Information Sciences 520 (2020) 130–141

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Cryptanalysis of a DNA-based image encryption scheme

Junxin Chen[a,b], Lei Chen[c], Yicong Zhou[b,*]

[a] College of Medicine and Biological Information Engineering, Northeastern University, Shenyang 110169, China
[b] Department of Computer and Information Science, University of Macau, Macau 999078, China
[c] School of Sciences, Beijing University of Posts and Telecommunications, Beijing 100876, China

ARTICLE INFO

Article history:
Received 6 January 2019
Revised 22 September 2019
Accepted 8 February 2020
Available online 10 February 2020

Keywords:
Cryptanalysis
Image encryption
Hénon-Sine map
DNA Coding

ABSTRACT

Recently, an image encryption scheme using 2D Hénon-Sine map and DNA coding approach is proposed. It adopts the permutation-diffusion architecture. The DNA random coding and exclusive OR is introduced for image diffusion, while image scrambling is implemented by pixel swapping operations. This paper reveals that this encryption scheme is not as secure as declared. Substitution boxes (s-boxes) are firstly employed to summarize the involved complicated DNA encryption operations, the whole encryption scheme is subsequently relaxed as an s-box-then-permutation cipher. Chosen-plaintext attack is feasible for recovering the equivalent secret elements of the s-boxes and permutation vector. The proposed concept of generalizing DNA encryption as s-box substitution is expected to be beneficial for security evaluation and theoretical design of DNA-based image encryption schemes in the future.

1. Introduction

Multimedia sharing and exchanging across public networks is becoming an important issue of our daily life. Together with the ever-growing appreciations of security and privacy, how to protect multimedia data against eavesdropping and abusing has drawn world-wide concerns. Encryption is the most straightforward protection way. In literature, traditional block ciphers has been reported un-suitable for multimedia data [2]. On the other hand, it is found that the properties of chaotic systems could be exploited to achieve satisfied permutation and diffusion performance for image encryption. Benefiting from the groundbreaking contributions of Fridrich [8,29], recent years have witnessed a booming of chaos-based image encryption schemes. They are mostly under the permutation-diffusion architecture. Their innovations are achieved in terms of pixel shuffling techniques [7,9], image diffusion means [3,24,30,32], encryption-compression concepts [14,19,45,47] as well as novel key stream generation mechanisms [16,17,23,37,46]. Besides, incorporation of optical transforms for image encryption also draw much concerns [12,26–28]. They can take advantages of high speed and parallel processing of optical transforms, and also benefit from the cryptographic features of chaotic systems. Some basic cryptographic requirements of chaotic ciphers have been standardized in [2].

Originating from the intrinsic advantageous such as inherent parallelism, low-power consumption and huge storage, DNA computing also emerges as a hot branch in cryptography. In such cryptosystems, DNA bases serve as the information carriers while biological regulations are employed as arithmetic principles. Ciphers combining DNA computing with chaos theory therefore becomes a matter of course, as they share mutually beneficial advantageous for designing cryptographic algo-

[*] Corresponding author at: Department of Computer and Information Science, University of Macau, Macau 999078, China.
E-mail address: yicongzhou@um.edu.mo (Y. Zhou).

https://doi.org/10.1016/j.ins.2020.02.024
0020-0255/© 2020 Elsevier Inc. All rights reserved.

rithms. The permutation-diffusion structure is also widely-adopted in such ciphers. The algorithm in [42] is a typical case. The plaintext is firstly transformed to DNA bases which are subsequently scrambled with the help of a hyper-chaotic system. Then DNA exclusive OR (XOR) is followed to mix the shuffled DNA bases with a secret matrix, finally a DNA decoding procedure is employed for the translation to pixel format. Such a design has been extended for color image encryption in [36], where DNA addition, subtraction and XOR operations are combined for pixel value modification. It is proposed in [35] to use joint pixel-level and DNA-level encryption for security promotion. Specifically, plain pixels are firstly transformed as DNA bases which are subsequently scrambled and DNA XORed for confusion and diffusion in DNA level. However, this ciphertext is encrypted again by XOR and modulo addition in pixel level. Random coding technique which means the encoding and decoding rules are distinct from each pixel is adopted in [6,10,11]. In this scenario, the secret rules can significantly promote the security level as coding with fixed rule is vulnerable against brute force attack [44]. Furthermore, researchers of [15] proposed to use DNA insertion and deletion to confuse the DNA bases. Hash function is nowadays introduced to DNA-based image ciphers frequently [4,5]. It is usually employed to generate a plaintext-related perturbation for updating the secret key. The encryption elements are therefore partly dependent on the plaintext itself, and further make the plaintext attack infeasible. In [34], *Wu* proposed a novel image encryption scheme based on the permutation-diffusion network. A prepositive diffusion procedure using DNA arithmetic is firstly conducted, then a permutation phase is implemented by pixel swapping operations. For randomness of the encryption elements, a two-dimensional Hénon-Sine map (2D-HSM) is developed and used for key stream generation. The widely-adopted security analysis suite has been conducted for security declaration.

Cryptanalysis is another important branch in cryptography. Of course, it is also of great value for security evaluation of the studied cipher [34]. Some DNA-based image encryption schemes have been found vulnerable against various attacks [1,22,43,44]. For example, it was found [43] that the DNA-based image cipher in [42] was insecure against chosen-plaintext attack. Several plaintext-ciphertext pairs are sufficient to retrieve the equivalent encryption masks, and then the received ciphertexts can be recovered directly. However, most of the cryptanalysis achievements [1,22,43] were proposed for a specified encryption scheme. They cannot be used for breaking other DNA-based image ciphers. In other words, the cryptanalysis approaches and attack methods proposed in [1,22,43], including the generalized analysis in [44], is not applicable for cracking this cipher [34].

In this paper, the security of *Wu*'s cipher [34] is re-evaluated. We firstly generalize the complicated DNA encryption operations (random encoding, XOR and random decoding) as substitution-boxes (s-boxes). The whole encryption scheme is subsequently relaxed as an s-box-then-permutation cipher. Furthermore, we reveal that the equivalent encryption elements can be retrieved by chosen-plaintext attack. Both of the theoretical cryptanalysis and attack procedures are given, and the feasibility has been experimentally validated. This work is different from and can be considered as extension of a related cryptanalysis [44], where *Zhang* focused on encryption algorithms that adopt secret yet fixed DNA coding rule for all pixels. Therefore, brute force attack can be introduced for breaking such DNA encryption patterns. We move one step further. In the studied cipher, the DNA encoding/decoding role is random for each particle which makes brute force attack infeasible. We firstly employ s-box to generalize the complex DNA encryption procedures, and this idea is expected to be beneficial for the future design and security evaluation of involved ciphers.

The primary contributions of this paper can be categorized as follows.

1) S-box is firstly employed to synthesize cryptographic effects of the complicated DNA encryption operations.
2) Security evaluation of an image encryption scheme using 2D Hénon-Sine map and DNA coding is conducted.
3) A chosen-plaintext attack which can retrieve equivalent encryption elements is proposed and verified.
4) Discussions regarding image ciphers incorporating chaos and DNA coding techniques are given.

The remainder of this paper is organized as follows. The encryption scheme under study will be sketched in the next section, while its cryptanalysis and chosen-plaintext attack are detailed in Section 3. The experimental validation and relative discussions are demonstrated in Section 4, and Section 5 concludes the paper finally.

2. The image cipher under study

We briefly review the studied encryption scheme here, interested readers can refer to Wu et al. [34] for more details.

2.1. Notations

Otherwise indicated, bold uppercase is used to denote an assemble which may be an image or a matrix. It generally refers to the secret elements such as permutation vector or diffusion masks. On the other hand, lowercase is adopted for representing a variable or element of corresponding assemble (in bold uppercase). The uppercase always denotes a constant. For example, a plain image $\boldsymbol{P}$ is with size $M \times N$ and thus has $L = M \times N$ pixels. It can be represented as $\boldsymbol{P} = \{p(1,1), p(1,2), \ldots, p(i,j), \ldots, p(M,N)\}$, or $\boldsymbol{P} = \{p(1), p(2), \ldots, p(i), \ldots, p(L)\}$.

2.2. Preliminaries

A new chaotic map, i.e., 2D-HSM, is developed for key stream generation in [34]. The mathematical formula of 2D-HSM is illustrated in Eq. (1), where $a \in (-\infty, +\infty)$ and $b \in (-\infty, +\infty)$ being control parameters. With the initial status $[x(0), y(0)]$

Table 1
Eight kinds of DNA coding rules.

Rule	A	T	C	G
Rule 1	00	11	10	01
Rule 2	00	11	01	10
Rule 3	11	00	10	01
Rule 4	11	00	01	10
Rule 5	10	01	00	11
Rule 6	01	10	00	11
Rule 7	10	01	11	00
Rule 8	01	10	11	00

Table 2
XOR operation of two DNA bases.

XOR	A	T	C	G
A	A	T	C	G
T	T	A	G	C
C	C	G	A	T
G	G	C	T	A

and control parameters [*a, b*], two serials of chaotic variables can be iteratively produced.

$$\begin{cases} x(n+1) = (1 - a \cdot \sin^2(x(n)) + y(n)) \bmod 1 \\ y(n+1) = (b \cdot x(n)) \bmod 1 \end{cases} \tag{1}$$

The DNA coding technique is employed for concealing pixels' gray value. A DNA sequence may have four kinds of nucleic acid bases, the so-called A (adenine), G (guanine), T (thymine), C (cytosine), respectively. Referring to Watson-Crick base pairing rules, A always pairs with T, and so are C and G. Considering complementary feature of the binary stream, i.e., 00 comply with 11 while 01 are complementary with 10, there are a total of 8 valid coding rules between binary stream and nucleic acid bases [44]. The coding rules are listed in Table 1. Besides, XOR in DNA format is also employed in [34], its implementation rule is given in Table 2.

2.3. The encryption procedures

To simplify the following cryptanalysis, $DNA\_p = DNA\_enc(p, rule)$ is defined as the DNA encoding function. It converts a 256-gray scale pixel p into a DNA base string DNA_p (including four DNA bases[1]), according to rule *rule*. On the other hand, $p = DNA\_dec(DNA\_p, rule)$ translates a DNA base string DNA_p into a 256-gray scale value p. The symbol $\oplus$ is employed to denote DNA XOR, as listed in Table 2. Without loss of generality, it also represents bit-wise XOR of two variables in binary format. Two combinations of control parameters and initial values of 2D-HSM, i.e., [a_1, b_1, $x_1(0)$, $y_1(0)$, a_2, b_2, $x_2(0)$, $y_2(0)$], jointly consist of the secret key of the encryption scheme. Assuming that the plaintext $\boldsymbol{P}$ is with size $M \times N$, the encryption procedures can be described as follows[2]

1) *Initialization.*
   a) Stretching the plain image into a one-dimension vector $\boldsymbol{P} = \{p(1), p(2), \ldots, p(L)\}$, where $L = M \times N$ denotes pixel counts (image length) of the plaintext.
   b) Producing two sets of chaotic state variables $\boldsymbol{X}_1 = \{x_1(1), x_1(2), \ldots, x_1(L)\}$ and $\boldsymbol{Y}_1 = \{y_1(1), y_1(2), \ldots, y_1(L)\}$, $\boldsymbol{X}_2 = \{x_2(1), x_2(2), \ldots, x_2(L)\}$ and $\boldsymbol{Y}_2 = \{y_2(1), y_2(2), \ldots, y_2(L)\}$. They are generated by iterating 2D-HSM with the input keys [a_1, b_1, $x_1(0)$, $y_1(0)$] and [a_2, b_2, $x_2(0)$, $y_2(0)$], respectively.
2) *Diffusion.*
   a) Calculating the modulo sum of the plain pixels, that is $temp = \sum_{i=1}^{L} p(i) \bmod 256$.

[1] 256 gray scales images are adopted in [34] and this paper, its extension to color images or high-resolution plaintexts are straightforward.

[2] The given encryption processes are slightly different from those in the original paper, yet the encryption cores are identical.

b) Sequentially masking the plain pixels $p(i)$ by Eq. (2).

$$\begin{cases} R_x = round(x_1(i) \times 10^{10}) \bmod 8 + 1 \\ R_y = round(y_1(i) \times 10^{10}) \bmod 8 + 1 \\ R_z = round(x_2(i) \times 10^{10}) \bmod 8 + 1 \\ R = round(y_2(i) \times 10^{10}) \bmod 256 \\ DNA_R = DNA\_enc(R, R_z) \\ DNA_1 = DNA\_enc(p(i), R_y) \\ DNA_2 = DNA_1 \oplus DNA_R \\ t(i) = DNA\_dec(DNA_2, R_x) \\ d(i) = t(i) \oplus temp \\ temp = d(i) \end{cases} . \tag{2}$$

c) After traversing all plain pixels, the diffusion image $\boldsymbol{D}$ is produced.

3) *Permutation*.

a) Generating the permutation vector $\boldsymbol{W} = \{w(1), w(2), \ldots, w(L)\}$ by quantizing $\boldsymbol{X}_1$ through Eq. (3). Then, removing redundant data until $\boldsymbol{W}$ contains un-repeated elements[3] from 1 to L.

$$\boldsymbol{W} = round(\boldsymbol{X}_1 \times 10^{10}) \bmod L + 1. \tag{3}$$

b) Performing permutation to the diffused image $\boldsymbol{D}$ according to

$$c(w(i)) = d(i), i \in [1, L]. \tag{4}$$

The resultant vector $\boldsymbol{C}$ denotes the final ciphertext of this encryption scheme. By defining $\mathcal{W}$ as the permutation function, $\boldsymbol{C}$ can also be described in matrix form, as given in Eq. (5).

$$\boldsymbol{C} = \mathcal{W}(\boldsymbol{D}). \tag{5}$$

3. Cryptanalysis and attack

3.1. Equivalent illustration

As shown in Eq. (2), DNA coding and XOR operations are firstly launched to convert the plain pixel $p(i)$ to a intermediate gray-scale value $t(i)$. The transformation from $p(i)$ to $t(i)$ is reversible and thus bijective. Therefore it can be simplified as

$$t(i) = s_i(p(i)). \tag{6}$$

Here, $\boldsymbol{S}_i = \{s_i(0), s_i(1), \ldots, s_i(256)\}$, $(i \in [1, L])$ is the s-box for substituting $p(i)$. It is introduced for synthesizing the encryption effects of R_x, R_y, R_z, R and the DNA operations (encoding, XOR and decoding), as described in Eq. (2). Specifically, $\boldsymbol{S}_i$ is a bijection from $\mathbb{L} = \{1, 2, \cdots, L\}$ to $\mathbb{L} = \{1, 2, \cdots, L\}$. It is determined by the secret key, and allotted for each plain pixel.

Therefore, the encryption process can be equivalently illustrated in Fig. 1.

Referring to Eqs. (2) and (6), we can further obtain

$$d(i) = sum\_p \oplus \prod\nolimits_{k=1}^{i} s_k(p(k)), \tag{7}$$

where $\prod_{k=1}^{i} s_k(p(k)) = s_1(p(1)) \oplus s_2(p(2)) \oplus \cdots \oplus s_i(p(i))$. As indicated from Eq. (4), $w(i)$ means the position of $d(i)$ in the ciphertext. On the contrary, $w^{-1}(i)$ is defined here as its inverse which means $c(i) = d(w^{-1}(i))$. Therefore, we can get the equivalent representation of the ciphertext as

$$c(i) = sum\_p \oplus \prod\nolimits_{k=1}^{w^{-1}(i)} s_k(p(k)). \tag{8}$$

Instead of the original secret key, i.e., $[a_1, b_1, x_1(0), y_1(0), a_2, b_2, x_2(0), y_2(0)]$, details of s-boxes $\boldsymbol{S}_i$, $(i \in [1, L])$ and permutation vector $w(i)/w^{-1}(i)$ can be regarded as equivalent key elements [18,20,21,40,41]. They are effective for recovering the ciphertexts.

3.2. Cryptanalysis

In cryptanalysis, chosen-plaintext attack refers to the scenario where adversary can freely select input plaintexts and obtain corresponding ciphertexts. Generally, he/she will elaborately construct some special images at first. Then, the secret key or equivalent encryption elements maybe recovered by artful deductions from the plaintext-ciphertext pairs. We use chosen-plaintext attack in the following cryptanalysis. Cryptanalysis of *Wu*'s encryption scheme [34] primary constitutes of the following procedures.

[3] As there maybe repeated numbers, complementary chaotic iterations may be required.

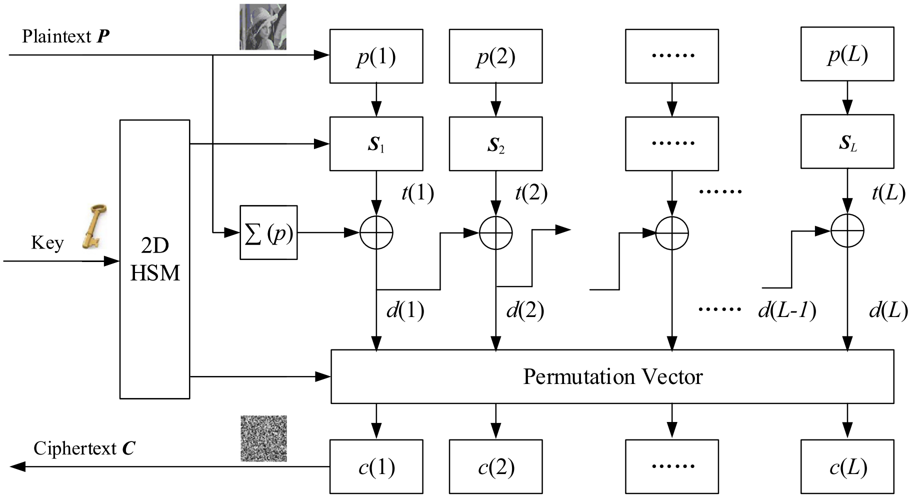


Fig. 1. Equivalent illustration of the encryption scheme under study.

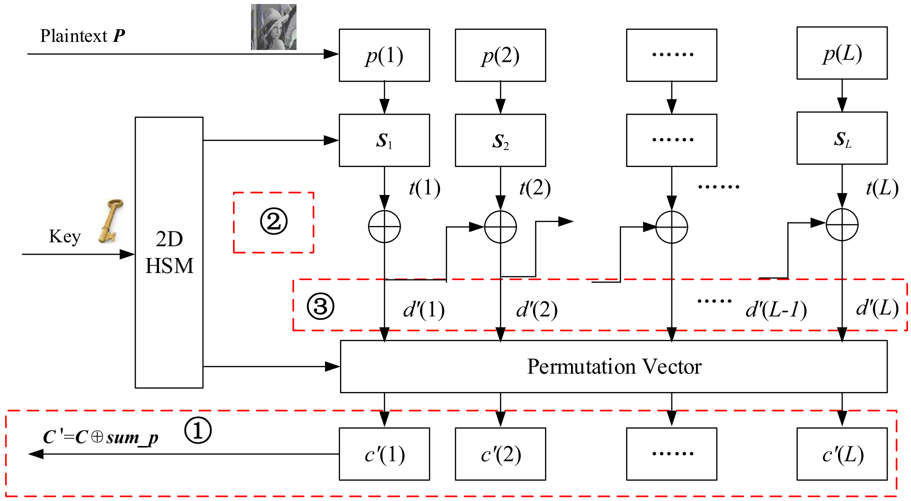


Fig. 2. The equivalent encryption diagram after removing $sum\_p = \sum_{j=1}^{L} p(j)$ mod 256.

1) *Removing the effect of* $sum\_p = \sum_{j=1}^{L} p(j)$ mod 256. As aforementioned, information of the plaintext is available in chosen-plaintext attack. Therefore, $sum\_p = \sum_{j=1}^{L} p(j)$ mod 256 is known.

- Referring to Eq. (8), the encryption effect of the plaintext-related constant sum_p can be straightforward counteracted as

$$c'(i) = sum\_p \oplus c(i) = \prod_{k=1}^{w^{-1}(i)} s_k(p(k)). \tag{9}$$

  The product $c'(i)$ is much helpful for the subsequent cryptanalysis.

- Besides, we further define $d'(i)$ as

$$d'(i) = sum\_p \oplus d(i) = \prod_{k=1}^{i} s_k(p(k)). \tag{10}$$

  Note that $d'(i)$ is not available before recovering the permutation vector $\boldsymbol{W}$. However, it is much beneficial for the theoretical deduction of $\boldsymbol{W}$.

- Combining Eqs. (5), (7)–(10), we can obtain

$$\begin{cases} c'(w(i)) = d'(i), & i \in [1, L] \\ \mathbf{C}' = \mathcal{W}(\mathbf{D}') \end{cases}. \tag{11}$$

- The theoretical illustration of this procedure is shown in Fig. 2. There are three differences in comparison with the original encryption scheme (Fig. 1). The first one is that we perform bit-wise XOR to the original ciphertext,

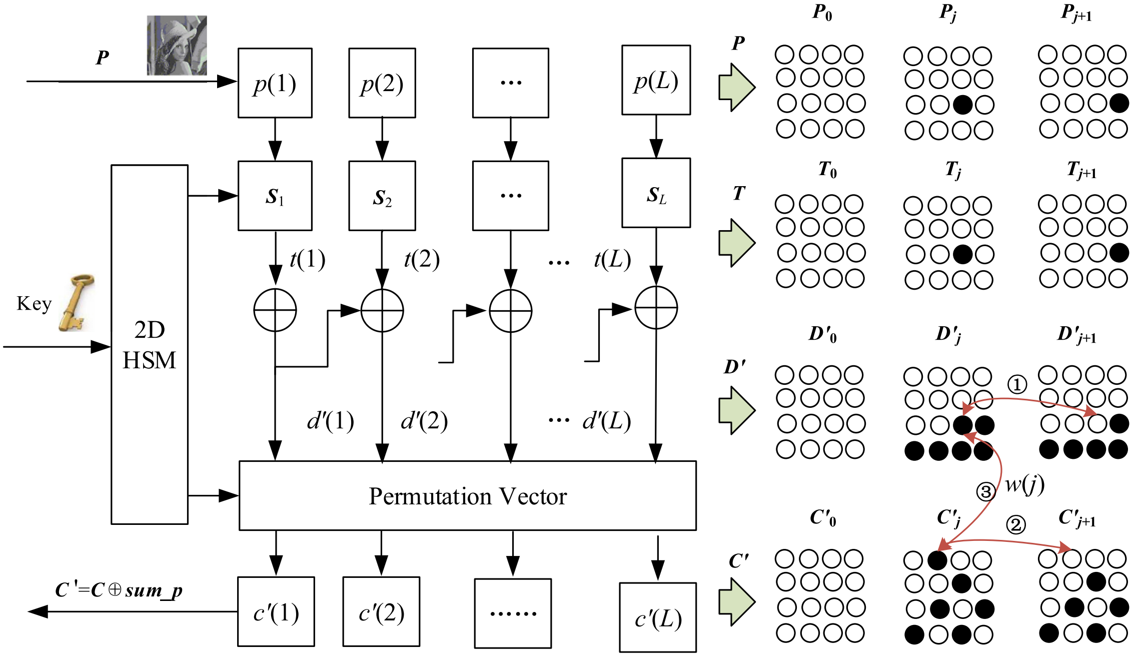


Fig. 3. Illustration of recovering the permutation vector.

as given in Eq. (9), the production is denoted as $\boldsymbol{C}'$. It is equivalent to performing XOR sum_p subsequent to the substitution box. As there is an original sum_p in the cipher, and considering that $sum\_p \oplus sum\_p = 0$, therefore it equals to remove the original sum_p, as illustrated in the second box (it is blank). After removing the original sum_p, the new intermediate product $d'(i)$ is obtained, given in the third box. As described in Eq. (11), $c'(i)$ is the permutation version of $d'(i)$.

- To be concluded, the encryption contribution of the plaintext-related constant $sum\_p = \sum_{j=1}^{L} p(j) \bmod 256$ can be removed by doing an XOR operation to the obtained ciphertext, as given in Eq. (9). The relaxed encryption procedures after removing sum_p are illustrated in Fig. 2, and also in the left half of Fig. 3.

2) *Recovering the permutation vector* **W**.

Fig. 3 is given for clearly illustrating the recovery of the permutation vector. In [34], also described in Figs. 1 and 2, the plain images are firstly stretched to vectors. Therefore, the pixels' coordinates are also given in one-dimensional fashion in the following cryptanalysis. However, these vectors (of images) are represented as a matrix in Fig. 3 for easy representation.

- Suppose there are three images $\boldsymbol{P}_0 = \{p_0(1), \ldots, p_0(j), p_0(j+1), \ldots, p_0(L)\}$, $\boldsymbol{P}_j = \{p_j(1), \cdots, p_j(j), p_j(j+1), \cdots, p_j(L)\}$, and $\boldsymbol{P}_{j+1} = \{p_{j+1}(1), \cdots, p_{j+1}(j), p_{j+1}(j+1), \cdots, p_{j+1}(L)\}$. In comparison with $\boldsymbol{P}_0$, $\boldsymbol{P}_j$ has only one different pixel $p_j(j)$, $\boldsymbol{P}_{j+1}$ also has only one different pixel $p_{j+1}(j+1)$. Their relationships are described in Eqs. (12) and (13), and the different pixels are filled in black in Fig. 3 for easy tracking.

$$\begin{cases} p_0(i) = p_j(i), & i \neq j \\ p_0(i) \neq p_j(i), & i = j \end{cases}. \tag{12}$$

$$\begin{cases} p_0(i) = p_{j+1}(i), & i \neq j+1 \\ p_0(i) \neq p_{j+1}(i), & i = j+1 \end{cases}. \tag{13}$$

- Referring to the equivalent encryption diagram, plain pixels are firstly substituted by the corresponding s-boxes. These s-boxes are identical for $\boldsymbol{P}_0$, $\boldsymbol{P}_j$ and $\boldsymbol{P}_{j+1}$, therefore, the resultant $\boldsymbol{T}_0$, $\boldsymbol{T}_j$ and $\boldsymbol{T}_{j+1}$ have identical relationships as their plaintexts. As illustrated in Fig. 3, $\boldsymbol{T}_0$ and $\boldsymbol{T}_j$ have one different pixel at j, $\boldsymbol{T}_0$ and $\boldsymbol{T}_{j+1}$ have one different pixel at $j+1$.
- When the substitution pixel $t(i)$ undergoes the following cascaded bit-wise XOR modules, their differences will spread to large scale of the products. Referring to Eq. (10) and Fig. 3, we can get the relationship between $\boldsymbol{D}_0'$ and $\boldsymbol{D}_j'$ as

$$\begin{cases} d_0'(i) = d_j'(i), & 1 \leq i \leq j-1 \\ d_0'(i) \neq d_j'(i), & j \leq i \leq L \end{cases}. \tag{14}$$

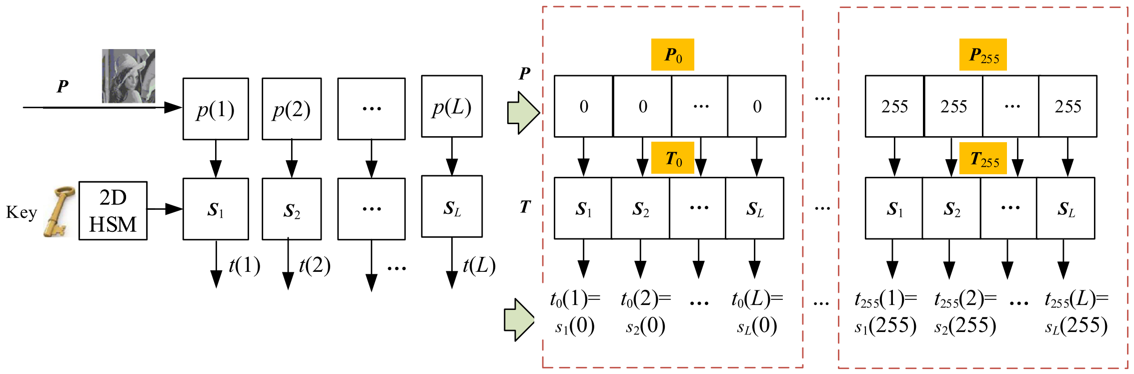


Fig. 4. Equivalent encryption scheme after recovering the permutation vector.

There are a total of $L - j + 1$ consecutive different pixels in $\boldsymbol{D}'_0$ and $\boldsymbol{D}'_j$.
Similarly, the relationship between $\boldsymbol{D}'_0$ and $\boldsymbol{D}'_{j+1}$ is

$$\begin{cases} d'_0(i) = d'_{j+1}(i), & 1 \le i \le j \\ d'_0(i) \ne d'_{j+1}(i), & j+1 \le i \le L \end{cases} . \tag{15}$$

There are a total of $L - j$ consecutive different pixels between $\boldsymbol{D}'_0$ and $\boldsymbol{D}'_{j+1}$.

- We define $\mathbb{A}_{(\boldsymbol{P}_i - \boldsymbol{P}_j)}$ is an assemble contains all the coordinates of different pixels between two images $\boldsymbol{P}_i$ and $\boldsymbol{P}_j$. Considering Eqs. (14) and (15), we can get

$$\begin{cases} \mathbb{A}_{(\boldsymbol{D}'_0 - \boldsymbol{D}'_j)} = \{j, j+1, \cdots, L\} \\ \mathbb{A}_{(\boldsymbol{D}'_0 - \boldsymbol{D}'_{j+1})} = \{j+1, \cdots, L\} \end{cases} . \tag{16}$$

- Permutation is the last process of the encryption, $\boldsymbol{D}'_0$, $\boldsymbol{D}'_j$ and $\boldsymbol{D}'_{j+1}$ will be scrambled to $\boldsymbol{C}'_0$, $\boldsymbol{C}'_j$ and $\boldsymbol{C}'_{j+1}$. However, permutation only changes pixel locations rather than modifying their values. Therefore, the counts of different pixels remain unchanged, but their positions are randomly distributed. Referring to Eqs. (11) and (16), it is obvious that

$$\begin{cases} \mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_j)} = \{w(j), w(j+1), \cdots, w(L)\} \\ \mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_{j+1})} = \{w(j+1), \cdots, w(L)\} \end{cases} . \tag{17}$$

- It is easy to move Eq. (17) one step further, we can get

$$w(j) = \mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_j)} - \mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_{j+1})}. \tag{18}$$

The primary steps for recovering $w(j)$, i.e., Eqs. (16)-(18), are illustrated in three red curves in Fig. 3. Besides, it is easy to conclude that $w(L) = \mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_L)}$. So that, the permutation vector can be recovered by

$$\begin{cases} w(j) = \mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_j)} - \mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_{j+1})} & 1 \le j \le L-1 \\ w(j) = \mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_j)} & j = L \end{cases} . \tag{19}$$

- In chosen-plaintext attack, $\boldsymbol{C}_0$ and $\boldsymbol{C}_j$ and thus $\boldsymbol{C}'_0$ and $\boldsymbol{C}'_j$ are all known, $\boldsymbol{P}_0$ and $\boldsymbol{P}_j$ can even be elaborately constructed. Therefore, the relationship between plaintext-ciphertext pairs and permutation vector can be revealed by Eqs. (9) and (19).
- To be concluded, the permutation vector $\boldsymbol{W}$ is recoverable with $L + 1$ chosen-plaintexts.

3) *Retrieving the s-boxes* $\boldsymbol{S}_i$.
With the knowledge of plaintext $\boldsymbol{P}$, ciphertext $\boldsymbol{C}$ and permutation vector $\boldsymbol{W}$, the intermediate value $\boldsymbol{T}$, as shown in Fig. 1 and Eq. (6), can be reversely deduced out from Eq. (20).

$$\begin{cases} \boldsymbol{C}' = \boldsymbol{C} \oplus \sum_{j=1}^{L} p(j) \bmod 256 \\ \boldsymbol{D}' = \mathcal{W}^{-1}(\boldsymbol{C}') \\ t(i) = d'(i) \oplus d'(i-1) \\ d'(0) = 0 \end{cases} . \tag{20}$$

Till now, the encryption scheme can be further relaxed to a s-box-only image cipher, as illustrated in the left part of Fig 4. The s-boxes, i.e., $\boldsymbol{S}_1, \cdots, \boldsymbol{S}_L$, are the last secret elements to be recovered. Each of them have 256 particles, correspond to the assumed 256 gray-level pixels.

- Using brute force attack, details of s_i can be sequentially retrieved. For example, if we use $\boldsymbol{P}_0 = \{0, 0, \cdots, 0\}$ as the input, and get corresponding $\boldsymbol{T}_0 = \{t_0(1), \ldots, t_0(j), \ldots t_0(L)\}$. Referring to Eq. (6) we can obviously get that $s_1(0) = t_0(1), s_2(0) = t_0(2), \cdots, s_L(0) = t_0(L)$. Mathematically, chosen-plaintexts are constructed by $\boldsymbol{P}_i = \{i, i, \cdots, i\}, (i \in [0, 255])$, then $\boldsymbol{T}_i$ can be obtained referring to Eq. (20). Further, these s-boxes are retrieved according to

$$\begin{cases} s_1(i) = t_i(1) \\ s_2(i) = t_i(2) \\ \vdots \\ s_L(i) = t_i(L) \end{cases}. \tag{21}$$

- To be concluded, 256 plain images are sufficient for retrieving the full knowledge of the s-boxes.

3.3. Attack procedures

Based on the aforementioned cryptanalysis, an executable chosen-plaintext attack is drawn as follows.

1) Constructing $L+1$ chosen-plaintexts $\boldsymbol{P}_0, \boldsymbol{P}_1, \cdots, \boldsymbol{P}_i, \cdots, \boldsymbol{P}_L$, their pixels are

$$\begin{aligned} & p_0(j) = 0, \quad j \in [1, L] \\ & p_1(j) = \begin{cases} 1, j = 1 \\ 0, otherwise \end{cases} \\ & \vdots \\ & p_i(j) = \begin{cases} 1, j = i \\ 0, otherwise \end{cases} \\ & \vdots \\ & p_L(j) = \begin{cases} 1, j = L \\ 0, otherwise \end{cases} \end{aligned}. \tag{22}$$

Then obtaining their ciphertexts, denoted as $\boldsymbol{C}_0, \boldsymbol{C}_1, \cdots, \boldsymbol{C}_i, \cdots, \boldsymbol{C}_L$, respectively.

2) Referring to Eq. (9), calculate $\boldsymbol{C}'_0, \boldsymbol{C}'_1, \cdots, \boldsymbol{C}'_i, \cdots, \boldsymbol{C}'_L$.

3) Comparing $\boldsymbol{C}'_0$ with $\boldsymbol{C}'_1, \cdots, \boldsymbol{C}'_i, \cdots, \boldsymbol{C}'_L$ respectively to get the coordinates of differential pixels, i.e., $\mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_1)}, \mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_2)}, \cdots, \mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_i)}, \cdots, \mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_L)}$.

4) Recovering the particles of permutation matrix $\boldsymbol{W}$ according to Eq. (19).

5) Constructing another 255 plaintexts denoted as $\boldsymbol{P}_{1-2}, \cdots, \boldsymbol{P}_{i-2}, \cdots, \boldsymbol{P}_{255-2}$, whose pixels are

$$p_{i-2}(j) = i, \quad i \in [1, 255], j \in [1, L].$$

Their ciphertexts are named as $\boldsymbol{C}_{1-2}, \cdots, \boldsymbol{C}_{i-2}, \cdots, \boldsymbol{C}_{255-2}$, respectively. Without loss of generality, $\boldsymbol{C}_0$ in Step 1) is re-written as $\boldsymbol{C}_{0-2}$ for consistency.

6) With the help of $\boldsymbol{C}_{0-2}, \boldsymbol{C}_{1-2}, \cdots, \boldsymbol{C}_{i-2}, \cdots, \boldsymbol{C}_{L-2}$ and the recovered permutation vector $\boldsymbol{W}$ in Step 4), details of the s-boxes $\boldsymbol{S}_i$ can be recovered according to Eqs. (20) and (21).

4. Results and discussions

The aforementioned cryptanalysis and chosen-plaintext attack have been experimentally verified. In these validating attempts, input key is randomly selected as $[a_1, b_1, x_1(0), y_1(0), a_2, b_2, x_2(0), y_2(0)] = [1.31, 1.32, 0.81, 0.82, 1.13, 1.14, 0.83, 0.84]$, and the gray-level of the test images are assumed as 256. The proposed attack is implemented by Matlab 2018a, and source codes are openly accessible[4]

4.1. Illustration experiment

The image size is firstly assumed as 3 × 3 for giving a clear illustration. The attack is implemented strictly according to the steps listed in Section 3.3.

- *Steps 1–2*. As there are 9 pixels in the plaintext, $9 + 1 = 10$ chosen-plaintexts are firstly employed to recover the permutation vector. They are constructed according to Eq. (22), and listed in the first two columns in Table 3. Referring to the attack procedures, these plaintexts are encrypted and then bit-wise XORed with corresponding *sum_p*. The resultant $\boldsymbol{C}$ and $\boldsymbol{C}'$ are also given in Table 3.

[4] You can access the source codes via https://github.com/lurenjia212/Breaking_DNA_Henon.

Table 3
Recovery of the permutation vector when the image size is 3 × 3 (attack steps 1–2).

Chosen-plaintexts $\boldsymbol{P}$		Ciphertexts $\boldsymbol{C}$		Products $\boldsymbol{C}'$ after removing *sum_p*	
$\boldsymbol{P}_0$	0,0,0,0,0,0,0,0,0	$\boldsymbol{C}_0$	61,253,201,213,202,33,255,236,203	$\boldsymbol{C}'_0$	61,253,201,213,202,33,255,236,203
$\boldsymbol{P}_9$	1,0,0,0,0,0,0,0,0	$\boldsymbol{C}_9$	60,252,200,212,203,34,254,237,202	$\boldsymbol{C}'_9$	$61,253,201,213,202,\underline{35},255,236,203$
$\boldsymbol{P}_8$	0,1,0,0,0,0,0,0,0	$\boldsymbol{C}_8$	60,252,200,212,203,33,254,237,203	$\boldsymbol{C}'_8$	$61,253,201,213,202,\underline{32},255,236,\underline{202}$
$\boldsymbol{P}_7$	0,0,1,0,0,0,0,0,0	$\boldsymbol{C}_7$	60,252,200,212,203,33,255,237,203	$\boldsymbol{C}'_7$	$61,253,201,213,202,\underline{32},\underline{254},236,\underline{202}$
$\boldsymbol{P}_6$	0,0,0,1,0,0,0,0,0	$\boldsymbol{C}_6$	60,252,200,212,201,34,252,237,200	$\boldsymbol{C}'_6$	$61,253,201,213,\underline{200},\underline{35},\underline{253},236,\underline{201}$
$\boldsymbol{P}_5$	0,0,0,0,1,0,0,0,0	$\boldsymbol{C}_5$	60,252,202,212,201,34,252,237,200	$\boldsymbol{C}'_5$	$61,253,\underline{203},213,\underline{200},\underline{35},\underline{253},236,\underline{201}$
$\boldsymbol{P}_4$	0,0,0,0,0,1,0,0,0	$\boldsymbol{C}_4$	62,252,202,212,201,34,252,237,200	$\boldsymbol{C}'_4$	$\underline{63},253,\underline{203},213,\underline{200},\underline{35},\underline{253},236,\underline{201}$
$\boldsymbol{P}_3$	0,0,0,0,0,0,1,0,0	$\boldsymbol{C}_3$	61,252,201,212,202,33,255,236,203	$\boldsymbol{C}'_3$	$\underline{60},253,\underline{200},213,\underline{203},\underline{32},\underline{254},\underline{237},\underline{202}$
$\boldsymbol{P}_2$	0,0,0,0,0,0,0,1,0	$\boldsymbol{C}_2$	62,254,202,212,201,34,252,239,200	$\boldsymbol{C}'_2$	$\underline{63},\underline{255},\underline{203},213,\underline{200},\underline{35},\underline{253},\underline{238},\underline{201}$
$\boldsymbol{P}_1$	0,0,0,0,0,0,0,0,1	$\boldsymbol{C}_1$	62,254,202,214,201,34,252,239,200	$\boldsymbol{C}'_1$	$\underline{63},\underline{255},\underline{203},\underline{215},\underline{200},\underline{35},\underline{253},\underline{238},\underline{201}$

Table 4
Recovery of the permutation vector when the image size is 3 × 3 (attack steps 3–4).

	Calculating $\mathbb{A}_{(\boldsymbol{C}'_0-\boldsymbol{C}'_j)}$	Recover $w(j)=\mathbb{A}_{(\boldsymbol{C}'_0-\boldsymbol{C}'_j)}-\mathbb{A}_{(\boldsymbol{C}'_0-\boldsymbol{C}'_{j+1})}$
$j=9$	6	$w(9)=6$
$j=8$	$6,\ \underline{9}$	$w(8)=9$
$j=7$	$6,\ \underline{7},\ 9$	$w(7)=7$
$j=6$	$\underline{5},\ 6,\ 7,\ 9$	$w(6)=5$
$j=5$	$\underline{3},\ 5,\ 6,\ 7,\ 9$	$w(5)=3$
$j=4$	$\underline{1},\ 3,\ 5,\ 6,\ 7,\ 9$	$w(4)=1$
$j=3$	$1,\ 3,\ 5,\ 6,\ 7,\ \underline{8},\ 9$	$w(3)=8$
$j=2$	$1,\ \underline{2},\ 3,\ 5,\ 6,\ 7,\ 8,\ 9$	$w(2)=2$
$j=1$	$1,\ 2,\ 3,\ \underline{4},\ 5,\ 6,\ 7,\ 8,\ 9$	$w(1)=4$

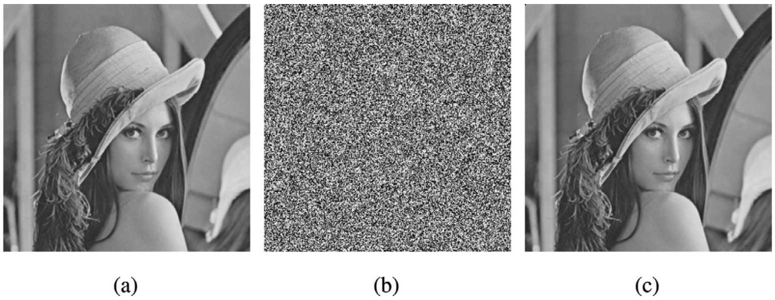
Table 5
Recovery of the substitution boxes when the image size is 3 × 3 (attack steps 5–6).

Chosen-plaintexts $\boldsymbol{P}$		Ciphertexts $\boldsymbol{C}$		Calculated intermediate elements $\boldsymbol{T}$		Information of $\boldsymbol{S}$
$\boldsymbol{P}_{0-2}$	0,0,0,0,0,0,0,0,0	$\boldsymbol{C}_{0-2}$	61,253,201,213,202,33,255,236,203	$\boldsymbol{T}_{0-2}$	$\underline{213},40,17,209,244,3,53,52,234$	$s_1(0)\to s_9(0)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\boldsymbol{P}_{99-2}$	$99,99,\cdots,\ 99$	$\boldsymbol{C}_{99-2}$	182,134,209,61,65,57,23,244,64	$\boldsymbol{T}_{99-2}$	$70,\underline{187},114,66,103,144,86,87,121$	$s_1(99)\to s_9(99)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\boldsymbol{P}_{179-2}$	$179,179,\cdots,\ 179$	$\boldsymbol{C}_{179-2}$	182,182,49,237,65,217,199,20,64	$\boldsymbol{T}_{179-2}$	$166,91,162,162,135,112,134,135,\underline{153}$	$s_1(179)\to s_9(179)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\boldsymbol{P}_{255-2}$	$255,255,\cdots,\ 255$	$\boldsymbol{C}_{255-2}$	202,10,193,221,61,41,247,228,60	$\boldsymbol{T}_{255-2}$	42,215,238,46,11,252,202,203,21	$s_1(255)\to s_9(255)$

- *Steps 3–4*. Steps 3 and 4 are the deduction processes of the permutation vector. We firstly obtain $\mathbb{A}_{(\boldsymbol{C}'_0-\boldsymbol{C}'_j)}$, i.e., the coordinates of the differential pixels between $\boldsymbol{C}'_0$ and its peers. The differential pixels have been underlined in Table 3, and their coordinates are given in the second column of Table 4. The following step for recovering the permutation matrix is to compare $\mathbb{A}_{(\boldsymbol{C}'_0-\boldsymbol{C}'_j)}$ according to Eq. (19). The differential particles between $\mathbb{A}_{(\boldsymbol{C}'_0-\boldsymbol{C}'_j)}$ and $\mathbb{A}_{(\boldsymbol{C}'_0-\boldsymbol{C}'_{j+1})}$ have been underlined in the second column of Table 4, and the corresponding recovered elements of the permutation vector are listed in the third column.
- *Steps 5–6*. The s-boxes are retrieved by the 5th and 6th steps of the proposed attack. A total of 256 chosen-plaintexts are required for completely recovering all the s-boxes. For each plaintext-ciphertext pair, we firstly use *sum_p* and the recovered permutation matrix $\boldsymbol{W}$ to obtain a intermediate matrix $\boldsymbol{T}$, as given in Eq. (20). We know that $\boldsymbol{T}$ refers to the production of the s-box substitution, the s-boxes can be thus inversely recovered according to Eq. (21). The results are partially listed in Table 5. For example, the first row lists the recovered elements by $\boldsymbol{P}_{0-2}$, they are $s_1(0)=213, s_2(0)=40,\cdots,s_9(0)=234$. In the encryption part, $p(1)=0$ will be transformed to $s_1(0)=213$ after the DNA encryption procedures (synthesized as the s-box $\boldsymbol{S}_1$). Similarly, $p(9)=0$ will be converted to $s_9(0)=234$ whereas $p(2)=99$ will be converted to $s_2(99)=187$.

With the obtained permutation vector and s-boxes, any received ciphertext can be recovered. Suppose that there is a plaintext $\boldsymbol{P}=\{0,99,212,22,199,126,58,77,179\}$, it is encrypted into $\boldsymbol{C}=\{159,162,160,25,30,241,17,103,104\}$ using the aforementioned secret key. The plaintext can be recovered referring to the inverse implementation of the equivalent encryption process that is depicted in Fig. 1.

- The first step is doing a inverse permutation, whose forward scrambling rule is given in Eq. (4) and the permutation vector has been recovered in Table 4. We can get $\boldsymbol{D}=\{25,162,103,159,160,30,17,104,241\}$.



(a) (b) (c)

Fig. 5. Experiment results of the proposed chosen-plaintext attack: (a) plain image lenna; (b) ciphertext using the given key; (c) recovered image with the retrieved equivalent key elements.

- The next issue is the conversion from $\boldsymbol{D}$ to $\boldsymbol{T}$. Through Eq. (2), it is obvious that $t(i) = d(i) \oplus d(i-1)$ with $d(0) = sum\_p$ for the first particle[5]The resultant $\boldsymbol{T} = \{213, 187, 197, 248, 63, 190, 15, 121, 153\}$.
- Finally, the plain pixels are sequentially recovered by searching the s-boxes, referring to Eq. (6). For example, we know that the 1st particle $t(1) = 213$. Then we search the s-box for the 1st pixel, i.e., $\boldsymbol{S}_1$, we can get $s_1(0) = 213$, as underlined in the first row of Table 5. Therefore, $p(1) = 0$. For the 2nd pixel, $t(2) = 187$ corresponds to $s_2(99) = 187$, thus $p(2) = 99$. The final recovered pixel is $p(9) = 179$ because $t(9) = 153 = s_9(179)$. Repeating sequentially for all pixels, the recovered image is $\boldsymbol{P} = \{0, 99, 212, 22, 199, 126, 58, 77, 179\}$. The particles $s_2(99) = 187$ and $s_9(179) = 153$ are also underlined in the corresponding rows in Table 5.
- As can be observed, the plaintext has been successfully recovered by the equivalent encryption elements. The proposed attack is thus validated.

4.2. Experimental results

The proposed attack is further verified when the image size is 256 × 256. The recovered permutation vector is partially given in Eq. (23). The $\boldsymbol{S}_{65536}$ which refers to the s-box for the last pixel, is partially demonstrated in Eq. (24). With these equivalent elements, the received ciphertexts can be successfully recovered. Fig. 5(a) is the standard lenna image, its ciphertext use aforementioned secret key is shown in Fig. 5(b). It has been successfully recovered, as demonstrated in Fig. 5(c).

$$\boldsymbol{W}_{\_rec} = \begin{bmatrix} 50975 & 17841 & \cdots & 64133 \\ 8909 & 4945 & \cdots & 30873 \\ \vdots & \vdots & \ddots & \vdots \\ 29305 & 41654 & \cdots & 46461 \end{bmatrix}_{256\times256} \tag{23}$$

$$\begin{bmatrix} 0 & 1 & \cdots & 15 \\ 16 & 17 & \cdots & 31 \\ \vdots & \vdots & \ddots & \vdots \\ 240 & 241 & \cdots & 255 \end{bmatrix}_{Plain\ \ pixel} \Leftrightarrow \begin{bmatrix} 49 & 33 & \cdots & 193 \\ 48 & 32 & \cdots & 192 \\ \vdots & \vdots & \ddots & \vdots \\ 62 & 46 & \cdots & 206 \end{bmatrix}_{Encrypted\ \ pixel} \tag{24}$$

4.3. Discussions

As described in Section 3.3, a total of $L + 255$ chosen-plaintexts are sufficient to recover the permutation vector and s-boxes. Referring to Eq. (19), particles of permutation vector can be produced one-by-one in a reverse order, which means $w(L)$ is firstly obtained and $\{w(L-1), w(L-2), \cdots, w(1)\}$ will be sequentially recovered. In this case, only $\boldsymbol{C}_0$, $\boldsymbol{C}_j$, $\mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_j)}$, and $\mathbb{A}_{(\boldsymbol{C}'_0 - \boldsymbol{C}'_{j+1})}$ are required for the recovery of $w(j)$, as indicated in Eq. (19). In other words, information of $\boldsymbol{C}_j/\boldsymbol{C}'_j$ is not required to be stored permanently, and the space complexity can be significantly relaxed to be acceptable by a personal computer.

Recently, plaintext-related key stream generation mechanism has been widely adopted in chaos-based image ciphers [31,33,38]. Typically, a plaintext-related variable (a random pixel or statistical feature of the image) is firstly extracted, and subsequently used to disturb the generation of key stream elements. In this scene, different plaintexts will bring about distinct key elements, and the avalanche effect is thus achieved. However, this plaintext-related perturbation should be

[5] In the original paper (Section 3.3 in [34]), $sum\_p = \sum_{j=1}^{L} p(j) \bmod 256$ is not obtainable. If so, the first pixel cannot be recovered correctly. However, brute force attack can be employed as a complementary step.

transmitted as component of secret key to the receiver. Otherwise, the key scheduling process cannot be correctly implemented and the decryption is consequently infeasible. Taking the cipher under study as an example, summation of plain image *sum_p* is *seed* of the diffusion process. However, it does not serve as component of secret key and further makes the first plain pixel cannot be correctly recovered.

Essentially, such encryption schemes with plaintext-related key stream generation mechanism cannot be regarded as one-time-pad (OTP) ciphers either. That is because the key stream of OTP should must be independent from the plaintext. This is a basic cryptographic requirement according to Shannon's principle [25]. Prior to the starting of secure transmission, key negotiation is firstly implemented without information of what will be encrypted and when the encryption is performed. In other words, the contracted key is feasible for encrypting/decrypting arbitrary plaintexts rather than a specified one [13,15,31,33]. However, input secret key or key stream elements of recently proposed OTP image ciphers [13,15,31,33] are produced with the extracted information of the plaintext. They cannot be generated without knowledge of plaintexts and hence doesn't comply with Shannon's theory. Besides, key streams of these ciphers are essentially produced case-by-case in practical applications. Such a pattern makes the key exchanging more complex and decrease the encryption efficiency at the same time.

However, it is quite another thing when the plaintext-related disturbance is retrievable in the decryption end. The fashion adopted in [6,39] is a typical case. It takes advantageous that statistical features of plaintext and permutation ciphertext are identical since permutation does not change pixel values. In [6], counts of A, T, C, G of the produced DNA base string are mixed as the plaintext-related constant to disturb key stream generation of the permutation process. Therefore, different plaintexts bring about individual permutation vectors. In this case, plaintext attacks to recover permutation vector [18,20,21] are consequently infeasible because the produced permutation vectors are different from images. Namely, the permutation vector in chosen-plaintext scenario is not the one in other interested encryption case. Besides, this perturbation can be self-adaptively recovered from the permutation ciphertext thus is not required to be transmitted as component of the key. Self-adaptive retrievable plaintext-related disturbance is advisable for the chaotic cipher in the future.

5. Conclusions

In this paper, the security of an image encryption scheme using Hénon-Sine map and DNA random coding is evaluated. Substitution boxes are firstly introduced to equivalently synthesize the encryption effects of DNA random coding and XOR operations. Details of the substitution boxes and permutation vector are regarded as equivalent encryption elements. Comprehensive cryptanalysis is conducted, based on which a chosen-plaintext attack is step-by-step described and experimentally verified. Discussions with respect to the design and cryptanalysis of chaotic ciphers are also given, and we hope they are beneficial for the future developments of chaotic cryptography.

Declaration of Competing Interest

The authors declare that they have no conflict of interest.

Acknowledgments

This work is funded by the National Natural Science Foundation of China (Nos. 61802055, 61771121, 61801104), in part by The Science and Technology Development Fund, Macau SAR (File no. 189/2017/A3), and by the Research Committee at University of Macau under Grants MYRG2016-00123-FST and MYRG2018-00136-FST.

References

[1] A. Akhavan, A. Samsudin, A. Akhshani, Cryptanalysis of an image encryption algorithm based on DNA encoding, Opt. Laser Technol. 95 (2017) 94–99.
[2] G. Alvarez, S. Li, Some basic cryptographic requirements for chaos-based cryptosystems, Int. J. Bifur. Chaos 16 (8) (2006) 2129–2151.
[3] A. Belazi, A.A.A. Ellatif, S. Belghith, A novel image encryption scheme based on substitution-permutation network and chaos, Signal Process. 128 (2016) 155–170.
[4] X. Chai, Y. Chen, L. Broyde, A novel chaos-based image encryption algorithm using DNA sequence operations, Opt. Lasers Eng. 88 (2017) 197–213.
[5] X. Chai, Z. Gan, K. Yang, Y. Chen, X. Liu, An image encryption algorithm based on the memristive hyperchaotic system, cellular automata and DNA sequence operations, Signal Process. 52 (2017) 6–19.
[6] J. Chen, Z. Zhu, L. Zhang, Y. Zhang, B. Yang, Exploiting self-adaptive permutation-diffusion and DNA random encoding for secure and efficient image encryption, Signal Process. 142 (2018) 340–353.
[7] A.-V. Diaconu, Circular inter–intra pixels bit-level permutation and chaos-based image encryption, Inf. Sci. 355 (2016) 314–327.
[8] J. Fridrich, Symmetric ciphers based on two-dimensional chaotic maps, Int. J. Bifur. Chaos 8 (6) (1998) 1259–1284.
[9] C. Fu, W. Meng, Y. Zhan, Z. Zhu, F.C.M. Lau, C.K. Tse, H. Ma, An efficient and secure medical image protection scheme based on chaotic maps, Comput. Biol. Med. 43 (8) (2013) 1000–1010.
[10] X.-Q. Fu, B.-C. Liu, Y.-Y. Xie, W. Li, Y. Liu, Image encryption-then-transmission using DNA encryption algorithm and the double chaos, IEEE Photon. J. 10 (3) (2018) 1–15.
[11] A. Girdhar, V. Kumar, A RGB image encryption technique using Lorenz and Rossler chaotic system on DNA sequences, Multimed. Tools Appl. (2018) 1–23.
[12] R. Girija, H. Singh, Enhancing security of double random phase encoding based on random S-Box, 3D Res. 9 (2) (2018) 15.
[13] R. Guesmi, M. Farah, A. Kachouri, M. Samet, A novel chaos-based image encryption using DNA sequence operation and secure hash algorithm SHA-2, Nonlinear Dyn. 3 (83) (2016) 1123–1136.
[14] M. Hamdi, R. Rhouma, S. Belghith, A selective compression-encryption of images based on SPIHT coding and Chirikov standard map, Signal Process. 131 (2017) 514–526.

[15] T. Hu, Y. Liu, L. Gong, S. Guo, H. Yuan, Chaotic image cryptosystem using DNA deletion and DNA insertion, Signal Process. 134 (2017) 234–243.
[16] Z. Hua, B. Zhou, Y. Zhou, Sine-transform-based chaotic system with FPGA implementation, IEEE Trans. Ind. Electron. 65 (3) (2018) 2557–2566.
[17] I. Hussain, J. Ahmed, A. Hussain, An image encryption technique based on coupled map lattice and one-time S-boxes based on complex chaotic system, J. Intell. Fuzzy Syst. 29 (4) (2015) 1493–1500.
[18] A. Jolfaei, X. Wu, V. Muthukkumarasamy, On the security of permutation-only image encryption schemes, IEEE Trans. Inf. Forensics Secur. 11 (2) (2016) 235–246.
[19] M. Kumar, A. Vaish, An efficient encryption-then-compression technique for encrypted images using SVD, Digit. Signal Process. 60 (2017) 81–89.
[20] C. Li, K.-T. Lo, Optimal quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks, Signal Process. 91 (4) (2011) 949–954.
[21] S. Li, C. Li, G. Chen, N.G. Bourbakis, K. Lo, A general quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks, Signal Process. 23 (3) (2008) 212–223.
[22] Y. Liu, J. Tang, T. Xie, Cryptanalyzing a RGB image encryption algorithm based on DNA encoding and chaos map, Opt. Laser Technol. 60 (2014) 111–115.
[23] K. Nosrati, C. Volos, A. Azemi, Cubature Kalman filter-based chaotic synchronization and image encryption, Signal Process. 58 (2017) 35–48.
[24] P. Ping, F. Xu, Y. Mao, Z. Wang, Designing permutation–substitution image encryption networks with Henon map, Neurocomputing 283 (2018) 53–63.
[25] C.E. Shannon, Communication theory of secrecy systems, Bell Syst. Tech. J.l 28 (4) (1949) 656–715.
[26] H. Singh, Nonlinear optical double image encryption using random-optical vortex in fractional Hartley transform domain, Opt. Appl. 47 (4) (2017).
[27] H. Singh, A. Yadav, S. Vashisth, K. Singh, Fully phase image encryption using double random-structured phase masks in gyrator domain, Appl. Opt. 53 (28) (2014) 6472–6481.
[28] H. Singh, A. Yadav, S. Vashisth, K. Singh, Double phase-image encryption using gyrator transforms, and structured phase mask in the frequency plane, Opt. Lasers Eng. 67 (2015) 145–156.
[29] E. Solak, O.T. Yildiz, Cryptanalysis of Fridrich's chaotic image encryption, Int. J. Bifur. Chaos 20 (5) (2010) 1405–1413.
[30] X. Tong, The novel bilateral - diffusion image encryption algorithm with dynamical compound chaos, J. Syst. Softw. 85 (4) (2012) 850–858.
[31] X. Wang, Y. Zhang, X. Bao, A novel chaotic image encryption scheme using DNA sequence operations, Opt. Lasers Eng. 73 (73) (2015) 53–61.
[32] Y. Wang, Y. Zhao, Q. Zhou, Z. Lin, Image encryption using partitioned cellular automata, Neurocomputing 275 (2018) 1318–1332.
[33] X. Wei, L. Guo, Q. Zhang, J. Zhang, S. Lian, A novel color image encryption algorithm based on DNA sequence operation and hyper-chaotic system, J. Syst. Softw. 85 (2) (2012) 290–299.
[34] J. Wu, X. Liao, B. Yang, Image encryption using 2D Hénon-Sine map and DNA approach, Signal Process. 153 (2018) 11–23.
[35] X. Wu, K. Wang, X. Wang, H. Kan, Lossless chaotic color image cryptosystem based on DNA encryption and entropy, Nonlinear Dyn. 90 (2) (2017) 855–875.
[36] X. Wu, K. Wang, X. Wang, H. Kan, J. Kurths, Color image DNA encryption using NCA map-based CML and one-time keys, Signal Process. 148 (2018) 272–287.
[37] Y.-G. Yang, J. Tian, H. Lei, Y.-H. Zhou, W.-M. Shi, Novel quantum image encryption using one-dimensional quantum cellular automata, Inf. Sci. 345 (2016) 257–270.
[38] G. Ye, X. Huang, An efficient symmetric image encryption algorithm based on an intertwining logistic map, Neurocomputing 251 (2017) 45–53.
[39] L.Y. Zhang, X. Hu, Y. Liu, K. Wong, J. Gan, A chaotic image encryption scheme owning temp-value feedback, Commun. Nonlinear Sci. Numer. Simul. 19 (10) (2014) 3653–3659.
[40] L.Y. Zhang, Y. Liu, C. Wang, J. Zhou, Y. Zhang, G. Chen, Improved known-plaintext attack to permutation-only multimedia ciphers, Inf. Sci. 430 (2018) 228–239.
[41] L.Y. Zhang, Y. Liu, K.-W. Wong, F. Pareschi, Y. Zhang, R. Rovatti, G. Setti, On the security of a class of diffusion mechanisms for image encryption, IEEE Trans. Cybern. (99) (2017) 1–13.
[42] Q. Zhang, L. Guo, X. Wei, A novel image fusion encryption algorithm based on DNA sequence operation and hyper-chaotic system, Optik 124 (18) (2013) 3596–3600.
[43] Y. Zhang, W. Wen, M. Su, M. Li, Cryptanalyzing a novel image fusion encryption algorithm based on DNA sequence operation and hyper-chaotic system, Optik 125 (4) (2014) 1562–1564.
[44] Y. Zhang, D. Xiao, W. Wen, K. Wong, On the security of symmetric ciphers based on DNA coding, Inf. Sci. 289 (2014) 254–261.
[45] Y. Zhang, J. Zhou, F. Chen, L.Y. Zhang, D. Xiao, B. Chen, X. Liao, A block compressive sensing based scalable encryption framework for protecting significant image regions, Int. J. Bifur. Chaos 26 (11) (2016) 1650191.
[46] Y.-Q. Zhang, X.-Y. Wang, A symmetric image encryption algorithm based on mixed linear–nonlinear coupled map lattice, Inf. Sci. 273 (2014) 329–351.
[47] N. Zhou, S. Pan, S. Cheng, Z. Zhou, Image compression–encryption scheme based on hyper-chaotic system and 2D compressive sensing, Opt. Laser Technol. 82 (2016) 121–133.