{\{18,3\}} & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
A^{(2)}_{\{21,3\}} & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
A^{(2)}_{\{28,3\}} & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
A^{(2)}_{\{12,4\}} & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
A^{(3)}_{\{31,4\}} & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
A^{(3)}_{\{31,4\}} & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{array}\right) \\
(a) \\
\left(\begin{array}{c|cccccccccc|ccccccc}
 & v_0 & v_6 & v_{19} & v_{22} & v_{26} & v_9 & v_{28} & v_5 & v_7 & v_{12} & v_2 & v_4 & v_{14} & v_{16} & v_{24} & v_{25} & v_{29} \\
\hline
A^{(3)}_{\{31,4\}} & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
A^{(3)}_{\{31,4\}} & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
A^{(2)}_{\{11,3\}} & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
\hline
A^{(2)}_{\{17,3\}} & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
A^{(2)}_{\{12,4\}} & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
A^{(2)}_{\{19,1\}} & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
A^{(2)}_{\{28,1\}} & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\
A^{(2)}_{\{18,3\}} & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\
A^{(2)}_{\{28,3\}} & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
A^{(2)}_{\{21,1\}} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\
A^{(2)}_{\{11,1\}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\
A^{(2)}_{\{21,3\}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
A^{(2)}_{\{18,1\}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0
\end{array}\right) \\
(b)
\end{array}
\tag{38}
$$

Partial Experiment of the Preimage Attack on 4-round Ascon-XOF. In this practical experiment, we attempt to find a preimage of 24-bit all-zero target. That is, the 24-bit $S^{(3)}_{\{i,0\}}$ $(i \in [16-19, 28-31, 48-63])$ of target are fixed to be zero. The attack procedures are as follows:

1. Set the 256-bit inner part in $A^{(0)}$ as fixed values, which satisfied the 50-bit condition, *i.e.*, $A^{(0)}_{\{*,1\}} = \texttt{0x8d0a0aa08d0a0aa0}$, $A^{(0)}_{\{*,3\}} = \texttt{0x890218ec890218ec}$, and $A^{(0)}_{\{*,2\}} = A^{(0)}_{\{*,4\}} = \texttt{0x0}$. In order to simplify the expressions for TA, we

Algorithm 16: Preimage Attack on 4-round $\mathsf{Ascon\text{-}XOF}$

Compute $S^{(3)}_{\{*,0\}} = p_L^{-1}(T)$
`/*` T `is the hash value */`
for 2^{ζ} *values of* (M_1, M_2) `/*` $\zeta = 116$ `*/`
do
    Compute the inner part of the 3rd block
    if *the 50 conditions are satisfied* `/* with probability of` 2^{-50} `*/`
    then
        for 2^{24} *values of* ■ *bits in* $\mathsf{A}^{(0)}$ **do**
            for $(e_1, e_2, \cdots, e_{20}) \in \mathbb{F}_2^{20}$ **do**
                $U \leftarrow [\ ]$
                for *14 free variables* ■ *in* $\mathsf{A}^{(0)} \in 2^{14}$ **do**
                    Assign $(e_1, e_2, \cdots, e_{20})$ to the last 20 expressions in Eq. (38) (b)
                    Deduce 20 dependent variables
                    Compute forward to other 10-bit ■/■ $\mathbf{u} \in \mathbb{F}_2^{10}$
                    $U[\mathbf{u}] \leftarrow \mathsf{A}^{(0)}[v_{\mathcal{R}}]$ `/*` $v_{\mathcal{R}}$ `represents all 34 red indexes */`
                end
                for $\mathbf{u} \in \mathbb{F}_2^{10}$ **do**
                    $L \leftarrow [\ ]$
                    for $(\mathsf{A}^{(0)}_{\{10,0\}}, \mathsf{A}^{(0)}_{\{20,0\}}, \mathsf{A}^{(0)}_{\{42,0\}}, \mathsf{A}^{(0)}_{\{52,0\}}) \in \mathbb{F}_2^4$ **do**
                        Compute forward to the 4-bit matching point v
                        $L[v] \leftarrow (\mathsf{A}^{(0)}_{\{10,0\}}, \mathsf{A}^{(0)}_{\{20,0\}}, \mathsf{A}^{(0)}_{\{42,0\}}, \mathsf{A}^{(0)}_{\{52,0\}})$
                    end
                    for *values in* $U[\mathbf{u}]$ **do**
                        Compute forward to the 4-bit matching point v'
                        for *values in* $L[v']$ **do**
                            Check if T is satisfied
                        end
                    end
                end
            end
        end
    end
end

set the 2 $\blacksquare$ bits $A^{(0)}_{\{11,0\}} = A^{(0)}_{\{50,0\}} = 1$, and the remaining 22 $\blacksquare$ bits to 0. The simplified equations are given in Eq. (39).

2. In our experiment, we traverse the values of the first 10-bit expressions $(e_1, e_2, \cdots, e_{10})$ and fix $e_{11} = e_{12} = \cdots = c_{20} = 0$ in Eq. (39), *i.e.*, $2^{10+14+4} = 2^{28}$ states are tested with MitM approach and it is expected to find $2^{28-24} = 2^4$ preimages.

Finally, there are $17 = 2^{4.1}$ preimages are searched to satisfy the 24-bit all-zero target, which is close to the theoretical expectation. The results are listed in Table 7. The experiment can be run on a computer with an i9-13900KF CPU and 32 GB of memory in seconds. The source codes are available via `https://anonymous.4open.science/r/Triangulation-MitM-7373`.

Round	First row of preimage $\left(A^{(0)}_{\{*,0\}}\right)$	24-bit target $S^{(3)}_{\{i,0\}}$ $(i \in [16-19, 28-31, 48-63])$
$r = 4$	8a5a10a889023a8a	000000
	0e1812e80e48aa4e	000000
	0d5880e40762a0aa	000000
	2d1018282a48b08e	000000
	007810c42668ba62	000000
	2d5a90ec8c02284a	000000
	8e380aa80e603a8e	000000
	a538888c8c28a866	000000
	a73888a88c20a206	000000
	ab589240a42032aa	000000

Table 7: 24-bit Partial Target Preimage Examples of 4-round Ascon-Xof

$$
\left\{
\begin{array}{ll}
e_1 = A^{(2)}_{\{49,3\}} = & A^{(0)}_{\{0,0\}} \oplus A^{(0)}_{\{22,0\}} \oplus A^{(0)}_{\{39,0\}} \oplus A^{(0)}_{\{57,0\}} \\
e_2 = A^{(2)}_{\{17,3\}} = & A^{(0)}_{\{0,0\}} \oplus A^{(0)}_{\{25,0\}} \oplus A^{(0)}_{\{32,0\}} \\
e_3 = A^{(2)}_{\{44,4\}} = & A^{(0)}_{\{6,0\}} \left(A^{(0)}_{\{26,0\}} \oplus A^{(0)}_{\{48,0\}}\right) \oplus A^{(0)}_{\{5,0\}} A^{(0)}_{\{16,0\}} \oplus A^{(0)}_{\{5,0\}} A^{(0)}_{\{25,0\}} \oplus A^{(0)}_{\{26,0\}} A^{(0)}_{\{28,0\}} \oplus A^{(0)}_{\{28,0\}} A^{(0)}_{\{48,0\}} \oplus \\
& A^{(0)}_{\{37,0\}} \left(A^{(0)}_{\{5,0\}} \oplus A^{(0)}_{\{9,0\}} \oplus A^{(0)}_{\{37,0\}}\right) \oplus A^{(0)}_{\{5,0\}} \oplus A^{(0)}_{\{34,0\}} \oplus A^{(0)}_{\{37,0\}} \oplus A^{(0)}_{\{44,0\}} \oplus A^{(0)}_{\{48,0\}} \oplus \\
& A^{(0)}_{\{44,0\}} \left(A^{(0)}_{\{5,0\}} \oplus A^{(0)}_{\{16,0\}} \oplus A^{(0)}_{\{25,0\}} \oplus A^{(0)}_{\{37,0\}} \oplus A^{(0)}_{\{44,0\}}\right) \oplus 1 \\
e_4 = A^{(2)}_{\{12,4\}} = & A^{(0)}_{\{5,0\}} \left(A^{(0)}_{\{5,0\}} \oplus A^{(0)}_{\{12,0\}} \oplus A^{(0)}_{\{37,0\}} \oplus A^{(0)}_{\{41,0\}}\right) \oplus A^{(0)}_{\{12,0\}} \left(A^{(0)}_{\{12,0\}} \oplus A^{(0)}_{\{37,0\}} \oplus A^{(0)}_{\{48,0\}} \oplus A^{(0)}_{\{57,0\}}\right) \oplus \\
& A^{(0)}_{\{16,0\}} A^{(0)}_{\{38,0\}} \oplus A^{(0)}_{\{37,0\}} A^{(0)}_{\{48,0\}} \oplus A^{(0)}_{\{37,0\}} A^{(0)}_{\{57,0\}} \oplus A^{(0)}_{\{38,0\}} A^{(0)}_{\{58,0\}} \oplus A^{(0)}_{\{16,0\}} A^{(0)}_{\{60,0\}} \oplus A^{(0)}_{\{58,0\}} A^{(0)}_{\{60,0\}} \oplus \\
& A^{(0)}_{\{2,0\}} \oplus A^{(0)}_{\{5,0\}} \oplus A^{(0)}_{\{12,0\}} \oplus A^{(0)}_{\{16,0\}} \oplus A^{(0)}_{\{37,0\}} \oplus A^{(0)}_{\{41,0\}} \oplus 1 \\
e_5 = A^{(2)}_{\{51,1\}} = & A^{(0)}_{\{12,0\}} \left(A^{(0)}_{\{2,0\}} \oplus A^{(0)}_{\{12,0\}} \oplus A^{(0)}_{\{34,0\}} \oplus A^{(0)}_{\{37,0\}} \oplus A^{(0)}_{\{41,0\}}\right) \oplus A^{(0)}_{\{2,0\}} A^{(0)}_{\{37,0\}} \oplus A^{(0)}_{\{51,0\}} \left(A^{(0)}_{\{34,0\}} \oplus A^{(0)}_{\{41,0\}}\right) \oplus \\
& A^{(0)}_{\{54,0\}} \left(A^{(0)}_{\{34,0\}} \oplus A^{(0)}_{\{37,0\}} \oplus A^{(0)}_{\{41,0\}} \oplus A^{(0)}_{\{44,0\}} \oplus A^{(0)}_{\{54,0\}}\right) \oplus A^{(0)}_{\{57,0\}} \left(A^{(0)}_{\{37,0\}} \oplus A^{(0)}_{\{44,0\}} \oplus A^{(0)}_{\{54,0\}}\right) \oplus \\
& A^{(0)}_{\{2,0\}} \oplus A^{(0)}_{\{5,0\}} \oplus A^{(0)}_{\{12,0\}} \oplus A^{(0)}_{\{34,0\}} \oplus A^{(0)}_{\{37,0\}} \oplus A^{(0)}_{\{41,0\}} \oplus A^{(0)}_{\{48,0\}} \oplus A^{(0)}_{\{54,0\}} \oplus A^{(0)}_{\{57,0\}} \oplus 1 \\
e_6 = A^{(2)}_{\{19,1\}} = & A^{(0)}_{\{19,0\}} \left(A^{(0)}_{\{2,0\}} \oplus A^{(0)}_{\{9,0\}}\right) \oplus A^{(0)}_{\{22,0\}} \left(A^{(0)}_{\{2,0\}} \oplus A^{(0)}_{\{5,0\}} \oplus A^{(0)}_{\{9,0\}} \oplus A^{(0)}_{\{12,0\}} \oplus A^{(0)}_{\{22,0\}}\right) \oplus \\
& A^{(0)}_{\{25,0\}} \left(A^{(0)}_{\{5,0\}} \oplus A^{(0)}_{\{12,0\}} \oplus A^{(0)}_{\{22,0\}}\right) \oplus A^{(0)}_{\{44,0\}} \left(A^{(0)}_{\{2,0\}} \oplus A^{(0)}_{\{5,0\}} \oplus A^{(0)}_{\{9,0\}} \oplus A^{(0)}_{\{34,0\}} \oplus A^{(0)}_{\{44,0\}}\right) \oplus \\
& A^{(0)}_{\{5,0\}} A^{(0)}_{\{34,0\}} \oplus A^{(0)}_{\{2,0\}} \oplus A^{(0)}_{\{5,0\}} \oplus A^{(0)}_{\{9,0\}} \oplus A^{(0)}_{\{16,0\}} \oplus A^{(0)}_{\{22,0\}} \oplus A^{(0)}_{\{25,0\}} \oplus A^{(0)}_{\{34,0\}} \oplus A^{(0)}_{\{37,0\}} \oplus A^{(0)}_{\{44,0\}} \oplus 1 \\
e_7 = A^{(2)}_{\{60,1\}} = & A^{(0)}_{\{2,0\}} \oplus A^{(0)}_{\{14,0\}} \oplus A^{(0)}_{\{22,0\}} \oplus A^{(0)}_{\{41,0\}} \oplus A^{(0)}_{\{44,0\}} \oplus A^{(0)}_{\{56,0\}} \oplus A^{(0)}_{\{57,0\}} \\
e_8 = A^{(2)}_{\{28,1\}} = & A^{(0)}_{\{9,0\}} \oplus A^{(0)}_{\{12,0\}} \oplus A^{(0)}_{\{14,0\}} \oplus A^{(0)}_{\{24,0\}} \oplus A^{(0)}_{\{25,0\}} \oplus A^{(0)}_{\{34,0\}} \oplus A^{(0)}_{\{46,0\}} \oplus A^{(0)}_{\{54,0\}} \oplus A^{(0)}_{\{56,0\}} \oplus 1 \\
e_9 = A^{(2)}_{\{50,3\}} = & A^{(0)}_{\{36,0\}} \oplus A^{(0)}_{\{58,0\}} \oplus 1 \\
e_{10} = A^{(2)}_{\{18,3\}} = & A^{(0)}_{\{4,0\}} \oplus A^{(0)}_{\{26,0\}} \\
e_{11} = A^{(2)}_{\{60,3\}} = & A^{(0)}_{\{4,0\}} \oplus A^{(0)}_{\{36,0\}} \oplus A^{(0)}_{\{41,0\}} \oplus A^{(0)}_{\{46,0\}} \oplus 1 \\
e_{12} = A^{(2)}_{\{28,3\}} = & A^{(0)}_{\{9,0\}} \oplus A^{(0)}_{\{14,0\}} \oplus A^{(0)}_{\{36,0\}} \oplus 1 \\
e_{13} = A^{(2)}_{\{53,1\}} = & A^{(0)}_{\{7,0\}} \oplus A^{(0)}_{\{12,0\}} \oplus A^{(0)}_{\{25,0\}} \oplus A^{(0)}_{\{28,0\}} \oplus A^{(0)}_{\{34,0\}} \oplus A^{(0)}_{\{37,0\}} \oplus A^{(0)}_{\{46,0\}} \oplus A^{(0)}_{\{56,0\}} \\
e_{14} = A^{(2)}_{\{21,1\}} = & A^{(0)}_{\{2,0\}} \oplus A^{(0)}_{\{5,0\}} \oplus A^{(0)}_{\{14,0\}} \oplus A^{(0)}_{\{39,0\}} \oplus A^{(0)}_{\{44,0\}} \oplus A^{(0)}_{\{46,0\}} \oplus A^{(0)}_{\{57,0\}} \oplus A^{(0)}_{\{60,0\}} \\
e_{15} = A^{(2)}_{\{43,1\}} = & A^{(0)}_{\{5,0\}} \oplus A^{(0)}_{\{36,0\}} \oplus A^{(0)}_{\{39,0\}} \\
e_{16} = A^{(2)}_{\{11,1\}} = & A^{(0)}_{\{4,0\}} \oplus A^{(0)}_{\{7,0\}} \oplus A^{(0)}_{\{14,0\}} \oplus A^{(0)}_{\{36,0\}} \oplus A^{(0)}_{\{37,0\}} \oplus 1 \\
e_{17} = A^{(2)}_{\{53,3\}} = & A^{(0)}_{\{12,0\}} A^{(0)}_{\{25,0\}} \oplus A^{(0)}_{\{12,0\}} A^{(0)}_{\{34,0\}} \oplus A^{(0)}_{\{25,0\}} A^{(0)}_{\{46,0\}} \oplus A^{(0)}_{\{34,0\}} A^{(0)}_{\{46,0\}} \oplus A^{(0)}_{\{4,0\}} \oplus \\
& A^{(0)}_{\{14,0\}} \oplus A^{(0)}_{\{36,0\}} \oplus A^{(0)}_{\{39,0\}} \oplus A^{(0)}_{\{46,0\}} \oplus A^{(0)}_{\{56,0\}} \oplus A^{(0)}_{\{61,0\}} \\
e_{18} = A^{(2)}_{\{21,3\}} = & A^{(0)}_{\{2,0\}} A^{(0)}_{\{14,0\}} \oplus A^{(0)}_{\{2,0\}} A^{(0)}_{\{44,0\}} \oplus A^{(0)}_{\{14,0\}} A^{(0)}_{\{57,0\}} \oplus A^{(0)}_{\{44,0\}} A^{(0)}_{\{57,0\}} \oplus A^{(0)}_{\{2,0\}} \oplus A^{(0)}_{\{7,0\}} \\
& \oplus A^{(0)}_{\{14,0\}} \oplus A^{(0)}_{\{24,0\}} \oplus A^{(0)}_{\{29,0\}} \oplus A^{(0)}_{\{36,0\}} \oplus A^{(0)}_{\{46,0\}} \oplus A^{(0)}_{\{57,0\}} \\
e_{19} = A^{(2)}_{\{50,1\}} = & A^{(0)}_{\{4,0\}} \oplus A^{(0)}_{\{12,0\}} \oplus A^{(0)}_{\{14,0\}} \oplus A^{(0)}_{\{25,0\}} \oplus A^{(0)}_{\{34,0\}} \oplus A^{(0)}_{\{46,0\}} \oplus A^{(0)}_{\{56,0\}} \oplus 1 \\
e_{20} = A^{(2)}_{\{18,1\}} = & A^{(0)}_{\{2,0\}} \oplus A^{(0)}_{\{4,0\}} \oplus A^{(0)}_{\{14,0\}} \oplus A^{(0)}_{\{36,0\}} \oplus A^{(0)}_{\{44,0\}} \oplus A^{(0)}_{\{46,0\}} \oplus A^{(0)}_{\{57,0\}}
\end{array}
\right. \tag{39}
$$

I MitM Preimage Attacks on Reduced Keccak

I.1 Memory Improved Preimage attack on 4-round Keccak[1024]

We reuse the same initialize structure of the attack on Keccak[1024] in [46], and get a new characteristic with lower attack memory, which is shown in Figure 27 (first part) and Figure 28 (second part). The starting state $\mathsf{A}^{(0)}$ contains 16 ■ bits and 216 ■ bits, and there are totally 464 conditions on ■ bits of $\pi^{(0)}$. Due to the CP-kernel property, the initial degree of freedoms (DoFs) of ■ and ■ are $\lambda^{+} = 8$ and $\lambda^{-} = 108$. In the computation $\mathsf{A}^{(0)}$ to $\mathsf{A}^{(3)}$, the consumed degrees of freedom (DoFs) of ■ and ■ are $l^{+} = 0$ and $l^{-} = 100$ bits, respectively. Therefore, $\mathrm{DoF}^{+} = 8,\ \mathrm{DoF}^{-} = 8$, and there are $\mathrm{DoM} = 8$ matching bits.

We use two message blocks (M_1, M_2) to build the attack following the framework of Qin *et al.* [45] in Figure 22. Given an inner part, the 464 conditions can be seen as a linear system of $1600 - 512 \times 2 - 116 = 460$ variables of M_2 (512×2 is the number of capacity bits, 116 are the DoFs of $\color{red}\blacksquare$ and $\color{blue}\blacksquare$ bits in $\mathsf{A}^{(0)}$), which will act as global parameters. The rank of the coefficient matrix of the linear system is 250. In other words, through some linear transformations, there are $464 - 250 = 214$ equations determined only by the bits of the inner part. To find a right inner part, we have to randomly test 2^{214} M_1 and there are $2^{460-250} = 2^{210}$ solutions of M_2, which make all the conditions hold.

Using TA, we can get 52 free variables and 56 dependent variables. We put the TA matrices in MATRIX/keccak[1024]_4r_eu23.txt at `https://anonymous.4open.science/r/Triangulation-MitM-7373`.

The steps for the 4-round MitM attack are given in Algorithm 17. The total preimage attack on 4-round Keccak[1024] is about $2^{512-\min\{\mathrm{DoF}^+,\mathrm{DoF}^-,\mathrm{DoM}\}} = 2^{504}$ time and 2^{52} memory.

I.2 MitM Preimage Attack on 4-round Keccak[1024]

Using the same model in [46], we get a new characteristic as shown in Figure 29. Due to the symmetry, in the full MitM path, the starting state $\mathsf{A}^{(0)}$ contains 24 $\color{blue}\blacksquare$ bits and 424 $\color{red}\blacksquare$ bits, and there are totally 338 conditions on $\color{gray}\blacksquare$ bits of $\pi^{(0)}$. Due to the CP-kernel property, the initial degree of freedoms (DoFs) of $\color{blue}\blacksquare$ and $\color{red}\blacksquare$ are $\lambda^+ = 12$ and $\lambda^- = 212$. In the computation $\mathsf{A}^{(0)}$ to $\mathsf{A}^{(3)}$, the consumed degrees of freedom (DoFs) of $\color{blue}\blacksquare$ and $\color{red}\blacksquare$ are $l^+ = 0$ and $l^- = 200$ bits, respectively. Therefore, $\mathrm{DoF}^+ = 12,\ \mathrm{DoF}^- = 12$, and there is $\mathrm{DoM} = 12$ matching bits.

We introduce 224 binary variables $v = \{v_0, v_1, \cdots, v_{223}\}$ and 224 binary variables $c = \{c_0, c_1, \cdots, c_{223}\}$. Those variables v_i's and c_i's are placed at the $24 + 424 = 448$ $\color{blue}\blacksquare$ and $\color{red}\blacksquare$ bits in $\mathsf{A}^{(0)}$. For example, set $\mathsf{A}^{(0)}_{\{0,0,0\}} = v_0$ and $\mathsf{A}^{(0)}_{\{0,1,0\}} = v_0 \oplus c_0$ due to the CP-kernel property.

We use two message blocks (M_1, M_2) to build the attack. Given a value for the inner part of the 2nd block, the 338 conditions on $\mathsf{A}^{(0)}$ will be a linear system on 352 variables of M_2, including $576 - 448 = 128$ $\color{gray}\blacksquare$ bits and 224 Binary variables $c = \{c_0, c_1, \cdots, c_{223}\}$. The rank of the coefficient matrix of the linear system is 266. There are $338 - 266 = 72$ conditions out of the total 338 only determined by the value of the inner part. Randomly test 2^{72} M_1 to expect one satisfying the 72 conditions of the inner part. Then for the right M_1, there are $2^{352-266} = 2^{86}$ solutions of M_2, which make all the 338 equations hold.

Using TA algorithm, we can get 118 free variables and 94 dependent variables. We put the TA matrices in MATRIX/keccak[1024]_4r.txt at `https://anonymous.4open.science/r/Triangulation-MitM-7373`.

The steps for the 4-round MitM attack are given in Algorithm 18. To find a 512-bit target preimage, we need $2^{\zeta_1 - 72 + \zeta_2 + \zeta_3 + 106 + 12 + 12} = 2^{512}$. Set $\zeta_1 = 274$, $\zeta_2 = 86$ and $\zeta_3 = 94$. The total preimage attack on 4-round Keccak[1024] is about $2^{512-\min\{\mathrm{DoF}^+,\mathrm{DoF}^-,\mathrm{DoM}\}} = 2^{500}$ time and 2^{118} memory.

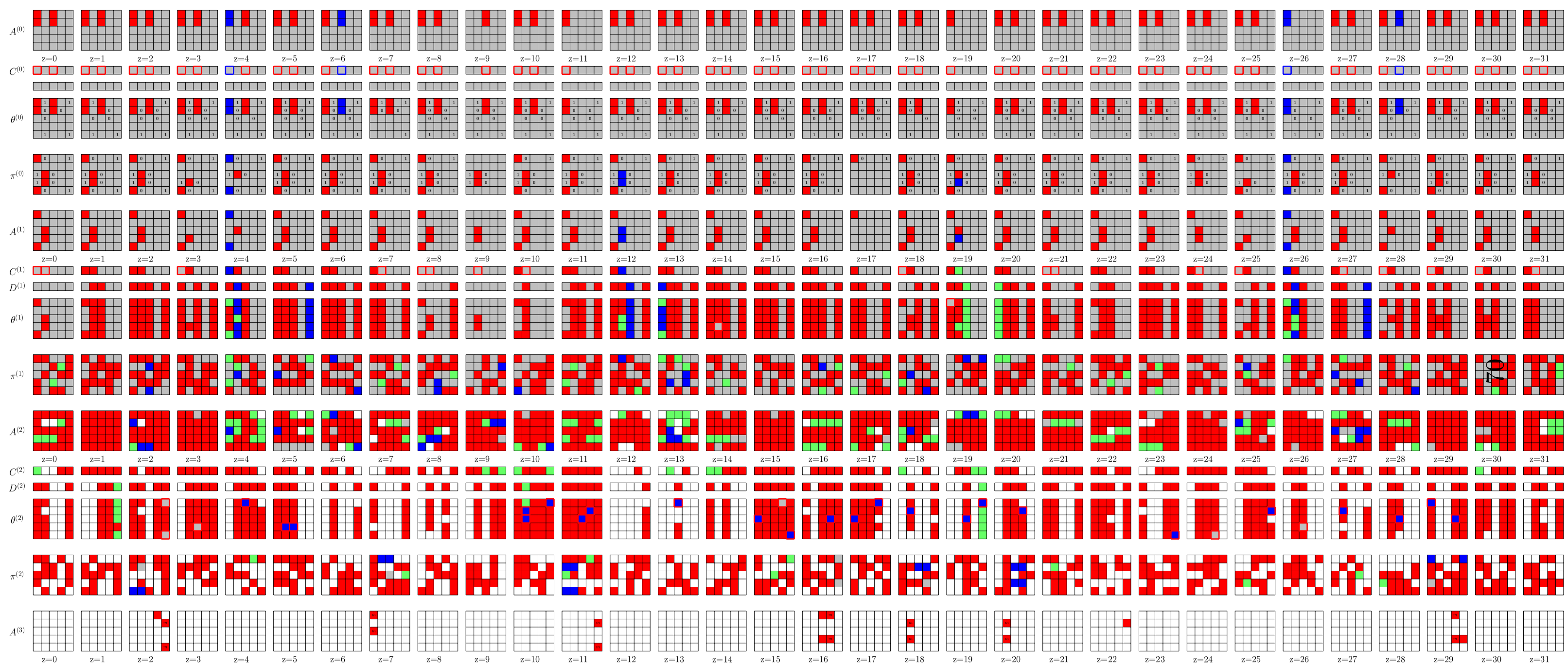

Fig. 27: Part-1: The 4-round MitM preimage attack on KECCAK[1024]

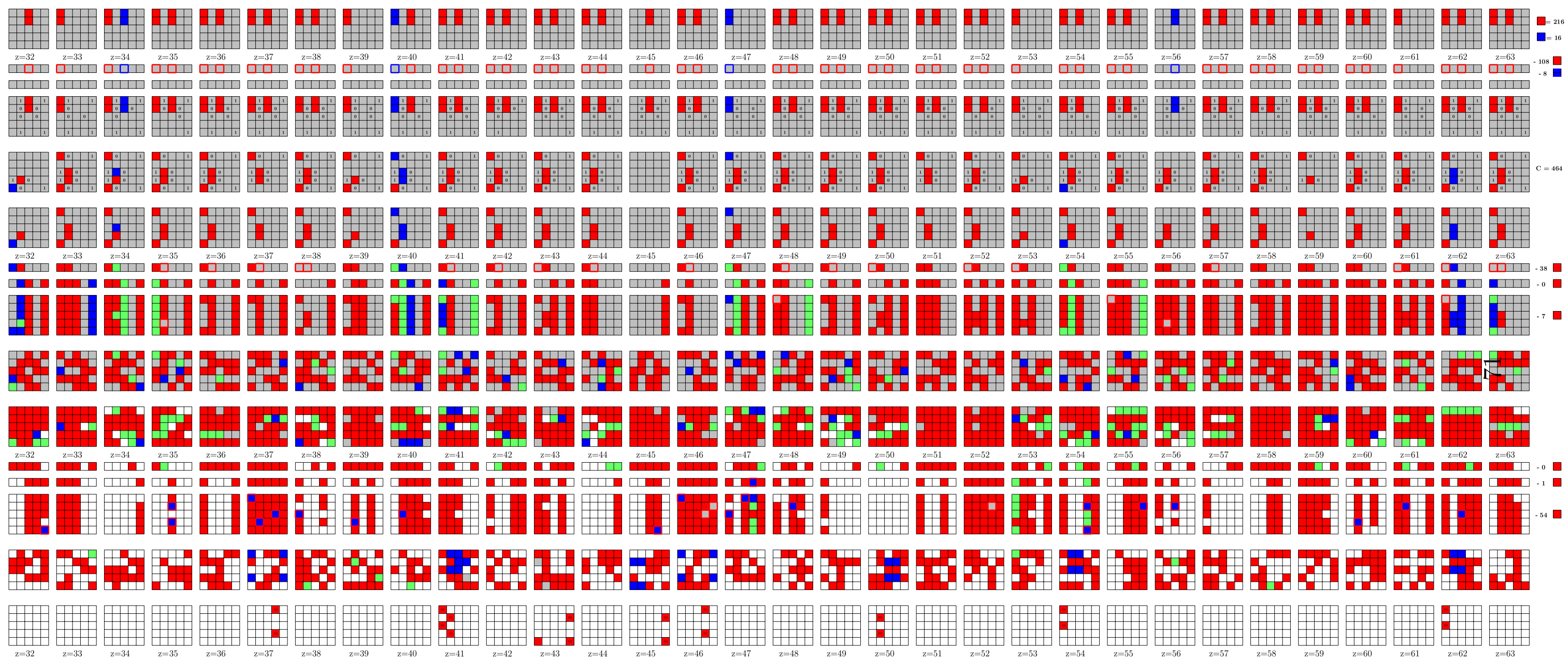

Fig. 28: Part-2: The 4-round MitM preimage attack on KECCAK[1024]

Algorithm 17: Memory Improved Preimage Attack on 4-round Keccak[1024]

Precompute inversely from the target to $\mathsf{A}^{(3)}$

for 2^{ζ_1} *values of* M_1 `/*` $\zeta_1 = 400$ `*/`

do

    Compute the inner part of the 2nd block and solve the system of 464 linear equations

    if *the equations have solutions* `/* with probability of` 2^{-214} `*/`

    then

        for *each of the* 2^{ζ_2} *solutions of* M_2 `/*` $\zeta_2 = 210 \leq 210$ `*/`

        do

            for 2^{ζ_3} *values* $(e_1, e_2, \cdots, e_{56}) \in \mathbb{F}_2^{56}$ **do**

                $U \leftarrow [\ ]$

                for *52 free variables* ■ *in* $\mathsf{A}^{(0)} \in 2^{52}$ **do**

                    Assign $(e_1, e_2, \cdots, e_{56})$ to 56 expressions

                    Deduce 56 dependent variables

                    Compute forward to other 44-bit ■/■ $\mathbf{u} \in \mathbb{F}_2^{44}$

                    $U[\mathbf{u}] \leftarrow A^{(0)}[v_{\mathcal{R}}]$ `/*` $v_{\mathcal{R}}$ `represents all 216 red indexes */`

                end

                for $\mathbf{u} \in \mathbb{F}_2^{44}$ **do**

                    $L \leftarrow [\ ]$

                    for $\mathsf{A}^{(0)}[v_{\mathcal{B}}] \in 2^8$ `/*` $v_{\mathcal{B}}$ `represents all 8 blue indexes */`

                    do

                        Compute forward to the 8-bit matching point v

                        $L[v] \leftarrow A^{(0)}[v_{\mathcal{B}}]$

                    end

                    for *values in* $U[\mathbf{u}]$ **do**

                        Compute forward the 8-bit matching point v'

                        for *values in* $L[v']$ **do**

                            if *it leads to the given hash value* **then**

                                Output the preimage

                            end

                        end

                    end

                end

            end

        end

    end

end

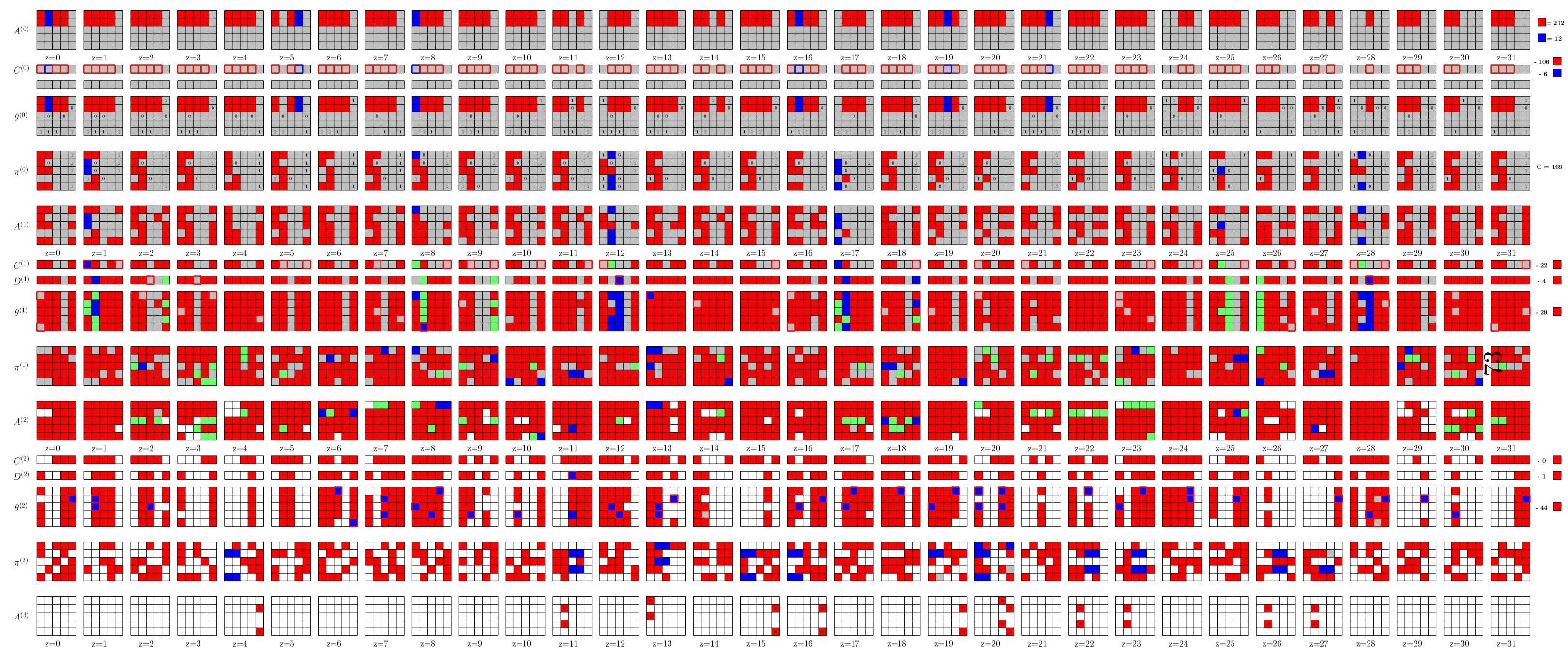

Fig. 29: The 4-round MitM preimage attack on Keccak[1024]

Algorithm 18: The Preimage Attack on 4-round KECCAK[1024]

Precompute inversely from the target to $\mathsf{A}^{(3)}$
for 2^{ζ_1} *values of* M_1 /* $\zeta_1 = 274$ */
do
    Compute the inner part of the 2nd block and solve the system of 338 linear equations
    if *the equations have solutions* /* with probability of 2^{-72} */
    then
        for *each of the* 2^{ζ_2} *solutions of* M_2 /* $\zeta_2 = 86 \leq 86$ */
        do
            for 2^{ζ_3} *values* $(e_1, e_2, \cdots, e_{94}) \in \mathbb{F}_2^{94}$ **do**
                $U \leftarrow [\ ]$
                for *118 free variables* *in* $\mathsf{A}^{(0)} \in 2^{118}$ **do**
                    Assign $(e_1, e_2, \cdots, e_{94})$ to 94 expressions
                    Deduce 94 dependent variables
                    Compute forward to other 106-bit / $\mathbf{u} \in \mathbb{F}_2^{106}$
                    $U[\mathbf{u}] \leftarrow A^{(0)}[v_{\mathcal{R}}]$ /* $v_{\mathcal{R}}$ represents all 212 red indexes */
                end
                for $\mathbf{u} \in \mathbb{F}_2^{106}$ **do**
                    $L \leftarrow [\ ]$
                    for $\mathsf{A}^{(0)}[v_{\mathcal{B}}] \in 2^{12}$ /* $v_{\mathcal{B}}$ represents all 12 blue indexes */
                    do
                        Compute forward to the 12-bit matching point v
                        $L[v] \leftarrow A^{(0)}[v_{\mathcal{B}}]$
                    end
                    for *values in* $U[\mathbf{u}]$ **do**
                        Compute forward the 12-bit matching point v'
                        for *values in* $L[v']$ **do**
                            if *it leads to the given hash value* **then**
                                Output the preimage
                            end
                        end
                    end
                end
            end
        end
    end
end

I.3 MitM Preimage Attacks on 4-round Keccak[768]

Using the same model in [46], we get a new characteristic as shown in Figure 30 (first part) and Figure 31 (second part). In the full path, the starting state $\mathsf{A}^{(0)}$ contains 28 ■ bits and 512 ■ bits, and there are totally 428 conditions on ■ bits of $\pi^{(0)}$. Due to the CP-kernel property, the initial degree of freedoms (DoFs) of ■ and ■ are $\lambda^+ = 17$ and $\lambda^- = 316$. In the computation $\mathsf{A}^{(0)}$ to $\mathsf{A}^{(3)}$, the consumed degrees of freedom (DoFs) of ■ and ■ are $l^+ = 0$ and $l^- = 299$ bits, respectively. Therefore, $\mathrm{DoF}^+ = 17,\ \mathrm{DoF}^- = 17$, and there are $\mathrm{DoM} = 17$ matching bits.

We introduce 333 binary variables $v = \{v_0, v_1, \cdots, v_{332}\}$ and 207 binary variables $c = \{c_0, c_1, \cdots, c_{206}\}$. Those variables v_i's and c_i's are placed at the $28+512 = 540$ ■ and ■ bits in $\mathsf{A}^{(0)}$. For example, set $\mathsf{A}^{(0)}_{\{0,0,0\}} = v_0$, $\mathsf{A}^{(0)}_{\{0,1,0\}} = v_1$ and $\mathsf{A}^{(0)}_{\{0,2,0\}} = v_0 \oplus v_1 \oplus c_0$ due to the CP-kernel property.

We use two message blocks (M_1, M_2) to build the attack. Given an inner part, the 428 conditions can be seen as a linear system of $1600 - 384 \times 2 - 333 = 499$ variables of M_2 (384×2 is the number of capacity bits, 333 are the DoFs of ■ and ■ bits in $\mathsf{A}^{(0)}$), which will act as global parameters, including 292 ■ of M_2 and 207 binary variables $c_{x,z}$'s. The rank of the coefficient matrix of the linear system is 343. In other words, through some linear transformations, there are $428 - 343 = 85$ equations determined only by the bits of the inner part. We have to randomly test 2^{85} M_1 to compute a right inner part satisfying the 85 equations. Then for a right inner part, there are $2^{499-343} = 2^{156}$ solutions of M_2, which make all the conditions hold.

Using TA, we can get 157 free variables and 159 dependent variables. We put the TA matrices in MATRIX/keccak[768]_4r.txt at https://anonymous.4open.science/r/Triangulation-MitM-7373.

The steps for the 4-round MitM attack are given in Algorithm 19. To find a 384-bit target preimage, we need $2^{\zeta_1 - 85 + \zeta_2 + \zeta_3 + 140 + 17 + 17} = 2^{384}$. Set $\zeta_1 = 85$, $\zeta_2 = 51$ and $\zeta_3 = 159$. The total preimage attack on 4-round Keccak[768] is about $2^{384 - \min\{\mathrm{DoF}^+, \mathrm{DoF}^-, \mathrm{DoM}\}} = 2^{367}$ time and 2^{157} memory.

J MitM Preimage Attacks on 10-round Gimli

The 10-round MitM characteristic of `Gimli-XOF-128` is shown in Figure 34. In the MitM path, the starting state $\mathsf{A}^{(0)}$ contains $\lambda^+ = 6$ ■ bit and $\lambda^- = 64$ ■ bits. There are 63 conditions on $\mathsf{A}^{(0)}$ following Qin *et al.* MitM framewok in Figure 22. In the computation from $\mathsf{A}^{(0)}$ to $\mathsf{A}^{(9)}$, the consumed degrees of freedom (DoFs) of ■ and ■ are $l^+ = 0$ and $l^- = 61$ bits, respectively. Therefore, $\mathrm{DoF}^+ = 6,\ \mathrm{DoF}^- = 3$, and there is $\mathrm{DoM} = 3$ matching bits.

The steps for the 10-round MitM attack are given in Algorithm 20. The total preimage attack on 10-round `Gimli-XOF-128` is about $2^{128 - \min\{\mathrm{DoF}^+, \mathrm{DoF}^-, \mathrm{DoM}\}} = 2^{125}$ time and 2^{64} memory.

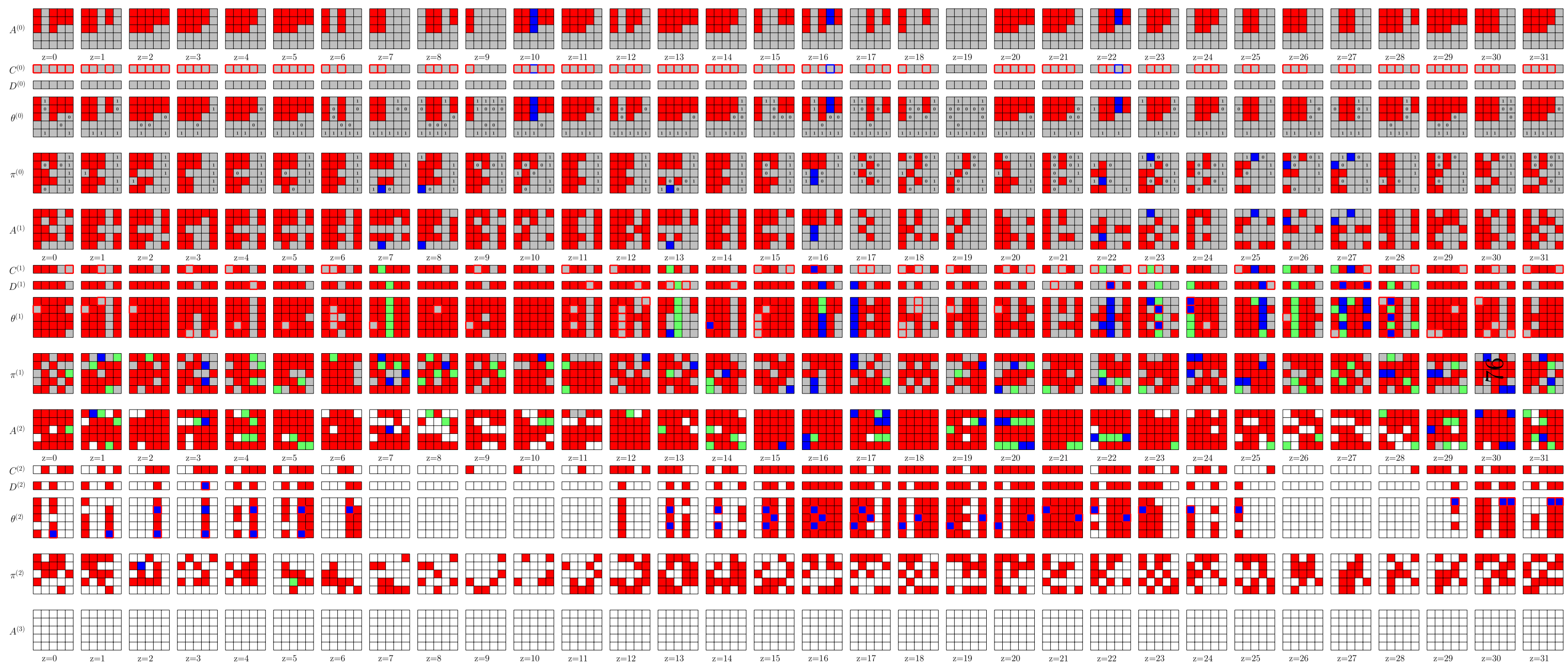

Fig. 30: Part-1: The 4-round Low Memory MitM preimage attack on KECCAK[768]

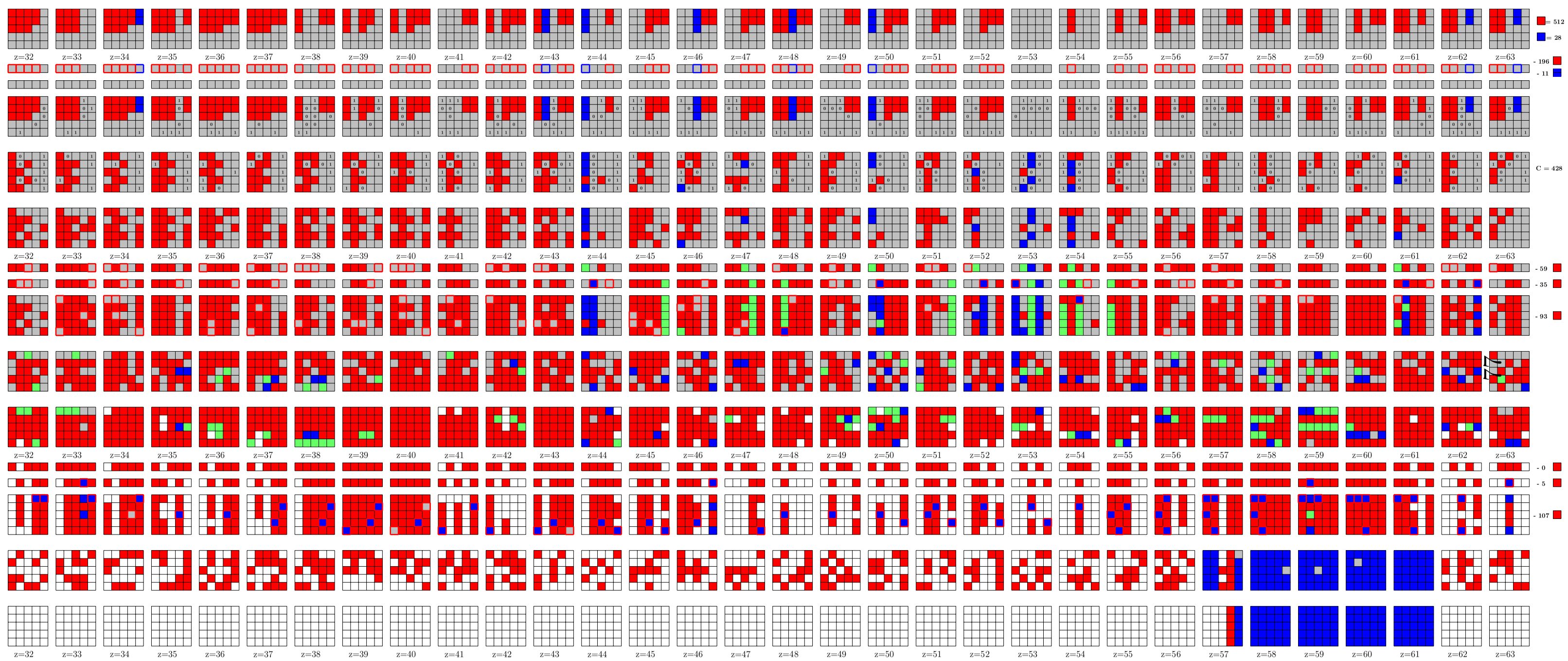

Fig. 31: Part-2: The 4-round Low Memory MitM preimage attack on KECCAK[768]

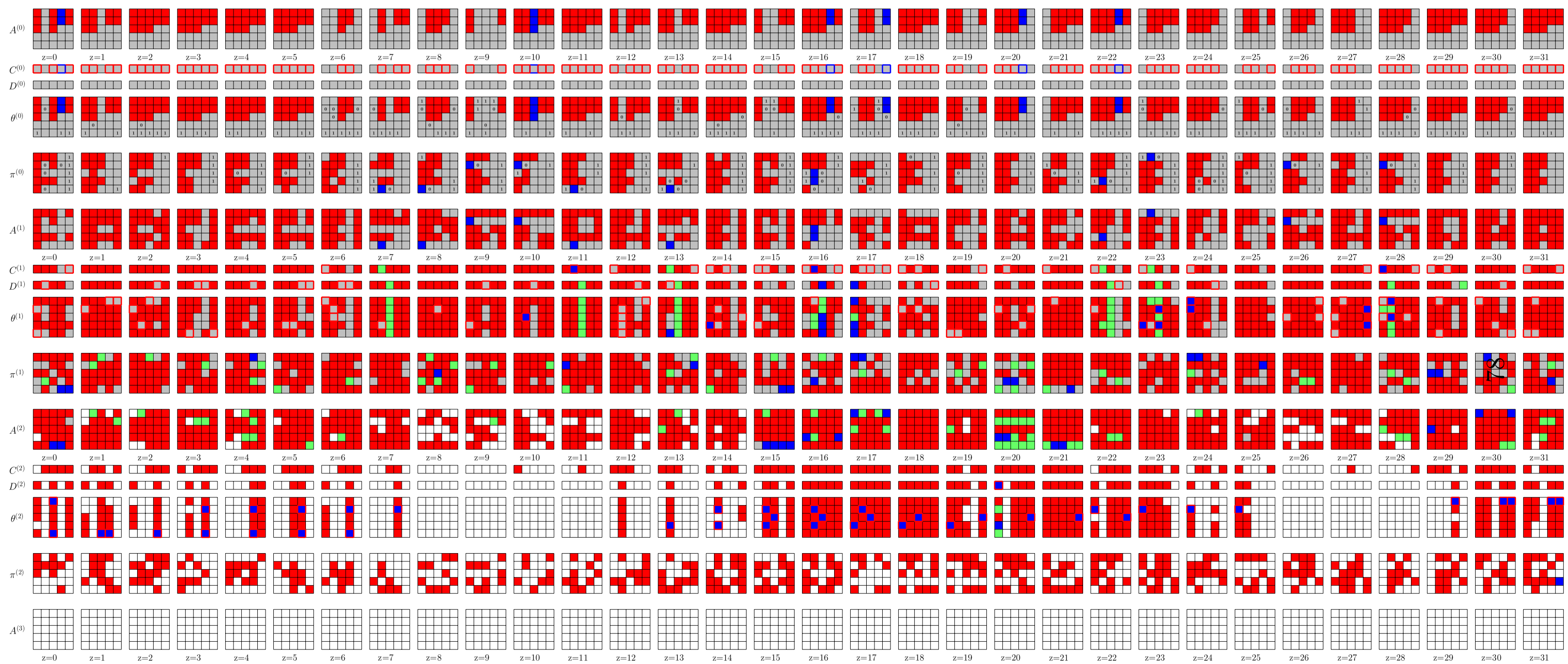

Fig. 32: Part-1: The 4-round MitM preimage attack on KECCAK[768]

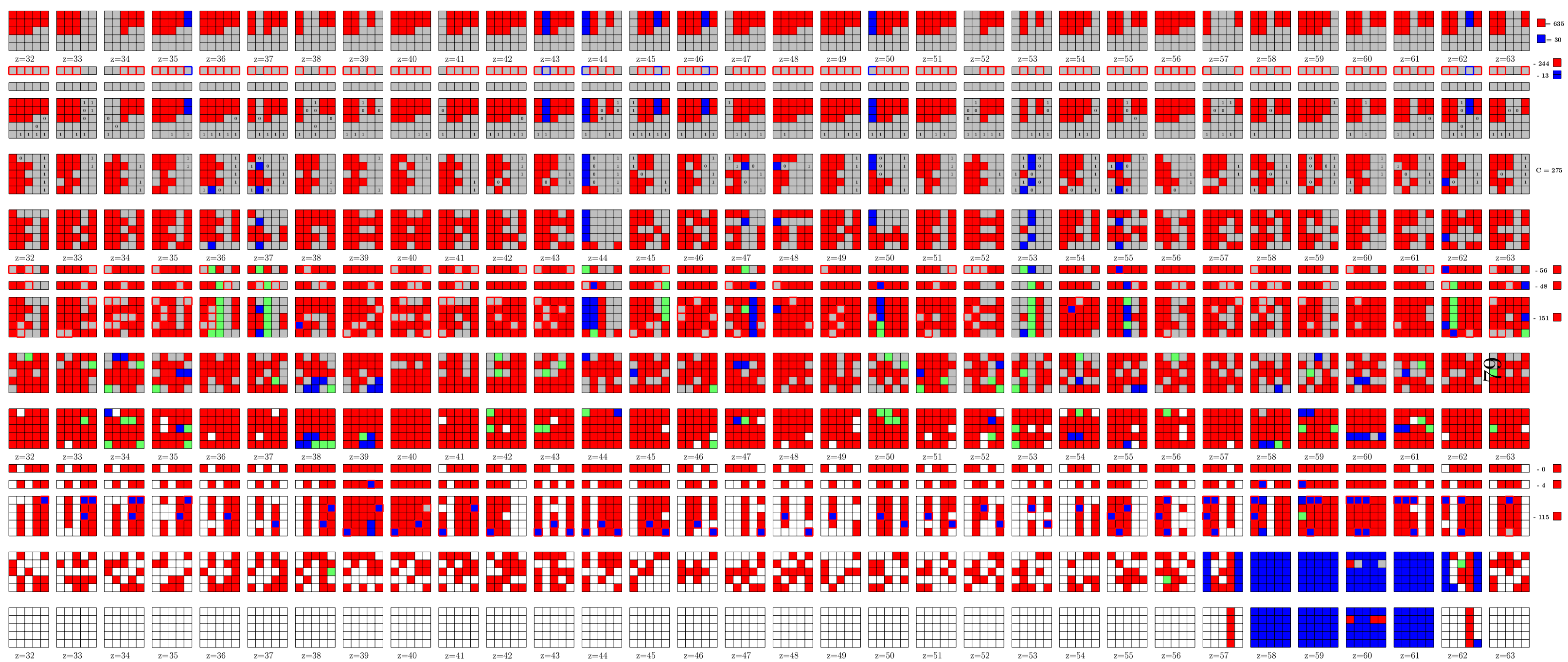

Fig. 33: Part-2: The 4-round MitM preimage attack on KECCAK[768]

Algorithm 19: The Low Memory Preimage Attack on 4-round Keccak[768]

Precompute inversely from the target to $\theta^{(3)}$
for 2^{ζ_1} *values of* M_1 `/*` $\zeta_1 = 85$ `*/`
do
  Compute the inner part of the 2nd block and solve the system of 428 linear equations
  if *the equations have solutions* `/* with probability of` 2^{-85} `*/`
  then
    for *each of the* 2^{ζ_2} *solutions of* M_2 `/*` $\zeta_2 = 51 \leq 156$ `*/`
    do
      for 2^{ζ_3} *values* $(e_1, e_2, \cdots, e_{159}) \in \mathbb{F}_2^{159}$ **do**
        $U \leftarrow [\,]$
        for *157 free variables* *in* $\mathsf{A}^{(0)} \in 2^{157}$ **do**
          Assign $(e_1, e_2, \cdots, e_{159})$ to 159 expressions
          Deduce 159 dependent variables
          Compute forward to other 140-bit / $\mathbf{u} \in \mathbb{F}_2^{140}$
          $U[\mathbf{u}] \leftarrow A^{(0)}[v_{\mathcal{R}}]$`/*` $v_{\mathcal{R}}$ `represents all 316 red indexes */`
        end
        for $\mathbf{u} \in \mathbb{F}_2^{140}$ **do**
          $L \leftarrow [\,]$
          for $\mathsf{A}^{(0)}[v_{\mathcal{B}}] \in 2^{17}$`/*` $v_{\mathcal{B}}$ `represents all 17 blue indexes */`
          do
            Compute forward to the 17-bit matching point v
            $L[v] \leftarrow \mathsf{A}^{(0)}[v_{\mathcal{B}}]$
          end
          for *values in* $U[\mathbf{u}]$ **do**
            Compute forward the 17-bit matching point v'
            for *values in* $L[v']$ **do**
              if *it leads to the given hash value* **then**
                Output the preimage
              end
            end
          end
        end
      end
    end
  end
end

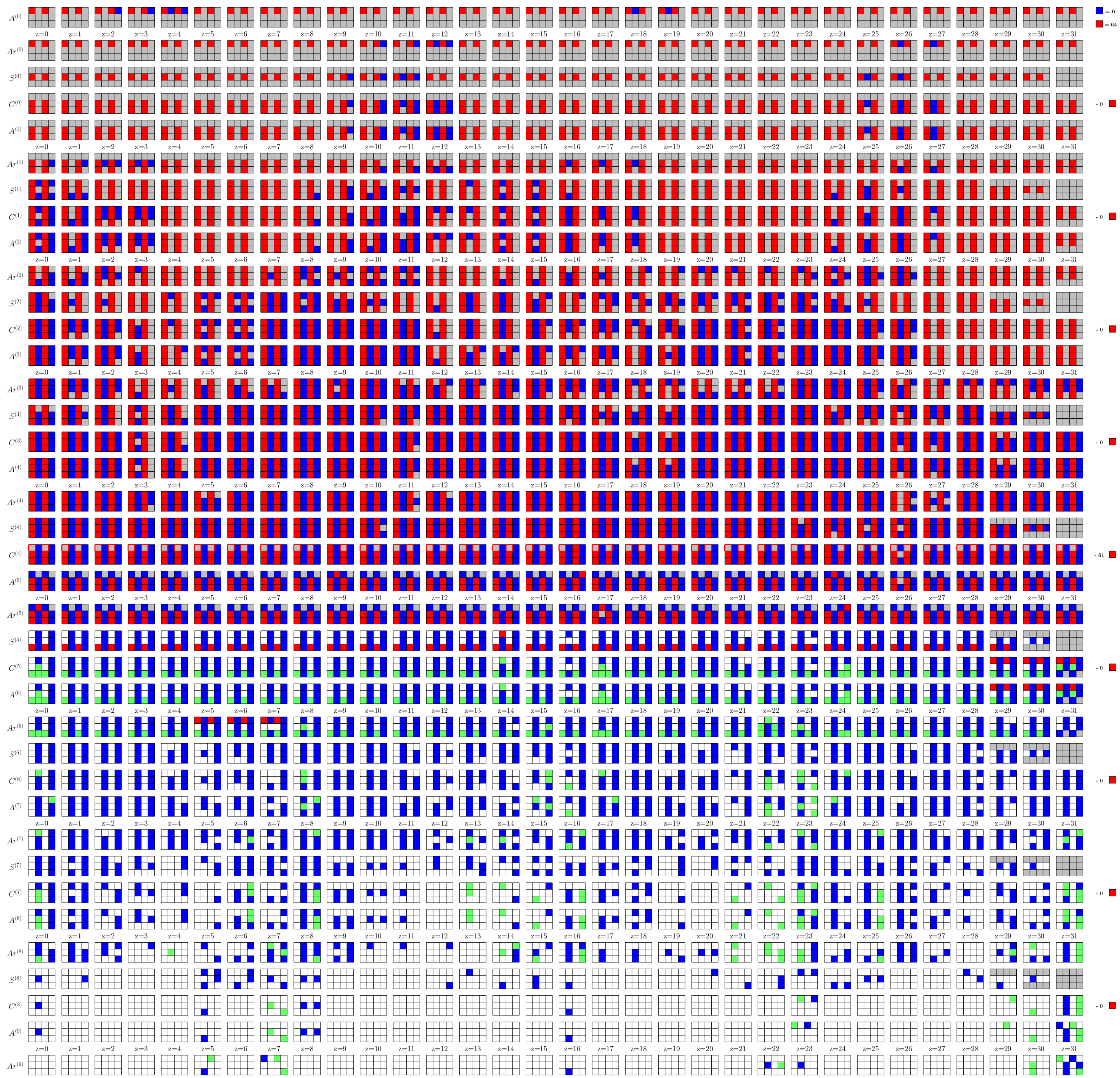

Fig. 34: The 10-round Preimage attack on `Gimli-XOF-128`

Algorithm 20: The Preimage Attack on 10-round Gimli-XOF-128

Precompute inversely from the target to $\mathsf{A}^{(9)}$

for 2^{71} *values of* M_1 `/*` $\zeta_1 = 71$ `*/`

do

  Compute the inner part of the 2nd block and solve the system of 63 linear equations

  if *the equations have solutions* `/* with probability of` 2^{-63} `*/`

  then

    for 2^{50} *values of the* ■ *bits in* M_2 **do**

      $U \leftarrow [\ ]$

      for *64 free variables* ■ *in* $\mathsf{A}^{(0)} \in 2^{64}$ **do**

        Compute forward to determine the 61-bit ■/■ $\mathbf{u} \in \mathbb{F}_2^{61}$

        $U[\mathbf{u}] \leftarrow \mathsf{A}^{(0)}[v_{\mathcal{R}}]$`/*` $v_{\mathcal{R}}$ `represents all 64 red indexes */`

      end

      for $\mathbf{u} \in \mathbb{F}_2^{61}$ **do**

        $L \leftarrow [\ ]$

        for $\mathsf{A}^{(0)}[v_{\mathcal{B}}] \in 2^6$`/*` $v_{\mathcal{B}}$ `represents all 6 blue indexes */`

        do

          Compute forward to the 3-bit matching point v

          $L[v] \leftarrow \mathsf{A}^{(0)}[v_{\mathcal{B}}]$

        end

        for *values in* $U[\mathbf{u}]$ **do**

          Compute forward the 3-bit matching point v'

          for *values in* $L[v']$ **do**

            if *it leads to the given hash value* **then**

              Output the preimage

            end

          end

        end

      end

    end

  end

end

K MITM Preimage Attack on Subterranean 2.0

`Subterranean 2.0`, designed by Daemen et al. [16], is a second round candidate of NIST LWC. For `Subterranean-XOF` with 256-bit digest ($n = 256$, $b = 257$, $r = 9$, $r' = 32$, $c = 248$), Lefevre and Mennink proved a tight bound of 224-bit preimage security [38] under ideal permutation model. The round function consists of four operations: $\chi : s_i \leftarrow s_i + (s_{i+1} + 1)s_{i+2}$, ι : constant addition, $\theta : s_i \leftarrow s_i + s_{i+3} + s_{i+8}$, and $\pi : s_i \leftarrow s_{12i}$. The internal state of each round is updated as : $A^{(r)} \xrightarrow{\chi \circ \iota} \chi^{(r)} \xrightarrow{\theta} \theta^{(r+1)} \xrightarrow{\pi} A^{(r+1)}$. The output function is $z_i = s_{12^{4i}} + s_{-12^{4i}}$, $(0 \leqslant i < 32)$ as shown in Table 8, e.g., when $i = 0$, $z_0 = s_1 + s_{256}$. After each 32-bit digest is squeezed out, 1-round function is executed to update the internal state.

Since DoFs are only consumed through the XOR operation, χ and θ operations can be represented by $\chi^* : s_i^* \leftarrow (s_{i+1} + 1)s_{i+2}$ and $\theta^* : s_i \leftarrow s_i + s_i^* + s_{i+3} + s_{i+3}^* + s_{i+8} + s_{i+8}^*$. Thus, the DoFs consumption can be only restricted in the θ^* operation. Here, we show a preimage attack against `Subterranean-XOF` as shown in Figure 35. The starting state starts at the internal state where the first 32-bit digest T_1 is squeezed out, denoted as $\mathsf{A}^{(0)}$. Since each bit of T_1 is deduced by $s_{12^{4i}} + s_{-12^{4i}}$, the MitM attribute of $(s_{12^{4i}}, s_{-12^{4i}})$ in $\mathsf{A}^{(0)}$ can be set as ($\textcolor{blue}{\blacksquare}$,$\textcolor{blue}{\blacksquare}$), ($\textcolor{red}{\blacksquare}$,$\textcolor{red}{\blacksquare}$) or ($\textcolor{gray}{\blacksquare}$,$\textcolor{gray}{\blacksquare}$) and is served as 1-bit degree of freedom. Therefore, there are $\lambda^+ = 114$ $\textcolor{blue}{\blacksquare}$ bits and $\lambda^- = 92$ $\textcolor{red}{\blacksquare}$ bits in $\mathsf{A}^{(0)}$. In the forward computation path, the consumed degrees of freedom (DoFs) of $\textcolor{blue}{\blacksquare}$ and $\textcolor{red}{\blacksquare}$ are $l^+ = 42$ and $l^- = 20$ bits, such that $\text{DoF}^+ = 72$, $\text{DoF}^- = 72$. In the matching phase, 1-bit matching point can be deduced if $(s_{12^{4i}}, s_{-12^{4i}})$ in $\mathsf{A}^{(r)}$ has no unknown $\square$ bit, for $1 \leq r \leq 3$. Hence, the final matching points are counted as $\text{DoM} = 72$ bits and are marked with "m" in Figure 35.

The detailed algorithm is given in Algorithm 21. In Line 25 to Line 36, 2^{72} solutions of $\mathsf{A}^{(0)}$ are left after the 72-bit matching. Hence, we need repeat $2^{\zeta+3+21+19+19} = 2^{224-72-72}$ times, i.e., $\zeta = 18$.

- In Line 2, $\mathsf{A}^{(1)}_{16}$ is a $\textcolor{gray}{\blacksquare}$ bit that is derived by imposing constraint on $\textcolor{red}{\blacksquare}$ bits. However, $\left(\textcolor{blue}{\overline{A^{(1)}_{15}}} \wedge \textcolor{gray}{A^{(1)}_{16}}\right)$ is involved in the constraint imposed on the $\mathsf{A}^{(2)}_{129}$ by consuming 1-bit $\textcolor{blue}{\blacksquare}$ DoF. Hence, it was considered at the outer loop. Similarly, $(A^{(1)}_{26}, A^{(1)}_{47})$ are two $\textcolor{gray}{\blacksquare}$ bits which are derived by imposing constraints on $\textcolor{blue}{\blacksquare}$ bits, but $\left(\textcolor{red}{\overline{A^{(1)}_{25}}} \wedge \textcolor{gray}{A^{(1)}_{26}}\right)$ and $\left(\textcolor{red}{\overline{A^{(1)}_{46}}} \wedge \textcolor{gray}{A^{(1)}_{47}}\right)$ are involved in the constraints imposed on $\mathsf{A}^{(2)}_{66}$ and $\mathsf{A}^{(2)}_{132}$ by consuming 2-bit $\textcolor{red}{\blacksquare}$ DoFs.
- In Line 4 to Line 11, using the TA algorithm, $\mathsf{A}^{(0)}_{200}$ can be determined by free variables $v_{\mathcal{R},1}$ and $\mathsf{A}^{(1)}_{16}$ as Eq. (40). Then, the remaining 86 free $\textcolor{red}{\blacksquare}$ bits are exhausted trivially. The time complexity of constructing table V is $2^{\zeta+3+5+86} = 2^{112}$. The memory complexity is $2^{5+86} = 2^{91}$. Compared with the table-based method in [26], the memory cost is reduced by a factor

of 2.

$$A_{16}^{(1)} = A_{192}^{(0)} \oplus (\overline{A_{193}^{(0,*)}} \wedge A_{194}^{(0)}) \oplus A_{195}^{(0)} \oplus (\overline{A_{196}^{(0)}} \wedge A_{197}^{(0,*)}) \oplus A_{200}^{(0)} \oplus (\overline{A_{201}^{(0)}} \wedge A_{202}^{(0)}) \tag{40}$$

- In Line 12 to Line 21, using the TA algorithm, 23 bits in $v_{\mathcal{B},1}^{*}$ can be determined by the 62 free bits in $v_{\mathcal{B},1}$ and the 21 free bits in $c_{\mathcal{B},1}$ as shown in Table 9. Then, the remaining 29 free $\blacksquare$ bits in $v_{\mathcal{B},2}$ are exhausted trivially. The time complexity of constructing table U is $2^{\zeta+3+21+62+29} = 2^{133}$. The memory complexity is $2^{62+29} = 2^{91}$. Compared with the table-based method in [26], the memory cost is reduced by a factor of 2^{23}.

Once finding such a proper $\mathsf{A}^{(0)}$, an inner collision on 248-bit capacity between $\mathsf{A}^{(0)}$ and the initial value can be found by Floyd's cycle finding algorithm with 2^{124} time and no memory. Therefore, the overall time complexity is about 2^{152}. The memory cost is about 2^{92} to store hash tables U and V.

i	state bits	i	state bits	i	state bits	i	state bits
0	(1, 256)	8	(64, 193)	16	(241, 16)	24	(4, 253)
1	(176, 81)	9	(213, 44)	17	(11, 246)	25	(190, 67)
2	(136, 121)	10	(223, 34)	18	(137, 120)	26	(30, 227)
3	(35, 222)	11	(184, 73)	19	(211, 46)	27	(140, 117)
4	(249, 8)	12	(2, 255)	20	(128, 129)	28	(225, 32)
5	(134, 123)	13	(95, 162)	21	(169, 88)	29	(22, 235)
6	(197, 60)	14	(15, 242)	22	(189, 68)	30	(17, 240)
7	(234, 23)	15	(70, 187)	23	(111, 146)	31	(165, 92)

Table 8: Mapping between state bits of matching phase in `Subterranean 2.0`

$c_{\mathcal{B},1}$	$\theta^{(0)}_{49}$, $\theta^{(0)}_{50}$, $\theta^{(0)}_{51}$, $\theta^{(0)}_{54}$, $\theta^{(0)}_{55}$, $\theta^{(0)}_{56}$, $\theta^{(0)}_{172}$, $\theta^{(0)}_{174}$, $\theta^{(0)}_{175}$, $\theta^{(0)}_{176}$, $\theta^{(0)}_{177}$, $\theta^{(0)}_{179}$, $\theta^{(0)}_{181}$, $\theta^{(0)}_{183}$, $\theta^{(1)}_{21}$, $\theta^{(1)}_{22}$, $\theta^{(1)}_{39}$, $\theta^{(1)}_{58}$, $\theta^{(1)}_{86}$, $\theta^{(1)}_{146}$, $\theta^{(1)}_{147}$, $\theta^{(1)}_{150}$, $\theta^{(1)}_{255}$
$c_{\mathcal{B},2}$	$\theta^{(1)}_{3}$, $\theta^{(1)}_{6}$, $\theta^{(1)}_{10}$, $\theta^{(1)}_{51}$, $\theta^{(1)}_{62}$, $\theta^{(1)}_{66}$, $\theta^{(1)}_{69}$, $\theta^{(1)}_{74}$, $\theta^{(1)}_{90}$, $\theta^{(1)}_{110}$, $\theta^{(1)}_{111}$, $\theta^{(1)}_{129}$, $\theta^{(1)}_{138}$, $\theta^{(1)}_{159}$, $\theta^{(1)}_{170}$, $\theta^{(1)}_{183}$, $\theta^{(1)}_{218}$, $\theta^{(1)}_{219}$, $\theta^{(1)}_{239}$
$v_{\mathcal{B},1}$	$\mathsf{A}^{(0)}_{56}$, $A^{(0)}_{58}$, $A^{(0)}_{62}$, $A^{(0)}_{69}$, $A^{(0)}_{71}$, $A^{(0)}_{72}$, $A^{(0)}_{74}$, $A^{(0)}_{75}$, $A^{(0)}_{77}$, $A^{(0)}_{80}$, $A^{(0)}_{82}$, $A^{(0)}_{83}$, $A^{(0)}_{84}$, $\mathsf{A}^{(0)}_{85}$, $A^{(0)}_{86}$, $\mathrm{A}^{(0)}_{87}$, $A^{(0)}_{88}(A^{(0)}_{169})$, $A^{(0)}_{89}$, $A^{(0)}_{91}$, $A^{(0)}_{92}(A^{(0)}_{165})$, $A^{(0)}_{93}$, $A^{(0)}_{94}$, $A^{(0)}_{96}$, $A^{(0)}_{97}$, $\mathsf{A}^{(0)}_{98}$, $A^{(0)}_{99}$, $A^{(0)}_{101}$, $A^{(0)}_{102}$, $\mathrm{A}^{(0)}_{103}$, $A^{(0)}_{104}$, $A^{(0)}_{105}$, $A^{(0)}_{106}$, $A^{(0)}_{107}$, $A^{(0)}_{108}$, $A^{(0)}_{109}$, $A^{(0)}_{110}$, $\mathsf{A}^{(0)}_{111}(A^{(0)}_{146})$, $A^{(0)}_{112}$, $A^{(0)}_{113}$, $A^{(0)}_{114}$, $A^{(0)}_{115}$, $\mathrm{A}^{(0)}_{116}$, $A^{(0)}_{117}(A^{(0)}_{140})$, $A^{(0)}_{118}$, $A^{(0)}_{119}$, $\mathsf{A}^{(0)}_{120}(A^{(0)}_{137})$, $A^{(0)}_{121}(A^{(0)}_{136})$, $A^{(0)}_{122}$, $A^{(0)}_{123}(A^{(0)}_{134})$, $A^{(0)}_{124}$, $A^{(0)}_{125}$, $\mathrm{A}^{(0)}_{127}$, $A^{(0)}_{129}(A^{(0)}_{128})$, $\mathsf{A}^{(0)}_{130}$, $A^{(0)}_{131}$, $A^{(0)}_{132}$, $A^{(0)}_{135}$, $A^{(0)}_{173}$, $A^{(0)}_{178}$, $A^{(0)}_{179}$, $A^{(0)}_{183}$, $A^{(0)}_{185}$
$v^{*}_{\mathcal{B},1}$	$\mathsf{A}^{(0)}_{57}$, $A^{(0)}_{59}$, $A^{(0)}_{61}$, $A^{(0)}_{63}$, $A^{(0)}_{65}$, $A^{(0)}_{66}$, $A^{(0)}_{76}$, $A^{(0)}_{78}$, $A^{(0)}_{79}$, $A^{(0)}_{90}$, $A^{(0)}_{95}(A^{(0)}_{162})$, $A^{(0)}_{100}$, $\mathsf{A}^{(0)}_{126}$, $A^{(0)}_{133}$, $\mathrm{A}^{(0)}_{172}$, $A^{(0)}_{174}$, $A^{(0)}_{175}$, $A^{(0)}_{176}(A^{(0)}_{81})$, $A^{(0)}_{177}$, $A^{(0)}_{180}$, $A^{(0)}_{181}$, $A^{(0)}_{182}$, $\mathsf{A}^{(0)}_{184}(A^{(0)}_{73})$
$v_{\mathcal{B},2}$	$\mathsf{A}^{(0)}_{138}$, $A^{(0)}_{139}$, $A^{(0)}_{141}$, $A^{(0)}_{142}$, $A^{(0)}_{143}$, $A^{(0)}_{144}$, $A^{(0)}_{145}$, $A^{(0)}_{147}$, $A^{(0)}_{148}$, $A^{(0)}_{149}$, $A^{(0)}_{150}$, $A^{(0)}_{151}$, $\mathsf{A}^{(0)}_{152}$, $A^{(0)}_{153}$, $\mathrm{A}^{(0)}_{154}$, $A^{(0)}_{155}$, $A^{(0)}_{156}$, $A^{(0)}_{157}$, $A^{(0)}_{158}$, $A^{(0)}_{159}$, $A^{(0)}_{160}$, $A^{(0)}_{161}$, $A^{(0)}_{163}$, $A^{(0)}_{164}$, $\mathsf{A}^{(0)}_{166}$, $A^{(0)}_{167}$, $A^{(0)}_{168}$, $A^{(0)}_{170}$, $\mathrm{A}^{(0)}_{171}$,

Table 9: The disjoint subsets of constraints and variables for triangulation

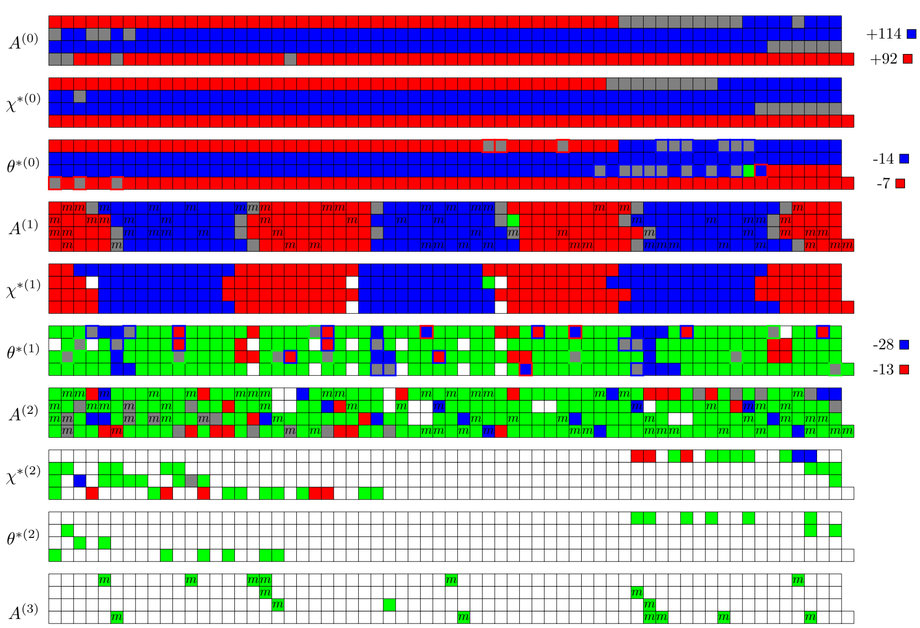

Fig. 35: The Full Round Preimage Attack on `Subterranean-XOF`

Algorithm 21: Preiamge Attack on Full Round Subterranean

for 2^{ζ} *possible values of the* ■ *cells in* $\mathsf{A}^{(0)}$, $\zeta = 18$ **do**
  for 2^3 *possible values of* $\mathsf{A}^{(1)}_{16} \| A^{(1)}_{26} \| A^{(1)}_{47}$ **do**
    $U \leftarrow [\,]$, $V \leftarrow [\,]$
    for *5-bit values of* $v_{\mathcal{R},1} = (A^{(0)}_{194}, A^{(0)}_{195}, A^{(0)}_{196}, A^{(0)}_{201}, A^{(0)}_{202})$ **do**
      Deduce 1-bit value of $\mathsf{A}^{(0)}_{200}$ according to $\mathsf{A}^{(1)}_{16}$
      for $v_{\mathcal{R},2} \in \mathbb{F}_2^{86}$ *of the remaining* ■ *cells in* $\mathsf{A}^{(0)}$ **do**
        Compute forward to the 19-bit $\mathbf{u}$ with $\mathsf{A}^{(1)}_{26}$ and $\mathsf{A}^{(1)}_{47}$
        $V[\mathbf{u}] \leftarrow v_{\mathcal{R},1} \| v_{\mathcal{R},2} \| A^{(0)}_{200}$
        /* There are $2^{5+86-19} = 2^{72}$ elements under each index */
      end
    end
    for $g_{\mathcal{B}} \in \mathbb{F}_2^{21}$ *of* $c_{\mathcal{B},1}$ /* except for $\theta^{(0)}_{50} = A^{(1)}_{47}$ and $\theta^{(0)}_{55} = A^{(1)}_{26}$ */
    do
      for *62-bit values of* $v_{\mathcal{B},1}$ **do**
        Deduce 23-bit values of $v^*_{\mathcal{B},1}$
        for $v_{\mathcal{B},2} \in \mathbb{F}_2^{29}$ *of the remaining* ■ *cells in* $\mathsf{A}^{(0)}$ **do**
          Compute forward to the 19-bit $c_{\mathcal{B},2}$ with $\mathsf{A}^{(1)}_{16}$
          $U[c_{\mathcal{B},2}] \leftarrow v_{\mathcal{B},1} \| v^*_{\mathcal{B},1} \| v_{\mathcal{B},2}$
          /* There are $2^{62+29-19} = 2^{72}$ elements under each index */
        end
      end
      for $\mathbf{u} \in \mathbb{F}_2^{19}$ **do**
        for $c_{\mathcal{B},2} \in \mathbb{F}_2^{19}$ **do**
          $L \leftarrow [\,]$
          for $v_{\mathcal{R}} \in V[\mathbf{u}]$ **do**
            Compute forward to the 72-bit matching points $End_{\mathcal{R}}$ and store $v_{\mathcal{R}}$ in L with $End_{\mathcal{R}}$ as index
          end
          for $v_{\mathcal{B}} \in V[c_{\mathcal{B},2}]$ **do**
            Compute forward to the 72-bit matching points $End_{\mathcal{B}}$
            for $v_{\mathcal{R}} \in L[End_{\mathcal{B}}]$ **do**
              Reconstruct the input state with $v_{\mathcal{R}}$ and $v_{\mathcal{B}}$
              if *it leads to the given 224-bit hash value* $(T_2, T_3, \cdots, T_8)$ **then**
                Output the preimage
              end
            end
          end
        end
      end
    end
  end
end