Improved Meet-in-the-Middle Attacks on Round-Reduced Crypton-256

Yonglin Hao

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

Abstract. The meet-in-the-middle (MITM) attack has prove to be efficient in analyzing the AES block cipher. Its efficiency has been increasing with the introduction of various techniques such as differential enumeration, key-dependent sieve, super-box etc. The recent MITM attack given by Li and Jin has successfully mounted to 10-round AES-256.
Crypton is an AES-like block cipher. In this paper, we apply the MITM method to the cryptanalysis of Crypton-256. Following Li and Jin's idea, we give the first 6-round distinguisher for Crypton. Based on the distinguisher as well as the properties of Crypton's simple key schedule, we successfully launch MITM attacks on Crypton-256 reduced to 9 and 10 rounds. For 9-round Crypton-256, our MITM attack can recover the 256-bit key with a time complexity $2^{173.05}$, a memory complexity $2^{241.17}$. For the 10-round version, we give two MITM attacks. The basic attack requires a time complexity $2^{240.01}$ and memory complexity $2^{241.59}$. The time/memory complexity of the advanced MITM attack on 10-round Crypton is $2^{245.05}/2^{209.59}$. Our MITM attacks share the same data complexity 2^{113} and their error rates are negligible.

Keywords: Cryptanalysis, Crypton, MITM, Efficient Differential Enumeration Technique, Key-Dependent Sieve, Super-Box

1 Introduction

The SPN-structural block cipher Crypton [1] was proposed by Lim in 1998 as a candidate algorithm for the Advanced Encryption Standard. It process 128-bit message blocks and supports key lengths vary from 64 to 256 bits. Later at FSE 1999, the designer introduced a revisited version of this block cipher named Crypton v1.0 [2] with the Sboxes and the key schedule modified (since the method used in this paper is applicable to both Crypton and Crypton v1.0, we only use "Crypton" referring both versions without specifical announcement). Although it was Rijindael [3] rather than Crypton that was selected as the official AES standard [4], Crypton share many similarities with AES and has been studied with various methods under both single-key and related-key models.

For the conventional single-key model, D' Halluin et al. proposed a square attack [5] on 6-round Crypton at FSE 1999. In ICISC 2001, an impossible differential attack on 6-round Crypton was given in [6]. In 2010, two improved impossible differential differential attacks were given by Mala et al. [7] and mount to 7-round Crypton. In ICISC 2013, Lin et al. launched a meet-in-the-middle

attack on 7-round Crypton [8]. Improved meet-in-the-middle attacks are later given by Liu et al. in [9] that reach 8- and 9-round Crypton. There is also biclique attacks that can attack full 12-round Crypton [10]. As to the related-key model, a related-key impossible differential attack has mounted to 9-round Crypton [11].

According to previous cryptanalytic results, we can regard MITM as the most efficient method for attacking Crypton since biclique is a brute-force-like method that exhaust the whole key space with marginal complexities. The current meet-in-the-middle attack on Crypton can reach 9 rounds. In this paper, we focus on the key-recovery attacks under the single-model. We are to give improved meet-in-the-middle results on Crypton-256 utilizing the techniques that have been successfully applied to the cryptanalysis of AES.

Related Works. The meet-in-the-middle (referred as MITM hereafter) method was first introduced by Diffie and Hellman in 1977 [12]. In the past decade, we have witnessed a large number of MITM results on block ciphers ([13,14,15,16] just to name some). Its popularity can be largely attributed to its high efficiency for attacking the AES block cipher [17,18,19,20,21,22,23]. Demirci and Selçuk launched the first MITM attack on AES at FSE 2008 [17]. At ASIACRYPT 2010, Dunkelman, Keller and Shamir [20] introduced the differential enumeration technique to MITM attacks and largely reduced the memory complexities. At EUROCRYPT 2013, Derbez, Fouque and Jean [21] modified Dunkelman et al.'s attack with the rebound-like idea. They gave MITM results mounting to 8-round AES-192 and 9-round AES-256. At FSE 2014, Li et al. [22] introduced the key-dependent sieve technique and achieved the most efficient attacks on 9-round AES-192/256. This work also introduce a method of spliting the whole attack into some weak-key attacks according to the relations between the subkeys in the online phase and the precomputation phase so that the memory complexities can be diminished. Recently, Li and Jin propose new MITM attacks on AES-256. They wisely construct a unique 6-round distinguisher and, using the technique of [22], successfully attack 10-round AES-256 [23].

Our Contributions. In this paper, we only focus on the Crypton with 256-bit key length referred as Crypton-256. Following idea in [23], we construct a 6-round distinguisher for Crypton-256. With this distinguisher, we propose MITM attacks on 9-round Crypton-256. Comparing with the previous 9-round attack, our result has lower time and memory complexities. We also propose two MITM attacks, referred as the basic attack and the advanced attack respectively, on 10-round Crypton-256. The basic attack requires a time complexity $2^{240.01}$, a memory compleixty $2^{241.59}$ and a data complexity 2^{113}. The advanced attack applies the idea in [22] to split the whole attack into 2^{32} weak-key attacks which lowers the memory requirement significantly. The time/memeory/data complexity of our advanced attack is $2^{245.05}/2^{209.59}/2^{113}$. To the our knowledge, these are the best key recovery results on Crypton-256 under the single key model (except for biclique). We summarize our results along with all existing single-key results on Crypton in Table 1.

Organization of the Paper. In Section 2, we give a brief introduction to Crypton-256 and provide some properties used in our attacks. In Section 3, we construct the 6-round distinguisher on Crypton-256. Section 4 details our improved MITM attack on 9-round Crypton-256. We describe our basic and advanced MITM attack on 10-round Crypton-256 in Section 5 Finally, Section 6 concludes the whole paper.

Table 1. Key-Recovery attacks on Crypton under the single-key model.

Round	Data	Time	Memory	Method	Source
7	2^{121}	$2^{116.2}$	-	ID	[7]
7	2^{113}	2^{113}	2^{91}	MITM	[8]
7	2^{32}	$2^{81.19}$	$2^{189.78}$	MITM	[9]
8	2^{32}	2^{209}	$2^{189.58}$	MITM	[9]
9	2^{120}	$2^{208.83}$	$2^{230.15}$	MITM	[9]
9	2^{104}	$2^{208.83}$	$2^{246.15}$	MITM	[9]
9	$\mathbf{2^{113}}$	$\mathbf{2^{173.05}}$	$\mathbf{2^{241.17}}$	**MITM**	**Section 4**
10	$\mathbf{2^{113}}$	$\mathbf{2^{240.01}}$	$\mathbf{2^{241.59}}$	**MITM**	**Section 5.1**
10	$\mathbf{2^{113}}$	$\mathbf{2^{245.05}}$	$\mathbf{2^{209.59}}$	**MITM**	**Section 5.2**
12	2^{44}	2^{254}	-	Biclique	[10]

ID: impossible differential; MITM: meet-in-the-middle

2 Preliminary

In the first part of this section, we give a brief introduction to Crypton-256 that omits all details irrelevant to our attacks. We refer interested reader to [1,2] for more information. In the second part, we give some definitions and properties that are used in our attacks.

2.1 Description of Crypton-256

Crypton is a 128-bit block cipher based on SPN design. It consists of 16 8-bit bytes represented by a 4×4 matrix numbered as follows:

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

Full Crypton has 12 rounds and each round consists of the 4 transformations as follows:
Nonlinear Substitution γ. This transformation consists of nibble-wise substitutions using four 8-bit S-boxes S_0, S_1 satisfying $S_0 = S_1^{-1}$. The Sboxes of Crypton share the same property with that of AES:

Property 1. Given Δ_{in} and Δ_{out} $\mathbb{F}_{2^8}\backslash\{0\}$, the equation $S_i(x) \oplus S(x \oplus \Delta_{in}) = \Delta_{out}$, has one solution on average.

Bit Permutation π. The bit permutation transformation π is a linear transformation that mix each byte column of the 4×4 array with XOR operations. It consists of 4 column-wise permutations namely $\pi_0, \ldots, \pi_3$ that used in parallel in each Crypton round. We denote the i-th ($i = 0, \ldots, 3$) column of A (defined as (1)) by $A^i = (a_{4i}, a_{4i+1}, a_{4i+2}, a_{4i+3})^T$. For the even-number rounds (Round 0,2,4,...,10), we have $\pi(A) = \big(\pi_3(A^3), \pi_2(A^2), \pi_1(A^1), \pi_0(A^0)\big)$; for the odd-number rounds (Round

1,3,...,11), we have $\pi(A) = \big(\pi_0(A^3), \pi_3(A^2), \pi_2(A^1), \pi_1(A^0)\big)$. Since we will use the property of π_0, we detail the operation of π_0. Let $(b_0, b_1, b_2, b_3)^T = \pi_0((a_0, a_1, a_2, a_3)^T)$, we have

$$\begin{aligned} b_0 &= (m_0 \wedge a_0) \oplus (m_1 \wedge a_1) \oplus (m_2 \wedge a_2) \oplus (m_3 \wedge a_3) \\ b_1 &= (m_1 \wedge a_0) \oplus (m_2 \wedge a_1) \oplus (m_3 \wedge a_2) \oplus (m_0 \wedge a_3) \\ b_2 &= (m_2 \wedge a_0) \oplus (m_3 \wedge a_1) \oplus (m_0 \wedge a_2) \oplus (m_1 \wedge a_3) \\ b_2 &= (m_3 \wedge a_0) \oplus (m_0 \wedge a_1) \oplus (m_1 \wedge a_2) \oplus (m_2 \wedge a_3) \end{aligned} \tag{2}$$

where $m_0 = 0$xfc, $m_0 = 0$xf3, $m_0 = 0$xcf, $m_3 = 0$x3f, and $\wedge$ represents the bit-wise AND operation. We refer readers to [1] for the definition of π_1, π_2 and π_2. The branch number of π_i $(i = 0, \ldots, 3)$ is 4. So we have Property 2.

Property 2. With the knowledge of any 5 out of the 8 input/output bytes of π_i operation, the other 3 bytes can also be determined uniquely.

Transposition τ The τ operation simply rearrange the positions of the bytes as follows:

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

Key Addition σ**:** σ_K is a simple bit-wise XOR the 16-byte state with the 16-byte key K, which is exactly the same with the AddRoundKey operation of AES.

Before the ecryption, Crypton-256 expand its 256-bit masterkey K to 13 subkeys 128-bit denoted as $k_0, \ldots, k_{12}$ through a key schedule that will described later. Then, for round number $r = 1, \ldots, 12$, we define the round function

$$\rho_r(\cdot) = \sigma_{k_r} \circ \tau \circ \pi \circ \gamma(\cdot)$$

We also define the linear transformation $\Phi(\cdot) = \tau \circ \pi \circ \tau(\cdot)$ So, for the plaintext P, the ciphertext C after a r-round Crypton-256 encryption can be summarized as:

$$C = \Phi \circ \rho_r \circ \ldots \circ \rho_1 \circ \sigma_{k_0}(P)$$

Key Schedule. For the 256-bit masterkey K, Crypton-256 first process it with a nonlinear operation to another 256-bit expanded key. Since this is a 1-to-1 projection, we can still use K to represent the expanded key. The 128-bit subkeys k_0 and k_1 are first initialized with the lowest significant 128 bits and the most significant 128 bits of K respectively. Then, for $i = 1, \ldots, 6$, k_{2i} (k_{2i+1}) is derived from k_{2i-2} (k_{2i-1}) with simply rotations and XORing round constants. So we have the following property.

Property 3. The subkeys $k_0, \ldots, k_{12}$ of Crypton-256 satisfy: the knowledge of any k_{2i} (k_{2i+1}) for any $i \in [0, 6]$ $(i \in [0, 5])$ is sufficient to deducing all $k_0, k_2, \ldots, k_{12}$ $(k_1, k_3, \ldots, k_{11})$. Furthermore, the knowledge of one byte in k_{2i} (k_{2i+1}) can uniquely one byte in every subkey $k_0, k_2, \ldots, k_{12}$ $(k_1, k_3, \ldots, k_{11})$.

Property 3 exists in both Crypton and Crypton v1.0. We need to utilize the key-byte relationship between k_5 and k_1 in our construction of distinguishers, so we specify the following two properties for Crypton and Crypton v1.0 respectively.

Property 4. In Crypton, the knowledge of $k_5[0,\ldots,7]$ can deduce 8 bytes of k_1 namely: $k_1[0,7,9,10,11,12,13,14]$; the knowledge of $k_4[4,7,10,13]$ can deduce 4 bytes $k_0[0,\ldots,3]$.

Property 5. In Crypton, the knowledge of $k_5[0,\ldots,7]$ can deduce 8 bytes of k_1 namely: $k_1[2,3,5,7,8,9,12,14]$; the knowledge of $k_4[2,4,9,15]$ can deduce $k_0[0,\ldots,3]$

Properties 3, 4 and 5 can be easily deduced referring to the key schedules in [1] and [2]. They can help us reduce the complexities of our attacks on a large scale.

2.2 Definitions and Properties of Crypton-256

Throughout the paper we use the following definitions and properties in our attack. Before our descriptions, we give the following notations that we use through this paper.

State $\mathbf{x_r^i}$: The 128-bit Crypton states are represented by different small letters (except for plaintexts P and ciphertexts C). 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 $u_r = \tau \circ \pi(k_r)$. The difference of state x is denoted by Δx. Besides, the superscript represents the position that the state lies in a sequence (or set).

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

Bit-wise operators:

$\wedge$ bit-wise AND.

$\|$ concatenate two strings of bits.

$\oplus$ bit-wise XOR.

Definition 1. ***(σ-set of Crypton)*** *A σ-set is a set of 256 128-bit Crypton-states that are all different in one byte (the active byte) and all equal in the other state bytes (the inactive bytes).*

Definition 2. ***(Super-box of Crypton)*** *Consider a 1-round encryption of Crypton:*

$$x_r \xrightarrow{\gamma} y_r \xrightarrow{\pi} z_r \xrightarrow{\tau} w_r \xrightarrow{\oplus k_r} x_{r+1} \xrightarrow{\gamma} y_{r+1}$$

The whole process can be divided into 4 Super-boxes $SSB_0,\ldots,SSB_3$ as

$$SSB_i : x_r[4i,\ldots,4i+3] \rightarrow y_{r+1}[i,i+4,i+8,i+12], \quad \textit{for } i \in [0,3].$$

The operation of SSB_i requires the knowledge of 4 key bytes $k_r[i,i+4,i+8,i+12]$ and it can be regarded as a 32-to-32 Sbox operation. Applying Property 1 to the super-boxes of Crypton, we acquire Property 6.

Property 6. Given Δ_{in} and Δ_{out} $\mathbb{F}_{2^{32}}\backslash\{0\}$, the equation $SSB_i(x) \oplus SSB(x \oplus \Delta_{in}) = \Delta_{out}$, has one solution on average.

3 The 6-Round Distinguisher for Crypton-256

Using [23]'s ideas, we construct a 6-round distinguisher on Crypton-256 using the differential enumerate technique and key-dependent sieve technique. According to Property 2, we further establish the equation in π of round 6 that

$$\begin{aligned}(m_0 \wedge y_6[0]) \oplus (m_3 \wedge y_6[2]) = (m_1 \wedge m_3 \oplus m_0) \wedge z_6[0] \\ \oplus (m_0 \wedge m_1 \oplus m_2 \wedge m_3) \wedge z_6[1] \\ \oplus (m_0 \wedge m_2 \oplus m_3) \wedge z_6[2]\end{aligned} \tag{3}$$

Let $e_{in} = m_0 \wedge y_6[0] \oplus m_3 \wedge y_6[2]$ and

$$\begin{aligned}e_{out} = (m_1 \wedge m_3 \oplus m_0) \wedge z_7[0] \\ \oplus \{(m_0 \wedge m_1) \oplus (m_2 \wedge m_3)\} \wedge x_7[4] \\ \oplus (m_0 \wedge m_2 \oplus m_3) \wedge x_7[8]\end{aligned}$$

then we have

$$\begin{aligned}e_{out} = e_{in} \oplus (m_1 \wedge m_3 \oplus m_0) \oplus k_7[0] \\ \oplus \{(m_0 \wedge m_1) \oplus (m_2 \wedge m_3)\} \oplus k_7[4] \\ \oplus (m_0 \wedge m_2 \oplus m_3) \oplus k_7[8].\end{aligned}$$

According to Property 4 and Property 5, the knowledge of $k_5[0, \ldots, 7]$ can deduce $k_1[12]$ for both Crypton and Crypton v1.0. Therefore, we construct a truncated differential characteristic in Figure 1 that can be used in both Crypton and Crypton v1.0. We also prove Theorem 1 that bounds the memory requirements of our attacks. The distinguisher as well as Theorem 1 follows the idea of Li and Jin in [23]. The proof of Theorem 1 uses both the differential enumeration technique of [21] and the key-dependent sieve of [22].

Theorem 1. *Let $(w_0, \ldots, w_{255})$ be a σ-set where the position of the active byte is $w_0[12]$, and $w_0^t[12] = t$ for $t \in [0, 255]$. Consider the encryption of the first 33 values $(w^0, \ldots, w^{32})$ of the σ-set through 6-round Crypton-256, in the case of that a message pair (w_0^i, w_0^j) of the σ-set conforms to the truncated differential characteristic in Figure 1, then the corresponding 256-bit ordered sequence $(e_{out}^1 \oplus e_{out}^0, \ldots, e_{out}^{32} \oplus e_{out}^0)$ only takes about 2^{240} values (out of the 2^{256} theoretical ones).*

Proof. Firstly, we show that, without the condition that a message pair (w_0^i, w_0^j) of the σ-set conforms to the truncated differential characteristic in Figure 1, the sequence $(e_{out}^1 \oplus e_{out}^0, e_{out}^2 \oplus e_{out}^0, \ldots, e_{out}^{32} \oplus e_{out}^0)$ is computed by the 46 bytes:

$$w_0^i[12] \| x_1^i[12] \| x_2^i[3, 7, 11, 15] \| x_3^i \| k_4 \| k_5[0, \ldots, 7] \tag{4}$$

In the following proof, we denote difference $x^m \oplus x^i$ by Δx^m ($m \in [0, 255]$). In round 0, it is explicit to compute $\Delta x_1^m[12]$ by the equation $\Delta x_1^m[12] = \Delta w_0^m[12] = w_0^m[12] \oplus w_0^i[12]$ with $w_0^i[12]$. As the value $x_1^i[12]$ is known, we can compute $\Delta y_1^m[12]$ with $\Delta y_1^m[12] = S(x_1^i[12]) \oplus S(\Delta x_1^m[12] \oplus x_1^i[12])$. Since the π, τ and σ_{k_r} operations are linear, we can further obtain the difference $\Delta x_2^m[3, 7, 11, 15]$

since $\Delta y_1^m[t] = 0$ for $t \neq 12$. Similarly, the knowledge of $x_2^i[3,7,11,15]$ enable us to acquire Δx_3^m. Since x_3^i and x_3^m are both known, we acquire the state values w_3^m and w_3^i. We can encrypt one more rounds with k_4 to acquire w_4^m and w_m^i. With the knowledge of w_4^m, w_m^i and $k_5[0,\ldots,7]$, we can acquire $w_5^m[0,\ldots,7]$ and $w_5^i[0,\ldots,7]$. Since we can deduce $k_6[0,1]$ from k_4 and acquire $y_6^m[0,1]$ and $y_6^i[0,1]$, we finally acquire e_{in}^m. According to (3), we have $e_{in}^m \oplus e_{in}^0 = e_{out}^m \oplus e_{out}^0$ for $m \in [1,32]$, which proves (4).

Adding the condition that w^i belongs to a pair (w^i, w^j) conforming the truncated differential Figure 1, we can determine the sequence with the following 33 bytes.

$$\Delta^j y_1[12] \| x_2^i[3,7,11,15] \| y_4^i \| y_5^i[0,\ldots,7] \| y_6^i[0,1] \| \Delta y_6^j[0,1] \tag{5}$$

We start by deducing forward. Since (w^i, w^j) follows the differential propagation in Figure 1, the knowledge of $\Delta y_1^j[12]$ is sufficient for us to acquire the difference $\Delta x_2^j[3,7,11,15]$. Adding the value $x_2^i[3,7,11,15]$, we can compute $x_2^j[3,7,11,15]$ and further deduce the difference Δx_3^j.

In the backward direction, we have known $\Delta^j y_6[0,1]$ and $\Delta z_6^j[0] = \Delta y_6^j[0,1] = 0$ so we can acquire the differences $\Delta z_6^j[0,1,2]$ according to Property 2. Combining the difference $\Delta y_6^j[0,1]$ with the value $y_6^i[0,1]$, we can compute backward to $\Delta y_5^j[0,\ldots,7]$. With the knowledge of the value $y_5^i[0,\ldots,7]$, we can compute backward to acquire Δy_4^j. Finally, adding y_4^i, we deduce the difference of Δy_3^j.

For each pair of difference $(\Delta x_3^j, \Delta y_3^j)$, we can acquire 1 corresponding $x_3^i \| y_3^i$ on average (Property 1). Besides, we can also deduce k_4 with x_3^i and y_4^i, deduce $k_5[0,\ldots,7]$ with $y_5^i[0,\ldots,7]$ and y_4^i, deduce $k_6[0,1]$ from $y_6^i[0,1]$ and $y_5^i[0,\ldots,7]$. Since the knowledge of k_4 can also deduce the value of $k_6[0,1]$ according to Property 3, $k_6[0,1]$ is a 16-bit filter. There is another limitation in $\Delta^j y_6[0,1]$: when $\Delta y_6^j[2,3] = 0$, only 2^8 out of the 2^{16} possible $\Delta^j y_6[0,1]$'s can make sure $\Delta z_6^j[0] = 0$. Therefore, the 2-byte difference $\Delta^j y_6[0,1]$ can only take 2^8 rather than 2^{16} values. We list all conforming $\Delta^j y_6[0,1]$'s in Appendix A. So the strength of filtering is $16 + 8 = 24$ bits.

The knowledge of $k_5[0,\ldots,7]$ enable us to deduce $k_1[12]$ (Property 4 and Property 5). We can also acquire k_2 from k_4. With k_2 and $k_1[12]$, we can decrypt $x_2^i[3,7,11,15]$ and finally acquire $w_0^i[12]$. Considering the 24-bit filtering, the sequence $(e_{out}^1 \oplus e_{out}^0, \ldots, e^{32} \oplus e_{out}^0)$ can take $2^{8\times 33-24} = 2^{240}$ values. □

4 The Attack on 9-Round Crypton-256

We apply the 6-round distinguisher in Section 3 to attack 9-round Crypton-256 by adding 1 round at the beginning and 2 rounds at the end. The extended truncated differential characteristic can be seen in Figure 2. The probability for a plaintext pair (P, P') to conform the characteristic is 2^{-144}. The procedure of the attack can be detailed as follows:

Precomputation Phase. In the precomputation phase, we construct a lookup table T_0 containing the 2^{240} sequences $(e_{out}^1 \oplus e_{out}^0, \ldots, e^{32} \oplus e_{out}^0)$ along with additional information to enhance the success probability of our attacks. We also construct another table T_1 to further save the time complexity of the online phase. The procedure of constructing T_0 and T_1 is described as follows:

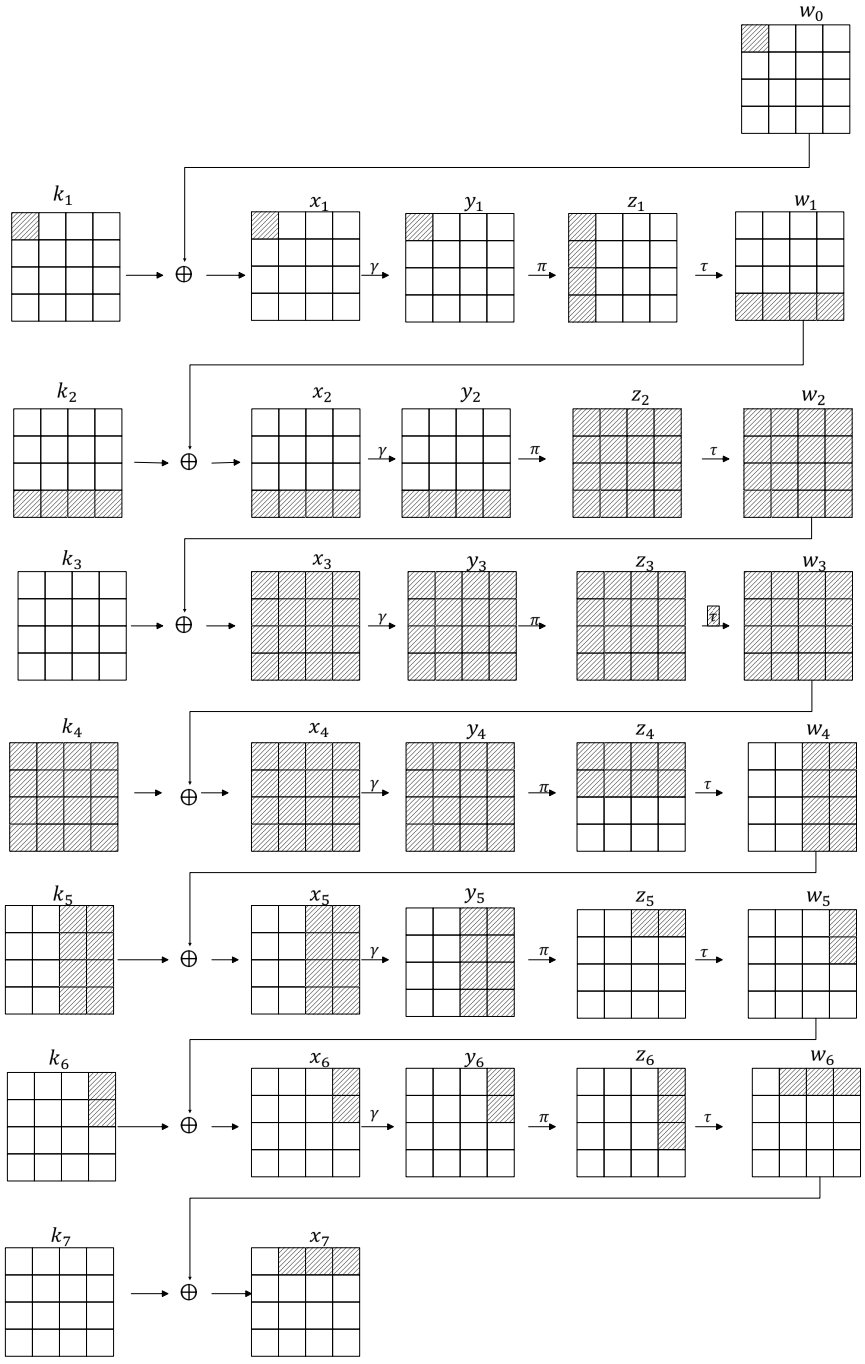

Figure 1. The 6-round distinguisher used in our attacks.

1. Initialize the lookup table T_0 as empty. For 2^{128} possible k_4, we do the following steps to construct T_0:
   (a) With the knowledge of k_4, deduce the 8 key bytes $k_6[0,\ldots,7]$.
   (b) Construct a table T_2 containing the information on the backward deduction by taking the following substeps:
      i. Guess $y_5[0,\ldots,7]$ and compute forward to $y_6[0,1]$ using k_6.
      ii. Guess $\Delta y_6[0,1]$ and assign $\Delta y_6[2,3]=0$. Compute forward to $\Delta z_6[0,\ldots,3]$ and reserve the 2^8 possible $\Delta y_6[0,1]$'s (Appendix A) that makes $\Delta z_6[3]=0$.
      iii. Combining the acquired information, we can compute backward to acquire both $x_5[0,\ldots,7]$ and $\Delta x_5[0,\ldots,7]$.
      iv. Store all possible $x_5[0,\ldots,7]$'s in T_2 under the index $\Delta x_5[0,\ldots,7]$. For each of the 2^{64} possible $\Delta x_5[0,\ldots,7]$'s, there are averaging 2^8 possible $x_5[0,\ldots,7]$ attached.
   (c) Construct a table T_3 containing the information on the forward deduction by taking the following substeps:
      i. Guess the 96-bit $\Delta y_2[3,7,11,15]\|\Delta x_5[0,\ldots,7]$ and deduce the differences $\Delta x_3\|\Delta y_4$.
      ii. With k_4, we acquire 1 conforming $x_3\|y_4$ on average according to Property 6.
      iii. We store $x_3\|\Delta x_5[0,\ldots,7]$ in T_3 indexed by $\Delta y_2[3,7,11,15]$. Each of the 2^{32} $\Delta y_2[3,7,11,15]$ is followed by 2^{64} corresponding $x_3\|\Delta x_5[0,\ldots,7]$'s.
   (d) For all 2^{40} possible $\Delta y_1[12]\|x_2[3,7,11,15]$'s, we match the two tables T_2, T_3 and gradually construct T_0 by taking the following steps:
      i. Deduce k_2 from k_4 (Property 4) and further acquire $y_1[12]\|x_1[12]$ with the knowledge of $k_2\|x_2[3,7,11,15]$.
      ii. Combine $y_1[12]$ and $\Delta y_1[12]$ to acquire $\Delta x_2[3,7,11,15]$. Adding the knowledge of $x_2[3,7,11,15]$, we further deduce $\Delta y_2[3,7,11,15]$.
      iii. Lookup the acquired $\Delta y_2[3,7,11,15]$ in T_3 and find averaging 2^{64} corresponding $x_3\|\Delta x_5[0,\ldots,7]$'s.
      iv. For each of the 2^{64} $x_3\|\Delta x_5[0,\ldots,7]$'s, we lookup T_2 and find 2^8 $x_5[0,\ldots,7]$'s. As k_4 is known, we can deduce w_4 from x_3 and further acquire $k_5[0,\ldots,7]$ with the knowledge of $x_5[0,\ldots,7]$.
      v. After deducing $k_1[12]$ from $k_5[0,\ldots,7]$, we acquire the value $w_0[0]$ through partial decryptions from $y_1[12]$.
      vi. Compute the sequence $(e^1_{out}\oplus e^0_{out},\ldots,e^{32}_{out}\oplus e^0_{out})$.
      vii. Deduce k_0 from k_4 and store $k_0[0,\ldots,3]$ along with the sequence in T_0 as $(e^1_{out}\oplus e^0_{out},\ldots,e^{32}_{out}\oplus e^0_{out})\|k_0[0,\ldots,3]$.
2. We construct another table T_1 for saving the time in the online phase:
   (a) For all the 120-bit subkey of $u_9[\lambda]||u_8[0,4,8]$ and the 96-bit value of $\overline{w}_8[\lambda]$ and obtain the values e_{out} where λ is the set of indices defined as

$$\lambda := \{0,1,2,4,5,6,8,9,10,12,13,14\} \tag{6}$$

   and will be frequently referred in the remainder of this paper.
   (b) Store e_{out} with the index of $u_8[0,4,8]\|u_9[\lambda]||\overline{w}_8[\lambda]$ in table T_1.

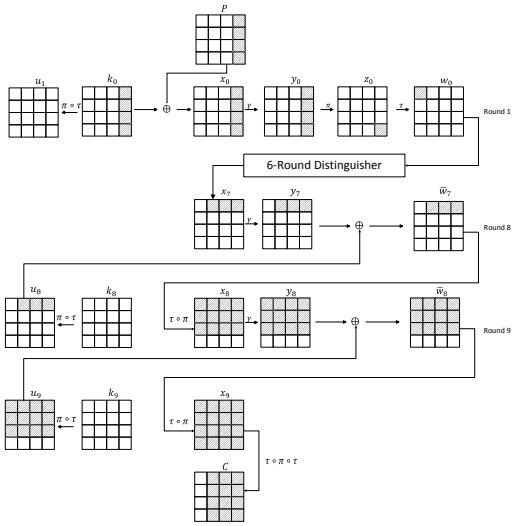

Figure 2. The truncated differential characteristic for attacking 9-round Crypton-256. The key bytes in shadow are deduced in Step 3 of the online phase.

Online Phase. We first find one message pair satisfying the truncated differential characteristic in Figure 2. Then we identify the σ-set, calculate the sequence $(e^1_{out} \oplus e^0_{out}, \ldots, e^{32}_{out} \oplus e^0_{out})$ and detect whether it belongs to the table T_0 built in the precomputation phase. The procedure of our attack is as follow:

1. Construct 2^{81} plaintext structures and query for their ciphertexts. In each structure, there are 2^{32} plaintexts s.t. $P[0,\ldots,3]$ take all the possible values, and the remaining 12 bytes are fixed to some constants. There are $\binom{2^{32}}{2} \approx 2^{63}$ plaintext pairs only having non-zero differences in byte positions 0,1,2,3. Therefore, we have now acquired $2^{81+63} = 2^{144}$ plaintext (P, P')'s (whose corresponding ciphertexts are (C, C')) conforming the starting point of the truncated characteristic. Since the characteristic has a probability 2^{-144}, we expect to find 1 pair follows the whole differential propagation in Figure 2.
2. Within each structure, select the pairs satisfying $\Delta C[12,\ldots,15] = 0$. This is a 32-bit filter so 2^{112} out of the 2^{144} pairs will remain.
3. For each of the remaining pair (P, P') (the corresponding (C, C') is also known), we do the following substeps:
   (a) Guess $\Delta w_0[12]$ and, assuming $\Delta w_0[0,4,8] = 0$, deduce the difference $\Delta y_0[0,\ldots,3]$. Combining $\Delta x_0[0,\ldots,3] = \Delta P[0,\ldots,3]$, we can deduce 1 value $x_0[0,\ldots,3] \| y_0[0,\ldots,3]$ on average (Property 1). We also deduce $k_0[0,\ldots,3]$ with the knowledge of $P[0,\ldots,3] \| x_0[0,\ldots,3]$.
   (b) Guess $\Delta y_7[0,1,2]$ and, assuming $\Delta y_7[3,\ldots,15] = 0$, deduce linearly the difference Δx_8. Compute Δy_8 as $\Delta y_8 = \Delta \overline{w}_8 = \pi \circ \tau(\Delta C)$. For each nonlinear difference $\Delta x_8[\lambda] \| \Delta y_8[\lambda]$, we can find 1 corresponding $x_8[\lambda] \| y_8[\lambda]$ according to Property 1 where λ is defined as (6). Since $x_9 = \tau \circ \pi \circ \tau(C)$,we linearly deduce $\overline{w}_8 = \tau \circ \pi(x_9)$ and further acquire the key $u_9[\lambda] = y_8[\lambda] \oplus \overline{w}_8[\lambda]$.

(c) Guess the subkey $u_8[0,1,2]$. With the knowledge of $k_0[0,\ldots,3]$, we start from P and acquire $w_0[0,4,8,12]$. Assign $w_0[0]$ to $0,\ldots,32$, acquire the corresponding $P^0,\ldots,P^{32}$ through partial decryptions and identify the ciphertexts $C^0,\ldots,C^{32}$ simultaneously. With $u_9[\lambda]||u_8[0,4,8]$ and $\overline{w}_8^t$ (linearly deduced from C^t for $t = 0,\ldots,32$), we can acquire e_{out}^t as well as the sequence $(e_{out}^1 \oplus e_{out}^0,\ldots,e_{out}^{32} \oplus e_{out}^0)$ by referring to the table T_1. Check whether the string $(e_{out}^1 \oplus e_{out}^0,\ldots,e_{out}^{32} \oplus e_{out}^0)\|k_0[0,\ldots,3]$ lies in T_0. If a match has been found, we identify the subkey $u_9[\lambda]\|u_8[0,4,8]$ as the correct key guess. Otherwise, go back to Step (a) (or change a new pair (P,P') and restart Step (a) when the all 2^{56} possible $\Delta w_0[0]\|\Delta y_7[0,1,2]\|u_8[0,1,2]$'s have been tested). The probability for a wrong guess to pass this test is $2^{240} \times 2^{-8\times 36} = 2^{-48}$

4. Now that we have acquired a candidate $u_9[\lambda]\|u_8[0,1,2]$, we exhaustively search the remaining 136-bit $u_9[3,7,11,15]\|u_8[3,\ldots,15]$ to recover the whole 256-bit masterkey.

Complexity Analysis: For each of the 2^{128} k_4's, the construction of T_2 in Step 2 requires $2^{8\times(8+2)} = 2^{80}$ encryptions. T_2 contains $2^{64+8} = 2^{72}$ records. The table T_3 in Step 3 requires 2^{96} encryptions and contains 2^{96} entries. The matching operations in Step 4 requires $2^{40} \times 2^8 \times 2^{64} \times 33 \approx 2^{117.05}$ encryptions. So the time complexity of the precomputation phase is $2^{128} \times (2^{80} + 2^{96} + 2^{117.05}) \approx 2^{245.05}$. The table T_0 contains 2^{240} entries and each of them has 36 bytes. We need $2^{240} \times 36/16 \approx 2^{241.17}$ 128-bit blocks to store T_0, which is also the memory complexity of the whole attack. The time complexity of the online phase is dominated by Step 3.(c) which is $2^{112} \times 2^8 \times 2^{24} \times 2^{24} \times 33 \approx 2^{173.05}$. The data complexity is $2^{81} \times 2^{32} = 2^{113}$. The successful probability of this attack is $1 - 2^{-48}$ according to Step 3(c).

5 Meet-in-the-Middle Attacks on 10-Round Crypton-256

We extend the attack in Section 4 by 1-round and acquire attacks on 10-round Crypton-256. We first describe the our basic attack in Section 5.1. Then, in Section 5.2, we show the advanced attack that optimizes the complexities by dividing the whole attack into a series of weak-key attacks, which is exactly the method used in [22,23]. The truncated differential characteristic can be seen in Figure 3

5.1 The Basic Attack

Our basic attack on 10-round Crypton-256 also consists of the precomputation phase and the online phase.

Precomputation Phase. The precomputation phase is identical to that of the previous section except for Step 1.(c).vii which is slightly changed as

– Deduce k_{10} from k_4 and store k_{10} along with the sequence in T_0 as $(e_{out}^1 \oplus e_{out}^0,\ldots,e_{out}^{32} \oplus e_{out}^0)\|k_{10}$.

With this change, we can acquire a higher success probability in the 10-round attack.

Online Phase. The identification of the pair conforming the truncated differential characteristic in Figure 3 is slightly complicated. The procedure is as follow:

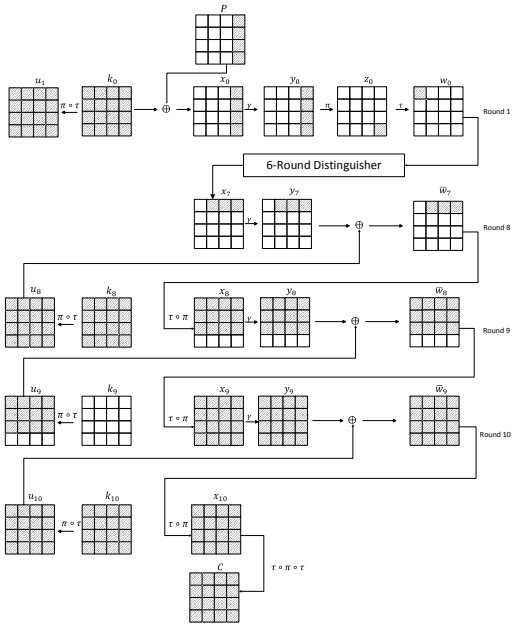

Figure 3. The truncated differential characteristic for attacking 10-round Crypton-256.

1. Exactly the same with the 9-round attack, we construct 2^{81} plaintext structures, query for their ciphertexts and acquire 2^{144} pairs in total.
2. For each of the 2^{144} pairs (P, P'), we do the following substeps:
   (a) Guess $\Delta y_8[\lambda]$ and deduce Δx_9 where λ is defined as (6). Deduce Δy_9 from ΔC and further acquire 1 $x_9\|y_9$ according to Property 1. The knowledge of y_9 and C further enable us to attain the whole k_{10}. Deduce k_0 from k_{10} and compute from (P, P') to the corresponding Δw_0. Discard the guess of $\Delta y_8[\lambda]$ if $\Delta w_0[0, 4, 8] \neq 0$. This is a 24-bit filter so the guess can go through this step with a probability 2^{-24}.
   (b) Guess $\Delta y_6[0, 1]$ among the 2^8 possibilities in Appendix A to conform the characteristic and compute forward to Δx_7. Deduce k_8 from k_{10}. Since $\Delta x_7\|k_8\|\Delta y_8[\lambda]$ are known, we can use Property 6 to acquire 1 $x_7[\lambda]\|y_8[\lambda]$. Since x_9 is also known, we acquire $u_9[\lambda]$ as well. Deduce k_0 from k_{10} and acquire the value w_0 from P. Change the value of $w_0[12]$ to $0, \ldots, 32$, compute backward to acquire the corresponding plaintexts $P^0, \ldots, P^{32}$ and ciphertexts $C^0, \ldots, C^{32}$.
   (c) For $2^{96-24} = 2^{72}$ deduced subkeys k_{10} passed through (a), we further acquire k_8 and its equivalence u_8. With the knowledge of $u_8[0, 4, 8]\|u_9[\lambda]\|k_{10}$, we compute backward to $\overline{w}_8[\lambda]$, check the table T_1 for the corresponding e_{out}, and finally acquire the sequence $(e^1_{out} \oplus e^0_{out}, \ldots, e^{32}_{out} \oplus e^0_{out})$. Check whether the combination $(e^1_{out} \oplus e^0_{out}, \ldots, e^{32}_{out} \oplus e^0_{out})\|k_{10}$ lies in the precomputed table T_0. If a match has been found, we go to Step 3. Otherwise, we discard the guess and go to Step (a) (or try a new (P, P') pair and restart Step (a) if the $\Delta y_8[\lambda]\|\Delta y_6[0, 1]$ of the current pair has been exhausted).

3. Now that we have acquired a candidate $u_9[\lambda]\|k_{10}$, we exhaustively search the remaining $u_9[3,7,11,15]$ to recover the whole 256-bit masterkey.

Complexity Analysis. The time complexity for constructing T_0 is still $2^{245.05}$. Since the each one of the 2^{240} T_0 entries is expanded from 36 to 48 bytes, the memory complexity increases accordingly to $2^{240} \times 48/16 \approx 2^{241.59}$ 128-bit blocks. In the online phase, the complexity of Step 1 is $2^{32+81} = 2^{113}$. In Step 2, for each of the 2^{144} pairs, the complexity of (a) is 2^{96}; the complexities of (b) and (c) are $2^{72+8} = 2^{80}$ and $2^{72+8} \times 33 \approx 2^{85.04}$ respectively. So the complexity of Step 2 in the online phase can be computed as $2^{144} \times (2^{96} + 2^{80} + 2^{85.04}) \approx 2^{240.01}$. The complexity of Step 3 of the online phase is only 2^{32}. Therefore, the overall complexity of the online phase is dominated by that of Step 2's $2^{240.01}$. The data complexity is still 2^{113}. The whole attack requires $2^{32+81} = 2^{113}$ plaintexts so the data complexity is 2^{113} as well. As to the success probability, the right pair combined with correct key guess can pass Step 2 with probability 1. An incorrect combination of plaintext pair and key guess can pass Step 2.(a) with probability 2^{-24}, Step (c) with probability $2^{240} \times 2^{-8\times(32+16)} = 2^{-264}$. So the success probability of the whole attack is no less than $1 - 2^{-288}$. To sum up, our attack on 10-round Crypton-256 can recover the whole 256-bit key with a time complexity $2^{240.01}$, a memory compleixty $2^{241.59}$, a data complexity 2^{113} and a negligible error rate 2^{-288}.

5.2 The Advanced Attack

In [18], Li et al. present that the whole attack can be divided into a series of weak-key attacks according to the relations between the subkeys in the online phase and the precomputation phase. This method has also been used in the 10-round attack on AES-256 in [23]. The linear expansion of the Crypton-256 key schedule enables us to make such a tradeoff even easier. In the precomputation phase, we only need to compute the table T_1 described as Section 5.1. And the attack procedure of the online phase is described as follows:

1. Same with Step 1 of the online phase in Section 5.1, we acquire 2^{144} plaintext pairs (P, P') conforming the difference and their ciphertexts are also known.
2. For each of the 2^{32} possible $k_4[i_0, \ldots, i_3]$, do the following substeps:
   Note: $(i_0, \ldots, i_3) = (4, 7, 10, 13)$ for Crypton according to Property 4 and $(i_0, \ldots, i_3) = (2, 4, 9, 15)$ for Crypton v1.0 following Property 5.
   (a) Guess the other 12 bytes of k_4 and construct the subtable T_0^* as described in the Section 5.1. T_0^* contains of 2^{208} $(e^1_{out} \oplus e^0_{out}, \ldots, e^{32}_{out} \oplus e^0_{out})\|k_{10}$.
   (b) Deduce the value $k_0[0, \ldots, 3]$ from $k_4[i_0, \ldots, i_3]$. Within each of the 2^{81} plaintext structures, partially encrypt the plaintexts to acquire $w_0[0, 4, 8, 12]$ and identify the pairs s.t. $\Delta w_0[0, 4, 8] = 0$. This is a 24-bit filter and $2^{63-24} = 2^{39}$ pairs are left in each structure, which makes 2^{120} remaining pairs in total.
   (c) Guess the unknown 12 bytes of k_4 and deduce the whole $k_{10}\|k_0$ from k_4. Within each of the 2^{81} structures (each structure contains 2^{39} pairs), we acquire $\overline{w}_8$ with k_{10} (or its equivalence u_{10}) and identify the pairs s.t. $\Delta\overline{w}_8[3, 7, 11, 15] = 0$. This is a 32-bit filter and $2^{39-32} = 2^7$ pairs are left within each structure making 2^{88} pairs in total.

(d) For each of the 2^{88} remaining pairs (P, P'), we do the following substeps:
   i. Deduce k_8 from k_{10}. Partially decrypt the ciphertext to acquire x_9 and $\overline{w}_8$, and further compute $\Delta y_8[\lambda]$. Guess the difference $\Delta y_6[0,1]$ among the 2^8 possible values in Appendix A and acquire the corresponding Δx_7. Since k_8 is known, we can acquire 1 value of $x_7[\lambda]\|y_8[\lambda]$ on average each of the combination $\Delta x_7\|k_8\|\Delta y_8[\lambda]$ (Property 6). We further deduce $u_9[\lambda]$ with the knowledge of $y_8[\lambda]$ and x_9.
   ii. Deduce k_0 from k_{10} and acquire w_0 from plaintexts. Change the value of $w_0[12]$ to $0, \ldots, 32$, compute backward for plaintexts $P^0, \ldots, P^{32}$ and find the corresponding ciphertexts $C^1, \ldots, C^{32}$.
   iii. For $t = 0, \ldots, 32$, we acquire the $\overline{w}_8^t[\lambda]$ with the knowledge $k_{10}\|u_9[\lambda]$ from C^t. Check T_1 and get e_{out}^t and further acquire the sequence $(e_{out}^1 \oplus e_{out}^0, \ldots, e_{out}^{32} \oplus e_{out}^0)$. Lookup T_0^* for the $(e_{out}^1 \oplus e_{out}^0, \ldots, e_{out}^{32} \oplus e_{out}^0)\|k_{10}$. Discard the subkey if no match can be found.

(e) For each $k_4[i_0, \ldots, i_3]$, after Step (d), there are about $1 + 2^{96} \times 2^{88} \times 2^8 \times 2^{-276} \approx 1$ subkeys remaining.

3. After the 2^{32} sub-attacks, there are $2^{32} \times 1 = 2^{32}$ subkeys $k_{10}\|u_8[\lambda]$ remaining. So we exhaustively search for the 2^{64} remaining $k_{10}\|u_8[\lambda]\|u_8[3, 7, 11, 15]$ to recover the full 256-bit key.

Complexity Analysis. The memory complexity is dominated by T_0^*'s $2^{208} \times 48/16 = 2^{209.59}$. Since the construction of the whole T_0 requires a time complexity $2^{245.05}$ as is mentioned in Section 4, the construction of one T_0^* is $2^{245.05-32} = 2^{213.05}$, so the time complexity of Step 2.(a) is $2^{32} \times 2^{213.05} = 2^{245.05}$. This is the dominant of the overall time complexity of our advanced MITM attack. The probability for a wrong key to pass Step 2.(e).iii is $2^{208} \times 2^{-8\times(32+16)} = 2^{-276}$. So the success probability is $1 - 2^{-276}$. The data complexity is still 2^{113}.

6 Conclusion

In this paper, we launch improved MITM attacks on Crypton-256 reduced to 9 and 10 rounds. By introducing the new techniques in [22,23], we successfully improve the existing MITM results on Cryton in both complexities and the number of attacked rounds. Our attacks can be applied to both the original Crypton in [1] and the revised version Crypton v1.0 in [2].

References

1. Lim, C.H.: Crypton: A new 128-bit block cipher. NIST AES Proposal (1998)
2. Lim, C.H.: A revised version of crypton: Crypton v1. 0. In: Fast Software Encryption, Springer (1999) 31–45
3. Daemen, J., Rijmen, V.: Aes proposal: Rijndael. (1998)
4. Daemen, J., Rijmen, V.: The design of Rijndael: AES-the advanced encryption standard. Springer Science & Business Media (2013)
5. DHalluin, C., Bijnens, G., Rijmen, V., Preneel, B.: Attack on six rounds of crypton. In: Fast Software Encryption, Springer (1999) 46–59
6. Cheon, J.H., Kim, M., Kim, K., Jung-Yeun, L., Kang, S.: Improved impossible differential cryptanalysis of rijndael and crypton. In: Information Security and CryptologyICISC 2001. Springer (2001) 39–49

7. Mala, H., Shakiba, M., Dakhilalian, M.: New impossible differential attacks on reduced-round crypton. Computer Standards & Interfaces **32**(4) (2010) 222–227
8. Lin, L., Wu, W., Wang, Y., Zhang, L.: General model of the single-key meet-in-the-middle distinguisher on the word-oriented block cipher. In Lee, H., Han, D., eds.: Information Security and Cryptology - ICISC 2013 - 16th International Conference, Seoul, Korea, November 27-29, 2013, Revised Selected Papers. Volume 8565 of Lecture Notes in Computer Science., Springer (2013) 203–223
9. Liu, C., Liao, F., Wei, H.: A meet-in-the-middle attack on reduced-round crypton. (2012)
10. Shakiba, M., Dakhilalian, M., Mala, H.: Non-isomorphic biclique cryptanalysis of full-round crypton. Computer Standards & Interfaces **41** (2015) 72–78
11. Wei, Y., Li, C., Sun, B.: Related-key impossible differential cryptanalysis on crypton and crypton v1. 0. In: Internet Security (WorldCIS), 2011 World Congress on, IEEE (2011) 227–232
12. Diffie, W.: Exhaustive crypianalysis of the nbs daia encrypiion siandard. (1977)
13. Dunkelman, O., Sekar, G., Preneel, B.: Improved meet-in-the-middle attacks on reduced-round des. In: Progress in Cryptology–INDOCRYPT 2007. Springer (2007) 86–100
14. Lu, J., Wei, Y., Pasalic, E., Fouque, P.A.: Meet-in-the-middle attack on reduced versions of the camellia block cipher. In: Advances in Information and Computer Security. Springer (2012) 197–215
15. Chen, J., Li, L.: Low data complexity attack on reduced camellia-256. In: Information Security and Privacy, Springer (2012) 101–114
16. Sekar, G., Mouha, N., Velichkov, V., Preneel, B.: Meet-in-the-middle attacks on reduced-round xtea. In: Topics in Cryptology–CT-RSA 2011. Springer (2011) 250–267
17. Demirci, H., Selçuk, A.A.: A meet-in-the-middle attack on 8-round aes. In: Fast Software Encryption, Springer (2008) 116–126
18. Demirci, H., Taşkın, İ., Çoban, M., Baysal, A.: Improved meet-in-the-middle attacks on aes. In: Progress in Cryptology-INDOCRYPT 2009. Springer (2009) 144–156
19. Wei, Y., Lu, J., Hu, Y.: Meet-in-the-middle attack on 8 rounds of the aes block cipher under 192 key bits. In: Information Security Practice and Experience. Springer (2011) 222–232
20. Dunkelman, O., Keller, N., Shamir, A.: Improved single-key attacks on 8-round aes-192 and aes-256. In: Advances in Cryptology-ASIACRYPT 2010. Springer (2010) 158–176
21. Derbez, P., Fouque, P.A., Jean, J.: Improved key recovery attacks on reduced-round aes in the single-key setting. In: Advances in Cryptology–EUROCRYPT 2013. Springer (2013) 371–387
22. Li, L., Jia, K., Wang, X.: Improved single-key attacks on 9-round aes-192/256. In: Fast Software Encryption, Springer (2014) 127–146
23. Li, R., Jin, C.: Meet-in-the-middle attacks on 10-round aes-256. Designs, Codes and Cryptography (2015) 1–13

Appendix

A The Conforming $\Delta y_6[0] \| \Delta y_6[1]$

There are 256 $\Delta y[0] \| \Delta y_1$ satisfying $\pi_0((\Delta y_6[0], \Delta y_6[1], 0, 0)^T) = (0, *, *, *)^T$. The conforming $\Delta y[0] \| \Delta y_1$ are as follows:
0‖0, 0‖1, 0‖2, 0‖3, 4‖4, 4‖5, 4‖6, 4‖7, 8‖8, 8‖9, 8‖a, 8‖b, c‖c, c‖d, c‖e, c‖f, 10‖10, 10‖11, 10‖12, 10‖13, 14‖14, 14‖15, 14‖16, 14‖17, 18‖18, 18‖19, 18‖1a, 18‖1b, 1c‖1c, 1c‖1d, 1c‖1e, 1c‖1f, 20‖20, 20‖21, 20‖22, 20‖23, 24‖24, 24‖25, 24‖26, 24‖27, 28‖28, 28‖29, 28‖2a, 28‖2b, 2c‖2c, 2c‖2d, 2c‖2e, 2c‖2f, 30‖30, 30‖31, 30‖32, 30‖33, 34‖34, 34‖35, 34‖36, 34‖37, 38‖38, 38‖39, 38‖3a, 38‖3b, 3c‖3c,

3c‖3d, 3c‖3e, 3c‖3f, 40‖0, 40‖1, 40‖2, 40‖3, 44‖4, 44‖5, 44‖6, 44‖7, 48‖8, 48‖9, 48‖a, 48‖b, 4c‖c, 4c‖d, 4c‖e, 4c‖f, 50‖10, 50‖11, 50‖12, 50‖13, 54‖14, 54‖15, 54‖16, 54‖17, 58‖18, 58‖19, 58‖1a, 58‖1b, 5c‖1c, 5c‖1d, 5c‖1e, 5c‖1f, 60‖20, 60‖21, 60‖22, 60‖23, 64‖24, 64‖25, 64‖26, 64‖27, 68‖28, 68‖29, 68‖2a, 68‖2b, 6c‖2c, 6c‖2d, 6c‖2e, 6c‖2f, 70‖30, 70‖31, 70‖32, 70‖33, 74‖34, 74‖35, 74‖36, 74‖37, 78‖38, 78‖39, 78‖3a, 78‖3b, 7c‖3c, 7c‖3d, 7c‖3e, 7c‖3f, 80‖0, 80‖1, 80‖2, 80‖3, 84‖4, 84‖5, 84‖6, 84‖7, 88‖8, 88‖9, 88‖a, 88‖b, 8c‖c, 8c‖d, 8c‖e, 8c‖f, 90‖10, 90‖11, 90‖12, 90‖13, 94‖14, 94‖15, 94‖16, 94‖17, 98‖18, 98‖19, 98‖1a, 98‖1b, 9c‖1c, 9c‖1d, 9c‖1e, 9c‖1f, a0‖20, a0‖21, a0‖22, a0‖23, a4‖24, a4‖25, a4‖26, a4‖27, a8‖28, a8‖29, a8‖2a, a8‖2b, ac‖2c, ac‖2d, ac‖2e, ac‖2f, b0‖30, b0‖31, b0‖32, b0‖33, b4‖34, b4‖35, b4‖36, b4‖37, b8‖38, b8‖39, b8‖3a, b8‖3b, bc‖3c, bc‖3d, bc‖3e, bc‖3f, c0‖0, c0‖1, c0‖2, c0‖3, c4‖4, c4‖5, c4‖6, c4‖7, c8‖8, c8‖9, c8‖a, c8‖b, cc‖c, cc‖d, cc‖e, cc‖f, d0‖10, d0‖11, d0‖12, d0‖13, d4‖14, d4‖15, d4‖16, d4‖17, d8‖18, d8‖19, d8‖1a, d8‖1b, dc‖1c, dc‖1d, dc‖1e, dc‖1f, e0‖20, e0‖21, e0‖22, e0‖23, e4‖24, e4‖25, e4‖26, e4‖27, e8‖28, e8‖29, e8‖2a, e8‖2b, ec‖2c, ec‖2d, ec‖2e, ec‖2f, f0‖30, f0‖31, f0‖32, f0‖33, f4‖34, f4‖35, f4‖36, f4‖37, f8‖38, f8‖39, f8‖3a, f8‖3b, fc‖3c, fc‖3d, fc‖3e, fc‖3f,