

One More Pair, More Information Gained: Improved Attacks on LowMC with Full S-Box Layers Using Two Plaintext/Ciphertext Pairs

Yimeng Sun[1,2,3], Jiamin Cui[1,2,3(✉)], Shiyao Chen[1,2,3], Meiqin Wang[1,2,3], Longzheng Cui[1,2,3], and Chao Niu[4]

[1] School of Cyber Science and Technology, Shandong University, Qingdao, Shandong, China
{sunyimeng,202321179}@mail.sdu.edu.cn, shiyao.chen@ntu.edu.sg, {cuijiamin,mqwang}@sdu.edu.cn
[2] State Key Laboratory of Cryptography and Digital Economy Security, Shandong University, Qingdao 266237, Shandong, China
[3] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan, China
[4] Ant Group, Beijing, China
niuchao.niu@antgroup.com

Abstract. Motivated by the LowMC cryptanalysis challenge, research in recent years focused more on attacking LowMC in the PICNIC application setting, i.e. an attacker can see only a single plaintext/ciphertext pair. It can be noted that in the security proof of PICNIC, LowMC is required to be secure under two plaintexts, it is thus meaningful to investigate the security of LowMC in this direction. At Crypto 2021, Liu et al. attack all three 4-round LowMC instances adopting full S-Box layers in PICNIC with only two chosen plaintexts. However, the research on cryptanalysis of LowMC using two plaintext/ciphertext pairs is yet far from complete. Previous works using a single known plaintext are better than those with two plaintexts in terms of attack complexity or attacked rounds when considering a comparable success probability.

In this paper, to address such counter-intuitive gaps, we develop an algebraic key-derived attack framework that exploits the algebraic structure of key-derived differences to construct lower-degree equation systems. This reduces the cost of solving the equations and works in the known-plaintext setting, which can be further enhanced with chosen-plaintext attack setting. We also present an improved difference enumeration attack framework. Finally, applying our new attack frameworks to LowMC, we achieve the first 8-round attacks on LowMC, as well as the first attacks on the other three instances. Also, previous attacks on LowMC from 4 to 7 rounds could be improved for almost all three LowMC instances in this paper. These results indicate that one more pair provides more information to improve attacks on LowMC compared with using only a single plaintext.

© The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd. 2027
M. H. Au and M. Yung (Eds.): Inscrypt 2026, LNCS 17226, pp. 85–105, 2027.
https://doi.org/10.1007/978-981-92-7675-2_5

Keywords: LowMC · PICNIC · linearization · algebraic attack · key recovery

1 Introduction

The LowMC family of block ciphers was originally proposed by Albrecht et al. [2] at Eurocrypt 2015, which was designed specifically for use cases in multi-party computation (MPC) and fully homomorphic encryption (FHE). To minimize the number of multiplications, LowMC is designed with a partial substitution-permutation (P-SPN) network with flexible parameters. As such, at CCS 2017, LowMC has been adopted as the underlying symmetric-key primitive in the post-quantum signature scheme PICNIC [7], which was one of the third-round candidates in the NIST Post-Quantum Cryptography competition[1]. In PICNIC, a plaintext/ciphertext pair (pt, ct) will be made open as the public key (verification key) of the signature and encryption key K is used as the secret key (signing key). The security of PICNIC is directly related to the difficulty of recovering K from a single pair (pt, ct). To gain a deeper understanding of the security in PICNIC setting, Rechberger et al. announced a cryptanalysis challenge[2] in 2020.

Soon after the publication of LowMC, the interpolation attack [9] and higher-order differential attack [10] directly made LowMC move to LowMCv2 although with high data complexity. To adapt to the attack setting of low allowable data complexity, at ToSC 2018, Rechberger et al. [14] developed *difference enumeration technique* to analyze the security of LowMCv2 with partial nonlinear layers using 3 or slightly more *chosen* plaintext/ciphertext pairs and promoted the further update of LowMCv2 to LowMCv3. PICNIC has also evolved with LowMC versions, and the latest PICNIC3 [1] adopts a full S-box layer in LowMC.[3] LowMC instances with full S-box layers are *our main focus* in this work.

Related Works. In recent years, motivated by the cryptanalysis challenge, the research on cryptanalysis of LowMC focuses more on key recovery attacks in PICNIC setting, i.e. the attacker can only see a single plaintext/ciphertext pair under *known-plaintext setting*. Following this attack setting, cryptanalysis of LowMC has been a productive line of research [3–5,8,12,15] since 2020. Banik et al. [3] first proposed guess-and-determine attack on 2-round LowMC with full S-box layers, which was later improved to 3 rounds by using 2-stage meet-in-the-middle (MITM) attack with gray-code based approach at Asiacrypt 2021 [4]. In a parallel work at Eurocrypt 2021, Dinur [8] developed a polynomial method, the so-called Dinur's algorithm, to attack all 4-round and two instances among three versions for 5-round LowMC instances with full S-box layers. Later, by exploring different guess strategies to achieve better time-memory trade-offs, Liu et al. [12] significantly improved the memory complexity for 3-/4-round LowMC and Sun

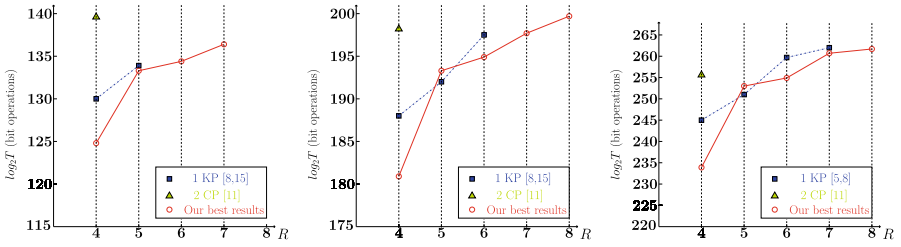
[1] https://csrc.nist.gov/projects/post-quantum-cryptography.
[2] https://lowmcchallenge.github.io.
[3] For simplicity, we will use LowMC instead of LowMCv3 and PICNIC instead of PICNIC3 in the rest of the paper.

et al. [15] reduced the huge memory cost for 5-/6-round LowMC. Recently, Banik et al. [5] enriched Dinur's algorithm with a specific linearization strategy, which yields the first attack on 7-round LowMC instance of 255-bit block size.

Beyond the standard one-pair PICNIC public-key setting, two-pair analyses are motivated by the security proof of PICNIC. The reduction of Chase et al. [7] relies on the PRF security of LowMC under chosen-plaintext access, where two oracle queries are permitted, as noted in footnote 11 and Appendix D. Although such attacks do not directly correspond to key recovery from a PICNIC public key, they examine LowMC in the query model underlying the proof. Moreover, if one additional plaintext/ciphertext pair significantly weakens the algebraic system, this reveals structural weaknesses of the primitive and helps clarify its algebraic security boundary. At Crypto 2021, Liu et al. [11] showed that all three recommended parameters of 4-round LowMC with full S-box layers can be attacked using *two chosen plaintexts* via the difference enumeration attack.

Our Contributions. For all three 4-round LowMC instances, we observe that previous attacks using only a single plaintext [5,8,12] are even better than those with two chosen plaintexts [11] in terms of the complexity and success probability. Meanwhile, no cryptanalysis with two plaintexts for 5-/6-/7-round LowMC is available yet. In this work, we develop an algebraic key-derived attack framework and present an improved difference enumeration attack framework.

The details of our results in this paper are summarized in Table 1.[4] Besides, for a better and more visualized comparison between our results and previous related works, Fig. 1(a), Fig. 1(b) and Fig. 1(c) are provided. In the following, we introduce our two attack frameworks to further explain these results.



(a) Block size and key size of 129 bits. (b) Block size and key size of 192 bits. (c) Block size and key size of 255 bits.

Fig. 1. Comparisons between our results with two plaintext/ciphertext pairs and previous best attacks with one known plaintext or two chosen plaintexts.

New Algebraic Key-Derived Attack Framework. We first reveal an algebraic property of the key-derived difference to build an equation system with

[4] To ensure a fair comparison, the complexity of the success probability lower than 0.5 is presented in Appendix B.

lower algebraic degree under *known-plaintext attack setting*, which is derived from two known plaintext/ciphertext pairs and their differences. Given that the equation system to solve for LowMC is built from a MITM method, one then should balance forward and backward rounds to reach an optimal choice of algebraic degree of the whole equation system. We observe that when the input difference is *chosen to be in a special form*, guessing some key bits at specific positions in the first round can reduce the algebraic degree, which makes the attack more effective for attacks with an *odd number of rounds*.

To verify the correctness of our attacks, we performed experiments on concrete LowMC instances. The code is available and we provide it at https://github.com/Momoqw/LowMC-full.git

Improved Difference Enumeration Attack Framework. The core idea is to avoid exhausting all possible differences in the second round, only part of differences in the second S-box layer will be enumerated. Then by introducing variables for the rest part of S-boxes in the second round and part of S-boxes in the third round, cubic equations can be set up, which will be conducive to derive a linear equation system with fewer variables to solve for the last round. Our improvements upon previous attacks [5,8,11,12] and our known-plaintext attack.

Organization. The rest of the paper is organized as follows. In Sect. 2, some background knowledge is introduced. In Sect. 3, we propose a new algebraic key-derived attack framework for LowMC using two plaintext/ciphertext pairs. Then, an improved difference enumeration attack is presented in Sect. 4. Finally, we conclude the paper in Sect. 5.

2 Preliminaries

2.1 Notation

Due to the various parameter sets of LowMC [2], we use n, k, m, and R to denote the block size, key size, number of S-boxes per round, and total number of rounds, respectively. Let $V = (a_0, a_1, \ldots, a_{i-1})$ be an i-bit vector, and let $V_0|V_1$ denote the concatenation of two vectors V_0 and V_1. For $0 \leq a < b \leq i-1$, $V[a:b]$ denotes the subvector from the a-th bit to the b-th bit of V. We denote $\binom{u}{\leq d} = \sum_{i=0}^{d} \binom{u}{i}$, where $\binom{u}{i}$ is the binomial coefficient. The probability that an event X occurs is denoted by $\Pr(X)$.

2.2 Description of LowMC

LowMC [2] is a family of SPN block ciphers proposed in Eurocrypt 2015. The parameter selection of LowMC is very flexible, and each user can choose the parameter independently and instantiate it.

Let pt denote the plaintext and $K = (k_0, \cdots, k_{n-1}) \in \mathbb{F}_2^n$ denote the master key. Before the first round function, there is a whitening key denoted by $K_0 = L_0(K)$, which is derived by multiplying a random full rank matrix of size $n \times k$

Table 1. Summary of attacks on LowMC instances with full S-box layers.

R	Gray** Code	(n, k, m)	$d = 1$				$d = 2$				
			T	M	Pro.†	Ref.	T‡	M	Pro.	Ref.	KP/CP
4	134.8	(129,129,43)	130	113	0.69	[8]	139.6	-	0.62	[11]	CP
			133.9	86.2	1.00	[5]	124.8	34.3	0.63	Sect. 4	CP
			133.8	42.6	0.69	[12]	126.1	106.3	0.69	Sect. 3.1	KP
4	197.9	(192,192,64)	188	164	0.69	[8]	198.2	-	0.82	[11]	CP
			192.9	123.8	0.64	[5]	205.8	-	0.99	[11]	CP
			195.0	61.2	0.69	[12]	180.9	52.2	0.83	Sect. 4	CP
							188.4	49.2	0.97	Sect. 4	CP
							182.1	154.5	0.69	Sect. 3.1	KP
4	261.0	(255,255,85)	245	218	0.69	[8]	255.6	-	0.848	[11]	CP
			252.9	162.2	0.91	[5]	265.6	-	0.986	[11]	CP
			255.8	79.6	0.69	[12]	233.9	61.3	0.84	Sect. 4	CP
							244.8	63.7	0.99	Sect. 4	CP
							237.9	202.6	0.69	Sect. 3.1	KP
5	135.8	(129,129,43)	135.7	88.0	0.95	[5]	134.4	122.9	0.69	Sect. 3.1	KP
			133.9	45.0	0.69	[15]	133.3	45.0	0.69	Sect. 3.2	CP
5	198.9	(192,192,64)	192	173	0.69	[8]	194.9	180.7	0.69	Sect. 3.1	KP
			196.9	127.2	0.96	[5]	193.3	174.0	0.69	Sect. 3.2	CP
			196.8	65.0	0.69	[15]					
5	262.0	(255,255,85)	251	228	0.69	[8]	254.9	235.2	0.69	Sect. 3.1	KP
			256.9	166.0	0.82	[5]	253.0	228.8	0.69	Sect. 3.2	CP
			258.9	84.6	0.69	[15]					
6	135.8	(129,129,43)	-	-	-	-	134.4	122.9	0.69	Sect. 3.1	KP
6	198.9	(192,192,64)	198.0	128.5	0.52	[5]	194.9	180.7	0.69	Sect. 3.1	KP
			197.5	65.6	0.69	[15]					
6	262.0	(255,255,85)	259.7	168.5	0.52	[5]	254.9	235.2	0.69	Sect. 3.1	KP
			260.5	86.0	0.69	[15]					
7	136.8	(129,129,43)	-	-	-	-	136.4	39.8	0.69	Sect. 3.2	CP
7	199.9	(192,192,64)	-	-	-	-	199.8	183.1	0.69	Sect. 3.1	KP
							197.7	66.0	0.69	Sect. 3.2	CP
7	263.0	(255,255,85)	262.0	170.5	0.57	[5]	261.7	246.1	0.69	Sect. 3.1	KP
							260.7	87.0	0.69	Sect. 3.2	CP
8	199.9	(192,192,64)	-	-	-	-	199.8	183.1	0.69	Sect. 3.1	KP
8	263.0	(255,255,85)	-	-	-	-	261.7	246.1	0.69	Sect. 3.1	KP

* T, M and Pro. represent the time, memory complexity (in $\log_2$) and success probability, respectively. d denotes the number of available plaintext/ciphertext pairs, T is given in the number of bit operations and M is given in bits. KP/CP means known or chosen plaintext setting.

** We take the complexity of Gray-code based exhaustive search technique in [6] as a baseline. The exhaustive search for any R-round LowMC using this technique requires $2 \cdot 2^{\lceil \frac{R}{2} \rceil} \cdot log_2 n \cdot 2^n$ bit-operations and $n^{2^{\lceil \frac{R}{2} \rceil}}$ bits of memory.

† We compare the results when the success probability is greater than 0.5. To ensure a fair comparison, the complexity of the success probability lower than 0.5 is presented in Appendix B. The success probability is not explicitly given in [8,15]. For a more complete comparison, we calculate the corresponding success probability in Section 2.3.

‡ We recalculate the complexity of [11] in the number of bit-operations, e.g. one encryption of r-round LowMC takes about $2Rn^2$ bit-operations.

with the master key K. The i-th round of LowMC consists of four operations: applying the S-box layer SB, multiplying with an invertible matrix $M_i \in \mathbb{F}_2^{n \times n}$, XORing with the n-bit round constants C_i and XORing with n-bit round key K_{i+1} $(0 \leq i \leq R-1)$. In the S-box layer, each 3-bit S-box has Algebraic Normal

Form (ANF): $S(x_0, x_1, x_2) = (z_0, z_1, z_2)$, where z_0, z_1, z_2 satisfy

$$z_0 = x_0 \oplus x_1 x_2,\ z_1 = x_0 \oplus x_1 \oplus x_0 x_2,\ z_2 = x_0 \oplus x_1 \oplus x_2 \oplus x_0 x_1.$$

The round key $K_{i+1} = L_{i+1}(K)$ is derived from a master key K by multiplying with a random invertible $n \times n$ matrix, and M_i and C_i are chosen independently and uniformly for $0 \leq i \leq R-1$.

Let $Enc_K^{(R)} : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an R-round LowMC block cipher defined by

$$Enc_K^{(R)}(pt) : = K_R \oplus F_{R-1}(\cdots F_1(K_1 \oplus F_0(K_0 \oplus pt))), \tag{1}$$

where $K_{i+1} \oplus F_i(\cdot) = K_{i+1} \oplus C_i \oplus M_i \cdot SB(\cdot)$ denotes the $(i+1)$-th round function.

Following the notation of [11], the difference of the $(i+1)$-th input state of SB is denoted by Δ_i and the difference of the corresponding output state is denoted by Δ_i^S. The difference of plaintexts is denoted by Δ_p, i.e. $\Delta_p = \Delta_0$. In our attacks, Δ_i and Δ_i^S will be represented as n-bit vectors.

$$\Delta_p \xrightarrow{K_0} \Delta_0 \xrightarrow{SB} \Delta_0^S \xrightarrow{M_0} \xrightarrow{C_0} \xrightarrow{K_1} \Delta_1 \longrightarrow \cdots \longrightarrow \Delta_{R-1} \xrightarrow{SB} \Delta_{R-1}^S \xrightarrow{M_{R-1}} \xrightarrow{C_{R-1}} \xrightarrow{K_R} \Delta_R.$$

The **compact differential trail** is defined as below:

Definition 1 [11]**.** *A differential trail $\Delta_0 \longrightarrow \Delta_1 \longrightarrow \cdots \longrightarrow \Delta_r$ is called a R-round compact differential trail when all (Δ_j, Δ_j^S) $(0 \leq j \leq R-1)$ and Δ_R are known.*

Difference Enumeration Attack Framework. Based on the notion of compact differential trails, the difference enumeration attack consists of two phases. In the first phase, called the *difference enumeration phase*, the attacker recovers the compact differential trails. In the second phase, called the *key-recovery phase*, the attacker derives the secret key from the recovered compact differential trails.

2.3 Methods for Solving Equation System of LowMC

Obviously, the problem of key-recovery attacks for LowMC with a given plaintext/ciphertext pair can always be reduced to solving a system of n equations in n key variables as

$$E(K) : f_0(K) = f_1(K) = \cdots = f_{n-1}(K) = 0, \tag{2}$$

where the algebraic degree of each f_i is bounded by $\deg(f_i) \leq \mathbf{d}$ and $K = (k_0, k_1, \cdots, k_{n-1}) \in \mathbb{F}_2^n$. Split the n-bit key variables K into two subsets $K_1 = (k_0, k_1, \cdots, k_{n-n_1-1})$ and $K_2 = (k_{n-n_1}, k_{n-n_1+1}, \cdots, k_{n-1})$ of size $n - n_1$ and n_1. Then, $E(K)$ can be rewritten as $E(K_1, K_2)$. In this subsection, we revisit two of the current state-of-the-art methods of solving multivariate equation systems generated by LowMC. This will benefit our later attacks on LowMC.

Dinur's Algorithm. Dinur's algorithm is a general method for solving multivariate equation systems over $\mathbb{F}_2$.

Definition 2 (Isolated Solution [8]). *A solution $\hat{K} = (\hat{K_1}, \hat{K_2})$ to $E(K_1, K_2)$ is called isolated (with respect to the variable partition (K_1, K_2)) if for all $K_2' \neq \hat{K_2}$, $(\hat{K_1}, K_2')$ is not a solution to E.*

Consider the equivalent representation $F(K_1, K_2) = \prod_{i=0}^{n-1}(f_i(K_1, K_2) \oplus 1)$. In order to recover $(\hat{K_1}, \hat{K_2})$, we first take $\ell = n_1 + 1$ different equations at random from $E(K_1, K_2)$ and construct the new equation system $\tilde{E}(K_1, K_2)$, which can be expressed as

$$\tilde{E}(K_1, K_2): \ \tilde{f}_0(K_1, K_2) = \tilde{f}_1(K_1, K_2) = \cdots = \tilde{f}_{\ell-1}(K_1, K_2) = 0.$$

Similarly, we denote the corresponding equivalent representation as $\tilde{F}(K_1, K_2)$. We choose to enumerate isolated solutions of $\tilde{E}$ instead of E and then verify them by E to take advantage of the low degree of $\tilde{F}(K_1, K_2)$. Then we can efficiently compute all n_1+1 partial sums to recover the isolated solutions. More details can be found in [8]. To reduce the cost of testing solutions, we perform N different choices for the equation system $\tilde{E}(x)$. The solution is retained and verified once it occurs more than once, so the success probability of Dinur's algorithm can be estimated as

$$Pr_{[\text{Din21}]} = \sum_{i=2}^{N} \binom{N}{i} \cdot (\frac{1}{2})^i \cdot (\frac{1}{2})^{N-i}.$$

Therefore, $Pr_{[\text{Din21}]} \approx 0.69$ when $N = 4$ and $Pr_{[\text{Din21}]} \approx 1$ when $N = 16$.

The total time complexity can be evaluated in bit operations as

$$N \cdot (2 \cdot \mathbf{d} \cdot \log_2(n) \cdot 2^{n_1} \cdot \binom{n - n_1}{\leq d_{\tilde{F}} - n_1 + 1}) + N \cdot (n_1 + 1) \cdot (n - n_1) \cdot 2^{n - n_1}, \quad (3)$$

where $d_{\tilde{F}} = (n_1 + 1) \cdot \mathbf{d}$. We use $T_{[\text{Din21}]}(n, n_1, \mathbf{d}, N)$ instead of Eq. (3) for convenience. Using Dinur's memory-efficient Möbius transform, the memory complexity of polynomial evaluation can be reduced to $M_{[\text{Din21}]} = 2 \cdot N \cdot (n_1 + 1) \cdot \binom{n-n_1}{\leq d_{\tilde{F}} - n_1 + 1}$ bits.

Improving Dinur's Algorithm for LowMC Using Guess Strategy. In [15], the authors improved the polynomial methods for LowMC equation systems based on the guess strategy in [12]. Assume an all-zero plaintext and consider the i-th S-box $S(k_{3i}, k_{3i+1}, k_{3i+2}) = (z_{3i}, z_{3i+1}, z_{3i+2})$ in the first round, $0 \leq i \leq m - 1$. Let $y_i = z_{3i}$. If we guess two input bits as $k_{3i+1} = g_{2i}$ and $k_{3i+2} = g_{2i+1}$, the 3-bit key variable takes value from the set

$$B^{g_{2i}g_{2i+1}} = \{(y_i \oplus g_{2i}g_{2i+1}, g_{2i}, g_{2i+1}) \mid y_i \in \{0, 1\}\},$$

where $g_{2i}, g_{2i+1} \in \{0, 1\}$. Then, after guessing $2m$ bits $G = (g_0, g_1, \cdots, g_{2m-1}) \in \{0, 1\}^{2m}$, the first round can be linearized. We obtain n equations of degree $\mathbf{d}'$ for LowMC instance over the partial space $K \in B^G = B^{g_0 g_1} \times \cdots \times B^{g_{2m-2} g_{2m-1}}$ and $\mathbf{d}' < \mathbf{d}$. Denote the equation system by $E_G(K)$ as

$$E_G(K): f_{G,0}(K) = f_{G,1}(K) = \cdots = f_{G,n-1}(K) = 0.$$

For any given $G = (g_0, g_1, \cdots, g_{2m-1})$, there is a one-to-one correspondence between $K = (k_0, k_1, \cdots, k_{3m-1}) \in B^G$ and $Y = (y_0, y_1, \cdots, y_{m-1})$.

Consequently, we compute Y in B^G using operations similar to those in Dinur's algorithm. More details can be found in [15]. To reduce the cost of testing solutions, we perform four different choices for the equation system $\tilde{E}_G(K)$. The solution is retained and verified once it occurs more than once, so the success probability can be estimated as

$$Pr_{[\text{SCW23}]} = \sum_{i=2}^{4} \binom{4}{i} \cdot (\frac{1}{2})^i \cdot (\frac{1}{2})^{4-i} \approx 0.69.$$

The total time complexity can be evaluated in bit operations as

$$2^{2n/3} \cdot (\binom{N_u}{D} \cdot 2rn^2 \cdot 2^{n_1/3} + 4 \cdot (n_1/3+1) \cdot N_u \cdot \sum_{i=0}^{D-1} \binom{N_u - 1}{i} + D \cdot 2^{N_u}), \quad (4)$$

where $N_u = (n-n_1)/3$ and $D = \mathbf{d}' \cdot l - n_1/3$. We use $T_{[\text{SCW23}]}(N_u, n_1, D)$ instead of Equation (4) for convenience. The memory complexity can be estimated as

$$M_{[\text{SCW23}]} = \binom{N_u}{D} \cdot n \cdot 2^{n_1/3} + 4 \cdot (n_1/3+1) \cdot 2^{N_u}.$$

3 Algebraic Key-Derived Attack Framework for LowMC Using Two Plaintext/Ciphertext Pairs

In the previous works, the problem of the key recovery of LowMC under a single known plaintext/ciphertext pair can be reduced to the problem of solving linear/nonlinear equation system using some polynomial methods. The guess strategy is commonly used to reduce the number of variables or the algebraic degree of the equations and therefore achieves a better time-memory trade-off [4,15]. In this section, we show how to leverage the relationship between two plaintext/ciphertext pairs to reduce the algebraic degree of the equation system without extra guesses. In Sect. 3.1, we propose a new MITM algebraic attack framework under two known plaintext/ciphertext pairs. To further optimize the attack for odd number of rounds, we restrict the input difference into a special form and explore different active positions. Based on this, we propose two advanced MITM algebraic attack frameworks in Sect. 3.2.

3.1 MITM Algebraic Key-Derived Attack Framework

Let X denote the plaintext. According to Eq. 1, $F_i(\cdot)$ is a quadratic function and the general expression for the output bit of the quadratic round function can be expressed by

$$F_i(U) = A \oplus \sum_{j=0}^{n-1} B_j u_j \oplus \sum_{j=0}^{n-1} \sum_{t=j+1}^{n-1} C_{j,t} u_j u_t,$$

where $U = (u_0, \ldots, u_{n-1})$, and $A, B_{i,j}, C_{i,j,t} \in \mathbb{F}_2$. Denote the upper bound of the algebraic degree for K in $Enc_K^{(R)}(X)$ by $\delta_K(Enc_K^{(R)}(X))$. Obviously, we have $\delta_K(Enc_K^{(R)}(X)) \leq 2^R$. Consider another plaintext X', we have

$$\Delta Enc_K^{(R)} = Enc_K^{(R)}(X) \oplus Enc_K^{(R)}(X').$$

The following lemma proves the upper bound of $\delta_K(\Delta Enc_K^{(R)})$.

Lemma 1. *Let $Enc_K^{(R)} \colon \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an R-round LowMC block cipher defined above. Given two plaintext X and X', we have $\delta_K(\Delta Enc_K^{(R)}) \leq 2^R - 1$.*

A detailed proof of Lemma 1 can be found in the Appendix A. Using the above strategy, we can construct the equation system for arbitrary two plaintext/ciphertext pairs with a MITM method. For an even number of rounds $R = 2\rho$, we consider the rounds 1 to ρ, the input difference of $(\rho+1)$-th round Δ_ρ can be expressed by an equation system of degree $2^\rho - 1$ in key variables. Similarly, consider rounds 2ρ to $\rho + 1$, the input difference of $(\rho+1)$-th round Δ_ρ can also be expressed as an equation system of degree $2^\rho - 1$ in key variables. Then we can obtain n equations of degree $2^\rho - 1$ in n variables by simply equating the two sets of equations. For an odd number of rounds $R = 2\rho + 1$, we can obtain n equations of degree $2^{\rho+1} - 1$ in n variables similarly.

Complexity Evaluation. Once we have this equation system, we can use Dinur's algorithm to solve it. Based on Eq. (3) and choosing $N = 4$, the time complexity can be evaluated in $T_{\mathrm{KP}} = T_{[\mathrm{Din21}]}(n, n_1, \mathbf{d}, 4)$ bit operations, where $\mathbf{d} = 2^\rho - 1$ when $R = 2\rho$ and $\mathbf{d} = 2^{\rho+1} - 1$ when $R = 2\rho + 1$. The memory complexity can be evaluated in $M_{\mathrm{KP}} = 8 \cdot (n_1 + 1) \cdot \binom{n-n_1}{\leq \mathbf{d}\cdot(n_1+1)-n_1+1}$ bits.

3.2 Advanced MITM Algebraic Attack Framework

The meet-in-the-middle approach produces equations of degree $2^\rho - 1$ both the forward and backward directions for even rounds. But for odd rounds, the extra round causes a rapid increase in the time complexity due to the unbalance of the algebraic degree. In this subsection, we further optimize the attack framework for odd rounds when enhancing to chosen-plaintext setting. This improvement is based on a new observation of the S-box.

Let $(x_0, x_1, x_2) \oplus (x_0', x_1', x_2') = (\Delta x_0, \Delta x_1, \Delta x_2)$, where x_i and x_i' are known for $i = \{0, 1, 2\}$. For $S(x_0 \oplus k_0, x_1 \oplus k_1, x_2 \oplus k_2) = (z_0, z_1, z_2)$ and $S(x_0' \oplus k_0, x_1' \oplus k_1, x_2' \oplus k_2) = (z_0', z_1', z_2')$, we have

$$\begin{aligned}
\Delta z_0 &= (\Delta x_0 \oplus \Delta x_1 \Delta x_2) \oplus \Delta x_1 (x_2 \oplus k_2) \oplus \Delta x_2 (x_1 \oplus k_1), \\
\Delta z_1 &= (\Delta x_0 \oplus \Delta x_1 \oplus \Delta x_0 \Delta x_2) \oplus \Delta x_0 (x_2 \oplus k_2) \oplus \Delta x_2 (x_0 \oplus k_0), \\
\Delta z_2 &= (\Delta x_0 \oplus \Delta x_1 \oplus \Delta x_2 \oplus \Delta x_0 \Delta x_1) \oplus \Delta x_0 (x_1 \oplus k_1) \oplus \Delta x_1 (x_0 \oplus k_0).
\end{aligned}$$

Then if $(\Delta x_0, \Delta x_1, \Delta x_2) = (0, 0, 1)$ and guess $(k_0, k_1) = (k_0^*, k_1^*)$, we have $(\Delta z_0, \Delta z_1, \Delta z_2) = (x_1 \oplus k_1^*, x_0 \oplus k_0^*, 1)$. Indeed, by choosing a special form of input

difference, i.e. only one active bit, the output difference is determined by guessing 2 key bits. Otherwise, if $(\Delta x_0, \Delta x_1, \Delta x_2) = (0, 1, 1)$ and $(k_0, k_1) = (k_0^*, k_1^*)$, we obtain $(\Delta z_0, \Delta z_1, \Delta z_2) = (1 \oplus x_2 \oplus k_2 \oplus x_1 \oplus k_1^*, 1 \oplus x_0 \oplus k_0^*, x_0 + k_0^*)$. An undetermined variable k_2 appears. The observation can be summarised as follows.

Observation 1. *Let $S(x_0 \oplus k_0, x_1 \oplus k_1, x_2 \oplus k_2) = (z_0, z_1, z_2)$ and $S(x_0' \oplus k_1, x_1' \oplus k_1, x_2' \oplus k_2) = (z_0', z_1', z_2')$. If the input difference $(\Delta x_0, \Delta x_1, \Delta x_2)$ has only one active bit, then the output difference $(\Delta z_0, \Delta z_1, \Delta z_2)$ is determined by guessing two key bits which correspond to the 2 inactive bits of the S-box.*

By exploring the different positions of active bits, we propose the following two advanced MITM algebraic attack frameworks as depicted in Fig. 2.

Activate Only One Bit. We consider the case that there is only one active S-box in the first round and this active S-box has only one active bit. Then the input difference of the 2nd round is determined by guessing 2 key bits according to Observation 1. Because we did not guess the input value of the inactive S-box in the first round, the input value of the key variable in the second round is quadratic, i.e. $\delta_K(Enc_K^{(1)}(X)) = 2$. According to Appendix A, we know that

$$\delta_K(\Delta Enc_K^{(R)}) = \max(\delta_K(\Delta Enc_K^{(R-1)}) + \max(\delta_K(Enc_K^{(R-1)}(X)), 1), 2\delta_K(\Delta Enc_K^{(R-1)})). \quad (5)$$

Let $R = 2$, it is easy to know that $\delta_K(\Delta Enc_K^{(2)}) = 0 + 2 = 2$. Recursively, we have

$$\delta_K(\Delta Enc_K^{(R)}) = 0 + 2 + 4 + \cdots + 2^{R-1} = 2^R - 2.$$

The observation can be summarized as follows.

Observation 2. *Let $Enc_K^{(R)} : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an R-round LowMC block cipher defined above. Select an input difference Δ_0 such that it has only one active bit. Given two plaintexts X and X' such that $X \oplus X' = \Delta_0$, we have $\delta_K(\Delta Enc_K^{(R)}) = 2^R - 2$ after guessing two key bits which correspond to the 2 inactive bits of the active S-box.*

Complexity Evaluation. Then we can obtain n equations of degree $2^{\rho+1} - 2$ in $n-2$ variables for an odd number of rounds $R = 2\rho+1$. We use Dinur's algorithm to solve the equation system we obtained. Based on Eq. (3) and choosing $N = 4$, the time complexity can be evaluated in $T_{\mathrm{CP}_1} = 4 \cdot T_{[\mathrm{Din21}]}(n-2, n_1, 2^{\rho+1} - 2, 4)$ bit operations. The memory complexity can be evaluated in $M_{\mathrm{CP}_1} = 8 \cdot (n_1 + 1) \cdot \binom{n-2-n_1}{\leq (2^{\rho+1}-2)\cdot(n_1+1)-n_1+1}$ bits.

For $n = 255$, $R = 5$, we have $\mathbf{d} = 2^{\rho+1} - 2 = 6$. If we choose $n_1 = 16$ and $\ell = n_1 + 1 = 17$, we get $T_{\mathrm{CP}_1} \approx 2^{253.0}$ bit operations and $M_{\mathrm{CP}_1} \approx 2^{228.9}$ bits.

Activate One Bit in Each Active S-box. Select an input difference such that the input difference of each active S-box has only one active bit. For active S-boxes, we guess two key bits which correspond to the 2 inactive bits of the

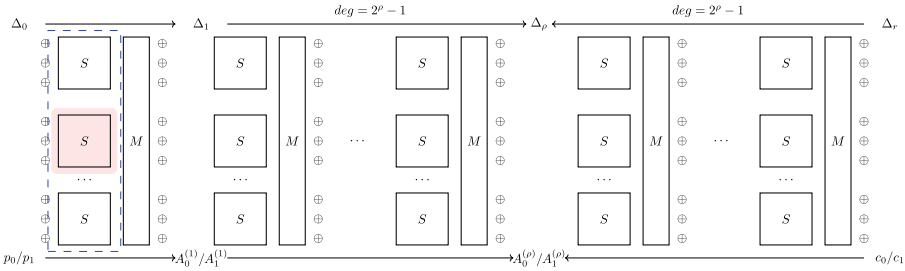


Fig. 2. The attack setup for odd rounds for LowMC instances. The S-box shown in red is linearized when we activate only one bit. The S-box shown in blue is linearized when we activate one bit in each active S-box. (Color figure online)

S-box. For inactive S-boxes, we just guess two random input bits. According to Eq. (5), we have $\delta_K(\Delta Enc_K^{(2)}) = 0 + 1 = 1$. Recursively, we have

$$\delta_K(\Delta Enc_K^{(R)}) = 0 + 1 + 2 + \cdots + 2^{R-2} = 2^{R-1} - 1.$$

The observation can be summarised as follows.

Observation 3. *Let $Enc_K^{(R)} : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an R-round LowMC block cipher defined above. Select an input difference Δ_0 such that it has only one active bit for each active S-box. Given two plaintexts X and X' such that $X \oplus X' = \Delta_0$, we guess:*

- *two key bits corresponding to the inactive bits of the active S-box;*
- *two random key bits of the inactive S-box.*

Then, we have $\delta_K(\Delta Enc_K^{(R)}) = 2^{R-1} - 1$.

Then we can obtain n equations of degree $2^{\rho-1} - 1$ in $n/3$ variables for an odd number of rounds $R = 2\rho + 1$ as depicted in Fig. 2.

Complexity Evaluation. We use the two polynomial methods introduced in Sect. 2.3 to achieve a better time-memory trade-off.

Key Recovery Using Dinur's Algorithm. Based on Eq. (3) and choosing $N = 4$, the time complexity can be evaluated in as $T_{\mathrm{CP}_2^{(1)}} = 2^{2n/3} \cdot T_{[\mathrm{Din21}]}(n/3, n_1, 2^\rho - 1, 4)$ bit operations. The memory complexity can be evaluated in bits as

$$M_{\mathrm{CP}_2^{(1)}} = 8 \cdot (n_1 + 1) \cdot \binom{n/3 - n_1}{\leq (2^\rho - 1) \cdot (n_1 + 1) - n_1 + 1}.$$

For $n = 255$, $R = 7$, we have $\mathbf{d} = 2^\rho - 1 = 7$. If we choose $n_1 = 4$ and $\ell = n_1 + 1 = 5$, we get $T_{\mathrm{CP}_2^{(1)}} \approx 2^{261.8}$ bit operations and $M_{\mathrm{CP}_2^{(1)}} \approx 2^{81.6}$ bits.

Key Recovery Using [15]**.** Using the polynomial method in [15], we can further optimize time complexity for some instances. After guessing the $2n/3$ key

variables, the whole key space is effectively divided into several partial spaces. Suppose that the input of the i-th S-box is $(x_{3i}+k_{3i}, x_{3i+1}+k_{3i+1}, x_{3i+2}+k_{3i+2})$ and the input difference of the i-th S-box is $(\Delta x_{3i}, \Delta x_{3i+1}, \Delta x_{3i+2}) = (0,0,1)$. If we guess two key variables as $(k_{3i}, k_{3i+1}) = (g_{2i}, g_{2i+1})$, then the 3-bit key variable takes value from the set

$$B^{g_{2i}g_{2i+1}} = \{g_{2i}, g_{2i+1}, x_{3i} \oplus g_{2i} \oplus x_{3i+1} \oplus g_{2i+1} \oplus x_{3i+2} \\ \oplus y_i \oplus (x_{3i} \oplus g_{2i})(x_{3i+1} \oplus g_{2i+1}) \mid y_i \in \{0,1\}\}.$$

The output difference can be computed as $(\Delta z_{3i}, \Delta z_{3i+1}, \Delta z_{3i+2}) = (x_{3i+1} + g_{2i+1}, x_{3i} + g_{2i}, 1)$, where x_{3i} and x_{3i+1} are known. So, after guessing $2m$ bits $G = (g_0, g_1, \cdots, g_{2m-1}) \in \{0,1\}^{2m}$, we obtain n equations of degree $d = 2^\rho - 1$ for $2\rho + 1$ round LowMC instance over the partial space. Based on Equation (4), the time complexity can be estimated as $T_{\mathrm{CP}_2^{(2)}} = T_{[\mathrm{SCW23}]}(N_u, n_1, D)$ bit operations. The memory complexity can be estimated as $M_{\mathrm{CP}_2^{(2)}} = \binom{N_u}{\leq D} \cdot n \cdot 2^{n_1/3} + 4 \cdot (n_1/3+1) \cdot 2^{N_u}$ bits, where $N_u = (n-n_1)/3$ and $D = (2^\rho - 1) \cdot (n_1/3 + 1) - n_1/3$ bits.

For $n = 255$, $R = 7$, we have $\mathbf{d}' = 2^\rho - 1 = 7$. If we choose $n_1 = 3$ and $l = n_1/3 + 1 = 2$, we get $T_{\mathrm{CP}_2^{(2)}} \approx 2^{260.7}$ bit operations and $M_{\mathrm{CP}_2^{(2)}} \approx 2^{87.0}$ bits.

In Appendix C, we compare the Advanced MITM Algebraic Attack Framework with the previous works [4,8,15].

4 Advanced Difference Enumeration Attack Framework for the Full S-Box Layer

In this section, we revisit the difference enumeration attack to explore the potential for further optimization of 4-round attacks. In [11], the problems of enumerating compact differential trails and retrieving the full key for the full S-box layer are reduced to the problem of solving linear equation systems using guess-and-determine strategy. However, these enumerated variables denote the difference or intermediate states and there are some inherent relation inside them. We use T_d and T_k to represent the time complexity of the difference enumeration phase and key-recovery phase, respectively. N_d represents the number of potentially correct compact differential trails after the difference enumeration phase.

4.1 Difference Enumeration Phase

As mentioned above, it is necessary to develop a new technique to construct the nonlinear equation system for the difference enumeration phase. One natural way to reduce the cost is to exploit cubic equations instead of enumerating directly according to the following observation.

Observation 4 [13]**.** *Denote the input difference and output difference of the 3-bit LowMC S-box by* $(\Delta x_0, \Delta x_1, \Delta x_2)$ *and* $(\Delta z_0, \Delta z_1, \Delta z_2)$*, respectively. The*

following 2 cubic equations are sufficient to describe its DDT:

$$(1 \oplus \Delta x_0)(1 \oplus \Delta x_1)(1 \oplus \Delta x_2) = (1 \oplus \Delta z_0)(1 \oplus \Delta z_1)(1 \oplus \Delta z_2), \quad (6)$$

$$(1 \oplus \Delta x_0)(1 \oplus \Delta x_1)(1 \oplus \Delta x_2) = \Delta x_0 \Delta z_0 \oplus \Delta x_1 \Delta z_1 \oplus \Delta x_2 \Delta z_2 \oplus 1. \quad (7)$$

For convenience, we denote by $a^{(i-1)}$ and $b^{(i-1)}$ the number of active S-boxes and inactive S-boxes in the i-th round respectively, where $1 \leq i \leq 4$. Since only one S-box is active in the first round, we have $a^{(0)} = 1$ and $b^{(0)} = n/3 - 1$. We set lower bounds $b^{(1)}_{min}$ and $b^{(3)}_{min}$ for the number of inactive S-boxes in the 2nd and 4th rounds, respectively. We only care about the difference trails that satisfy $b^{(1)} \geq b^{(1)}_{min}$ and $b^{(3)} \geq b^{(3)}_{min}$. The success probability will be discussed later. An overview of the new attack framework is depicted in Fig. 3.

We still enumerate the 4 possible differences in the first round. In the second round, we introduce $2(m - b^{(1)}_{min})$ variables $(u_0, \cdots, u_{2(m-b^{(1)}_{min})-1})$ to represent Δ_1^S according to Observation 5. For the first t S-boxes in the 3rd round, we can introduce $3t$ variables $(v_0, \cdots, v_{3t-1})$ to represent $\Delta_2^S[0\colon 3t-1]$, i.e. the first $3t$ bits of Δ_2^S. Based on Observation 4, we can obtain the corresponding $2t$ cubic equations. Notice that these $2(m - b^{(1)}_{min})$ variables $(u_0, \cdots, u_{2(m-b^{(1)}_{min})-1})$ represent the output difference of $(m - b^{(1)}_{min})$ S-boxes, so they take value from a constrained space. We enumerate the $2^{1.86e}$ output differences[5] of the first e S-boxes. For each possible transition, we can construct $2t$ cubic equations of $2(m - e - b^{(1)}_{min}) + 3t$ variables, which can be solved efficiently using Dinur's algorithm [8]. For each solution, the correctness can be checked via DDT.

Now only the output differences of the last $(m-t)$ S-boxes in the 3rd round are not determined. Since the input difference is known, it is sufficient to introduce $2(m-t)$ variables $(w_0, \cdots, w_{2(m-t)-1})$ to represent the output difference. Based on Observation 5, we can construct at least $m - b^{(3)}_{min} + 3b^{(3)}_{min}$ linear equations of $2(m-t)$ variables. If $2(m-t) \leq m + 2b^{(3)}_{min}$, i.e. $t \geq m/2 - b^{(3)}_{min}$, we can easily solve them by Gaussian elimination. If $t < m/2 - b^{(3)}_{min}$, it is necessary to enumerate all the $2^{m-2t-2b^{(3)}_{min}}$ variables.

Observation 5 [11]. *For each non-zero input difference $(\Delta x_0, \Delta x_1, \Delta x_2)$, its valid output differences form an affine space of dimension 2. A similar property also applies to the inverse of the S-box.*

The whole procedure can be described as follows.

Step 1. Choose an input difference Δ_0 such that there will be only one active S-box in the first round, i.e. $a^{(0)} = 1$ and $b^{(0)} = n/3 - 1$.

Step 2. Encrypt a plaintext pair whose XOR difference is Δ_0 for R rounds and obtain the corresponding XOR difference Δ_R of the ciphertexts.

Step 3. Enumerate 4 possible state differences in the first round. For each $\Delta_0^S[0 : 3m-1]$:

[5] The average number of reachable output differences over the S-box for a uniformly randomly chosen input difference is around $2^{1.86}$ [14].

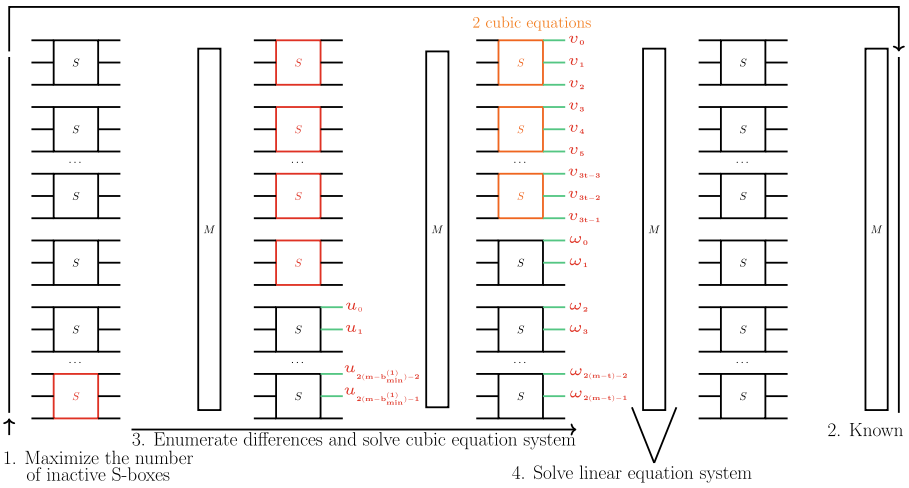


Fig. 3. The improved difference enumeration attack framework for the full S-box layer. The S-boxes used to enumerate differences are colored in red. The S-boxes used to build cubic equation systems are colored in orange. The green line indicates where the variable is introduced. (Color figure online)

3.1 Introduce $2(m - b_{min}^{(1)})$ variables $(u_0, \cdots, u_{2(m-b_{min}^{(1)})-1})$ to represent Δ_1^S and introduce $3t$ variables $(v_0, \cdots, v_{3t-1})$ to represent $\Delta_2^S[0\colon 3t-1]$.

3.2 Enumerate $2^{1.86e}$ output differences for the first $2e$ variables in $(u_0, \cdots, u_{2(m-b_{min}^{(1)})-1})$. For each possible transition, construct $2t$ cubic equations of $2(m - e - b_{min}^{(1)}) + 3t$ variables. Then we perform Dinur's algorithm on them and find the solution. The correctness can be checked via DDT. For each possible $\Delta_1^S[0\colon 3m-1]|\Delta_2^S[0\colon 3t-1]$, move to Step 4.

Step 4. For each possible $\Delta_1^S[0\colon 3m-1]|\Delta_2^S[0\colon 3t-1]$, we introduce $2(m-t)$ variables $(w_0, \cdots, w_{2(m-t)-1})$ to represent $\Delta_2^S[3t : 3m-1]$. Compute the state difference backwards from Δ_r to Δ_3^S. Based on Observation 5, we construct at least $m+2b_{min}^{(3)}$ linear equations. If $2(m-t) \leq m+2b_{min}^{(3)}$, i.e. $t \geq m/2 - b_{min}^{(3)}$, we can solve them by Gaussian elimination. Otherwise, we need to enumerate $(m - 2t - 2b_{min}^{(3)})$ variables. The correctness can be checked via DDT.

Complexity Evaluation. In Step 3, according to Eq. (3) and choosing $N = 16$, the time complexity can be evaluated in bit operations as

$$T_1 = 4 \cdot 2^{1.86 \cdot e} \cdot T_{[\text{Din21}]}(2\,m - 2b_{min}^{(1)} - 2e + 3t, uu_1, 3, 16). \tag{8}$$

We use N_1 to represent the number of solutions of the cubic equation system in Step 3.2. Note that the probability of Eqs. (6) and (7) both being true is $\frac{29}{64}$,

which has also been experimentally verified. Then we can estimate N_1 as

$$N_1 = 2^{2\,m-2b_{min}^{(1)}-2e+3t} \cdot (\frac{29}{64})^t.$$

In Step 4, if $t \geq m/2 - b_{min}^{(3)}$, there is at most one solution by using Gaussian elimination. The time complexity in this step is $T_2 = (m + 2b_{min}^{(3)})^3$. Otherwise, the time complexity in this step is $T_2 = 2^{m-2b_{min}^{(3)}-2t} \cdot (m + 2b_{min}^{(3)})^3$. So the total time complexity of the difference enumeration phase is $T_d = T_1 + 4 \cdot 2^{1.86e} \cdot N_1 \cdot T_2$ bit operations.

If using Dinur's memory-efficient Möbius transform, the memory complexity is evaluated in bits as

$$M = 8(uu_1 + 1) \begin{pmatrix} 2\,m - 2b_{min}^{(1)} - 2e + 3t - uu_1 \\ \leq 3 \cdot (uu_1 + 1) - uu_1 + 1 \end{pmatrix} + (m + 2b_{min}^{(1)})^2.$$

The time complexity is affected by both $(b_{min}^{(1)}, b_{min}^{(3)})$ and the choice of (e, t, uu_1), which collectively define the theoretical upper bound on time complexity. In Table 4, we select the best (e, t, uu_1) and $(b_{min}^{(1)}, b_{min}^{(3)})$ for different instances.

Probability of an Isolated Solution. Since we use Dinur's algorithm to solve the equation system in Step 3, it is necessary to demonstrate that the system admits an isolated solution with high probability. The variable partition associates a solution to the equation system with $2^{uu_1} - 1$ different assignments, where each possible assignment satisfies the system of $2t$ equations with probability $(\frac{29}{64})^t$. Therefore, the probability that the system admits an isolated solution is at least $1 - 2^{uu_1} \cdot (\frac{29}{64})^t$, which is typically very close to 1 according to Table 4. A more detailed analysis is thoroughly explained on page 14 of reference [8].

4.2 Key-Recovery Phase

In the key-recovery phase, we show how to retrieve the full key from the recovered compact differential trails. Due to the influence of the full S-box layer, the number of compact differential trails increases rapidly, $N_d = 2^{1.86m \cdot R - n}$. To keep $N_d T_k < 2^k$, it is very important to further optimize T_k. Instead of the simple guess-and-determine strategy, we utilize more information of the S-boxes to construct the linear equation system based on Observation 6.

Observation 6 [11]. *For each valid non-zero difference transition $(\Delta x_0, \Delta x_1, \Delta x_2) \rightarrow (\Delta z_0, \Delta z_1, \Delta z_2)$, the inputs conforming to such a difference transition will form an affine space of dimension 1. In addition, (z_0, z_1, z_2) becomes linear in (x_0, x_1, x_2), i.e. the S-box is freely linearized for a valid non-zero difference transition. A similar property also applies to the inverse of the S-box.*

Set two counters **a** and **b**. The whole procedure can be described as follows. For each possible differential trail, start from the ciphertext and check the S-boxes of the last two rounds in the backward direction round by round and one

by one. If the S-box is active, increase **a** by 1. According to Observation 6, we can obtain 2 linear equations in terms of the key and the intermediate variables. Otherwise, increase **b** by 1 and introduce 3 intermediate variables to represent its input. Then, check the S-boxes of the 2nd round one by one. If the S-box is active, increase **a** by 1. According to Observation 6, we can obtain 2 linear equations in terms of the key and the intermediate variables.

Once the condition $2\mathbf{a} \geq 3\mathbf{b} + n$ is met during this process, we only need to solve $2\mathbf{a}$ linear equations to uniquely determine the key. Otherwise, applying Gaussian elimination on the first $2\mathbf{a}$ linear equations will allow us to obtain $3\mathbf{b} + n - 2\mathbf{a}$ free variables and enumerate them. The correctness of the solution can be easily verified via a single plaintext/ciphertext pair.

Choice of the Threshold. Obviously, we have $\mathbf{a} = \sum_{i=1}^{i=3} a^{(i)}$ and $\mathbf{b} = b^{(2)} + b^{(3)}$. Since the time complexity of the key-recovery phase is affected by the number of inactive S-boxes **b**, we choose the threshold $b_{max}^{(i)}$ for each of the last three rounds such that the sum of the number of inactive S-boxes in the last three rounds reaches $\sum_{i=1}^{i=3} b_{max}^{(i)}$ is the worst case, i.e. when the sum of the number of inactive S-boxes in the last three rounds is less than $\sum_{i=1}^{i=3} b_{max}^{(i)}$, the time complexity will not be higher. At this time, the minimum number of active S-boxes for each of the last three rounds is $a_{min}^{(i)} = m - b_{max}^{(i)}$. In other words, the time complexity is highest when $\mathbf{a} = \sum_{i=1}^{i=3} a_{min}^{(i)}$ and $\mathbf{b} = b_{max}^{(2)} + b_{max}^{(3)}$.

To ensure that the success probability is not affected too much, we choose the threshold $b_{max}^{(i)} = \lfloor 0.26m \rfloor$ based on the well-known statistical property

$$\Pr(b^{(i)} \leq b_{max}^{(i)}) = \sum_{i=0}^{\lfloor 0.26\,m \rfloor} \binom{m}{i} \times \left(\frac{7}{8}\right)^{m-i} \times \left(\frac{1}{8}\right)^{i} \approx 1,$$

where $1 \leq i \leq 3$. Considering that $b_{min}^{(1)}$ and $b_{min}^{(3)}$ are set in the difference enumeration phase, the success probability can be calculated as

$$Pr = \sum_{k=\sum_{i=1}^{i=3} a_{min}^{(i)}}^{3\,m-b_{min}^{(1)}-b_{min}^{(3)}} \sum_{j=a_{min}^{(2)}}^{m} \sum_{i=a_{min}^{(1)}}^{m-b_{min}^{(1)}} \binom{m}{i} \cdot \binom{m}{j} \cdot \binom{m}{k-j-i} \cdot \left(\frac{7}{8}\right)^{k} \cdot \left(\frac{1}{8}\right)^{3\,m-k},$$

where $a_{min}^{(3)} \leq k - j - i \leq m - b_{min}^{(3)}$.

The specific parameter choices are provided in Appendix D.

Complexity Evaluation of Key-Recovery Phase. As mentioned above, the time complexity is the worst case when $\mathbf{a} = \sum_{i=1}^{i=3} a_{min}^{(i)}$ and $\mathbf{b} = b_{max}^{(2)} + b_{max}^{(3)}$. If $2 \cdot \mathbf{a} \geq 3 \cdot \mathbf{b} + n$, we only need to solve $2 \cdot \mathbf{a}$ linear equations and $T_k = (2 \cdot \mathbf{a})^3$. Otherwise, there will be $3 \cdot \mathbf{b} + n - 2 \cdot \mathbf{a}$ free variables and $T_k = 2^{3 \cdot \mathbf{b} + n - 2 \cdot \mathbf{a}} \cdot (2 \cdot \mathbf{a})^3$. Considering $b_{min}^{(1)}$ and $b_{min}^{(3)}$, the estimated number of differential trails that can be recovered is

$$N_d = 4 \cdot 2^{1.86 \cdot (3 \cdot m - b_{min}^{(1)} - b_{min}^{(3)}) - n}.$$

Therefore, the overall complexity of the key recovery phase can be estimated as $N_d T_k$ as the time complexity of verifying candidate solutions is negligibly.

5 Conclusion

In this work, aiming to advance the state-of-the-art key recovery attacks on LowMC instances with full S-box layers using two plaintext/ciphertext pairs, we developed a new algebraic key-derived attack framework and proposed an improved differential enumeration attack framework. We identify first attacks on some LowMC instances that cannot be attacked using previous methods. Better attacks are further obtained on LowMC from 4 to 7 rounds for almost all three instances. All these results given in this paper, as well as our new attack frameworks, we believe, could provide more insights into the cryptanalysis of LowMC with low allowable data complexity, which has not yet come to an end.

Acknowledgements. We sincerely thank the anonymous reviewers for providing valuable comments to help us improve the overall quality of the paper. This work is supported by the National Key R&D Program of China (Grant No. 2024YFA1013000), the National Natural Science Foundation of China (Grant No. U2336207), and the National Cryptologic Science Fund of China (Grant No. 2025NCSF01013), and Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (JYB2025XDXM114), and Shandong Provincial Natural Science Foundation (ZR2026QC1032).

A The Proof of Lemma 1

In this section, we present a detailed proof of Lemma 1.

Proof When $R = 1$, considering the i-th output bit of $Enc_K^{(1)}(X)$, we have

$$Enc_K^{(1)}(X)[i] = A \oplus \sum_{j=0}^{n-1} B_j(x_j \oplus k_j) \oplus \sum_{j=0}^{n-1} \sum_{t=j+1}^{n-1} C_{j,t}(x_j \oplus k_j)(x_t \oplus k_t),$$

And

$$\Delta Enc_K^{(1)}[i] = \sum_{j=0}^{n-1} B_j \Delta x_j \oplus \sum_{j=0}^{n-1} \sum_{t=j+1}^{n-1} C_{j,t}(\Delta x_t(x_j \oplus k_j) \oplus \Delta x_j(x_t \oplus k_t) \oplus \Delta x_j \Delta x_t)$$

Obviously, $\Delta Enc_K^{(1)}[i]$ is a linear equation of K. So we have $\delta_K(\Delta Enc_K^{(1)}) \leq 2^1 - 1 = 1$.

We proceed by induction on R. Assuming that this lemma holds for $R = s$, we now prove that it also holds for $R = s + 1$. Similarly, considering the i-th output bit of $Enc_K^{(s+1)}(X)$ we can get

$$\begin{aligned} Enc_K^{(s+1)}(X)[i] = A &\oplus \sum_{j=0}^{n-1} B_j(Enc_K^{(s)}(X)[j] \oplus L_{s+1}(K)[j]) \\ &\oplus \sum_{j=0}^{n-1} \sum_{t=j+1}^{n-1} C_{j,t}(Enc_K^{(s)}(X)[j] \oplus L_{s+1}(K)[j])(Enc_K^{(s)}(X)[t] \oplus L_{s+1}(K)[t]), \end{aligned}$$

and

$$\begin{aligned} \Delta Enc_K^{(s+1)}[i] = &\sum_{j=0}^{n-1}\sum_{t=j+1}^{n-1} C_{j,t}(\Delta Enc_K^{(s)}[t](Enc_K^{(s)}(X)[j] \oplus L_{s+1}(K)[j]) \\ &\oplus \sum_{j=0}^{n-1} B_j(\Delta Enc_K^{(s)}[j]) \oplus \Delta Enc_K^{(s)}[j]\Delta Enc_K^{(s)}[t] \\ &\oplus \Delta Enc_K^{(s)}[j](Enc_K^{(s)}(X)[t] \oplus L_{s+1}(K)[t]). \end{aligned} \tag{9}$$

Since $\delta_K(\Delta Enc_K^{(s)}[i]) \leq 2^s - 1$ and $\delta_K(Enc_K^{(s)}(X)[i]) \leq 2^s$, we have $\delta_K(\Delta Enc_K^{(s+1)}[i]) \leq 2^{s+1} - 1$.

B Comparison Under Low Success Probability

In this section, we compare the time complexity of our work with [11] under a low success probability. The time complexity of our attack primarily depends on the parameters $(e, t, uu_1, b_{min}^{(1)}, b_{min}^{(3)}, b_{max})$. By properly choosing the parameters (as explained in Sect. 4), we can significantly reduce the time complexity while keeping the attack effective. A summary of our results is provided in Table 2.

Table 2. Summary of the attacks on LowMC under low success probability. T, M and Pro. represent the time, memory complexity (in $\log_2$) and success probability, respectively.

(n, k, m, r)	e	t	uu_1	$b_{min}^{(1)}$	$b_{min}^{(3)}$	b_{max}	T	M	Pro.	Ref.
(129, 129, 43, 4)	–	–	–	–	–	–	121.0	–	0.24	[11]
	37	11	9	5	5	11	120.9	32.3	0.40	Ours
(192, 192, 64, 4)	–	–	–	–	–	–	174.2	–	0.247	[11]
	54	13	11	8	8	16	170.9	38.6	0.307	Ours
(255, 255, 85, 4)	–	–	–	–	–	–	227.0	–	0.2465	[11]
	73	22	7	10	10	22	222.8	57.9	0.3967	Ours

C Comparison of the Advanced MITM Algebraic Attack Framework with Previous Works

In this section, we compare the Advanced MITM Algebraic Attack Framework with the previous works [4,8,15]. Different from the analysis in [4,8,15], our work uses the derivation of the two plaintext/ciphertext pairs to purely reduce the algebraic degree of the equation system. In [8], the author observes that multiplying the 3 output polynomial of degree $2^{\rho+1}$ for an S-box yields a degree of $4 \cdot 2^{\rho}$ instead of $3 \cdot 2^{\rho+1}$. Using this property, they successfully reduce the

algebraic degree of 3-multiplied polynomials from 24 to 16 for 5-round LowMC and from 48 to 32 for 7-round LowMC. In our strategy, when activating only one bit in the input difference, the algebraic degree of the 3-multiplied polynomial reaches 18 for 5-round LowMC and 42 for 7-round LowMC, which is larger than that of [8]. So, even though our attack improves upon the results of [4,15], it does not achieve better results compared with [8]. If we activate one bit in each active S-box, the algebraic degree of 3-multiplied polynomial is 9 for 5-round LowMC and 21 for 7-round LowMC, which is much smaller than that of [8]. However, our attack does not achieve better results for 5-round LowMC compared to [8] due to the high cost of linearization. As the number of rounds increases, the algebraic degree plays a more significant role in the overall complexity, which provides a better trade-off. For $R = 7$, our strategy achieves better time complexity.

To make a more direct comparison, we provide detailed parameters for different attack methods when $R = 2\rho + 1$ is odd in Table 3. e and u denote the number of guess bits (linearization) and the number of variables, respectively. deg denotes the algebraic degree of polynomial and $\deg'$ denotes the algebraic degree of the 3-multiplied polynomial as mentioned above.

Table 3. Choices of e, u, deg and $\deg'$ in different work for odd-round attack.

e	u	deg	$\deg'$	Ref.
2	$n-2$	$2^{\rho+1}-2$	$3\cdot(2^{\rho+1}-2)$	Sect. 3.2*
$2n/3$	$n/3$	$2^{\rho}-1$	$3\cdot(2^{\rho}-1)$	Sect. 3.2**
–	n	$2^{\rho+1}-1$	$3\cdot(2^{\rho+1}-1)$	Sect. 3.1
$n/3$	$2n/3$	2^{ρ}	$3\cdot 2^{\rho}$	[4]
$2n/3$	$n/3$	2^{ρ}	$3\cdot 2^{\rho}$	[15]
–	n	$2^{\rho+1}$	$4\cdot 2^{\rho}$	[8]

* Activate one bit in the input difference.
** Activate one bit in each active S-box.

D The Parameter Selection in Section 3

In this section, we present the parameter selection used in Sect. 3. Since $b_{max}^{(1)} = b_{max}^{(2)} = b_{max}^{(3)}$, for convenience, we represent $b_{max}^{(i)}$ with b_{max} and $1 \le i \le 3$. In Table 4, we select the best (e, t, uu_1) so that the time complexity is optimal. The values of $(b_{min}^{(1)}, b_{min}^{(3)})$ and b_{max} in different cases are also given.

Table 4. Choices of $(e, t, uu_1, b_{min}^{(1)}, b_{min}^{(3)}, b_{max})$ for different (n, k, m)

R	n	k	m	e	t	uu_1	$b_{min}^{(1)}$	$b_{min}^{(3)}$	b_{max}	Pro.
4	129	129	43	37	11	9	4	4	11	0.63
4	192	192	64	58	19	7	5	5	16	0.84
4	192	192	64	60	19	17	3	3	16	0.97
4	255	255	85	75	26	7	7	7	22	0.84
4	255	255	85	78	26	7	4	4	22	0.99

References

1. The Picnic Signature Algorithm Specification (2020). https://github.com/microsoft/Picnic/blob/master/spec/spec-v3.0.pdf
2. Albrecht, M.R., Rechberger, C., Schneider, T., Tiessen, T., Zohner, M.: Ciphers for MPC and FHE. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 430–454. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46800-5_17
3. Banik, S., Barooti, K., Durak, F.B., Vaudenay, S.: Cryptanalysis of LowMC instances using single plaintext/ciphertext pair. IACR Trans. Symmetric Cryptol. **2020**(ARTICLE), 130–146 (2020)
4. Banik, S., Barooti, K., Vaudenay, S., Yan, H.: New attacks on LowMC instances with a single plaintext/ciphertext pair. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021, Part I. LNCS, vol. 13090, pp. 303–331. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92062-3_11
5. Banik, S., Caforio, A., Vaudenay, S.: New attacks on LowMC using partial sets in the single-data setting. CIC **1**(1) (2024)
6. Bouillaguet, C., et al.: Fast exhaustive search for polynomial systems in $\mathbb{F}_2$. In: Mangard, S., Standaert, F.-X. (eds.) CHES 2010. LNCS, vol. 6225, pp. 203–218. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15031-9_14
7. Chase, M., et al.: Post-quantum zero-knowledge and signatures from symmetric-key primitives. In: Thuraisingham, B., Evans, D., Malkin, T., Xu, D. (eds.) Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, 30 October–03 November 2017, pp. 1825–1842. ACM (2017)
8. Dinur, I.: Cryptanalytic applications of the polynomial method for solving multivariate equation systems over GF(2). In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12696, pp. 374–403. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77870-5_14
9. Dinur, I., Liu, Y., Meier, W., Wang, Q.: Optimized interpolation attacks on LowMC. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9453, pp. 535–560. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48800-3_22
10. Dobraunig, C., Eichlseder, M., Mendel, F.: Higher-order cryptanalysis of LowMC. In: Kwon, S., Yun, A. (eds.) ICISC 2015. LNCS, vol. 9558, pp. 87–101. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-30840-1_6
11. Liu, F., Isobe, T., Meier, W.: Cryptanalysis of full LowMC and LowMC-M with algebraic techniques. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol.

12827, pp. 368–401. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84252-9_13
12. Liu, F., Meier, W., Sarkar, S., Isobe, T.: New low-memory algebraic attacks on LowMC in the picnic setting. IACR Trans. Symmetric Cryptol. 102–122 (2022)
13. Liu, F., Sarkar, S., Wang, G., Meier, W., Isobe, T.: Algebraic meet-in-the-middle attack on LowMC. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022, Part I. LNCS, vol. 13791, pp. 225–255. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22963-3_8
14. Rechberger, C., Soleimany, H., Tiessen, T.: Cryptanalysis of low-data instances of full LowMCv2. IACR Trans. Symmetric Cryptol. 163–181 (2018)
15. Sun, Y., Cui, J., Wang, M.: Improved attacks on LowMC with algebraic techniques. IACR Trans. Symmetric Cryptol. **2023**(4), 143–165 (2023). https://doi.org/10.46586/TOSC.V2023.I4.143-165