A Meet-in-the-middle Attack on Round-Reduced mCrypton

Yonglin Hao, Dongxia Bai

Department of Computer Science and Technology,
Tsinghua Universtiy, Beijing 100084, China
haoyl12@mails.tsinghua.edu.cn, baidx10@mails.tsinghua.edu.cn

Abstract. The meet-in-the-middle (MITM) attack on AES is a great success. In this paper, we apply the method to the lightweight SPN block cipher mCrypton.
We prove that the multiset technique used to analyze AES can not be applied directly to mCrypton due to the scarcity of information. As a solution, we replace the unordered multiset with the ordered sequence. We lower the memory requirement from 2^{100} to 2^{44} using the efficient differential enumeration technique.
Based on these modifications, we construct a MITM attack on 7-round mCrypton-64/96/128 with complexities of 2^{44} 64-bit blocks and 2^{57} encryptions.
We further extend the attack to 8 and 9 rounds for mCrypton-128 by adding some key-bridging techniques. The 8-round attack requires 2^{44} blocks and 2^{96} encryptions while the 9-round attack needs 2^{120} blocks and 2^{116} encryptions.

1 Introduction

The meet-in-the-middle (MITM) attack was first introduced by Diffie and Hellman in 1977[1]. Due to the domination of differential and linear attacks, it did not draw too much attention in early 1990s.

However, the Advanced Encryption Standard competition, which lasts from 1997 to 2000, greatly promoted the cryptanalysis of block ciphers. After the study of AES gets heated, the meet-in-the-middle strategy also shows a trend of revivification. In the past decade, the MITM scenario has become one of the most fruitful cryptanalysis method. It has been used to analyze block ciphers such as DES[12], KASUMI[15], IDEA[8],XTEA[14], KTANTAN[9] and Camellia[17][18]. It also shows good efficiency in the cryptanalysis of hash functions[10][11][13] and is adapted to attack against public key cryptosystem NTRU[16].

Among all the results of MITM attack, the most impressive ones come from the cryptanalysis on AES block cipher in single-key setting[2][3][4][5][6].

1.1 Previous MITM Attacks on AES

Demirci and Selçuk launched the first MITM attack on AES at FSE 2008 [2]. They consider a set of functions mapping one active byte to one byte after 4 rounds and prove that each byte of ciphertext can be determined by 25 byte parameters. Based on this observation, they construct a 5-round distinguisher on AES and develop a MITM attack on 7-round AES-192 and 8-round AES-256. This MITM attack has an obvious advantage. Since their distinguisher is effective on any message m, the data complexity of their attack is quite low (approximately 2^{32}). But its weakness is also undeniable: the high memory complexity. Actually, the basic attack is appropriate for AES-256. In order to make it work on the 192-bit version, Demirci et al. have to introduce some time/memory tradeoff, which makes their results on AES-192 quite marginal. Although modifications have been made in [3] and [4], the crisis of memory requirement remains severe.

The first breakthrough was made by Dunkelman, Keller and Shamir in [5] at ASIACRYPT 2010. They develop some new ideas to lower the memory complexity of the previous MITM attacks. Firstly, they store the unordered multiset of the function rather than the whole ordered sequence. They prove that after such an abbreviation, there is still enough information to make the attack work. Furthermore, the multiset technique also avoids a byte of subkey guess in the online phase. Secondly, they use a differential enumeration technique and reduce the number of parameters determining the set of functions from 24 to 16. More precisely, Dunkelman et al. only store the multisets determined by message pairs conforming a particular differential characteristic. Such a restriction on messages diminish the size of the pre-computed table by a large margin

and enable them to attack the 128-bit version of AES. Besides, [5] also reveals some weaknesses in key schedule and further reduce the time complexity of the attack on 8-round AES-192. Dunkelman et al.'s results greatly improve the efficiency of MITM attack on AES, but they are not the best results at that time[19].

Recently at EUROCRYPT 2013, Derbez, Fouque and Jean improved the results of [5]. Using their efficient differential enumeration technique, Derbez et al. further reduce the number of decisive byte parameters from 16 to 10. With this modification, they not only reduce the overall complexity of the existing MITM attacks on 7 and 8 round AES, but mount to 9 rounds for AES-256 as well[6]. Due to their contributions, the memory requirement is no longer the bottleneck of MITM attack on AES. It also makes the MITM attack surpass the impossible differential attack in overall efficiencies.

1.2 Applying to mCrypton

In standard single-key model, the MITM attack has proved to be a great success in analyzing AES block cipher. We try to apply the method to mCrypton.

mCrypton is a 64-bit lightweight block cipher introduced in 2006 by Lim and Korkishko[7], which is a reduced version of Crypton[20]. It is specifically designed for resource-constrained devices like RFID tags and sensors in wireless sensor networks. Like AES, mCrypton is also a SPN block cipher. According to key length, mCrypton has three versions namely mCrypton-64/96/128, which is in high accordance with AES-128/192/256.

In recent years, there are quite a few new results on mCrypton. Park launched a related-key rectangle attack on 8-round mCrypton-128 in 2009[21]. Then, in [22], Mala et al. gave a related-key impossible differential cryptanalysis on 9-round mCrypton-96/128. There are two biclique results on full-round mCrypton: [23] managed to attack mCrypton-96/128 and [24] further adapted the methods to all three versions. As to classical single-key model, the latest result, to the best of our knowledge, is the collision attack introduced in [25].

Table 1. Results of the Attacks on mCrypton.

Version	Rounds	Data	Time	Memory	Method	Reference
64	**7**	$\mathbf{2^{57}}$	$\mathbf{2^{57}}$	$\mathbf{2^{44}}$	**MITM**	**Section 4**
	12	2^{48}	$2^{63.38}$	–	Biclique	[24]
96	**7**	$\mathbf{2^{57}}$	$\mathbf{2^{57}}$	$\mathbf{2^{44}}$	**MITM**	**Section 4**
	8	2^{48}	2^{65}	$2^{81.6}$	CA	[25]
	9	$2^{59.9}$	$2^{74.9}$	$2^{63.9}$	RKIDC	[22]
	12	$2^{27.54}$	$2^{94.09}$	2^{20}	Biclique	[23]
	12	2^{48}	$2^{94.81}$	–	Biclique	[24]
128	**7**	$\mathbf{2^{57}}$	$\mathbf{2^{57}}$	$\mathbf{2^{44}}$	**MITM**	**Section 4**
	8	2^{46}	2^{46}	–	RKR	[21]
	8	2^{48}	2^{65}	$2^{81.6}$	CA	[25]
	8	$\mathbf{2^{57}}$	$\mathbf{2^{96}}$	$\mathbf{2^{44}}$	**MITM**	**Section 5.1**
	9	$2^{59.7}$	$2^{66.7}$	$2^{55.7}$	RKIDC	[22]
	9	$\mathbf{2^{53}}$	$\mathbf{2^{116}}$	$\mathbf{2^{120}}$	**MITM**	**Section 5.2**
	12	$2^{20.1}$	$2^{125.84}$	2^{20}	Biclique	[23]
	12	2^{48}	$2^{126.56}$	–	Biclique	[24]

CA: Collision Attack
RKIDC: Related-Key Impossible Differential Cryptanalysis.
MITM: Meet-in-the-Middle Attack.
RKR: Related-Key Rectangle Attack.

In this paper, we apply the methods of [6] to mCrypton and, with some modifications, we improve some of the existing single-key results:

Our results. Firstly, we find that the multiset technique can NOT be applied to mCrypton directly because of the information scarcity. As a solution, we replace the unordered multiset with ordered sequence and lower the memory requirement using the differential enumeration technique. After making these changes, we launch a MITM attack on 7 rounds for all three versions of mCrypton with a memory complexity of 2^{44} 64-bit blocks. This is the best single-key result for mCrypton-64. Then, we show some properties of key schedule and manage to attack 8 and 9 rounds for mCrypton-128. This is also the first single-key attack on 9-round mCrypton-128. As can be seen from Table 1, compared with the collision attack, our MITM method has an obvious advantage in memory requirement, which allows us to attack the short-key version and extend the attack on mCrypton-128 to 9 rounds. But on the other hand, the high time complexity becomes a bottleneck and impedes us to attack 8 or more rounds for mCrypton-96.

Organization of the Paper. In section 2, we describe the mCrypton block cipher and introduce the previous attack procedure on AES. Then, in Section 3, we give the definitions of σ-set and multiset and explain why the multiset technique does not work on the cryptanalysis of mCrypton. Also in this section, we further give our solution and prove its effectiveness. In Section 4, we describe our basic MITM attack on 7-round mCrypton-64/96/128 in detail and analyze its complexity. In Section 5, we extend the basic attack to 8-round and 9-round mCrypton-128 using some key bridging techniques. Finally, we summarize our paper in Section 6.

2 Preliminary

This part contains some background information of our attack. It also gives the notations and units used in this article. As is commonly accepted, the plaintexts are denoted by p and ciphertexts by c.

2.1 Description of mCrypton

mCrypton is a 64-bit lightweight block cipher based on SPN design. It consists of 16 4-bit nibbles which are represented by a 4×4 matrix as follows:

$$A = \begin{pmatrix} a_0 & a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 & a_7 \\ a_8 & a_9 & a_{10} & a_{11} \\ a_{12} & a_{13} & a_{14} & a_{15} \end{pmatrix} \tag{1}$$

It has three versions, categorized by key length, namely mCrypton-64/96/128. All the three versions have 12 rounds and each round consists of 4 transformations as follows.

Nonlinear Substitution γ. This transformation consists of nibble-wise substitutions using four 4-bit S-boxes $S_i (0 \leq i \leq 3)$. The four S-boxes has relationship:

$$S_0 = S_2^{-1}, \quad S_1 = S_3^{-1}.$$

According to our experiments, the S-boxes of mCrypton have the same property with the S-box of AES (Property 1).

Property 1. Given Δ_i and Δ_o two non-zero differences in $\mathbb{F}_{16}$, the equation

$$S_t(x) \oplus S_t(x \oplus \Delta_i) = \Delta_o, \quad \forall t \in [0, 3]$$

has one solution on average.

Bit Permutation π. The bit permutation transformation π has the same function with the MixColumns transformation of AES. It mixes each column of the 4×4 matrix A. For column $i (0 \leq i \leq 3)$, it uses the corresponding column permutations π_i. Suppose

$$A = (A_0, A_1, A_2, A_3)$$

where A is the 4×4 matrix and A_i is its i-th column. Then, we have

$$\pi(A) = (\pi_0(A_0), \pi_1(A_1), \pi_2(A_2), \pi_3(A_3)).$$

According to [7], each π_i is defined for nibble columns $a = (a_0, a_1, a_2, a_3)^t$ and $b = (b_0, b_1, b_2, b_3)^t$ by

$$b = \pi_i(a) \Leftrightarrow b_j = \bigoplus_{k=0}^{3} (m_{(i+j+k) mod 4} \bullet a_k).$$

The symbol $\bullet$ means bit-wise AND and the masking nibbles m_i are given by

$$m_0 = 1110_2, m_1 = 1101_2, m_2 = 1011_2, m_2 = 0111_2.$$

π transformation is an involution, which means $\pi = \pi^{-1}$. It has a differential brunch number of 4.

Column-To-Row Transposition τ. This is simply the ordinary matrix transposition. It moves the nibble from the position (i, j) to position (j, i).

Key Addition σ. It is a simple bit-wise XOR operation and resembles the AddRoundKey operation of AES. The r-th round ($1 \leq r \leq 12$) of mCrypton applied to a 64-bit state x can be denoted by

$$\rho_{k_r}(x) = \sigma_{k_r} \circ \tau \circ \pi \circ \gamma(x).$$

Like AES, mCrypton also performs an initial key addition transformation (σ_{k_0}) before round 1. In addition, mCrypton adds a linear operation $\phi = \tau \circ \pi \circ \tau$ after round 12. So, the whole process of mCrypton encryption is

$$c = \phi \circ \rho_{k_{12}} \circ ... \circ \rho_{k_1} \circ \sigma_{k_0}(p)$$

Since we use some key bridging skills to analyze mCrypton-128, we briefly introduce the key schedule of mCrypton-128:

Key Schedule of mCrypton-128. The 128-bit internal register

$$U = (U_0, U_1, U_2, U_3, U_4, U_5, U_6, U_7)$$

is first initialized with the 128-bit user key. Each $U_i (0 \leq i \leq 7)$ is a 16-bit (4-nibble) word, occupying a row of the 4×4 matrix. Round keys $k_r (0 \leq r \leq 12)$ are computed consecutively as follows:

$$T \leftarrow S(U_0) \oplus C_r, T_i \leftarrow T \bullet M_i$$

$$k_r = (U_1 \oplus T_0, U_2 \oplus T_1, U_3 \oplus T_2, U_4 \oplus T_3)$$

$$U \leftarrow (U_5, U_6, U_7, U_0^{<<3}, U_1, U_2, U_3, U_4^{<<8}).$$

S is the nibble-wise S-box operation using S-box S_0. C_r is the round constant word for round r. Masking words M_i is to take the i-th nibble of a word:

$$M_0 = 0xf000, M_1 = 0x0f00, M_2 = 0x00f0, M_3 = 0x000f.$$

The symbol $X^{<<n}$ means left rotation of a 16-bit word X by n bits.

2.2 Notations and Units

Here, we summarize the notations that we use through this paper.

State $\mathbf{x_r^i}$: The 64-bit mCrypton state is represented by different small letters. Plaintexts and ciphertexts are represented by p and c. In the r-th round, we denote the internal state after σ_{k_r} transformation by x_r, after γ by y_r, after π by z_r and after τ by w_r. k_r represents the round key while u_r is calculated linearly from k_r with $u_r = \pi \circ \tau(k_r)$. The difference of state x is denoted by Δx. Besides, the superscript represents the position that the state lies in a sequence (or set).

Nibble $\mathbf{x[i]}$: We refer to the i-th nibble of a state x by $x[i]$, and use $x[i, ..., j]$ for nibbles at positions from i to j. The nibbles of the state is numbered as the matrix in equation (1).

Bit $\mathbf{x[i]|_k}$: Each nibble has 4 bits numbered 4,3,2,1 from left to right. If we refer to bit k of nibble $x[i]$, we denote it by $x[i]|_k$.

Tuple $(\mathbf{p}, \mathbf{p'}, \mathbf{k})$: (p, p') represents the plaintext pair and k is the subkey guesses that make (p, p') conform a particular differential characteristic. So within each tuple (p, p', k), (p, p') is the candidate of right pair and k is the candidate of correct key guesses.

Bit-wise operators:

$\|$ concatenate two strings of bits.

$\oplus$ bit-wise XOR.

$\bullet$ bit-wise AND.

In this paper, memory complexities of our attacks are measured by the number of 64-bit mCrypton blocks and time complexities by mCrypton encryptions (decryptions).

2.3 The MITM Attack on AES

The method we use in this paper mainly depends on the MITM attack given by Derbez, Fouque and Jean in [6], so we make a brief introduction of their work first.

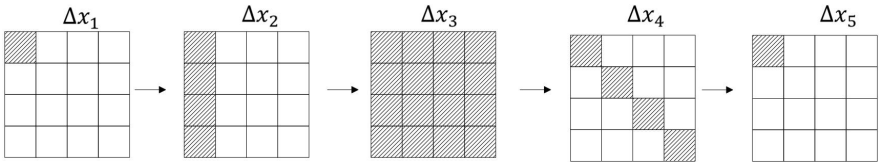

Fig. 1. The 4-round truncated differential characteristic for AES. Dashed nibbles are active.

In [6], they denote the σ-set by $\{x_1^0, ..., x_1^{255}\}$. Their attack can be summarized in 3 steps:

Precomputation

a. In this step, they build a lookup table T containing 2^{80} multisets. Each of the multisets is constructed from a particular σ-set whose x_1^0 is a member of the pair conforming the truncated differential characteristic in Figure 1.

Online

b. In this step, they guess several key bytes in the first and last few AES rounds and find the right message pairs conforming a particular differential characteristic through partial encryptions (decryptions). Having acquired the members of the right pairs and utilizing the key guesses, they deduce the σ-sets they want.

c. Using the key guesses, they deduce the multisets from the σ-sets and check whether the multisets exist in the pre-computed lookup table T. If so, the key guesses are correct with high probability.

In this MITM attack, T contains $|T| = 2^{80}$ values and the total number of multisets is $N_m = 2^{506.17}$, so the error rate of this attack, denoted by Pe, can be computed with

$$Pe = \frac{|T|}{N_m} \tag{2}$$

which is approximately $2^{-426.17}$.

According to (2), Pe depends on the size of pre-computed table $|T|$ and the total number of multisets N_m. Pe can be low only when $|T|$ is much smaller than N_m. For AES, we have $|T| << N_m$. But as described in the following section, this is not true for mCrypton.

3 Problems and Solutions

In this section, we first define the multiset of mCrypton and explain why it can not be used in our MITM attack. As a solution, we replace the unordered multisets with ordered sequences and lower the memory requirement by a large margin using the differential enumeration technique introduced in [5] and [6].

3.1 The Helplessness of the Multiset Technique

Similar to the definitions of AES multiset in [5] and [6], we can define the σ-set and multiset of mCrypton.

Definition 1. ***(σ-set of mCrypton).*** *A σ-set is a set of 16 64-bit mCrypton-states that are all different in one nibble (the active nibble) and all equal in the other state nibbles (the inactive nibbles).*

Definition 2. ***(Multisets of nibbles).*** *A multiset is a generalized concept of set. It is a set that allows elements to appear more than once. For mCrypton, a multiset consists of 15 nibbles and can take as many as $\binom{2^4+15-1}{15} \approx 2^{28.6}$ different values.*

In [5], the authors constructed a truncated differential characteristic. In [6], it is proved that with the help of the truncated characteristic, the needed multisets can be determined by 10 byte parameters and take only 2^{80} values. Based on the similarities between mCrypton and AES, we can also construct a differential characteristic and lower the multiset number to 2^{40}. For example, suppose the mCrypton σ-set

$$S_\sigma = \{x_1^0, x_1^1, ..., x_1^{15}\}$$

has an active nibble at position 0 and x_1^0 belongs to a pair conforming to the differential characteristic in Figure 2.

Then, the multiset after 4 rounds of mCrypton encryption

$$S_M = \{x_5^1 \oplus x_5^0[0], ..., x_5^{15} \oplus x_5^0[0]\}$$

can be determined by 10 nibble parameters namely:

$$\Delta x_1[0], x_2^0[0,1,2,3], x_4^0[0,4,8,12], \Delta z_4[0].$$

So we only need to store 2^{40} multisets in the pre-computed lookup table.

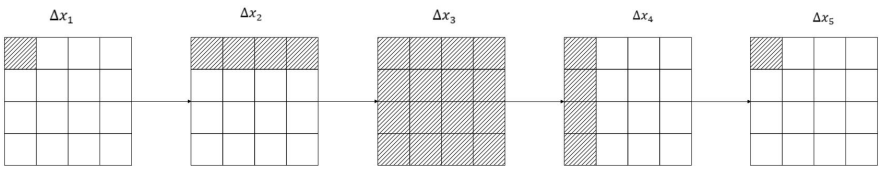

Fig. 2. The 4-round truncated differential characteristic. Dashed nibbles are active.

However, according to Definition 2, there are totally about $2^{28.6}$ multisets in theory, which means the pre-computed lookup table is quite likely to cover all the multisets. In this case, the pre-computed table will become useless in filtering and retrieving subkeys.

So, the multiset technique can not be applied directly to the MITM attacks on mCrypton.

3.2 The Ordered Sequence and the Efficient Differential Enumeration Technique

The failure of multiset technique in attacking mCrypton actually results from the lack of information. As a straightforward solution, we replace the multiset with ordered sequence. The sequence is the same with that of [25] but we use the efficient differential enumeration method of [6] to diminish the memory requirement.

In the following part of this paper, the σ-set with an active nibble at position $j (0 \leq j \leq 15)$ is denoted by

$$A_j = \{x_1^0, ..., x_1^{15}\} \tag{3}$$

where x_1^0 is decided by a differential characteristic and $x_1^i (1 \leq i \leq 15)$ satisfies

$$x_1^i \oplus x_1^0[j] = i. \tag{4}$$

So the corresponding ordered sequence constituted by the l-th nibble ($l \in [0, 15]$) of x_5 is denoted by

$$B_j^l = (x_5^1 \oplus x_5^0[l], ..., x_5^{15} \oplus x_5^0[l]). \tag{5}$$

Though the difference seems quite insignificant, the total number of ordered sequences soars dramatically to 2^{60} and allows us to launch our attack. Furthermore, in the following analysis, we can prove that such a great increment of information only enhance the complexity of pre-computation by a small factor and it does not affect the efficiency of online phase.

Proposition 1. *$\forall j \in [0, 15]$ and $\forall l \in [0, 15]$. Let the σ-set be*

$$A_j = \{x_1^0, ..., x_1^{15}\}$$

Then, the corresponding sequence

$$B_j^l = (x_5^1 \oplus x_5^0[l], x_5^2 \oplus x_5^0[l], \cdots, x_5^{15} \oplus x_5^0[l])$$

can be fully determined by 25 nibble parameters:

- *1 nibble of x_1^0.*
- *The full 16-nibble state x_3^0;*
- *4 nibbles of x_2^0.*
- *4 nibbles of x_4^0.*

Proof. We just let $j = 0$ and $l = 0$. Then, the 25 nibbles required are:

$$x_1^0[0], x_2^0[0, 1, 2, 3], x_3^0[0, \cdots, 15], x_4^0[0, 4, 8, 12].$$

For the t-th element of B_0^0 ($t \in [1, 15]$), the difference $x_5^t \oplus x_5^0[0]$ can be deduced from $x_4^0[0, 4, 8, 12]$ and $x_4^t \oplus x_4^0[0, 4, 8, 12]$.

$x_4^t \oplus x_4^0[0, 4, 8, 12]$ requires the knowledge of $x_3^0[0, ..., 15]$ and $x_3^t \oplus x_3^0[0, \cdots, 15]$.

$x_3^t \oplus x_3^0[0, ..., 15]$ is generated linearly from $y_2^t \oplus y_2^0[0, 1, 2, 3]$, which can be deduced from $x_2^0[0, 1, 2, 3]$ and $x_2^t \oplus x_2^0[0, 1, 2, 3]$.

$x_2^t \oplus x_2^0[0, 1, 2, 3]$ is generated linearly from $y_1^t \oplus y_1^0[0]$, which requires the knowledge of $x_1^0[0]$ and $x_1^t[0] \oplus x_1^0[0]$. According to equation (4), we have $x_1^t[0] \oplus x_1^0[0] = t$. Hence, all the nibble parameters required are: $x_1^0[0]$, $x_2^0[0, 1, 2, 3]$, x_3^0, $x_4^0[0, 4, 8, 12]$. □

We deliberately restrict that the x_1^0 of σ-set should belong to one of the right pairs satisfying the differential characteristic in Figure 2. This method is called the differential enumeration technique in [6] and it can diminish the size of lookup table from 2^{100} to 2^{44} 64-bit blocks (Proposition 2).

Proposition 2. *If the x_1^0 of a σ-set A_0 belongs to a pair satisfying the differential characteristic in Figure 2, the corresponding sequence B_0^0 can only take 2^{44} values.*

Proof. According to Proposition 1, B_0^0 is determined by 25 nibbles namely:

$$x_1^0[0], x_2^0[0, 1, 2, 3], x_3^0[0, \cdots, 15], x_4^0[0, 4, 8, 12].$$

The 11 nibbles

$$x_1^0[0], \Delta x_1[0], x_2^0[0, 1, 2, 3], x_4^0[0, 4, 8, 12], \Delta z_4[0]$$

can only take as many as 2^{44} values and deduce the values of the 25 nibbles determining B_0^0.

The knowledge of $x_1^0[0]$ and $\Delta x_1[0]$ is sufficient to deduce $\Delta x_2[0, 1, 2, 3]$. Combining $x_2^0[0, 1, 2, 3]$ and $\Delta x_2[0, 1, 2, 3]$, we get the 16-nibble difference Δx_3.

Similarly, we can deduce $\Delta y_4[0, 4, 8, 12]$ from $\Delta z_4[0]$. Adding the knowledge of $x_4^0[0, 4, 8, 12]$, the 16-nibble differential Δy_3 is determined.

Since $y_3 = \gamma(x_3)$, according to the property of mCrypton S-boxes (Property 1), we can only get one value on average for each of the 16-nibble state x_3.

This is the way we deduce the sequence B_0^0 from the σ-set A_0. □

4 Attacks on 7-Round mCrypton-64/96/128

We use the differential characteristic in Figure 2 and launch our basic attack on 7-round mCrypton-64/96/128. The complete differential path can be seen in Figure 5 in Appendix A.

The algorithm is as follows:

1. **Pre-computation:** According to Proposition 2, we consider all the possible values of the 11 nibble parameters namely
$$x_1^0[0], \Delta x_1[0], x_2^0[0,1,2,3], x_4^0[0,4,8,12], \Delta z_4[0]$$
and deduce the 2^{44} corresponding sequences B_0^0. We store the B_0^0s in a hash table T_s.
2. **Online:**
Stage 1: Finding the right pair:
(i) Encrypt 2^{41} structures of 2^{16} plaintexts with active nibbles at positions 0,4,8,12. So there are about 2^{31} pairs, (p, p'), in each structure.
(ii) For each structure, find the pairs whose ciphertexts have no difference in all nibbles except for positions $0, 4, 8, 12$. Store the remaining pairs in a hash table T_c. This is a 48-bit filter, so approximately 2^{24} (p, p') remain.
(iii) We assume that the (x_1, x_1') only have non-zero difference at position 0. So, for each remaining (p, p'), we guess $\Delta x_1[0]$ and compute $\Delta y_0[0,4,8,12]$ linearly. Since $\Delta p = \Delta x_0$ and $y_0 = \gamma(x_0)$, we can deduce the possible values of $k_0[0,4,8,12]$. According to Property 1, about 2^{28} tuples $(p, p', k_0[0,4,8,12])$ are acquired.
(iv) For the tuples $(p, p', k_0[0,4,8,12])$, we assume the Δy_5 of the message pair only have non-zero difference at position 0. Then we guess $\Delta y_5[0]$ and deduce the $\Delta x_6[0...3]$ linearly. Since $\Delta y_6 = \tau(\Delta c)$, we can deduce the possible values of $u_7[0...3]$. According to Property 1, 2^{32} tuples of $(p, p', k_0[0,4,8,12], u_7[0...3])$ are acquired.
Stage 2: Looking up the ordered sequence:
(v) For each of the 2^{32} tuples $(p, p', k_0[0,4,8,12], u_7[0...3])$, we deduce 2^4 sequences B_0^0:
(a) Either choose p or p' and denote it by p^0. With the knowledge of $k_0[0,4,8,12]$, we deduce $z_0^0[0,4,8,12]$ through partial encryptions.
(b) For any $i \in [1,15]$, $z_0^i[0,4,8,12]$ is determined by
$$z_0^i[0] = z_0^0[0] \oplus i$$
$$z_0^i[4,8,12] = z_0^0[4,8,12].$$
Then $p^i[0,4,8,12]$ is deduced through partial decryption while other nibbles of p^i are identical to those of p^0. So we have a sequence of plaintexts:
$$(p^0, p^1, ..., p^{15}).$$
(c) After encrypting the sequence of plaintexts and partially decrypting the ciphertexts with the knowledge of $u_7[0...3]$. we get a sequence:
$$(x_6^0[0...3], ..., x_6^{15}[0...3]).$$
(d) Guess $u_6[0]$ and deduce the 15-nibble sequence B_0^0 through partial decryptions. We have
$$B_0^0 = (x_5^1 \oplus x_5^0[0], ..., x_5^{15} \oplus x_5^0[0]).$$
(vi) Check whether B_0^0 exists in the pre-computed lookup table T_s. If so, the (p, p') is the right pair and the key guesses $k_0[0,4,8,12]$, $u_7[0...3]$, $u_6[0]$ are correct with high probability. The error rate is $2^{44-60} = 2^{-16}$ to be precise.

Memory Complexity:
In the pre-computation phase, T_s contains 2^{44} sequences and each sequence occupies 60 bits of space, which is much larger than T_c's 2^{24} message pairs. So the memory complexity of this attack is dominated by T_s' 2^{44} 64-bit blocks. Since each sequence has 15 nibbles, it requires $2^{44} \times 15 \approx 2^{48}$ encryptions to construct the lookup table.

Time & Data Complexity:
As to online phase, Stage 1 encrypts $2^{41} \times 2^{16} = 2^{57}$ plaintexts while Stage 2 is dominated by step (d) which requires $2^{32} \times 2^4 \times 16 = 2^{40}$ encryptions. So the time and data complexity of our attack are both 2^{57}.

5 Extending the Basic Attack to 8 and 9 Rounds for mCrypton-128

Due to the complicated key schedule, we fail in extending our basic attack to more rounds for mCrypton-96. But using the key bridging technique, we can still attack 8-round and 9-round mCrypton-128.

5.1 8-Round Attack on mCrypton-128

After studying the key schedule of mCrypton-128, we find that the knowledge of k_8 can deduce some bits in k_0 and reduce the time complexity of the online phase:

Proposition 3. *By the key schedule of mCrypton-128, knowledge of the entire 16-nibble k_8 allows deduce $k_0[6]$, 3 bits of $k_0[2]$ and 1 bit of $k_0[14]$.*

Proof. According to the key schedule of mCrypton-128, the relationship of the 8 bits are:

$$k_0[2]|_{4,3,2} = k_8[3]|_{3,2,1} \tag{6}$$
$$k_0[6]|_{4,3,2} = k_8[7]|_{3,2,1} \tag{7}$$
$$k_0[6]|_1 = k_8[4]|_4 \tag{8}$$
$$k_0[14]|_1 = k_8[12]|_4 \tag{9}$$

The readers may refer to [7] for the detailed key schedule of mCrypton-128. □

In order to make full use of Proposition 3, we deliberately change the positions of active nibbles (Figure 3). Within each structure of plaintexts, the active nibbles are located at position 2,6,10 and 14. The σ-set is

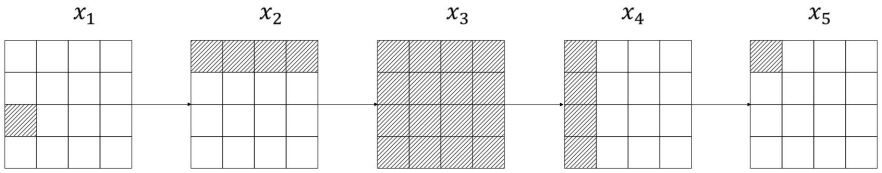

Fig. 3. The differential characteristic for 8-round mCrypton-128.

A_8 and the corresponding ordered sequence becomes B_8^0.

The size of pre-computed lookup table is still 2^{44} 64-bit blocks and the 11 decisive nibbles become

$$x_1^0[8], \Delta x_1[12], x_2^0[0,1,2,3], x_4^0[0,4,8,12], \Delta z_4[0].$$

Refer to Figure 6 in Appendix A for the complete differential characteristic.

For the online stage, the first step remains the same with the basic attack in Section 4. We also need to consider 2^{41} structures of plaintexts so the data complexity is still 2^{57}.

We hypothesize that each of the $2^{41} \times \binom{2^{16}}{2} \approx 2^{72}$ plaintext pairs (p, p') conforms the differential characteristic, so the possible values of k_8 and $k_0[2,6,10,14]$ can be determined with several simple steps of guessing possible subkey nibbles and filtering the unqualified tuples. We call the following steps the **Guessing and Filtering** process. Note that such a process is operated structure by structure so we do not have to store all the ciphers in a table:

Step 1 For each of the 2^{72} plaintext pairs (p, p'), we guess 2^{16} possible $\Delta y_6[0,1,2,3]$ and deduce the corresponding Δx_7. With the knowledge of Δx_7 and $\Delta y_7 = \tau(\Delta c)$, we further acquire the possible values of k_8. We acquire 2^{88} tuples of (p, p', k_8).

Step 2 According to Proposition 3, we deduce $k_0[6]$ from k_8 and acquire $(y_0[6], y_0'[6])$. Discard the tuple (p, p', k_8) if $\Delta y_0[6]|_4 \neq 0$ since

$$\Delta y_0[6] = m_3 \bullet \Delta z_0[2] \tag{10}$$

where $m_3 = 0111_2$. It's a one-bit filter so there are 2^{87} tuples remaining.

Step 3 For each of the remaining tuples, we traverse the 2 possible values of $\Delta z_0[2]$[1], linearly compute the corresponding $\Delta y_0[2, 6, 10, 14]$, and deduce the possible $k_0[2, 6, 10, 14]$. At this point, we have $2^{87} \times 2 = 2^{88}$ tuples of $(p, p', k_0[2, 6, 10, 14], k_8)$.

Step 4 For each tuple, we check whether k_8 and $k_0[2, 14]$ conform equations (7) and (9). Since this is a 4-bit filter, about 2^{84} tuples are expected to remain.

After the Guessing and Filtering process, the following two steps are natural and in high accordance with those in Section 4:

1. For each of the 2^{84} tuples, we assume Δy_5 only has non-zero difference at position 0, guess $\Delta y_5[0]$, and deduce $u_7[0, ..., 3]$. According to Property 1, we can attain about 2^{88} tuples of
$$(p, p', k_0[2, 6, 10, 14], u_7[0, 1, 2, 3], k_8).$$
2. Then we guess $u_6[0]$, construct the corresponding ordered sequence, and check whether the sequence exist in the pre-computed lookup table. This step dominates the time complexity of this attack with $2^{88} \times 2^4 \times 16 = 2^{96}$ encryptions (decryptions).

To sum up, the 8-round attack on mCrypton-128 has a time complexity of 2^{96}. The data complexity is still 2^{57}. The memory complexity is still dominated by the pre-computed lookup table, which is 2^{44} 64-bit blocks to be precise.

5.2 9-Round Attack on mCrypton-128

The key bridging technique we use in 9-round attack is as follows:

Proposition 4. *By the key schedule of mCrypton-128, the knowledge of the entire 16-nibble k_9 allows to deduce $k_0[0, 3]$*

Proof. According to the key schedule, we have

$$k_0[3]|_{4,3,2} = k_9[12]|_{3,2,1}$$
$$k_0[3]|_1 = k_9[13]|_4$$
$$k_0[0] = S_0((k_9[8]|_{2,1} \| k_9[9]|_{4,3})) \oplus (k_9[13]|_{3,2,1} \| k_9[14]|_4) \oplus 1$$

These relationships can be deduced easily from the key schedule. □

For mCrypton-128, if we add one round, each ordered sequence need $25 + 16 = 41$ decisive nibbles and accordingly, the pre-computed table has to store $2^{4\times(11+16)} = 2^{108}$ values, which exceeds the total number of ordered sequences (2^{60}). Again, the scarcity of information becomes the barrier in our extension.

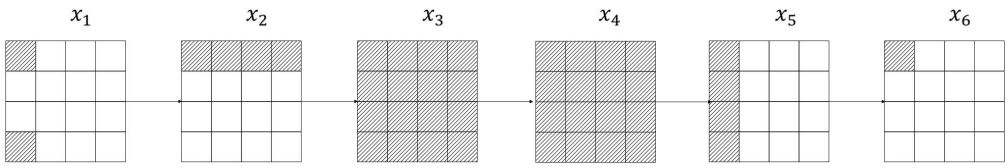

Fig. 4. The 5-round differential characteristic for attacking 9-round mCryton-128.

As a solution, we consider the σ-set with 2 active nibbles and contains 256 64-bit states. In order to make full use of Proposition 4, we construct the 5-round differential characteristic in Figure 4 and deliberately choose positions 0 and 12 as active nibbles. The σ-set should be denoted by $A_{0,12}$:

$$A_{0,12} = \{x_1^0, x_1^1, ..., x_1^{255}\}$$

[1] Three bits of $\Delta z_0[2]$ are known according to (10).

For the elements of $A_{0,12}$, their superscripts are defined $i(0 \leq i \leq 255)$ as follows:

$$i = (x_1^i \oplus x_1^0[0]|_{4,3,2,1)} \| (x_1^i \oplus x_1^0[12]|_{4,3,2,1)}.$$

The corresponding sequence $B_{0,12}^0$ consists of 255 nibbles:

$$B_{0,12}^0 = (x_6^1 \oplus x_6^0[0], x_6^2 \oplus x_6^0[0], ..., x_6^{255} \oplus x_6^0[0]).$$

In this way, the total number of ordered sequence soars to $2^{4\times 255} = 2^{1020}$, which is sufficient for us to launch an attack. Each $B_{0,12}^0$ is decided by 43 nibbles and the pre-computed table stores 2^{116} values determined by 29 nibble parameters namely:

$$x_1^0[0,12], \Delta x_1[0,12], x_2^0[0,1,2,3], x_3^0, x_5^0[0,4,8,12], \Delta z_5[0].$$

Since each $B_{0,12}^0$ requires $4 \times 255 = 1020$ bits for storage, the memory complexity of pre-computation is $2^{116} \times 1020/64 \approx 2^{120}$ 64-bit blocks. To construct the lookup table, we also need $2^{116} \times 256 \times 2^{-4} = 2^{120}$ encryptions. The complete differential characteristic can be seen in Figure 7 in Appendix A.

As to the online part of the attack, we need 2^{21} structures of plaintexts with active nibbles at positions 0,3,4,7,8,11,12,15. So the data complexity is $2^{21} \times 2^{4\times 8} = 2^{53}$. For the convenience of interpretation, we define the set λ as follows:

$$\lambda = \{0, 3, 4, 7, 8, 11, 12, 15\}.$$

The online phase of the 9-round attack on mCrypton-128 can be summarized in the following 5 steps:

1. We firstly encrypt the 2^{53} plaintexts. Then, for each of the $2^{32+31} \times 2^{21} = 2^{84}$ message pairs, we determine 2^{16} possible k_9 by considering the possible values of $\Delta y_6[0,1,2,3]$ and acquire $2^{16+84} = 2^{100}$ tuples (p, p', k_9).
2. For each tuple, we can deduce $k_0[0,3]$ from k_9 according to Proposition 4. After the Guessing and Filtering process, we have approximately 2^{100} tuples of $(p, p', k_0[\lambda], k_9)$. The process requires 2^{100} encryptions.
3. For each $(p, p', k_0[\lambda], k_9)$, we guess $\Delta y_6[0]$ and deduce 2^4 possible values of $u_8[0...3]$ under the hypothesis that y_6 only has non-zero difference at position 0. This step takes 2^{104} partial decryptions and attains 2^{104} tuples of $(p, p', k_0[\lambda], u_8[0, ..., 3], k_9,)$
4. For each of the 2^{104} tuples, we guess $u_7[0]$ and deduce the ordered sequence $B_{0,12}^0$ through partial decryptions. This step constructs 2^{108} sequences and dominates the time complexity of the online phase with $2^{104} \times 2^4 \times 256 = 2^{116}$ encryptions (decryptions).
5. Finally, we check whether any of the 2^{108} sequences exist in the pre-computed lookup table. If this is the case, the corresponding key guesses $k_0[\lambda]$, $u_7[0]$, $u_8[0...3]$ and k_9 should be correct with high probability. The error rate is $2^{120}/2^{1020} = 2^{-900}$ to be precise.

To sum up, the 9-round attack on mCrypton-128 has a memory complexity of 2^{120} 64-bit blocks and a data complexity of 2^{53} chosen plaintexts. The time complexity of online phase is 2^{116} and additional 2^{120} encryptions are required in the preprocessing phase to construct the lookup table.

6 Conclusion

In this article, we consider the standard single-key attack model and prove that the multiset technique can NOT be used directly on lightweight SPN block cipher mCrypton. We solve this problem by turning unordered multisets to ordered sequences and diminish the size of lookup table with the efficient differential enumeration technique.

With these methods, we manage to launch a MITM attack on 7-round mCrypton of all versions. As far as we know, this is the best result on mCrypton-64 in the single-key model.

Based on some key bridging techniques, we extend the basic attack to 8 rounds for mCrypton-128. Our attack has significant advantages over the previous collision attack in memory complexity requiring only 2^{44} 64-bit mCrypton blocks.

Adding some other modifications, we further mount to 9 rounds for mCrypton-128 with memory complexity 2^{120} and time complexity 2^{116}. This is also the first 9-round single-key attack on mCrypton-128.

To be honest, the MITM attack on mCrypton does not seem as efficient as its application to AES. But by using the differential enumeration technique, we lower the memory complexity by a large margin, which enable us to attack short-key version and extend the basic attack to more rounds.

References

1. Diffie, W., Hellman, M. E. (1977). Special feature exhaustive cryptanalysis of the NBS Data Encryption Standard. Computer, 10(6), 74-84.
2. Demirci, H., Selçuk, A. A. (2008, January). A meet-in-the-middle attack on 8-round AES. In Fast Software Encryption (pp. 116-126). Springer Berlin Heidelberg.
3. Demirci, H., Taşkın, İ., Çoban, M., Baysal, A. (2009). Improved meet-in-the-middle attacks on AES. In Progress in Cryptology-INDOCRYPT 2009 (pp. 144-156). Springer Berlin Heidelberg.
4. Wei, Y., Lu, J., Hu, Y. (2011). Meet-in-the-middle attack on 8 rounds of the AES block cipher under 192 key bits. In Information Security Practice and Experience (pp. 222-232). Springer Berlin Heidelberg.
5. Dunkelman, O., Keller, N., Shamir, A. (2010). Improved single-key attacks on 8-round AES-192 and AES-256. In Advances in Cryptology-ASIACRYPT 2010 (pp. 158-176). Springer Berlin Heidelberg.
6. Derbez, P., Fouque, P. A., Jean, J. (2013). Improved Key Recovery Attacks on Reduced-Round AES in the Single-Key Setting. In Advances in Cryptology - EUROCRYPT 2013 (pp. 371-387). Springer Berlin Heidelberg.
7. Lim, C. H., Korkishko, T. (2006). mCrypton - A lightweight block cipher for security of low-cost RFID tags and Sensors. In Information Security Applications (pp. 243-258). Springer Berlin Heidelberg.
8. Demirci, H., Selçuk, A. A., Türe, E. (2004, January). A new meet-in-the-middle attack on the IDEA block cipher. In Selected Areas in Cryptography (pp. 117-129). Springer Berlin Heidelberg.
9. Bogdanov, A., Rechberger, C. (2011, January). A 3-subset meet-in-the-middle attack: cryptanalysis of the lightweight block cipher KTANTAN. In Selected Areas in Cryptography (pp. 229-240). Springer Berlin Heidelberg.
10. Aoki, K., Sasaki, Y. (2009). Meet-in-the-middle preimage attacks against reduced SHA-0 and SHA-1. In Advances in Cryptology-CRYPTO 2009 (pp. 70-89). Springer Berlin Heidelberg.
11. Sasaki, Y. (2011, January). Meet-in-the-middle preimage attacks on AES hashing modes and an application to whirlpool. In Fast Software Encryption (pp. 378-396). Springer Berlin Heidelberg.
12. Dunkelman, O., Sekar, G., Preneel, B. (2007). Improved meet-in-the-middle attacks on reduced-round DES. In Progress in Cryptology C INDOCRYPT 2007 (pp. 86-100). Springer Berlin Heidelberg.
13. Sasaki, Y., Aoki, K. (2009, January). Meet-in-the-middle preimage attacks on double-branch hash functions: Application to RIPEMD and others. In Information Security and Privacy (pp. 214-231). Springer Berlin Heidelberg.
14. Sekar, G., Mouha, N., Velichkov, V., Preneel, B. (2011). Meet-in-the-middle attacks on reduced-round XTEA. In Topics in CryptologyCCT-RSA 2011 (pp. 250-267). Springer Berlin Heidelberg.
15. Jia, K., Yu, H., Wang, X. (2011). A Meet-in-the-Middle Attack on the Full KASUMI. IACR Cryptology ePrint Archive, 2011, 466.
16. Howgrave-Graham, N. (2007). A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In Advances in Cryptology-CRYPTO 2007 (pp. 150-169). Springer Berlin Heidelberg.
17. Lu, J., Wei, Y., Pasalic, E., Fouque, P. A. (2012). Meet-in-the-middle attack on reduced versions of the Camellia block cipher. In Advances in Information and Computer Security (pp. 197-215). Springer Berlin Heidelberg.
18. Chen, J., Li, L. (2012, January). Low data complexity attack on reduced camellia-256. In Information Security and Privacy (pp. 101-114). Springer Berlin Heidelberg.
19. Lu, J., Dunkelman, O., Keller, N., Kim, J. (2008). New impossible differential attacks on AES. In Progress in Cryptology-INDOCRYPT 2008 (pp. 279-293). Springer Berlin Heidelberg.
20. Lim, C. H. (1998). CRYPTON: A new 128-bit block cipher. NIST AES Proposal.
21. Park, J. H. (2009). Security analysis of mCrypton proper to low-cost ubiquitous computing devices and applications. International Journal of Communication Systems, 22(8), 959-969.
22. Mala, H., Dakhilalian, M., Shakiba, M. (2012). Cryptanalysis of mCryptonA lightweight block cipher for security of RFID tags and sensors. International Journal of Communication Systems, 25(4), 415-426.
23. Shakiba, M., Dakhilalian, M., Mala, H. (2013). Non-isomorphic Biclique Cryptanalysis and Its Application to Full-Round mCrypton. IACR Cryptology ePrint Archive, 2013, 141.
24. Jeong, K., Kang, H., Lee, C., Sung, J., Hong, S., Lim, J. I. (2013). Weakness of lightweight block ciphers mCrypton and LED against biclique cryptanalysis. Peer-to-Peer Networking and Applications, 1-17.
25. Kang, J., Jeong, K., Sung, J., Hong, S., Lee, K. Collision Attacks on AES-192/256, Crypton-192/256, mCrypton-96/128 and Anubis. Journal of Applied Mathematics, `http://downloads.hindawi.com/journals/jam/aip/713673.pdf`

Appendix

A The Complete Differential Characteristics Used in This Article

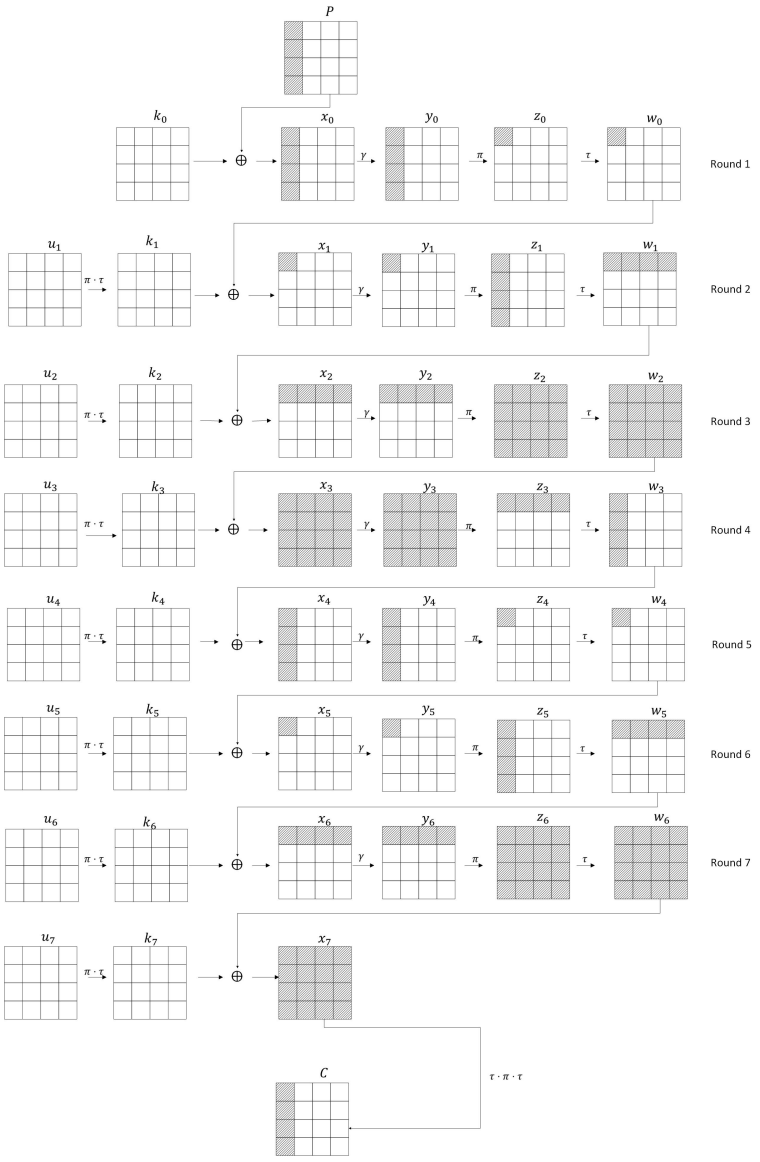

Fig. 5. Complete 7-round differential characteristic used in Section 4

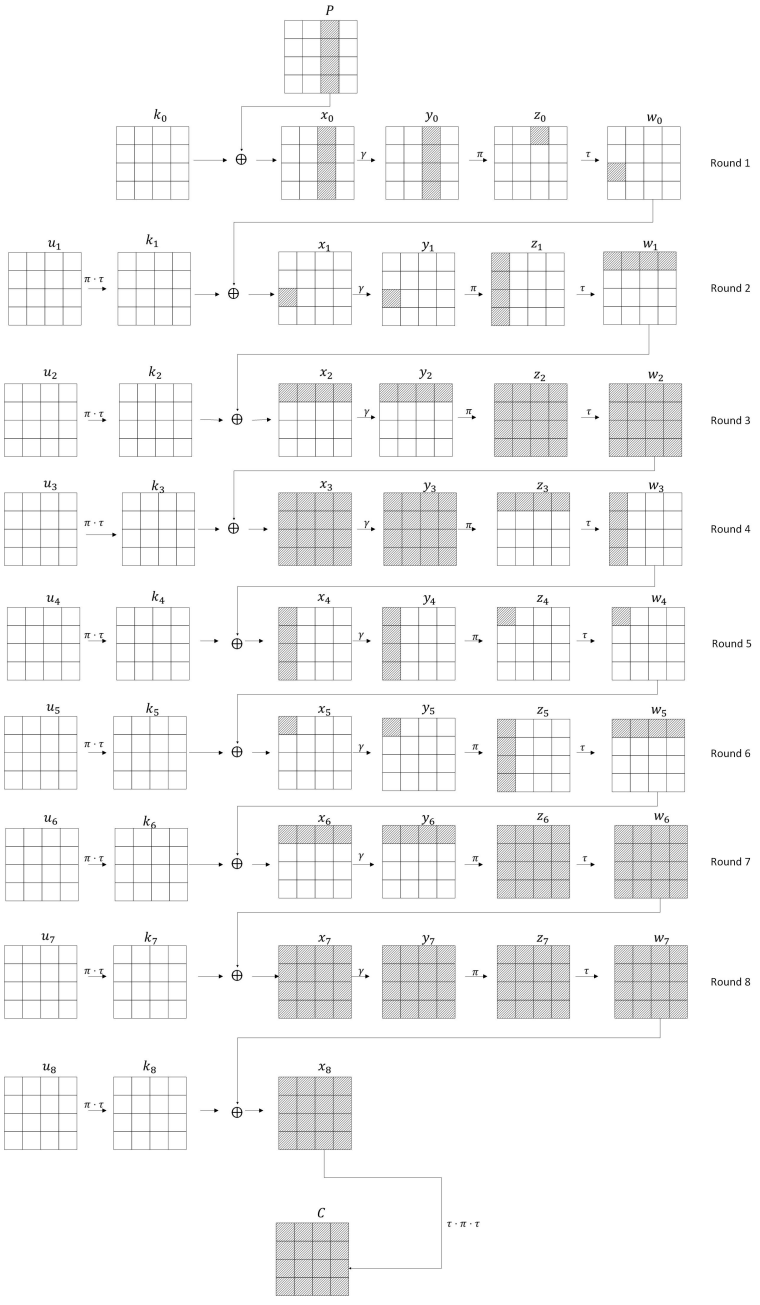

Fig. 6. Complete 8-round differential characteristic used in Section 5.1

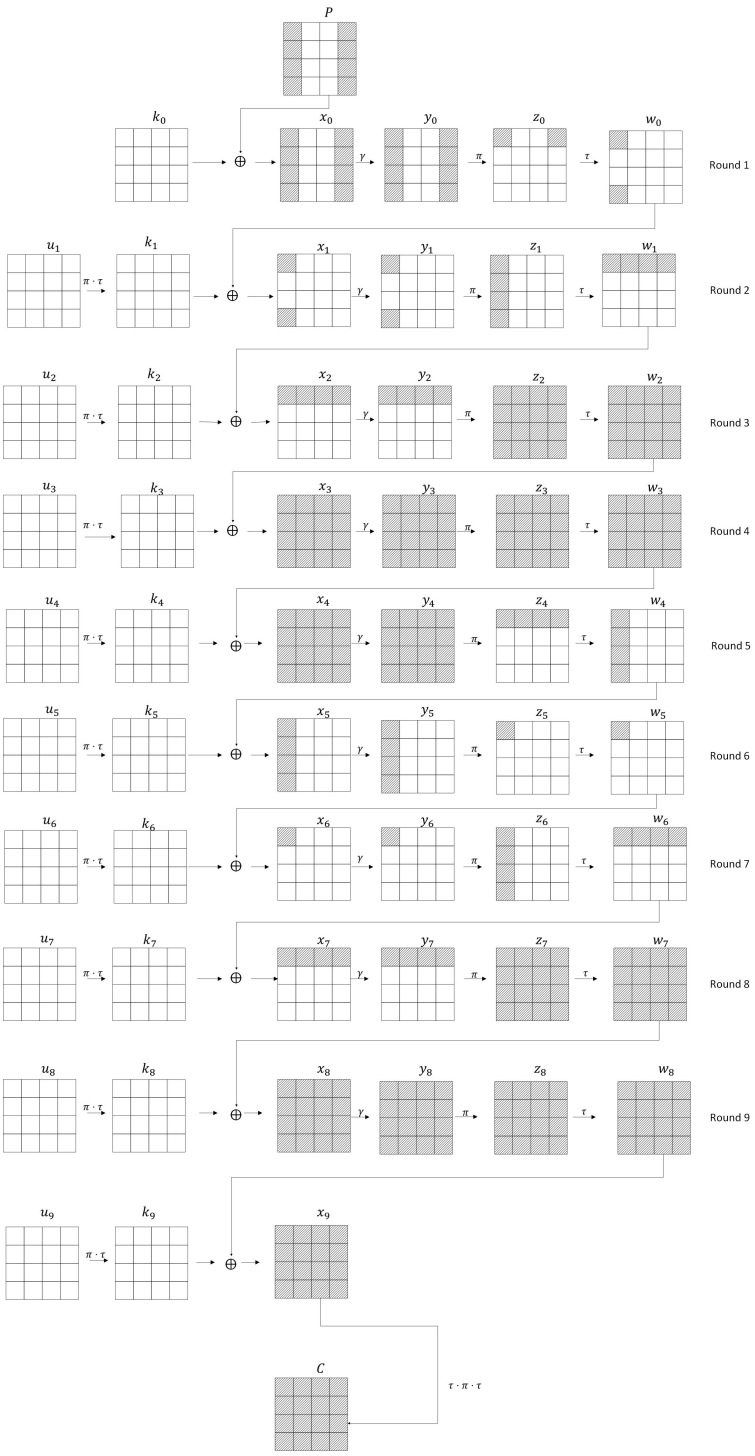

Fig. 7. Complete 9-round differential characteristic used in Section 5.2