

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

Boxin Zhao¹, Qingliang Hou³, 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

44 attacks on 3-/4-round Ascon are reduced from $2^{24}/2^{34}$ to $2^{14}/2^{14}$, which
 45 have been partially implemented to verify the attacks.

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

48 1 Introduction

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

65 The MitM attack proposed by Diffie and Hellman in 1977 is a time-memory
 66 trade-off cryptanalysis of symmetric-key primitives [21]. Currently, the MitM at-
 67 tack has been successfully applied to block ciphers and hash functions with more
 68 sophisticated techniques, such as the internal state guessing [29], splice-and-cut
 69 [1], initial structure [50], bicliques [8,35], 3-subset MitM [9], (indirect) partial
 70 matching [1,50], guess-and-determine [51,33], sieve-in-the-middle [13], match-
 71 box [31], dissection [23], non-linear initial structure [32], MitM in differential
 72 view [36,30], and differential MitM [12], etc. Automating MitM attack was first
 73 reported in CRYPTO 2011 and 2016 [11,20], which present attacks on AES
 74 with low data complexity, or even a single plaintext-ciphertext pair. In 2018,
 75 Sasaki [48] first tried to automate MitM with Mixed Integer Linear Program-
 76 ming (MILP). At EUROCRYPT 2021, Bao *et al.* [2] built a fully automated
 77 MitM preimage attack using MILP on AES-like hashing. Later, the automated
 78 MitM models were improved with more techniques by Dong *et al.* [26], Bao *et*
 79 *al.* [3], and Chen *et al.* [14], or further developed for the sponge functions by
 80 Schrottenloher and Stevens [52,53], Qin *et al.* [45], and Dong *et al.* [27].

81 In the MitM attacks, as shown in Figure 1, the iterative round-based compu-
 82 tation of the compression function or block cipher is divided at a certain round
 83 (starting point) into two chunks. The two chunks are computed independently
 84 and end at a common matching point. In both chunks, the computation involves
 85 different message words, denoted by N^+ and N^- respectively. So one chunk

86 computes all possible values of the involved message words N^+ independently
87 of the message words N^- involved in the other chunk. The different words N^+
88 and N^- are called the *neutral words*. At EUROCRYPT 2009, Sasaki and Aoki
89 proposed the *initial structure* (IS) technique with the purpose of skipping sev-
90 eral rounds at the beginning of two chunks to enhance the MitM attack [50].
91 As shown in Figure 1, the two chunks are in the opposite direction, and only
92 a few consecutive starting rounds are overlapped, which form the so-called IS.
93 Although the two sets of neutral words N^+ and N^- appear simultaneously at
94 these rounds, they are only involved in the computation of one chunk each. This
95 is achieved by assigning some linear constraints to the values of neutral words of
96 one chunk, such that different values lead to constant impact on the computation
97 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

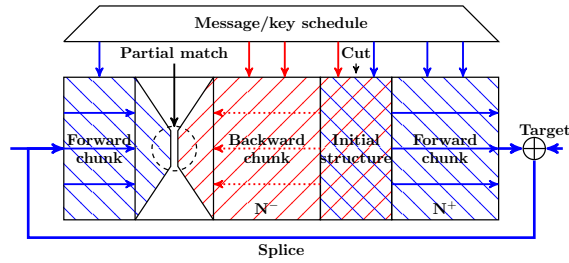


Fig. 1: The Splice-and-cut MitM Attack [1]

98
99 2021, Dong *et al.* [26] noticed that in many potential MitM characteristics, the
100 two sets of neutral words N^+ and N^- are constrained by some nonlinear con-
101 straints, such that different values lead to a constant impact on the computation
102 of the opposite chunk only if the nonlinear system holds. Therefore, Dong *et al.*
103 presented a table-based technique to obtain the constrained space of the values
104 of the neutral words. The drawback of Dong *et al.*'s approach is that it requires
105 a huge amount of memory to prepare two hash tables, and many attacks based
106 on this method need huge memory, *e.g.*, [26,45].

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

111 In this paper, motivated by the structured Gaussian elimination paradigm
112 developed by LaMacchia *et al.* [37] and Bender *et al.* [6], we improve the tri-
113 angulation algorithm to deal with the case where there are still many unpro-
114 cessed equations but no variable exists in only one equation (the original TA
115 will terminate immediately). Moreover, with the help of the improved triangula-
116 tion 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}	2^{24}	2^{128}	Sect. 4.2
	MitM	$4^{\dagger}/10$	1KP	2^{112}	2^{56}	2^{128}	Sect. 4.3
	MitM	$5/10$	1KP	2^{120}	2^{96}	2^{128}	[19]
	MitM	$5/10$	1KP	2^{120}	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	$5/10$	2^8 CP	2^{40}	small	2^{128}	[55]
	R-Boomerang	$5/10$	2^9 ACC	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	$6/12$	2^8 CP	$2^{109.6}$	$2^{109.6}$	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	$6/14$	2^8 CP	2^{122}	2^{113}	2^{256}	[19]
	MitM	$7/14$	2^8 CP	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

\dagger : 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 ($i \geq 0$, $MC^{(-1)} = P$) 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}	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}	2^{52}		Sect. I.1
		Algebraic†	4/24	2^{502}	2^{482}		[22]
		MitM	4/24	2^{500}	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}	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}	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}	2^8		Sect. 4.6
Xoodoo-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}	2^{37}		Sect. G.1
	Collision	MitM	3/12	$2^{60.5}$	$2^{60.5}$	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

- 180 – AddRoundKey (AK): XOR a round key $RK^{(i)}$ into the state $MC^{(i-1)}$ to pro-
 181 duce $A^{(i)}$. The key schedule for AES is given in Figure 15, 16, 17 in Supple-
 182 mentary Material B.
 183 – SubBytes (SB): Substitute each cell of $A^{(i)}$ according to an S-box to get $SB^{(i)}$.
 184 – ShiftRows (SR): For $N_{col} = 4, 6$, rotate the i th row of $SB^{(i)}$ to the left by i
 185 bytes ($i = 0, 1, 2, 3$). For $N_{col} = 8$, rotate the 0, 1, 2, 3rd row to the left by
 186 0, 1, 3, 4 bytes, respectively.
 187 – MixColumns (MC): Update each column of $SR^{(i)}$ by left-multiplying an MDS
 188 matrix shown in Eq. (25) in Supplementary Material B to get $MC^{(i)}$.

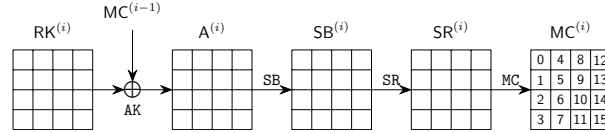


Fig. 2: One Round of AES

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

191 Rijndael. Taking the FAEST-128 as an example, which is based on AES-128, the
 192 plaintext-ciphertext pair $(P, C = \text{AES-128}(k, P))$ is used as the public key of the
 193 signature scheme (verification key) and encryption key k is used as the secret
 194 key (signing key). If an adversary can recover the encryption key k given only
 195 a single plaintext-ciphertext pair (P, C) of AES-128, *i.e.*, the public key of the
 196 signature scheme, then he can compute the secret signing key k . This allows
 197 him to forge a signature by following exactly the honest prover protocol with
 198 the recovered signing key k . This demonstrates that a key recovery attack with
 199 one data complexity on AES-128 leads to a signature forgery on FAEST. In
 200 FAEST-192/-256 based on AES-192/-256, the size of k is larger than the block
 201 size, and FAEST-192/-256 uses two (P, C) pairs as the public key (verification
 202 key). Because one (P, C) pair with only 128-bit information cannot prove a 192
 203 or 256-bit knowledge of the secret signing key k . Therefore, the key-recovery
 204 attack on AES-192/-256 with two plaintext-ciphertext pairs matters for FAEST-
 205 192/-256. FAEST additionally uses Rijndael in Even-Mansour (EM) mode, *i.e.*,
 206 FAEST-EM, where Rijndael block cipher is used as a permutation in EM mode.
 207 The original key of Rijndael block cipher is published as part of the public key
 208 (along with one plaintext-ciphertext), and the new block cipher Rijndael-EM's
 209 key is the secret signing key. Therefore, the original key of Rijndael block cipher
 210 is a known constant when performing the key-recovery attack on Rijndael-EM.

211 2.2 Preliminaries of Basic Meet-in-the-Middle Attack

212 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 $I^\mathcal{E}/I^\mathcal{K}$.
213 $\mathcal{R}^\mathcal{E}/\mathcal{R}^\mathcal{K}$	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^+ = \mathcal{B}^\mathcal{E} + \mathcal{B}^\mathcal{K} $	The initial degree of freedom for the forward computation.
$\lambda^- = \mathcal{R}^\mathcal{E} + \mathcal{R}^\mathcal{K} $	The initial degree of freedom for the backward computation.
214 π^+/π^-	Certain constraints on the starting state.

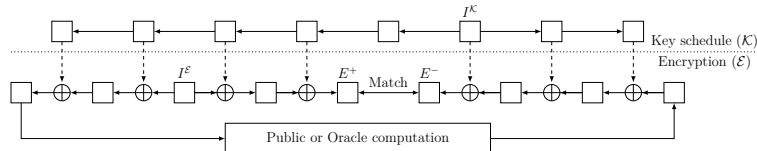


Fig. 3: A high-level overview of the MitM attacks [26]

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 $\mathcal{N} = \{0, 1, \dots, n-1\}$ or $\bar{\mathcal{N}} = \{0, 1, \dots, \bar{n}-1\}$ 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 $\mathcal{G}^\mathcal{K} = \bar{\mathcal{N}} - \mathcal{B}^\mathcal{K} \cup \mathcal{R}^\mathcal{K}$. 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 $\pi^+ = (\pi_1^+, \dots, \pi_{\ell^+}^+)$ 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 $\pi^- = (\pi_1^-, \dots, \pi_{\ell^-}^-)$ 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 $(I^\mathcal{E}[\mathcal{R}^\mathcal{E}], I^\mathcal{K}[\mathcal{R}^\mathcal{K}])$. π^+ and π^- 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 $\mathbf{c}^+ = (a_1, \dots, a_{\ell^+}) \in \mathbb{F}_2^{w \cdot \ell^+}$ and $\mathbf{c}^- = (b_1, \dots, b_{\ell^-}) \in \mathbb{F}_2^{w \cdot \ell^-}$, 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:

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

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 $\mathbf{c}^+ = (a_1, \dots, a_{\ell^+})$, 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

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

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 $\text{DoF}^+ = \lambda^+ - \ell^+$ the *degree of freedom (DoF)*

243 for the forward computation. Similarly, the solution space of $(I^{\mathcal{E}}[\mathcal{R}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{R}^{\mathcal{K}}])$
 244 induced by Eq. (2) is denoted by $\mathbb{R}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathbf{c}^-)$, whose size is $2^{w \cdot (\lambda^- - \ell^-)}$.
 245 We call $\text{DoF}^- = \lambda^- - \ell^-$ the *DoF for the backward computation*. Assume the
 246 computation connecting $E^+[\mathcal{M}^+]$ and $E^-[\mathcal{M}^-]$ forms an m -cell filter, which is
 247 denoted as the degree of matching ($\text{DoM} = m$). The MitM attack is Algorithm
 248 1. To find a preimage of h -cell target, the complexity of Algorithm 1 is about

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

249 where \mathcal{T}_{pre} is the time complexity to precompute $\mathbb{B}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathbf{c}^+)$ and
 250 $\mathbb{R}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathbf{c}^-)$ 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 \mathbf{c}^+ , and \mathbf{c}^- to some constants.
 - 2 Solve Eq. (1) and (2) to obtain $\mathbb{B}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathbf{c}^+)$ and
 $\mathbb{R}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathbf{c}^-)$.
 - 3 For values in $\mathbb{B}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathbf{c}^+)$, compute $E^+[\mathcal{M}^+]$ and insert it into L
 - 4 For values in $\mathbb{R}(I^{\mathcal{E}}[\mathcal{G}^{\mathcal{E}}], I^{\mathcal{K}}[\mathcal{G}^{\mathcal{K}}], \mathbf{c}^-)$, 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.
-

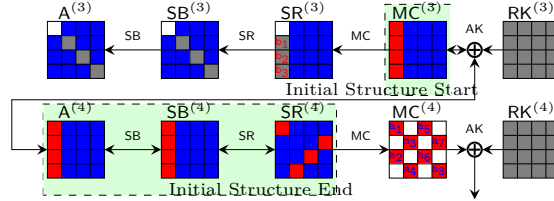


Fig. 4: Sasaki and Aoki's initial structure of MitM attack on AES

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

259 neutral words,

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

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

266 **Dong *et al.*'s Nonlinear Constrained Neutral Words [26].** As noticed by
 267 Dong *et al.* [26], the Eq. (1) and (2) of many interesting MitM characteristics
 268 are nonlinear systems in practice, and there is no efficient method to solve them.
 269 Therefore, Dong *et al.* presented a table-based technique in Algorithm 2 which
 270 can be applied in attacks relying on such MitM characteristics without solving
 271 the equations. The major drawback of Dong *et al.*'s approach is that it would
 272 require a huge amount of memory to prepare two hash tables V and U , and
 273 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

```

1  $V \leftarrow [], U \leftarrow []$ 
2 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
3    $\mathbf{v} \leftarrow \pi^+(I^\mathcal{E}[\mathcal{G}^\mathcal{E}], I^\mathcal{K}[\mathcal{G}^\mathcal{K}], I^\mathcal{E}[\mathcal{B}^\mathcal{E}], I^\mathcal{K}[\mathcal{B}^\mathcal{K}])$  by Eq. (1)
4   Insert  $(I^\mathcal{E}[\mathcal{B}^\mathcal{E}], I^\mathcal{K}[\mathcal{B}^\mathcal{K}])$  into  $V$  at index  $\mathbf{v}$ 
5 end
6 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
7    $\mathbf{u} \leftarrow \pi^-(I^\mathcal{E}[\mathcal{G}^\mathcal{E}], I^\mathcal{K}[\mathcal{G}^\mathcal{K}], I^\mathcal{E}[\mathcal{R}^\mathcal{E}], I^\mathcal{K}[\mathcal{R}^\mathcal{K}])$  by Eq. (2)
8   Insert  $(I^\mathcal{E}[\mathcal{R}^\mathcal{E}], I^\mathcal{K}[\mathcal{R}^\mathcal{K}])$  into  $U$  at index  $\mathbf{u}$ 
9 end
```

274 2.3 Triangulation Algorithm (TA)

275 The triangulation algorithm (TA) was introduced by Khovratovich, Biryukov,
 276 and Nikolic [34] at CT-RSA 2009, which is a Gaussian-like elimination pro-
 277 cess 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 in-
cludes many variables. The more such variables we have among the dependent
variables, the more conditions are satisfied at no cost. The heart of the triangu-
lation 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 sim-
plified 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.

$$\left\{ \begin{array}{l} F(x \oplus s) \oplus v = 0, \\ G(x \oplus u) \oplus s \oplus L(y \oplus z) = 0, \\ v \oplus G(u \oplus s) = 0, \\ H(z \oplus s \oplus v) \oplus t = 0, \\ u \oplus H(t \oplus x) = 0, \end{array} \right. \quad (5) \quad \left\{ \begin{array}{l} L(y \oplus z) \oplus G(u \oplus x) \oplus s = 0, \\ z \oplus H^{-1}(t \oplus v \oplus s) = 0, \\ t \oplus H^{-1}(u \oplus x) \oplus v \oplus s = 0, \\ u \oplus G^{-1}(v \oplus s) \oplus x = 0, \\ v \oplus F(x \oplus s) = 0. \end{array} \right. \quad (6)$$

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

$$P(s \oplus v \oplus t) \oplus z = 0, \quad (7)$$

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} .

$$\text{Before TA: } \begin{pmatrix} s & t & u & v & x & y & z \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \xrightarrow[\text{Khovratovich } et al.'s \text{ TA}]{\text{after extract } y} \begin{pmatrix} y & s & t & u & v & x & z \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ \hline & & & & & & \text{free variables} \end{pmatrix} \quad (8)$$

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.

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

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

- 363 1. **Construct the system of equations:** Given the system of equations, fix
 364 the predefined values to constants.
 365 2. **Initialize all variables and equations as unprocessed:** Mark all vari-
 366 ables and all equations as unprocessed.
 367 3. **Find the variable involved in only one unprocessed equation:**
 368 (a) Search for a variable that appears in only one unprocessed equation. If
 369 such a variable exists, mark the equation and the variable as processed.
 370 (b) *If no such variable can be found, perform the following steps:*
 371 i. *Count the number of variables present in each unprocessed equation.*
 372 ii. *Identify the unprocessed equations that contain the largest number of*
 373 *variables.*
 374 iii. *Remove one of the equations in (ii) from the system and mark it as*
 375 *processed. This reduces the scale of the remaining system.*
 376 4. **Repeat Step 3 until all equations have been processed:** *Continue*
 377 *searching for variables involved in a single equation or removing equations*
 378 *with the maximum number of variables until no unprocessed equations exist.*
 379 5. **Assign free variables:** After all equations are processed, mark all remain-
 380 ing unprocessed variables as free.
 381 6. **Solve the system:** Assign random values to the free variables. Using these
 382 values, compute the remaining variables by substituting them back into the
 383 processed equations.

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

389 Now let's continue to consider the example in Eq. (8), the whole process is
 390 illustrated in Eq. (9). After extracting y , instead of immediately marking the
 391 remaining variables as free, we analyze the number of variables included in each
 392 remaining unprocessed equation and prioritize the equations with the largest
 393 number of variables (4-th and 6-th row in the first matrix of Eq. (9)), which
 394 are highlighted in **bold**. Label one of them as processed and remove it from
 395 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.

$$\begin{array}{ccc}
 \begin{pmatrix} y & s & t & u & v & x & z \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} & \xrightarrow[6th\ row]{remove} & \begin{pmatrix} y & s & t & u & v & x & z \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{pmatrix} \\
 \begin{pmatrix} y & z & t & s & u & v & x \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} & \xrightarrow[u, v]{extract} & \begin{pmatrix} y & z & t & s & u & v & x \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \\
 & & \xrightarrow[z]{extract} & \begin{pmatrix} y & z & t & s & u & v & x \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \\
 & & & \xrightarrow[t]{extract} & \begin{pmatrix} y & z & t & s & u & v & x \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \\
 & & & & \xrightarrow[finally]{extract} & \begin{pmatrix} y & z & t & s & u & v & x \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \\
 & & & & & \text{free}
 \end{array} \tag{9}$$

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.*, $\ell^- = 6$.

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

Following Dong *et al.*'s method in Algorithm 2, we have to traverse 7-byte variables and compute $\mathbf{u} \leftarrow (b_1, b_2, \dots, b_6) \in \mathbb{F}_2^{48}$ 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: $(I^\mathcal{E}[\mathcal{G}^\mathcal{E}], I^\mathcal{K}[\mathcal{G}^\mathcal{K}]) \in \mathbb{F}_2^{w \cdot (|\mathcal{G}^\mathcal{E}| + |\mathcal{G}^\mathcal{K}|)}$

```

1 for  $(b_2, b_6, b_5, b_3, b_1) \in \mathbb{F}_2^{8 \times 5}$  do
2    $V \leftarrow [], U \leftarrow []$ 
3   for  $(x, s) \in \mathbb{F}_2^{8 \times 2}$  do
4     Compute  $v$  from  $\pi_1^-()$ 
5     Compute  $u$  from  $\pi_3^-()$ 
6     Compute  $t$  from  $\pi_5^-()$ 
7     Compute  $z$  from  $\pi_6^-()$ 
8     Compute  $y$  from  $\pi_2^-()$ 
9     Compute  $\mathbf{u} = b_4$  by equations marked by cyan
10    Store  $U[\mathbf{u}] \leftarrow (x, s, v, u, t, z, y)$ 
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 $\mathbb{R}(I^\mathcal{E}[\mathcal{G}^\mathcal{E}], I^\mathcal{K}[\mathcal{G}^\mathcal{K}], \mathbf{c}^-)$, with $|\mathbb{R}(I^\mathcal{E}[\mathcal{G}^\mathcal{E}], I^\mathcal{K}[\mathcal{G}^\mathcal{K}], \mathbf{c}^-)| = 2^{8 \cdot (\lambda^- - \ell^-)} = 2^8$, 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 $(\widehat{A}^{(2)}[4], \widehat{RK}^{(2)}[12, 13], \widehat{A}^{(1)}[3, 4])$. Then, store all the solutions in a hash table under the index of 4-byte value $\mathbf{u} = (\widehat{SR}^{(3)}[1, 4], \widehat{RK}^{(5)}[0], \widehat{A}^{(1)}[14])$ (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 \blacksquare mean linear combinations of \blacksquare and \blacksquare . In the MitM path, the starting state $RK^{(0)}$, whose bytes are denoted as k_0 to k_{15} , contains $\lambda^+ = 4$ \blacksquare bytes and $\lambda^- = 12$ \blacksquare bytes. In the computation from $RK^{(0)}$ to $RK^{(5)}$, from $A^{(0)}$ to $SR^{(2)}$, and from $SR^{(4)}$ to $MC^{(2)}$, the consumed degrees of freedom (DoFs) of \blacksquare and \blacksquare are $\ell^+ = 3$ and $\ell^- = 9$ bytes, respectively. Therefore, $\text{DoF}^+ = 1$, $\text{DoF}^- = 3$, and there is $\text{DoM} = 1$ matching byte in round 2. The 9 consumed DoFs of \blacksquare on $A^{(1)}[3]$, $A^{(1)}[4]$,

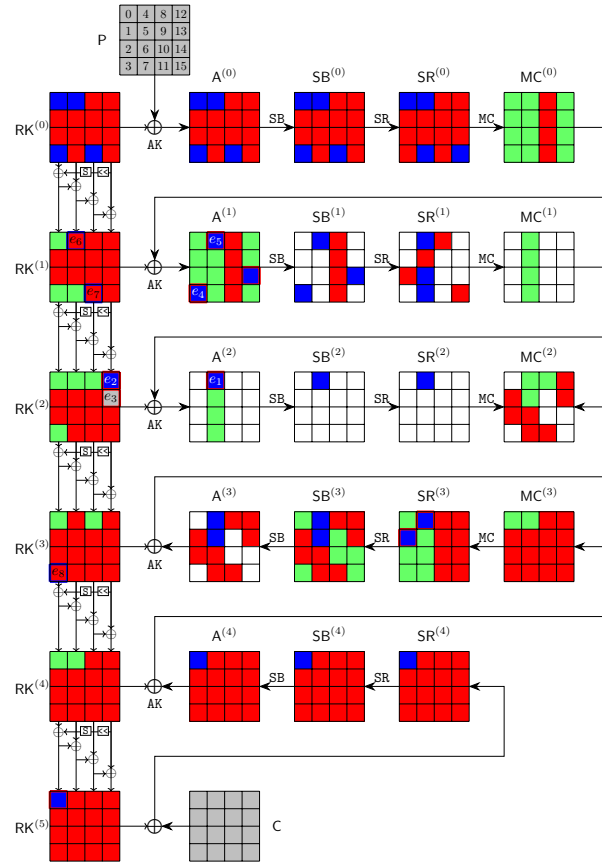


Fig. 5: 5-round attack on AES-128

483 $A^{(1)}[14]$, $RK^{(2)}[12]$, $RK^{(2)}[13]$, $A^{(2)}[4]$, $SR^{(3)}[1]$, $SR^{(3)}[4]$, and $RK^{(5)}[0]$ marked by
 484 $\blacksquare/\blacksquare$ in Figure 5 are a system on the byte-variables of $RK^{(0)}$ in Eq. (11).

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

485 Excluding the parts of Eq. (11) related to blue bytes, we get Eq. (12) which is
 486 only related to the red bytes, where the boxed parts are deleted⁵.

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

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

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

⁵In our MitM attack (see Line 14 to 27 of Algorithm 4), we need the 9 bytes $(A^{(1)}[3], A^{(1)}[4], \dots)$ 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 $(\hat{A}^{(1)}[3], \hat{A}^{(1)}[4], \dots)$, so that the red bytes have a constant effect on the 9 bytes $(A^{(1)}[3], A^{(1)}[4], \dots)$.

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

$$\begin{aligned}
& \left(\begin{array}{c|cccccccccccccc} & 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{A}^{(1)}[3] & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ \widehat{A}^{(1)}[4] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ \widehat{A}^{(1)}[14] & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ \widehat{A}^{(2)}[4] & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ \widehat{RK}^{(2)}[12] & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ \widehat{RK}^{(2)}[13] & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ \widehat{SR}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \widehat{SR}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \widehat{RK}^{(5)}[0] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array} \right) & \left(\begin{array}{c|cccccccccccccc} & 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{SR}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \widehat{SR}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \widehat{RK}^{(5)}[0] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \hline \widehat{A}^{(1)}[3] & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ \widehat{A}^{(1)}[4] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ \widehat{A}^{(1)}[14] & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ \widehat{A}^{(2)}[4] & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ \widehat{RK}^{(2)}[12] & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ \widehat{RK}^{(2)}[13] & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{array} \right) \\
& \quad (a) & \quad (b) \\
& \left(\begin{array}{c|cccccccccccccc} & 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{SR}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \widehat{SR}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \widehat{RK}^{(5)}[0] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \hline \widehat{A}^{(2)}[4] & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ \widehat{A}^{(1)}[3] & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ \widehat{A}^{(1)}[4] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ \widehat{A}^{(1)}[14] & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ \widehat{RK}^{(2)}[12] & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ \widehat{RK}^{(2)}[13] & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{array} \right) & \left(\begin{array}{c|cccccccccccccc} & 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{SR}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \widehat{SR}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \widehat{RK}^{(5)}[0] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \hline \widehat{A}^{(1)}[14] & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ \hline \widehat{A}^{(2)}[4] & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ \widehat{A}^{(1)}[3] & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ \widehat{A}^{(1)}[4] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ \widehat{RK}^{(2)}[12] & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ \widehat{RK}^{(2)}[13] & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{array} \right) \\
& \quad (c) & \quad (d) \\
& \left(\begin{array}{c|cccccccccccccc} & 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{SR}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \widehat{SR}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \widehat{RK}^{(5)}[0] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \hline \widehat{A}^{(1)}[14] & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ \hline \widehat{A}^{(2)}[4] & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ \widehat{RK}^{(2)}[12] & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ \widehat{A}^{(1)}[3] & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ \widehat{A}^{(1)}[4] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ \widehat{RK}^{(2)}[13] & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{array} \right) & \left(\begin{array}{c|cccc|cccccc} & 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{SR}^{(3)}[1] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \widehat{SR}^{(3)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \widehat{RK}^{(5)}[0] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \hline \widehat{A}^{(1)}[14] & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ \hline \widehat{A}^{(2)}[4] & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ \widehat{RK}^{(2)}[12] & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ \widehat{RK}^{(2)}[13] & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ \hline \widehat{A}^{(1)}[3] & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ \widehat{A}^{(1)}[4] & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \end{array} \right) \\
& \quad (e) & \quad (f) \quad \text{free variables}
\end{aligned}$$

503 Finally, we extract 5 dependent variables $\widehat{RK}^{(0)}[7, 1, 2, 5, 9] = (k_7, k_1, k_2, k_5, k_9)$
504 marked in green in Eq. (13)-(f) from the rows of $\widehat{A}^{(2)}[4]$, $\widehat{RK}^{(2)}[12]$, $\widehat{RK}^{(2)}[13]$,
505 $\widehat{A}^{(1)}[3]$, and $\widehat{A}^{(1)}[4]$. The others are 7 free variables $\widehat{RK}^{(0)}[6, 8, 10, 12, 13, 14, 15] =$
506 $(k_6, k_8, k_{10}, k_{12}, k_{13}, k_{14}, k_{15})$. The values $(e_1, e_2, e_3, e_4, e_5) \in \mathbb{F}_2^{40}$ are assigned
507 to the expressions for the red bytes of $(\widehat{A}^{(2)}[4], \widehat{RK}^{(2)}[12], \widehat{RK}^{(2)}[13], \widehat{A}^{(1)}[3],$
508 $\widehat{A}^{(1)}[4])$, then given the values of the seven free variables, the dependent variables
509 $(k_9, k_5, k_2, k_1, k_7)$ are deduced in sequence. Thereafter, the values of $\widehat{SR}^{(3)}[1],$
510 $\widehat{SR}^{(3)}[4], \widehat{RK}^{(5)}[0]$ and $\widehat{A}^{(1)}[14]$ are deduced directly.

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

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

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

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

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

518 *Analysis of Algorithm 4.* In Line 12 to 27, $2^{\zeta+24+16+32+8+8} = 2^{128}$ states should
 519 be tested to recover the 128-bit key; therefore, $\zeta = 40$. According to Eq. (3), \mathcal{T}_{pre}
 520 corresponds to the time complexity of Line 7-9, which is about $2^{\zeta+24+16+40} =$
 521 2^{120} 1-round AES. Therefore, the first part of Eq. (3) dominates the overall
 522 complexity, which is about $2^{128-8 \cdot \min\{\text{DoF}^+, \text{DoF}^-, \text{DoM}\}} = 2^{120}$ 5-round AES. The
 523 memory complexity to store U is about 2^{40} .

524 4.2 Practical-Memory Key-Recovery Attack on 4-full-round AES-128

525 The attack leverages the new representation of AES's key schedule by Leurent
 526 and Pernot [39]. They introduced a new basis $S^{(i)}$ to derive the round keys, *i.e.*,
 527 $RK^{(i)} = C_0 \cdot S^{(i)}$ as shown in Figure 6, where C_0 is a 16×16 binary matrix given
 528 Eq. (26) in Supplementary Material B.

529 The MitM path is shown in Figure 6. The starting state $S^{(2)}$, whose bytes
 530 are denoted by k_0 to k_{15} , contains $\lambda^+ = 1$ ■ byte and $\lambda^- = 6$ ■ bytes. The
 531 consumed DoFs of ■ and ■ are $\ell^+ = 0$ and $\ell^- = 5$ bytes, respectively. The
 532 $\ell^- = 5$ constraints (marked by ■/■) form a system of 5 nonlinear equations in
 533 Eq. (17) (deleting the boxed parts). Therefore, $\text{DoF}^+ = 1$, $\text{DoF}^- = 1$, and there
 534 is $\text{DoM} = 1$ matching byte in round 1. Using new TA, we get 3 free variables
 535 $S^{(2)}[7, 10, 13] = (k_7, k_{10}, k_{13})$ and 3 dependent variables $S^{(2)}[0, 1, 4] = (k_0, k_1, k_4)$.
 536 The matrices before and after the improved TA are shown in Eq. (16). The steps
 537 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)

```

1 for  $2^\zeta$  values of  $(e_1, e_2, e_3, e_4, e_5) \in \mathbb{F}_2^{40}$  do
2   for  $(e_6, e_7, e_8) \in \mathbb{F}_2^{24}$  do
3     for  $RK^{(0)}[14, 15] \in \mathbb{F}_2^{16}$  do
4        $U \leftarrow []$ 
5        $(\widehat{A}^{(1)}[4], \widehat{RK}^{(2)}[12], \widehat{RK}^{(2)}[13], \widehat{A}^{(1)}[3], \widehat{A}^{(2)}[4]) \leftarrow (e_1, e_2, e_3, e_4, e_5)$ 
6       for  $RK^{(0)}[6, 8, 10, 12, 13] \in \mathbb{F}_2^{40}$  do
7         Compute  $RK^{(0)}[7, 1, 2, 5, 9] = (k_7, k_1, k_2, k_5, k_9)$  by Eq. (13)-(f)
8         Compute  $\mathbf{u} = (\widehat{SR}^{(3)}[1], \widehat{SR}^{(3)}[4], \widehat{RK}^{(5)}[0], \widehat{A}^{(1)}[14]) \in \mathbb{F}_2^{32}$ 
9          $U[\mathbf{u}] \leftarrow RK^{(0)}[1, 2, 5 - 10, 12 - 15] \in \mathbb{F}_2^{8 \times 12}$ 
10        /* 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  $\mathbf{u}$ . */
11      end
12      for  $\mathbf{u} \in \mathbb{F}_2^{32}$  do
13         $L \leftarrow []$ 
14        for  $RK^{(0)}[0] = k_0 \in \mathbb{F}_2^8$  do
15          Compute  $RK^{(0)}[3, 4, 11] = (k_3, k_4, k_{11})$  by Eq. (15)
16          Compute the 1-byte matching point
17           $v = \text{SR}^{(2)}[4] \oplus e \cdot \text{MC}^{(2)}[4] \oplus b \cdot \text{MC}^{(2)}[5]$ 
18           $L[v] \leftarrow (k_0, k_3, k_4, k_{11})$ 
19        end
20        for values in  $U[\mathbf{u}]$  do
21          Compute the 1-byte matching point
22           $v' = e \cdot \text{MC}^{(2)}[4] \oplus b \cdot \text{MC}^{(2)}[5] \oplus d \cdot \text{MC}^{(2)}[6] \oplus 9 \cdot \text{MC}^{(2)}[7]$ 
23          for values in  $L[v']$  do
24            if  $E(\text{Key} = RK^{(0)}, P) = C$  then
25              return  $RK^{(0)}$ 
26            end
27          end
28        end
29      end
30    end
31  end
32 end

```

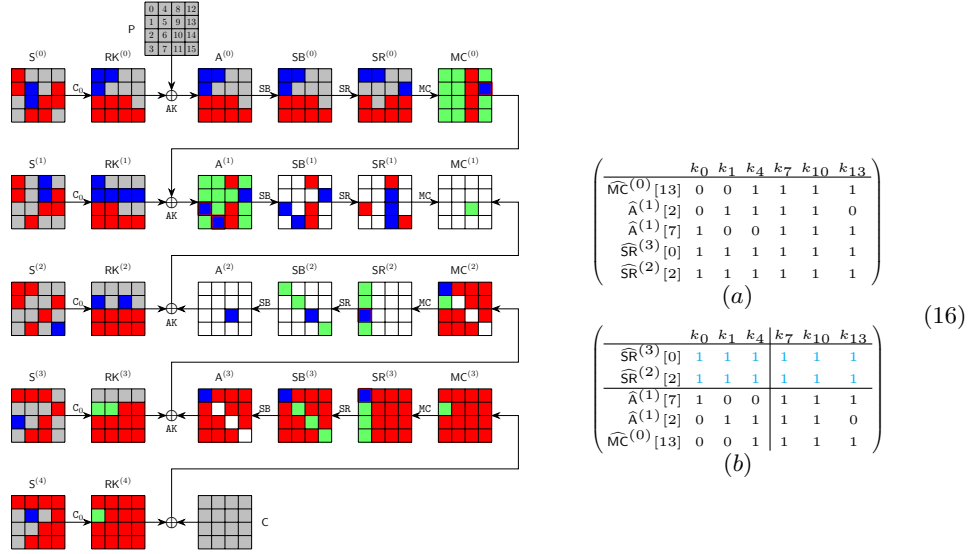


Fig. 6: 4-round attack on AES-128

538 $2^{128-8 \cdot \min\{\text{DoF}^+, \text{DoF}^-, \text{DoM}\}} = 2^{120}$. The memory is 2^{24} to store U .

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

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

- 541 1. Data collection: encrypt the plaintext $P = 0$ with key $\text{S}^{(2)} = 0$ to get the
 542 ciphertext C .
- 543 2. Given $(P = 0, C)$ and 12-byte key information to recover the other 4-byte key
 544 $\text{S}^{(2)}[7, 10, 13, 15]$. If the recovered $\text{S}^{(2)}[7, 10, 13, 15] = 0$, the attack succeeds.
 545 The 12-byte key information includes 9 bytes $\text{S}^{(2)}[2, 3, 5, 6, 8, 9, 11, 12, 14] = 0$ and 3-byte key relations of $(\widehat{\text{MC}}^{(0)}[13], \widehat{\text{A}}^{(1)}[2, 7]) = (e_1, e_2, e_3)$ on the 6 red
 546 key bytes of $\text{S}^{(2)}$, i.e., assign (e_1, e_2, e_3) to the last 3 expressions of Eq.
 547 (16)-(b). From (P, C) and $\text{S}^{(2)} = 0$, pre-compute the 3-byte $(e_1, e_2, e_3) =$
 548 $(0x75, 0x00, 0xc6)$.
 549 In our experiment, $(P = 0, C)$ and 12-byte key information $(\text{S}^{(2)}[2, 3, 5, 6,$
 550 $8, 9, 11, 12, 14] = 0, (e_1, e_2, e_3) = (0x75, 0x00, 0xc6))$ are given, the goal is

552 to recover $S^{(2)}[7, 10, 13, 15] = 0$. In brute-force search, the time will be 2^{32} .
 553 With the given information, we actually conduct the experiment from Line
 554 3 to Line 20 according to Algorithm 5. The time is 2^{24} with 2^{24} memory⁶.

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

557 <https://github.com/boxindex/Triangulation-MitM>

558 We implemented the experiment on a computer with an i9-13900KF CPU
 559 and 32GB of memory. It takes about 200 seconds, while the brute force needs
 560 2^{32} evaluations of 4-round AES-128, which takes about 3200 seconds in the same
 561 platform using the same code for AES.

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

```

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

562 4.3 New Attack on 4-full-round AES-128 with 2^{112} Complexity

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

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

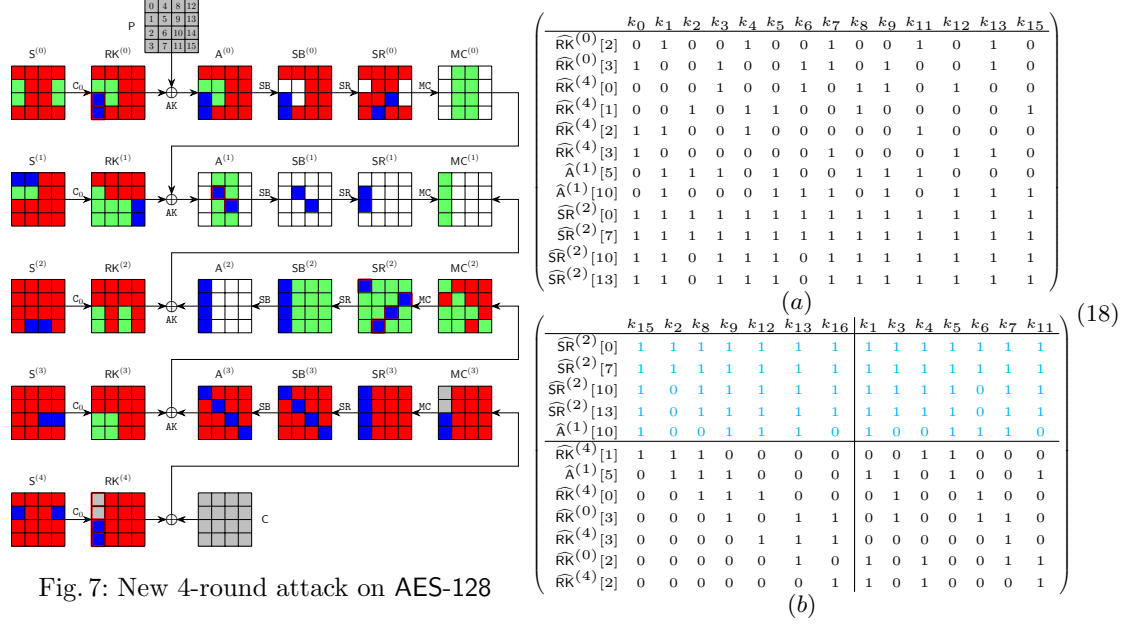


Fig. 7: New 4-round attack on AES-128

565 and \blacksquare are $\ell^+ = 0$ and $\ell^- = 12$ bytes, respectively. The $\ell^- = 12$ constraints
 566 (marked by $\blacksquare/\blacksquare$ in Figure 7) form a system of 12 equations in Eq. (19). There-
 567 fore, $\text{DoF}^+ = 2$, $\text{DoF}^- = 2$, and $\text{DoM} = 2$. Using new TA, we get 7 free variables
 568 $S^{(3)}[1, 3, 4, 5, 6, 7, 11]$ and 7 dependent variables $S^{(3)}[0, 2, 8, 9, 12, 13, 15]$. The mat-
 569 rices before and after TA are given in Eq. (18). The steps for the MitM attack are
 570 given in Algorithm 6. The time complexity is about $2^{128-8\cdot\min\{\text{DoF}^+, \text{DoF}^-, \text{DoM}\}} =$
 571 2^{112} . The memory is 2^{56} to store U .

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

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

```

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

```

572 4.4 Two-Plaintext Key-Recovery Attack on 6-round AES-192

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

577 The 6-round MitM path is given in Figure 8, where the starting state $S^{(3)}$,
 578 whose bytes are denoted as k_0, k_1, \dots, k_{23} , contains $\lambda^+ = 2$ blue bytes and $\lambda^- = 21$
 579 red bytes. The consumed degrees of freedom (DoFs) of blue and red are $\ell^+ = 0$ and
 580 $\ell^- = 19$ bytes, respectively. Therefore, $\text{DoF}^+ = 2$, $\text{DoF}^- = 2$, and there are
 581 $\text{DoM} = 2$ matching bytes in round 2. We get 19 equations as Eq. (20) (deleting
 582 the boxed parts) on the red bytes of $S^{(3)}$ for the 19 consumed DoFs of red marked
 583 by red/blue in Figure 8.

584 Using the new TA, we get 9 free variables $S^{(3)}[0, 3, 4, 13, 14, 15, 16, 21, 23]$
 585 and 12 dependent variables $S^{(3)}[1, 2, 5, 6, 7, 11, 12, 17, 18, 19, 20, 22]$. The matrices
 586 after TA are given in Eq. (21). The 6-round MitM attack is given in Al-
 587 gorithm 10 in Supplementary Material C. The total time complexity is about
 588 $2^{192-8 \cdot \min\{\text{DoF}^+, \text{DoF}^-, \text{DoM}\}} = 2^{176}$. The memory is 2^{72} to store U .

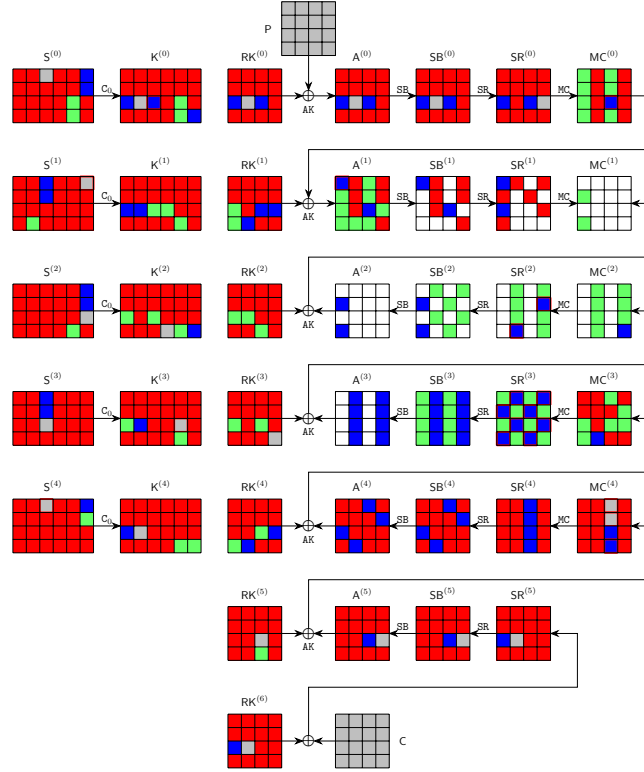


Fig. 8: The 6-round attack on AES-192

597 $S^{(1)}[0, 6, 14, 18, 20, 22, 27, 28, 30]$ and 14 dependent variables $S^{(1)}[1, 2, 3, 4, 5, 7, 9,$
598 $10, 11, 16, 17, 21, 29, 31]$. The matrices after TA is given in Eq. (22). The steps for
599 the 7-round MitM attack are given in Algorithm 11 in Supplementary Material
600 D. The total time complexity is about $2^{256-8 \cdot \min\{\text{DoF}^+, \text{DoF}^-, \text{DoM}\}} = 2^{248}$. The
601 memory is 2^{72} to store U .

$$\begin{pmatrix}
& 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} \\
\text{SR}^{(4)}[1] & 1 \\
\text{SR}^{(4)}[4] & 1 \\
\text{SR}^{(4)}[11] & 1 \\
\text{SR}^{(4)}[14] & 1 \\
\text{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 \\
\text{MC}^{(1)}[14] & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
\hat{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 \\
\text{MC}^{(0)}[14] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\
\text{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 \\
\text{MC}^{(5)}[11] & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\text{MC}^{(5)}[8] & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hat{A}^{(2)}[3] & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
\hat{K}^{(0)}[1] & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\hat{A}^{(1)}[2] & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\
\hat{A}^{(1)}[13] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\
\hat{K}^{(3)}[18] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
\hat{A}^{(1)}[10] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
\hat{K}^{(3)}[10] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
\hat{A}^{(1)}[6] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\
\hat{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 \\
\hat{A}^{(1)}[12] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
\text{MC}^{(0)}[15] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0
\end{pmatrix} \quad (22)$$

602 4.6 Improved Preimage Attack on 10-round AES-256-DM

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

611 We reuse the 10-round MitM characteristic in [26] as shown in Figure 20 in
612 Supplementary Material E. In the MitM path, the starting states are $A^{(4)}$ and
613 $S^{(3)}$. The 1-byte matching occurs in the MC operation in round 8. In $S^{(3)}$, there
614 are 19 ■ cells, 1 ■ cell and 12 ■ cells. In $A^{(4)}$, there are 8 ■ cells, 8 ■ cells. Hence,
615 the total initial DoFs are $\lambda^+ = 19 + 8 = 27$ cells for ■ cells, and $\lambda^- = 1 + 8 = 9$
616 for ■ cells. The consumed DoFs of ■ and DoFs of ■ are $\ell^+ = 26$ and $\ell^- = 8$
617 bytes, respectively. Therefore, $\text{DoF}^+ = 1$, $\text{DoF}^- = 1$, and there is $\text{DoM} = 1$
618 matching byte.

619 We obtain the 18 equations on ■ bytes of $S^{(3)}$ for the consumed DoFs of blue
620 bytes in $S^{(1)}[13, 14]$, $S^{(2)}[12, 13, 14, 15]$, $S^{(4)}[1]$, $K^{(2)}[0, 3, 4, 5, 9]$, $\text{SR}^{(1)}[3, 6, 9, 12]$
621 and $\text{SR}^{(9)}[6, 12]$, where the expressions of $\text{SR}^{(1)}[3, 6, 9, 12]$ are obtained by $\text{MC}^{-1}(\text{RK}^{(1)})$,
622 and assign $a_i (0 \leq i < 18)$ to them. Using the new TA, we get 4 free variables
623 $S^{(3)}[8, 16, 18, 24]$ and 15 dependent variables $S^{(3)}[0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 17, 27,$

28, 31]. The matrices before and after TA are shown in Eq. (31) in Supplementary Material E, where the bytes of $S^{(3)}$ are denoted as k_0, k_1, \dots, 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 \blacksquare in $S^{(3)}$ as 0 in Line 1, and 15 a_i as 0 in Line 2. Traverse 4 free variables $\blacksquare S^{(3)}[8, 16, 18, 24]$ 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 \blacksquare bytes in $MC^{(3)}, MC^{(4)}$ except $MC^{(4)}[0]$ to 0 and traverse 3-byte $A^{(5)}[2, 6]$ and $MC^{(4)}[0]$, 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/boxindex/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.*, $\text{Rijndael}(X) \oplus X = P \oplus C$. 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 $k = X \oplus P$.

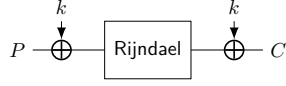


Fig. 9: Rijndael-EM

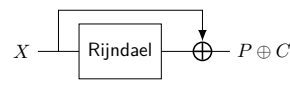


Fig. 10: Equivalent Form

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

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

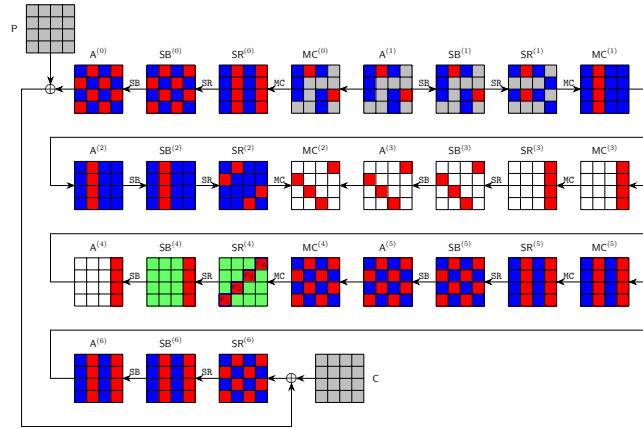


Fig. 11: The 7-round attack on Rijndael-EM-128

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

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

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

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

```

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

```

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

```

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

```

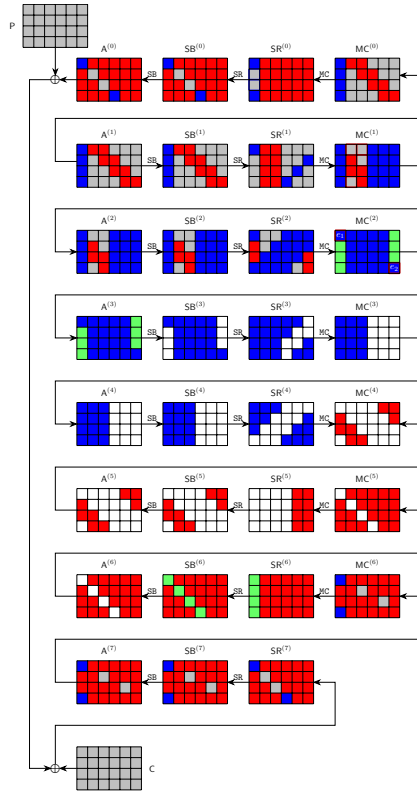


Fig. 12: 8-round Rijndael-EM-192

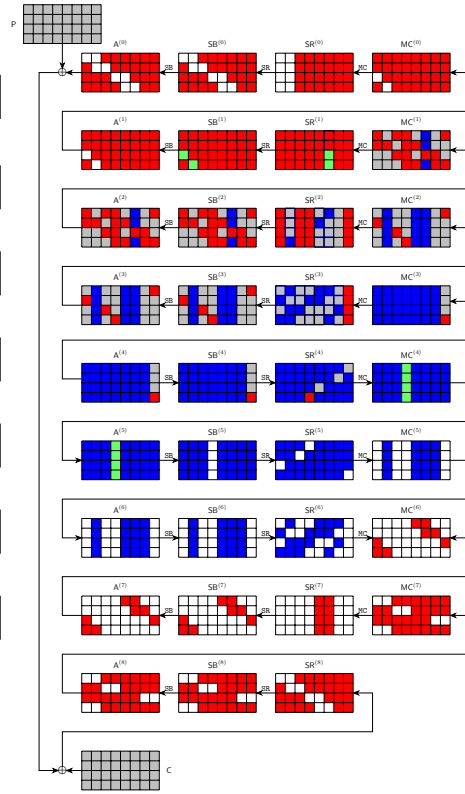


Fig. 13: 9-round Rijndael-EM-256

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

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

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

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

```

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

- 745 11. Charles Bouillaguet, Patrick Derbez, and Pierre-Alain Fouque. Automatic search
746 of attacks on round-reduced AES and applications. In Phillip Rogaway, editor,
747 *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference*,
748 *Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture*
749 *Notes in Computer Science*, pages 169–187. Springer, 2011.
- 750 12. Christina Boura, Nicolas David, Patrick Derbez, Gregor Leander, and María Naya-
751 Plasencia. Differential meet-in-the-middle cryptanalysis. In Helena Handschuh and
752 Anna Lysyanskaya, editors, *CRYPTO 2023, Proceedings, Part III*, volume 14083
753 of *Lecture Notes in Computer Science*, pages 240–272. Springer, 2023.
- 754 13. Anne Canteaut, María Naya-Plasencia, and Bastien Vayssière. Sieve-in-the-middle:
755 Improved MITM attacks. In *CRYPTO 2013, Proceedings, Part I*, pages 222–240.
- 756 14. Shiyao Chen, Jian Guo, Eik List, Danping Shi, and Tianyu Zhang. Diving deep
757 into the preimage security of aes-like hashing. In Marc Joye and Gregor Leander,
758 editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International*
759 *Conference on the Theory and Applications of Cryptographic Techniques, Zurich,*
760 *Switzerland, May 26-30, 2024, Proceedings, Part I*, volume 14651 of *Lecture Notes*
761 *in Computer Science*, pages 398–426. Springer, 2024.
- 762 15. Joan Daemen, Seth Hoffert, Michaël Peeters, Gilles Van Assche, and Ronny Van
763 Keer. Xoodoo, a lightweight cryptographic scheme. *IACR Trans. Symmetric*
764 *Cryptol.*, 2020(S1):60–87, 2020.
- 765 16. Joan Daemen, Pedro Maat Costa Massolino, Alireza Mehrdad, and Yann
766 Rotella. The subterranean 2.0 cipher suite. *IACR Trans. Symmetric Cryptol.*,
767 2020(S1):262–294, 2020.
- 768 17. Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced*
769 *Encryption Standard*. Information Security and Cryptography. Springer, 2002.
- 770 18. Mathieu Degré, Patrick Derbez, Lucie Lahaye, and André Schrottenloher. New
771 models for the cryptanalysis of ASCON. *IACR Cryptol. ePrint Arch.*, page 298,
772 2024.
- 773 19. Patrick Derbez. *Meet-in-the-middle attacks on AES*. PhD thesis, Ecole Normale
774 Supérieure de Paris-ENS Paris, 2013.
- 775 20. Patrick Derbez and Pierre-Alain Fouque. Automatic search of meet-in-the-middle
776 and impossible differential attacks. In *CRYPTO 2016, Proceedings, Part II*, volume
777 9815, pages 157–184. Springer, 2016.
- 778 21. Whitfield Diffie and Martin E. Hellman. Special feature exhaustive cryptanalysis
779 of the NBS data encryption standard. *Computer*, 10(6):74–84, 1977.
- 780 22. Itai Dinur. Cryptanalytic applications of the polynomial method for solving mul-
781 ti-variate equation systems over GF(2). In *EUROCRYPT 2021, Proceedings, Part*
782 *I*, volume 12696, pages 374–403. Springer, 2021.
- 783 23. Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Efficient dissection
784 of composite problems, with applications to cryptanalysis, knapsacks, and combi-
785 natorial search problems. In *CRYPTO 2012*, volume 7417, pages 719–740.
- 786 24. Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. As-
787 con v1.2: Lightweight authenticated encryption and hashing. *J. Cryptol.*, 34(3):33,
788 2021.
- 789 25. Xiaoyang Dong, Jian Guo, Shun Li, and Phuong Pham. Triangulating rebound
790 attack on aes-like hashing. In Yevgeniy Dodis and Thomas Shrimpton, editors,
791 *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptol-*
792 *ogy Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022,*
793 *Proceedings, Part I*, volume 13507 of *Lecture Notes in Computer Science*, pages
794 94–124. Springer, 2022.

- 795 26. Xiaoyang Dong, Jialiang Hua, Siwei Sun, Zheng Li, Xiaoyun Wang, and Lei Hu.
796 Meet-in-the-middle attacks revisited: Key-recovery, collision, and preimage attacks.
797 In *CRYPTO 2021, Proceedings, Part III*, volume 12827, pages 278–308. Springer.
- 798 27. Xiaoyang Dong, Boxin Zhao, Lingyue Qin, Qingliang Hou, Shun Zhang, and Xi-
799 aoyun Wang. Generic mitm attack frameworks on sponge constructions. In Leonid
800 Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 -*
801 *44th Annual International Cryptology Conference, Santa Barbara, CA, USA, Au-*
802 *gust 18-22, 2024, Proceedings, Part IV*, volume 14923 of *Lecture Notes in Computer*
803 *Science*, pages 3–37. Springer, 2024.
- 804 28. Orr Dunkelman, Nathan Keller, Eyal Ronen, and Adi Shamir. The retracing
805 boomerang attack, with application to reduced-round AES. *J. Cryptol.*, 37(3):32,
806 2024.
- 807 29. Orr Dunkelman, Gautham Sekar, and Bart Preneel. Improved meet-in-the-middle
808 attacks on reduced-round DES. In *INDOCRYPT 2007, Proceedings*, volume 4859,
809 pages 86–100. Springer, 2007.
- 810 30. Thomas Espitau, Pierre-Alain Fouque, and Pierre Karpman. Higher-order differ-
811 ential meet-in-the-middle preimage attacks on SHA-1 and BLAKE. In *CRYPTO*
812 *2015, Proceedings, Part I*, volume 9215, pages 683–701. Springer, 2015.
- 813 31. Thomas Fuhr and Brice Minaud. Match box meet-in-the-middle attack against
814 KATAN. In *FSE 2014*, pages 61–81, 2014.
- 815 32. Jian Guo, San Ling, Christian Rechberger, and Huaxiong Wang. Advanced meet-
816 in-the-middle preimage attacks: First results on full Tiger, and improved results on
817 MD4 and SHA-2. In *ASIACRYPT 2010, Proceedings*, volume 6477, pages 56–75.
- 818 33. Hosein Hadipour and Maria Eichlseder. Autoguess: A tool for finding guess-and-
819 determine attacks and key bridges. In Giuseppe Ateniese and Daniele Venturi,
820 editors, *Applied Cryptography and Network Security - 20th International Confer-*
821 *ence, ACNS 2022, Rome, Italy, June 20-23, 2022, Proceedings*, volume 13269 of
822 *Lecture Notes in Computer Science*, pages 230–250. Springer, 2022.
- 823 34. Dmitry Khovratovich, Alex Biryukov, and Ivica Nikolic. Speeding up collision
824 search for byte-oriented hash functions. In Marc Fischlin, editor, *Topics in Crypt-*
825 *ology - CT-RSA 2009, The Cryptographers’ Track at the RSA Conference 2009,*
826 *San Francisco, CA, USA, April 20-24, 2009. Proceedings*, volume 5473 of *Lecture*
827 *Notes in Computer Science*, pages 164–181. Springer, 2009.
- 828 35. Dmitry Khovratovich, Gaëtan Leurent, and Christian Rechberger. Narrow-
829 bicliques: Cryptanalysis of full IDEA. In David Pointcheval and Thomas Johans-
830 son, editors, *EUROCRYPT 2012, Proceedings*, volume 7237, pages 392–410.
- 831 36. Simon Knellwolf and Dmitry Khovratovich. New preimage attacks against reduced
832 SHA-1. In *CRYPTO 2012, Proceedings*, volume 7417, pages 367–383.
- 833 37. Brian A. LaMacchia and Andrew M. Odlyzko. Solving large sparse linear systems
834 over finite fields. In Alfred Menezes and Scott A. Vanstone, editors, *Advances*
835 *in Cryptology - CRYPTO ’90, 10th Annual International Cryptology Conference,*
836 *Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, volume 537 of
837 *Lecture Notes in Computer Science*, pages 109–133. Springer, 1990.
- 838 38. Charlotte Lefevre and Bart Mennink. Tight preimage resistance of the sponge
839 construction. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in*
840 *Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference,*
841 *CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part*
842 *IV*, volume 13510 of *Lecture Notes in Computer Science*, pages 185–204. Springer,
843 2022.
- 844 39. Gaëtan Leurent and Clara Pernot. New Representations of the AES Key Schedule.
845 Cryptology ePrint Archive, Report 2020/1253, 2020.

- 846 40. Gaëtan Leurent and Clara Pernot. New representations of the AES key schedule.
847 In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology*
848 *- EUROCRYPT 2021 - 40th Annual International Conference on the Theory and*
849 *Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021,*
850 *Proceedings, Part I*, volume 12696 of *Lecture Notes in Computer Science*, pages
851 54–84. Springer, 2021.
- 852 41. Fukang Liu, Takanori Isobe, and Willi Meier. Exploiting weak diffusion of gimli:
853 Improved distinguishers and preimage attacks. *IACR Trans. Symmetric Cryptol.*,
854 2021(1):185–216, 2021.
- 855 42. Fukang Liu, Takanori Isobe, Willi Meier, and Zhonghao Yang. Algebraic attacks on
856 round-reduced Keccak. In *ACISP 2021, Proceedings*, volume 13083, pages 91–110.
- 857 43. Florian Mendel, Christian Rechberger, Martin Schl  ffer, and S  ren S. Thomsen.
858 The rebound attack: Cryptanalysis of reduced whirlpool and gr  stl. In *FSE 2009,*
859 *2009*, pages 260–276, 2009.
- 860 44. Pawel Morawiecki, Josef Pieprzyk, and Marian Srebrny. Rotational cryptanalysis
861 of round-reduced Keccak. In *FSE 2013*, volume 8424, pages 241–262.
- 862 45. Lingyue Qin, Jialiang Hua, Xiaoyang Dong, Hailun Yan, and Xiaoyun Wang. Meet-
863 in-the-middle preimage attacks on sponge-based hashing. In Carmit Hazay and
864 Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd An-*
865 *annual International Conference on the Theory and Applications of Cryptographic*
866 *Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part IV*, volume 14007
867 of *Lecture Notes in Computer Science*, pages 158–188. Springer, 2023.
- 868 46. Lingyue Qin, Jialiang Hua, Xiaoyang Dong, Hailun Yan, and Xiaoyun Wang. Meet-
869 in-the-middle preimage attacks on sponge-based hashing. In *EUROCRYPT 2023,*
870 *Proceedings, Part IV*, volume 14007, pages 158–188. Springer, 2023.
- 871 47. Sondre R  njom, Navid Ghaedi Bardeh, and Tor Hellese  th. Yoyo tricks with AES.
872 In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASI-*
873 *ACRYPT 2017 - 23rd International Conference on the Theory and Applications*
874 *of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017,*
875 *Proceedings, Part I*, volume 10624 of *Lecture Notes in Computer Science*, pages
876 217–243. Springer, 2017.
- 877 48. Yu Sasaki. Integer linear programming for three-subset meet-in-the-middle attacks:
878 Application to GIFT. In *IWSEC 2018*, volume 11049, pages 227–243.
- 879 49. Yu Sasaki. Meet-in-the-middle preimage attacks on AES hashing modes and an
880 application to whirlpool. In Antoine Joux, editor, *Fast Software Encryption -*
881 *18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011,*
882 *Revised Selected Papers*, volume 6733 of *Lecture Notes in Computer Science*, pages
883 378–396. Springer, 2011.
- 884 50. Yu Sasaki and Kazumaro Aoki. Finding preimages in full MD5 faster than exhaus-
885 tive search. In *EUROCRYPT 2009, Proceedings*, volume 5479, pages 134–152.
- 886 51. Yu Sasaki, Lei Wang, Shuang Wu, and Wenling Wu. Investigating fundamental
887 security requirements on whirlpool: Improved preimage and collision attacks. In
888 Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology - ASIACRYPT*
889 *2012 - 18th International Conference on the Theory and Application of Cryptology*
890 *and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, vol-
891 *ume 7658 of Lecture Notes in Computer Science*, pages 562–579. Springer, 2012.
- 892 52. Andr   Schrottenloher and Marc Stevens. Simplified MITM modeling for permuta-
893 tions: New (quantum) attacks. In *CRYPTO 2022, Proceedings, Part III*, volume
894 13509, pages 717–747. Springer, 2022.

- 895 53. André Schrottenloher and Marc Stevens. Simplified modeling of MITM attacks
896 for block ciphers: new (quantum) attacks. *IACR Cryptol. ePrint Arch.*, page 816,
897 2023.
- 898 54. Tyge Tiessen. Polytopic cryptanalysis. In Marc Fischlin and Jean-Sébastien Coron,
899 editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International*
900 *Conference on the Theory and Applications of Cryptographic Techniques, Vienna,*
901 *Austria, May 8-12, 2016, Proceedings, Part I*, volume 9665 of *Lecture Notes in*
902 *Computer Science*, pages 214–239. Springer, 2016.
- 903 55. Michael Tunstall. Improved "partial sums"-based square attack on AES. In
904 Pierangela Samarati, Wenjing Lou, and Jianying Zhou, editors, *SECRYPT 2012 -*
905 *Proceedings of the International Conference on Security and Cryptography, Rome,*
906 *Italy, 24-27 July, 2012, SECRYPT is part of ICETE - The International Joint*
907 *Conference on e-Business and Telecommunications*, pages 25–34. SciTePress, 2012.

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

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

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

$$\begin{cases} c_0 = 9 \cdot \text{MC}^{(3)}[0] \oplus e \cdot \text{MC}^{(3)}[1] \oplus b \cdot \text{MC}^{(3)}[2] \oplus d \cdot \text{MC}^{(3)}[3] \\ c_1 = d \cdot \text{MC}^{(3)}[0] \oplus 9 \cdot \text{MC}^{(3)}[1] \oplus e \cdot \text{MC}^{(3)}[2] \oplus b \cdot \text{MC}^{(3)}[3] \\ c_2 = b \cdot \text{MC}^{(3)}[0] \oplus d \cdot \text{MC}^{(3)}[1] \oplus 9 \cdot \text{MC}^{(3)}[2] \oplus e \cdot \text{MC}^{(3)}[3] \end{cases} \quad (23)$$

928 There are 2^8 possibilities for the values of $\text{MC}^{(3)}[0-3]$ given the constraints.
 929 Similarly, the backward chunk imposes 8 additional constraints on the bytes
 930 $\text{MC}^{(4)}[0, 2, 5, 7, 8, 10, 13, 15]$.

931 **Partial Matching Through MixColumns** The MitM approach relies on
 932 the MixColumns operation to validate matches. Since each column in $\text{SR}^{(1)}$ and
 933 $\text{MC}^{(1)}$ contains five known bytes, one match is derived per column.

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

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

937 Forward and Backward Computations

- 938 1. **Forward Computation:** The neutral bytes from $\text{MC}^{(3)}$ propagate forward
 939 to $\text{SR}^{(1)}$. By traversing all 2^8 possibilities of $\text{MC}^{(3)}[0-3]$, the resulting values
 940 of $\text{SR}^{(1)}$ are stored in table L_1 , by the value of $\text{SR}^{(1)}$ as the left part of Eq. (24)
 941 (i.e. $d \cdot \text{SR}^{(1)}[0] \oplus e \cdot \text{SR}^{(1)}[2]$).
- 942 2. **Backward Computation:** Similarly, the red neutral bytes from $\text{SR}^{(4)}$ prop-
 943 agate backward to $\text{MC}^{(1)}$. These values are stored in table L_2 , indexed by
 944 $\text{MC}^{(1)}$ as the right part of Eq. (24).
- 945 3. **Partial Matching:** The two tables L_1 and L_2 are compared to identify
 946 32-bit partial matches based on the indices, allowing the preimage to be
 947 recovered efficiently.

948 **B New Representation of AES's Key Schedule**

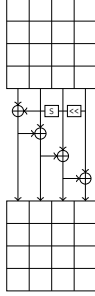


Fig. 15: AES-128

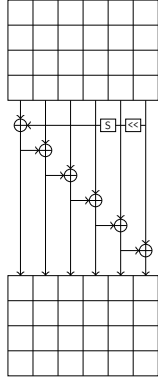


Fig. 16: AES-192

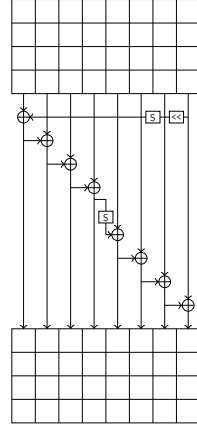


Fig. 17: AES-256

$$MC = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \quad MC^{-1} = \begin{bmatrix} e & b & d & 9 \\ 9 & e & b & d \\ d & 9 & e & b \\ b & d & 9 & e \end{bmatrix} \quad (25)$$

949 At EUROCRYPT 2021, Leurent and Pernot introduced the new representa-
 950 tion of AES's key schedule [40]. For AES-128, denote the 16-byte round key by
 951 $RK^{(i)} = (a_0, a_1, \dots, a_{15})$. Leurent *et al.* derived a new basis $S^{(i)} = (s_0, s_1, \dots, s_{15})$.
 952 The relations between $RK^{(i)}$ and $S^{(i)}$ are $S^{(i)} = A \cdot (RK^{(i)})^\dagger$ and $RK^{(i)} =$
 953 $C_0 \cdot (S^{(i)})^\dagger$, where A is a matrix in Eq. (26) and $C_0 = A^{-1}$. Denote the state of
 954 next round by $S^{(i)} = (s'_0, s'_1, \dots, s'_{15})$, the update process is illustrated in Figure
 955 18, and the details listed in Eq. (27). For more details of AES-192/256, please
 956 refer to [40].

$$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 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 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 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 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 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 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 & 0 & 0 & 0 & 0 & 0 & 1 & 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 & 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 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \quad C_0 = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 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 & 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 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 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 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 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 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 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 & 0 & 0 & 0 & 0 & 0 & 1 \\ 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 & 0 & 0 & 0 & 0 \end{pmatrix} \quad (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} \quad (26)$$

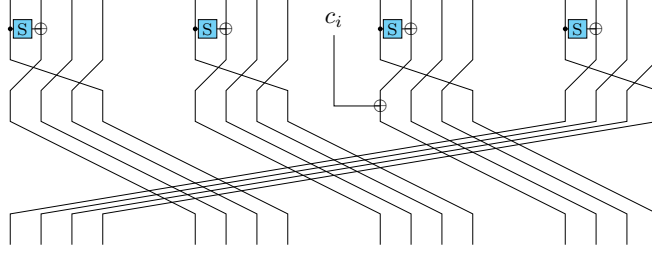


Fig. 18: New Representations of AES-128's key schedule [40]

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

957 C Key-Recovery Attack on 6-round AES-192 with Two 958 Plaintext-Ciphertext Pairs

959 The 6-round MitM characteristic is shown in Figure 8. Denote the bytes of the
960 starting state $S^{(3)}$ by k_0, k_1, \dots, k_{23} . The 19 expressions on the red bytes of
961 $S^{(3)}$ for the 19 consumed DoFs of \blacksquare are shown as Eq. (20) (deleting the boxed
962 parts). The matrices before and after TA are shown in Eq. (28). The steps for
963 the 6-round MitM attack are given in Algorithm 10.

$$\begin{pmatrix}
\begin{array}{c}
\widehat{S}^{(1)}[20] \\
\widehat{S}^{(4)}[8] \\
\widehat{R}^{(0)}[10] \\
\widehat{R}^{(3)}[18] \\
\widehat{MC}^{(0)}[10] \\
\widehat{A}^{(1)}[0] \\
\widehat{MC}^{(4)}[8] \\
\widehat{MC}^{(4)}[9] \\
\widehat{MC}^{(4)}[11] \\
\widehat{SR}^{(3)}[1] \\
\widehat{SR}^{(3)}[3] \\
\widehat{SR}^{(3)}[4] \\
\widehat{SR}^{(3)}[6] \\
\widehat{SR}^{(3)}[9] \\
\widehat{SR}^{(3)}[11] \\
\widehat{SR}^{(3)}[12] \\
\widehat{SR}^{(3)}[14] \\
\widehat{SR}^{(2)}[7] \\
\widehat{SR}^{(2)}[13]
\end{array}
&
\begin{array}{c}
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} \\
0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \ 0 \\
0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 0 \ 0 \\
0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 0 \ 0 \ 0 \\
0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \\
1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 0 \ 0 \ 1 \ 0 \ 1 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \\
1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \\
1 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \\
0 \ 0 \ 0 \ 1 \ 0 \ 1 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \\
1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \\
1 \ 1 \\
1 \ 1 \\
1 \ 1 \\
1 \ 1 \\
1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \\
1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \\
1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \\
0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 1 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \\
0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0
\end{array}
\end{pmatrix}
\begin{pmatrix}
\begin{array}{c}
\widehat{SR}^{(3)}[1] \\
\widehat{SR}^{(3)}[3] \\
\widehat{SR}^{(3)}[4] \\
\widehat{SR}^{(3)}[6] \\
\widehat{SR}^{(3)}[9] \\
\widehat{SR}^{(3)}[11] \\
\widehat{SR}^{(3)}[12] \\
\widehat{R}^{(3)}[18] \\
\widehat{SR}^{(3)}[14] \\
\widehat{S}^{(1)}[20] \\
\widehat{A}^{(1)}[0] \\
\widehat{MC}^{(4)}[9] \\
\widehat{R}^{(0)}[10] \\
\widehat{MC}^{(4)}[8] \\
\widehat{SR}^{(2)}[13] \\
\widehat{MC}^{(0)}[10] \\
\widehat{SR}^{(2)}[7] \\
\widehat{MC}^{(4)}[11] \\
\widehat{S}^{(4)}[8]
\end{array}
&
\begin{array}{c}
k_{18} \ k_{17} \ k_{22} \ k_2 \ k_6 \ k_7 \ k_{12} \ k_{19} \ k_1 \ k_5 \ k_{11} \ k_{20} \\
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 \ 1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \\
0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \\
0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \\
1 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \\
0 \ 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \\
0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \\
0 \ 0 \ 0 \ 1 \ 1 \ 1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 0 \\
0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \\
0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 1 \\
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 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \ 0 \\
0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \\
0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \\
0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1
\end{array}
&
\begin{array}{c}
k_0 \ k_3 \ k_4 \ k_{13} \ k_{14} \ k_{15} \ k_{16} \ k_{21} \ k_{23} \\
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 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \\
1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \\
1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \\
1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \\
0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \\
0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \\
1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 0 \\
1 \ 0 \ 0 \ 1 \ 1 \ 1 \ 1 \ 0 \ 0 \\
0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \\
1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 0 \ 0 \ 0 \\
0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \\
1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \\
1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \\
0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1
\end{array}
\end{pmatrix}
\quad (28)$$

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 $S^{(1)}$ are denoted as k_0, k_1, \dots, k_{31} . The 22 expressions on the red bytes of $S^{(1)}$ for the 22 consumed DoFs of \blacksquare are shown as Eq. (29) (deleting the boxed parts). Using TA, we can get 9 free variables $S^{(1)}[0, 6, 14, 18, 20, 22, 27, 28, 30]$ and 14 dependent variables $S^{(1)}[1, 2, 3, 4, 5, 7, 9, 10, 11, 16, 17, 21, 29, 31]$, 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 AES-192 with 2 (P, C)

```

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

```

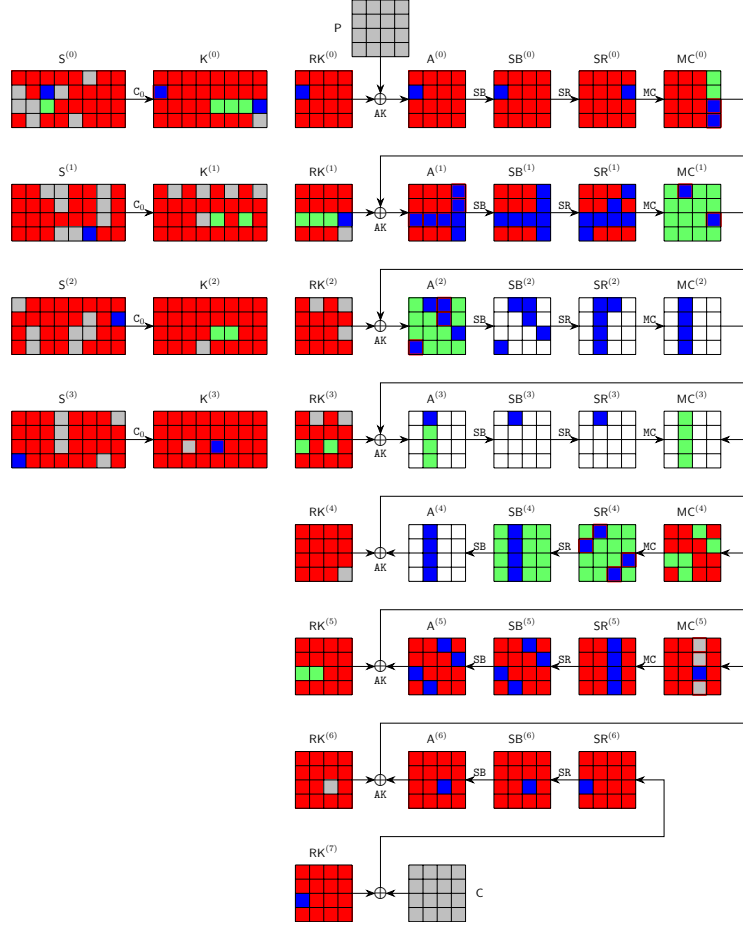


Fig. 19: The 7-round attack on AES-256

$$\begin{cases}
\widehat{K}^{(0)}[1] = k_{10} \oplus k_{16} \oplus k_{30} \oplus k_4 \oplus S(k_{23}) \\
\widehat{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{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 \oplus k_{23} \\
\widehat{MC}^{(0)}[14] = S(k_6) \oplus 2 \cdot S(k_{22} \oplus k_2) \oplus 3 \cdot S(k_{14} \oplus k_{20}) \oplus SR^{(0)}[13] \\
\widehat{MC}^{(0)}[15] = 3 \cdot S(k_6) \oplus S(k_{22} \oplus k_2) \oplus 2 \cdot S(k_{14} \oplus k_{20}) \oplus SR^{(0)}[13] \\
\widehat{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}) \\
\quad \oplus k_9 \oplus k_{29} \oplus k_3 \oplus S(k_8) \oplus k_{23} \\
\widehat{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)) \\
\quad \oplus k_3 \oplus k_{23} \\
\widehat{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}) \\
\quad \oplus k_{29} \oplus k_{23} \\
\widehat{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 \oplus 3 \cdot SR^{(0)}[13] \\
\widehat{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} \oplus 2 \cdot SR^{(0)}[13] \\
\widehat{MC}^{(1)}[4] = 2 \cdot SR^{(1)}[4] \oplus 3 \cdot SR^{(1)}[5] \oplus SR^{(1)}[7] \oplus SR^{(1)}[6] \\
\widehat{MC}^{(1)}[14] = SR^{(1)}[13] \oplus 3 \cdot SR^{(1)}[6] \oplus SR^{(1)}[12] \oplus 2 \cdot SR^{(1)}[14] \\
\widehat{A}^{(2)}[3] = 3 \cdot SR^{(1)}[0] \oplus SR^{(1)}[1] \oplus RK^{(2)}[3] \oplus SR^{(1)}[2] \oplus 2 \cdot SR^{(1)}[3] \\
\widehat{A}^{(2)}[8] = 2 \cdot SR^{(1)}[8] \oplus SR^{(1)}[11] \oplus RK^{(2)}[8] \oplus 3 \cdot SR^{(1)}[9] \oplus SR^{(1)}[10] \\
\widehat{A}^{(2)}[9] = SR^{(1)}[8] \oplus SR^{(1)}[11] \oplus RK^{(2)}[9] \oplus 2 \cdot SR^{(1)}[9] \oplus 3 \cdot SR^{(1)}[10] \\
\widehat{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} \\
\quad \oplus S(k_{17}) \oplus S(k_{11} \oplus S(k_{10} \oplus S(k_9))) \oplus k_{12} \\
\widehat{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))) \\
\quad \oplus k_{10} \oplus S(k_9) \\
\widehat{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}))) \\
\quad \oplus k_{26} \oplus S(k_{25}) \\
\widehat{SR}^{(4)}[1] = 9 \cdot MC^{(4)}[0] \oplus e \cdot MC^{(4)}[1] \oplus b \cdot (S^{(2)}[3] \oplus S^{(2)}[9] \oplus S^{(2)}[23]) \oplus d \cdot MC^{(4)}[3] \\
\quad \oplus b \cdot (S^{(2)}[29] \oplus A^{(5)}[2]) \\
\widehat{SR}^{(4)}[4] = e \cdot MC^{(4)}[4] \oplus b \cdot MC^{(4)}[5] \oplus d \cdot (S^{(2)}[9] \oplus A^{(5)}[6]) \oplus 9 \cdot RK^{(2)}[29] \oplus b \cdot (S^{(2)}[29] \oplus A^{(5)}[7]) \\
\widehat{SR}^{(4)}[11] = d \cdot RK^{(5)}[8] \oplus 9 \cdot MC^{(4)}[9] \oplus e \cdot MC^{(4)}[10] \oplus b \cdot MC^{(4)}[11] \oplus d \cdot A^{(5)}[8] \\
\widehat{SR}^{(4)}[14] = b \cdot MC^{(4)}[12] \oplus d \cdot RK^{(5)}[13] \oplus 9 \cdot MC^{(4)}[14] \oplus e \cdot MC^{(4)}[15] \oplus d \cdot A^{(5)}[13]
\end{cases}$$

(29)

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

```

1  for  $2^{\zeta}$  values of  $S^{(1)}[8, 12, 13, 15, 19, 24, 25, 26] \in \mathbb{F}_2^{64}$  do
2      for  $(e_1, e_2, \dots, e_{14}) \in \mathbb{F}_2^{112}$  do
3           $U \leftarrow []$ 
4          for  $S^{(1)}[0, 6, 14, 18, 20, 22, 27, 28, 30] \in \mathbb{F}_2^{72}$  do
5              Assign  $(e_1, e_2, \dots, e_{14})$  to the last 14 expressions in Eq. (30) (b)
6              Deduce  $S^{(1)}[1, 2, 3, 4, 5, 7, 9, 10, 11, 16, 17, 21, 29, 31]$ 
7              Compute 8-byte
                   $\mathbf{u} = (\widehat{SR}^{(4)}[1, 4, 11, 14], \widehat{MC}^{(1)}[4, 14], \widehat{A}^{(2)}[9], \widehat{MC}^{(0)}[14]) \in \mathbb{F}_2^{64}$ 
8               $U[\mathbf{u}] \leftarrow S^{(1)}[v_{\mathcal{R}}] /* v_{\mathcal{R}}$  represents all 23 red indexes */
9          end
10         for  $\mathbf{u} \in \mathbb{F}_2^{64}$  do
11              $L \leftarrow []$ 
12             for  $S^{(1)}[23] \in \mathbb{F}_2^8$  do
13                 Compute forward and backward to the 1-byte matching point  $v$ 
14                  $L[v] \leftarrow S^{(1)}[23]$ 
15             end
16             for values in  $U[\mathbf{u}]$  do
17                 Compute forward and backward to the 1-byte matching point  $v'$ 
18                 for values in  $L[v']$  do
19                     if  $E(Key = S^{(1)}, P_1) = C_1$  and  $E(Key = S^{(1)}, P_2) = C_2$  then
20                         return  $S^{(1)}$ 
21                     end
22                 end
23             end
24         end
25     end
26 end

```

981 We experiment with the 5-byte partial target preimage attack by fixing
 982 the 5 target bytes $T[0,1,2,6,12] = 0$. The 4 preimages are produced with
 983 $T[0,1,2,6,12] = 0$, which are listed in Table 3.

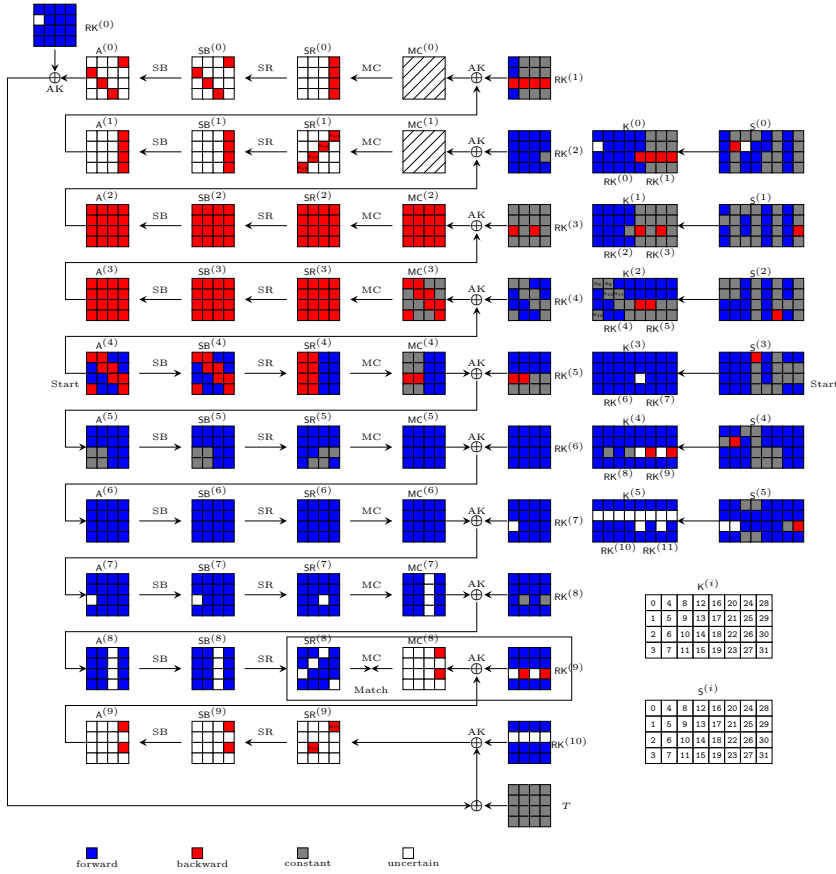


Fig. 20: The MitM attack on 10-round AES-256

$$\begin{pmatrix}
& 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 \\
a_3 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 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 \\
a_5 & 1 & 0 & 0 & 0 & 1 & 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 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
a_{11} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\
a_{12} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
a_{13} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 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 & 1 & 0 & 0 & 0 \\
a_{16} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
a_{17} & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\
a_{18} & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{pmatrix}
\begin{pmatrix}
& 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 & 1 & 1 & 1 \\
a_{16} & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
a_8 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
a_3 & 1 & 1 & 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 & 0 \\
a_4 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
a_{14} & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
a_{12} & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
a_6 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
a_5 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
a_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
a_7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
a_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
a_{13} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
a_{10} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\
a_{15} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
a_9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\
a_{11} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0
\end{pmatrix}
\tag{a}$$

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

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 $A_{\{x,y,z\}}^{(r)}$ 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

```

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

```

Rounds	$S^{(3)}$	$A^{(4)}$	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, 0xae, 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, 0xd3, 0x66	0xcd, 0x10, 0xe2, 0x6e, 0x40, 0xd8, 0xf8, 0xc3, 0x58, 0x19, 0x1a, 0x8e, 0x34, 0xd3, 0xb, 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, 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 AFS-256

996 round as follows:

$$\begin{aligned}
\theta : \theta_{\{x,y,z\}}^{(r)} &= A_{\{x,y,z\}}^{(r)} \oplus D_{\{x,z\}}^{(r)}, D_{\{x,z\}}^{(r)} = C_{\{x-1,z\}}^{(r)} \oplus C_{\{x+1,z-1\}}^{(r)}, C_{\{x,z\}}^{(r)} = \sum_{y'=0}^4 A_{\{x,y',z\}}^{(r)}, \\
\rho : \rho_{\{x,y,z\}}^{(r)} &= \theta_{\{x,y,z-\gamma[x,y]\}}^{(r)}, \\
\pi : \pi_{\{y,2x+3y,z\}}^{(r)} &= \rho_{\{x,y,z\}}^{(r)}, \\
\chi : \chi_{\{x,y,z\}}^{(r)} &= \pi_{\{x,y,z\}}^{(r)} \oplus (\pi_{\{x+1,y,z\}}^{(r)} \oplus 1) \cdot \pi_{\{x+2,y,z\}}^{(r)}, \\
\iota : A^{(r+1)} &= \chi^{(r)} \oplus RC_r, RC_r \text{ is round-dependent constant,}
\end{aligned} \tag{32}$$

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

999 This paper focuses on Keccak[1024] and Keccak[768], as well as SHA3-512
1000 and SHA3-384. For Keccak, the message is padded with “10*1”, which is a single
1001 bit 1 followed by the minimum number of 0s and followed by a single bit 1. For
1002 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 for Keccak

1003 F.2 Ascon-Hash and Ascon-XOF

1004 The Ascon family [24] includes the hash functions Ascon-Hash and Ascon-Hasha
1005 as well as the extendable output functions Ascon-XOF and Ascon-XOFa with
1006 sponge-based modes of operations.

1007 **Ascon Permutation.** The inner permutation applies 12 round functions to a 320-
1008 bit state. The state A is split into five 64-bit words, and denote $A_{\{x,y\}}^{(r)}$ to be the
1009 x -th (column) bit of the y -th (row) 64-bit word, where $0 \leq y \leq 4$, $0 \leq x \leq 63$.
1010 The round function consists of three operations p_C , p_S , and p_L . Denote the
1011 internal states of round r as $A^{(r)} \xrightarrow{p_S \circ p_C} S^{(r)} \xrightarrow{p_L} A^{(r+1)}$.

- 1012 – **Addition of Constants** p_C : $A_{\{*,2\}}^{(r)} = A_{\{*,2\}}^{(r)} \oplus RC_r$.
- 1013 – **Substitution Layer** p_S : For each x , this step updates the columns $A_{\{x,*\}}^{(r)}$
1014 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

1015 $(b_0, b_1, b_2, b_3, b_4) \in \mathbb{F}_2^5$, where a_0 is the most significant bit. The algebraic
 1016 normal form (ANF) of the Sbox is as follows:

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

1017 – **Linear Diffusion Layer p_L :**

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

1018 **Ascon-Hash and Ascon-XOF.** The state A is composed of the outer part with 64
 1019 bits $A_{\{*,0\}}$ and the inner part 256 bits $A_{\{*,i\}}$ ($i = 1, 2, 3, 4$). For **Ascon-Hash**,
 1020 the output size is 256 bits, and the security claim is 2^{128} . For **Ascon-XOF**, the
 1021 output can have arbitrary length and the security claim against preimage attack
 1022 is $\min(2^{128}, 2^l)$, where l is the output length.

1023 F.3 Xoodyak and Xoodoo Permutation

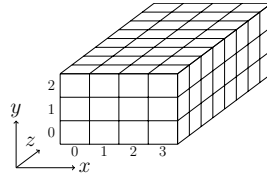


Fig. 21: Toy version of the Xoodoo state. The order in y is opposite to Keccak

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 $0 \leq x \leq 3$, $0 \leq y \leq 2$, $0 \leq z \leq 31$. 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

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

1024

$$\begin{aligned}
\theta : \theta_{\{x,y,z\}}^{(r)} &= A_{\{x,y,z\}}^{(r)} \oplus \sum_{y'=0}^2 (A_{\{x-1,y',z-5\}}^{(r)} \oplus A_{\{x-1,y',z-14\}}^{(r)}), \\
\rho_{\text{west}} : \rho_{\{x,0,z\}}^{(r)} &= \theta_{\{x,0,z\}}^{(r)}, \rho_{\{x,1,z\}}^{(r)} = \theta_{\{x-1,1,z\}}^{(r)}, \rho_{\{x,2,z\}}^{(r)} = \theta_{\{x,2,z-11\}}^{(r)}, \\
\iota : \iota_{\{0,0,z\}}^{(r)} &= \rho_{\{0,0,z\}}^{(r)} \oplus RC_r, \text{ where } RC_r \text{ is round-dependent constant,} \\
\chi : \chi_{\{x,y,z\}}^{(r)} &= \iota_{\{x,y,z\}}^{(r)} \oplus (\iota_{\{x,y+1,z\}}^{(r)} \oplus 1) \cdot \iota_{\{x,y+2,z\}}^{(r)}, \\
\rho_{\text{east}} : A_{\{x,0,z\}}^{(r+1)} &= \chi_{\{x,0,z\}}^{(r)}, A_{\{x,1,z\}}^{(r+1)} = \chi_{\{x,1,z-1\}}^{(r)}, A_{\{x,2,z\}}^{(r+1)} = \chi_{\{x-2,2,z-8\}}^{(r)}.
\end{aligned} \tag{34}$$

1025 Xoodyak can serve as a XOF, i.e. Xoodyak-XOF, which offers arbitrary output
 1026 length l . The preimage resistance is $\min(2^{128}, 2^l)$. We target on Xoodyak-XOF
 1027 with output of 128-bit hash value and 128-bit absorbed message block.

1028 G MITM Attacks on 3-round Xoodyak-XOF

1029 The specification of Xoodyak [15] (one of the finalists of NIST LWC) is given
 1030 in F.3. The Xoodyak-XOF offers an arbitrary output length l and the preimage
 1031 resistance is $\min(2^{128}, 2^l)$. We target Xoodyak-XOF with a 128-bit digest against
 the preimage attack and collision attack.

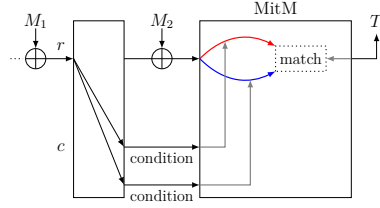


Fig. 22: MitM Preimage Attack [45]

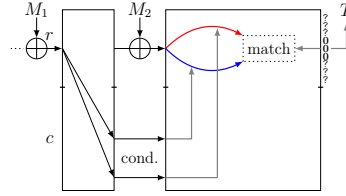


Fig. 23: MitM Collision Attack [27]

1032
1033

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

1034 Following Qin *et al.*'s framework [45] of MitM preimage attack on sponge func-
 1035 tion in Figure 22, we launch a new 3-round MitM preimage attack on Xoodyak-
 1036 XOF with a new 3-round MitM path given in Figure 24.

1037 The starting state $A^{(0)}$ contains $\lambda^+ = 7$ ■ bits and $\lambda^- = 67$ ■ bits. In the
 1038 computation from $A^{(0)}$ to $\iota^{(2)}$, the consumed degrees of freedom (DoFs) of ■ and
 1039 ■ are $l^+ = 0$ and $l^- = 60$ bits, respectively. Therefore, $\text{DoF}^+ = 7$, $\text{DoF}^- = 7$,
 1040 and there is $\text{DoM} = 7$ matching bits as in Eq. (35) with the deterministic relations
 1041 of $\iota^{(2)}$:

$$\begin{aligned}
\chi_{\{1,2,8\}}^{(2)} &= \iota_{\{1,2,8\}}^{(2)} \oplus (\iota_{\{1,0,8\}}^{(2)} \oplus 1) \cdot \iota_{\{1,1,8\}}^{(2)}, \chi_{\{1,2,10\}}^{(2)} = \iota_{\{1,2,10\}}^{(2)} \oplus (\iota_{\{1,0,10\}}^{(2)} \oplus 1) \cdot \iota_{\{1,1,10\}}^{(2)}, \\
\chi_{\{2,2,10\}}^{(2)} &= \iota_{\{2,2,10\}}^{(2)} \oplus (\iota_{\{2,0,10\}}^{(2)} \oplus 1) \cdot \iota_{\{2,1,10\}}^{(2)}, \chi_{\{1,2,17\}}^{(2)} = \iota_{\{1,2,17\}}^{(2)} \oplus (\iota_{\{1,0,17\}}^{(2)} \oplus 1) \cdot \iota_{\{1,1,17\}}^{(2)}, \\
\chi_{\{1,2,26\}}^{(2)} &= \iota_{\{1,2,26\}}^{(2)} \oplus (\iota_{\{1,0,26\}}^{(2)} \oplus 1) \cdot \iota_{\{1,1,26\}}^{(2)}, \chi_{\{1,2,31\}}^{(2)} = \iota_{\{1,2,31\}}^{(2)} \oplus (\iota_{\{1,0,31\}}^{(2)} \oplus 1) \cdot \iota_{\{1,1,31\}}^{(2)}, \\
\chi_{\{2,2,31\}}^{(2)} &= \iota_{\{2,2,31\}}^{(2)} \oplus (\iota_{\{2,0,31\}}^{(2)} \oplus 1) \cdot \iota_{\{2,1,31\}}^{(2)}.
\end{aligned} \tag{35}$$

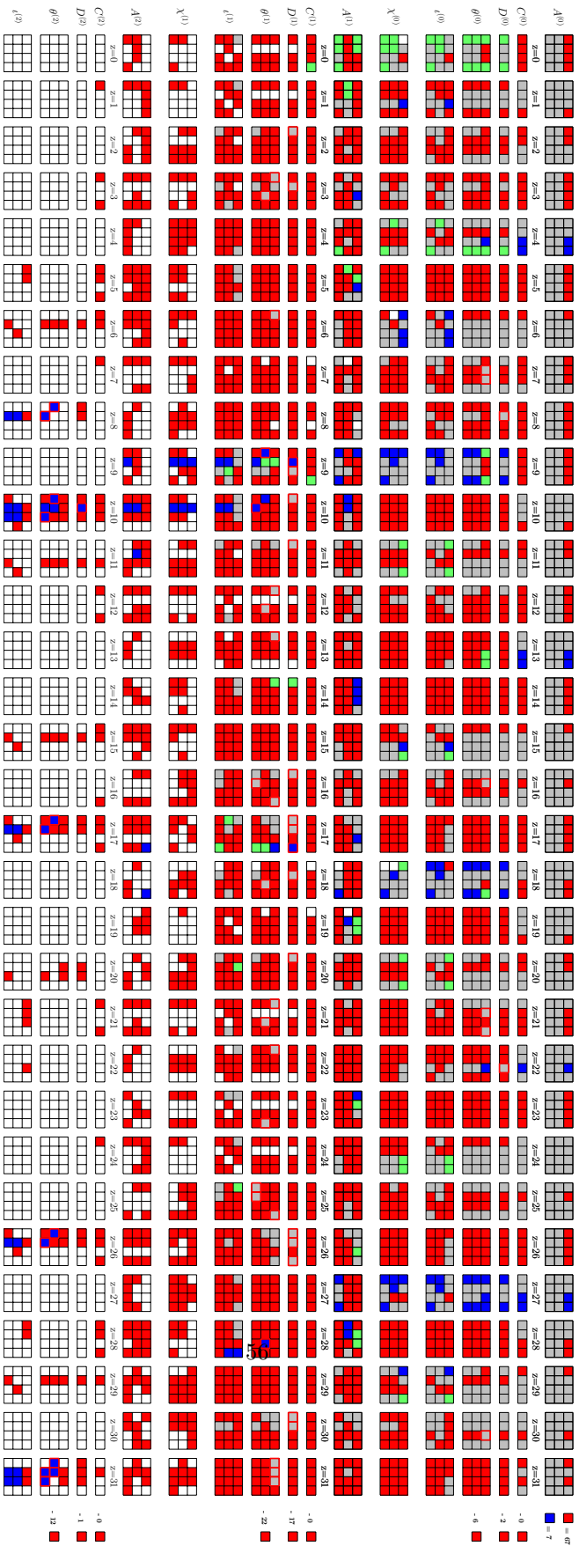


Fig. 24: The improved MitM preimage attack on 3-round Xoodyak-XOF

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.

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

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 $2^{128-\min\{\text{DoF}^+, \text{DoF}^-, \text{DoM}\}} = 2^{121}$ 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

```

1 for  $2^{\zeta_1}$  values of  $M_1$  /*  $\zeta_1 = 72$  */
2 do
3   Compute the inner part of the 2nd block
4   Solve the system of 70 linear equations
5   if the equations have solutions /* with probability of  $2^{-38}$  */
6   then
7     for  $2^{\zeta_2}$  solutions of  $M_2$  /*  $\zeta_2 = 20 \leq 20$  */
8     do
9        $U \leftarrow []$ 
10      for  $(e_1, e_2, \dots, e_{30}) \in \mathbb{F}_2^{30}$  do
11        for 37 free variables  $\blacksquare$  in  $A^{(0)} \in 2^{37}$  do
12          Assign  $(e_1, e_2, \dots, e_{30})$  to 30 expressions
13          Deduce the 30 dependent variables
14          Compute forward to other 30-bit  $\blacksquare/\blacksquare$   $\mathbf{u} \in \mathbb{F}_2^{30}$ 
15           $U[\mathbf{u}] \leftarrow A^{(0)}[v_{\mathcal{R}}]$  /*  $v_{\mathcal{R}}$  represents 67 red indexes */
16        end
17      for  $\mathbf{u} \in \mathbb{F}_2^{30}$  do
18         $L \leftarrow []$ 
19        for  $A^{(0)}[v_{\mathcal{B}}] \in 2^7$  /*  $v_{\mathcal{B}}$  represents 7 blue indexes */
20        do
21          Compute forward to the 7-bit matching point  $v$ 
22           $L[v] \leftarrow A^{(0)}[v_{\mathcal{B}}]$ 
23        end
24        for values in  $U[\mathbf{u}]$  do
25          Compute forward the 7-bit matching point  $v'$ 
26          for values in  $L[v']$  do
27            if it leads to the given hash value then
28              Output the preimage
29            end
30          end
31        end
32      end
33    end
34  end
35 end
36 end

```

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

```

1 Fix the 7-bit matching point to 0
2 for  $2^{\zeta_1}$  values of  $M_1$  /*  $\zeta = 38$  */
3 do
4   Compute the inner part of the 2nd block
5   Solve the system of 70 linear equations
6   if the equations have solutions /* with probability of  $2^{-38}$  */
7   then
8     for  $2^{\zeta_2}$  solutions of  $M_2$  /*  $\zeta_2 = 0 \leq 20$  */
9     do
10       $U \leftarrow []$ 
11      for  $2^{23.5}$  values of  $(e_1, e_2, \dots, e_{30}) \in \mathbb{F}_2^{30}$  do
12        for 37 free variables  $\blacksquare$  in  $A^{(0)} \in 2^{37}$  do
13          Assign  $(e_1, e_2, \dots, e_{30})$  to 30 expressions
14          Deduce the 30 dependent variables
15          Compute forward to other 30-bit  $\blacksquare/\blacksquare$   $\mathbf{u} \in \mathbb{F}_2^{30}$ 
16           $U[\mathbf{u}] \leftarrow A^{(0)}[v_{\mathcal{R}}]$  /*  $v_{\mathcal{R}}$  represents all 67 red indexes */
17        end
18      for  $\mathbf{u} \in \mathbb{F}_2^{30}$  do
19         $L \leftarrow []$ 
20        for  $A^{(0)}[v_{\mathcal{B}}] \in 2^7$  /*  $v_{\mathcal{B}}$  represents all 7 blue indexes */
21        do
22          Compute forward to the 7-bit matching point  $v$ 
23           $L[v] \leftarrow A^{(0)}[v_{\mathcal{B}}]$ 
24        end
25        for values in  $U[\mathbf{u}]$  do
26          Compute forward the 7-bit matching point  $v'$ 
27          for values in  $L[v']$  do
28            Compute the 128-bit target  $h$  and store the
               $(M_1, M_2, h)$  in  $L_1$  indexed by  $h$ 
29            if the size of  $L_1$  is  $2^{(128-7)/2} = 2^{60.5}$  then
30              Check  $L_1$  and return  $(M_1, M_2)$  and  $(M'_1, M'_2)$ 
                with the same  $h$ 
31            end
32          end
33        end
34      end
35    end
36  end
37 end
38 end

```

1071 H MITM Preimage Attacks on Ascon-XOF

1072 The description of Ascon-XOF are given in Supplementary Material F.2. The 320-
 1073 bit state A of Ascon is divided into five 64-bit words, and denote $A_{\{x,y\}}^{(r)}$ to be the
 1074 x -th (column) bit of the y -th (row) 64-bit word, where $0 \leq y \leq 4$, $0 \leq x \leq 63$.

1075 We explore the symmetry in the x -axis to speed up the search by cutting the
 1076 full 64-bit word into 32-bit word. Therefore, the linear operation works modular
 1077 32 instead of 64. For example, the linear operation in the second row changes
 1078 from the original $A_{\{*,1\}}^{(r+1)} \leftarrow S_{\{*,1\}}^{(r)} \oplus (S_{\{*,1\}}^{(r)} \ggg 61) \oplus (S_{\{*,1\}}^{(r)} \ggg 39)$ into the current
 1079 $A_{\{*,1\}}^{(r+1)} \leftarrow S_{\{*,1\}}^{(r)} \oplus (S_{\{*,1\}}^{(r)} \ggg 29) \oplus (S_{\{*,1\}}^{(r)} \ggg 7)$.

1080 H.1 3-round Preimage Attack on Ascon-XOF

1081 The 3-round MitM characteristic is shown in Figure 25. The starting state $A^{(0)}$
 1082 in the full MitM path contains $\lambda^+ = 14$ ■ bits and $\lambda^- = 24$ ■ bits. There
 1083 are 48 conditions on ■ bits of $A^{(0)}$. In the computation from $A^{(0)}$ to $A^{(2)}$, the
 1084 consumed DoFs of ■ and ■ are $l^+ = 0$ and $l^- = 10$ bits, respectively. Therefore,
 1085 $\text{DoF}^+ = 14$, $\text{DoF}^- = 14$, and there are $\text{DoM} = 14$ matching bits.

1086 Using new TA to system of 10 constraints on red ■ bits, we get 14 free vari-
 1087 ables and 10 dependent variables, the matrices before and after TA are shown
 1088 in Eq. (37)⁷. The 3-round attack is given in Algorithm 15. The total time com-
 1089 plexity is about $2^{128 - \min\{\text{DoF}^+, \text{DoF}^-, \text{DoM}\}} = 2^{114}$ time and 2^{14} memory.

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

⁷Note 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.

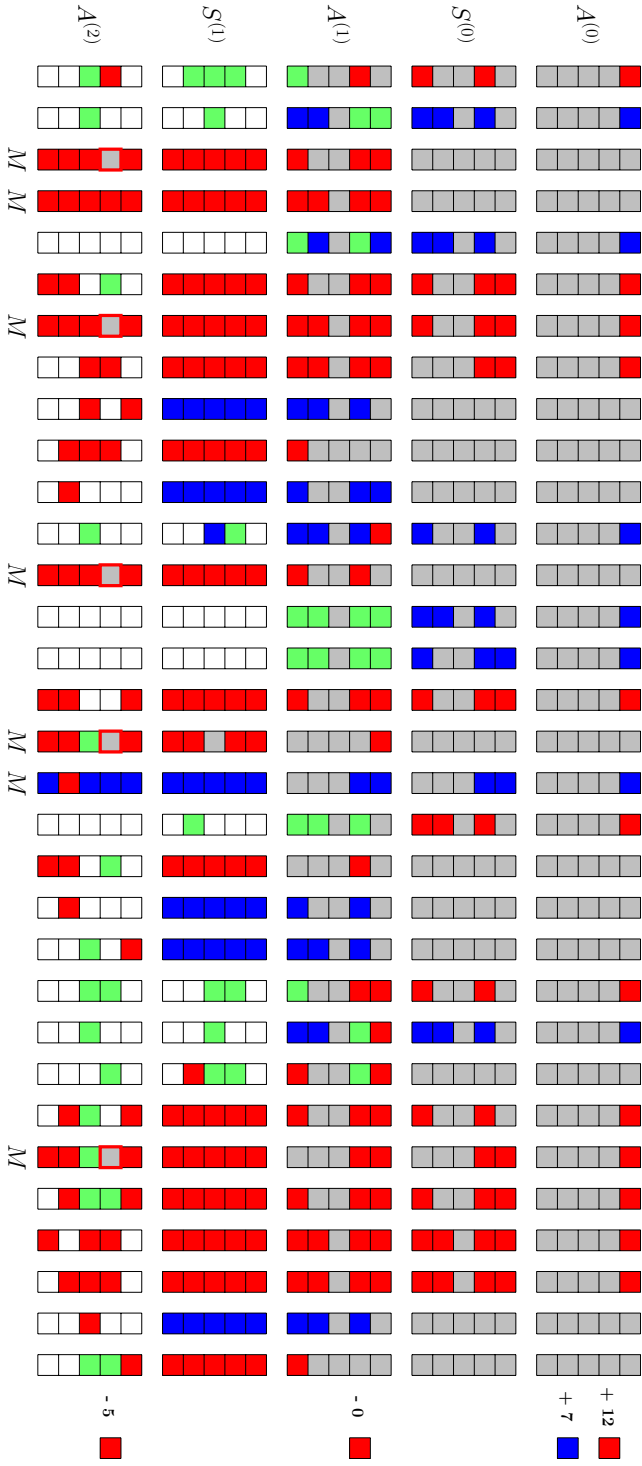


Fig. 25: The 3-round Preimage attack on Ascon-XOF. Since we explore the symmetry in the x -axis to speed up the search by cutting the full 64-bit word into 32-bit word, only 32-bit state version of Ascon-XOF is displayed here.

$$\begin{pmatrix}
\begin{array}{c|cccccccccccc}
& v_0 & v_5 & v_6 & v_7 & v_{15} & v_{18} & v_{22} & v_{25} & v_{26} & v_{27} & v_{28} & v_{29} \\
\hline
A_{\{2,1\}}^{(2)} & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\
A_{\{6,1\}}^{(2)} & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
A_{\{12,1\}}^{(2)} & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
A_{\{16,1\}}^{(2)} & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
A_{\{26,1\}}^{(2)} & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1
\end{array}
&
\begin{array}{c|cccccccc|cccc}
& v_7 & v_{25} & v_5 & v_6 & v_{22} & v_0 & v_{15} & v_{18} & v_{26} & v_{27} & v_{28} & v_{29} \\
\hline
A_{\{26,1\}}^{(2)} & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\
A_{\{2,1\}}^{(2)} & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
A_{\{12,1\}}^{(2)} & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
A_{\{16,1\}}^{(2)} & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\
A_{\{26,1\}}^{(2)} & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1
\end{array}
\end{pmatrix} \quad (37)$$

(a)
(b)

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

```

1  Compute  $S_{\{*,0\}}^{(3)} = p_L^{-1}(T)$  /*  $T$  is the hash value */
2  for  $2^\zeta$  values of  $(M_1, M_2)$  /*  $\zeta = 114$  */
3  do
4      Compute the inner part of the 3rd block
5      if the 48 conditions are satisfied /* with probability of  $2^{-48}$  */
6      then
7          for  $2^{24}$  values of  $\blacksquare$  bits in  $A^{(0)}$  do
8              for  $(e_1, e_2, \dots, e_{10}) \in \mathbb{F}_2^{10}$  do
9                   $L \leftarrow []$ 
10                 for  $A^{(0)}[v_B] \in 2^{14}$  /*  $v_B$  represents all 14 blue indexes */
11                 do
12                     Compute forward to the 14-bit matching point  $v$ 
13                      $L[v] \leftarrow A^{(0)}[v_B]$ 
14                 end
15                 for 14 free variables  $\blacksquare$  in  $A^{(0)} \in 2^{14}$  do
16                     Assign  $(e_1, e_2, \dots, e_{10})$  to the last 10 expressions in Eq.
17                     (37) (b)
18                     /* Due to symmetry, the complete matrix in Eq. (37)
19                     (b) contains 10 rows and 24 columns. */
20                     Deduce 10 dependent variables
21                     Compute forward to the 14-bit matching point  $v'$ 
22                     for values in  $L[v']$  do
23                         Check if  $T$  is satisfied
24                     end
25                 end
26             end
27         end
28     end
29 end

```

1090 **Partial Experiment of the Preimage Attack on 3-round Ascon-XOF.**
1091 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 $S_{\{i,0\}}^{(2)}$ ($i \in [2, 3, 6, 12, 16, 17, 26, 34, 35, 38, 44, 48, 49, 58]$) and another 18-bit target $S_{\{j,0\}}^{(2)}$ ($j \in [0, 1, 4, 5, 7, 8, 9, 10, 11, 13, 14, 15, 18, 19, 20, 21, 22, 23]$) 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.*, $A_{\{*,1\}}^{(0)} = 0xc8142340c8142340$, $A_{\{*,3\}}^{(0)} = 0x8713427087134270$, and $A_{\{*,2\}}^{(0)} = A_{\{*,4\}}^{(0)} = 0x0$. In order to simplify the expressions for TA, we set 7 \blacksquare bits $A_{\{i,0\}}^{(0)} = 1$ for $i \in [12, 16, 19, 30, 44, 48, 51]$, and the remaining 17 \blacksquare 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, \dots, 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 $(A_{\{*,0\}}^{(0)})$	32-bit target $S_{\{i,0\}}^{(2)}$ ($i \in [0 - 23, 26, 34, 35, 38, 44, 48, 49, 58]$)
$r = 3$	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 $A^{(0)}$ in the full MitM path contains $\lambda^+ = 4 \blacksquare$ bits and $\lambda^- = 34 \blacksquare$ bits. In the computation from $A^{(0)}$ to $A^{(3)}$, the consumed degrees of freedom (DoFs) of \blacksquare and \blacksquare are $l^+ = 0$ and 26 bits, respectively. Additionally, there are 4 consumed DoFs of \blacksquare to make

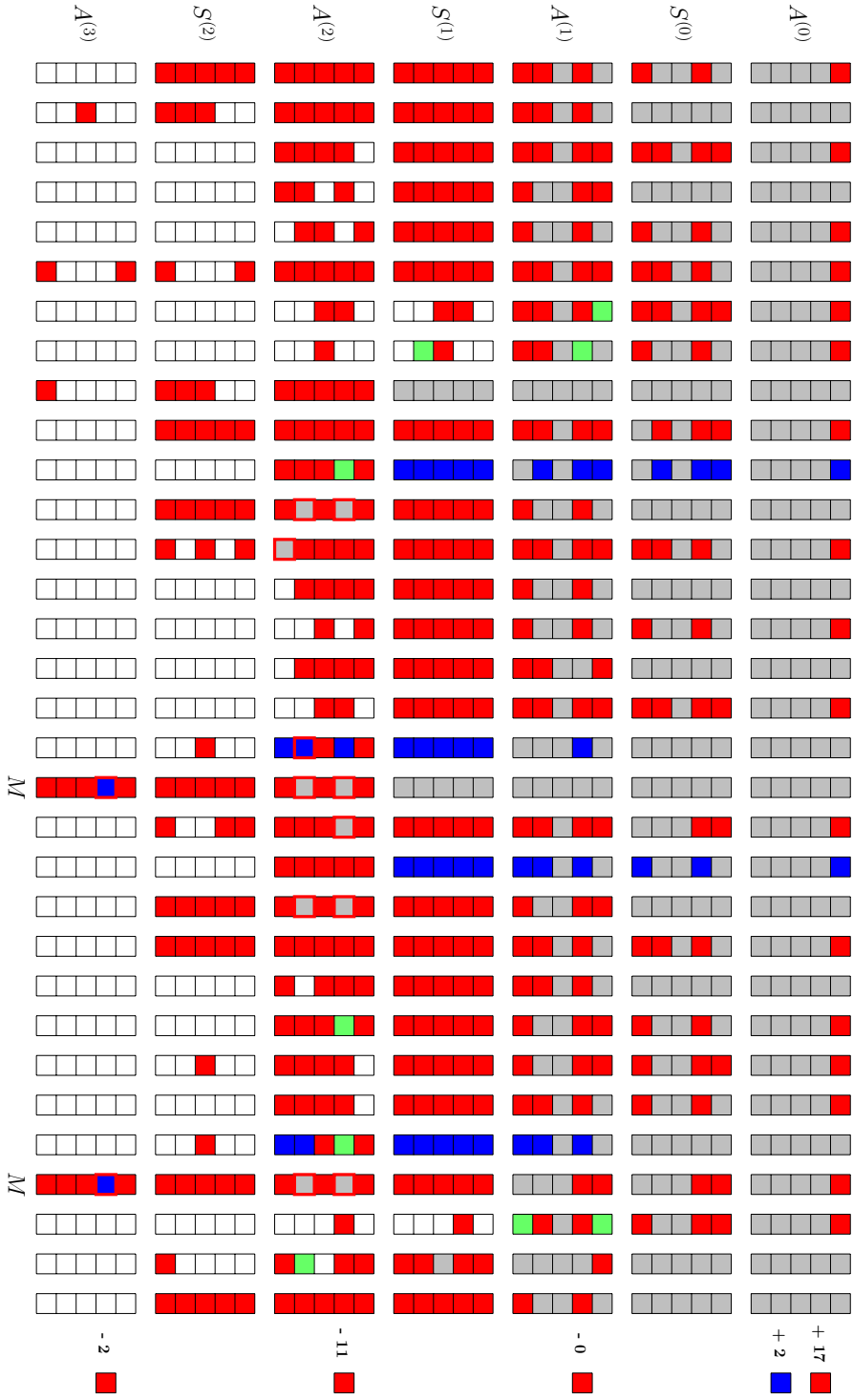


Fig. 26: The 4-round Preimage attack on Ascon-XOF. Since we explore the symmetry in the x -axis to speed up the search by cutting the full 64-bit word into 32-bit word, only 32-bit state version of Ascon-XOF is displayed here.

1115 $a_0 + a_2 + a_4$ become \blacksquare for matching points, so that $l^- = 26 + 4 = 30$. Therefore,
1116 $\text{DoF}^+ = 4$, $\text{DoF}^- = 4$, and there are $\text{DoM} = 4$ matching bits.
1117 Using the new TA, we can get 14 free variables and 20 dependent variables,
1118 the matrices before and after TA are shown in Eq. (38). The 4-round attack is
1119 given in Algorithm 16. The total time complexity is about $2^{128 - \min\{\text{DoF}^+, \text{DoF}^-, \text{DoM}\}} =$
1120 2^{124} time and 2^{14} memory.

$$\begin{aligned}
& \begin{pmatrix} & 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} \\ A_{\{11,1\}}^{(2)} & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ A_{\{18,1\}}^{(2)} & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ A_{\{19,1\}}^{(2)} & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ A_{\{21,1\}}^{(2)} & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ A_{\{28,1\}}^{(2)} & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ A_{\{11,3\}}^{(2)} & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ A_{\{17,3\}}^{(2)} & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ A_{\{18,3\}}^{(2)} & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ A_{\{21,3\}}^{(2)} & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ A_{\{28,3\}}^{(2)} & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ A_{\{12,4\}}^{(2)} & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ A_{\{31,4\}}^{(3)} & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ A_{\{31,4\}}^{(3)} & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \\
& \quad (a) \\
& \begin{pmatrix} & 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} \\ A_{\{31,4\}}^{(3)} & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ A_{\{31,4\}}^{(3)} & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ A_{\{11,3\}}^{(2)} & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ A_{\{17,3\}}^{(2)} & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ A_{\{12,4\}}^{(2)} & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ A_{\{19,1\}}^{(2)} & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ A_{\{28,1\}}^{(2)} & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ A_{\{18,3\}}^{(2)} & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ A_{\{28,3\}}^{(2)} & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ A_{\{21,1\}}^{(2)} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ A_{\{11,1\}}^{(2)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ A_{\{21,3\}}^{(2)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ A_{\{18,1\}}^{(2)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \\
& \quad (b)
\end{aligned} \tag{38}$$

1121 **Partial Experiment of the Preimage Attack on 4-round Ascon-XOF.** In
1122 this practical experiment, we attempt to find a preimage of 24-bit all-zero target.
1123 That is, the 24-bit $S_{\{i,0\}}^{(3)}$ ($i \in [16 - 19, 28 - 31, 48 - 63]$) of target are fixed to be
1124 zero. The attack procedures are as follows:

- 1125 1. Set the 256-bit inner part in $A^{(0)}$ as fixed values, which satisfied the 50-bit
1126 condition, *i.e.*, $A_{\{*,1\}}^{(0)} = 0x8d0a0aa08d0a0aa0$, $A_{\{*,3\}}^{(0)} = 0x890218ec890218ec$,
1127 and $A_{\{*,2\}}^{(0)} = A_{\{*,4\}}^{(0)} = 0x0$. In order to simplify the expressions for TA, we

Algorithm 16: Preimage Attack on 4-round Ascon-XOF

```

1  Compute  $S_{\{*,0\}}^{(3)} = p_L^{-1}(T)$ 
2  /*  $T$  is the hash value */
3  for  $2^\zeta$  values of  $(M_1, M_2)$  /*  $\zeta = 116$  */
4  do
5    Compute the inner part of the 3rd block
6    if the 50 conditions are satisfied /* with probability of  $2^{-50}$  */
7    then
8      for  $2^{24}$  values of  $\blacksquare$  bits in  $A^{(0)}$  do
9        for  $(e_1, e_2, \dots, e_{20}) \in \mathbb{F}_2^{20}$  do
10          $U \leftarrow []$ 
11         for 14 free variables  $\blacksquare$  in  $A^{(0)} \in 2^{14}$  do
12           Assign  $(e_1, e_2, \dots, e_{20})$  to the last 20 expressions in Eq.
13           (38) (b)
14           Deduce 20 dependent variables
15           Compute forward to other 10-bit  $\blacksquare/\blacksquare$   $\mathbf{u} \in \mathbb{F}_2^{10}$ 
16            $U[\mathbf{u}] \leftarrow A^{(0)}[v_{\mathcal{R}}]$  /*  $v_{\mathcal{R}}$  represents all 34 red indexes */
17         end
18         for  $\mathbf{u} \in \mathbb{F}_2^{10}$  do
19            $L \leftarrow []$ 
20           for  $(A_{\{10,0\}}^{(0)}, A_{\{20,0\}}^{(0)}, A_{\{42,0\}}^{(0)}, A_{\{52,0\}}^{(0)}) \in \mathbb{F}_2^4$  do
21             Compute forward to the 4-bit matching point  $v$ 
22              $L[v] \leftarrow (A_{\{10,0\}}^{(0)}, A_{\{20,0\}}^{(0)}, A_{\{42,0\}}^{(0)}, A_{\{52,0\}}^{(0)})$ 
23           end
24           for values in  $U[\mathbf{u}]$  do
25             Compute forward to the 4-bit matching point  $v'$ 
26             for values in  $L[v']$  do
27               Check if  $T$  is satisfied
28             end
29           end
30         end
31       end
32     end
33 end

```

1128 set the 2 ■ bits $A_{\{11,0\}}^{(0)} = A_{\{50,0\}}^{(0)} = 1$, and the remaining 22 ■ bits to 0. The
 1129 simplified equations are given in Eq. (39).

1130 2. In our experiment, we traverse the values of the first 10-bit expressions
 1131 $(e_1, e_2, \dots, e_{10})$ and fix $e_{11} = e_{12} = \dots = e_{20} = 0$ in Eq. (39), *i.e.*, $2^{10+14+4} =$
 1132 2^{28} states are tested with MitM approach and it is expected to find $2^{28-24} =$
 1133 2^4 preimages.

1134 Finally, there are $17 = 2^{4.1}$ preimages are searched to satisfy the 24-bit all-
 1135 zero target, which is close to the theoretical expectation. The results are listed
 1136 in Table 7. The experiment can be run on a computer with an i9-13900KF
 1137 CPU and 32 GB of memory in seconds. The source codes are available via
 1138 <https://anonymous.4open.science/r/Triangulation-MitM-7373>.

Round	First row of preimage $(A_{\{*,0\}}^{(0)})$	24-bit target $S_{\{i,0\}}^{(3)}$ ($i \in [16 - 19, 28 - 31, 48 - 63]$)
$r = 4$	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

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

1139 I MitM Preimage Attacks on Reduced KECCAK

1140 I.1 Memory Improved Preimage attack on 4-round KECCAK[1024]

1141 We reuse the same initialize structure of the attack on KECCAK[1024] in [46],
1142 and get a new characteristic with lower attack memory, which is shown in Figure
1143 27 (first part) and Figure 28 (second part). The starting state $A^{(0)}$ contains 16
1144 ■ bits and 216 ■ bits, and there are totally 464 conditions on ■ bits of $\pi^{(0)}$. Due
1145 to the CP-kernel property, the initial degree of freedoms (DoFs) of ■ and ■ are
1146 $\lambda^+ = 8$ and $\lambda^- = 108$. In the computation $A^{(0)}$ to $A^{(3)}$, the consumed degrees of
1147 freedom (DoFs) of ■ and ■ are $l^+ = 0$ and $l^- = 100$ bits, respectively. Therefore,
1148 $\text{DoF}^+ = 8$, $\text{DoF}^- = 8$, and there are $\text{DoM} = 8$ matching bits.

1149 We use two message blocks (M_1, M_2) to build the attack following the frame-
 1150 work of Qin *et al.* [45] in Figure 22. Given an inner part, the 464 conditions can
 1151 be seen as a linear system of $1600 - 512 \times 2 - 116 = 460$ variables of M_2 (512×2
 1152 is the number of capacity bits, 116 are the DoFs of \blacksquare and \blacksquare bits in $A^{(0)}$), which
 1153 will act as global parameters. The rank of the coefficient matrix of the linear
 1154 system is 250. In other words, through some linear transformations, there are
 1155 $464 - 250 = 214$ equations determined only by the bits of the inner part. To find a
 1156 right inner part, we have to randomly test $2^{214} M_1$ and there are $2^{460-250} = 2^{210}$
 1157 solutions of M_2 , which make all the conditions hold.

1158 Using TA, we can get 52 free variables and 56 dependent variables. We put
 1159 the TA matrices in MATRIX/keccak[1024]_4r_eu23.txt at [https://anonymous.
 1160 4open.science/r/Triangulation-MitM-7373](https://anonymous.4open.science/r/Triangulation-MitM-7373).

1161 The steps for the 4-round MitM attack are given in Algorithm 17. The total
 1162 preimage attack on 4-round KECCAK[1024] is about $2^{512-\min\{\text{DoF}^+, \text{DoF}^-, \text{DoM}\}} =$
 1163 2^{504} time and 2^{52} memory.

1164 I.2 MitM Preimage Attack on 4-round KECCAK[1024]

1165 Using the same model in [46], we get a new characteristic as shown in Figure
 1166 29. Due to the symmetry, in the full MitM path, the starting state $A^{(0)}$ contains
 1167 24 \blacksquare bits and 424 \blacksquare bits, and there are totally 338 conditions on \blacksquare bits of $\pi^{(0)}$.
 1168 Due to the CP-kernel property, the initial degree of freedoms (DoFs) of \blacksquare and
 1169 \blacksquare are $\lambda^+ = 12$ and $\lambda^- = 212$. In the computation $A^{(0)}$ to $A^{(3)}$, the consumed
 1170 degrees of freedom (DoFs) of \blacksquare and \blacksquare are $l^+ = 0$ and $l^- = 200$ bits, respectively.
 1171 Therefore, $\text{DoF}^+ = 12$, $\text{DoF}^- = 12$, and there is $\text{DoM} = 12$ matching bits.

1172 We introduce 224 binary variables $v = \{v_0, v_1, \dots, v_{223}\}$ and 224 binary
 1173 variables $c = \{c_0, c_1, \dots, c_{223}\}$. Those variables v_i 's and c_i 's are placed at the
 1174 $24 + 424 = 448$ \blacksquare and \blacksquare bits in $A^{(0)}$. For example, set $A_{\{0,0,0\}}^{(0)} = v_0$ and $A_{\{0,1,0\}}^{(0)} =$
 1175 $v_0 \oplus c_0$ due to the CP-kernel property.

1176 We use two message blocks (M_1, M_2) to build the attack. Given a value for
 1177 the inner part of the 2nd block, the 338 conditions on $A^{(0)}$ will be a linear system
 1178 on 352 variables of M_2 , including $576 - 448 = 128$ \blacksquare bits and 224 Binary variables
 1179 $c = \{c_0, c_1, \dots, c_{223}\}$. The rank of the coefficient matrix of the linear system is
 1180 266. There are $338 - 266 = 72$ conditions out of the total 338 only determined by
 1181 the value of the inner part. Randomly test $2^{72} M_1$ to expect one satisfying the
 1182 72 conditions of the inner part. Then for the right M_1 , there are $2^{352-266} = 2^{86}$
 1183 solutions of M_2 , which make all the 338 equations hold.

1184 Using TA algorithm, we can get 118 free variables and 94 dependent variables.
 1185 We put the TA matrices in MATRIX/keccak[1024]_4r.txt at [https://anonymous.
 1186 4open.science/r/Triangulation-MitM-7373](https://anonymous.4open.science/r/Triangulation-MitM-7373).

1187 The steps for the 4-round MitM attack are given in Algorithm 18. To find a
 1188 512-bit target preimage, we need $2^{\zeta_1-72+\zeta_2+\zeta_3+106+12+12} = 2^{512}$. Set $\zeta_1 = 274$,
 1189 $\zeta_2 = 86$ and $\zeta_3 = 94$. The total preimage attack on 4-round KECCAK[1024] is
 1190 about $2^{512-\min\{\text{DoF}^+, \text{DoF}^-, \text{DoM}\}} = 2^{500}$ time and 2^{118} memory.

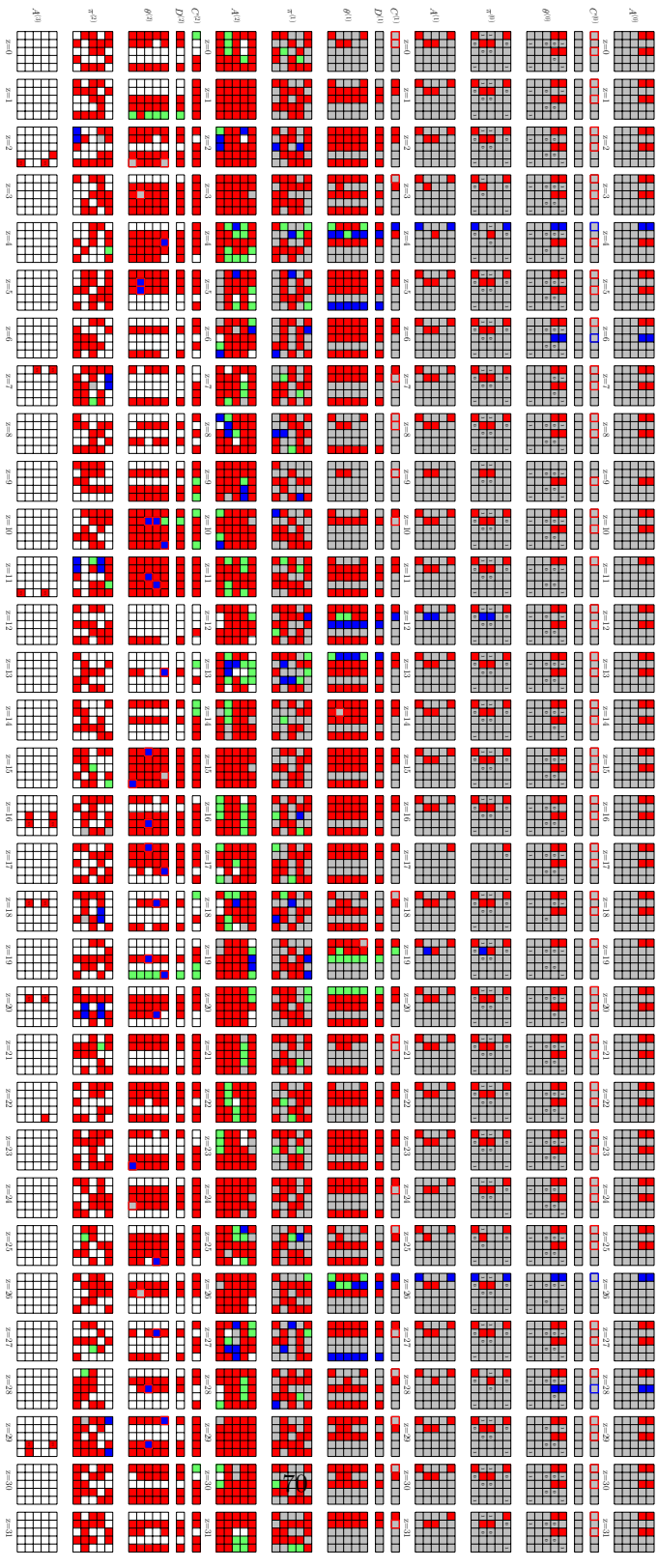


Fig. 27: Part-1: The 4-round MitM preimage attack on KECCAK[1024]

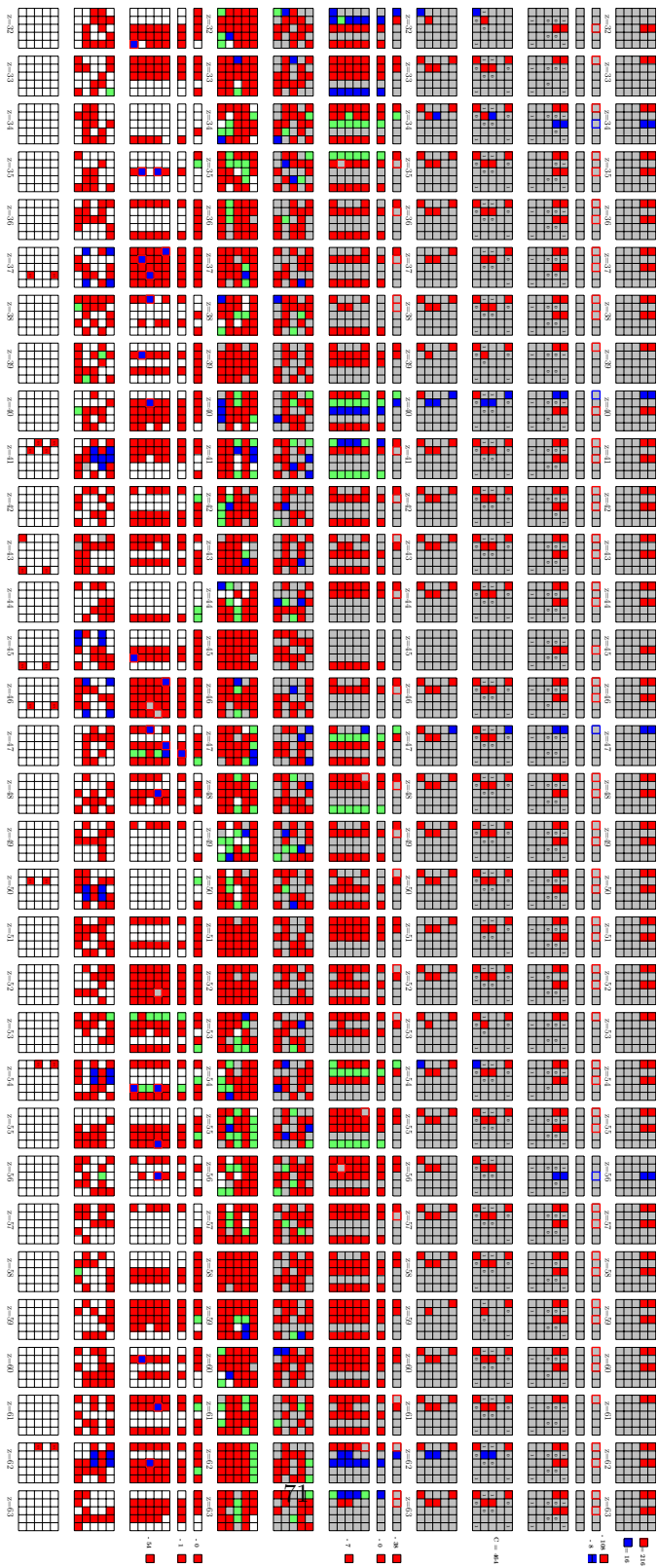


Fig. 28: Part-2: The 4-round MitM preimage attack on KECCAK[1024]

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

```

1  Precompute inversely from the target to  $A^{(3)}$ 
2  for  $2^{\zeta_1}$  values of  $M_1$     /*  $\zeta_1 = 400$  */
3  do
4      Compute the inner part of the 2nd block and solve the system of 464
        linear equations
5      if the equations have solutions /* with probability of  $2^{-214}$  */
6      then
7          for each of the  $2^{\zeta_2}$  solutions of  $M_2$     /*  $\zeta_2 = 210 \leq 210$  */
8          do
9              for  $2^{\zeta_3}$  values  $(e_1, e_2, \dots, e_{56}) \in \mathbb{F}_2^{56}$  do
10                  $U \leftarrow []$ 
11                 for 52 free variables  $\blacksquare$  in  $A^{(0)} \in 2^{52}$  do
12                     Assign  $(e_1, e_2, \dots, e_{56})$  to 56 expressions
13                     Deduce 56 dependent variables
14                     Compute forward to other 44-bit  $\blacksquare/\blacksquare$   $\mathbf{u} \in \mathbb{F}_2^{44}$ 
15                      $U[\mathbf{u}] \leftarrow A^{(0)}[v_{\mathcal{R}}]$  /*  $v_{\mathcal{R}}$  represents all 216 red indexes */
16                 end
17                 for  $\mathbf{u} \in \mathbb{F}_2^{44}$  do
18                      $L \leftarrow []$ 
19                     for  $A^{(0)}[v_{\mathcal{B}}] \in 2^8$  /*  $v_{\mathcal{B}}$  represents all 8 blue indexes */
20                     do
21                         Compute forward to the 8-bit matching point  $v$ 
22                          $L[v] \leftarrow A^{(0)}[v_{\mathcal{B}}]$ 
23                     end
24                     for values in  $U[\mathbf{u}]$  do
25                         Compute forward the 8-bit matching point  $v'$ 
26                         for values in  $L[v']$  do
27                             if it leads to the given hash value then
28                                 Output the preimage
29                             end
30                         end
31                     end
32                 end
33             end
34         end
35     end
36 end

```

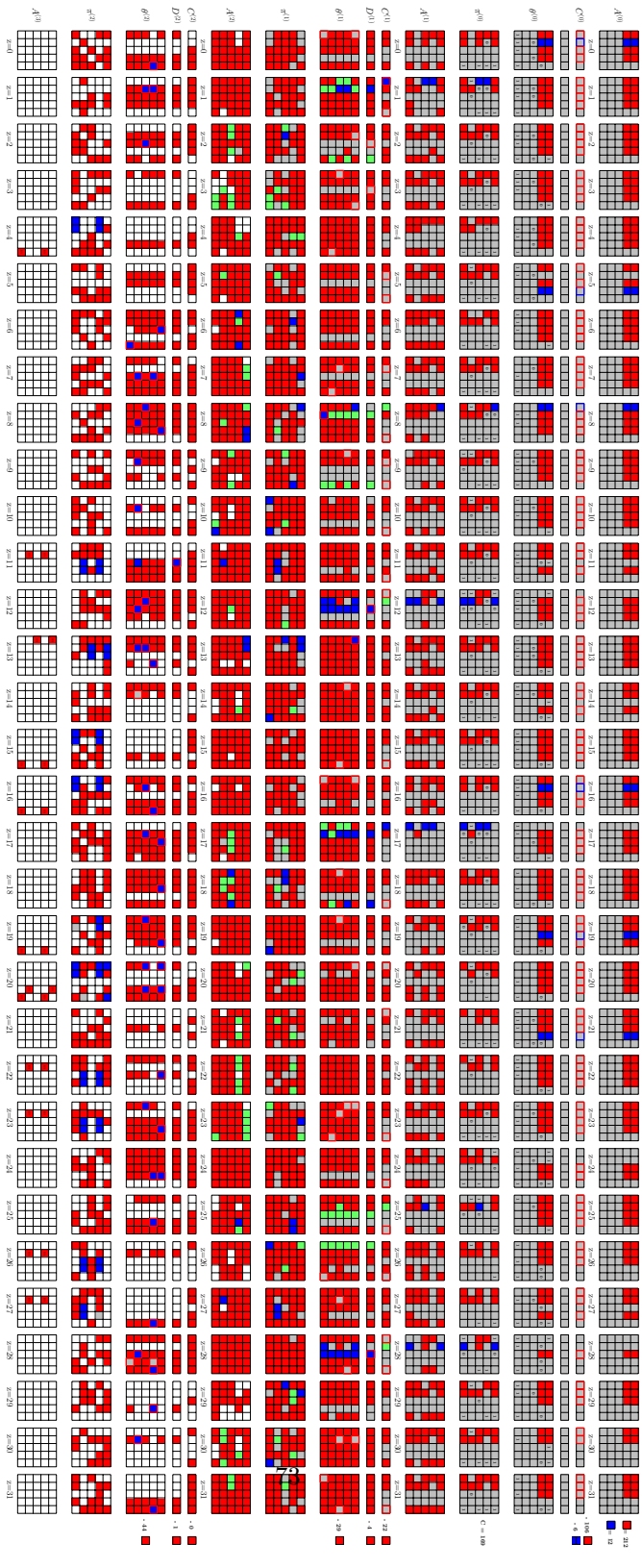


Fig. 29: The 4-round MitM preimage attack on KECCAK[1024]

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

```

1 Precompute inversely from the target to  $A^{(3)}$ 
2 for  $2^{\zeta_1}$  values of  $M_1$  /*  $\zeta_1 = 274$  */
3 do
4   Compute the inner part of the 2nd block and solve the system of 338
   linear equations
5   if the equations have solutions /* with probability of  $2^{-72}$  */
6   then
7     for each of the  $2^{\zeta_2}$  solutions of  $M_2$  /*  $\zeta_2 = 86 \leq 86$  */
8     do
9       for  $2^{\zeta_3}$  values  $(e_1, e_2, \dots, e_{94}) \in \mathbb{F}_2^{94}$  do
10         $U \leftarrow []$ 
11        for 118 free variables  $\blacksquare$  in  $A^{(0)} \in 2^{118}$  do
12          Assign  $(e_1, e_2, \dots, e_{94})$  to 94 expressions
13          Deduce 94 dependent variables
14          Compute forward to other 106-bit  $\blacksquare/\blacksquare$   $\mathbf{u} \in \mathbb{F}_2^{106}$ 
15           $U[\mathbf{u}] \leftarrow A^{(0)}[v_{\mathcal{R}}]$  /*  $v_{\mathcal{R}}$  represents all 212 red indexes
          */
16        end
17        for  $\mathbf{u} \in \mathbb{F}_2^{106}$  do
18           $L \leftarrow []$ 
19          for  $A^{(0)}[v_{\mathcal{B}}] \in 2^{12}$  /*  $v_{\mathcal{B}}$  represents all 12 blue
          indexes */
20          do
21            Compute forward to the 12-bit matching point  $v$ 
22             $L[v] \leftarrow A^{(0)}[v_{\mathcal{B}}]$ 
23          end
24          for values in  $U[\mathbf{u}]$  do
25            Compute forward the 12-bit matching point  $v'$ 
26            for values in  $L[v']$  do
27              if it leads to the given hash value then
28                Output the preimage
29              end
30            end
31          end
32        end
33      end
34    end
35  end
36 end

```

1191 I.3 MitM Preimage Attacks on 4-round KECCAK[768]

1192 Using the same model in [46], we get a new characteristic as shown in Figure 30
 1193 (first part) and Figure 31 (second part). In the full path, the starting state $A^{(0)}$
 1194 contains 28 ■ bits and 512 ■ bits, and there are totally 428 conditions on ■ bits
 1195 of $\pi^{(0)}$. Due to the CP-kernel property, the initial degree of freedoms (DoFs) of ■
 1196 and ■ are $\lambda^+ = 17$ and $\lambda^- = 316$. In the computation $A^{(0)}$ to $A^{(3)}$, the consumed
 1197 degrees of freedom (DoFs) of ■ and ■ are $l^+ = 0$ and $l^- = 299$ bits, respectively.
 1198 Therefore, $\text{DoF}^+ = 17$, $\text{DoF}^- = 17$, and there are $\text{DoM} = 17$ matching bits.

1199 We introduce 333 binary variables $v = \{v_0, v_1, \dots, v_{332}\}$ and 207 binary
 1200 variables $c = \{c_0, c_1, \dots, c_{206}\}$. Those variables v_i 's and c_i 's are placed at the
 1201 $28 + 512 = 540$ ■ and ■ bits in $A^{(0)}$. For example, set $A_{\{0,0,0\}}^{(0)} = v_0$, $A_{\{0,1,0\}}^{(0)} = v_1$
 1202 and $A_{\{0,2,0\}}^{(0)} = v_0 \oplus v_1 \oplus c_0$ due to the CP-kernel property.

1203 We use two message blocks (M_1, M_2) to build the attack. Given an inner part,
 1204 the 428 conditions can be seen as a linear system of $1600 - 384 \times 2 - 333 = 499$
 1205 variables of M_2 (384×2 is the number of capacity bits, 333 are the DoFs of ■
 1206 and ■ bits in $A^{(0)}$), which will act as global parameters, including 292 ■ of M_2
 1207 and 207 binary variables $c_{x,z}$'s. The rank of the coefficient matrix of the linear
 1208 system is 343. In other words, through some linear transformations, there are
 1209 $428 - 343 = 85$ equations determined only by the bits of the inner part. We
 1210 have to randomly test 2^{85} M_1 to compute a right inner part satisfying the 85
 1211 equations. Then for a right inner part, there are $2^{499-343} = 2^{156}$ solutions of M_2 ,
 1212 which make all the conditions hold.

1213 Using TA, we can get 157 free variables and 159 dependent variables. We put
 1214 the TA matrices in MATRIX/keccak[768]_4r.txt at <https://anonymous.4open.science/r/Triangulation-MitM-7373>.
 1215

1216 The steps for the 4-round MitM attack are given in Algorithm 19. To find a
 1217 384-bit target preimage, we need $2^{\zeta_1 - 85 + \zeta_2 + \zeta_3 + 140 + 17 + 17} = 2^{384}$. Set $\zeta_1 = 85$,
 1218 $\zeta_2 = 51$ and $\zeta_3 = 159$. The total preimage attack on 4-round KECCAK[768] is
 1219 about $2^{384 - \min\{\text{DoF}^+, \text{DoF}^-, \text{DoM}\}} = 2^{367}$ time and 2^{157} memory.

1220 J MitM Preimage Attacks on 10-round Gimli

1221 The 10-round MitM characteristic of Gimli-XOF-128 is shown in Figure 34. In
 1222 the MitM path, the starting state $A^{(0)}$ contains $\lambda^+ = 6$ ■ bit and $\lambda^- = 64$
 1223 ■ bits. There are 63 conditions on $A^{(0)}$ following Qin *et al.* MitM framework
 1224 in Figure 22. In the computation from $A^{(0)}$ to $A^{(9)}$, the consumed degrees of
 1225 freedom (DoFs) of ■ and ■ are $l^+ = 0$ and $l^- = 61$ bits, respectively. Therefore,
 1226 $\text{DoF}^+ = 6$, $\text{DoF}^- = 3$, and there is $\text{DoM} = 3$ matching bits.

1227 The steps for the 10-round MitM attack are given in Algorithm 20. The total
 1228 preimage attack on 10-round Gimli-XOF-128 is about $2^{128 - \min\{\text{DoF}^+, \text{DoF}^-, \text{DoM}\}} =$
 1229 2^{125} time and 2^{64} memory.

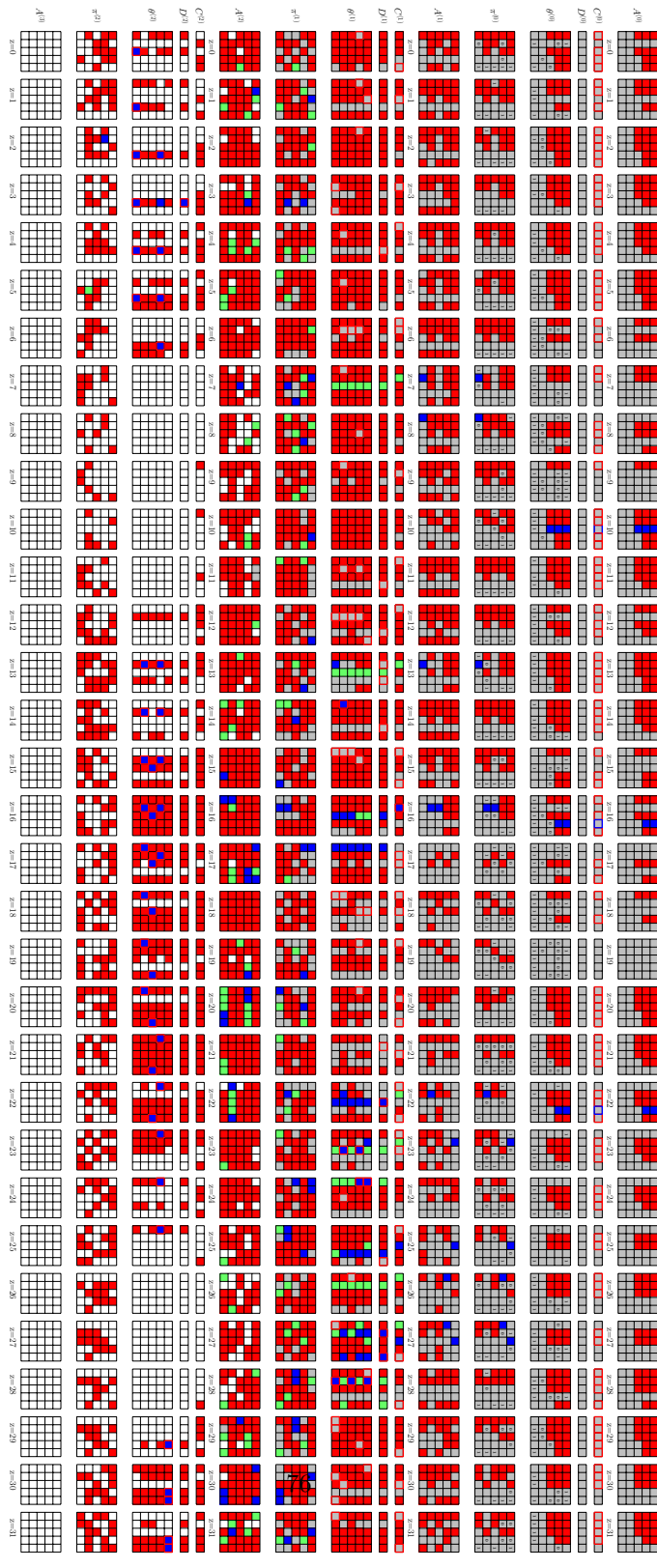


Fig. 30: Part-1: The 4-round Low Memory MITM preimage attack on KECCAK[768]

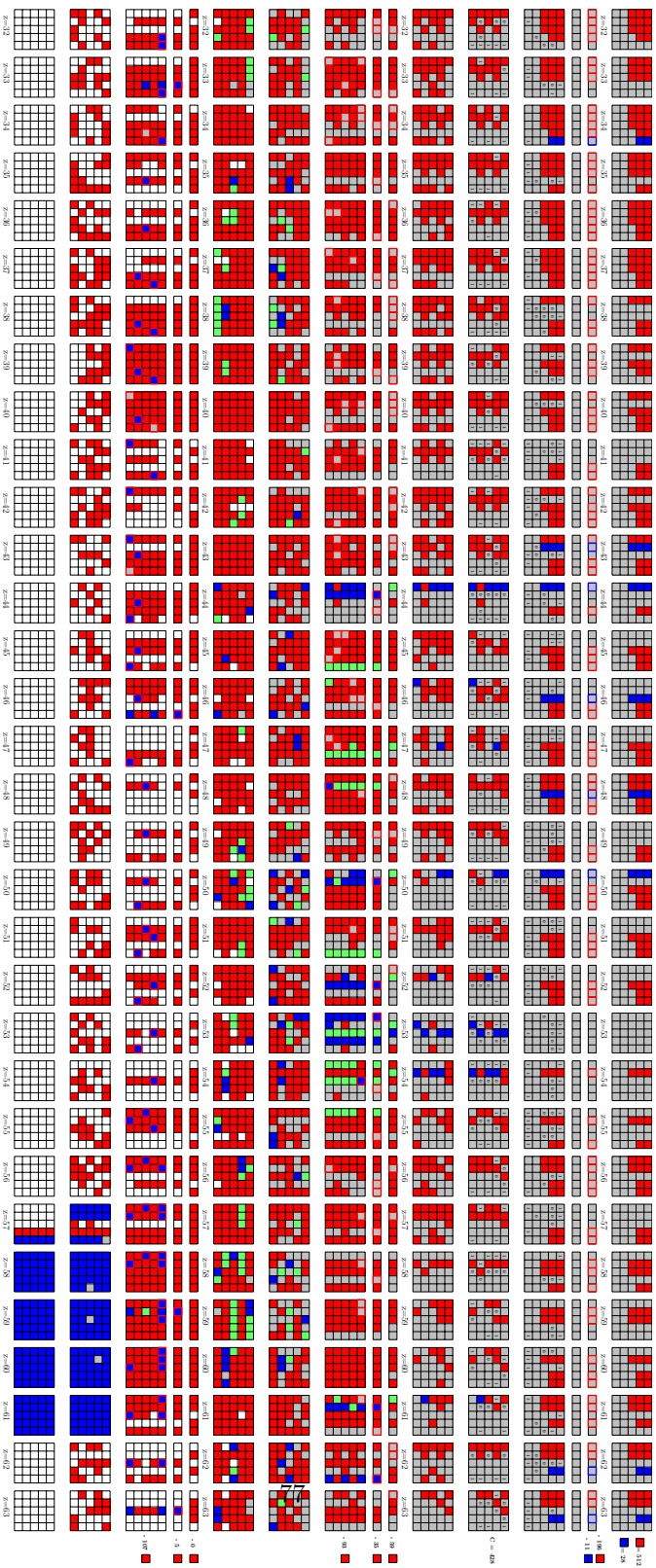


Fig. 31: Part-2: The 4-round Low Memory MitM preimage attack on KECCAK[768]

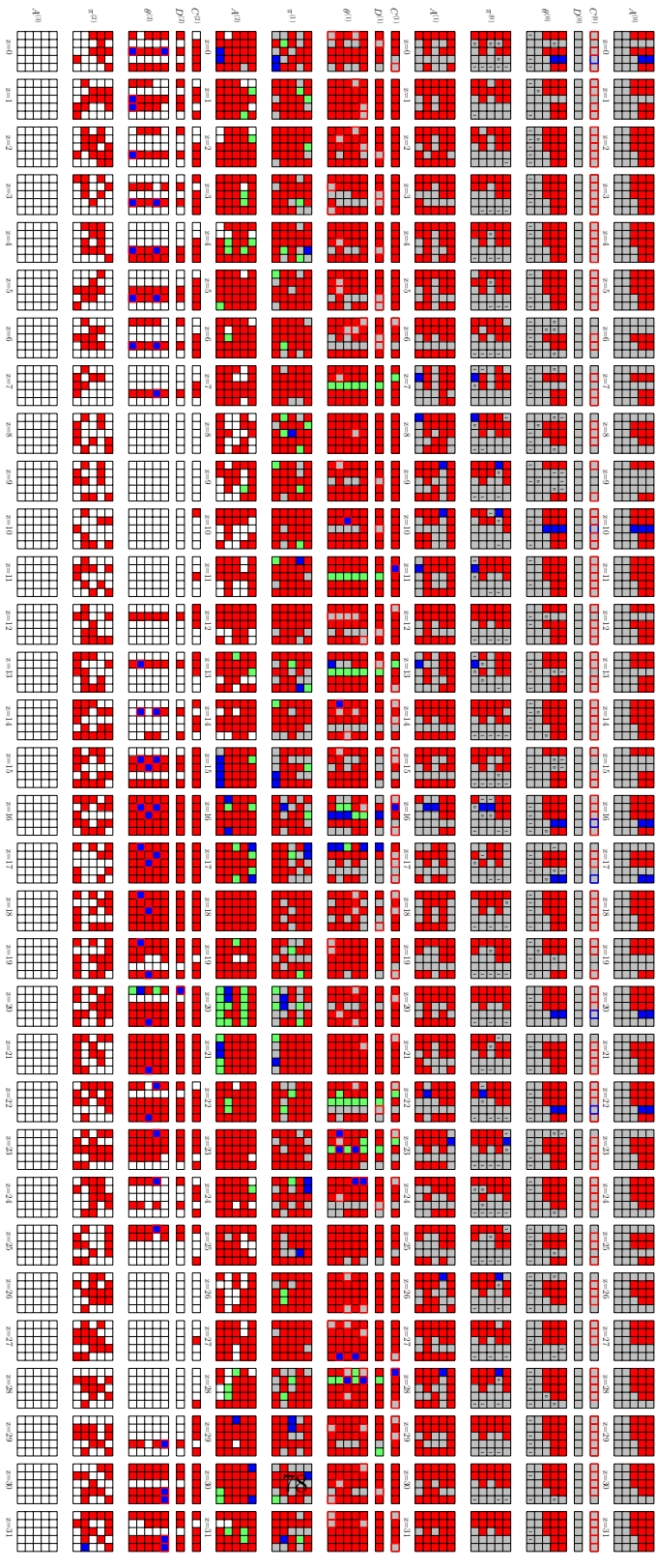


Fig. 32: Part-1: The 4-round MITM preimage attack on KECCAK[768]

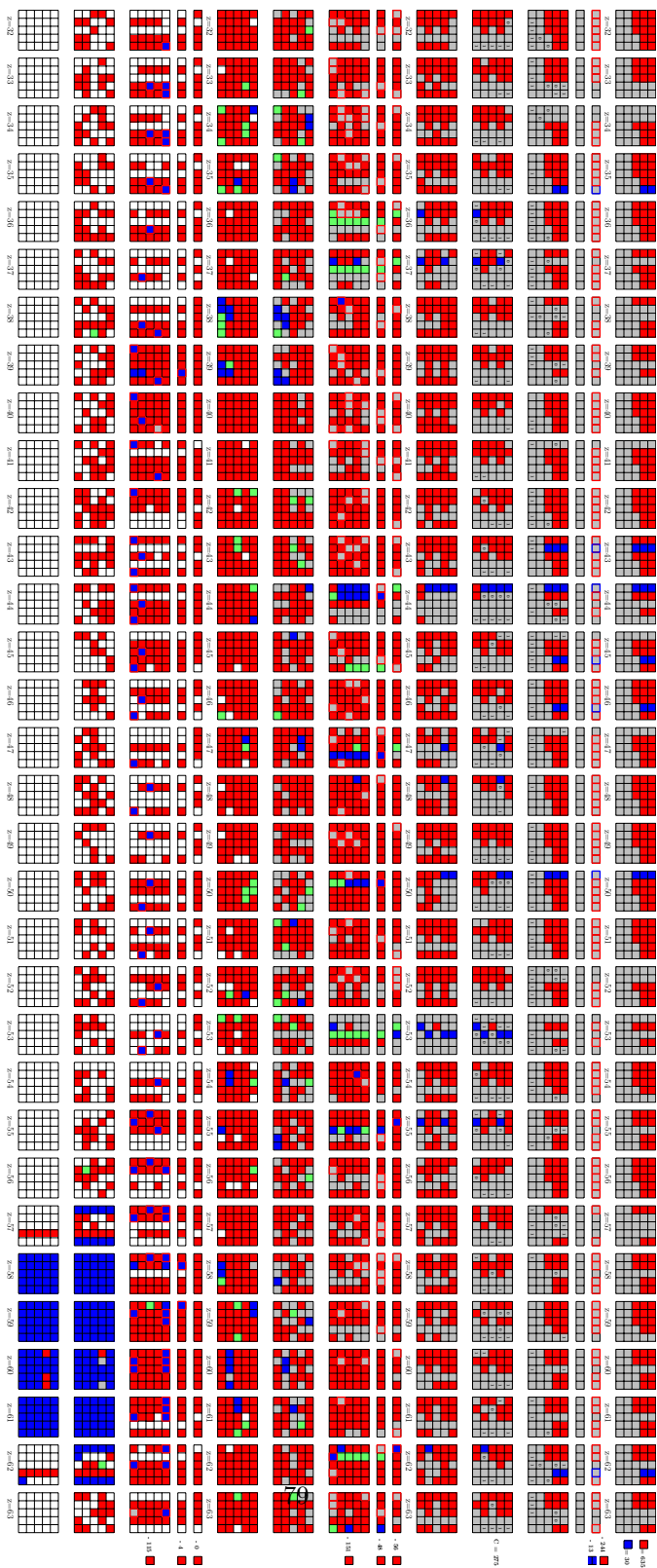


Fig. 33: Part-2: The 4-round MITM preimage attack on KECCAK[768]

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

```

1  Precompute inversely from the target to  $\theta^{(3)}$ 
2  for  $2^{\zeta_1}$  values of  $M_1$     /*  $\zeta_1 = 85$  */
3  do
4      Compute the inner part of the 2nd block and solve the system of 428
        linear equations
5      if the equations have solutions /* with probability of  $2^{-85}$  */
6      then
7          for each of the  $2^{\zeta_2}$  solutions of  $M_2$     /*  $\zeta_2 = 51 \leq 156$  */
8          do
9              for  $2^{\zeta_3}$  values  $(e_1, e_2, \dots, e_{159}) \in \mathbb{F}_2^{159}$  do
10                  $U \leftarrow []$ 
11                 for 157 free variables  $\blacksquare$  in  $A^{(0)} \in 2^{157}$  do
12                     Assign  $(e_1, e_2, \dots, e_{159})$  to 159 expressions
13                     Deduce 159 dependent variables
14                     Compute forward to other 140-bit  $\blacksquare/\blacksquare$   $u \in \mathbb{F}_2^{140}$ 
15                      $U[u] \leftarrow A^{(0)}[v_{\mathcal{R}}]$  /*  $v_{\mathcal{R}}$  represents all 316 red indexes
                        */
16                 end
17                 for  $u \in \mathbb{F}_2^{140}$  do
18                      $L \leftarrow []$ 
19                     for  $A^{(0)}[v_{\mathcal{B}}] \in 2^{17}$  /*  $v_{\mathcal{B}}$  represents all 17 blue
                        indexes */
20                     do
21                         Compute forward to the 17-bit matching point  $v$ 
22                          $L[v] \leftarrow A^{(0)}[v_{\mathcal{B}}]$ 
23                     end
24                     for values in  $U[u]$  do
25                         Compute forward the 17-bit matching point  $v'$ 
26                         for values in  $L[v']$  do
27                             if it leads to the given hash value then
28                                 Output the preimage
29                             end
30                         end
31                     end
32                 end
33             end
34         end
35     end
36 end

```

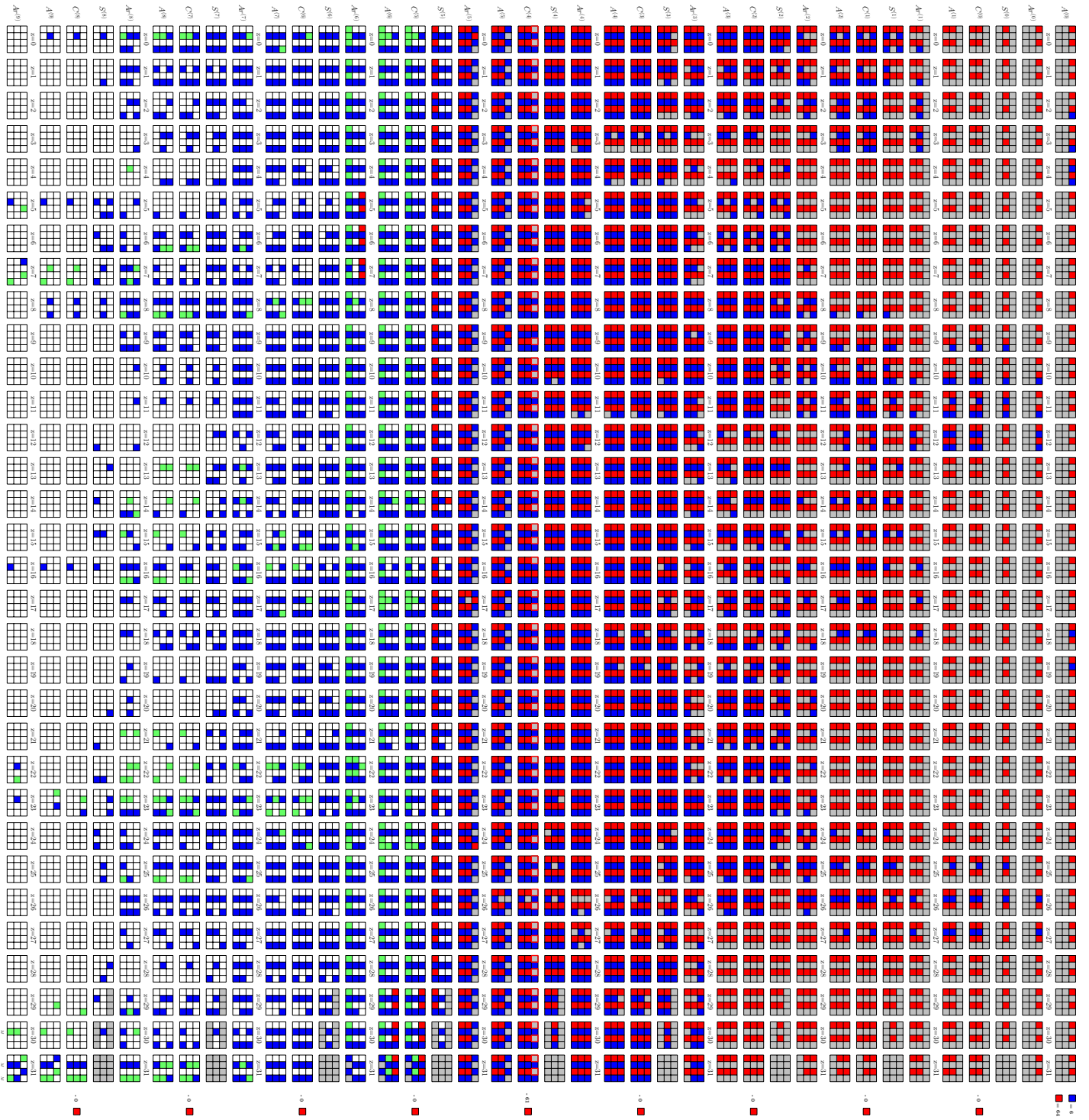


Fig. 34: The 10-round Preimage attack on Gimli-XOF-128

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

```

1 Precompute inversely from the target to  $A^{(9)}$ 
2 for  $2^{71}$  values of  $M_1$  /*  $\zeta_1 = 71$  */
3 do
4   Compute the inner part of the 2nd block and solve the system of 63 linear
   equations
5   if the equations have solutions /* with probability of  $2^{-63}$  */
6   then
7     for  $2^{50}$  values of the  $\blacksquare$  bits in  $M_2$  do
8        $U \leftarrow []$ 
9       for 64 free variables  $\blacksquare$  in  $A^{(0)} \in 2^{64}$  do
10        Compute forward to determine the 61-bit  $\blacksquare/\blacksquare$   $\mathbf{u} \in \mathbb{F}_2^{61}$ 
11         $U[\mathbf{u}] \leftarrow A^{(0)}[v_{\mathcal{R}}]$  /*  $v_{\mathcal{R}}$  represents all 64 red indexes */
12      end
13      for  $\mathbf{u} \in \mathbb{F}_2^{61}$  do
14         $L \leftarrow []$ 
15        for  $A^{(0)}[v_{\mathcal{B}}] \in 2^6$  /*  $v_{\mathcal{B}}$  represents all 6 blue indexes */
16        do
17          Compute forward to the 3-bit matching point  $v$ 
18           $L[v] \leftarrow A^{(0)}[v_{\mathcal{B}}]$ 
19        end
20        for values in  $U[\mathbf{u}]$  do
21          Compute forward the 3-bit matching point  $v'$ 
22          for values in  $L[v']$  do
23            if it leads to the given hash value then
24              Output the preimage
25            end
26          end
27        end
28      end
29    end
30  end
31 end

```

1230 K MITM Preimage Attack on Subterranean 2.0

1231 **Subterranean 2.0**, designed by Daemen et al. [16], is a second round candidate
 1232 of NIST LWC. For **Subterranean-XOF** with 256-bit digest ($n = 256$, $b = 257$,
 1233 $r = 9$, $r' = 32$, $c = 248$), Lefevre and Mennink proved a tight bound of 224-
 1234 bit preimage security [38] under ideal permutation model. The round function
 1235 consists of four operations: $\chi : s_i \leftarrow s_i + (s_{i+1} + 1)s_{i+2}$, ι : constant addition,
 1236 $\theta : s_i \leftarrow s_i + s_{i+3} + s_{i+8}$, and $\pi : s_i \leftarrow s_{12i}$. The internal state of each round
 1237 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 $z_i =$
 1238 $s_{12^{4i}} + s_{-12^{4i}}$, ($0 \leq i < 32$) as shown in Table 8, e.g., when $i = 0$, $z_0 = s_1 + s_{256}$.
 1239 After each 32-bit digest is squeezed out, 1-round function is executed to update
 1240 the internal state.

1241 Since DoFs are only consumed through the XOR operation, χ and θ oper-
 1242 ations can be represented by $\chi^* : s_i^* \leftarrow (s_{i+1} + 1)s_{i+2}$ and $\theta^* : s_i \leftarrow s_i + s_i^* +$
 1243 $s_{i+3} + s_{i+3}^* + s_{i+8} + s_{i+8}^*$. Thus, the DoFs consumption can be only restricted in
 1244 the θ^* operation. Here, we show a preimage attack against **Subterranean-XOF**
 1245 as shown in Figure 35. The starting state starts at the internal state where the
 1246 first 32-bit digest T_1 is squeezed out, denoted as $A^{(0)}$. Since each bit of T_1 is
 1247 deduced by $s_{12^{4i}} + s_{-12^{4i}}$, the MitM attribute of $(s_{12^{4i}}, s_{-12^{4i}})$ in $A^{(0)}$ can be
 1248 set as $(\blacksquare, \blacksquare)$, $(\blacksquare, \blacksquare)$ or $(\blacksquare, \blacksquare)$ and is served as 1-bit degree of freedom. Therefore,
 1249 there are $\lambda^+ = 114 \blacksquare$ bits and $\lambda^- = 92 \blacksquare$ bits in $A^{(0)}$. In the forward computa-
 1250 tion path, the consumed degrees of freedom (DoFs) of \blacksquare and \blacksquare are $l^+ = 42$ and
 1251 $l^- = 20$ bits, such that $\text{DoF}^+ = 72$, $\text{DoF}^- = 72$. In the matching phase, 1-bit
 1252 matching point can be deduced if $(s_{12^{4i}}, s_{-12^{4i}})$ in $A^{(r)}$ has no unknown \square bit,
 1253 for $1 \leq r \leq 3$. Hence, the final matching points are counted as $\text{DoM} = 72$ bits
 1254 and are marked with “m” in Figure 35.

1255 The detailed algorithm is given in Algorithm 21. In Line 25 to Line 36,
 1256 2^{72} solutions of $A^{(0)}$ are left after the 72-bit matching. Hence, we need repeat
 1257 $2^{\zeta+3+21+19+19} = 2^{224-72-72}$ times, i.e., $\zeta = 18$.

- 1258 - In Line 2, $A_{16}^{(1)}$ is a \blacksquare bit that is derived by imposing constraint on \blacksquare bits.
 1259 However, $\left(\overline{A_{15}^{(1)}} \wedge A_{16}^{(1)}\right)$ is involved in the constraint imposed on the $A_{129}^{(2)}$ by
 1260 consuming 1-bit \blacksquare DoF. Hence, it was considered at the outer loop. Similarly,
 1261 $(A_{26}^{(1)}, A_{47}^{(1)})$ are two \blacksquare bits which are derived by imposing constraints on \blacksquare
 1262 bits, but $\left(\overline{A_{25}^{(1)}} \wedge A_{26}^{(1)}\right)$ and $\left(\overline{A_{46}^{(1)}} \wedge A_{47}^{(1)}\right)$ are involved in the constraints
 1263 imposed on $A_{66}^{(2)}$ and $A_{132}^{(2)}$ by consuming 2-bit \blacksquare DoFs.
- 1264 - In Line 4 to Line 11, using the TA algorithm, $A_{200}^{(0)}$ can be determined by
 1265 free variables $v_{\mathcal{R},1}$ and $A_{16}^{(1)}$ as Eq. (40). Then, the remaining 86 free \blacksquare
 1266 bits are exhausted trivially. The time complexity of constructing table V
 1267 is $2^{\zeta+3+5+86} = 2^{112}$. The memory complexity is $2^{5+86} = 2^{91}$. Compared
 1268 with the table-based method in [26], the memory cost is reduced by a factor

1269

of 2.

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

1270

- 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 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} .

1271

1272

1273

1274

1275

1276

1277

1278

1279

Once finding such a proper $A^{(0)}$, an inner collision on 248-bit capacity between $A^{(0)}$ 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**

$c_{\mathcal{B},1}$	$\theta_{49}^{(0)}, \theta_{50}^{(0)}, \theta_{51}^{(0)}, \theta_{54}^{(0)}, \theta_{55}^{(0)}, \theta_{56}^{(0)}, \theta_{172}^{(0)}, \theta_{174}^{(0)}, \theta_{175}^{(0)}, \theta_{176}^{(0)}, \theta_{177}^{(0)}, \theta_{179}^{(0)}, \theta_{181}^{(0)}, \theta_{183}^{(0)}, \theta_{21}^{(1)}, \theta_{22}^{(1)}, \theta_{39}^{(1)}, \theta_{58}^{(1)}, \theta_{86}^{(1)}, \theta_{146}^{(1)}, \theta_{147}^{(1)}, \theta_{150}^{(1)}, \theta_{255}^{(1)}$
$c_{\mathcal{B},2}$	$\theta_3^{(1)}, \theta_6^{(1)}, \theta_{10}^{(1)}, \theta_{51}^{(1)}, \theta_{62}^{(1)}, \theta_{66}^{(1)}, \theta_{69}^{(1)}, \theta_{74}^{(1)}, \theta_{90}^{(1)}, \theta_{110}^{(1)}, \theta_{111}^{(1)}, \theta_{129}^{(1)}, \theta_{138}^{(1)}, \theta_{159}^{(1)}, \theta_{170}^{(1)}, \theta_{183}^{(1)}, \theta_{218}^{(1)}, \theta_{219}^{(1)}, \theta_{239}^{(1)}$
$v_{\mathcal{B},1}$	$A_{56}^{(0)}, A_{58}^{(0)}, A_{62}^{(0)}, A_{69}^{(0)}, A_{71}^{(0)}, A_{72}^{(0)}, A_{74}^{(0)}, A_{75}^{(0)}, A_{77}^{(0)}, A_{80}^{(0)}, A_{82}^{(0)}, A_{83}^{(0)}, A_{84}^{(0)}, A_{85}^{(0)}, A_{86}^{(0)}, A_{87}^{(0)}, A_{88}^{(0)}, A_{169}^{(0)}, A_{89}^{(0)}, A_{91}^{(0)}, A_{92}^{(0)}, A_{165}^{(0)}, A_{93}^{(0)}, A_{94}^{(0)}, A_{96}^{(0)}, A_{97}^{(0)}, A_{98}^{(0)}, A_{99}^{(0)}, A_{101}^{(0)}, A_{102}^{(0)}, A_{103}^{(0)}, A_{104}^{(0)}, A_{105}^{(0)}, A_{106}^{(0)}, A_{107}^{(0)}, A_{108}^{(0)}, A_{109}^{(0)}, A_{110}^{(0)}, A_{111}^{(0)}, A_{146}^{(0)}, A_{112}^{(0)}, A_{113}^{(0)}, A_{114}^{(0)}, A_{115}^{(0)}, A_{116}^{(0)}, A_{117}^{(0)}, A_{140}^{(0)}, A_{118}^{(0)}, A_{119}^{(0)}, A_{120}^{(0)}, A_{137}^{(0)}, A_{121}^{(0)}, A_{136}^{(0)}, A_{122}^{(0)}, A_{123}^{(0)}, A_{134}^{(0)}, A_{124}^{(0)}, A_{125}^{(0)}, A_{127}^{(0)}, A_{129}^{(0)}, A_{128}^{(0)}, A_{130}^{(0)}, A_{131}^{(0)}, A_{132}^{(0)}, A_{135}^{(0)}, A_{173}^{(0)}, A_{178}^{(0)}, A_{179}^{(0)}, A_{183}^{(0)}, A_{185}^{(0)}$
$v_{\mathcal{B},1}^*$	$A_{57}^{(0)}, A_{59}^{(0)}, A_{61}^{(0)}, A_{63}^{(0)}, A_{65}^{(0)}, A_{66}^{(0)}, A_{76}^{(0)}, A_{78}^{(0)}, A_{79}^{(0)}, A_{90}^{(0)}, A_{95}^{(0)}, A_{162}^{(0)}, A_{100}^{(0)}, A_{126}^{(0)}, A_{133}^{(0)}, A_{172}^{(0)}, A_{174}^{(0)}, A_{175}^{(0)}, A_{176}^{(0)}, A_{81}^{(0)}, A_{177}^{(0)}, A_{180}^{(0)}, A_{181}^{(0)}, A_{182}^{(0)}, A_{184}^{(0)}, A_{73}^{(0)}$
$v_{\mathcal{B},2}$	$A_{138}^{(0)}, A_{139}^{(0)}, A_{141}^{(0)}, A_{142}^{(0)}, A_{143}^{(0)}, A_{144}^{(0)}, A_{145}^{(0)}, A_{147}^{(0)}, A_{148}^{(0)}, A_{149}^{(0)}, A_{150}^{(0)}, A_{151}^{(0)}, A_{152}^{(0)}, A_{153}^{(0)}, A_{154}^{(0)}, A_{155}^{(0)}, A_{156}^{(0)}, A_{157}^{(0)}, A_{158}^{(0)}, A_{159}^{(0)}, A_{160}^{(0)}, A_{161}^{(0)}, A_{163}^{(0)}, A_{164}^{(0)}, A_{166}^{(0)}, A_{167}^{(0)}, A_{168}^{(0)}, A_{170}^{(0)}, A_{171}^{(0)}$

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

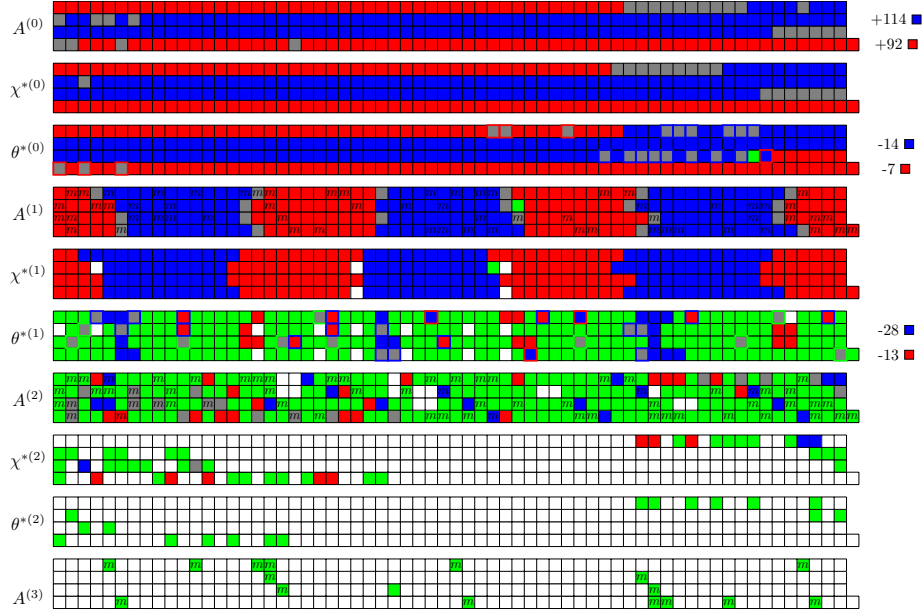


Fig. 35: The Full Round Preimage Attack on Subterranean-XOF

Algorithm 21: Preimage Attack on Full Round Subterranean

```

1 for  $2^\zeta$  possible values of the  $\blacksquare$  cells in  $A^{(0)}$ ,  $\zeta = 18$  do
2   for  $2^3$  possible values of  $A_{16}^{(1)} \| A_{26}^{(1)} \| A_{47}^{(1)}$  do
3      $U \leftarrow [], V \leftarrow []$ 
4     for 5-bit values of  $v_{\mathcal{R},1} = (A_{194}^{(0)}, A_{195}^{(0)}, A_{196}^{(0)}, A_{201}^{(0)}, A_{202}^{(0)})$  do
5       Deduce 1-bit value of  $A_{200}^{(0)}$  according to  $A_{16}^{(1)}$ 
6       for  $v_{\mathcal{R},2} \in \mathbb{F}_2^{86}$  of the remaining  $\blacksquare$  cells in  $A^{(0)}$  do
7         Compute forward to the 19-bit  $u$  with  $A_{26}^{(1)}$  and  $A_{47}^{(1)}$ 
8          $V[u] \leftarrow v_{\mathcal{R},1} \| v_{\mathcal{R},2} \| A_{200}^{(0)}$ 
9         /* There are  $2^{5+86-19} = 2^{72}$  elements under each index */
10      end
11    end
12    for  $g_{\mathcal{B}} \in \mathbb{F}_2^{21}$  of  $c_{\mathcal{B},1}$  /* except for  $\theta_{50}^{(0)} = A_{47}^{(1)}$  and  $\theta_{55}^{(0)} = A_{26}^{(1)}$  */
13      do
14        for 62-bit values of  $v_{\mathcal{B},1}$  do
15          Deduce 23-bit values of  $v_{\mathcal{B},1}^*$ 
16          for  $v_{\mathcal{B},2} \in \mathbb{F}_2^{29}$  of the remaining  $\blacksquare$  cells in  $A^{(0)}$  do
17            Compute forward to the 19-bit  $c_{\mathcal{B},2}$  with  $A_{16}^{(1)}$ 
18             $U[c_{\mathcal{B},2}] \leftarrow v_{\mathcal{B},1} \| v_{\mathcal{B},1}^* \| v_{\mathcal{B},2}$ 
19            /* There are  $2^{62+29-19} = 2^{72}$  elements under each index */
20          end
21        end
22        for  $u \in \mathbb{F}_2^{19}$  do
23          for  $c_{\mathcal{B},2} \in \mathbb{F}_2^{19}$  do
24             $L \leftarrow []$ 
25            for  $v_{\mathcal{R}} \in V[u]$  do
26              Compute forward to the 72-bit matching points  $End_{\mathcal{R}}$ 
27              and store  $v_{\mathcal{R}}$  in  $L$  with  $End_{\mathcal{R}}$  as index
28            end
29            for  $v_{\mathcal{B}} \in V[c_{\mathcal{B},2}]$  do
30              Compute forward to the 72-bit matching points  $End_{\mathcal{B}}$ 
31              for  $v_{\mathcal{R}} \in L[End_{\mathcal{B}}]$  do
32                Reconstruct the input state with  $v_{\mathcal{R}}$  and  $v_{\mathcal{B}}$ 
33                if it leads to the given 224-bit hash value
34                   $(T_2, T_3, \dots, T_8)$  then
35                  Output the preimage
36                end
37              end
38            end
39          end
40        end
41      end

```
