Cryptanalysis of an image encryption scheme*

Shujun Li[1], Chengqing Li[2], Kwok-Tung Lo[1] and Guanrong Chen[2]

[1] Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong SAR, China

[2] Department of Electronic Engineering, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon Tong, Hong Kong SAR, China

Abstract

We present cryptanalysis of an image encryption scheme, which is based on the base-switching (BS) lossless compression algorithm. The following conclusions are reached: 1. the size of the key space, i.e., the security against brute-force attacks, was greatly overestimated by the designers; and 2. the scheme is not secure against known/chosen-plaintext/ciphertext attacks. A real example is given to show the feasibility of the proposed chosen-plaintext attack. In addition, some other minor problems of the scheme are also pointed out. © 2006 SPIE and IS&T. [DOI: 10.1117/1.2360897]

1 Introduction

In last three decades, many encryption methods have been proposed to protect digital images and videos. They are useful for providing special security demands in real applications, such as multimedia message services, pay TV, video teleconferencing, medical imaging, military image databases, etc. Most schemes aim to realize sufficiently fast encryption algorithms with an acceptable level of security. For the state of the art of image and video encryption, the readers are referred to Refs. 1–3. Note that many image and video encryption schemes are not sufficiently secure and can be broken via some efficient attack methods. So, one should be very careful to find a suitable solution for a real application of image encryption.

Since most digital images are stored and transmitted in a compressed format, many image encryption schemes are designed by incorporating encryption into compression. To achieve a better balance between encryption and compression, some researchers developed new compression algorithms to benefit encryption. In Ref. 4, a joint compression-encryption scheme was proposed, based on a base-switching (BS) lossless compression algorithm that was proposed by the same authors in Ref. 5. This scheme works on 3×3 subimages. In the compression stage, a 7-bit base value is obtained to represent each subimage, and in the encryption stage the base value is substituted via a polynomial mapping function or bitwise XOR operations, which is controlled by the secret key. To further enhance the security, feedback from previous subimages is also used. It was claimed that the joint compression-encryption scheme had a much higher level of security than some other designs proposed in Refs. 6–9.

Though the joint compression-encryption scheme has been proposed for many years, no cryptanalysis work has been reported. In this work, we restudy the security of this scheme. We found that it is not as secure as claimed.[4] The following conclusions are reached: 1. its key space size was greatly overestimated by the designers; and 2. the scheme is not secure against known/chosen-plaintext/ciphertext attacks. Some minor problems and errors in Ref. 4 are also been pointed out.

This work is organized as follows. In the next section, a brief introduction to the joint compression-encryption scheme is given and some minor problems of the scheme are discussed. The cryptanalysis of the polynomial-based encryption scheme is given in Sec. 3, with a real example of the chosen-plaintext attack of the scheme.

*Shujun Li is the corresponding author, contact him via `http://www.hooklee.com`.

The cryptanalysis of the XOR-based encryption scheme is given in Sec. 4, and the last section concludes this study.

2 Joint Compression-Encryption Scheme

The scheme is a simple combination of the BS lossless compression algorithm and a substitution cipher with plaintext feedback. Both of the two stages work on 3×3 subimages. In the following, we separately introduce the two stages. For more details on the scheme, see Refs. 4, 5.

2.1 Compression Algorithm

The basic idea of the BS lossless compression algorithm is as follows. Given a 3×3 subimage represented by

$$g = \begin{bmatrix} g_0 & g_1 & g_2 \\ g_3 & g_4 & g_5 \\ g_6 & g_7 & g_8 \end{bmatrix}, \tag{1}$$

denote the minimal pixel value by $m = \min(g)$ and the base value by $b = \max(g) - \min(g) + 1$, and then represent the subimage as an 9-digit number with the radix b:

$$g' = (g'_8 \cdots g'_0)_b = \sum_{i=0}^{8} (g'_i \times b^i), \tag{2}$$

where $g'_i = g_i - m \in \{0, \cdots, b-1\}$. Then, one can represent the original 3×3 subimage as $\{b; m; g'\}$, which needs at most $16 + \lceil \log_2 b^9 \rceil$ bits to store if two bytes are used to represent b and m, respectively. Since for most subimages b is sufficiently small, $16 + \lceil \log_2 b^9 \rceil$ will be less than $8 \times 9 = 72$, which is the original bit size of g. This makes it possible to use a number of radix b to represent the original subimage, and it leads to a lossless compression of the image.

The previous idea does not work well when $b \geqslant 75$. In this case, $16 + \lceil \log_2 b^9 \rceil > 72$, so the bit size of the subimage is expanded, not compressed. To overcome this problem, the authors of Ref. 4 suggested three different rules to realize the compression algorithm, according to the value of b, as follows:

- Rule 1: when $1 \leqslant b \leqslant 11$, the encoding format is $\{b; m; g'\}$, where b is represented as a 7-bit integer.
- Rule 2: when $12 \leqslant b \leqslant 127$, the encoding format is $\{b; m; P(i_{\min}, i_{\max}); \hat{g}'\}$, where b is represented as a 7-bit integer, $P(i_{\min}, i_{\max})$ is a 7-bit integer used to denote the positions of the minimal and the maximal pixel-values[1], and $\hat{g}'$ is the 7-digit number with the radix b obtained by removing the minimal and the maximal pixel values.
- Rule 3: when $128 \leqslant b \leqslant 256$, the encoding format is $\{128; g\}$, where 128 is a "dummy" base value represented as a 7-bit integer $(0000000)_2$ (which is not used in the previous two rules)[2].

It is easy to calculate that at most $7 + 8 + \lceil \log_2 11^9 \rceil = 47$ bits are needed in rule 1, and at most $7 + 8 + 7 + \lceil \log_2 127^7 \rceil = 71$ bits are needed in rule 2. Though 79 bits are needed in rule 3, i.e., seven more bits are needed, generally an image can still be compressed effectively, since only a few percent of subimages satisfy $b \geqslant 128$ in most natural images.

2.2 Encryption Algorithm

After the previous compression stage, the 7-bit base value of each subimage is encrypted via one polynomial mapping over $\{1, \cdots, 128\}$ as follows[3]:

$$f(b) = \{[k_0 + k_1(b-1) + \cdots + k_n(b-1)^n] \bmod 128\} + 1, \tag{3}$$

where $K = \{k_0, \cdots, k_n\}$ serves as the secret key. To make a unique decryption of b possible, f must be a bijective mapping over $\{1, \cdots, 128\}$. Then, the decryption procedure can be represented by $b = f^{-1}[f(b)]$.

[1] Only $9 \times 8 = 72$ possible combinations of the positions, so $\lceil \log_2 72 \rceil = 7$ bits are enough to represent $P(i_{\min}, i_{\max})$.

[2] In Ref. 4, it is not explicitly mentioned that 128 should be represented as 0, but it is the only way to represent 128 with 7 bits.

[3] In Sec. 3.2 of Ref. 4, the degree is denoted by m, which conflicts with the notation used in the lossless compression part. In this work, we use n to replace m to avoid confusion.

To further enhance the security of the previous basic scheme, plaintext feedback was suggested in a t-layer scheme to encrypt the p'th subimage:

$$F(b_p) = \left\{ \left[\sum_{q=1}^{\min(p,t)} f_q(b_{p-q+1}) \right] \bmod 128 \right\} + 1, \tag{4}$$

where $f_1, \cdots, f_t$ are t polynomial mappings. In this enhanced scheme, the secret key K is composed of the secret parameters of the t polynomial mappings:

$$K_{f_1} = \{k_{f_1,0}, \cdots, k_{f_1,n_1}\}, \cdots, K_{f_t} = \{k_{f_t,0}, \cdots, k_{f_t,n_t}\},$$

where $n_1, \cdots, n_t$ are degrees of $f_1, \cdots, f_t$, respectively. To correctly decrypt the base value b, only f_1 needs to be a bijective mapping over $\{1, \cdots, 128\}$, and the decryption procedure becomes:

$$b_p = f_1^{-1} \left\{ \left[F(b_p) - 1 - \sum_{q=2}^{\min(p,t)} f_q(b_{p-q+1}) \right] \bmod 128 \right\}. \tag{5}$$

In Sec. 3.2 of Ref. 4, it was claimed that the key space of the basic scheme is 128!, since there are 128! bijective mappings over $\{1, \cdots, 128\}$. Similarly, in the enhanced scheme, the key space was claimed to be $(128!)^t$ for grayscaled images and $(128!)^{3t}$ for RGB color images. In Sec. 3.3 of Ref. 4, as an example, t was assumed to be the number of all subimages in a 512×512 image, and it was shown that the key space is $(128!)^{\lfloor 512/3 \rfloor \times \lfloor 512/3 \rfloor}$, which was claimed to be much larger than the key spaces of the image encryption schemes proposed[6–9] (see Table 1 of Ref. 4). However, in this paper, we demonstrate that the prior claims on the key space of the joint compression-encryption scheme are all wrong. We also point out that the scheme is not secure against known/chosen-plaintext attacks.

Besides the previous schemes based on polynomials modulo 128, the one-time pad is also suggested to realize the encryption function: $f(b) = [(b-1) \oplus k] + 1$, where $\oplus$ denotes bitwise XOR and k is a 7-bit key. In this case, the enhanced scheme should also be accordingly changed to realize the encryption and decryption functions. However, in Ref. 4 it was not mentioned how to do so. Without loss of generality, in this work we assume that the encryption function is modified as follows:

$$F(b_p) = \left\{ \bigoplus_{q=1}^{\min(p,t)} [f_q(b_{p-q+1}) - 1] \right\} + 1 = \left\{ \bigoplus_{q=1}^{\min(p,t)} \left[(b_{p-q+1} - 1) \oplus k_{f_q} \right] \right\} + 1. \tag{6}$$

2.3 Simpler Representation of Polynomial-Based Scheme

By replacing the base value b by $(b-1) \bmod 128$ and $f(b)$ by $[f(b)-1] \bmod 128$, the polynomial-based encryption scheme can be represented in a simpler form. Note that such a modification makes no any influence on the encryption/decryption procedures and the security of the encryption scheme. In this case, the set of all possible base values becomes $\{0, \cdots, 127\}$, and the encryption function in Eq. (3) becomes

$$f(b) = (k_0 + k_1 b + k_2 b^2 + \cdots + k_n b^n) \bmod 128. \tag{7}$$

The polynomial-based enhanced scheme can also be simplified in a similar way. With the prior representation, the polynomial-based encryption scheme is actually based on "permutation polynomials modulo 128" (see Chap. 4 of Ref. 10 and also Refs. 11–13)[4], which are defined as follows and have been used in some existing ciphers, including the well-known RSA public-key cipher.[16–23]

Definition 1 *An integer polynomial $f(x) = a_0 + a_1 x + \cdots + a_n x^n$ of degree n modulo m is called a permutation polynomial of degree n modulo m if it induces a bijection over $\{0, \cdots, m-1\}$, i.e., $\forall x_1 \not\equiv x_2 \pmod{m}$, $f(x_1) \not\equiv f(x_2) \pmod{m}$.*

The cryptanalysis results given in this work is mainly based on the mathematical results on permutation polynomials modulo m and another kind of integer polynomial, – null polynomials modulo m (see Definition 2).

Definition 2 *An integer polynomial $f(x) = a_0 + a_1 x + \cdots + a_n x^n$ of degree n modulo m is called a null polynomial of degree n modulo m if $\forall x \in \mathbb{Z}$, $f(x) \equiv 0 \pmod{m}$.*

[4] They are also called "substitution polynomials" or "polynomials representing all integers modulo m" in some early literature.[14, 15]

Due to the extreme complexity of the underlying mathematical problems, it is impossible to include all the mathematical results and the lengthy deduction in this cryptanalysis work. Instead, we give a complete discussion on permutation polynomials modulo m in Ref. 11 and a discussion on null polynomials modulo m in Ref. 24, respectively. Note that some fundamental results have been published,[13, 25–29] so the main focus of Refs. 11, 24 is a complete summary of all established results[5], with some newly derived corollaries that will be used in the cryptanalysis part (Sec. 3) of this work.

2.4 Remarks on Some Minor Problems of the Scheme

We close this section by pointing out a few minor problems of the scheme proposed in Ref. 4, leaving the major task of cryptanalysis to the rest of the work.

2.4.1 Two minor errors

One minor error is about the use of the congruence operation $\equiv$ in Ref. 4. Following the definition of congruence, $a \equiv b \pmod{m}$ means that $a - b$ is dividable by a number n, i.e., $(a-b)/n$ is an integer. However, in Secs. 3.2 and 3.3 of Ref. 4, $a \equiv b \pmod{128}$ is used to denote the fact that a equals to $(b \bmod 128)$, and $a \equiv b \pmod{128} + 1$ is used to denote the fact that a equals to $(b \bmod 128) + 1$. It is obvious that there are misuses in the sense of mathematics. The correct use of $\equiv$ in Secs. 3.2 and 3.3 of Ref. 4 should be $=$. This error has been corrected in Sec. 2.2 of this work.

There exists another error in Sec. 3.1 of Ref. 4. Throughout the section, the base-b number is represented as $b = (g'_0 g'_1 \cdots g'_8)_b$. However, following Eq. (6) of Ref. 4, $g' = \sum_{i=0}^{8} g'_i \times b^i$, which means $g' = (g'_8 \cdots g'_0)_b$. Fortunately, this error does not influence the algorithm at all. In Sec. 2.2 of this work, we have already unified the format of g' by adopting the second format: $g' = (g'_8 \cdots g'_0)_b = \sum_{i=0}^{8} g'_i \times b^i$.

2.4.2 Stream cipher or block cipher?

In Sec. 1 of Ref. 4, it was said that "our method is a kind of stream cipher." However, in our opinion, the joint compression-encryption scheme is more like a block cipher than a stream cipher.

At first, let us consider the basic scheme, in which each base value is encrypted in a fixed method independent of its position. However, in most cryptography literature, a stream cipher is defined as follows:[30]

> "In cryptography, a stream cipher is a symmetric cipher in which the plaintext digits are encrypted one at a time, and in which the transformation of successive digits varies during the encryption."

Another definition given in Ref. 31 says that a stream cipher encrypts texts with internal memory, while a block cipher is memoryless. As a typical feature, in a stream cipher, generally a long keystream is generated from a short secret key and then is used to encrypt the plaintext bit by bit. In the basic joint compression-encryption scheme, there does not exists such a keystream. So, we believe that the basic scheme should be a 7-bit block cipher, not a stream cipher. As shown later in the next section, the insecurity of the encryption scheme is partially caused by the extremely short block size.

The enhanced scheme is a little more complicated. The use of plaintext feedback makes the encryption dependent on the position of each subimage. However, a long keystream is still not involved in this scheme. The secret key is directly used to control $\min(p,t)$ transformations exerted on $\min(p,t)$ base values. We believe that the enhanced scheme is also more like a block cipher than a stream cipher.

2.4.3 Error-propagation problem

The use of plaintext feedback makes the scheme sensitive to errors in the ciphertext: if one error occurs in a position, all the following plain pixels will be influenced and cannot be correctly decrypted with a high probability. In addition, the encipher and the decipher will lose synchronization, if the bit size of one base value is wrongly decoded. This damages all decryption results after the synchronization loss occurs. It means that the encryption scheme can only be used in noise-free situations. By changing plaintext feedback to ciphertext feedback, this problem can be fixed.

[5] Some new proofs of known results have also been provided in Refs. 11, 24.

3 Cryptanalysis of Polynomial-Based Encryption Scheme

In this section, we analyze the security of the polynomial-based encryption scheme applied to gray-scaled images. The obtained results can be easily generalized to RGB color images by considering each RGB image as three independent grayscaled images.

3.1 Key Space

In the encryption scheme under study, the key is used to generate a bijective mapping through integer polynomials. So the size of the key space is equal to the number of all distinct bijective mappings[6] that can be generated from all distinct keys.

In Ref. 4, the key space of the basic scheme is simply estimated as the number of all distinct bijective mappings over $\{1, \cdots, 128\}$: 128!. However, this estimation is just the upper bound of the key space, due to the following facts:[11] 1. not all bijections can be induced by polynomials modulo 128; 2. not all polynomials can induce bijections over $\{1, \cdots, 128\}$; and 3. different polynomials may induce the same bijection over $\{1, \cdots, 128\}$, i.e., there exist equivalent polynomials modulo 128.

In the following, based on theoretical results on permutation polynomials obtained in Ref. 11, we give an exact estimation of the size of the key space, which is actually much smaller than the upper bound 128!. Without loss of generality, we adopt Eq. (7) as the encryption function here to simplify the discussion.

Before introducing the useful results in Ref. 11, we first give some preliminary definitions.

Definition 3 *Given two integer polynomials of degree $\leqslant n$: $f(x) = a_n x^n + \cdots + a_1 x + a_0$ and $g(x) = b_n x^n + \cdots + b_1 x + b_0$, if $\forall i = 0 \sim n$, $a_i \equiv b_i \pmod{m}$, we say $f(x)$ is congruent to $g(x)$ modulo m, or $f(x)$ and $g(x)$ are congruent (polynomials) modulo m, denoted by $f(x) \equiv g(x) \pmod{m}$.*

Definition 4 *A set of polynomials of degree n (or $\leqslant n$) modulo m is a complete system of polynomial residues of degree n modulo m, if for every polynomial of degree n (or $\leqslant n$) modulo m there is one and only one congruent polynomial in this set.*

Definition 5 *Denote $\omega_1(m)$ the least integer $n \geqslant 1$ such that there exists a monic null polynomial of degree n modulo m and call it the least monic null polynomial of degree n modulo m.*

Based on the previous definitions, let us introduce some notations. In each complete system of polynomial residues of degree $\leqslant n$ modulo p^d, denote the number of permutation polynomials of degree $\leqslant n$ modulo p^d by $N_{pp}(\leqslant n, p^d)$, and the number of all integer polynomials of degree $\leqslant n$ modulo p^d by $N_p(\leqslant n, p^d)$. Similarly, denote the number of null polynomials of degree $\leqslant n$ modulo p^d by $N_{np}(\leqslant n, p^d)$, and the number of all distinct permutations (bijective mapping) induced from all polynomials of degree $\leqslant n$ modulo m by $N_b(\leqslant n, p^d)$. One can see that $N_b(\leqslant n, p^d)$ actually denotes the size of the key space.

In Ref. 11, Sec. 5.3, the following results on permutation polynomials modulo a prime power p^d have been proved.

Theorem 1 *Assume p is a prime, $d \geqslant 2$, $n \geqslant 2p-1$. Then,* $\dfrac{N_{pp}(\leqslant n, p^d)}{N_p(\leqslant n, p^d)} = \dfrac{(p-1)^p (p-1)!}{p^{2p-1}}$.

Theorem 2 *Assume p is a prime and $d \geqslant 1$. Then,* $N_b(\leqslant n, p^d) = \dfrac{N_{pp}(\leqslant n, p^d)}{N_{np}(\leqslant n, p^d)}$.

Since two congruent polynomials modulo m induce the same mapping modulo m, $N_b(\leqslant n, 2^7)$ actually denotes the number of all possible encryption functions induced from polynomials of degree $\leqslant n$ modulo 128 shown in Eq. (7). Since there are $n+1$ coefficients in a polynomial of degree $\leqslant n$ modulo p^d, one can easily calculate that $N_p(\leqslant n, p^d) = (p^d)^{n+1} = p^{d(n+1)}$. So,

$$N_{pp}(\leqslant n, p^d) = N_p(\leqslant n, p^d) \cdot \frac{(p-1)^p (p-1)!}{p^{2p-1}} = p^{dn+(d-2p+1)} (p-1)^p (p-1)!.$$

When $p = 2$ and $d = 7 > 2p - 1 = 3$, i.e., $p^d = 128$, one has $N_{pp}(\leqslant n, 2^7) = 2^{7n+4}$.

Next, consider the value of n. We have the following lemma (see Sec. 2.6 of Ref. 11 or Sec. 2.5 of Ref. 24), which gives the upper bound of n modulo p^d.

[6] They can also be considered as a 128×128 S-boxes.

Lemma 1 *Every polynomial of degree $\geqslant \omega_1(m)$ modulo m has one equivalent polynomial of degree $\leqslant \omega_1(m)-1$ modulo m.*

The previous lemma implies that the maximal value of n is $\omega_1(m)-1$. For the encryption scheme under study, it means that all encryption functions induced from polynomials of degree $\geqslant \omega_1(128)$ can be induced from a polynomial of degree $\leqslant \omega_1(128)-1$. In other words, $\forall n \geqslant \omega_1(128)$, it is true that $N_b(\leqslant n, 2^7) = N_b(\leqslant \omega_1(128)-1, 2^7)$. As a result, we can assume that $n \leqslant \omega_1(128)-1$.

From Lemma 31 of Ref. 24, one has $\omega_1(128) = 2^3 = 8$ and a monic null polynomial of degree $\omega_1(128)$ modulo 128 can be derived from Lemma 39 of Ref. 24: $f(x) = \prod_{i=0}^{7}(x-i)$. Then, from Theorem 43 of Ref. 24, one has $N_{np}(\leqslant \omega_1(128)-1, 2^7) = 2^{\frac{2^3(2^2+2-2)}{2}} = 2^{16}$. Consequently, from Theorem 2, we can get the size of key space as follows:

$$N_b(\leqslant n, p^d) \leqslant N_b(\leqslant \omega_1(128)-1, p^d) = \frac{N_{pp}(\leqslant \omega_1(128)-1, 2^7)}{N_{np}(\leqslant \omega_1(128)-1, 2^7)} = \frac{2^{7(\omega_1(128)-1)+4}}{2^{16}} = 2^{37}. \tag{8}$$

That is, when $n = \omega_1(128)-1 = 7$ the key space of the basic scheme is only 2^{37}, which is much smaller than $128! \approx 2^{716}$. When $n < 7$, the key space will be even smaller than 2^{37}. As is well known, in today's digital computing speed, a key space of size 2^{37} is far from being secure.[32] It is generally recommended that the key space should not be smaller than $O(2^{100})$.

For the enhanced scheme, only f_1 should be a bijection over $\{1, \cdots, 128\}$ and other $t-1$ polynomial mappings can be arbitrary. Assume the number of distinct polynomial mappings of degree $\leqslant n$ modulo p^d is $N_{pm}(\leqslant n, p^d)$. In a similar way as above, one can derive that

$$N_{pm}(\leqslant n, 2^7) \leqslant N_{pm}(\leqslant \omega_1(128)-1, 2^7) = N_p(\leqslant 7, 2^7)/2^{16} = 2^{7(7+1)}/2^{16} = 2^{40}.$$

Thus, one can deduce that the key space of the enhanced scheme is not greater than $2^{37} \cdot 2^{40(t-1)} = 2^{40t-3}$, which is also much smaller than $(128!)^t \approx 2^{716t}$. Note that the key space will not be so large if the user only wants to break the first p $(< t)$ subimages. In this case, the key space will be $2^{40p-3} < 2^{40t-3}$. This means that the joint compression-encryption scheme has an increasing security distribution, not a uniform one, with respect to the position of the concerned sub-images. As is well known in cryptography, this is not a desirable property.[32] Regarding the non-uniform security of the enhanced scheme, it is meaningless to assign t as large as the number of all sub-images in order to enhance the security (as suggested in Sec. 3.3 of Ref. 4). For example, given a 512×512 image, assigning $t = \lfloor 512/3 \rfloor \times \lfloor 512/3 \rfloor$ can only provide $2^{40\lfloor t/2 \rfloor - 3}$ possible keys for the top half of the image which, however, may contains almost all useful information of the whole image in some cases.

From the prior analysis, one can see that the key space is dependent on the values of $n_1, \cdots, n_t \in \{1, \cdots, \omega_1(128)-1\}$ and $t \in \mathbb{Z}^+$. Using polynomials of higher degrees and higher t is the way to increase the key space. However, since at least $\sum_{i=1}^{t}(2n_i-1)$ multiplications[7] and $\sum_{i=1}^{t}(n_i+3)+(t+1)$ additions[8] are required for each subimage, there exists a tradeoff between the key space and the encryption speed. As a solution to this problem, one can store each polynomial mapping modulo 128 as a look-up table (LUT) with 128 input-output entries. Though these LUTs will occupy $256t$ bytes of additional memory, they are useful in dramatically increasing the encryption speed and relaxing the prior tradeoff. In this case, only t fast LUT operations are required for the encryption of each base value. However, even in this case the value of t should not be too large to achieve a sufficiently fast encryption speed. When the image is relatively large, it is generally impractical to assign t as large as the number of all subimages.

3.2 Chosen-Plaintext/Ciphertext Attack

3.2.1 Breaking the basic scheme

Breaking the look-up table with 128 chosen plain/cipher base values. Since the polynomial mapping f can be stored as a LUT in the encryption part, the LUT can be used as an equivalent of the secret key. By choosing the base values of 128 plain/cipher subimage as $1, \cdots, 128$, respectively, one can immediately recover all the 128 input-output entries of the LUT, which can then be used as an equivalent of the secret key for future encryption and decryption purposes. Apparently, this simple chosen-plaintext/ciphertext attack is essentially

[7] For each polynomial mapping f_i, in total $2n_i-1$ multiplications are needed: n_i-1 for calculating the powers of $b-1$ and n_i for calculating $k_j(b-1)^j$ $(j = 1 \sim n)$.

[8] For each polynomial mapping f_i, in total n_i+3 additions are needed: one addition for $b-1$, n_i for the sum of the $n+1$ addends $k_j(b-1)^j$ $(j = 1 \sim n)$, one for mod128, and one for +1 at the end of the right side of Eq. (3). Besides these additions, $t+1$ additions are still required in Eq. (4): $t-1$ additions for the sum of the t polynomial mappings, and another two for mod 128 and "+1", respectively.

due to the extremely short block size of the encryption scheme.[32] To maintain a practical security in today's digital world, the modulus 128 should be increased to be 2^n, where n is cryptographically large (generally $n \geqslant 128$ is recommended).

Breaking the encryption polynomial with 32 chosen plain base values. The previous simple chosen-plaintext/ciphertext attack needs to choose all the 128 plain/cipher base values. In fact, by choosing only 32 plain base values, one can derive an equivalent polynomial of $f(b)$ modulo 128 and further derive other $128 - 32 = 96$ unknown input-output entries of the LUT. In the following, we focus on this advanced chosen-plaintext attack. Note that this attack does not have a chosen-ciphertext counterpart[9].

In Sec. 5.4 of Ref. 11, an algorithm is given to determine all polynomials modulo p^d that induce a given mapping over $\{0, \cdots, p^d-1\}$, with less than p^d input-output entries of the mapping. Apparently, this algorithm can be concretized for the encryption scheme under study to realize the advanced chosen-plaintext attack. In the following, we use Eq. (7) as the encryption function for a simpler discussion. Basically, the algorithm can be divided into three parts: 1. decomposing the original polynomial into a subpolynomial tree and getting the lowest coefficients of all subpolynomials with 32 chosen plain base values; 2. determining all the coefficients of the sub-polynomials; and 3. determining the coefficients of a polynomial equivalent to the original one modulo 128.

First, let us decompose the polynomial $f(b)$ modulo 2^7 as follows.

- *Step 1a: decomposing* $f(b)$ *modulo* 2^7. Assuming $b = 2x_1 + b_0$, where $b_0 = (b \bmod 2) \in \{0, 1\}$ and $x_1 = \lfloor b/2 \rfloor$, the polynomial $f(b)$ becomes 2 subpolynomials of degree $\leqslant 6$ modulo $2^7 = 128$, as follows:

$$f_{b_0}(x_1) \equiv \sum_{i=6}^{0} 2^i k_i^{(b_0)} x_1^i \pmod{2^7},$$

where $k_i^{(0)} = k_i$ and $k_i^{(1)} = \sum_{j=n}^{i} \binom{j}{i} k_j$. Note that the terms of x_1 of degree $\geqslant 7$ vanish modulo 2^7. Taking $x_1 \equiv 0 \pmod{2^6}$, one immediately has $k_0^{(b_0)} \equiv f(b_0) \pmod{2^7}$. Then, subtracting $k_0^{(b_0)}$ from both sides of the prior congruence, one has

$$f_{b_0}^*(x_1) \equiv \sum_{i=6}^{1} 2^{i-1} k_i^{(b_0)} x_1^i \equiv \frac{f(2x_1 + b_0) - f(b_0)}{2} \pmod{2^6}.$$

- *Step 1b: decomposing* $f_{b_0}^*(x_1)$ *modulo* 2^6. In each $f_{b_0}^*(x_1)$, assuming $x_1 = 2x_2 + b_1$, where $b_1 = (x_1 \bmod 2) \in \{0, 1\}$ and $x_2 = \lfloor x_1/2 \rfloor$, one has four subpolynomials of degree $\leqslant 3$ modulo 2^6:

$$f_{b_1,b_0}(x_2) \equiv \sum_{i=3}^{1} 2^{2i-1} k_i^{(b_1,b_0)} x_2^i + k_0^{(b_1,b_0)} \pmod{2^6},$$

where $k_0^{(0,b_0)} = 0$, $k_0^{(1,b_0)} = \sum_{j=6}^{1} 2^{j-1} k_j^{(b_0)}$, and when $1 \leqslant i \leqslant 3$, $k_i^{(0,b_0)} = k_i^{(b_0)}$ and $k_i^{(1,b_0)} = \sum_{j=6}^{i} \binom{j}{i} 2^{j-i} k_j^{(b_0)}$. Note that the terms of x_2 of degree $\geqslant 4$ vanish modulo 2^6. Taking $x_2 \equiv 0 \pmod{2^5}$, one has $k_0^{(b_1,b_0)} \equiv f_{b_0}^*(b_1) \pmod{2^6}$. Then, subtracting $k_0^{(b_1,b_0)}$ from both sides, one has

$$f_{b_1,b_0}^*(x_2) \equiv \sum_{i=3}^{1} 2^{2(i-1)} k_i^{(b_1,b_0)} x_2^i \equiv \frac{f_{b_0}^*(2x_2 + b_1) - f_{b_0}^*(b_1)}{2} \pmod{2^5}.$$

- *Step 1c: decomposing* $f_{b_1,b_0}^*(x_2)$ *modulo* 2^5. In each $f_{b_1,b_0}^*(x_2)$, assuming $x_2 = 2x_3 + b_2$, where $b_2 = (x_2 \bmod 2) \in \{0, 1\}$ and $x_3 = \lfloor x_2/2 \rfloor$, one has eight subpolynomials of degree $\leqslant 2$ modulo 2^5:

$$f_{b_2,b_1,b_0}(x_3) \equiv \sum_{i=2}^{1} 2^{3i-2} k_i^{(b_2,b_1,b_0)} x_3^i + k_0^{(b_2,b_1,b_0)} \pmod{2^5},$$

where $k_0^{(0,b_1,b_0)} = 0$, $k_0^{(1,b_1,b_0)} = \sum_{j=3}^{1} 2^{2(j-1)} k_j^{(b_1,b_0)}$, and when $1 \leqslant i \leqslant 2$, $k_i^{(0,b_1,b_0)} = k_i^{(b_1,b_0)}$ and $k_i^{(1,b_1,b_0)} = \sum_{j=3}^{i} \binom{j}{i} 2^{2(j-i)} k_j^{(b_1,b_0)}$. Similarly, taking $x_3 \equiv 0 \pmod{2^4}$, one has $k_0^{(b_2,b_1,b_0)} \equiv f_{b_1,b_0}^*(b_2)$

[9] As shown next, the chosen-plaintext attack needs 32 special plain base values: $0, \cdots, 15, 16x_4, \cdots, 16x_4 + 15$, where $x_4 \not\equiv 0 \pmod{2}$. Such a special requirement cannot be ensured in the chosen-ciphertext attack without knowing the secrete mapping.

(mod 2^5). Then, subtracting $k_0^{(b_2,b_1,b_0)}$ from both sides, one has

$$f^*_{b_2,b_1,b_0}(x_3) \equiv \sum_{i=2}^{1} 2^{3(i-1)} k_i^{(b_2,b_1,b_0)} x_3^i \equiv \frac{f^*_{b_1,b_0}(2x_3+b_2) - f^*_{b_1,b_0}(b_2)}{2} \pmod{2^4}.$$

- *Step 1d: decomposing $f^*_{b_2,b_1,b_0}(x_3)$ modulo 2^4.* In each $f^*_{b_2,b_1,b_0}(x_3)$, assuming $x_3 = 2x_4 + b_3$, where $b_3 = (x_3 \bmod 2) \in \{0,1\}$ and $x_4 = \lfloor x_3/2 \rfloor$, one has 16 subpolynomials of degree $\leqslant 1$ modulo 2^4:

  $$f_{b_3,b_2,b_1,b_0}(x_4) \equiv 2k_1^{(b_3,b_2,b_1,b_0)} x_4 + k_0^{(b_3,b_2,b_1,b_0)} \pmod{2^4},$$

  where $k_0^{(0,b_2,b_1,b_0)} = 0$, $k_0^{(1,b_2,b_1,b_0)} = \sum_{j=2}^{1} 2^{3(j-1)} k_j^{(b_2,b_1,b_0)}$, $k_1^{(0,b_2,b_1,b_0)} = k_1^{(b_2,b_1,b_0)}$ and $k_1^{(1,b_2,b_1,b_0)} = \sum_{j=2}^{1} j 2^{3(j-1)} k_j^{(b_2,b_1,b_0)} \equiv k_1^{(b_2,b_1,b_0)} \pmod{2^4}$.

With the previous decomposition, one can derive the undetermined coefficients of these subpolynomials from bottom to top in the following steps.

- *Step 2a: determining $f_{b_3,b_2,b_1,b_0}(x_4)$ modulo 2^4.* Taking $x_4 \equiv 0 \pmod{2^3}$, one can solve that $k_0^{(b_3,b_2,b_1,b_0)} \equiv f^*_{b_2,b_1,b_0}(b_3) \pmod{2^4}$, and then choosing $x_4 \not\equiv 0 \pmod{2}$, one has

  $$\begin{aligned} 2k_1^{(b_3,b_2,b_1,b_0)} &\equiv \bar{x}_4 \left[f^*_{b_2,b_1,b_0}(2x_4+b_3) - k_0^{(b_3,b_2,b_1,b_0)} \right] \pmod{2^4} \\ &\equiv \bar{x}_4 \left[f^*_{b_2,b_1,b_0}(2x_4+b_3) - f^*_{b_2,b_1,b_0}(b_3) \right] \pmod{2^4}, \end{aligned}$$

  where $\bar{x}_4$ denotes an inverse of x_4 modulo 2^4. Thus, $f_{b_3,b_2,b_1,b_0}(x_4)$ is uniquely determined modulo 2^4. *Note that According to the definitions of all the involved sub-polynomials, with the 16 determined sub-polynomials $\{f_{b_3,b_2,b_1,b_0}(x_4)\}_{0\leqslant b_0,b_1,b_2,b_3\leqslant 1}$ modulo 2^4 and the following 14 determined values:*

  $$\left\{k_0^{(b_2,b_1,b_0)}\right\}_{0\leqslant b_0,b_1,b_2\leqslant 1} \text{ modulo } 2^5,\ \left\{k_0^{(b_1,b_0)}\right\}_{0\leqslant b_0,b_1\leqslant 1} \text{ modulo } 2^6 \text{ and } \left\{k_0^{(b_0)}\right\}_{0\leqslant b_0\leqslant 1} \text{ modulo } 2^7,$$

  one can uniquely determine the permutation polynomial $f(b)$ modulo 2^7. If the attacker only wants to reveal unknown input-output entries in the LUT, he can quit at this point.

- *Step 2b: determining $f^*_{b_2,b_1,b_0}(x_3)$ modulo 2^4 and $f_{b_2,b_1,b_0}(x_3)$ modulo 2^5.* From the relation between $\left\{k_i^{(b_2,b_1,b_0)}\right\}_{1\leqslant i\leqslant 2}$ and $\left\{k_i^{(b_3,b_2,b_1,b_0)}\right\}_{\substack{0\leqslant b_3\leqslant 1\\ 0\leqslant i\leqslant 1}}$, one has $k_1^{(b_2,b_1,b_0)} \equiv k_1^{(0,b_2,b_1,b_0)} \equiv k_1^{(1,b_2,b_1,b_0)} \pmod{2^4}$ and $2^3 k_2^{(b_2,b_1,b_0)} \equiv k_0^{(1,b_2,b_1,b_0)} - k_1^{(b_2,b_1,b_0)} \pmod{2^4}$.

  Considering that $2k_1^{(b_3,b_2,b_1,b_0)}$ has been uniquely determined modulo 2^4 in the above step and that $k_1^{(0,b_2,b_1,b_0)} \equiv k_1^{(1,b_2,b_1,b_0)} \pmod{2^4}$, we can get the following result:

  $$\forall b_0, b_1, b_2 \in \{0,1\}, \left\{k_1^{(0,b_2,b_1,b_0)}, k_1^{(1,b_2,b_1,b_0)}\right\}$$

  has two candidate values modulo 2^4. This means that $\left\{k_1^{(b_2,b_1,b_0)}, 2^3 k_2^{(b_2,b_1,b_0)}\right\}$ has two candidates modulo 2^4. So, $f^*_{b_2,b_1,b_0}(x_3)$ has two candidate polynomials modulo 2^4, i.e., $f_{b_2,b_1,b_0}(x_3)$ has two candidate polynomials modulo 2^5.

  Note that we calculate the value of $2^3 k_2^{(b_2,b_1,b_0)}$ modulo 2^4, instead of the value of $k_2^{(b_2,b_1,b_0)}$ modulo 2, to facilitate the following discussions (the same hereafter).

- *Step 2c: determining $f^*_{b_1,b_0}(x_2)$ modulo 2^5 and $f_{b_1,b_0}(x_2)$ modulo 2^6.* In a similar way, one can derive that $k_1^{(b_1,b_0)} \equiv k_1^{(0,b_1,b_0)} \pmod{2^5}$, $2^2 k_2^{(b_1,b_0)} \equiv 2^2 k_2^{(0,b_1,b_0)} \pmod{2^5}$ and

  $$\begin{aligned} 2^4 k_3^{(b_1,b_0)} &\equiv k_0^{(1,b_1,b_0)} - k_1^{(b_1,b_0)} - 2^2 k_2^{(b_1,b_0)} \equiv k_0^{(1,b_1,b_0)} - k_1^{(0,b_1,b_0)} - 2^2 k_2^{(0,b_1,b_0)} \pmod{2^5} \\ &\equiv k_1^{(1,b_1,b_0)} - k_1^{(b_1,b_0)} - 2^3 k_2^{(b_1,b_0)} \equiv k_1^{(1,b_1,b_0)} - k_1^{(0,b_1,b_0)} - 2^3 k_2^{(0,b_1,b_0)} \pmod{2^5} \\ &\equiv 2^2 \left[k_2^{(1,b_1,b_0)} - k_2^{(b_1,b_0)} \right] \equiv 2^2 k_2^{(1,b_1,b_0)} - 2^2 k_2^{(0,b_1,b_0)} \pmod{2^5}. \end{aligned}$$

In the previous congruences, rows 1 & 2 lead to the result that $2^2k_2^{(0,b_1,b_0)} \equiv k_1^{(1,b_1,b_0)} - k_0^{(1,b_1,b_0)} \pmod{2^5}$, and rows 1 & 3 lead to $2^2k_2^{(1,b_1,b_0)} \equiv k_0^{(1,b_1,b_0)} - k_1^{(0,b_1,b_0)} \pmod{2^5}$. Substituting the results into the congruences, one has

$$\begin{aligned} k_1^{(b_1,b_0)} &\equiv k_1^{(0,b_1,b_0)} \pmod{2^5}, \\ 2^2k_2^{(b_1,b_0)} &\equiv k_1^{(1,b_1,b_0)} - k_0^{(1,b_1,b_0)} \pmod{2^5}, \\ 2^4k_3^{(b_1,b_0)} &\equiv 2k_0^{(1,b_1,b_0)} - \left[k_1^{(1,b_1,b_0)} + k_1^{(0,b_1,b_0)}\right] \pmod{2^5}. \end{aligned}$$

That is, $\forall b_0, b_1 \in \{0,1\}$, $\left\{2^{2(i-1)}k_i^{(b_1,b_0)}\right\}_{1\leqslant i\leqslant 3}$ is determined by $\left\{k_0^{(1,b_1,b_0)}, k_1^{(b_2,b_1,b_0)}\right\}_{0\leqslant b_2\leqslant 1}$ uniquely modulo 2^5. Since $k_1^{(b_2,b_1,b_0)}$ has two candidate values modulo 2^4 and $2k_0^{(1,b_1,b_0)} - \left[k_1^{(1,b_1,b_0)} + k_1^{(0,b_1,b_0)}\right] \equiv 0 \pmod{2^4}$, there are eight candidates of $f^*_{b_1,b_0}(x_2)$ modulo 2^5, i.e., eight candidates of $f_{b_1,b_0}(x_2)$ modulo 2^6.

- *Step 2d: determining $f^*_{b_0}(x_1)$ modulo 2^6 and $f_{b_0}(x_1)$ modulo 2^7.* First, one has $k_1^{(b_0)} \equiv k_1^{(0,b_0)} \pmod{2^6}$, $2k_2^{(b_0)} \equiv 2k_2^{(0,b_0)} \pmod{2^6}$ and $2^2k_3^{(b_0)} \equiv 2^2k_3^{(0,b_0)} \pmod{2^6}$. Then, one has a system of congruences

$$\begin{bmatrix} 1 & 1 & 1 \\ 4 & 5 & 6 \\ \binom{4}{2} & \binom{5}{2} & \binom{6}{2} \\ \binom{4}{3} & \binom{5}{3} & \binom{6}{3} \end{bmatrix} \begin{bmatrix} 2^3k_4^{(b_0)} \\ 2^4k_5^{(b_0)} \\ 2^5k_6^{(b_0)} \end{bmatrix} \equiv \begin{bmatrix} k_0^{(1,b_0)} - \left[k_1^{(b_0)} + 2k_2^{(b_0)} + 2^2k_3^{(b_0)}\right] \\ k_1^{(1,b_0)} - \left[k_1^{(b_0)} + 2\cdot 2k_2^{(b_0)} + 3\cdot 2^2k_3^{(b_0)}\right] \\ 2k_2^{(1,b_0)} - \left[2k_2^{(b_0)} + 3\cdot 2^2k_3^{(b_0)}\right] \\ 2^2k_3^{(1,b_0)} - 2^2k_3^{(b_0)} \end{bmatrix} \pmod{2^6}. \tag{9}$$

Substituting $k_1^{(b_0)} \equiv k_1^{(0,b_0)} \pmod{2^6}$, $2k_2^{(b_0)} \equiv 2k_2^{(0,b_0)} \pmod{2^6}$ and $2^2k_3^{(b_0)} \equiv 2^2k_3^{(0,b_0)} \pmod{2^6}$ into the previous equation and then solve the sub-system formed by the first three congruences, one has

$$\begin{bmatrix} 2^3k_4^{(b_0)} \\ 2^4k_5^{(b_0)} \\ 2^5k_6^{(b_0)} \end{bmatrix} \equiv \begin{bmatrix} 15k_0^{(1,b_0)} - \left[5k_1^{(1,b_0)} + 10k_1^{(0,b_0)}\right] + \left[2k_2^{(1,b_0)} - 6\cdot 2k_2^{(0,b_0)}\right] - 3\cdot 2^2k_3^{(0,b_0)} \\ -24k_0^{(1,b_0)} + \left[9k_1^{(1,b_0)} + 15k_1^{(0,b_0)}\right] - \left[2\cdot 2k_2^{(1,b_0)} - 8\cdot 2k_2^{(0,b_0)}\right] + 3\cdot 2^2k_3^{(0,b_0)} \\ 10k_0^{(1,b_0)} - \left[4k_1^{(1,b_0)} + 6k_1^{(0,b_0)}\right] + \left[2k_2^{(1,b_0)} - 3\cdot 2k_2^{(0,b_0)}\right] - 2^2k_3^{(0,b_0)} \end{bmatrix} \pmod{2^6}. \tag{10}$$

The last congruence of Eq. (9) gives a constraint of the coefficients:

$$2^2k_3^{(1,b_0)} \equiv 20k_0^{(1,b_0)} - 10\left[k_1^{(1,b_0)} + k_1^{(0,b_0)}\right] + 4\left[2k_2^{(1,b_0)} - 2k_2^{(0,b_0)}\right] - 2^2k_3^{(0,b_0)} \pmod{2^6}. \tag{11}$$

Since $2^2k_3^{(1,b_0)}$ does not occur in Eq. (10), there always exists a unique solution for each candidate set of $\left\{k_0^{(1,b_0)}, k_1^{(b_1,b_0)}, 2k_2^{(b_1,b_0)}, 2^2k_3^{(0,b_0)}\right\}_{0\leqslant b_1\leqslant 1}$ modulo 2^6 when Eq. (11) holds. Assuming the number of all candidate polynomials of $f_{b_0}(x_1)$ obtained in this step is N, we show later that $N = 2^8 = 256$.

Finally, one can carry out the last step, *step 3*, to solve all equivalent polynomials of $f(b)$ modulo 2^7. Given a valid set of $\left\{k_i^{(b_0)}\right\}_{\substack{0\leqslant b_0\leqslant 1\\ 0\leqslant i\leqslant 6}}$ modulo 2^7, the coefficients of $f(b)$ can be uniquely solved modulo 2^7. Without loss of generality, assume the polynomial is of degree $\leqslant n = 2\cdot 7 - 1 = 13$ modulo 2^7, i.e., the number of unknown coefficients is 14. Then, one obtains the following system of congruences:

$$\begin{bmatrix} \boldsymbol{A}_0 \\ \boldsymbol{A}_1 \end{bmatrix} \begin{bmatrix} k_0 \\ \vdots \\ k_{13} \end{bmatrix} \equiv \begin{bmatrix} \boldsymbol{B}_0 \\ \boldsymbol{B}_1 \end{bmatrix} \pmod{2^7}, \tag{12}$$

where

$$\boldsymbol{A}_0 = \begin{bmatrix} \boldsymbol{I}_{7\times 7} & \boldsymbol{0}_{7\times 7} \end{bmatrix} = \left[\begin{array}{ccccc|ccccc} 1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 & 0 & \cdots & 0 \end{array}\right]_{7\times 14},$$

$$\boldsymbol{A}_1=\begin{bmatrix}\boldsymbol{A}_1^{(L)} & \boldsymbol{A}_1^{(R)}\end{bmatrix}=\left[\begin{array}{ccccc|cccc}1 & 1 & 1 & \cdots & 1 & 1 & 1 & \cdots & 1\\ 0 & 1 & \binom{2}{1} & \cdots & \binom{6}{1} & \binom{7}{1} & \binom{8}{1} & \cdots & \binom{13}{1}\\ 0 & 0 & 1 & \cdots & \binom{6}{2} & \binom{7}{2} & \binom{8}{2} & \cdots & \binom{13}{2}\\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & 0 & \cdots & 1 & \binom{7}{6} & \binom{8}{6} & \cdots & \binom{13}{6}\end{array}\right]_{7\times 14},$$

and $\boldsymbol{B}_{b_0}=\begin{bmatrix}k_0^{(b_0)} & k_1^{(b_0)} & \cdots & k_6^{(b_0)}\end{bmatrix}^T$ (for $b_0=0,1$). Calculating the determinant of the matrix on the left side (or, by Lemma 1 in Ref. 33), one immediately has

$$\begin{vmatrix}\boldsymbol{A}_0\\ \boldsymbol{A}_1\end{vmatrix}=|\boldsymbol{I}|\cdot\left|\boldsymbol{A}_1^{(R)}\right|=1.$$

Thus, $\{k_i\}_{0\leqslant i\leqslant 13}$ can be uniquely solved modulo 2^7, once $\boldsymbol{B}_0$ and $\boldsymbol{B}_1$ are both fixed modulo 2^7. Solving this system of congruences, one arrives at the following set of solutions:

$$\begin{bmatrix}k_0\\ \vdots\\ k_{13}\end{bmatrix}\equiv\begin{bmatrix}\boldsymbol{A}_0\\ \boldsymbol{A}_1\end{bmatrix}^{-1}\begin{bmatrix}\boldsymbol{B}_0\\ \boldsymbol{B}_1\end{bmatrix}\equiv\boldsymbol{A}^{-1}\begin{bmatrix}\boldsymbol{B}_0\\ \boldsymbol{B}_1\end{bmatrix}\pmod{2^7}, \tag{13}$$

where

$$\begin{aligned}\boldsymbol{A}^{-1} &= \begin{bmatrix}\boldsymbol{A}_0\\ \boldsymbol{A}_1\end{bmatrix}^{-1}\bmod 2^7\\ &= \left[\begin{array}{ccccccc|ccccccc}1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ \hline 76 & 100 & 50 & 46 & 44 & 100 & 121 & 52 & 104 & 74 & 8 & 36 & 120 & 1\\ 49 & 16 & 6 & 112 & 122 & 112 & 21 & 79 & 33 & 105 & 35 & 49 & 47 & 122\\ 76 & 101 & 42 & 76 & 12 & 46 & 93 & 52 & 103 & 84 & 79 & 112 & 13 & 15\\ 88 & 32 & 15 & 96 & 72 & 96 & 35 & 40 & 56 & 89 & 69 & 6 & 22 & 108\\ 4 & 4 & 118 & 89 & 100 & 116 & 107 & 124 & 0 & 14 & 1 & 72 & 18 & 15\\ 118 & 80 & 106 & 48 & 61 & 48 & 7 & 10 & 38 & 64 & 20 & 81 & 43 & 122\\ 100 & 50 & 46 & 44 & 100 & 121 & 127 & 28 & 50 & 82 & 44 & 28 & 121 & 1\end{array}\right].\end{aligned}$$

The previous equation shows that each valid set of $\left\{k_i^{(b_0)}\right\}_{\substack{0\leqslant b_0\leqslant 1\\ 0\leqslant i\leqslant 6}}$ corresponds to an equivalent polynomial of $f(b)$ of degree $\leqslant 13$ modulo 2^7. For each candidate polynomial $f_{b_0}(x_1)$ modulo 2^7, there are $2^{1+\cdots+6}=2^{21}$ sets of candidate values of $\left\{k_i^{(b_0)}\right\}_{0\leqslant i\leqslant 6}$ modulo 2^7, among which $k_i^{(b_0)}$ has 2^i candidate values. So, the number of equivalent polynomials of $f(b)$ modulo 2^7 is $(N2^{21})^2=N^2 2^{42}$. From a theorem in Ref. 24 (Theorem 3 next), $N^2 2^{42}$ is equal to the number of null polynomials of degree $\leqslant 13$ modulo 2^7, which is $2^{7(13-7)+\frac{2^3(2^2+2-2)}{2}}=2^{58}$ (see Sec. 4.5 of Ref. 24). Thus, one has $N^2 2^{42}=2^{58}\Rightarrow N=2^{\frac{58-42}{2}}=2^8=256$.

Theorem 3 *Two polynomials, $f_1(x)$ and $f_2(x)$, are equivalent polynomials modulo m if and only if $f_1(x)-f_2(x)$ is a null polynomial modulo m.*

Though the prior procedure can output all 2^{48} equivalent polynomials of $f(b)$ of degree $\leqslant 13$ modulo 2^7, the complexity of deriving all equivalent polynomials is relatively high. Actually, it is sufficient to randomly take one equivalent polynomial as a representative. To do so, one can choose the first candidate polynomial $f_{b_0}(x_1)$ and randomly select one valid set of $\left\{k_i^{(b_0)}\right\}_{\substack{0\leqslant b_0\leqslant 1\\ 0\leqslant i\leqslant 6}}$ in *step 2d*. In this way, the complexity becomes much lower (about hundreds of matrix operations modulo 2^i). If the degree of the obtained polynomial is greater than $\omega_1(128)-1=7$ modulo 2^7, one can further reduce it to be a polynomial of degree $\leqslant 7$ modulo 2^7. In addition, based on this representative polynomial, one can also list all equivalent polynomials, since all null polynomials of degree $\leqslant 13$ modulo 2^7 can be listed following the theoretical results in Ref. 24.

In the previous attack, some plain base values are needed in the decomposition procedure to derive the values of $\left\{k_0^{(b_0)}, k_0^{(b_1,b_0)}, k_0^{(b_2,b_1,b_0)}\right\}_{0\leqslant b_0,b_1,b_2\leqslant 1}$ and in *Step 2a* to derive the values of $\left\{k_i^{(b_3,b_2,b_1,b_0)}\right\}_{\substack{0\leqslant b_0,b_1,b_2,b_3\leqslant 1\\ 0\leqslant i\leqslant 1}}$. In the decomposition procedure, the base values should be chosen in $\{0,\cdots,7\}$, and in *Step 2a*, the base values should be chosen in $\{0,\cdots,15\}\cup\{16x_4,\cdots,16x_4+15\}$, where $x_4\not\equiv 0 \pmod 2$. In total one needs to choose 32 base values. Choosing $x_4=1$, the 32 chosen base values forms a set $\{0,\cdots,31\}$.

3.2.2 Breaking the enhanced scheme

For the enhanced scheme with t polynomial mappings, each polynomial modulo 128 can be broken one by one, by carrying out the basic attack on the subimages one by one.

Choose 32 plain images such that all base values of the i'th plain image are i. Then, the first secret polynomial, f_1, can be broken via the basic chosen-plaintext attack, by working on the first base value of each plain image. Next, apply induction on the index of the subimage, $j=2\sim t$. Since $f_1\sim f_{j-1}$ have been successfully broken, they can be removed from the encryption function of the j'th base value. That is, the encryption of the j'th base value is reduced to be the basic scheme by the secret polynomial f_i, so f_i can be broken in the same way via the basic attack. Apparently, the computational complexity of the inductive attack is t times of the complexity of the basic attack.

3.3 Known-Plaintext Attack

The known-plaintext attack can be considered as a weak case of the previous chosen-plaintext attack. Once the attacker observes 32 plain base values that satisfy the requirement of the advanced chosen-plaintext attack, he can immediately carry out the attack to break the encryption scheme. Similarly, if the attacker can observe 128 distinct plain/cipher base values, he can immediately carry out the basic chosen-plaintext attack to recover the LUT as an equivalent key. Since a typical image may contain thousands of subimages[10], one can easily collect enough base values with a high probability in real attacks.

3.4 Example of Chosen-Plaintext Attack

In the basic scheme, when the encryption function is $f(b)=(1+3b+2b^2) \bmod 128$, let us try to find at least one permutation polynomial equivalent to $f(b)$ modulo 128 via the prior chosen-plaintext attack by choosing 32 plain base values, $b=0,\cdots,31$, and the 32 corresponding ciphertexts: $\{f(b)\}_{0\leqslant b\leqslant 31}$. We describe the attack step by step as follows.

In *step 1a*, we have the input-output entries of $f(b)$, $f_0(x_1)$ and $f_1(x_1)$ shown in Table 1.

Table 1: The values of $\{f(b)\}_{0\leqslant b\leqslant 31}$, $\{f_0(x_1)\}_{0\leqslant x_1\leqslant 15}$ and $\{f_1(x_1)\}_{0\leqslant x_1\leqslant 15}$ modulo 2^7.

b	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$f(b)$ mod 2^7	1	6	15	28	45	66	91	120	25	62	103	20	69	122	51	112
b	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$f(b)$ mod 2^7	49	118	63	12	93	50	11	104	73	46	23	4	117	106	99	96
x_1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$f_0(x_1)$ mod 2^7	1	15	45	91	25	103	69	51	49	63	93	11	73	23	117	99
$f_1(x_1)$ mod 2^7	6	28	66	120	62	20	122	112	118	12	50	104	46	4	106	96

From these entries, we have $k_0^{(0)}\equiv f(0)\equiv 1 \pmod{2^7}$, $k_0^{(1)}\equiv f(1)\equiv 6 \pmod{2^7}$, and the entries of $\left\{f_{b_0}^*(x_1)\right\}_{0\leqslant b_0\leqslant 1}$ modulo 2^6 shown in Table 2.

Table 2: The values of $\{f_0^*(x_1)\}_{0\leqslant x_1\leqslant 15}$ and $\{f_1^*(x_1)\}_{0\leqslant x_1\leqslant 15}$ modulo 2^6.

x_1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$f_0^*(x_1)$ mod 2^6	0	7	22	45	12	51	34	25	24	31	46	5	36	11	58	49
$f_1^*(x_1)$ mod 2^6	0	11	30	57	28	7	58	53	56	3	22	49	20	63	50	45

[10] For example, a 512 × 512 image has 28,900 subimages, which is much larger than 32, the number of required base values.

In *step 1b*, in a similar way, we have $k_0^{(1,0)} \equiv f_0^*(1) \equiv 7 \pmod{2^6}$, $k_0^{(1,1)} \equiv f_1^*(1) \equiv 11 \pmod{2^6}$ and the input-output entries of $\left\{f_{b_1,b_0}^*(x_2)\right\}_{0\leqslant b_0,b_1\leqslant 1}$ modulo 2^5 shown in Table 3.

Table 3: The values of $\left\{f_{(b_1,b_0)}^*(x_2)\right\}_{\substack{0\leqslant x_2\leqslant 7\\ 0\leqslant b_0,b_1\leqslant 1}}$ modulo 2^5.

x_2	0	1	2	3	4	5	6	7
$f_{(0,0)}^*(x_2) \bmod 2^5$	0	11	6	17	12	23	18	29
$f_{(1,0)}^*(x_2) \bmod 2^5$	0	19	22	9	12	31	2	21
$f_{(0,1)}^*(x_2) \bmod 2^5$	0	15	14	29	28	11	10	25
$f_{(1,1)}^*(x_2) \bmod 2^5$	0	23	30	21	28	19	26	17

In *step 1c*, we have $k_0^{(1,0,0)} \equiv f_{(0,0)}^*(1) \equiv 11 \pmod{2^5}$, $k_0^{(1,1,0)} \equiv f_{(1,0)}^*(1) \equiv 19 \pmod{2^5}$, $k_0^{(1,0,1)} \equiv f_{(0,1)}^*(1) \equiv 15 \pmod{2^5}$, $k_0^{(1,1,1)} \equiv f_{(1,1)}^*(1) \equiv 23 \pmod{2^5}$ and the entries of $\left\{f_{b_2,b_1,b_0}^*(x_3)\right\}_{0\leqslant b_0,b_1,b_2\leqslant 1}$ modulo 2^4 shown in Table 4.

Table 4: The values of $\left\{f_{(b_2,b_1,b_0)}^*(x_3)\right\}_{\substack{0\leqslant x_3\leqslant 3\\ 0\leqslant b_0,b_1,b_2\leqslant 1}}$ modulo 2^4.

x_3	0	1	2	3	x_3	0	1	2	3
$f_{(0,0,0)}^*(x_3) \bmod 2^4$	0	3	6	9	$f_{(1,0,0)}^*(x_3) \bmod 2^4$	0	3	6	9
$f_{(0,1,0)}^*(x_3) \bmod 2^4$	0	11	6	1	$f_{(1,1,0)}^*(x_3) \bmod 2^4$	0	11	6	1
$f_{(0,0,1)}^*(x_3) \bmod 2^4$	0	7	14	5	$f_{(1,0,1)}^*(x_3) \bmod 2^4$	0	7	14	5
$f_{(0,1,1)}^*(x_3) \bmod 2^4$	0	15	14	13	$f_{(1,1,1)}^*(x_3) \bmod 2^4$	0	15	14	13

Since $f_{(0,b_1,b_0)}^*(x_3) \equiv f_{(1,b_1,b_0)}^*(x_3) \pmod{2^4}$, in *Step 1d* we only need to decompose $\left\{f_{(0,b_1,b_0)}^*(x_3)\right\}_{0\leqslant b_0,b_1\leqslant 1}$. We have the results in Table 5.

Table 5: The values of $\{f_{(b_3,b_2,b_1,b_0)}(x_4)\}_{\substack{0\leqslant x_4\leqslant 1\\ 0\leqslant b_0,b_1,b_2,b_3\leqslant 1}}$ modulo 2^4.

(b_3,b_2,b_1,b_0)	(0,0,0,0)	(1,0,0,0)	(0,0,1,0)	(1,0,1,0)	(0,0,0,1)	(1,0,0,1)	(0,0,1,1)	(1,0,1,1)
$f_{b_3,b_2,b_1,b_0}(0)$	0	3	0	11	0	7	0	15
$f_{b_3,b_2,b_1,b_0}(1)$	6	9	6	1	14	5	14	13

From Table 5, in *step 2a*, we can determine that $k_0^{(1,0,0,0)} \equiv 3 \pmod{2^4}$, $k_0^{(1,0,1,0)} \equiv 11 \pmod{2^4}$, $k_0^{(1,0,0,1)} \equiv 7 \pmod{2^4}$, $k_0^{(1,0,1,1)} \equiv 15 \pmod{2^4}$; $2k_1^{(0,0,0,0)} \equiv 2k_1^{(1,0,0,0)} \equiv 2k_1^{(0,0,1,0)} \equiv 2k_1^{(0,0,1,0)} \equiv 6 \pmod{2^4}$, $2k_1^{(0,0,0,1)} \equiv 2k_1^{(1,0,0,1)} \equiv 2k_1^{(0,0,1,1)} \equiv 2k_1^{(1,0,1,1)} \equiv 14 \pmod{2^4}$.

Then, in *step 2b*, from $k_1^{(b_2,b_1,b_0)} \equiv k_1^{(0,b_2,b_1,b_0)} \pmod{2^4}$ and $2^3k_2^{(b_2,b_1,b_0)} \equiv k_0^{(1,b_2,b_1,b_0)} - k_1^{(b_2,b_1,b_0)} \pmod{2^4}$, we have the results shown in Table 6.

Table 6: The values of $\left\{k_1^{(b_2,b_1,b_0)}\right\}_{0\leqslant b_0,b_1,b_2\leqslant 1}$ and $\left\{2^3k_2^{(b_2,b_1,b_0)}\right\}_{0\leqslant b_0,b_1,b_2\leqslant 1}$ modulo 2^4.

(b_1,b_0)	(0,0)		(1,0)		(0,1)		(1,1)	
$k_1^{(0,b_1,b_0)} \equiv k_1^{(1,b_1,b_0)} \pmod{2^4}$	3	11	3	11	7	15	7	15
$2^3k_2^{(0,b_1,b_0)} \equiv 2^3k_2^{(1,b_1,b_0)} \pmod{2^4}$	0	8	8	0	0	8	8	0

Next, from the results in *step 2c* discussed in Sec. 3.2.1, we can get candidate values of $\left\{2^{2(i-1)}k_i^{(b_1,b_0)}\right\}_{\substack{0\leqslant b_0,b_1\leqslant 1\\ 1\leqslant i\leqslant 3}}$ modulo 2^5 as shown in Table 7, where the first (second) row of $2^2k_2^{(b_1,b_0)}$ corresponds to the first (second) row of $2^4k_3^{(b_1,b_0)}$. In Table 7, note that $2^4k_3^{(b_1,b_0)}$ is uniquely determined by $k_1^{(b_1,b_0)}$ and $2^2k_2^{(b_1,b_0)}$ modulo 2^5.

Next, *step 2d* starts. In Eq. (9), the vector at the right side should be congruent to zero modulo 2^3. From the data shown in Table 7, we have $k_1^{(1,b_0)} - k_1^{(0,b_0)} \equiv 2\cdot 2k_2^{(0,b_0)} \equiv 0 \pmod{2^3}$ and from Eq. (9) we have $k_1^{(1,b_0)} - \left[k_1^{(0,b_0)} + 2\cdot 2k_2^{(0,b_0)} + 3\cdot 2^2k_3^{(0,b_0)}\right] \equiv 0 \pmod{2^3}$, then it is true that $2^2k_3^{(0,b_0)} \equiv 0 \pmod{2^3}$. Next,

Table 7: The candidate values of $\left\{2^{2(i-1)}k_i^{(b_1,b_0)}\right\}_{\substack{0\leqslant b_0,b_1\leqslant 1\\ 1\leqslant i\leqslant 3}}$ modulo 2^5.

(b_1, b_0)	(0,0)				(1,0)				(0,1)				(1,1)			
$k_1^{(b_1,b_0)} \pmod{2^5}$	$\boxed{3}$	11	$\boxed{19}$	27	3	$\boxed{11}$	19	$\boxed{27}$	$\boxed{7}$	15	$\boxed{23}$	31	7	$\boxed{15}$	23	$\boxed{31}$
$2^2k_2^{(b_1,b_0)} \pmod{2^5}$	24	0	8	16	16	24	0	8	24	0	8	16	16	24	0	8
	$\boxed{8}$	16	$\boxed{24}$	0	0	$\boxed{8}$	16	$\boxed{24}$	$\boxed{8}$	16	$\boxed{24}$	0	0	$\boxed{8}$	16	$\boxed{24}$
$2^4k_3^{(b_1,b_0)} \pmod{2^5}$	16	0	16	0	0	16	0	16	16	0	16	0	0	16	0	16
	$\boxed{0}$	16	$\boxed{0}$	16	16	$\boxed{0}$	16	$\boxed{0}$	$\boxed{0}$	16	$\boxed{0}$	16	16	$\boxed{0}$	16	$\boxed{0}$

$2^2k_3^{(1,b_0)} - 2^2k_3^{(0,b_0)} \equiv 0 \pmod{2^3}$ holds in Eq. (9), so we further have $2^2k_3^{(1,b_0)} \equiv 0 \pmod{2^3}$. In a similar way, we can also get $2k_2^{(0,b_0)} \equiv 4 \pmod{2^3}$ and $2k_2^{(1,b_0)} \equiv 4 \pmod{2^3}$. From these constraints, only the bold-faced values in Table 7 are valid.

Further, from the fact that the second row in Eq. (10) is congruent to 0 modulo 2^4, we have

$$24\cdot 7 - (9\cdot 11 + 15\cdot 3) + (8-0) - 3\cdot 2^2k_3^{(0,0)} \equiv 0 \pmod{2^4} \Rightarrow 2^2k_3^{(0,0)} \equiv 0 \pmod{2^4}.$$

In the same way, we have $2^2k_3^{(0,1)} \equiv 0 \pmod{2^4}$. In addition, note that the fourth row in Eq. (9) is congruent to 0 modulo 2^4; we immediately get $2^2k_3^{(1,0)} \equiv 2^2k_3^{(1,1)} \equiv 0 \pmod{2^4}$. That is, $2^2k_3^{(b_1,b_0)} \equiv 0 \pmod{2^4}$ holds for any b_0, b_1. Combining the previous results, Table 8 can be obtained, from which we can verify that

$$20k_0^{(1,b_0)} - 10\left[k_1^{(1,b_0)} + k_1^{(0,b_0)}\right] + 4\left[2k_2^{(1,b_0)} - 2k_2^{(0,b_0)}\right] - 2^2k_3^{(0,b_0)} \equiv 0 \pmod{2^4}$$

holds for any candidate values of $\left\{k_0^{(1,b_0)}, k_1^{(b_1,b_0)}, 2k_2^{(b_1,b_0)}, 2^2k_3^{(0,b_0)}\right\}_{0\leqslant b_1\leqslant 1}$. That is, there always exists $2^2k_3^{(1,b_0)} \equiv 0 \pmod{2^4}$, such that Eq. (11) holds. So, this constraint is canceled.

Table 8: The values of $\left\{k_1^{(b_1,b_0)}\right\}_{0\leqslant b_0,b_1\leqslant 1}$ modulo 2^5 and $\left\{2k_2^{(b_1,b_0)}, 2^2k_3^{(b_1,b_0)}\right\}_{0\leqslant b_0,b_1\leqslant 1}$ modulo 2^4.

(b_1, b_0)	(0,0)		(1,0)		(0,1)		(1,1)	
$k_1^{(b_1,b_0)} \pmod{2^5}$	3	19	11	27	7	23	15	31
$2k_2^{(b_1,b_0)} \pmod{2^4}$	4	12	4	12	4	12	4	12
$2^2k_3^{(b_1,b_0)} \pmod{2^4}$	0							

Now the only constraint of the coefficients is that

$$10k_0^{(1,b_0)} - \left[4k_1^{(1,b_0)} + 6k_1^{(0,b_0)}\right] + \left[2k_2^{(1,b_0)} - 3\cdot 2k_2^{(0,b_0)}\right] - 2^2k_3^{(0,b_0)} \equiv 2^5k_6^{(b_0)} \equiv 0 \pmod{2^5}.$$

By taking $b_0 = 0, 1$, we can easily verify that $10k_0^{(1,b_0)} - \left[4k_1^{(1,b_0)} + 6k_1^{(0,b_0)}\right] \equiv 0 \pmod{2^5}$, so

$$\left[2k_2^{(1,b_0)} - 3\cdot 2k_2^{(0,b_0)}\right] - 2^2k_3^{(0,b_0)} \equiv 0 \pmod{2^5} \Rightarrow 2k_2^{(1,b_0)} \equiv 3\cdot 2k_2^{(0,b_0)} + 2^2k_3^{(0,b_0)} \pmod{2^5}.$$

Considering the prior constraint and the relationship between the coefficients shown in Table 8, we can calculate the number of all candidate sets of $\left\{k_0^{(1,b_0)}, k_1^{(1,b_0)}, 2k_2^{(b_1,b_0)}, 2^2k_3^{(0,b_0)}\right\}_{0\leqslant b_1\leqslant 1}$ to be $(4\times 4\times 4)\times 2\times 2 = 2^8$. This agrees with the theoretical result given in Sec. 3.2.1.

So, we can freely choose one candidate set to get $\{f_{b_0}(x_1)\}_{0\leqslant b_0\leqslant 1}$. When $b_0 = 0$, choosing $k_1^{(0,0)} \equiv 3 \pmod{2^6}$, $k_1^{(1,0)} \equiv 11 \pmod{2^6}$, $2k_2^{(0,0)} \equiv 2k_2^{(1,0)} \equiv 4 \pmod{2^6}$, $2^2k_3^{(0,0)} \equiv 0 \pmod{2^6}$, we can get $k_1^{(0)} \equiv 3 \pmod{2^6}$, $2k_2^{(0)} \equiv 4 \pmod{2^6}$, $2^2k_3^{(0)} \equiv 0 \pmod{2^6}$ and $2^3k_4^{(0)} \equiv 2^4k_5^{(0)} \equiv 2^5k_6^{(0)} \equiv 0 \pmod{2^6}$. That is, $f_0^*(x_1) \equiv 3x_1 + 4x_1^2 \pmod{2^6}$, i.e., $f_0(x_1) \equiv 1 + 6x_1 + 8x_1^2 \pmod{2^7}$. Similarly, when $b_0 = 1$, choosing $k_1^{(0,1)} \equiv 7 \pmod{2^6}$, $k_1^{(1,1)} \equiv 15 \pmod{2^6}$, $2k_2^{(0,1)} \equiv 2k_2^{(1,1)} \equiv 4 \pmod{2^6}$, $2^2k_3^{(0,1)} \equiv 0 \pmod{2^6}$, we can get $k_1^{(1)} \equiv 7 \pmod{2^6}$, $2k_2^{(1)} \equiv 4 \pmod{2^6}$, $2^2k_3^{(1)} \equiv 0 \pmod{2^6}$ and $2^3k_4^{(1)} \equiv 2^4k_5^{(1)} \equiv 2^5k_6^{(1)} \equiv 0 \pmod{2^6}$. That is, $f_1^*(x_1) \equiv 7x_1 + 4x_1^2 \pmod{2^6}$, i.e., $f_1(x_1) \equiv 6 + 14x_1 + 8x_1^2 \pmod{2^7}$.

With the prior sub-polynomials $f_0(x_1) = 1 + 6x_1 + 8x_1^2$ and $f_1(x_1) = 6 + 14x_1 + 8x_1^2$ modulo 2^7, we can carry out the last step–*Step 3*. We can randomly choose valid values of $k_0^{(0)}, \cdots, k_6^{(0)}, k_0^{(1)}, \cdots, k_6^{(1)}$ and substitute them into Eq. (13) to get an equivalent polynomial of $f(b)$ modulo 2^7. Here, we choose the simplest set of these values modulo 2^7: $k_0^{(0)} = 1$, $k_1^{(0)} = 3$, $k_2^{(0)} = 2$, $k_3^{(0)} = k_4^{(0)} = k_5^{(0)} = k_6^{(0)} = 0$, $k_0^{(1)} = 6$, $k_1^{(1)} = 7$, $k_2^{(1)} = 2$ and $k_3^{(1)} = k_4^{(1)} = k_5^{(1)} = k_6^{(1)} = 0$. Then, solving Eq. (13), we immediately get $f(b) \equiv 1 + 3b + 2b^2 \pmod{2^7}$. One can see that we have successfully recovered the original polynomial modulo 2^7. This completes the chosen-plaintext attack.

If we choose other values of $k_0^{(0)}, \cdots, k_6^{(0)}, k_0^{(1)}, \cdots, k_6^{(1)}$, we may get equivalent polynomials different from the original one. For example, in the previous values, if we change $k_1^{(0)}$ from 3 to $64 + 3 = 67$, we get $f(b) \equiv 1 + 67b + 2b^2 + 64b^9 = (1 + 3b + 2b^2) + 64(b + b^9) \pmod{2^7}$. One can easily verify that this polynomial is really equivalent to $f(b) = 1 + 3b + 2b^2$ modulo 2^7, since $64(b + b^9) \equiv 0 \pmod{2^7}$ holds for any integer b. If the attacker wants to determine all equivalent polynomials of $f(b)$ modulo 128, he needs to enumerate all candidates of $f_0(x_1)$ and $f_1(x_1)$ modulo 2^6 in *step 2d* and all different values of the coefficients $k_0^{(0)}, \cdots, k_6^{(0)}, k_0^{(1)}, \cdots, k_6^{(1)}$ modulo 2^7 in *step 3*. However, in most cases, it is needless to do so.

4 Cryptanalysis of XOR-Based Encryption Scheme

4.1 Key Space

The key space of the XOR-based scheme is even smaller, compared with the polynomial-based one. For the basic scheme, the key is a 7-bit integer, so the size of the key space is only $2^7 = 128 \ll (128!)$. For the enhanced scheme, the key is t 7-bit integers and the key space size is $2^{7t} \ll (128!)^t$.

4.2 Known/Chosen-Plaintext/Ciphertext Attack

As is well-known in cryptography,[32] XOR-based ciphers are not secure against known/chosen-plaintext/ciphertext attacks at all. For the basic encryption scheme, given only one known (or chosen) plain/cipher base value, one can immediately derive that $k = [f(b) - 1] \oplus (b - 1)$. Similarly, in the enhanced scheme, the t subkeys can be derived as follows if the t leading base values of a plain image and the corresponding base values in the cipher image are all known (or chosen):

$$k_{f_p} = [F(b_p) - 1] \oplus (b_1 - 1) \oplus F^*(b_1, \cdots, b_{p-1}),$$

where $F^*(b_1, \cdots, b_{p-1}) = \bigoplus_{q=2}^{p} [f_q(b_{p-q+1}) - 1]$ when $2 \leqslant p \leqslant t$ and $F^*(b_1, \cdots, b_{p-1}) = 1$ when $p = 1$.

5 Conclusion

This work evaluates a joint compression-encryption scheme for digital images proposed in Ref. 4. It is found that the encryption scheme is very weak against known/chosen-plaintext/ciphertext attacks. It is also found that the key space was overestimated by the designers. The cryptanalysis study leads to a conclusion that the image encryption scheme proposed in Ref. 4 cannot be used in applications that require a high level of security.

Acknowledgments

This research was partially supported by The Hong Kong Polytechnic University's Postdoctoral Fellowships Program under grant number G-YX63.

References

1. A. Uhl and A. Pommer, *Image and Video Encryption: From Digital Rights Management to Secured Personal Communication*, Springer Science + Business Media, Inc., New York, USA (2005).

2. B. Furht, D. Socek, and A. M. Eskicioglu, "Fundamentals of multimedia encryption techniques," in *Multimedia Security Handbook*, edited by B. Furht and D. Kirovski, CRC Press, LLC, Boca Raton, Florida, USA, chapter 3, 93–132 (2004).

3. S. Li, G. Chen, and X. Zheng, "Chaos-based encryption for digital images and videos," in *Multimedia Security Handbook*, edited by B. Furht and D. Kirovski, CRC Press, LLC, Boca Raton, Florida, USA, chapter 4, 133–167, preprint available online at `http://www.hooklee.com/pub.html` (2004).

4. T.-J. Chuang and J.-C. Lin, "New approach to image encryption," *J. Electronic Imaging* **7**(2), 350–356 (1998).

5. T.-J. Chuang and J.-C. Lin, "A new algorithm for lossless still image compression," *Pattern Recognition* **31**(9), 1343–1352 (1998).

6. N. G. Bourbakis and C. Alexopoulos, "Picture data encryption using SCAN patterns," *Pattern Recognition* **25**(6), 567–581 (1992).

7. H. K.-C. Chang and J.-L. Liu, "An image encryption scheme based on quadtree compression scheme," in *Proc. Int. Computer Symposium (ICS'94)*, Hsinchu, Taiwan, China, 230–237 (1994).

8. C. Alexopoulos, N. G. Bourbakis, and N. Ioannou, "Image encryption method using a class of fractals," *J. Electronic Imaging* **4**(3), 251–259 (1995).

9. C. J. Kuo, "Novel image encryption technique and its application in progressive transmission," *J. Electronic Imaging* **2**(4), 345–351 (1993).

10. R. Lidl, G. L. Mullen, and G. Turnwald, *Dickson Polynomials*, Longman Scientific & Technical, Harlow, Essex, UK (1993).

11. S. Li, "Permutation polynomials modulo m," arXiv:math.NT/0509523, available online at `http://arxiv.org/abs/math/0509523` (2005).

12. Q. Sun and D. Wan, *Permutation Polynomials and Their Applications*, Liaoning Education Press, Shenyang, China, (in Chinese) (1987).

13. R. L. Rivest, "Permutation polynomials modulo 2^w," *Finite Fields and Their Applications* **7**(2), 287–292 (2001).

14. R. D. Carmichael, *Introduction to The Theory of Groups of Finite Order*, Dover Publications, Inc., New York, USA (1956).

15. L. E. Dickson, *History of the Theory of Numbers, volume III: Quadratic and Higher Forms*, AMS Chelsea Publishing, Providence, Rhode Island, USA (1992).

16. R. Lidl and W. B. Wüller, "Permutation polynomials in RSA-cryptosystems," in *Advances in Cryptology – Crypto'83*, edited by D. Chaum, Plenum Press, New York, USA, 293–301 (1983).

17. T. Matsumoto and H. Imai, "A class of assymetric crypto-systems based on polynomials over finite fields," in *Abstracts of Papers of IEEE International Symposium on Information Theory (ISIT'83)*, 131–132 (1983).

18. P. Delsarte, Y. Desmedt, A. M. Odlyzko, and P. Piret, "Fast cryptanalysis of the Matsumoto-Imai public key scheme," in *Advances in Cryptology – EuroCrypt'84*, edited by T. Beth, N. Cot, and I. Ingemarsson, Springer-Verlag, Berlin, Germany, *Lecture Notes in Computer Science*, volume 209, 142–149 (1985).

19. R. Lidl, "On cryptosystems based on permutation polynomials and finite fields," in *Advances in Cryptology – EuroCrypt'84*, edited by T. Beth, N. Cot, and I. Ingemarsson, Springer-Verlag, Berlin, Germany, *Lecture Notes in Computer Science*, volume 209, 10–15 (1985).

20. J. J. Cade, "A new public-key cipher which allows signatures," Presented at the Second SIAM Conference on Applied Linear Algebra, Haleigh, NC (April 30 - May 2, 1985).

21. N. S. James, R. Lidl, and H. Niederreiter, "Breaking the Cade cipher," in *Advances in Cryptology – Crypto'86*, edited by A. M. Odlyzko, Springer-Verlag, Berlin, Germany, *Lecture Notes in Computer Science*, volume 263, 60–63 (1987).

22. J. J. Cade, "A modification of a broken public-key cipher," in *Advances in Cryptology – Crypto'86*, edited by A. M. Odlyzko, Springer-Verlag, Berlin, Germany, *Lecture Notes in Computer Science*, volume 263, 64–83 (1987).

23. R. L. Rivest, M. J. B. Robshaw, R. Sidney, and Y. L. Yin, "The RC6 block cipher," available online at http://theory.lcs.mit.edu/~rivest/rc6.pdf (1998).

24. S. Li, "Null polynomials modulo m," arXiv:math.NT/0510217, available online at http://arxiv.org/abs/math/0510217 (2005).

25. G. Mullen and H. Stevens, "Permutation functions $\pmod{m}$," *Acta Mathematica Hungarica* **44**(3-4), 237–241 (1984).

26. J. Sun and O. Y. Takeshita, "Interleavers for turbo codes using permutation polynomials over integer rings," *IEEE Trans. Information Theory* **51**(1), 101–119 (2005).

27. G. Keller and F. R. Olson, "Counting permutation functions $\pmod{p^n}$," *Duke Mathematical Journal* **35**(4), 835–838 (1968).

28. A. J. Kempner, "Polynomials and their residue systems," *Transactions of the American Mathematical Society* **22**(2), 240–266 (1921).

29. A. J. Kempner, "Polynomials and their residue systems," *Transactions of the American Mathematical Society* **22**(3), 267–288 (1921).

30. Wikipedia, "Stream cipher," http://en.wikipedia.org/wiki/Stream_cipher (2005).

31. T. W. Cusick, C. Ding, and A. Renvall, *Stream Ciphers and Number Theory*, Elsevier B.V., Amsterdam, The Netherlands, revised edition (2004).

32. B. Schneier, *Applied Cryptography – Protocols, Algorithms, and Souce Code in C*, John Wiley & Sons, Inc., New York, USA, 2nd edition (1996).

33. S. Li, "Evaluating two determinants," arXiv:math.NT/0509350, available online at http://arxiv.org/abs/math/0509350 (2005).