Security Analysis of an Image Encryption Based on the Kronecker Xor Product, the Hill Cipher and the Sigmoid Logistic Map

George Teșeleanu1,2

1 Advanced Technologies Institute, 10 Dinu Vintilă, Bucharest, Romania
2 Simion Stoilow Institute of Mathematics of the Romanian Academy, 21 Calea Grivitei, Bucharest, Romania

Keywords: Image Encryption Scheme, Chaos Based Encryption, Cryptanalysis.

Abstract: In 2023, Mfungo et al. introduce an image encryption scheme that employs the Kronecker xor product, the Hill cipher and a chaotic map. Their proposal uses the chaotic map to dynamically generate two out of the three secret keys employed by their scheme. Note that both keys are dependent on the size of the original image, while the Hill key is static. Despite the authors’ assertion that their proposal offers sufficient security (149 bits) for transmitting color images over unsecured channels, we found that this is not accurate. To support our claim, we present a chosen plaintext attack that requires 2 oracle queries and has a worse case complexity of $O(2^{32})$. Note that in this case Mfungo et al.’s scheme has a complexity of $O(2^{33})$, and thus our attack is two times faster than an encryption. The reason why this attack is viable is that the two keys remain unchanged for different plaintext images of the same size, while the Hill key remains unaltered for all images.

1 INTRODUCTION

The security risks associated with digital images, particularly theft and unauthorized distribution, have been amplified by the widespread use of social media. Consequently, researchers have devoted significant attention to this issue and have developed various techniques to encrypt images. Chaotic maps have emerged as a popular choice due to their high sensitivity to initial conditions and previous states, which makes predicting their behavior difficult. As a result, several novel cryptographic algorithms based on chaos have been developed. However, many image encryption schemes based on chaotic maps suffer from critical security vulnerabilities due to inadequate security analysis and a lack of design guidelines. In fact, numerous compromised schemes exist, which are listed non-exhaustively in Table 1. For further information, please refer to (Zolfaghari and Koshiba, 2022; Muthu and Murali, 2021; Hosny, 2020; Ozkaynak, 2018).

In (Mfungo et al., 2023), the authors propose a new image encryption scheme that combines the Kronecker xor product, Hill cipher and sigmoid logistic map. More specifically, their algorithm starts by shifting the values in each row of all $4 \times 4$ image blocks using the AES shift row operation. Then, the algorithm performs a bitwise xor between the top value of each odd or even column and all other values in the corresponding even or odd column, excluding the top value. Next, the Hill Cipher encrypts each $4 \times 4$ block of the result. The resulting image is then xor-ed with a key generated using the sigmoid logistic map. To further obscure the image’s pixels, the result is transformed using the Kronecker xor product. Finally, another key generated using the sigmoid logistic map is xor-ed with the output to obtain the encrypted image. Since the sigmoid logistic map is simply used as a pseudorandom number generator (PRNG) and the scheme’s weakness is independent of the employed generator, we omit its description and simply consider the two keys as being randomly generated.

The focus of this paper is to carry out a security analysis of the Mfungo et al. scheme (Mfungo et al., 2023). We describe a chosen plaintext attack, which would allow an attacker to decrypt all images of a specific size. To execute such an attack, the adversary would need to access the ciphertexts of 2 chosen plaintexts. Once the attacker has this information in his possession, he can easily extract the secret keys. According to the authors, the largest image size that they were able to handle with their available computational resources was limited to $256 \times 256$ pixels. Thus, in this case, the key recovery and the encryption

---

Teșeleanu, G. Security Analysis of an Image Encryption Based on the Kronecker Xor Product, the Hill Cipher and the Sigmoid Logistic Map. DOI: 10.5220/0012307800003648
Paper published under CC license (CC BY-NC-ND 4.0) In Proceedings of the 10th International Conference on Information Systems Security and Privacy (ICISSP 2024), pages 467-473
ISBN: 978-989-758-683-5; ISSN: 2184-4356
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Ltd.
Table 1: Broken chaos based image encryption algorithms.

|--------|-----------|---------------------|-------------------------|-----------------|---------------------|---------------------|-------------------|---------------------|------------------|------------------|------------------|------------------|-----------------|

Structure of the Paper. We provide the necessary preliminaries in Section 2. An alternative description of Mfungo et al.’s scheme is outlined in Section 3. In Section 4 we show how an attacker can recover all three secret keys in a chosen plaintext scenario. We conclude in Section 5.

2 PRELIMINARIES

Notations. In this paper, the subset \{1, \ldots, s−1\} \subseteq \mathbb{N} is denoted by [1,s]. The action of selecting a random element \( x \) from a sample space \( X \) is represented by \( x \leftarrow X \), 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 Mfungo et al. Image Encryption Scheme

In this section we present Mfungo et al.’s encryption (Algorithm 2) and decryption (Algorithm 3) algorithms as described in (Mfungo et al., 2023). Note that \( W \) and \( H \) must be divisible by 4.

The first step of the encryption process consists in breaking the image in \( 4 \times 4 \) blocks and then circular shifting row \( i \) of each block to the left by \( i \) positions. The exact function is provided in Algorithm 1 as shift rows. Note that the function takes as input one of the following matrices

\[
\text{shift} \leftarrow \begin{bmatrix}
0 & 1 & 2 & 3 \\
1 & 2 & 3 & 0 \\
2 & 3 & 0 & 1 \\
3 & 0 & 1 & 2
\end{bmatrix}
\]

or

\[
\text{inv\_shift} \leftarrow \begin{bmatrix}
0 & 1 & 2 & 3 \\
3 & 0 & 1 & 2 \\
2 & 3 & 0 & 1 \\
1 & 2 & 3 & 0
\end{bmatrix}
\]

one for encryption and the other one for decryption. Then the top values of the resulting matrix are preserved, while all values in even columns\(^1\) are xor-ed with the top value of the previous odd column. In the case of odd columns, the values are xor-ed with the top value of the next column, except their top value. The corresponding function is xor between pairwise columns from Algorithm 1. Using a secret \( 4 \times 4 \) matrix \( h \), each row of each \( 4 \times 4 \) block is multiplied with \( h \). Hill encryption is presented in Algorithm 1, Hill. The resulting image is then xor-ed with \( k^{(1)} \). Another diffusion layer is then added, i.e. the rows are moved down with 3 positions (see Algorithm 1, shift columns). The Kronecker xor transformation is then applied. More precisely, the authors apply the Kronecker product between the image and itself, with the following modifications: the product between two elements from two distinct positions is replaced by xor, while the ones from the same position remain unaltered. The pseudo-code is given in the Kronecker xor transformation function from Algorithm 1. Finally, we perform a final xor with the second key \( k^{(2)} \).

To decrypt we simply perform all the inverse operations in reverse order. Note that when reversing the Kronecker xor transformation, we should recover the matrices from all \( W \times H \) block and take a majority vote for each byte. This is done in order to provide protection against data loss and noise alteration. Basically, the compression of the Kronecker xor transformation is used as a repetition code. Since, we consider the ideal case when oracle answers are relayed unaltered, we simply recover the image from the first \( W \times H \) block.

3 A NEW LOOK AT MFUNGO et al.’S SCHEME

In this section we present an alternative description of Mfungo et al.’s scheme. More precisely, we show

\(^1\)except their top values
Algorithm 1: Helper Functions.

```plaintext
Function shift_rows(P, shift)
  for i ∈ [0, W) and at each step increment i with 4 do
    for j ∈ [0, H) do
      for k ∈ [0, 4) do
        index ← i + shift k, j mod 4
        Q[i+k, j] ← P[index, j]
      return Q

Function xor_between_pairwise_columns(P)
  for i ∈ [0, W) do R[i, 0] ← P[i, 0]
  for i ∈ [0, W) and at each step increment i with 2 do
    for j ∈ [1, H) do
      R[i, j] ← P[i, j] ⊕ P[i+1, 0]
      R[i+1, j] ← P[i+1, j] ⊕ P[i, 0]
  return R

Function Hill(P, h)
  for i ∈ [0, W) and at each step increment i with 4 do
    for j ∈ [0, H) do
      S[i, j] ← P[i, j] h_0, 0 + P[i+1, j] h_0, 1 + P[i+2, j] h_0, 2 + P[i+3, j] h_0, 3 mod 256
      S[i+1, j] ← P[i, j] h_1, 0 + P[i+1, j] h_1, 1 + P[i+2, j] h_1, 2 + P[i+3, j] h_1, 3 mod 256
      S[i+2, j] ← P[i, j] h_2, 0 + P[i+1, j] h_2, 1 + P[i+2, j] h_2, 2 + P[i+3, j] h_2, 3 mod 256
      S[i+3, j] ← P[i, j] h_3, 0 + P[i+1, j] h_3, 1 + P[i+2, j] h_3, 2 + P[i+3, j] h_3, 3 mod 256
  return S

Function shift_columns(P, n)
  for i ∈ [0, W) and j ∈ [0, H) do
    T[i, j] ← P[i, j+n mod H]
  return T

Function Kronecker_xor_transformation(P)
  for i ∈ [0, W) and j ∈ [0, H) do
    for k ∈ [0, W) and ℓ ∈ [0, H) do
      if i = k and j = ℓ then
        U[i, W+k, j+H+ℓ] ← P[i, j]
      else
        U[i, W+k, j+H+ℓ] ← P[i, j] ⊕ P[k, ℓ]
  return U

Function compress_Kronecker_xor_transformation(P)
  for i ∈ [0, W) and j ∈ [0, H) do
    if i = 0 and j = 0 then
      T[i, j] ← P[i, j]
    else
      T[i, j] ← P[i, j] ⊕ P[0, 0]
  return T
```

How to combine $k^{(1)}$ and $k^{(2)}$ into a single key $k^{(3)}$. The alternative encryption and decryption algorithms are provided in Algorithms 4 and 5.

We further show how we derived the equivalent description of lines 4-7, Algorithm 2. After the shift_row operation we obtain

$$T[i, j] ← S[i, j] + k^{(1)}_{i, j} mod H.$$  

Applying the Kronecker transformation we get

$$U[i, W+k, j+H+ℓ] ← T[i, j] = S[i, j] + k^{(1)}_{i, j} mod H + k^{(1)}_{i, j} mod H.$$  

When $i = k$ and $j = ℓ$ and

$$U[i, W+k, j+H+ℓ] ← T[i, j] + T[k, ℓ]$$

$$= S[i, j] + k^{(1)}_{i, j} mod H + k^{(1)}_{i, j} + k^{(1)}_{i, j} mod H$$

$$= (S[i, j] + k^{(1)}_{i, j} mod H + k^{(1)}_{i, j} + k^{(1)}_{i, j} mod H),$$
Algorithm 2: Encryption algorithm.

Input: A plaintext $P$, two secret keys $k^{(1)}$ and $k^{(2)}$, and a secret matrix $h$

Output: A ciphertext $C$

1. $Q \leftarrow \text{shift\_rows}(P, \text{shift})$
2. $R \leftarrow \text{xor\_between\_pairwise\_columns}(Q)$
3. $S \leftarrow \text{Hill}(R,h)$
4. for $i \in [0,W]$ and $j \in [0,H)$ do
   4.1. $S_{i,j} \leftarrow S_{i,j} \oplus k^{(1)}_{i,j}$
5. $T \leftarrow \text{shift\_columns}(S,3)$
6. $U \leftarrow \text{Kronecker\_xor\_transformation}(T)$
7. for $i \in [0,W^2]$ and $j \in [0,H^2)$ do
   7.1. $C_{i,j} \leftarrow U_{i,j} \oplus k^{(2)}_{i,j}$
8. return $C$

Algorithm 3: Decryption algorithm.

Input: A ciphertext $C$, two secret keys $k^{(1)}$ and $k^{(2)}$, and a secret matrix $h$

Output: A plaintext $P$

1. for $i \in [0,W^2]$ and $j \in [0,H^2)$ do
   1.1. $U_{i,j} \leftarrow C_{i,j} \oplus k^{(1)}_{i,j}$
2. $T \leftarrow \text{compress\_Kronecker\_xor\_transformation}(U)$
3. $S \leftarrow \text{shift\_columns}(T,-3)$
4. for $i \in [0,W]$ and $j \in [0,H)$ do
   4.1. $S_{i,j} \leftarrow S_{i,j} \oplus k^{(1)}_{i,j}$
5. $R \leftarrow \text{Hill}(S,h^{-1})$
6. $Q \leftarrow \text{xor\_between\_pairwise\_columns}(R)$
7. $P \leftarrow \text{shift\_rows}(Q, \text{inv\_shift})$
8. return $P$

otherwise. Finally, we get

$C_{i,k+3j+3 \mod H} \oplus k^{(2)}_{i,k+3j+3 \mod H}$

if $i = k$ and $j = \ell$

$K_{r}(sc(S,3)) = S_{i,j+3 \mod H}$

otherwise. Therefore, if we define $k^{(3)}$ as follows

$k^{(3)}_{i,k+3j+3 \mod H} = k^{(1)}_{i,k+3j+3 \mod H} \oplus k^{(2)}_{i,k+3j+3 \mod H}$

Algorithm 4: Equivalent encryption algorithm.

Input: A plaintext $P$, a secret key $k^{(3)}$, and a secret matrix $h$

Output: A ciphertext $C$

1. $Q \leftarrow \text{shift\_rows}(P, \text{shift})$
2. $R \leftarrow \text{xor\_between\_pairwise\_columns}(Q)$
3. $S \leftarrow \text{Hill}(R,h)$
4. $T \leftarrow \text{shift\_columns}(S,3)$
5. $U \leftarrow \text{Kronecker\_xor\_transformation}(T)$
6. for $i \in [0,W^2]$ and $j \in [0,H^2)$ do
   6.1. $C_{i,j} \leftarrow U_{i,j} \oplus k^{(3)}_{i,j}$
7. return $C$

Algorithm 5: Equivalent decryption algorithm.

Input: A ciphertext $C$, a secret key $k^{(3)}$, and a secret matrix $h$

Output: A plaintext $P$

1. for $i \in [0,W^2]$ and $j \in [0,H^2)$ do
   1.1. $U_{i,j} \leftarrow C_{i,j} \oplus k^{(3)}_{i,j}$
2. $T \leftarrow \text{compress\_Kronecker\_xor\_transformation}(U)$
3. $S \leftarrow \text{shift\_columns}(T,-3)$
4. $R \leftarrow \text{Hill}(S,h^{-1})$
5. $Q \leftarrow \text{xor\_between\_pairwise\_columns}(R)$
6. $P \leftarrow \text{shift\_rows}(Q, \text{inv\_shift})$
7. return $P$

4 CHOSEN PLAINTEXT ATTACK

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

In the first step of our attack we aim to retrieve \(k^{(3)}\). This can be easily done if we encrypt an image \(I_0\) with all its pixels set to 0. By setting all the pixels to 0, after passing the image through lines 1-5, Algorithm 4 we end up with the same image \(I_0\). Therefore, we retrieve the key from \(k^{(3)} = C_{i,j}\).

Let
\[
P_{0,4,0,4} = \begin{bmatrix} P_{0,0} & P_{1,0} & P_{2,0} & P_{3,0} \\ P_{0,1} & P_{1,1} & P_{2,1} & P_{3,1} \\ P_{0,2} & P_{1,2} & P_{2,2} & P_{3,2} \\ P_{0,3} & P_{1,3} & P_{2,3} & P_{3,3} \end{bmatrix}.
\]

Now we aim to find the secret matrix \(h\). Hence, we create an image \(I_h\) such that
\[
P_{0,4,0,4} \leftarrow \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}
\]
and the remaining pixels are set to 0. Since we are only interested in the first \(4 \times 4\) block, we will only study its evolution. Thus, after the shift_row and xor_between_pairwise_columns operations we obtain
\[
Q_{0,4,0,4} \leftarrow \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}
\]
and
\[
R_{0,4,0,4} \leftarrow \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.
\]

Therefore, we obtain that
\[
S_{0,4,0,4} \leftarrow \begin{bmatrix} h_{0,0} & h_{1,0} & h_{2,0} & h_{3,0} \\ h_{0,1} & h_{1,1} & h_{2,1} & h_{3,1} \\ h_{0,2} & h_{1,2} & h_{2,2} & h_{3,2} \\ h_{0,3} & h_{1,3} & h_{2,3} & h_{3,3} \end{bmatrix}.
\]
is the transpose of \(h\). Since we already know \(k^{(3)}\) and the remaining operations are easily reversible, it results that we can retrieve \(h\) from the ciphertext corresponding to \(I_h\). The formal description of our CPA attack is provided in Algorithm 6.

The complexity of Algorithm 6 is \(O(H^2W^2 + 2HW)\) and we need 2 oracle queries. Note that Mfungo et al.’s encryption scheme has a complexity of \(O(2H^2W^2 + 8HW)\) and according to the authors the maximum image size that they experimented on is \(H = W = 256\). Thus, in this case, our attack has a complexity of \(O(2^{32})\), while Mfungo et al.’s scheme has one of \(O(2^{23})\). Remark that if we already recovered \(h\) in a previous iteration, we only need to run lines 2-5, Algorithm 6. Thus, the complexity becomes \(O(1)\) and we need 1 oracle query.

5 CONCLUSIONS

In (Mfungo et al., 2023), the authors presented a scheme for encrypting images using a combination of the Kronecker xor product, Hill cipher, and a chaotic map. They claimed that their proposal provided a security strength of 149 bits. However, our analysis of the scheme’s security has revealed that its actual strength is only \(O(2^{32})\) in the worst-case scenario. Note that the attack only requires two oracle queries. Consequently, the proposed cryptosystem fails to meet the necessary security strength needed to protect confidential information.

REFERENCES

tion Scheme Based on Multiple Chaotic Substitution Boxes. IEEE Access, 9:93795–93802.


