IEICE TRANS. FUNDAMENTALS, VOL.E107–A, NO.8 AUGUST 2024

PAPER

Accurate False-Positive Probability of Multiset-Based Demirci-Selçuk Meet-in-the-Middle Attacks

Dongjae LEE[†], *Nonmember*, **Deukjo HONG**[††a)], *Member*, **Jaechul SUNG**[†††], *and* **Seokhie HONG**[†], *Nonmembers*

SUMMARY In this study, we focus on evaluating the false-positive probability of the Demirci-Selçuk meet-in-the-middle attack, particularly within the context of configuring precomputed tables with multisets. During the attack, the adversary effectively reduces the size of the key space by filtering out the wrong keys, subsequently recovering the master key from the reduced key space. The false-positive probability is defined as the probability that a wrong key will pass through the filtering process. Due to its direct impact on the post-filtering key space size, the false-positive probability is an important factor that influences the complexity and feasibility of the attack. However, despite its significance, the false-positive probability of the multiset-based Demirci-Selçuk meet-in-the-middle attack has not been thoroughly discussed, to the best of our knowledge. We generalize the Demirci-Selçuk meet-in-the-middle attack and present a sophisticated method for accurately calculating the false-positive probability. We validate our methodology through toy experiments, demonstrating its high precision. Additionally, we propose a method to optimize an attack by determining the optimal format of precomputed data, which requires the precise false-positive probability. Applying our approach to previous attacks on `AES` and `ARIA`, we have achieved modest improvements. Specifically, we enhance the memory complexity and time complexity of the offline phase of previous attacks on 7-round `AES-128/192/256`, 7-round `ARIA-192/256`, and 8-round `ARIA-256` by factors ranging from $2^{0.56}$ to 2^{3}. Additionally, we have improved the overall time complexity of attacks on 7-round `ARIA-192/256` by factors of $2^{0.13}$ and $2^{0.42}$, respectively.

key words: *Demirci-Selçuk meet-in-the-middle attack, false-positive probability, optimization, AES, ARIA*

1. Introduction

The Demirci-Selçuk meet-in-the-middle ($\mathcal{DS}$-MITM) attack is one of the most powerful cryptanalysis techniques for numerous cryptographic primitives. Distinguished from conventional meet-in-the-middle attacks that partition target primitives into two independent parts, this technique derives its name from the two authors who first proposed the technique [12]. The $\mathcal{DS}$-MITM attack separates the target cipher into three independent parts, with the middle part serving as the distinguisher. The attack exploits the characteristic that a set of states called *δ-set* can be distinguished from a randomly chosen set of states even after several rounds of encryption.

The $\mathcal{DS}$-MITM attack is generally divided into offline and online phases, and has the following process: In the offline phase, the adversary stores the possible outcomes of the distinguisher in a precomputed table. In the online phase, the adversary selects suitable plaintexts, expecting them to fulfill the distinguisher's input condition after several rounds, and subsequently acquires corresponding ciphertexts. The adversary then partially decrypts these ciphertexts by guessing parts of the round keys and verifying the result against the precomputed table.

When the result is absent from the precomputed table, the guessed key is discarded, progressively reducing the key space. This judgment is based on the assumption that results from the wrong keys are randomly distributed. Therefore, there is a possibility that a result from the wrong key is included in the precomputed table. In this context, the false-positive probability, defined as the probability of a wrong key passing through the filtering process, influences the size of the post-filtering key space.

Following the filtering process, the adversary proceeds to recover the master key from the reduced key space. The method for this step relies on the characteristics of the target primitives. The adversary can either exhaustively search the reduced key space or employ a dedicated approach that exploits the weaknesses of the key schedule algorithm. As a result, the overall complexity of the attack is determined by the cost of filtering the wrong keys to reduce the key space and recovering the master key from the reduced key space.

The original $\mathcal{DS}$-MITM attack constructs a precomputed table with sequences [12]. In this scenario, the false-positive probability can be derived by calculating the ratio of the number of sequences stored in the precomputed table to the number of theoretically possible sequences. However, when the adversary transforms the sequences into multisets to reduce the memory complexity [8], the false-positive probability becomes significantly different from the simple ratio. This difference arises because the multisets are not uniformly distributed; in other words, the number of sequences associated with each multiset is different.

While this aspect has been mentioned in previous works [1], [6], [8], to the best of our knowledge, no such study has presented a precise method for calculating the false-positive probability of multiset-based $\mathcal{DS}$-MITM attacks. In [8], the authors calculated probabilities by assuming that all multisets follow a specific form based on a Poisson ap-

Manuscript received November 13, 2023.
Manuscript revised January 4, 2024.
Manuscript publicized March 15, 2024.

[†]Institute of Cyber Security & Privacy (ICSP), Korea University, Republic of Korea.
[††]Department of Computer Sciences and Artificial Intelligence, Jeonbuk National University, Republic of Korea.
[†††]Department of Mathematics, University of Seoul, Republic of Korea.
a) E-mail: deukjo.hong@jbnu.ac.kr (Corresponding author)

DOI: 10.1587/transfun.2023EAP1145

proximation[†]. Subsequently, [1], [6] employed the same underlying assumption when presenting their calculations of false-positive probabilities. However, our method reveals minor flaws in the probabilities presented by these works. Our study addresses the need for a precise method in the context of multiset-based $\mathcal{DS}$-MITM attacks. By providing an accurate methodology for calculating false-positive probabilities, we aim to enhance the accuracy and effectiveness of the $\mathcal{DS}$-MITM attacks.

1.1 Related Works

The idea of $\mathcal{DS}$-MITM attack started with the square property, which was first introduced in a proposal for SQUARE [7], the forerunner of AES. In the proposal of SQUARE, the designers presented attacks up to 6-round using the square property. In [15], Gilbert and Minier proposed a 4-round distinguisher by extending the square property and an attack on 7-round AES-192 and AES-256. Demirci and Selçuk extended the distinguisher to 5-round and presented an attack on 8-round AES-256 [12].

The cost of $\mathcal{DS}$-MITM attack proposed by Demirci and Selçuk is dominated by the memory complexity. Dunkelman et al. proposed a differential enumeration technique that can significantly reduce memory complexity [8]. They also found that memory complexity could be further reduced by storing precomputed data as a multiset rather than a sequence. Derbez et al. improved the attack by finding that the distinguisher proposed by Dunkelman et al. can be described using fewer parameters [6]. Li and Jin presented the 6-round distinguisher on AES-256 and the attack on 10-round AES-256 [18].

Many studies have applied the $\mathcal{DS}$-MITM attack technique to cryptographic primitives other than AES. It has been applied to several block ciphers, such as ARIA [1], PRINCE [10], TWINE [2], CLEFIA [19], Camellia [9], and Midori64 [20], and some Feistel constructions [13], [14]. There are also studies on the automatic search of distinguishers for use in $\mathcal{DS}$-MITM attacks [4], [5], [21]. Most recently, the quantum version of $\mathcal{DS}$-MITM attack has been studied [3], [16], and according to [3], $\mathcal{DS}$-MITM attack is the best cryptanalysis technique on AES using quantum computers.

1.2 Our Contributions

In this paper, our focus centers on the false-positive probability of the $\mathcal{DS}$-MITM attack when precomputed tables are configured with multisets. Our most significant contribution is a new methodology for determining an accurate value of the false-positive probability. Additionally, we present optimization methods that leverage the accurate false-positive probability values and apply these optimization methods to previous attacks on AES and ARIA, achieving modest improvements in complexity. More details about our contributions are as follows.

(1) Accurate calculation of the false-positive probability.

For the first time, we present a method to accurately calculate the false-positive probability of multiset-based $\mathcal{DS}$-MITM attacks. While precise probabilities can be obtained through an exhaustive examination of each element in the precomputed table, this approach is computationally infeasible due to the substantial precomputed table size. (e.g., 2^{80} for [6] and 2^{128} for [1].)

Our approach departs from explicitly considering individual elements within the precomputed table. Instead, we assume that these elements follow a specific probability distribution, as outlined in Assumptions 1 and 2. Building upon these assumptions, we calculate the conditional expectation value of the false-positive probability given the size of the precomputed table. To obtain this value, we derive some expressions and present a method for its estimation using dynamic programming technique (refer to Lemma 2, Theorem 1, and Algorithm 1).

To validate the accuracy of our method, we perform toy experiments. Specifically, we sample 2^{20} sequences that can be the outcome of the distinguisher presented in [6] and [1], respectively. We then transform the sequences into multisets and compare the values calculated using exhaustive examination and our approach. The results, presented in Table 5, demonstrate the high accuracy of our method. Additionally, through our approach, we identify a discrepancy of 2^9 in the false-positive probability presented in [1], [6], [8][††].

(2) Optimization method for $\mathcal{DS}$-MITM attack and its application to previous attacks on AES and ARIA.

We introduce methods to optimize the attacks by modifying the format of the precomputed data. Specifically, we can either reduce the number of elements or truncate their ranges. Subsequently, we delve into the effects of these format modifications on memory, time, and data complexity. In this context, obtaining an accurate false-positive probability becomes crucial. By comparing the complexities associated with each format, we can determine the optimal format for the precomputed data and thereby optimize the attacks.

Applying our optimization methods to the previous $\mathcal{DS}$-MITM attacks on 7-round AES-128/192/256 [6], 7-round ARIA-192/256, and 8-round ARIA-256 [1], we identify the optimal format for each attack and achieve modest improvements in complexity. As a result, we enhance the time complexity or memory complexity of the attacks by factors ranging from $2^{0.13}$ to 2^3. A comparison of the previous attacks and our optimized attacks is summarized in Table 1.

1.3 Paper Outline

Section 2 provides preliminaries necessary for understanding this paper. We introduce the notations and symbols

[†]We provide a comprehensive discussion of this assumption in Sect. 4.2.

[††]We remark that despite the presence of these flawed false-positive probabilities in [1], [6], [8], the corresponding attacks remain effective.

Table 1 Comparison of previous attacks with our optimized attacks.

Version	Rounds	Memory	Time			Data (CP)	Reference
			Overall	Offline	Online		
AES-all	7	2^{82}	2^{113}	$2^{85.5}$	2^{113}	2^{113}	[6]
	7	$2^{80.15}$	2^{113}	$2^{82.5}$	2^{113}	2^{113}	This paper
	7	2^{90}	2^{105} ⊛	$2^{93.5}$	2^{105} ⊛	2^{105}	[6]
	7	$2^{88.81}$	2^{105}	$2^{93.5}$	2^{105}	2^{105}	This paper
ARIA-192	7	2^{130}	$2^{135.1}$	$2^{134.1}$	$2^{134.2}$	2^{113}	[1]
	7	$2^{129.44}$	$2^{134.97}$	$2^{133.1}$	$2^{134.2}$	2^{113}	This paper
ARIA-256	7	2^{130}	$2^{136.1}$	$2^{136.1}$	$2^{127.1}$	2^{113}	[1]
	7	$2^{129.44}$	$2^{135.68}$	$2^{135.1}$	$2^{134.1}$	2^{113}	This paper
	7	$2^{129.44}$	$2^{136.1}$	$2^{136.1}$	$2^{127.1}$	2^{113}	This paper
	8	2^{138}	$2^{245.9}$	2^{142}	$2^{245.9}$	2^{113}	[1]
	8	$2^{136.81}$	$2^{245.9}$	2^{142}	$2^{245.9}$	2^{113}	This paper

CP refers to the chosen plaintext.

Time complexity is measured in encryption units.

Memory complexity is measured in 128-bit blocks.

⊛ We revise the time complexity to 2^{105}, which includes the encryption of 2^{105} data.

used throughout this work, and we briefly describe AES and ARIA. In Sect. 3, we present a generalized framework for the $\mathcal{DS}$-MITM attack. Section 4 presents a method for accurately calculating the false-positive probability of the $\mathcal{DS}$-MITM attack. In Sect. 5, we present optimization methods that require an accurate false-positive probability and apply these methods to previous works on AES and ARIA. We improve the previous $\mathcal{DS}$-MITM attacks on 7-round AES-128/192/256, 7-round ARIA-192/256, and 8-round ARIA-256. Finally, Sect. 6 concludes the paper.

2. Preliminaries

In this section, we introduce the notations used throughout the paper and briefly describe the specification of AES and ARIA.

2.1 Notations

Definition 1 (δ-Set, [7]). *A δ-set for a byte-oriented cipher is a set of 2^8 states that are all different in 1 byte and are all equal in the remaining bytes.*

Definition 2 (b-δ-Set, [13]). *A b-δ-set is a set of 2^b states that are all different in b bits (active bits) and are all equal in the remaining bits (inactive bits).*

A b-δ-set generalizes the concept of a δ-set, with a δ-set being equivalent to an 8-δ-set. Since this study encompasses ciphers beyond the byte-oriented type, the b-δ-set notation is mainly utilized for explanation throughout the paper.

Definition 3 (Sequence). *A sequence is a finite ordered list of elements. Sequence s is called (u,v)-sequence if s has u elements and all elements are non-negative integers less than 2^v.*

Definition 4 (Multiset). *A multiset is a modification of the concept of a set that allows for multiple instances for each of its elements. Multiset m is called (u,v)-multiset if m has u elements and all elements are non-negative integers less than 2^v.*

We utilize parentheses to represent sequences, e.g., a (u,v)-*sequence* is denoted as $(e_1, e_2, \ldots, e_u)$, with each e_i being a non-negative integer less than 2^v. In the case of multisets, square brackets are used, e.g., a (u,v)-*multiset* is represented as $[e_1, e_2, \ldots, e_u]$, where each e_i is a non-negative integer less than 2^v.

Definition 5 (Set of Sequences and Multisets). *Let $\mathcal{S}_{u,v}$ be a set of all possible (u,v)- sequence and $\mathcal{M}_{u,v}$ be a set of all possible (u,v)-multiset.*

The size of $\mathcal{S}_{u,v}$ is 2^{uv}, while the size of $\mathcal{M}_{u,v}$ is $\binom{u+2^v-1}{u}$. Since $|\mathcal{M}_{u,v}| < |\mathcal{S}_{u,v}|$, (u,v)-*multiset* require fewer bits for representation than a (u,v)-*sequence*.

Definition 6. *Let $\theta_{u,v}$ be a function that maps a sequence to a multiset defined by*

$$\theta_{u,v} : \mathcal{S}_{u,v} \to \mathcal{M}_{u,v}$$
$$(a_1, \ldots, a_u) \mapsto [a_1, \ldots, a_u].$$

Definition 7 (Tables Based on Sequences and Multisets). *$\mathcal{T}^{(u,v)\text{-}seq}$ is referred to as a (precomputed) table based on (u,v)-sequences if*

$$\mathcal{T}^{(u,v)\text{-}seq} \subset \mathcal{S}_{u,v}.$$

Similarly, $\mathcal{T}^{(u,v)\text{-}mul}$ is a (precomputed) table based on (u,v)-multisets if

$$\mathcal{T}^{(u,v)\text{-}mul} \subset \mathcal{M}_{u,v}.$$

2.2 Brief Description of AES

AES has been the most widely used block cipher since it was

selected as an encryption standard at an open competition organized by NIST in 2000 [11]. It adopts a substitution-permutation network (SPN) structure, and its round function consists of four operations: SubBytes (SB), ShiftRows (SR), MixColumns (MC), and AddRoundKey (ARK). AES has three modes: AES-128, AES-192, and AES-256. The number of rounds of each mode is 10, 12, and 14, and the key size is 128, 192, and 256-bit. The MixColumns operation in the last round is omitted. The 128-bit internal state X is treated as a 4×4 byte matrix, where each byte represents a value in $GF(2^8)$:

$$X = \begin{pmatrix} a_0 & a_4 & a_8 & a_{12} \\ a_1 & a_5 & a_9 & a_{13} \\ a_2 & a_6 & a_{10} & a_{14} \\ a_3 & a_7 & a_{11} & a_{15} \end{pmatrix}.$$

We denote i-th byte of internal state X as $X[i] = a_i$. A concise overview of the four operations is provided below. For more comprehensive information, please refer to [11].

- SubBytes: The S-box S is applied on each byte of the state as follows:

$$\begin{pmatrix} a_0 & a_4 & a_8 & a_{12} \\ a_1 & a_5 & a_9 & a_{13} \\ a_2 & a_6 & a_{10} & a_{14} \\ a_3 & a_7 & a_{11} & a_{15} \end{pmatrix} \rightarrow \begin{pmatrix} S(a_0) & S(a_4) & S(a_8) & S(a_{12}) \\ S(a_1) & S(a_5) & S(a_9) & S(a_{13}) \\ S(a_2) & S(a_6) & S(a_{10}) & S(a_{14}) \\ S(a_3) & S(a_7) & S(a_{11}) & S(a_{15}) \end{pmatrix}.$$

- ShiftRows: The state is permuted as follows:

$$\begin{pmatrix} a_0 & a_4 & a_8 & a_{12} \\ a_1 & a_5 & a_9 & a_{13} \\ a_2 & a_6 & a_{10} & a_{14} \\ a_3 & a_7 & a_{11} & a_{15} \end{pmatrix} \rightarrow \begin{pmatrix} a_0 & a_4 & a_8 & a_{12} \\ a_5 & a_9 & a_{13} & a_1 \\ a_{10} & a_{14} & a_2 & a_6 \\ a_{15} & a_3 & a_7 & a_{11} \end{pmatrix}.$$

- MixColumns: Multiply each column of the state by a matrix M_{MC}. M_{MC} is defined as follows:

$$M_{\mathsf{MC}} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix}.$$

- AddroundKey: The 128-bit round key is XORed to a state.

2.3 Brief Description of ARIA

ARIA is a block cipher proposed in ICISC 2003 by Kwon et al. and has been the Korean encryption standard since 2004 [17]. It adopts an SPN structure and its round function consists of three operations: Substitution Layer (SL), Diffusion Layer (DL), and AddRoundKey (ARK). ARIA has three modes: ARIA-128, ARIA-192, and ARIA-256. The number of rounds of each mode is 10, 12, and 14, and the key size is 128, 192, and 256-bit. The Diffusion Layer operation of the last round is omitted. The 128-bit internal state is treated as a 16×1 byte matrix, where each byte represents a value in $GF(2^8)$:[†]

$$(a_0 \ a_1 \ a_2 \ a_3 \ \ldots \ a_{14} \ a_{15})^{\mathsf{T}}.$$

A concise overview of the three operations is provided below. For more comprehensive information, please refer to [17].

- Substitution Layer: Two S-boxes, S_1 and S_2, along with their inverses S_1^{-1} and S_2^{-1}, are utilized. For odd rounds, from a_0 to a_{15}, the following S-boxes are applied:

$$S_1, \ S_2, \ S_1^{-1}, \ S_2^{-1}, \ S_1, \ \ldots, \ S_2^{-1}, \ S_1, \ S_2, \ S_1^{-1}, \ S_2^{-1}.$$

  For even rounds, from a_0 to a_{15}, the following S-boxes are applied:

$$S_1^{-1}, \ S_2^{-1}, \ S_1, \ S_2, \ S_1^{-1}, \ \ldots, S_2, \ S_1^{-1}, \ S_2^{-1}, \ S_1, \ S_2.$$

- Diffusion Layer: Multiply the state by a matrix M_{DL}. M_{DL} is defined as follows:

$$M_{\mathsf{DL}} = \begin{pmatrix} 0&0&0&1&1&0&1&0&1&1&0&0&0&1&1&0 \\ 0&0&1&0&0&1&0&1&1&1&0&0&1&0&0&1 \\ 0&1&0&0&1&0&1&0&0&0&1&1&1&0&0&1 \\ 1&0&0&0&0&1&0&1&0&0&1&1&0&1&1&0 \\ 1&0&1&0&0&1&0&0&1&0&0&1&0&0&1&1 \\ 0&1&0&1&1&0&0&0&0&1&1&0&0&0&1&1 \\ 1&0&1&0&0&0&0&1&0&1&1&0&1&1&0&0 \\ 0&1&0&1&0&0&1&0&1&0&0&1&1&1&0&0 \\ 1&1&0&0&1&0&0&1&0&0&1&0&0&1&0&1 \\ 1&1&0&0&0&1&1&0&0&0&0&1&1&0&1&0 \\ 0&0&1&1&0&1&1&0&1&0&0&0&0&1&0&1 \\ 0&0&1&1&1&0&0&1&0&1&0&0&1&0&1&0 \\ 0&1&1&0&0&0&1&1&0&1&0&1&1&0&0&0 \\ 1&0&0&1&0&0&1&1&1&0&1&0&0&1&0&0 \\ 1&0&0&1&1&1&0&0&0&1&0&1&0&0&1&0 \\ 0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&1 \end{pmatrix}.$$

- AddRoundKey: The 128-bit round key is XORed to a state.

3. The Framework for $\mathcal{DS}$-MITM Attack

In this section, we present the framework for the $\mathcal{DS}$-MITM attack. To begin, we define the notations necessary to describe the framework. The notations introduced in this section are summarized in Table 2 and illustrated in Fig. 1 along with an example. We denote the target cipher as E, consisting of R rounds. The block size and key size of the target cipher are denoted by n and k, respectively. We partition the target cipher into three components: E_1, E_2, and E_3, such that $E = E_3 \circ E_2 \circ E_1$, with E_1, E_2, and E_3 consisting of r_1, r_2, and r_3 rounds, respectively. Notably, E_2 corresponds to the distinguisher, while E_1 and E_3 represent the parts preceding and succeeding the distinguisher, respectively.

[†]Note that figures depicting the truncated differential trails of ARIA use a 4×4 matrix representation for the state as AES. Refer to Figs. A· 3 and A· 4.

Table 2 Summary of notations for $\mathcal{DS}$-MITM attack framework.

E	Target cipher.
(R, k, n)	(number of rounds, key size, block size) of E.
E_1, E_2, E_3	$E = E_3 \circ E_2 \circ E_1$, E_2 corresponds to the distinguisher.
r_1, r_2, r_3	number of rounds of E_1, E_2, and E_3.
N	The number of the outputs of distinguisher can take.
$\mathcal{T}$	A precomputed table.
D	The number of bits required to store one element(multiset) in $\mathcal{T}$.
(pos_p, α_p), (pos_c, α_c), (pos_{st}, α_{st}), (pos_{ed}, α_{ed})	(position, number of bits) of active bits in the plaintext, ciphertext, and the start and end of the distinguisher.
p_f, p_b	The probability of transition of difference from pos_p to pos_{st} and from pos_c to pos_{ed}.
k_1, k_3	The number of partial key candidates for E_1 and E_3 considered in step 2 of the online phase are 2^{k_1} and 2^{k_3}, respectively.
T_{dist}	The cost of calculating a single element in the sequence or multiset during the offline phase.
T_{multi}	The cost of calculating a single element in the sequence or multiset during step 2 of the online phase.
T_{rem}	The cost of recovering the master key, given the partial keys for E_1 and E_3.
P_{fp}	The false-positive probability, representing the probability that a multiset constructed from a wrong guess is included in $\mathcal{T}$.

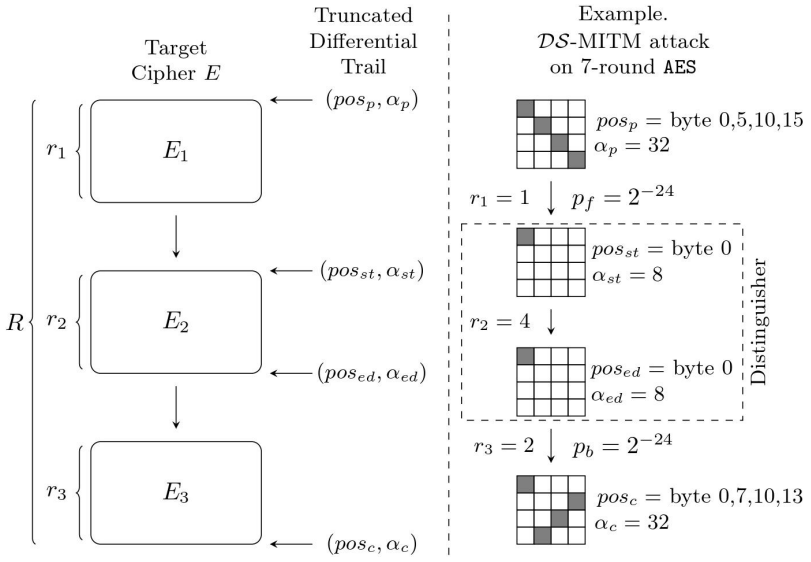

Fig. 1 Our framework (left) and an example of its application to the $\mathcal{DS}$-MITM attack on 7-round AES (right) [6]. The detailed truncated differential trail of the attack is shown in Fig. A· 1.

The $\mathcal{DS}$-MITM attack is built on a full-round truncated differential trail for the target cipher (after [8] introduced the differential enumeration technique). We generalize the truncated differential trail as follows. We denote the positions and number of active bits at the start and end of the distinguisher as pos_{st}, α_{st}, pos_{ed}, and α_{ed}. The differences pos_{st} and pos_{ed} naturally propagate in backward and forward directions, respectively. We denote the position and number of active bits in the plaintext and ciphertext as pos_p, α_p, pos_c, and α_c. We define the probability of transition from pos_p to pos_{st} as p_f and the probability of transition from pos_c to pos_{ed} as p_b. Building upon the defined notations, we generalize the distinguisher as stated in Proposition 1.

Proposition 1. *If a pair of messages $(\mathcal{P}, \mathcal{P}')$ conforms to the given truncated differential trail, then the sequence (or multiset) of differences between the distinguisher's outputs, obtained from the b-δ-set constructed from $\mathcal{P}$, can only take N values.*

The $\mathcal{DS}$-MITM attack consists of the offline phase and the online phase. The details of each phase are as follows:

(1) Offline Phase.

In the offline phase, we precompute all possible N sequences, transform them into multisets, and store the resulting multisets in table $\mathcal{T}$. We denote the memory required to store a single multiset as D-bit. Then, the memory complexity of the offline phase becomes $(N \cdot D)$-bit. The time complexity of the offline phase is $N \cdot 2^b \cdot T_{\mathsf{dist}}$, with T_{dist} denoting the cost of finding one element in each multiset.

(2) Online Phase.

The online phase is divided into three steps, and each step is as follows:

Step 1. In the first step, we find message pairs that conform to the given plaintext and ciphertext differential (pos_p and pos_c). We prepare a structure of 2^{α_p} plaintexts, where the bits of pos_p take all possible values, and the remaining bits are equal. In one structure, we can find $2^{\alpha_p}(2^{\alpha_p}-1)/2 \approx 2^{2\alpha_p-1}$ plaintext pairs. The probability that the corresponding ciphertext pair has active bits only at pos_c is $2^{-n+\alpha_c}$. Therefore, to obtain one pair that has active bits only at pos_p and pos_c, $2^{n-2\alpha_p-\alpha_c+1}$ structures are required on average. Considering the forward propagation from pos_p to pos_{st} and backward propagation from pos_c to pos_{ed}, to obtain a pair that conforms to the entire R-round differential trail, $2^{n-2\alpha_p-\alpha_c+1} \cdot (p_f p_b)^{-1}$ structures are required on average. Therefore, $2^{n-\alpha_p-\alpha_c+1} \cdot (p_f p_b)^{-1}$ chosen-plaintexts and $2^{n-\alpha_p-\alpha_c+1} \cdot (p_f p_b)^{-1}$ encryptions are required in the first step.

Step 2. After the first step, we have $(p_f p_b)^{-1}$ right pairs that conform to pos_p and pos_c. In the second step, from these pairs, we identify a pair that conforms to the entire R-round differential trail by utilizing the precomputed table $\mathcal{T}$. Let the number of suggestions for the partial key of E_1 be 2^{k_1}. We generate a set of plaintexts for every possible pair and the key suggestion that forms a b-δ-set after r_1 rounds. We obtain a set of corresponding ciphertexts, partially decrypt them using the partial key for E_3, and derive a multiset from the partially decrypted ciphertexts[†]. Let the number of suggestions for the partial key of E_3 be 2^{k_3}. We then check whether this multiset is listed in $\mathcal{T}$. If it is not listed, we discard this possibility, thereby progressively reducing the key space. The time complexity of step 2 is $(p_f p_b)^{-1} \cdot 2^{k_1+k_3} \cdot \left(2^b \cdot T_{\mathsf{multi}}\right)$, where T_{multi} represents the cost of calculating one element in the multiset, given the message pair and the partial keys for E_1 and E_3.

We construct $(p_f p_b)^{-1} \cdot 2^{k_1+k_3}$ multisets in step 2. Let the false-positive probability, $P_{F.P.}$, be a probability that a multiset constructed from the wrong pair or wrong partial key is in $\mathcal{T}$[††]. All the wrong multisets are expected to be filtered out if $(p_f p_b)^{-1} \cdot 2^{k_1+k_3} \cdot P_{F.P.} < 1$. Otherwise, $1+(p_f p_b)^{-1} \cdot 2^{k_1+k_3} \cdot P_{F.P.}$ multisets are expected to survive. This implies that $1+(p_f p_b)^{-1} \cdot 2^{k_1+k_3} \cdot P_{F.P.}$ suggestions for partial keys of E_1 and E_3 survive after step 2.

Step 3. In the final step, we recover the master key from the reduced key space. This step's generalization is challenging due to its reliance on difficult-to-generalize processes, such as key schedules. Let T_{rem} denote the cost of recovering the master key when the partial keys of E_1 and E_3 are uniquely determined in step 2. Since we anticipate that $1 + (p_f p_b)^{-1} \cdot 2^{k_1+k_3} \cdot P_{F.P.}$ suggestions for the partial key will survive, the time complexity of step 3 becomes $(1 + (p_f p_b)^{-1} \cdot 2^{k_1+k_3} \cdot P_{F.P.}) \cdot T_{\mathsf{rem}}$.

Consequently, the overall complexity of our framework for $\mathcal{DS}$-MITM attacks is as follows:

$$\begin{aligned}
&\text{Memory Complexity} = N \cdot D \text{ bits},\\
&\text{Time Complexity} = T_{\mathsf{off}} + T_{\mathsf{on}},\\
&\begin{cases}
T_{\mathsf{off}} = N \cdot 2^b \cdot T_{\mathsf{dist}},\\
T_{\mathsf{on}} = T_1 + T_2 + T_3,\\
\quad \begin{cases}
T_1 = (p_f p_b)^{-1} \cdot 2^{n-\alpha_p-\alpha_c+1},\\
T_2 = (p_f p_b)^{-1} \cdot 2^{k_1+k_3} \cdot \left(2^b \cdot T_{\mathsf{multi}}\right),\\
T_3 = (1 + (p_f p_b)^{-1} \cdot 2^{k_1+k_3} \cdot P_{F.P.}) \cdot T_{\mathsf{rem}},
\end{cases}
\end{cases}\\
&\text{Data Complexity} : (p_f p_b)^{-1} \cdot 2^{n-\alpha_p-\alpha_c+1}.
\end{aligned}$$

4. Accurate Calculation of False Positive Probability

In this section, we present a method for obtaining the accurate false-positive probability and validate our method through toy experiments. Furthermore, our methodology reveals flaws in the false-positive probability presented in previous works [1], [6], [8].

We rely on Assumption 1 as a fundamental requirement to establish the definition of the false-positive probability.

Assumption 1. *Let s be a sequence obtained from a wrong guess (wrong pair or wrong partial key), with the format of s being a (u,v)-sequence. We assume that s is uniformly distributed over $\mathcal{S}_{u,v}$.*

Based on Assumption 1, the false-positive probability is defined differently depending on whether the precomputed table is configured with sequences or multisets. If the table is configured with sequences, it is the probability that a sequence obtained from a wrong guess is included in the table. Alternatively, when the table is configured with multisets, the false-positive probability refers to the probability that a sequence obtained from a wrong guess is associated with one of the multisets in the table. Formally, the false-positive probability is defined as stated in Definition 8.

Definition 8. *The false-positive probabilities $P_{\mathsf{fp}}^{\mathcal{T}^{(u,v)\text{-}seq}}$ and $P_{\mathsf{fp}}^{\mathcal{T}^{(u,v)\text{-}mul}}$ for each case, where the precomputed table (denoted as $\mathcal{T}^{(u,v)\text{-}seq}$ and $\mathcal{T}^{(u,v)\text{-}mul}$, respectively) is composed of (u,v)-sequence and (u,v)-multiset are defined as follows:*

$$P_{\mathsf{fp}}^{\mathcal{T}^{(u,v)\text{-}seq}} = \frac{|\mathcal{T}^{(u,v)\text{-}seq}|}{|\mathcal{S}_{u,v}|}$$

$$P_{\mathsf{fp}}^{\mathcal{T}^{(u,v)\text{-}mul}} = \frac{|\{s : s \in \mathcal{S}_{u,v}, \theta_{u,v}(s) \in \mathcal{T}^{(u,v)\text{-}mul}\}|}{|\mathcal{S}_{u,v}|}$$

We can determine the precise false-positive probability by counting and aggregating the number of sequences

[†]Note that all plaintexts are already included in the structure considered in step 1. Therefore, obtaining the corresponding ciphertexts does not increase the data complexity.

[††]Formal definition for the false-positive probability is discussed in Sect. 4.

associated with each multiset in $\mathcal{T}^{(u,v)\text{-}mul}$, as defined in Definition 8. However, this approach is infeasible due to the significantly large size of the precomputed table. Our method is designed to handle scenarios where exact calculations are computationally prohibitive. We propose an alternative approach to estimate the conditional expected value of the false-positive probability given that the size of the table is N, i.e., $\mathrm{E}\left[P_{\mathsf{fp}}^{\mathcal{T}^{(u,v)\text{-}mul}} \middle| |\mathcal{T}^{(u,v)\text{-}mul}| = N\right]$.

It is important to note that our approach does not explicitly consider the specific elements present in the precomputed table. Instead, we assume that these elements follow a certain probability distribution, as outlined in Assumption 2.

Assumption 2. *Let $\mathcal{T}^{(u,v)\text{-}seq}$ be a precomputed table generated in the offline phase where the elements are stored in (u,v)-sequence format. We assume that all elements of $T^{(u,v)\text{-}seq}$ are independent and uniformly distributed over $\mathcal{S}_{u,v}$*

First, for each (u,v)-*multiset* m, we define $p(m)$ as the probability that a randomly chosen (u,v)-*sequence* is associated with m, as provided in Definition 9.

Definition 9. *Given (u,v)-multiset m, the probability of m, $p(m)$, is defined as*

$$\begin{aligned} p(m) &= \Pr\left[\theta_{u,v}(s) = m, s \in \mathcal{S}_{u,v}\right] \\ &= \frac{|\{s : s \in \mathcal{S}_{u,v}, \theta_{u,v}(s) = m\}|}{|\mathcal{S}_{u,v}|} \end{aligned}$$

Now, we introduce Lemma 1, which focuses on the probability that a (u,v)-*multiset* m is included in the precomputed table given the size of the table is N.

Lemma 1. *For a given (u,v)-multiset m, the probability that m is included in $\mathcal{T}^{(u,v)\text{-}mul}$, which has a size of N, is as follows:*

$$\Pr\left[m \in \mathcal{T}^{(u,v)\text{-}mul} \middle| |\mathcal{T}^{(u,v)\text{-}mul}| = N\right] = 1 - (1 - p(m))^N.$$

Proof. Let $\mathcal{T}^{(u,v)\text{-}seq} = \{s_1, s_2, \cdots, s_N\}$ be a precomputed table based on (u,v)-*sequences*, which corresponds to $\mathcal{T}^{(u,v)\text{-}mul}$. Then,

$$\begin{aligned} &\Pr\left[m \in \mathcal{T}^{(u,v)\text{-}mul} \middle| |\mathcal{T}^{(u,v)\text{-}mul}| = N\right] \\ &= 1 - \Pr\left[m \notin \mathcal{T}^{(u,v)\text{-}mul} \middle| |\mathcal{T}^{(u,v)\text{-}mul}| = N\right] \\ &= 1 - \prod_{i=1}^{N} \Pr\left[\theta_{u,v}(s_i) \neq m\right] \\ &\qquad\qquad (\because \text{all } s_i \text{ are independent}) \\ &= 1 - \prod_{i=1}^{N} \{1 - p(m)\} \\ &= 1 - (1 - p(m))^N \end{aligned}$$

□

Building upon Lemma 1, we introduce Lemma 2, which focuses on the conditional expected false-positive probability given the size of the precomputed table.

Lemma 2. *Given the size of the precomputed table N, the following equation holds:*

$$\mathrm{E}\left[P_{F.P.}^{\mathcal{T}^{(u,v)\text{-}mul}} \middle| |\mathcal{T}^{(u,v)\text{-}mul}| = N\right] = \sum_{i=1}^{N} \left\{ \sum_{m \in \mathcal{M}_{u,v}} (-1)^{i+1} \binom{N}{i} p(m)^{i+1} \right\}.$$

The proof of Lemma 2 is presented in Fig. 2. From Lemma 2, we define $Approx[x]$ which represents the first x terms. Specifically, for $x \geq 1$

$$Approx[x] = \sum_{i=1}^{x} \left\{ \sum_{m \in \mathcal{M}_{u,v}} (-1)^{i+1} \binom{N}{i} p(m)^{i+1} \right\}. \quad (1)$$

For simplicity, we define Δ_x as:

$$\Delta_x = (-1)^{x+1} \cdot \binom{N}{x} \cdot \left\{ \sum_{m \in \mathcal{M}_{u,v}} p(m)^{x+1} \right\} \quad \text{for } x \geq 1. \quad (3)$$

Then, we can rewrite Eq. (1) as follows:

$$Approx[x] = \sum_{i=1}^{x} \Delta_i.$$

Lemma 3 and Theorem 1 demonstrate that when x is even, $Approx[x]$ serves as the lower bound for $\mathrm{E}\left[P_{F.P.}^{\mathcal{T}^{(u,v)\text{-}mul}} \middle| |\mathcal{T}^{(u,v)\text{-}mul}| = N\right]$, and when x is odd, it functions as the upper bound.

Lemma 3. *Given two positive integers m and n such that $m \leq n$, and a real number x such that $0 \leq x \leq 1$,*

$$(1-x)^n \leq \sum_{i=0}^{m} (-1)^i \binom{n}{i} x^i, \text{ if } m \text{ is even,}$$

$$(1-x)^n \geq \sum_{i=0}^{m} (-1)^i \binom{n}{i} x^i, \text{ if } m \text{ is odd.}$$

Proof. We prove the lemma using mathematical induction. If m is even, it is true for $n = m$ because both sides are equal. Suppose it is true for $n = k$. Then, for $n = k + 1$,

$$\begin{aligned} (1-x)^{k+1} &= (1-x)^k (1-x) \\ &\leq \left(\sum_{i=0}^{m} (-1)^i \binom{k}{i} x^i \right) (1-x) \\ &= \sum_{i=0}^{m} \left((-1)^i \binom{k}{i} x^i + (-1)^{i+1} \binom{k}{i} x^{i+1} \right) \end{aligned}$$

Proof.

$$
\begin{aligned}
&\mathrm{E}\left[P_{F.P.}^{\mathcal{T}^{(u,v)}\text{-}mul} \,\middle|\, |\mathcal{T}^{(u,v)}\text{-}mul| = N \right] \\
&= \frac{\mathrm{E}\left[\left| \left\{ s : s \in \mathcal{S}_{u,v},\ \theta_{u,v}(s) \in \mathcal{T}^{(u,v)}\text{-}mul \right\} \right| \,\middle|\, |\mathcal{T}^{(u,v)}\text{-}mul| = N \right]}{|\mathcal{S}_{u,v}|} \\
&= \frac{\sum_{s \in \mathcal{S}_{u,v}} \Pr\left[\theta_{u,v}(s) \in \mathcal{T}^{(u,v)}\text{-}mul \,\middle|\, |\mathcal{T}^{(u,v)}\text{-}mul| = N \right]}{|\mathcal{S}_{u,v}|} \\
&= \frac{\sum_{m \in \mathcal{M}_{u,v}} \left\{ \Pr\left[m \in \mathcal{T}^{(u,v)}\text{-}mul \,\middle|\, |\mathcal{T}^{(u,v)}\text{-}mul| = N \right] \times |\{ s : s \in \mathcal{S}_{u,v}, \theta_{u,v}(s) = m \}| \right\}}{|\mathcal{S}_{u,v}|} \\
&= \sum_{m \in \mathcal{M}_{u,v}} \left\{ \left(1 - (1 - p(m))^N \right) \times \frac{|\{ s : s \in \mathcal{S}_{u,v}, \theta_{u,v}(s) = m \}|}{|\mathcal{S}_{u,v}|} \right\} \quad \text{(by Lemma 1)} \\
&= \sum_{m \in \mathcal{M}_{u,v}} \left\{ \left(1 - (1 - p(m))^N \right) \times p(m) \right\} \qquad (2) \\
&= \sum_{m \in \mathcal{M}_{u,v}} \left\{ \binom{N}{1} p(m)^2 - \binom{N}{2} p(m)^3 + \cdots + (-1)^N \binom{N}{N} p(m)^{N+1} \right\} \\
&= \sum_{i=1}^{N} \left\{ \sum_{m \in \mathcal{M}_{u,v}} (-1)^{i+1} \binom{N}{i} p(m)^{i+1} \right\}
\end{aligned}
$$

□

Fig. 2 Proof of Lemma 2.

$$
\begin{aligned}
&= \binom{k}{0} + \sum_{i=1}^{m} \left((-1)^i \binom{k}{i} x^i + (-1)^i \binom{k}{i-1} x^i \right) \\
&\qquad + (-1)^{m+1} \binom{k}{m} x^{m+1} \\
&= \binom{k+1}{0} + \sum_{i=1}^{m} \left((-1)^i \binom{k+1}{i} x^i \right) - \binom{k}{m} x^{m+1} \\
&\le \sum_{i=0}^{m} (-1)^i \binom{k+1}{i} x^i .
\end{aligned}
$$

Similarly, if m is odd, it is true for $n = m$ because both sides are equal. Suppose it is true for $n = k$. Then, for $n = k + 1$,

$$
\begin{aligned}
&(1-x)^{k+1} = (1-x)^k (1-x) \\
&\ge \left(\sum_{i=0}^{m} (-1)^i \binom{k}{i} x^i \right) (1-x) \\
&= \binom{k}{0} + \sum_{i=1}^{m} \left((-1)^i \binom{k}{i} x^i + (-1)^i \binom{k}{i-1} x^i \right) \\
&\qquad + (-1)^{m+1} \binom{k}{m} x^{m+1} \\
&= \binom{k+1}{0} + \sum_{i=1}^{m} \left((-1)^i \binom{k+1}{i} x^i \right) + \binom{k}{m} x^{m+1} \\
&\ge \sum_{i=0}^{m} (-1)^i \binom{k+1}{i} x^i .
\end{aligned}
$$

□

Theorem 1. *For an even positive integer x,*

$$
Approx[x] \le \mathrm{E}\left[P_{F.P.}^{\mathcal{T}^{(u,v)}\text{-}mul} \,\middle|\, |\mathcal{T}^{(u,v)}\text{-}mul| = N \right]
$$

and for an odd positive integer x,

$$
\mathrm{E}\left[P_{F.P.}^{\mathcal{T}^{(u,v)}\text{-}mul} \,\middle|\, |\mathcal{T}^{(u,v)}\text{-}mul| = N \right] \le Approx[x].
$$

Proof. From Eq. (2),

$$
\begin{aligned}
&\mathrm{E}\left[P_{F.P.}^{\mathcal{T}^{(u,v)}\text{-}mul} \,\middle|\, |\mathcal{T}^{(u,v)}\text{-}mul| = N \right] \\
&\qquad = \sum_{m \in \mathcal{M}_{u,v}} \left\{ \left(1 - (1 - p(m))^N \right) \right\} \times p(m).
\end{aligned}
$$

If x is even, from Lemma 3,

$$
\begin{aligned}
&\mathrm{E}\left[P_{F.P.}^{\mathcal{T}^{(u,v)}\text{-}mul} \,\middle|\, |\mathcal{T}^{(u,v)}\text{-}mul| = N \right] \\
&\ge \sum_{m \in \mathcal{M}_{u,v}} \left\{ 1 - \sum_{i=0}^{x} (-1)^i \binom{N}{i} p(m)^i \right\} \times p(m) \\
&= \sum_{m \in \mathcal{M}_{u,v}} \left\{ \sum_{i=1}^{x} (-1)^{i+1} \binom{N}{i} p(m)^{i+1} \right\} \\
&= Approx[x].
\end{aligned}
$$

Similarly, if x is odd, from Lemma 3,

$$\begin{aligned}
&\mathrm{E}\left[P_{F.P.}^{\mathcal{T}^{(u,v)\text{-}mul}} \,\middle|\, |\mathcal{T}^{(u,v)\text{-}mul}| = N\right] \\
&\leq \sum_{m\in\mathcal{M}_{u,v}} \left\{1 - \sum_{i=0}^{x}(-1)^i\binom{N}{i}p(m)^i\right\} \times p(m) \\
&= \sum_{m\in\mathcal{M}_{u,v}} \left\{\sum_{i=1}^{x}(-1)^{i+1}\binom{N}{i}p(m)^{i+1}\right\} \\
&= Approx[x].
\end{aligned}$$

□

We will now describe the computation of Δ_x, focusing specifically on the evaluation of the expression:

$$\sum_{m\in\mathcal{M}_{u,v}} p(m)^{x+1}. \tag{4}$$

Let $a_0, a_1, \cdots, a_{2^v-1}$ represent the counts of occurrences of the numbers $0, 1, \cdots, 2^v - 1$ in the multiset (u,v)-*multiset* m. We can compute $p(m)$ as follows:

$$p(m) = \frac{u!}{a_0!a_1!\cdots a_{2^v-1}!} \times \frac{1}{|S_{u,v}|}.$$

Rewriting Eq. (4) yields[†]

$$2^{-uv(x+1)} \times \left\{\sum_{a_0+\cdots+a_{2^v-1}=u} \left(\frac{u!}{a_0!a_1!\cdots a_{2^v-1}!}\right)^{x+1}\right\}. \tag{5}$$

Next, we present a dynamic programming-based algorithm for computing Eq. (5). For this purpose, we define

$$A_{i,j,k,l} = \sum_{a_0+\cdots+a_{i-1}=j} \left\{\left(\frac{l!}{a_0!\cdots a_{i-1}!}\right)^k\right\}.$$

The challenge now is to calculate $A_{2^v,u,x+1,u}$. We can compute it directly based on the following properties of $A_{i,j,k,l}$:

$$\begin{aligned}
&A_{i,j,k,l} \\
&= \begin{cases} \left(\dfrac{l!}{j!}\right)^k, & \text{if } i = 1. \\ \displaystyle\sum_{m=0}^{j}\left\{A_{i-1,m,k,l} \cdot \left(\frac{1}{(j-m)!}\right)^k\right\}, & \text{otherwise.} \end{cases}
\end{aligned}$$

The detailed process is described in Algorithm 1.

Finally, we rewrite Eq. (3), as follows:

$$\Delta_x = (-1)^{x+1} \cdot \binom{N}{x} \cdot 2^{-uv(x+1)} \cdot A_{2^v,u,x+1,u}. \tag{6}$$

Now, we demonstrate that the difference between

$$Approx[1] \text{ and } \mathrm{E}\left[P_{F.P.}^{\mathcal{T}^{(u,v)\text{-}mul}} \,\middle|\, |\mathcal{T}^{(u,v)\text{-}mul}| = N\right]$$

is negligible. Based on Theorem 1, we have

[†]$|\mathcal{S}_{u,v}| = 2^{uv}$.

Algorithm 1: An algorithm to calculate $A_{i,j,k,l}$

```
if i = 1 then
    return (l!/j!)^k
for y ← 0 to j do
    dp[1][y] ← (l!/y!)^k
for x ← 2 to i do
    for y ← 0 to j do
        dp[x][y] ← 0
        for z ← 0 to y do
            dp[x][y] ←
                dp[x][y] + dp[x − 1][z] × (1/(y−z)!)^k
return dp[i][j]
```

Table 3 Comparison of Δ_1 and Δ_2 according to u and v.

(u,v)	Δ_1	Δ_2	(u,v)	Δ_1	Δ_2
$(256,8)$	$2^{-378.6}$	$-2^{-732.8}$	$(64,8)$	$2^{-141.8}$	$-2^{-281.9}$
$(256,7)$	$2^{-211.6}$	$-2^{-403.7}$	$(64,7)$	$2^{-83.6}$	$-2^{-163.7}$
$(256,6)$	$2^{-95.0}$	$-2^{-179.3}$	$(64,6)$	$2^{-31.4}$	$-2^{-57.6}$
$(256,5)$	$2^{-20.6}$	$-2^{-36.1}$	$(32,8)$	$2^{-59.7}$	$-2^{-119.8}$
$(128,8)$	$2^{-251.3}$	$-2^{-494.3}$	$(32,7)$	$2^{-29.2}$	$-2^{-58.1}$
$(128,7)$	$2^{-147.0}$	$-2^{-282.3}$	$(16,8)$	$2^{-4.09}$	$-2^{-9.01}$
$(128,6)$	$2^{-63.4}$	$-2^{-117.7}$			

$$\begin{aligned}
Approx[2] &\leq \mathrm{E}\left[P_{F.P.}^{\mathcal{T}^{(u,v)\text{-}mul}} \,\middle|\, |\mathcal{T}^{(u,v)\text{-}mul}| = N\right] \\
&\leq Approx[1].
\end{aligned}$$

Equivalently,

$$\Delta_1 + \Delta_2 \leq \mathrm{E}\left[P_{F.P.}^{\mathcal{T}^{(u,v)\text{-}mul}} \,\middle|\, |\mathcal{T}^{(u,v)\text{-}mul}| = N\right] \leq \Delta_1.$$

We calculate Δ_1 and Δ_2 for various (u,v) under $N = 2^{80}$, which is the case of the distinguisher presented in [6]. The results are described in Table 3 and we can see that $|\Delta_2|$ is negligible compared to $|\Delta_1|$, suggesting that Δ_1 serves as a satisfactory approximation of $\mathrm{E}\left[P_{F.P.}^{\mathcal{T}^{(u,v)\text{-}mul}} \,\middle|\, |\mathcal{T}^{(u,v)\text{-}mul}| = N\right]$.

From Eq. (6),

$$Approx[1] = \Delta_1 = N \cdot 2^{-2uv} \cdot A_{2^v,u,2,u}. \tag{7}$$

If we precalculate $2^{-2uv} \cdot A_{2^v,u,2,u}$, we can readily determine the false-positive probability given a specific value of N. Table 4 lists $2^{-2uv} \cdot A_{2^v,u,2,u}$ according to u and v.

4.1 Toy Experiments

To validate the accuracy of our approach, we conducted toy experiments, comparing the false-positive probability based on Definition 8 with the probability obtained through our methodology. These experiments were performed within a small-scale environment, where the false-positive probability based on Definition 8 can be calculated within a reasonable computational complexity.

Table 4 $(2^{-2uv} \cdot A_{2^v,u,2,u})$ according to u and v.

u \ v	8	7	6	5	4	3	2	1
256	$2^{-458.60}$	$2^{-291.62}$	$2^{-175.03}$	$2^{-100.62}$	$2^{-55.40}$	$2^{-28.79}$	$2^{-13.48}$	$2^{-4.83}$
128	$2^{-331.25}$	$2^{-226.99}$	$2^{-143.44}$	$2^{-85.13}$	$2^{-47.91}$	$2^{-25.29}$	$2^{-11.98}$	$2^{-4.33}$
64	$2^{-221.76}$	$2^{-163.63}$	$2^{-111.44}$	$2^{-69.61}$	$2^{-40.42}$	$2^{-21.80}$	$2^{-10.49}$	$2^{-3.83}$
32	$2^{-139.74}$	$2^{-109.17}$	$2^{-80.06}$	$2^{-53.91}$	$2^{-32.94}$	$2^{-18.32}$	$2^{-8.99}$	$2^{-3.33}$
16	$2^{-84.09}$	$2^{-68.43}$	$2^{-53.12}$	$2^{-38.53}$	$2^{-25.39}$	$2^{-14.85}$	$2^{-7.51}$	$2^{-2.84}$
8	$2^{-48.78}$	$2^{-40.86}$	$2^{-33.02}$	$2^{-25.34}$	$2^{-18.01}$	$2^{-11.37}$	$2^{-6.04}$	$2^{-2.35}$
4	$2^{-27.43}$	$2^{-23.45}$	$2^{-19.48}$	$2^{-15.55}$	$2^{-11.69}$	$2^{-7.98}$	$2^{-4.59}$	$2^{-1.87}$
2	$2^{-15.00}$	$2^{-13.01}$	$2^{-11.01}$	$2^{-9.02}$	$2^{-7.05}$	$2^{-5.09}$	$2^{-3.19}$	$2^{-1.42}$

We carried out experiments involving the random sampling of 2^{20} sequences that could serve as the outputs of the distinguisher presented in [6] and [1]. Subsequently, we transformed sequences to multisets and then computed the false-positive probabilities as defined in Definition 8. Further, we employed our proposed methodology to estimate the false-positive probability. The estimation $(Approx[1])$ was obtained by multiplying the values in Table 4 by a factor of 2^{20}.

First, we introduce the distinguisher for 4-round AES in [6] as Proposition 2, which will be used for our toy experiments.

Proposition 2. *([6]) Suppose a pair of states $(\mathcal{P}, \mathcal{P}')$ conforms to the truncated differential trail shown in Fig. 3. Let $\{\mathcal{P}^0, \mathcal{P}^1, \ldots \mathcal{P}^{255}\}$ is a 8-δ-set where $\mathcal{P}_0 = \mathcal{P}$, and $\{C^0, C^1, \ldots C^{255}\}$ is a set of states where C_i is a state of $\mathcal{P}_i$ after four full AES rounds of encryption. Then the sequence $(\Delta C^0[0], \Delta C^1[0], \ldots \Delta C^{255}[0])$ (or the multiset $[\Delta C^0[0], \Delta C^1[0], \ldots \Delta C^{255}[0]]$) can only take 2^{80} values, where $\Delta C^i = C^i \oplus C^0$.*

To provide a concise proof of Proposition 2 and to describe the sampling process, we define some notations. We consider a 4-round AES and we denote the intermediate states prior to SubByte, ShiftRow, MixColumn, and AddroundKey operations of round i associated with $\mathcal{P}$ as x_i, y_i, z_i, and w_i, respectively. Additionally, we denote the states related to $\mathcal{P}'$ as x'_i, y'_i, z'_i, and w'_i. Similarly, the intermediate states corresponding to $\mathcal{P}^j$ are represented as X_i^j, Y_i^j, Z_i^j, and W_i^j. A visual representation of these notations is provided in Fig. 3.

We briefly provide the proof of Proposition 2, which is required to explain the sampling process. For a more detailed proof, we refer the reader to [6], [8], [12]. The works in [8], [12] demonstrated that the sequence $(\Delta C^0[0], \Delta C^1[0], \ldots \Delta C^{255}[0])$ is determined by the 24 bytes $x_2[0,1,2,3]$, x_3[0-15], and $x_4[0,5,10,15]$. Additionally, [6] established that if $(\mathcal{P}, \mathcal{P}')$ conforms to the truncated differential trail shown in Fig. 3, then these 24 bytes are constrained by the 10 bytes $\Delta z_1[0]$, $x_2[0,1,2,3]$, $\Delta w_4[0]$, and $z_4[0,1,2,3]$. As a result, the sequence $(\Delta C^0[0], \Delta C^1[0], \ldots \Delta C^{255}[0])$ can assume 2^{80} values.

Our sampling process consists of two phases. In the initial phase, we randomly assign values to 10 specific bytes: $\Delta z_1[0]$, $x_2[0,1,2,3]$, $\Delta w_4[0]$, and $z_4[0,1,2,3]$. Subsequently, we derive the corresponding values for 24 bytes: $x_2[0,1,2,3]$, x_3[0-15], and $x_4[0,5,10,15]$. In the second phase, we utilize the computed values from the first phase to derive $(\Delta C^0[0], \Delta C^1[0], \ldots \Delta C^{255}[0])$. We repeat the two phases until we have gathered 2^{20} sequences. More details about the two phases are as follows:

(1) First Phase.

1. Randomly assign values to $\Delta z_1[0]$, $x_2[0,1,2,3]$, $\Delta w_4[0]$, and $z_4[0,1,2,3]$.
2. Compute $x_2[0,1,2,3]$, x_3[0-15], $x_4[0,5,10,15]$.
   a. Calculate $\Delta x_2[0,1,2,3]$ from $\Delta z_1[0]$.
   b. Calculate $\Delta z_2[0,7,10,13]$ from $x_2[0,1,2,3]$ and $\Delta x_2[0,1,2,3]$. Then, compute Δx_3 using $\Delta z_2[0,7,10,13]$.
   c. Calculate $\Delta z_4[0,1,2,3]$ from $\Delta w_4[0]$. Then, compute $x_4[0,5,10,15]$ and $x'_4[0,5,10,15]$ using $z_4[0,1,2,3]$ and $\Delta z_4[0,1,2,3]$.
   d. Calculate $\Delta x_4[0,5,10,15]$ from $x_4[0,5,10,15]$ and $x'_4[0,5,10,15]$. Then, compute Δy_3 using $\Delta x_4[0,5,10,15]$.
   e. Solve 16 S-box differential equations between Δx_3 and Δy_3 to obtain x_3[0-15]. If no solution is found, return to Step 1. If solutions are discovered, proceed to the Second Phase with each solution.

(2) Second Phase.

Note that X_2^0[0-3] = x_2[0-3], X_3^0[0-15] = x_3[0-15], and $X_4^0[0,5,10,15] = x_4[0,5,10,15]$.

1. For $i = 0, 1, \ldots, 255$, do the followings:
   a. Assign i to $\Delta Z_1^i[0]$ and calculate ΔX_2^i[0-3] from

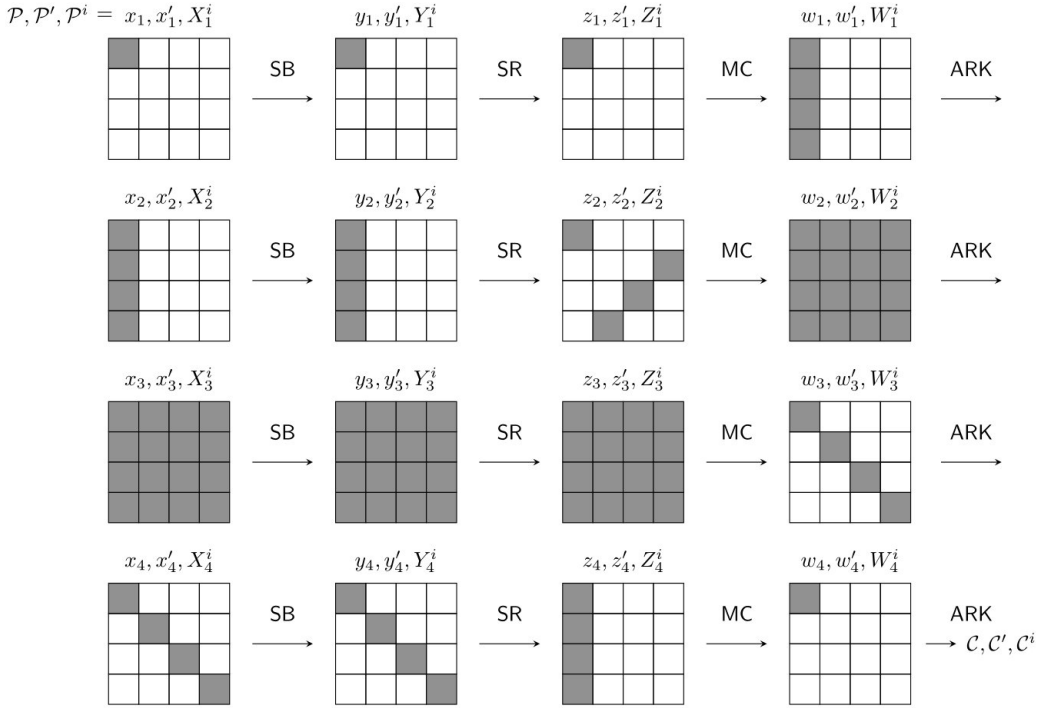

Fig. 3 Truncated differential trails of 4-round AES corresponding to Proposition 2.

$\Delta Z_1^i[0]$.

b. Calculate $X_2^i[0\text{-}3]$ from $X_2^0[0\text{-}3]$ and $\Delta X_2^i[0\text{-}3]$. Then, compute $Z_2^i[0,7,10,13]$ using $X_2^i[0\text{-}3]$.

c. Calculate $\Delta Z_2^i[0,7,10,13]$ from $Z_2^0[0,7,10,13]$ and $Z_2^i[0,7,10,13]$. Then, compute $\Delta X_3^i[0\text{-}15]$ using $\Delta Z_2^i[0,7,10,13]$.

d. Calculate $X_3^i[0\text{-}15]$ from $X_3^0[0\text{-}15]$ and $\Delta X_3^i[0\text{-}15]$. Then compute $Z_3^i[0\text{-}15]$ using $X_3^i[0\text{-}15]$.

e. Calculate $\Delta Z_3^i[0\text{-}15]$ from $Z_3^0[0\text{-}15]$ and $Z_3^i[0\text{-}15]$. Then compute $\Delta X_4^i[0,5,10,15]$ using $\Delta Z_3^i[0\text{-}15]$.

f. Calculate $X_4^i[0,5,10,15]$ from $X_4^0[0,5,10,15]$ and $\Delta X_4^i[0,5,10,15]$. Then, compute $Z_4^i[0\text{-}3]$ using $X_4^i[0,5,10,15]$.

g. Calculate $\Delta Z_4^i[0\text{-}3]$ from $Z_4^0[0,1,2,3]$ and $\Delta Z_4^i[0\text{-}3]$. Then, compute $C^i[0]$ using $\Delta Z_4^i[0\text{-}3]$.

2. Store a sequence $(C^0[0], C^1[0], \ldots, C^{255}[0])$ to $\mathcal{T}_{\mathtt{AES}}^{(256,8)\text{-}seq}$.

After sampling 2^{20} sequences, we perform an exhaustive calculation of the false-positive probability, as defined in Definition 8. For different values of u and v, we transform 2^{20} sequences into (u,v)-*multisets* and aggregate the number of (u,v)-*sequences* associated with each multiset. We then divide the aggregated value by $|S_{u,v}|$. Specifically:

1. Initialize $P_{\mathsf{fp}}^{\mathcal{T}_{\mathtt{AES}}^{(u,v)\text{-}mul}}$ with 0.
2. For all $(256,8)$-sequence $s \in \mathcal{T}_{\mathtt{AES}}^{(256,8)\text{-}seq}$, do the followings:

a. Suppose $s = (a_0, a_1, \ldots, a_{255})$. We transform s into a (u,v)-*multiset* m by adjusting the number of elements and truncating their ranges:

$$m = [a_0 \bmod 2^v, a_1 \bmod 2^v, \ldots, a_{u-1} \bmod 2^v]. \tag{8}$$

Additionally, we define b_i as the count of occurrences of i among the elements of m, where $0 \le i < 2^v$.

b. Add the number of (u,v)-*sequences* that are associated with m to $P_{\mathsf{fp}}^{\mathcal{T}_{\mathtt{AES}}^{(u,v)\text{-}mul}}$:

$$P_{\mathsf{fp}}^{\mathcal{T}_{\mathtt{AES}}^{(u,v)\text{-}mul}} \leftarrow P_{\mathsf{fp}}^{\mathcal{T}_{\mathtt{AES}}^{(u,v)\text{-}mul}} + \frac{u!}{b_0! b_1! \cdots b_{2^v-1}!}.$$

3. Divide $P_{\mathsf{fp}}^{\mathcal{T}_{\mathtt{AES}}^{(u,v)\text{-}mul}}$ by $|\mathcal{S}_{u,v}|$:

$$P_{\mathsf{fp}}^{\mathcal{T}_{\mathtt{AES}}^{(u,v)\text{-}mul}} \leftarrow \frac{P_{\mathsf{fp}}^{\mathcal{T}_{\mathtt{AES}}^{(u,v)\text{-}mul}}}{|\mathcal{S}_{u,v}|}.$$

Similarly, we sampled and calculated the false-positive probability for the case of ARIA [1]. The corresponding distinguisher is described as Proposition 4, and the truncated differential trail is illustrated in Fig. A· 3. The results of our toy experiments are summarized in Table 5. Notably, the differences between $P_{\mathsf{fp}}^{\mathcal{T}_{\mathtt{AES}}^{(u,v)\text{-}mul}}$, $P_{\mathsf{fp}}^{\mathcal{T}_{\mathtt{ARIA}}^{(u,v)\text{-}mul}}$, and $Approx[1]$ are negligible. This provides compelling evidence that validates the accuracy of our method in estimating the false-positive probability.

The codes for our methodology and toy experiments, as

Table 5 Toy experiment results.

(u,v)	$*$	$**$	$***$	(u,v)	$*$	$**$	$***$
$(256,8)$	-438.6	-439.26	-439.52	$(64,3)$	-1.80	-1.80	-1.8
$(256,7)$	-271.62	-271.70	-271.20	$(32,8)$	-119.74	-119.74	-119.75
$(256,6)$	-155.03	-155.13	-155.00	$(32,7)$	-89.17	-89.17	-89.17
$(256,5)$	-80.62	-80.63	-80.60	$(32,6)$	-60.06	-60.07	-60.06
$(256,4)$	-35.40	-35.40	-35.39	$(32,5)$	-33.91	-33.90	-33.90
$(256,3)$	-8.79	-8.79	-8.78	$(32,4)$	-12.94	-12.93	-12.94
$(128,8)$	-311.25	-311.28	-311.24	$(16,8)$	-64.09	-64.09	-64.09
$(128,7)$	-206.99	-207.04	-206.93	$(16,7)$	-48.43	-48.43	-48.43
$(128,6)$	-123.44	-123.50	-123.42	$(16,6)$	-33.12	-33.12	-33.12
$(128,5)$	-65.13	-65.13	-65.14	$(16,5)$	-18.53	-18.53	-18.53
$(128,4)$	-27.91	-27.90	-27.9	$(16,4)$	-5.39	-5.39	-5.39
$(128,3)$	-5.29	-5.29	-5.29	$(8,8)$	-28.78	-28.78	-28.78
$(64,8)$	-201.76	-201.76	-201.76	$(8,7)$	-20.86	-20.86	-20.86
$(64,7)$	-143.63	-143.63	-143.63	$(8,6)$	-13.02	-13.02	-13.02
$(64,6)$	-91.44	-91.43	-91.45	$(8,5)$	-5.34	-5.34	-5.34
$(64,5)$	-49.61	-49.60	-49.61	$(4,8)$	-7.43	-7.43	-7.43
$(64,4)$	-20.42	-20.42	-20.43	$(4,7)$	-3.45	-3.45	-3.45

$* : \log_2(Approx[1])$, $** : \log_2\left(P_{\mathsf{fp}}^{\mathcal{T}_{\mathtt{AES}}^{(u,v)\text{-}mul}}\right)$, $*** : \log_2\left(P_{\mathsf{fp}}^{\mathcal{T}_{\mathtt{ARIA}}^{(u,v)\text{-}mul}}\right)$.

well as more detailed results, can be found at https://github.com/DongjaeLee-Dd2dD2/Accurate-FPP-DSMITM.

4.2 Analyzing Flaws in Previous False-Positive Probability Estimates

First, we provide an overview of [8]'s approach to estimating false-positive probability. The authors assumed that the multisets follow the Poisson distribution. Based on this assumption, they suggested that the most probable form of $(256,8)$-*multiset* is as follows: among values ranging from 0 to 255, 94 values do not appear, 94 appear once, 47 appear twice, 16 appear three times, 4 appear four times, and 1 is expected to appear five times in the multiset. In other words, the multisets $[a_0, a_1, \ldots, a_{255}]$ that satisfy the following six properties are the most probable.

1. 94 values do not appear:
   $\#\{i \in \{0,1,\ldots,255\} : \#\{j : a_j = i\} = 0\} = 94$.
2. 94 values appear once:
   $\#\{i \in \{0,1,\ldots,255\} : \#\{j : a_j = i\} = 1\} = 94$.
3. 47 values appear twice:
   $\#\{i \in \{0,1,\ldots,255\} : \#\{j : a_j = i\} = 2\} = 47$.
4. 16 values appear three times:
   $\#\{i \in \{0,1,\ldots,255\} : \#\{j : a_j = i\} = 3\} = 16$.
5. 4 values appear four times:
   $\#\{i \in \{0,1,\ldots,255\} : \#\{j : a_j = i\} = 4\} = 4$.
6. 1 value appears five times:
   $\#\{i \in \{0,1,\ldots,255\} : \#\{j : a_j = i\} = 5\} = 1$.

[8] then estimated the false-positive probability based on the assumption that all multisets, including both those stored in the pre-computed table and those obtained from the wrong key guess, follow this form. The total number of multisets following this form is calculated as

$$\binom{256}{94} \cdot \binom{162}{94} \cdot \binom{68}{47} \cdot \binom{21}{16} \cdot \binom{5}{4} \cdot \binom{1}{1} = 2^{467.6}.$$

Table 6 Comparison of false-positive probabilities between our methodology and previous works.

(N,u,v)	False-positive Probability		Reference
	Ours (Δ_1)	Previous works	
$(2^{80},256,8)$	$2^{-378.6}$	$2^{-387.6}$	[6]
$(2^{128},256,8)$	$2^{-330.6}$	$2^{-339.6}$	[1]
$(2^{136},256,8)$	$2^{-322.6}$	$2^{-331.6}$	[1]

Consequently, [8] estimated the false-positive probability as $N \cdot 2^{-467.6}$, where N is the size of the pre-computed table. Subsequently, [6] and [1] followed a similar methodology in estimating false-positive probability.

However, assuming that all multisets have the same form, although it is the most probable, is excessive and results in minor flaws in the probabilities derived from this assumption. According to our methodology, the accurate false-positive probability when constructing the pre-computed table with $(256,8)$-*multiset* is $N \cdot 2^{-458.6\dagger}$. This implies a discrepancy of 2^9 between the probabilities presented in the previous works and the correct false-positive probabilities. The accurate values are 2^9 times greater than the suggested probabilities.

Specifically, [6] and [1] presented false-positive probabilities of $2^{-387.6}$, $2^{-339.6}$, and $2^{-331.6}$ for the parameter sets $(N,u,v) = (2^{80},256,8), (2^{128},256,8)$, and $(2^{136},256,8)$, respectively. In contrast, our methodology yields the correct false-positive probabilities for the same parameter sets, namely $2^{-378.60}$, $2^{-330.60}$, and $2^{-322.60\dagger\dagger}$. The comparison of false-positive probabilities between our methodology and previous works is summarized in Table 6.

†Please refer to Eq. (7) and Table 4.

††It is important to note that while the probabilities provided in [1], [6], [8] have a flaw, the feasibility of their attacks is not affected.

5. Optimizing the $\mathcal{DS}$-MITM Attack and Its Applications

As illustrated in Eq. (8) we can transform (u, v)-*sequence* to (u', v')-*multiset*, where $u' \leq u$ and $v' \leq v$:

$$(a_0, a_1, \ldots, a_{u-1}) \rightarrow [a_0 \bmod 2^{v'}, a_1 \bmod 2^{v'}, \ldots, a_{u'-1} \bmod 2^{v'}].$$

This transformation leads to a reduction in memory complexity. Simultaneously, it introduces changes to several factors influencing attack complexity, including the false-positive probability. In this section, we present a method to determine the optimal format for precomputed data by analyzing the effects of this transformation on attack complexity. Specifically, we rework the framework presented in Sect. 3 according to the format of precomputed data. Subsequently, we apply our method to previous works on `AES` and `ARIA` to achieve modest improvements.

Assuming the format of the precomputed data is (u, v)-*multiset*, the memory complexity is determined as $N \cdot \lceil \log_2(|M_{u,v}|) \rceil$. Additionally, since obtaining a single multiset requires the computation of u elements, the time complexity of the offline phase becomes $N \cdot u \cdot T_{\mathsf{dist}}$. Step 1 of the online phase remains unaffected, as it is the process of finding message pairs that conform to the plaintext and ciphertext differentials.

Careful consideration is needed when adjusting u, as it might lead to an increase in the number of key bits that need to be guessed during the online phase. This is because the b-δ-*set* might not be preserved when subjected to a substitution operation, depending on the value of b and the size of the unit to which the operation is applied. We will illustrate this point using the example of the distinguisher outlined in Proposition 2. After a SubBytes operation in `AES`, an 8-δ-*set* retains its structure, while a 7, 6, ...-δ-*set* does not. Consequently, when we take the value of u as 128, 64, ..., an additional key byte associated with $X_1^0[0]$ needs to be guessed (Refer to Fig. 3). We denote the number of guessed bits within the partial key for E_1 based on the value of u as $k_{1,u}$. Consequently, we can rewrite the framework as follows:

$$\begin{aligned}
&\text{Memory Complexity} = N \cdot \lceil \log_2(|M_{u,v}|) \rceil \\
&\text{Time Complexity} = T_{\mathsf{off}} + T_{\mathsf{on}}, \\
&\begin{cases} T_{\mathsf{off}} = N \cdot u \cdot T_{\mathsf{dist}}, \\ T_{\mathsf{on}} = T_1 + T_2 + T_3, \\ \quad \begin{cases} T_1 = (p_f p_b)^{-1} \cdot 2^{n-\alpha_p-\alpha_c+1}, \\ T_2 = (p_f p_b)^{-1} \cdot 2^{k_{1,u}+k_3} \cdot u \cdot T_{\mathsf{multi}}, \\ T_3 = (1 + (p_f p_b)^{-1} \cdot 2^{k_{1,u}+k_3} \cdot P_{F.P.}) \cdot T_{\mathsf{rem}}, \end{cases} \end{cases} \\
&\text{Data Complexity} : (p_f p_b)^{-1} \cdot 2^{n-\alpha_p-\alpha_c+1}.
\end{aligned}$$

We can readily calculate the attack complexity based on u and v using the framework. Consequently, by comparing the resulting complexities, we can determine the optimal format for the precomputed data.

5.1 Applications

We apply our optimization methods to previous works on `AES` and `ARIA`, resulting in modest improvements to the $\mathcal{DS}$-MITM attack on 7-round `AES-128/192/256` [6], 7-round `ARIA-192/256`, and 8-round `ARIA-256` [1]. The parameters for each attack are summarized in Table 7. Except for T_{dist} for $\mathcal{A}_1$ and $\mathcal{A}2$, the remaining parameters can be readily obtained from the previous works. Specifically, the parameters for `AES` are available in [6], and those for `ARIA` can be found in [1]. The value of T_{dist} for $\mathcal{A}_1$ and $\mathcal{A}_2$ can be calculated by considering the number of S-boxes. While more advanced attacks have been proposed for `AES` (such as the attacks on 8- and 9-round `AES` in [3], [6]), we do not address these attacks in our study because our optimization methods offer no further room for improvement.

5.1.1 Optimizing the $\mathcal{DS}$-MITM Attacks on 7-Round AES

We have optimized two versions of the $\mathcal{DS}$-MITM attacks on 7-round `AES-128/192/256`, which are based on the two distinguishers presented in [6]. The first distinguisher is described as Proposition 2, which was discussed in Section 4. The corresponding full-round truncated differential trail is illustrated in Fig. A· 1 in Appendix Appendix.

Based on our framework, the complexity of the attack is as follows:

– Memory Complexity = $2^{80} \cdot \lceil \log_2(|M_{u,v}|) \rceil$

– Time Complexity = $T_{\mathsf{off}} + T_{\mathsf{on}}$,

$T_{\mathsf{off}} = 2^{77.5} \cdot u$, $T_{\mathsf{on}} = T_1 + T_2 + T_3$,

$T_1 = 2^{113}$,

$$T_2 = \begin{cases} 2^{75}, & \text{if } u = 256, \\ 2^{75} \cdot u, & \text{otherwise}, \end{cases}$$

$$T_3 = \begin{cases} (1 + 2^{72} \cdot P_{F.P.}) \cdot 2^{80}, & \text{if } u = 256, \\ (1 + 2^{80} \cdot P_{F.P.}) \cdot 2^{80}, & \text{otherwise}, \end{cases}$$

– Data Complexity = 2^{113}.

In this attack, the dominant factor for the time complexity is T_1, and T_2 remains negligible regardless of the value of u. We can reduce either u or v while maintaining $T_3 < 2^{113}$. Specifically, this reduction must satisfy the following condition: $P_{F.P.} < 2^{-39}$ if $u = 256$, and $P_{F.P.} < 2^{-47}$ otherwise. According to Table 4, the optimal values are $u = 32$ and $v = 8$. Consequently, we achieve a reduction in memory complexity by a factor of $2^{1.85}$ and a reduction in the offline phase's time complexity by a factor of 2^3, while the other complexities remain unchanged.

The second distinguisher is described as Proposition 3.

Proposition 3. *([6]) If a pair of messages $(\mathcal{P}, \mathcal{P}')$ conforms to the truncated differential trail of Fig. A· 2, then the multiset*

Table 7 Parameters of the attacks.

Attack Group	$\mathcal{A}_1$			$\mathcal{A}_2$			$\mathcal{A}_3$		$\mathcal{A}_4$
Reference	[6]						[1]		
Target Cipher	AES						ARIA		
Version	-128	-192	-256	-128	-192	-256	-192	-256	-256
Distinguisher	Prop. 2			Prop. 3			Prop. 4		Prop. 5
Truncated Differential Trail	Fig. A· 1			Fig. A· 2			Fig. A· 3		Fig. A· 4
N	2^{80}			2^{88}			2^{128}		2^{136}
n	128								
$(k_{1,u}, k_3), u = 256$	(8, 16)			(8, 16)			(8, 16)		(8, 64)
$(k_{1,u}, k_3), u \neq 256$	(16, 16)			(16, 16)			(16, 16)		(16, 64)
(R, r_1, r_2, r_3)	(7, 1, 4, 2)			(7, 1, 4, 2)			(7, 1, 4, 2)		(8, 1, 4.5, 2.5)
(p_f, p_b)	$(2^{-24}, 2^{-24})$			$(2^{-48}, 2^{-24})$			$(2^{-48}, 2^{-48})$		$(2^{-48}, 2^{-120})$
$(\alpha_p, \alpha_c, \alpha_{st}, \alpha_{ed})$	(32, 32, 8, 8)			(64, 32, 16, 8)			(56, 56, 8, 8)		(56, 128, 8, 8)
T_{dist}	$2^{-2.5}$			$2^{-2.5}$			$2^{-1.9}$		2^{-2}
T_{multi}	2^{-5}			2^{-5}			$2^{-2.9}$		$2^{-2.1}$
T_{rem}	2^{80}			2^{88}			$2^{134.2}$	$2^{126.2}$	2^{16}

of differences between the distinguisher's outputs, obtained from the 8-δ*-set constructed from* $\mathcal{P}$*, can only take* 2^{88} *values.*

Based on our framework, the complexity of the attack is as follows:

– Memory Complexity = $2^{88} \cdot \lceil \log_2(|M_{u,v}|) \rceil$

– Time Complexity = $T_{\text{off}} + T_{\text{on}}$,

$T_{\text{off}} = 2^{85.5} \cdot u$, $T_{\text{on}} = T_1 + T_2 + T_3$,

$T_1 = 2^{105}$,

$$T_2 = \begin{cases} 2^{99}, & \text{if } u = 256, \\ 2^{99} \cdot u, & \text{otherwise,} \end{cases}$$

$$T_3 = \begin{cases} (1 + 2^{96} \cdot P_{F.P.}) \cdot 2^{88}, & \text{if } u = 256, \\ (1 + 2^{104} \cdot P_{F.P.}) \cdot 2^{88}, & \text{otherwise,} \end{cases}$$

– Data Complexity = 2^{105}.

In this attack, the dominant factor for the time complexity is T_1. In this case, we cannot adjust u because, when we take the value for u as 128 or 64, $T_1 + T_2$ becomes greater than 2^{105}, and if $u \leq 32$, T_3 becomes greater than 2^{105}. According to Table 4, the optimal value for v is 6. This reduction allows us to decrease the memory complexity by a factor of $2^{1.19}$ while keeping the other complexities unchanged.

5.1.2 Optimizing the $\mathcal{DS}$-MITM Attacks on 7-Round ARIA-192/256

We optimize the $\mathcal{DS}$-MITM attacks on 7-round ARIA-192/256 presented in [1]. The distinguisher used in the attack is described as Proposition 4.

Proposition 4. *([1]) If a pair of messages* $(\mathcal{P}, \mathcal{P}')$ *conforms to the truncated differential trail of Fig. A· 3, then the multiset of differences between distinguisher's outputs, obtained from the* 8-δ*-set constructed from* $\mathcal{P}$*, can only take* 2^{128} *values.*

Before optimizing the attack, it is important to note that the attack on ARIA-192 is consistent with our framework, while the attack on ARIA-256 takes a slightly different approach. Based on our framework, the complexity of an attack on 7-round ARIA-192 is as follows:

– Memory Complexity = $2^{128} \cdot \lceil \log_2(|M_{u,v}|) \rceil$

– Time Complexity = $T_{\text{off}} + T_{\text{on}}$,

$T_{\text{off}} = 2^{126.1} \cdot u$, $T_{\text{on}} = T_1 + T_2 + T_3$,

$T_1 = 2^{113}$,

$$T_2 = \begin{cases} 2^{125.1}, & \text{if } u = 256, \\ 2^{125.1} \cdot u, & \text{otherwise,} \end{cases}$$

$$T_3 = \begin{cases} (1 + 2^{120} \cdot P_{F.P.}) \cdot 2^{134.2}, & \text{if } u = 256, \\ (1 + 2^{128} \cdot P_{F.P.}) \cdot 2^{134.2}, & \text{otherwise,} \end{cases}$$

– Data Complexity = 2^{113}.

In this attack, the dominant factor for the time complexity is T_3. Therefore, we can adjust either u or v while ensuring that $P_{F.P.} < 2^{-120}$ when $u = 256$, and $P_{F.P.} < 2^{-128}$ otherwise. Based on Table 4, the optimal values are $u = 128$ and $v = 8$. As a result, we achieve a reduction in the memory

complexity, offline time complexity, and overall time complexity by factors of $2^{0.56}$, 2, and $2^{0.13}$, respectively, while the other complexities remain unchanged.

The attack on ARIA-256 repeats the process four times while changing the position of the active byte. Specifically, it iterates through the offline phase and steps 1 and 2 of the online phases, progressively reducing the key space. Then, the master key is recovered from the reduced key space. Consequently, the time and data complexities are quadrupled, while the memory complexity can be maintained by alternately conducting the offline and online phases. In summary, the complexity of this attack can be summarized as follows:

– Memory Complexity = $2^{128} \cdot \lceil \log_2(|M_{u,v}|) \rceil$
– Time Complexity = $T_{\text{off}} + T_{\text{on}}$,
$T_{\text{off}} = 2^{128.1} \cdot u$, $T_{\text{on}} = T_1 + T_2 + T_3$,
$T_1 = 2^{115}$,
$$T_2 = \begin{cases} 2^{127.1}, & \text{if } u = 256, \\ 2^{127.1} \cdot u, & \text{otherwise,} \end{cases}$$
$$T_3 = \begin{cases} (1 + (2^{120} \cdot P_{F.P.})^4) \cdot 2^{126.2}, & \text{if } u = 256, \\ (1 + (2^{128} \cdot P_{F.P.})^4) \cdot 2^{126.2}, & \text{otherwise,} \end{cases}$$
– Data Complexity = 2^{113}.

In this attack, the dominant factor for the time complexity is T_{off}. We can consider two approaches for this attack. The first is to consider the overall time complexity as important, and the second is to prioritize the time complexity of the online phase. In the first case, the optimal values are $u = 128$ and $v = 8$, leading to a reduction in the memory complexity, offline time complexity, and overall time complexity by factors of $2^{0.56}$, 2, and $2^{0.42}$, respectively. In the second case, the optimal values are $u = 256$ and $v = 7$, reducing the memory complexity by a factor of $2^{0.56}$ while the other complexities remain the same.

5.1.3 Optimizing the $\mathcal{DS}$-MITM Attack on 8-Round ARIA-256

We optimize the $\mathcal{DS}$-MITM attack on 8-round ARIA-256 presented in [1]. The distinguisher used in the attack is described as Proposition 5.

Proposition 5. *([1]) If a pair of messages $(\mathcal{P}, \mathcal{P}')$ conforms to the truncated differential trail of Fig. A·4, then the multiset of differences between the distinguisher's outputs, obtained from the 8-δ-set constructed from $\mathcal{P}$, can only take 2^{136} values.*

Based on our framework, the complexity of an attack is as follows:

– Memory Complexity = $2^{136} \cdot \lceil \log_2(|M_{u,v}|) \rceil$
– Time Complexity = $T_{\text{off}} + T_{\text{on}}$,
$T_{\text{off}} = 2^{134} \cdot u$, $T_{\text{on}} = T_1 + T_2 + T_3$,
$T_1 = 2^{113}$,
$$T_2 = \begin{cases} 2^{245.9}, & \text{if } u = 256, \\ 2^{245.9} \cdot u, & \text{otherwise,} \end{cases}$$
$$T_3 = \begin{cases} (1 + 2^{240} \cdot P_{F.P.}) \cdot 2^{16}, & \text{if } u = 256, \\ (1 + 2^{248} \cdot P_{F.P.}) \cdot 2^{16}, & \text{otherwise,} \end{cases}$$
– Data Complexity = 2^{113}.

In this attack, the dominant factor for the time complexity is T_2, and therefore, we cannot adjust the value of u. Based on Table 4, the optimal value for v is 6. Consequently, we can reduce the memory complexity by a factor of $2^{1.19}$ while keeping the other complexities unchanged.

6. Conclusion

This study introduced a novel method for calculating the false-positive probability of the multiset-based $\mathcal{DS}$-MITM attack. Additionally, we presented optimization methods for improving the attack and applied them to previous works on AES and ARIA. These efforts led to moderate improvements in memory and time complexity.

Acknowledgments

This work was supported in part by the Military Crypto Research Center funded by the Defense Acquisition Program Administration (DAPA) and the Agency for Defense Development (ADD) under Grant UD210027XD, and in part by the National Research Foundation of Korea (NRF) grant funded by the Korean Government Ministry of Science and ICT (MSIT) under Grant 2021R1A2C1005946.

References

[1] Akshima, D. Chang, M. Ghosh, A. Goel, and S.K. Sanadhya, "Improved meet-in-the-middle attacks on 7 and 8-round ARIA-192 and ARIA-256," INDOCRYPT 2015, vol.9462 of LNCS, pp.198–217, Springer, Heidelberg, Dec. 2015.
[2] A. Biryukov, P. Derbez, and L. Perrin, "Differential analysis and meet-in-the-middle attack against round-reduced TWINE," FSE 2015, vol.9054 of LNCS, pp.3–27, Springer, Heidelberg, March 2015.
[3] X. Bonnetain, M.N. Plasencia, and A. Schrottenloher, "Quantum security analysis of AES," IACR Trans. Symm. Cryptol., vol.2019, no.2, pp.55–93, 2019.
[4] P. Derbez and P.A. Fouque, "Exhausting Demirci-Selçuk meet-in-the-middle attacks against reduced-round AES," FSE 2013, vol.8424 of LNCS, pp.541–560, Springer, Heidelberg, March 2014.
[5] P. Derbez and P.A. Fouque, "Automatic search of meet-in-the-middle and impossible differential attacks," CRYPTO 2016, Part II, vol.9815 of LNCS, pp.157–184, Springer, Heidelberg, Aug. 2016.
[6] P. Derbez, P.A. Fouque, and J. Jean, "Improved key recovery attacks on reduced-round AES in the single-key setting," EUROCRYPT 2013, vol.7881 of LNCS, pp.371–387, Springer, Heidelberg, May 2013.
[7] J. Daemen, L.R. Knudsen, and V. Rijmen, "The block cipher Square," FSE 1997, vol.1267 of LNCS, pp.149–165, Springer, Heidelberg, Jan. 1997.
[8] O. Dunkelman, N. Keller, and A. Shamir, "Improved single-key

attacks on 8-round AES-192 and AES-256," ASIACRYPT 2010, vol.6477 of LNCS, pp.158–176, Springer, Heidelberg, Dec. 2010.
[9] X. Dong, L. Li, K. Jia, and X. Wang, "Improved attacks on reduced-round Camellia-128/192/256," CT-RSA 2015, vol.9048 of LNCS, pp.59–83, Springer, Heidelberg, April 2015.
[10] P. Derbez and L. Perrin, "Meet-in-the-middle attacks and structural analysis of round-reduced PRINCE," FSE 2015, vol.9054 of LNCS, pp.190–216, Springer, Heidelberg, March 2015.
[11] J. Daemen and V. Rijmen, "Aes proposal: Rijndael," 1999.
[12] H. Demirci and A.A. Selçuk, "A meet-in-the-middle attack on 8-round AES," FSE 2008, vol.5086 of LNCS, pp.116–126, Springer, Heidelberg, Feb. 2008.
[13] J. Guo, J. Jean, I. Nikolic, and Y. Sasaki, "Meet-in-the-middle attacks on generic Feistel constructions," ASIACRYPT 2014, Part I, vol.8873 of LNCS, pp.458–477, Springer, Heidelberg, Dec. 2014.
[14] J. Guo, J. Jean, I. Nikolic, and Y. Sasaki, "Meet-in-the-middle attacks on classes of contracting and expanding Feistel constructions," IACR Trans. Symm. Cryptol., vol.2016, no.2, pp.307–337, 2016.
[15] H. Gilbert and M. Minier, "A collision attack on 7 rounds of rijndael," AES Candidate Conference, vol.230, pp.230–241, April 2000.
[16] A. Hosoyamada and Y. Sasaki, "Quantum Demiric-Selçuk meet-in-the-middle attacks: Applications to 6-round generic Feistel constructions," SCN 18, vol.11035 of LNCS, pp.386–403, Springer, Heidelberg, Sept. 2018.
[17] D. Kwon, J. Kim, S. Park, S.H. Sung, Y. Sohn, J.H. Song, Y. Yeom, E. Yoon, S. Lee, J. Lee, S. Chee, D. Han, and J. Hong, "New block cipher: ARIA," ICISC 03, vol.2971 of LNCS, pp.432–445, Springer, Heidelberg, Nov. 2004.
[18] R. Li and C. Jin, "Meet-in-the-middle attacks on 10-round AES-256," Des. Codes Cryptogr., vol.80, no.3, pp.459–471, 2016.
[19] L. Li, K. Jia, X. Wang, and X. Dong, "Meet-in-the-middle technique for truncated differential and its applications to CLEFIA and Camellia," FSE 2015, vol.9054 of LNCS, pp.48–70, Springer, Heidelberg, March 2015.
[20] L. Lin and W. Wu, "Meet-in-the-middle attacks on reduced-round Midori64," IACR Trans. Symm. Cryptol., vol.2017, no.1, pp.215–239, 2017.
[21] D. Shi, S. Sun, P. Derbez, Y. Todo, B. Sun, and L. Hu, "Programming the Demirci-Selçuk meet-in-the-middle attack with constraints," ASIACRYPT 2018, Part II, vol.11273 of LNCS, pp.3–34, Springer, Heidelberg, Dec. 2018.

Appendix: Truncated Differential Trails of AES and ARIA

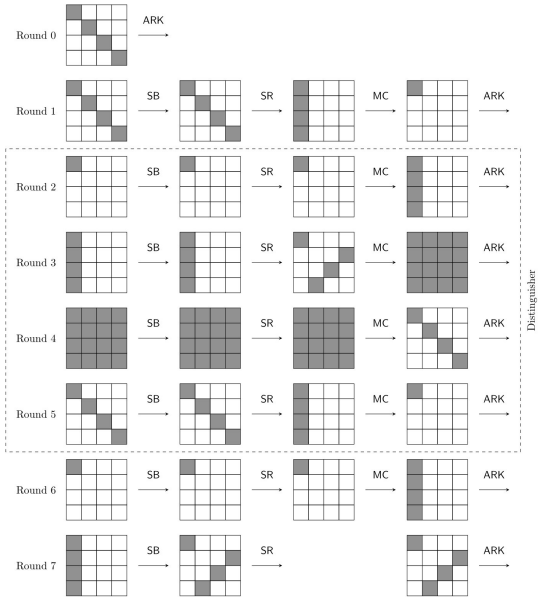

Fig. A · 1 Truncated differential trails of 7-round AES corresponding to Proposition 2 [6].

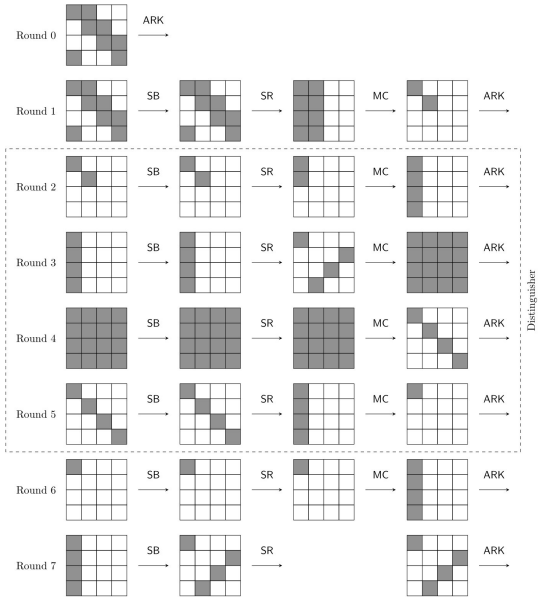

Fig. A · 2 Truncated differential trails of 7-round AES corresponding to Proposition 3 [6].

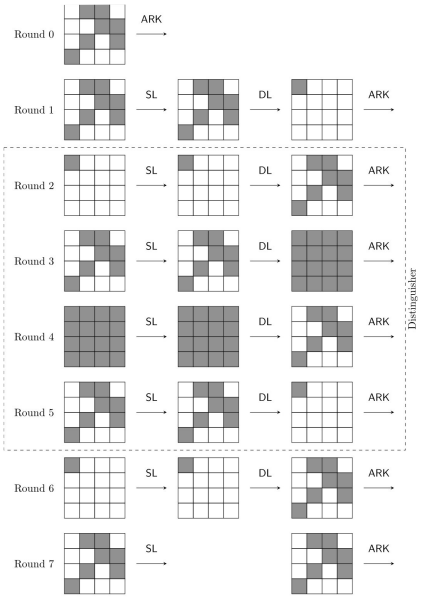

Fig. A · 3 Truncated differential trails of 7-round `ARIA` corresponding to Proposition 4 [1].

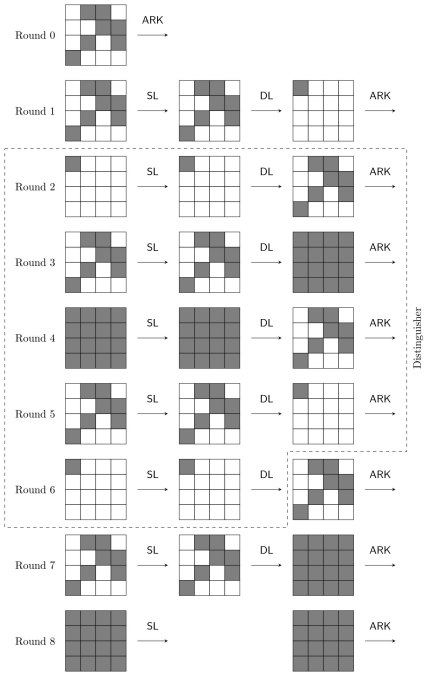

Fig. A · 4 Truncated differential trails of 8-round `ARIA` corresponding to Proposition 5 [1].

Dongjae Lee received the Ph.D. degree in information security from Korea University in 2023. He currently works as a postdoctoral researcher at the Institute of Cyber Security & Privacy (ICSP), Korea University. His research interests include symmetric cryptography and post-quantum cryptography.

Deukjo Hong received the B.S. and M.S. degrees in mathematics and the Ph.D. degree in information security from Korea University, in 1999, 2002, and 2006, respectively. From 2007 to 2015, he was worked at ETRI. He currently holds the position of an Associate Professor with the Department of Information Technology and Engineering, Jeonbuk National University. He was a Guest Researcher at NIST from September 2021 to February 2023. His research interest includes symmetric cryptography.

Jaechul Sung received the Ph.D. degree in mathematics from Korea University in 2002. He worked as a senior researcher of Korea Information Security Agency (KISA) from July 2002 to January 2004. He is now a professor of Department of Mathematics at University of Seoul. His research interests include cryptography, symmetric cryptosystems, hash functions, and MACs.

Seokhie Hong received the M.S. and Ph.D. degrees in mathematics from Korea University in 1997 and 2001, respectively. He worked for SECURITY Technologies Inc. from 2000 to 2004. From 2004 to 2005, he worked as a postdoctoral researcher with COSIC at KU Leuven, Belgium. Since 2005, he has been in Korea University, where he is now working in the Graduate School of Cyber Security. His specialty lies in the area of information security, and his research interests include cryptography, public and symmetric cryptosystems, hash functions, and MACs.