Security Analysis of a Color Image Encryption Scheme Based on a Fractional-Order Hyperchaotic System

George Teşeleanu

[1] Advanced Technologies Institute
10 Dinu Vintilă, Bucharest, Romania
tgeorge@dcti.ro
[2] Simion Stoilow Institute of Mathematics of the Romanian Academy
21 Calea Grivitei, Bucharest, Romania

Abstract. In 2022, Hosny *et al.* introduce an image encryption scheme that employs a fractional-order chaotic system. Their approach uses the hyper-chaotic system to generate the system's main parameter, namely a secret permutation which is dependent on the size and the sum of the pixels of the source image. According to the authors, their scheme offers adequate security (*i.e.* 498 bits) for transmitting color images over unsecured channels. Nevertheless, in this paper we show that the scheme's security is independent on the secret parameters used to initialize the hyper-chaotic system. More precisely, we provide a chosen plaintext/ciphertext attack whose complexity is $\mathcal{O}(6(WH)^2)$ and needs WH oracle queries, where W and H are the width and the height of the encrypted image. For example, for an image of size 4000×3000 (12 megapixels image) we obtain a security margin of 49.61 bits, which is 10 times lower than the claimed bound.

1 Introduction

The widespread use of social media has raised concerns about the security of digital images, particularly the risk of theft and unauthorized distribution. As a result, this issue has raised significant attention, leading researchers to develop various techniques for encrypting images. Among these approaches, chaotic maps have become a popular choice due to their high sensitivity to initial conditions and previous states. This desirable property makes it difficult to predict their behavior, leading to the development of several novel cryptographic algorithms based on chaos. However, many of these image encryption schemes suffer from critical security vulnerabilities due to inadequate security analysis and a lack of design guidelines. In fact, there have been numerous compromised schemes, which we provide in a non-exhaustive list in Table 1. For more information, please refer to [9, 27, 29, 49].

Scheme	[45]	[24]	[40]	[13]	[14]	[34]	[4]	[11]	[28]	[12]	[25]
Broken by	[20]	[39]	[2]	[43]	[1]	[42]	[11]	[17]	[16]	[47]	[36]
Scheme	[31]	[21]	[32]	[33]	[44]	[46]	[15]	[30]	[26]	[6]	[7]
Broken by	[38]	[23]	[41]	[48]	[5]	[22]	[8]	[18]	[19]	[35]	[37]

Table 1. Broken chaos based image encryption algorithms.

In [10], the authors propose a novel encryption scheme based on the 4D hyper-chaotic Chen system combined with a Fibonacci Q-matrix. Before encrypting the image, the authors first decompose it into its primary color channels: red, green and blue. Then they process each channel independently. More precisely, they use six secret parameters, the size of the image and the sum of its pixels to initialize the hyper-chaotic system. Then they discard a part of the system's outputs and the remaining ones are used to generate a random permutation. After scrambling the image according to the computed permutation, they apply the Fibonacci Q-matrix for each 2×2 image blocks. Finally, they recombine the resulting three encrypted images into one image. Since the Chen system is simply used as a pseudorandom number generator (PRNG) and the scheme's weakness is independent of the employed generator, we omit its description.

In this paper, we conduct a security analysis of the Hosny *et al.* scheme [10]. We describe a chosen plaintext attack and a chosen ciphertext attack, which would allow an attacker to decrypt all images of a specific size. To execute such attacks, the attacker would need to access the ciphertexts or plaintexts of at most WH chosen plaintexts or chosen ciphertexts. Once the attacker has this information in his possession, he can proceed to running his attack. Note that the attack's complexity is not related to the size of secret parameters of the PRNG used in the encryption scheme. It is solely determined by the size of the image being attacked. In the case of 2 megapixels[3] images, the complexity of the attack is estimated to be $\mathcal{O}(2^{44.33})$. On the other hand, in the case of 12 megapixels[4], we obtain an estimate of $\mathcal{O}(2^{49.61})$. The security gap between the estimated complexity of the attack and the claimed security level of the Hosny *et al.*'s scheme is quite large in both cases (*i.e.* 10 times lower). According to [3], a security of 80 bits is considered only for legacy systems, and thus it should not be used for applying cryptographic protection. Therefore, Hosny *et al.* scheme does not provide sufficient assurances in order to be used in practice.

Structure of the paper. We provide the necessary preliminaries in Section 2. In Sections 3 and 4 we show how an attacker can recover the secret permutation in a chosen plaintext/ciphertext scenario. We conclude in Section 5.

[3] $W \times H = 1600 \times 1200$

[4] $W \times H = 4000 \times 3000$

Algorithm 1: Encryption algorithm.

Input: A plaintext P and a secret key K
Output: A ciphertext C
Generate a random permutation S using $PRNG(K, IC)$.
%image scrambling
for $i \in [0, L)$ **do** $R_i \leftarrow P_{S_i}$
%add diffusion
for $i \in [0, L)$ *and at each step increment* i *with* 2 **do**
  $C_i \leftarrow 89 \cdot R_i + 55 \cdot R_{i+1} \bmod 256$
  $C_{i+1} \leftarrow 55 \cdot R_i + 34 \cdot R_{i+1} \bmod 256$
return C

Algorithm 2: Decryption algorithm.

Input: A ciphertext C and a secret key K
Output: A plaintext P
%remove diffusion
for $i \in [0, L)$ *with increment step* 2 **do**
  $R_i \leftarrow 34 \cdot C_i + 201 \cdot C_{i+1} \bmod 256$
  $R_{i+1} \leftarrow 201 \cdot C_i + 89 \cdot C_{i+1} \bmod 256$
Generate permutation S using $PRNG(K, IC)$ and compute its inverse S^{-1}.
%image descrambling
for $i \in [0, L)$ **do** $P_i \leftarrow R_{S_i^{-1}}$
return P

2 Preliminaries

Notations. In this paper, the subset $\{1, \ldots, s-1\} \in \mathbb{N}$ is denoted by $[1, s)$. The action of selecting a random element x from a sample space X is represented by $x \xleftarrow{\$} X$, while $x \leftarrow y$ indicates the assignment of value y to variable x. By H and W we denote an image's height and width.

2.1 Hosny *et al.* Image Encryption Scheme

In this section we present Hosny *et al.*'s encryption (Algorithm 1) and decryption (Algorithm 2) algorithms as described in [10]. Let W be even. Before the encryption/decryption process starts, the image of size $W \times H \times 3$ is split into three channels each of size $W \times H$. Afterwards each channel image is converted into a vector of size $L = W \cdot H$ and is processed independently. At the end, the resulting vectors are translated back into images of size $W \times H$ and then they are recombined into a final image of size $W \times H \times 3$. Please note that the PRNG has as input a secret key K and a public function f dependent on the sum of the image's pixels and L. For simplify, we refer to $f(\sum_{i=0}^{L-1} P_i, L)$ as the initial condition of the PRNG and we denote it by IC. Remark that in the processes

Algorithm 3: Computing the sum of the image's pixels.

Function *compute_sum*(C)
  for $i \in [0, L)$ *with increment step* 2 **do**
    $R_i \leftarrow 34 \cdot C_i + 201 \cdot C_{i+1} \bmod 256$
    $R_{i+1} \leftarrow 201 \cdot C_i + 89 \cdot C_{i+1} \bmod 256$
  $\Sigma = 0$
  for $i \in [0, L)$ **do** $\Sigma \leftarrow \Sigma + R_i$
  return Σ

of encryption and decryption we use the following Fibonacci Q-matrix Q^{10} and its inverse Q^{-10} modulo 256

$$Q^{10} = \begin{bmatrix} 89 & 55 \\ 55 & 34 \end{bmatrix} \text{ and } Q^{-10} = \begin{bmatrix} 34 & 201 \\ 201 & 89 \end{bmatrix}.$$

An important remark is that the sum of the image's pixels can be easily recovered from the encrypted image.[5] More precisely, since S only switches the pixels' position, we can compute the sum by removing the diffusion step. The exact method is presented in Algorithm 3.

3 Chosen Plaintext Attack

A chosen plaintext attack (CPA) is a scenario in which the attacker A briefly gains access to the encryption machine $\mathcal{O}_{enc}$ and is permitted to query it with various inputs. In this way, A generates specific plaintexts that can facilitate his attack and uses $\mathcal{O}_{enc}$ to obtain the corresponding ciphertexts. We demonstrate in this paper that Hosny *et al.*'s image encryption scheme is vulnerable to such attacks.

To help convey the intuition behind our CPA attack, we will begin by presenting a toy example before formally presenting our attack. We recommend the reader to read the examples and Algorithms 4 to 7 in parallel. Therefore, we assume that we work with an image that only has pixel values between 0 and 7. We devise an attack in the following four cases.

In the first case, we work with an image that has all its pixel values equal to i, for an $i \in [0, 8)$. If we apply a random permutation to this image, the resulting vector R has all its values equal to i. Therefore, if we receive a ciphertext C, we can easily remove the diffusion step and then check if the resulting R vector has all its values equal to an i. If this is the case, then the encrypted image has all its pixels equal to i. In the case of Hosny *et al.*'s image encryption scheme, this part of the attack is presented in Algorithm 4.

[5] Note that this is not mentioned in the original paper.

Algorithm 4: Check for images with all pixel value equal.

Function *check_equal_values*(C)
  for $i \in [0, L)$ *with increment step* 2 **do**
    $R_i \leftarrow 34 \cdot C_i + 201 \cdot C_{i+1} \bmod 256$
    $R_{i+1} \leftarrow 201 \cdot C_i + 89 \cdot C_{i+1} \bmod 256$
  for $j \in [0, 256)$ **do**
    if *all* $R_i = i$ **then return** R
  return $\perp$

For the remaining cases, we assume that we know the sum Σ of the pixels of the target image[6]. In the second case we consider images whose pixel sum is between 3 and $6 \cdot 7/2$. As an example, we consider the following target image of length $L = 6$

$$P_0 = 1, \quad P_1 = 3, \quad P_2 = 1, \quad P_3 = 4, \quad P_4 = 6, \quad P_5 = 2.$$

Then $\Sigma = 17$. We now use a greedy approach to construct two plaintexts that can aid us in computing the secret permutation S. First we check to see the interval for the sum

$$5 \cdot 6/2 = 15 \leq \Sigma < 6 \cdot 7/2 = 21$$

and we set the following parameters[7] $moth = 5$, $flea = 0$, $\Sigma' = \Sigma - 15 = 2$ and $\alpha = \Sigma' \bmod 6 = 2$. Then we generate the following intermediary attack plaintexts

$$P_0 = 1, \quad P_1 = 3, \quad P_2 = 4, \quad P_3 = 5, \quad P_4 = 0, \quad P_5 = 0$$
$$P_0 = 4, \quad P_1 = 5, \quad P_2 = 0, \quad P_3 = 0, \quad P_4 = 1, \quad P_5 = 3.$$

We can see that their sum is $13 = \Sigma - 2\alpha$. Now we add the two αs to obtain the final attack plaintexts

$$P_0 = 1, \quad P_1 = 3, \quad P_2 = 4, \quad P_3 = 5, \quad P_4 = 2, \quad P_5 = 2$$
$$P_0 = 4, \quad P_1 = 5, \quad P_2 = 2, \quad P_3 = 2, \quad P_4 = 1, \quad P_5 = 3.$$

We can easily see that both attack plaintexts have the same pixel sum and the same size as the target image. Therefore, the PRNG will generate the same random permutation S as in the case of the target plaintext. The advantage of the attack plaintexts is that we can track how the values $1, 3, 4$ and 5 are permuted by S, and thus recover S. Thus, after receiving the corresponding ciphertexts, we first remove the diffusion step and then we track the resulting

[6] It can be easily computed using Algorithm 3 when we have access to its corresponding ciphertext.

[7] please see Algorithm 6 for their exact usage

positions of the values $1, 3, 4$ and 5. Corroborated with the initial positions we can recover the secret permutation S, and hence recover the target plaintext.

In the third case we consider images whose pixel sum is between $6 \cdot 7/2$ and $6 \cdot 7/2 + (L-6) \cdot 7$. As an example, lets consider the following target image of length $L = 8$

$$P_0 = 5, \quad P_1 = 0, \quad P_2 = 1, \quad P_3 = 3, \quad P_4 = 7, \quad P_5 = 4, \quad P_6 = 6, \quad P_7 = 2.$$

Then $\Sigma = 28$. First we set our parameters $moth = 6$, $flea = 0$, $\Sigma' = \Sigma - 21 = 7$ and $\alpha = \Sigma' \bmod 7 = 0$. Then we generate the following intermediary attack plaintexts

$$\begin{array}{llllllll} P_0 = 1, & P_1 = 2, & P_2 = 3, & P_3 = 4, & P_4 = 5, & P_5 = 6, & P_6 = 0, & P_7 = 0 \\ P_0 = 3, & P_1 = 4, & P_2 = 5, & P_3 = 6, & P_4 = 0, & P_5 = 0, & P_6 = 1, & P_7 = 2. \end{array}$$

We can see that their sum is $21 = \Sigma - (moth + 1)$. Now we add the $moth + 1$ value to obtain the final attack plaintexts

$$\begin{array}{llllllll} P_0 = 1, & P_1 = 2, & P_2 = 3, & P_3 = 4, & P_4 = 5, & P_5 = 6, & P_6 = 7, & P_7 = 0 \\ P_0 = 3, & P_1 = 4, & P_2 = 5, & P_3 = 6, & P_4 = 7, & P_5 = 0, & P_6 = 1, & P_7 = 2. \end{array}$$

As in the previous case, we can track the 1 to 6 values, and thus determine the secret permutation S. Note that we do not track the $moth + 1$ value, since there can be more than one. Also, if $\alpha \neq 0$, we also remove this value from the tracked ones, since there will be two of them.

The last case is for images whose pixel sum is between $6 \cdot 7/2 + (L-6) \cdot 7$ and $7 \cdot L$. As an example, lets consider the following target image of length $L = 6$

$$P_0 = 4, \quad P_1 = 6, \quad P_2 = 5, \quad P_3 = 7, \quad P_4 = 5, \quad P_5 = 6.$$

Then $\Sigma = 33$. First we check to see the interval for the sum

$$(6-2) \cdot (7+2)/2 + (6-6+2) \cdot 7 = 32 \leq \Sigma < (5-2) \cdot (8+2)/2 + (6-5+2) \cdot 7 = 36$$

and we set $moth = 3$, $flea = 3$, $\Sigma' = \Sigma - 15 = 18$ and $\alpha = \Sigma' \bmod 7 = 4$. Then we generate the following intermediary attack plaintexts

$$\begin{array}{llllll} P_0 = 5, & P_1 = 6, & P_2 = 0, & P_3 = 0, & P_4 = 0, & P_5 = 0 \\ P_0 = 0, & P_1 = 0, & P_2 = 5, & P_3 = 6, & P_4 = 0, & P_5 = 0 \\ P_0 = 0, & P_1 = 0, & P_2 = 0, & P_3 = 0, & P_4 = 5, & P_5 = 6. \end{array}$$

Now we add the required number of 7s

$$\begin{array}{llllll} P_0 = 5, & P_1 = 6, & P_2 = 7, & P_3 = 7, & P_4 = 7, & P_5 = 0 \\ P_0 = 7, & P_1 = 7, & P_2 = 5, & P_3 = 6, & P_4 = 7, & P_5 = 0 \\ P_0 = 7, & P_1 = 7, & P_2 = 7, & P_3 = 0, & P_4 = 5, & P_5 = 6. \end{array}$$

Algorithm 5: Check sum interval.

Function $check\_interval(\Sigma, L)$
  $moth \leftarrow \perp$
  for $i \in [0, 254)$ **do**
    if $L > i$ *and* $i(i+1)/2 \leq \Sigma < (i+1)(i+2)/2$ **then**
      $moth \leftarrow i$
      $flea \leftarrow 0$
      $\Sigma' \leftarrow \Sigma - i(i+1)/2$
      $\alpha \leftarrow \Sigma' \bmod (i+1)$
      break
    else if $L \geq 254$ *and* $254 \cdot 255/2 \leq \Sigma < 254 \cdot 255/2 + (L - 254) \cdot 255$ **then**
      $moth \leftarrow 254$
      $flea \leftarrow 0$
      $\Sigma' \leftarrow \Sigma - 254 \cdot 255/2$
      $\alpha \leftarrow \Sigma' \bmod 255$
      break
    else if $L \geq 254 - i$ *and* $(254 - i)(255 + i)/2 + (L - 254 + i) \cdot 255 \leq \Sigma < (253 - i)(256 + i)/2 + (L - 253 + i) \cdot 255$ **then**
      $moth \leftarrow 253 - i$
      $flea \leftarrow i + 1$
      $\Sigma' \leftarrow \Sigma - (253 - i)(256 + i)/2$
      $\alpha \leftarrow \Sigma' \bmod 255$
      break
  return $moth, flea, \Sigma', \alpha$

We can see that their sum is $32 = \Sigma - 2\alpha + 7$. Now we add the two αs and remove a 7 to obtain the final attack plaintexts

$$\begin{aligned} &P_0 = 5, \quad P_1 = 6, \quad P_2 = 7, \quad P_3 = 7, \quad P_4 = 4, \quad P_5 = 4 \\ &P_0 = 7, \quad P_1 = 7, \quad P_2 = 5, \quad P_3 = 6, \quad P_4 = 4, \quad P_5 = 4 \\ &P_0 = 7, \quad P_1 = 7, \quad P_2 = 4, \quad P_3 = 4, \quad P_4 = 5, \quad P_5 = 6. \end{aligned}$$

As in the previous case, we can track the 5 and 6 values, and hence determine the secret permutation S. This exhausts all the possible cases that we can attack, when the sum of the target plaintext is known. In the case of Hosny *et al.*'s image encryption scheme, Algorithm 5 describes the part of the attack that checks which of the three cases we are in. The construction of the attack images is presented in Algorithm 6, while the recovery of the secret permutation is given in Algorithm 7. Once the sum of the target image is known, Algorithm 8 includes all the necessary steps needed to recover the target image. The full chosen plaintext attack is given in Algorithm 9.

To compute the complexity of our attack we consider that the operations modulo 256 have constant complexity $\mathcal{O}(1)$. Also, we consider the worst case

Algorithm 6: Compute attack image.

```
Function set_attack_image(j, L, B, moth, flea, Σ', α)
    for i ∈ [0, L) do  P_i ← 0
    if α = 0 then
        for i ∈ [jB, (j+1)B) do  P_{i mod L} ← flea + (i mod B) + 1
    else
        for i ∈ [jB, (j+1)B) do
            if flea + (i mod B) + 1 < α then  P_{i mod L} ← flea + (i mod B) + 1
            else  P_{i mod L} ← flea + (i mod B) + 2
    i ← 0, k ← 0
    while k < Σ'/(moth + 1) and i < L do
        if P_i = 0 then
            if flea = 0 then  P_i =← moth + 1
            else  P_i ← 255
            k ← k + 1
        i ← i + 1
    if α ≠ 0 then
        t ← i, k ← 0
        while k < 2 and i < L do
            if P_i = 0 then
                P_i ← α
                k ← k + 1
            i ← i + 1
        if k < 2 then
            i ← t
            while k < 2 and i ≥ 0 do
                if P_i = 255 then
                    P_i ← α
                    k ← k + 1
                i ← i − 1
return P
```

possible $B = 1$. Therefore, we obtain that the complexities of Algorithms 3 to 7 are $\mathcal{O}(4L)$, $\mathcal{O}(259L)$, $\mathcal{O}(1)$, $\mathcal{O}(L)$ and $\mathcal{O}(L)$, respectively.

In the case of Algorithm 8, we make at most L oracle queries and we have a complexity of $\mathcal{O}(2L^2)$. Since Algorithm 8's complexity dominates Algorithm 9 we obtain the same performance for the full attack. Note that Algorithm 9 recovers only a single color channel. Thus, to perform the full attack on the entire image, L queries are needed, and the overall runtime is approximately $\mathcal{O}(6L^2)$. For example, if we encrypt 2 megapixels[8] images we obtain a complexity of $\mathcal{O}(2^{44.33})$ and $2^{20.87}$ oracle queries. In the case of 12 megapixels[9], we obtain $\mathcal{O}(2^{49.61})$ and $2^{23.51}$ oracle queries.

[8] $W \times H = 1600 \times 1200$

[9] $W \times H = 4000 \times 3000$

Algorithm 7: Recover secret permutation.

Function $recover\_s(R, nb, j, L, B, moth, flea, \Sigma', \alpha)$
  if $flea = 0$ **then** $val \leftarrow moth + 1$
  else $val \leftarrow 255$
  if $\alpha = 0$ **then**
    for $i \in [0, L)$ **do**
      $flag \leftarrow$ `false`
      if $j < nb$ **then** $flag \leftarrow$ `true`
      else if $R_i \leq (L \bmod B) + flea$ **then** $flag \leftarrow$ `true`
      if $flag =$ `true` *and* $R_i \neq 0$ *and* $R_i \neq val$ **then**
        $S_i \leftarrow jB + R_i - 1 - flea \bmod L$
  else
    for $i \in [0, L)$ **do**
      $flag \leftarrow$ `false`
      if $j < nb$ **then** $flag \leftarrow$ `true`
      else if $R_i \leq (L \bmod B) + 1 + flea$ **then** $flag \leftarrow$ `true`
      if $flag =$ `true` *and* $R_i \neq 0$ *and* $R_i \neq val$ *and* $R_i \neq \alpha$ **then**
        if $R_i < \alpha$ **then** $S_i \leftarrow jB + R_i - 1 - flea \bmod L$
        else $S_i \leftarrow jB + R_i - 2 - flea \bmod L$
  return S

Algorithm 8: Chosen plaintext attack once Σ is known.

Function $cpa\_sum(\Sigma)$
  $moth, flea, \Sigma', \alpha \leftarrow check\_interval(\Sigma, L)$
  if $moth = \perp$ *or* $moth = 0$ *or* $moth = 1$ **then**
    return $\perp$
  if $\alpha = 0$ **then** $B \leftarrow moth$
  else $B \leftarrow moth - 1$
  $nb \leftarrow L/B$
  $nb_r \leftarrow \lceil L/B \rceil - nb$
  for $j \in [0, nb + nb_r)$ **do**
    $P \leftarrow set\_attack\_image(j, L, B, moth, flea, \Sigma', \alpha)$
    Send the plaintext P to the encryption oracle $\mathcal{O}_{enc}$.
    Receive the ciphertext C from the encryption oracle $\mathcal{O}_{enc}$.
    for $i \in [0, L)$ *with increment step* 2 **do**
      $R_i \leftarrow 34 \cdot C_i + 201 \cdot C_{i+1} \bmod 256$
      $R_{i+1} \leftarrow 201 \cdot C_i + 89 \cdot C_{i+1} \bmod 256$
    $S \leftarrow recover\_s(R, nb, j, L, B, moth, flea, \Sigma', \alpha)$
  return S

4 Chosen Ciphertext Attack

In contrast to a chosen plaintext attack, a chosen ciphertext attack (CCA) assumes that the attacker A briefly gains access to the decryption machine $\mathcal{O}_{dec}$. A then generates specific ciphertexts that can assist his attack and uses $\mathcal{O}_{dec}$ to

Algorithm 9: Chosen plaintext attack.

Input: A ciphertext C
Output: The targeted plaintext P
Function $cpa\_main(C)$
  $temp \leftarrow check\_equal\_values(C)$
  if $temp \neq \perp$ **then** **return** $temp$
  $\Sigma \leftarrow compute\_sum(C)$
  $S \leftarrow cpa\_sum(\Sigma)$
  Compute the inverse permutation S^{-1}.
  for $i \in [0, L)$ *with increment step* 2 **do**
    $R_i \leftarrow 34 \cdot C_i + 201 \cdot C_{i+1} \mod 256$
    $R_{i+1} \leftarrow 201 \cdot C_i + 89 \cdot C_{i+1} \mod 256$
  for $i \in [0, L)$ **do** $P_i \leftarrow R_{S_i^{-1}}$
  return P

Algorithm 10: Chosen ciphertext attack once Σ is known.

Function $cca\_sum(\Sigma)$
  $moth, flea, \Sigma', \alpha \leftarrow check\_interval(\Sigma, L)$
  if $moth = \perp$ *or* $moth = 0$ *or* $moth = 1$ **then**
    return $\perp$
  if $\alpha = 0$ **then** $B \leftarrow moth$
  else $B \leftarrow moth - 1$
  $nb \leftarrow L/B$
  $nb_r \leftarrow \lceil L/B \rceil - nb$
  for $j \in [0, nb + nb_r)$ **do**
    $R \leftarrow set\_attack\_image(j, L, B, moth, flea, \Sigma', \alpha)$
    for $i \in [0, L)$ *with increment step* 2 **do**
      $C_i \leftarrow 89 \cdot R_i + 55 \cdot R_{i+1} \mod 256$
      $C_{i+1} \leftarrow 55 \cdot R_i + 34 \cdot R_{i+1} \mod 256$
    Send the plaintext C to the decryption oracle $\mathcal{O}_{dec}$.
    Receive the plaintext P from the decryption oracle $\mathcal{O}_{dec}$.
    $S^{-1} \leftarrow recover\_s(P, nb, j, L, B, moth, flea, \Sigma', \alpha)$
  return S^{-1}

obtain the corresponding plaintexts. In this scenario, we describe an attack on Hosny *et al.*'s cryptosystem.

The main difference between the CPA and the CCA is that in the first case we have to remove the diffusion step after receiving the ciphertext for $\mathcal{O}_{enc}$ in order to get to our markers, while in the second we have to add the diffusion step before sending the ciphertexts to $\mathcal{O}_{dec}$. We present our proposed attack in Algorithm 11. Note that in the case of the CCA we do not have to compute the inverse permutation. Also, the CCA's complexity and number of oracle queries are the same as in the case of the CPA.

Algorithm 11: Chosen ciphertext attack.

Input: A ciphertext C
Output: The targeted plaintext P
Function $cpa\_main()$
  $temp \leftarrow check\_equal\_values(C)$
  if $temp \neq \perp$ **then return** $temp$
  $\Sigma \leftarrow compute\_sum(C)$
  $S^{-1} \leftarrow cca\_sum(\Sigma)$
  for $i \in [0, L)$ *with increment step* 2 **do**
    $R_i \leftarrow 34 \cdot C_i + 201 \cdot C_{i+1} \bmod 256$
    $R_{i+1} \leftarrow 201 \cdot C_i + 89 \cdot C_{i+1} \bmod 256$
  for $i \in [0, L)$ **do** $P_i \leftarrow R_{S_i^{-1}}$
  return P

5 Conclusions

The authors of [10] introduced an image encryption scheme based on a hyperchaotic system that they claimed to have a security strength of 498 bits. However, our security analysis revealed that the true security strength of Hosny *et al.*'s scheme is roughly $\mathcal{O}(2^{50})$. Additionally, our analysis shows that the attack requires at most 2^{24} oracle queries. Consequently, according to [3], the system fails to meet the necessary security strength needed to protect sensitive information.

References

1. Alanazi, A.S., Munir, N., Khan, M., Asif, M., Hussain, I.: Cryptanalysis of Novel Image Encryption Scheme Based on Multiple Chaotic Substitution Boxes. IEEE Access **9**, 93795–93802 (2021)
2. Arroyo, D., Diaz, J., Rodriguez, F.: Cryptanalysis of a One Round Chaos-Based Substitution Permutation Network. Signal Processing **93**(5), 1358–1364 (2013)
3. Barker, E.: NIST SP800-57 Recommendation for Key Management, Part 1: General. Tech. rep., NIST (2020)
4. Chen, J.x., Zhu, Z.l., Fu, C., Zhang, L.b., Zhang, Y.: An Efficient Image Encryption Scheme Using Lookup Table-Based Confusion and Diffusion. Nonlinear Dynamics **81**(3), 1151–1166 (2015)
5. Chen, J., Chen, L., Zhou, Y.: Cryptanalysis of a DNA-Based Image Encryption Scheme. Information Sciences **520**, 130–141 (2020)
6. Essaid, M., Akharraz, I., Saaidi, A., Mouhib, A.: A New Approach of Image Encryption Based on Dynamic Substitution and Diffusion Operations. In: SysCoBIoTS 2019. pp. 1–6. IEEE (2019)
7. Essaid, M., Akharraz, I., Saaidi, A., Mouhib, A.: Image Encryption Scheme Based on a New Secure Variant of Hill Cipher and 1D Chaotic Maps. Journal of Information Security and Applications **47**, 173–187 (2019)
8. Fan, H., Zhang, C., Lu, H., Li, M., Liu, Y.: Cryptanalysis of a New Chaotic Image Encryption Technique Based on Multiple Discrete Dynamical Maps. Entropy **23**(12), 1581 (2021)

9. Hosny, K.M.: Multimedia Security Using Chaotic Maps: Principles and Methodologies, vol. 884. Springer (2020)
10. Hosny, K.M., Kamal, S.T., Darwish, M.M.: Novel encryption for color images using fractional-order hyperchaotic system. Journal of Ambient Intelligence and Humanized Computing **13**(2), 973–988 (2022)
11. Hu, G., Xiao, D., Wang, Y., Li, X.: Cryptanalysis of a Chaotic Image Cipher using Latin Square-Based Confusion and Diffusion. Nonlinear Dynamics **88**(2), 1305–1316 (2017)
12. Hua, Z., Zhou, Y.: Design of Image Cipher Using Block-Based Scrambling and Image Filtering. Information sciences **396**, 97–113 (2017)
13. Huang, X., Sun, T., Li, Y., Liang, J.: A Color Image Encryption Algorithm Based on a Fractional-Order Hyperchaotic System. Entropy **17**(1), 28–38 (2014)
14. Khan, M.: A Novel Image Encryption Scheme Based on Multiple Chaotic S-Boxes. Nonlinear Dynamics **82**(1), 527–533 (2015)
15. Khan, M., Masood, F.: A Novel Chaotic Image Encryption Technique Based on Multiple Discrete Dynamical Maps. Multimedia Tools and Applications **78**(18), 26203–26222 (2019)
16. Li, M., Lu, D., Wen, W., Ren, H., Zhang, Y.: Cryptanalyzing a Color Image Encryption Scheme Based on Hybrid Hyper-Chaotic System and Cellular Automata. IEEE access **6**, 47102–47111 (2018)
17. Li, M., Lu, D., Xiang, Y., Zhang, Y., Ren, H.: Cryptanalysis and Improvement in a Chaotic Image Cipher Using Two-Round Permutation and Diffusion. Nonlinear Dynamics **96**(1), 31–47 (2019)
18. Li, M., Wang, P., Liu, Y., Fan, H.: Cryptanalysis of a Novel Bit-Level Color Image Encryption Using Improved 1D Chaotic Map. IEEE Access **7**, 145798–145806 (2019)
19. Li, M., Wang, P., Yue, Y., Liu, Y.: Cryptanalysis of a Secure Image Encryption Scheme Based on a Novel 2D Sine–Cosine Cross-Chaotic Map. Journal of Real-Time Image Processing **18**(6), 2135–2149 (2021)
20. Li, S., Zheng, X.: Cryptanalysis of a Chaotic Image Encryption Method. In: ISCAS 2002. vol. 2, pp. 708–711. IEEE (2002)
21. Liu, L., Hao, S., Lin, J., Wang, Z., Hu, X., Miao, S.: Image Block Encryption Algorithm Based on Chaotic Maps. IET Signal Processing **12**(1), 22–30 (2018)
22. Liu, Y., Qin, Z., Liao, X., Wu, J.: Cryptanalysis and Enhancement of an Image Encryption Scheme Based on a 1-D Coupled Sine Map. Nonlinear Dynamics **100**(3), 2917–2931 (2020)
23. Ma, Y., Li, C., Ou, B.: Cryptanalysis of an Image Block Encryption Algorithm Based on Chaotic Maps. Journal of Information Security and Applications **54**, 102566 (2020)
24. Matoba, O., Javidi, B.: Secure Holographic Memory by Double-Random Polarization Encryption. Applied Optics **43**(14), 2915–2919 (2004)
25. Mfungo, D.E., Fu, X., Wang, X., Xian, Y.: Enhancing Image Encryption with the Kronecker Xor Product, the Hill Cipher, and the Sigmoid Logistic Map. Applied Sciences **13**(6) (2023)
26. Mondal, B., Behera, P.K., Gangopadhyay, S.: A Secure Image Encryption Scheme Based on a Novel 2D Sine–Cosine Cross-Chaotic (SC3) Map. Journal of Real-Time Image Processing **18**(1), 1–18 (2021)
27. Muthu, J.S., Murali, P.: Review of Chaos Detection Techniques Performed on Chaotic Maps and Systems in Image Encryption. SN Computer Science **2**(5), 1–24 (2021)

28. Niyat, A.Y., Moattar, M.H., Torshiz, M.N.: Color Image Encryption Based on Hybrid Hyper-Chaotic System and Cellular Automata. Optics and Lasers in Engineering **90**, 225–237 (2017)
29. Özkaynak, F.: Brief Review on Application of Nonlinear Dynamics in Image Encryption. Nonlinear Dynamics **92**(2), 305–313 (2018)
30. Pak, C., An, K., Jang, P., Kim, J., Kim, S.: A Novel Bit-Level Color Image Encryption Using Improved 1D Chaotic Map. Multimedia Tools and Applications **78**(9), 12027–12042 (2019)
31. Pak, C., Huang, L.: A New Color Image Encryption Using Combination of the 1D Chaotic Map. Signal Processing **138**, 129–137 (2017)
32. Shafique, A., Shahid, J.: Novel Image Encryption Cryptosystem Based on Binary Bit Planes Extraction and Multiple Chaotic Maps. The European Physical Journal Plus **133**(8), 1–16 (2018)
33. Sheela, S., Suresh, K., Tandur, D.: Image Encryption Based on Modified Henon Map Using Hybrid Chaotic Shift Transform. Multimedia Tools and Applications **77**(19), 25223–25251 (2018)
34. Song, C., Qiao, Y.: A Novel Image Encryption Algorithm Based on DNA Encoding and Spatiotemporal Chaos. Entropy **17**(10), 6954–6968 (2015)
35. Teşeleanu, G.: Security Analysis of a Color Image Encryption Scheme Based on Dynamic Substitution and Diffusion Operations. In: ICISSP 2023. pp. 410–417. SCITEPRESS (2023)
36. Teşeleanu, G.: Security Analysis of an Image Encryption Based on the Kronecker Xor Product, the Hill Cipher and the Sigmoid Logistic Map. In: ICISSP 2024. pp. 467–473. SCITEPRESS (2024)
37. Teşeleanu, G.: Security Analysis of an Image Encryption Scheme Based on a New Secure Variant of Hill Cipher and 1D Chaotic Maps. In: ICISSP 2024. pp. 745–749. SCITEPRESS (2024)
38. Wang, H., Xiao, D., Chen, X., Huang, H.: Cryptanalysis and Enhancements of Image Encryption Using Combination of the 1D Chaotic Map. Signal processing **144**, 444–452 (2018)
39. Wang, L., Wu, Q., Situ, G.: Chosen-Plaintext Attack on the Double Random Polarization Encryption. Optics Express **27**(22), 32158–32167 (2019)
40. Wang, X., Teng, L., Qin, X.: A Novel Colour Image Encryption Algorithm Based on Chaos. Signal Processing **92**(4), 1101–1108 (2012)
41. Wen, H., Yu, S.: Cryptanalysis of an Image Encryption Cryptosystem Based on Binary Bit Planes Extraction and Multiple Chaotic Maps. The European Physical Journal Plus **134**(7), 1–16 (2019)
42. Wen, H., Yu, S., Lü, J.: Breaking an Image Encryption Algorithm Based on DNA Encoding and Spatiotemporal Chaos. Entropy **21**(3), 246 (2019)
43. Wen, H., Zhang, C., Huang, L., Ke, J., Xiong, D.: Security Analysis of a Color Image Encryption Algorithm Using a Fractional-Order Chaos. Entropy **23**(2), 258 (2021)
44. Wu, J., Liao, X., Yang, B.: Image Encryption Using 2D Hénon-Sine Map and DNA Approach. Signal processing **153**, 11–23 (2018)
45. Yen, J.C., Guo, J.I.: A New Chaotic Key-Based Design for Image Encryption and Decryption. In: ISCAS 2000. vol. 4, pp. 49–52. IEEE (2000)
46. Yosefnezhad Irani, B., Ayubi, P., Amani Jabalkandi, F., Yousefi Valandar, M., Jafari Barani, M.: Digital Image Scrambling Based on a New One-Dimensional Coupled Sine Map. Nonlinear Dynamics **97**(4), 2693–2721 (2019)

47. Yu, F., Gong, X., Li, H., Wang, S.: Differential Cryptanalysis of Image Cipher Using Block-Based Scrambling and Image Filtering. Information Sciences **554**, 145–156 (2021)
48. Zhou, K., Xu, M., Luo, J., Fan, H., Li, M.: Cryptanalyzing an Image Encryption Based on a Modified Henon Map Using Hybrid Chaotic Shift Transform. Digital Signal Processing **93**, 115–127 (2019)
49. Zolfaghari, B., Koshiba, T.: Chaotic Image Encryption: State-of-the-Art, Ecosystem, and Future Roadmap. Applied System Innovation **5**(3), 57 (2022)