Development of Standard Hill Technology for Image Encryption over a 256-element Body

Abdellatif JarJar

Abstract

This document traces the new technologies development based on a deep classical Hill method improvement. Based on the chaos, this improvement begins with the 256 element body construction, which is to replace the classic ring used by all encryption systems. In order to facilitate the application of algebraic operators on the pixels, two substitution tables will be created, the first represents the discrete exponential, while the second represents the discrete exponential. At the same time, a large invertible matrix whose structure will be explained in detail will be the subject of the advanced classical Hill technique improvement. To eliminate any linearity, this matrix will be accompanied by dynamic vectors to install an affine transformation. The simulation of a large number of images of different sizes and formats checked by our algorithm ensures the robustness of our method.

Key Words: Chaotic map, 256-Bady, Discrete exponential, Discrete logarithm

I. INTRODUCTION

The operation of digital data and its transmission through the network are uncertain operations and are vulnerable to various attacks, cryptography is becoming the most effective means for data security. In the literature, almost all classical methods are still vulnerable to statistical and differential attacks.

2.1. Conventional Hill technique

This technique, discovered by HILL [1],[2] in 1929, was only applicable to the text. It is based on two main steps. The first step is to divide the message to be encrypted into natural number characters, and the second step is to be replaced in a carefully selected ring as usually Z/26Z or Z/256Z. The difficulty of constructing a large invertible matrix prompts researcher to only use a matrix with n ≥ 4. Equation 1 fully describes this standard technique

\[ \{ C_i = KC_i \} \quad \forall \ i \geq 1. \quad (1) \]

With \( C_i \) is the clear block, \( C'_i \) is the encrypted block, and \( K \) is the encryption key. Each \( C'_i \) block is translated to an element of a well selected \( G_2^m \) ring. In such a case, the encryption matrix \( K \) is assigned coefficients in the same ring \( G_2^m \). Due to the high degree of linearity, this technique is always exposed to selected plain text and known statistical attacks. On the other hand, the high correlation between adjacent pixels and diagonal pixels of the image makes this technique unsuitable for image encryption. Finally, there is no chain in the encryption system, so this method is vulnerable to differential attacks. The decryption operation is described by next equation.

\[ \{ C_i = K^{-1}C'_i \} \quad \forall \ i \geq 1. \quad (2) \]

2.2. Hill’s classic method survey

Several successive developments in the methodology have taken place over time, but all using a reference ring such as \( G_{256} \) or \( G_{26} \) which significantly reduces the number of invertible matrixes and increases the risk of brutal attacks.

A first improvement [3-4-5] consists in modifying at each iteration; the encryption matrix by a secret permutation
(h) and fixed in the ring (G_{256}), on the rows or on the columns. This improvement is given by the equation 3

\[
\begin{align*}
C_i' &= K_i C_i \\
C_i' &= h(K_{i-1}) C_i & \forall i \geq 2,
\end{align*}
\]

where \( h(K_{i-1}) \) is the transform of the matrix \( K_{i-1} \) by fixed permutation (h). Other improvements accompany the static encryption matrix of a translation vector (T), and this to overcome the problem of uniform blocks [6 – 7] and null blocks, still others modify the translation vector at each iteration by a linear transformation provided by a fixed matrix (Q) of size \((n, n)\), not necessarily inversible. This method is described by equation 4.

\[
\begin{align*}
C_i' &= K_i \oplus T_i, \\
T_i &= QT_{i-1} With \ i \geq 2, \\
C_{i+1}' &= K_i \oplus T_i & \forall i \geq 1.
\end{align*}
\]

Moreover, taking advantage of the properties of the involuntary matrices, a new technique for constructing invertible matrices of random size will be determined.

## II. THE PROPOSED METHOD

This new technology that works at the pixel level is explained in the following aspects, and its value is regarded as an element of the built company.

![Fig. 1. Steps of realization of the algorithm.](image)

Finally, a detailed analysis of our methodology performance will be discussed and compared with other reference systems.

### Step 1: Chaotic sequences Development

Our algorithm uses two of the most famous and widely used chaotic maps in cryptography.

#### (1) The Logistics’ Map

Due to its high sensitivity to initial conditions, chaos is largely utilized symmetric cryptography for the construction of cipher keys [14], [15], [16].

\[
\begin{align*}
\mu & \in [3, 75] & \mu & \in [3, 75] \\
0.5 & \leq u_0 & 4 & , u_n+1 = \mu u_n (1 - u_n).
\end{align*}
\]
(2) HENON’S Map

Henon’s chaotic two-dimensional map was first discovered in 1978. It is described by equation below.

\[
\begin{align*}
\begin{cases}
v_0, w_0 & = 0.3, b \in [1.07-1.4] \\
v_n+1 & = 1 + w_n - av_n^2 \\
w_n+1 & = bv_n 
\end{cases}
\]
\]

(6)

We can convert the two-dimensional map expression to a one-dimensional map that is easy to implement in the encryption system. This formula is described by next equation.

\[
\begin{align*}
\begin{cases}
v_0, v_1 & \text{in } [0, 1], a = 0.3, b \in [1.07-1.4] \\
v_n+2 & = 1 - av_n^2 + bv_n 
\end{cases}
\]
\]

(7)

(3) Chaotic used vector design

Our work requires the construction of three chaotic vectors (CL), (KR) and (KL) with a coefficient of (G_{256}), and the binary (CR) vector will be regarded as the control vector. This construct is seen by the following algorithm:

\[
\text{Alg2} = \begin{cases}
\text{for } i = 1 \text{ to } 3nm \\
\\
\text{CL}(i) = \text{mod} \left( E \left( \frac{u(i) + 2v(i)}{3} \right) \times 10^{11,254} + 1 \right) \\
\text{KL}(i) = \text{mod} \left( E \left( \frac{w(i) + u(i) + v(i)}{3} \right) \times 10^{11,253} + 2 \right) \\
\text{KR}(i) = \text{mod} \left( E \left( \frac{KL(i) + CL(i)}{2} \right) \right) \\
\text{if } u(i) \geq \frac{v(i) + w(i)}{2} \text{ then } \\
\text{CR}(i) = 0 \text{ else } CR(i) = 1 \\
\text{end if} \\
\text{Next } i
\end{cases}
\]

We note that

\[
\forall i \in [1, 3nm] \begin{cases}
\text{CL}(i) \neq 0 \\
\text{KL}(i) \neq 0 \\
\text{KR}(i) \neq 0
\end{cases}
\]

These elements are all non-zero; as a result, they are invertible within the built body.

Step 2: F_{256} Body Construction

The most important step is to create an entity with 256 elements, which will replace the classical (G_{256}) in the calculation.

(1) Mathematical overview

For it

\[
F_{256} = \left\{ h(x) \in \mathbb{F}_d[x] : h \leq 7 \right\}
\]

Let \( p(x) \) eighth-order polynomial and irreducible in \( \mathbb{F}[x] \). We define two internal composition laws described by the following formula on such a set.

\[
\begin{align*}
\text{First Internal Composition Law} \\
h(x) \oplus k(x) = (h(x) + k(x)) \text{ modulo } 2, \\
\text{Second Internal Composition Law} \\
h(x) \otimes k(x) = (h(x)k(x)) \text{ modulo } p(x).
\end{align*}
\]

(8)

It is easy to prove that these two internal composition laws provide the (\( F_{256}, \oplus, \otimes \)) with a commutative finite body structure with 256 elements.

(2) F_{256} Elements Representation

Any element of the \( F_{256} \) body can be represented in five different forms:

a) Polynomial Writing

We know that

\[
F_{256} = \left\{ h(x) \in \mathbb{F}_d[x] : h \leq 7 \right\}.
\]

(9)

Consequently, any element can be written in the form of a polynomial of degree at most equal to 7 with (\( G_2 \)) components. For example:

\[
h(x) = x^6 + x^4 + x^3 + x^2 + 1, \quad q(x) = x^2 + x
\]

b) Vector Writing

Any element of the body \( F_{256} \) can be represented as a size vector (1, 8) with a coefficient in (\( G_2 \))

\[
\begin{array}{c|c}
\text{Polynomial Writing} & \text{Vector Writing} \\
\hline
h(x) = x^6 + x^4 + x^3 + x^2 + 1 & (0,1,0,1,1,0,1,1) \\
q(x) = x^2 + x & (0,0,0,0,1,1,0)
\end{array}
\]

c) Binary Writing
By simple conversion from vector writing to binary writing, any element of such a subject can be written in binary form.

<table>
<thead>
<tr>
<th>Polynomial Writing</th>
<th>Vector Writing</th>
<th>Binary Writing</th>
</tr>
</thead>
<tbody>
<tr>
<td>( h(x) = x^5 + x^4 + x^3 + x^2 + 1 )</td>
<td>((0,1,0,1,1,1,0))</td>
<td>01011000</td>
</tr>
<tr>
<td>( g(x) = x^2 + x )</td>
<td>((0,0,0,0,1,1,0))</td>
<td>00001100</td>
</tr>
</tbody>
</table>

**d) Integers Writing**

All body elements are displayed in 8 bits, consequently their value is located between 0 and 255. So, we have

\[ F_{256} = \{0,1,2,3,\ldots,255\} \]

<table>
<thead>
<tr>
<th>Polynomial Writing</th>
<th>Vector Writing</th>
<th>Binary Writing</th>
</tr>
</thead>
<tbody>
<tr>
<td>( h(x) = x^4 + x^3 + x + 1 )</td>
<td>((0,1,0,1,1,1,0))</td>
<td>01011000</td>
</tr>
<tr>
<td>( g(x) = x^2 + x )</td>
<td>((0,0,0,0,1,1,0))</td>
<td>00001100</td>
</tr>
</tbody>
</table>

This notation can be extended to the coefficient matrix in \( F_{256} \). For example:

\[
M = \begin{pmatrix} x^2 + x + 1 & x \\ x^3 + x^2 & x^6 + x^4 + 1 \end{pmatrix} \Rightarrow M = \begin{pmatrix} 7 & 2 \\ 12 & 81 \end{pmatrix}
\]

**e) Discrete Exponential Writing**

Note that, \((F_{256}, \bigotimes, \bigoplus)\) is a finite body, consequently, the set \((F_{256}, \bigotimes)\) is a cyclic group. As a result, it is generated by a single element \( g(x) \) closely related to the constructor polynomial \( p(x) \). This can be illustrated by the following formula:

\[
F_{256}^* = \{ h(x) \in F[x] \text{ that } d^\text{th} \leq 7, \text{ and } h(x) \neq 0 \} = F_{256} - \{0\}
\]

\[
\{ \forall h \in F_{256}^* \exists ! i \in [0, 254] : h(x) = g(x)^i \mod (p(x)), \}

by convention \( g(x)^{255} = 0 \), \hspace{1cm} \text{(9)}

where \( i \) is called the exponential notation of the polynomial \( h(x) \).

*We note that* \( \text{Exp}(i) = h(x) \)

We confirm that the change of the generator \( g(x) \) will lead to a fundamental change in the sign of the exponent, which will result in serious distortion of the entire encryption system.

**f) Discrete logarithm Writing**

Function \((\text{Exp})\) is bijective, and its inverse function is the function defined by the following formula \((\text{Log})\):

\[
\text{AI3} \begin{cases} 
\text{for } i = 0 \text{ to } 254 \\ 
\text{Log}(\text{Exp}(i)) = i \\ 
\text{Next } i \\
\text{by convention } \text{Log}(0) = 255
\end{cases}
\]

This rating will greatly facilitate algebraic calculations.

**3) Algebraic operations over \( F_{256} \)**

The two algebraic operations will be defined from the two tables constructed to facilitate the calculations.

**a) The multiplication**

To facilitate the multiplication of \( F_{256} \) elements, it is recommended to use the two notations \((\text{Exp})\) and \((\text{Log})\).

This technique is clarified by the equation below:

\[
g(x)^i \bigotimes g(x)^j = \begin{cases} 
0 & \text{if } i = 255 \text{ or } j = 255, \\
\text{else} & g(x)^{\text{mod}(i+j, 255)}.
\end{cases} \hspace{1cm} \text{(10)}
\]

So, we can deduce the equation below

\[
\{ x_i \bigotimes x_j = \begin{cases} 
0 & \text{if } x_i = 0 \text{ or } x_j = 0 \\
\text{Exp(mod(Log(x_i) + Log(x_j), 256))}
\end{cases} \} \hspace{1cm} \text{(11)}
\]

Please note that multiplication is closely related to the choice of generator \( g(x) \).

**b) \( F_{256} \) Inverse of elements**

The calculation of the inverse of the elements of the basic set is very important and very useful in the decryption process.

**i. Inverse for addition**

We know that

\[
\forall x \in F_{256} \hspace{0.5cm} x \bigoplus x = 0. \hspace{1cm} \text{(12)}
\]

**ii. Inverse for multiplication**

\[
\forall x \in F_{256} \hspace{0.5cm} x^{-1} = \text{Exp}(255 - \text{Log}(x)). \hspace{1cm} \text{(13)}
\]

Note that any non-zero element is invertible in \( F_{256} \).

**4) Matrix analysis in body \( F_{256} \)**

Every matrix used in this system are all in coefficients in \( F_{256} \).

**a) Image of a vector by a matrix \((3,3)\)**

The multiplication of a size matrix \((3,3)\) and a size vector \((1,3)\) is determined by the following formula below:
\[
\begin{pmatrix}
a & b & c \\
d & e & f \\
g & h & i
\end{pmatrix} \otimes \begin{pmatrix}
a \\
d \\
g
\end{pmatrix} = \begin{pmatrix}
a \otimes a \otimes b \otimes \beta \oplus c \otimes \delta \\
d \otimes a \otimes e \otimes \beta \oplus f \otimes \delta \\
g \otimes a \otimes h \otimes \beta \oplus i \otimes \delta
\end{pmatrix}
\]
\]
(14)

b) Second order determinant
The determinant of a second order matrix is defined by the equation below.

\[
\begin{vmatrix}
a & b \\
c & d
\end{vmatrix} = a \otimes d \oplus c \otimes b,
\]
(15)

So the matrix A of size (l,l) is invertible \( \iff \det(A) \neq 0 \).

This greatly increases the number of invertible matrices. We know that the number of invertible size matrix \((p, p)\) in \( F_{256} \) is:

\[
\delta = \prod_{i=1}^{p-1} (2^p - 2^i) \gg 2^{100}.
\]
(16)

This proves that the brutal attacks on the matrices in \( F_{256} \) of higher order are remote.

III. INSTALL THE NEW CRYPTO SYSTEM

Throughout the document, the pixel intensity values of the color image pixels will be considered as elements of \( F_{256} \). Our method is articulated on the following points.

(1) Original image Vectorization

After extraction of the three color channels \((RGB)\) and their conversion into vectors \((Vr),(Vg),(Vb)\), a cohabitation is carried to form the vector \(X(x_1,x_2,...,x_{3nm})\). To apply Hill's new method, the vector \((X)\) must be cut into blocks of the size of \((r_h)\) calculated from the chaotic map and the original image size.

(2) \((r_h)\) value Determination

\[
r_h = \left( \text{mod} \left( E \left( 10^{30} \sum_{i=2}^{nm} \frac{u(i) + \sup(u(i), v(i))}{2} \right) , 6 \right) + 15 \right).
\]
(17)

So, we can conduct as:

\[
15 \leq r_h \leq 20.
\]
(18)

(3) Size vector image Adaptation

In order to implement the new technology, we need to cut the image vector \((X)\) into large and small blocks \((2r_h)\). This operation follows the following formula:

\[
\begin{cases}
\text{let } 3nm \equiv s [2r_h] \\
l = 3nm - s \\
t = \frac{l}{2r_h}
\end{cases}
\]
(19)

The vector \((X)\) must be imputed by \((s)\) pixels by the following method:

\[
\begin{aligned}
\text{Amputated pixel storage} & \quad \text{for } i = 1 \text{ to } s \\
& \quad \text{if } CR(i+l) = 0 \text{ then} \\
& \quad \quad XD(i) = X(i+l) \oplus CL(i+l) \\
& \quad \text{else} \\
& \quad \quad XD(i) = X(i+l) \oplus KL(i+l) \end{aligned}
\]
end if

\[
\begin{aligned}
\text{Amputated pixel storage} & \quad \text{for } i = 1 \text{ to } s \\
& \quad \text{if } CR(i+l) = 0 \text{ then} \\
& \quad \quad XD(i) = X(i+l) \oplus CL(i+l) \\
& \quad \text{else} \\
& \quad \quad XD(i) = X(i+l) \oplus KL(i+l) \end{aligned}
\]
end if

We noticed that this decomposition is completely controlled by the decision vector \((CR)\).

(4) \((2r_h)\)-Bit Blocks Decomposition

In parallel, convert two chaotic vectors \((KR)\) and \((KL)\) into matrices \((MR)\) and \((ML)\) of size \((t, 2r_h)\) following Fig. 2. After adjusting the size of the image vector, convert the latter to a matrix \((MC)\) of size \((t, 2r_h)\) as shown in Fig. 3.

Fig. 2. Converting two chaotic vectors.
4.4.1 Involutive matrix

A is an involutive matrix if and only if we have
\[
A = \begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix} \text{ of size } (r_h, r_h) \text{ with } (r) \in G_{256}^2
\]
We got
\[
A^2 = \begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix} \begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix} = \begin{pmatrix} A_1^2 \oplus A_3A_2 & A_2A_1 \oplus A_4A_2 \\ A_1A_3 \oplus A_3A_4 & A_2^2 \oplus A_2A_3 \end{pmatrix} = (I \ 0) \begin{pmatrix} 0 & I \\ 0 & I \end{pmatrix}. \tag{21}
\]

4.1. Encryption matrix construction

According to our technical steps, it will be easier to construct a large invertible matrix based on involute blocks and non-empty eigenvalue matrices

4.4.1 Involutive matrix

A is an involutive matrix if and only if we have
\[
A = \begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix} \text{ of size } (r_h, r_h) \text{ with } (r) \in G_{256}^2
\]
We got
\[
A^2 = \begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix} \begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix} = \begin{pmatrix} A_1^2 \oplus A_3A_2 & A_2A_1 \oplus A_4A_2 \\ A_1A_3 \oplus A_3A_4 & A_2^2 \oplus A_2A_3 \end{pmatrix} = (I \ 0) \begin{pmatrix} 0 & I \\ 0 & I \end{pmatrix}. \tag{21}
\]

IV. NEW IMPROVEMENT CLASSICAL HILL TECHNIQUE

The difficulty of reversing large matrices forces researchers to use matrices with sizes generally less than 5. However, due to linearity, classical HILL methods are still subject to statistical attacks. Our algorithm overcomes this anomaly by constructing an arbitrarily large invertible matrix, accompanied by chaotic vectors generated from the chaotic map used under binary chaotic vector control.

First, the initialization (VI) design vector of size \((1, 2r_h)\) must be recalculated to change the value of the starting block. Ultimately, the (VI) value is provided by the next algorithm.

\[
\text{Alg9} \begin{cases} 
\text{for } i = 2 \text{ to } t \\
\text{for } j = 2 \text{ to } 2r_h \\
V_I(i) = 0 \\
V_I(i) = V_I(i) \oplus MC(i, j) \\
\end{cases}
\]

Next \(j, i\)

To surpass the uniform image problem (Black, White ...) the setup value (VI) will be combined with the chaotic vector (TT) specified by the following algorithm.

\[
\text{Alg10} \begin{cases} 
\text{for } i = 1 \text{ to } 2r_h \\
V_I(i) = V_I(i) \oplus CL(i) \\
\end{cases}
\]

Next \(i\)

The value calculated from the clear image and the chaotic map, will only be used to change the value of the start pixel and restart the encryption process.

\[
\text{Alg11} \begin{cases} 
\text{for } i = 1 \text{ to } 2r_h \\
CM(1, i) = CM(1, i) \oplus V_I(i) \\
\end{cases}
\]

Next \(i\)
Or we can take as:

\[
\begin{align*}
A_2 &= k(I \oplus A_1), \\
A_3 &= k^{-1}(I \oplus A_1), \\
A_4 &= -A_1 = A_1^r,
\end{align*}
\]

So:

\[
A = \begin{pmatrix}
A_1 & k(I \oplus A_1) \\
-k^{-1}(I \oplus A_1) & A_1^r
\end{pmatrix}.
\]

(25)

4.1.2. D matrix building

The eigenvalue \((D)\) matrix has the form as:

\[
D = \begin{pmatrix}
e_1 & \cdots & 0 \\
\vdots & \ddots & \vdots \\
0 & \cdots & e_{2^r n}
\end{pmatrix} \quad \forall \ i \in [1, r], e_i \in F_{256}^*.
\]

(26)

The number of matrices \((D)\) is much higher than \(2^{16 r h}\).

Finally, the new Hill matrix will have the following form:

\[
H = A \otimes D \otimes A.
\]

(27)

\[
X' = \left( (H \otimes (X)) \oplus MK \right) \oplus MR.
\]

(28)

V. ORIGINAL IMAGE ENCRYPTION

After preparing the original image and constructing all the parameters, the following figure will explain the encryption process in detail.

Fig. 4. Clear image encryption.

Fig. 5. Decryption process.

A decryption function can be described as:

\[
\text{Improved Hill inverse function}
\]

\[
X' = \left( (H \otimes (X)) \oplus MK \right) \oplus MR
\]

\[
X' \oplus \text{Exp}(255 - \text{Log}(MR)) = \left( (H \otimes (X)) \oplus MK \right)
\]

\[
(X) = H^{-1} \left( (X' \oplus \text{Exp}(255 - \text{Log}(MR)) \oplus MK) \right)
\]

\[
\text{Reverse diffusion}
\]

\[
\text{We have } CM(i + 1 :) = \Pi (MC(i :)) \oplus CM(i + 1 :)
\]

\[
MC(i :) = \Pi^{-1} \left( CM(i + 1 :) \oplus CM(i + 1 :) \right)
\]

VI. DECRYPTING THE ENCRYPTED IMAGE

Our technique is a symmetric encryption system using a spread function, which forces us to start the decryption process from the last block to the first block, and then recalculate the initialization vector to extract the exact value of the first block. The figure below illustrates the decryption process.

\[
\Pi(CM(i + 1 :)) = MC(i :) \oplus CM(i + 1 :).
\]

(29)
The polynomial \( p(x) = x^8 + x^7 + x^2 + x + 1 \) is irreducible and eighth order on \( F[x] \), so it is a candidate for this study in the construction of the simulation body \( F_{256} \). In addition, the polynomial \( g(x) = x \) is a generator of such agents. Under these conditions, the (TS) dispersion index table is shown below.

Table 1. Discrete exponential table.

| \( T \) | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 100 | 101 | 102 | 103 | 104 | 105 | 106 | 107 | 108 | 109 | 110 | 111 | 112 | 113 | 114 | 115 | 116 | 117 | 118 | 119 | 120 | 121 | 122 | 123 | 124 | 125 | 126 | 127 | 128 | 129 | 130 | 131 | 132 | 133 | 134 | 135 | 136 | 137 | 138 | 139 | 140 | 141 | 142 | 143 | 144 | 145 | 146 | 147 | 148 | 149 | 150 | 151 | 152 | 153 | 154 | 155 | 156 | 157 | 158 | 159 | 160 | 161 | 162 | 163 | 164 | 165 | 166 | 167 | 168 | 169 | 170 | 171 | 172 | 173 | 174 | 175 | 176 | 177 | 178 | 179 | 180 | 181 | 182 | 183 | 184 | 185 | 186 | 187 | 188 | 189 | 190 | 191 | 192 | 193 | 194 | 195 | 196 | 197 | 198 | 199 | 200 | 201 | 202 | 203 | 204 | 205 | 206 | 207 | 208 | 209 | 210 | 211 | 212 | 213 | 214 | 215 | 216 | 217 | 218 | 219 | 220 | 221 | 222 | 223 | 224 | 225 | 226 | 227 | 228 | 229 | 230 | 231 | 232 | 233 | 234 | 235 | 236 | 237 | 238 | 239 | 240 | 241 | 242 | 243 | 244 | 245 | 246 | 247 | 248 | 249 | 250 | 251 | 252 | 253 | 254 | 255 |

Example:

\( TS(10,3) = \text{Exp}(163) = 253. \)

So

\( \text{Log}(253) = 163 \)

By applying inverse permutation, a table of discrete logarithms can be derived from a table of discrete exponents. The two tables are used mutually in the field \( (F_{256}) \).

\[
152^{-1} = \text{Exp} \left( 255 - \text{Log} \left( 152 \right) \right) = \text{Exp} \left( 255 - 80 \right) = \text{Exp} \left( 174 \right) = 179.
\]

So

\[
\begin{align*}
183 \otimes 250 &= \text{Exp} \left( \text{mod} \left( \text{Log} \left( 152 \right) + \text{Log} \left( 250 \right) \right) \right) \\
&= \text{Exp} \left( 80 + 163 \right) = \text{Exp} \left( 243 \right) = 31
\end{align*}
\]

In matrix notation,

\[
M = \begin{pmatrix} x^2 + x + 1 \\ x^3 + x^2 \\ x^6 + x^4 + 1 \end{pmatrix} \Rightarrow M = \begin{pmatrix} 7 \\ 12 \\ 81 \end{pmatrix}.
\]

So

\[
\begin{pmatrix} 123 \\ 211 \\ 183 \end{pmatrix},
\]

and

\[
\begin{pmatrix} 106 \\ 101 \\ 251 \end{pmatrix}.
\]

### VIII. INVESTIGATION OF CRYPTO SYSTEM PERFORMANCE

In this section, all the experiments were performed on a large color image database and using a core i7 personal computer, 16Gb memory, 500Gb hard disk under the matlab software running under windows 7. Some of the most used reference images in cryptography and tested by our approach.

Table 2. Images encrypted by our approach.

<table>
<thead>
<tr>
<th>Image</th>
<th>Original Image</th>
<th>Original Image Histogram</th>
<th>Encrypted Image</th>
<th>Encrypted Image Histogram</th>
</tr>
</thead>
<tbody>
<tr>
<td>260-265</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>266-270</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>271-275</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>276-280</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1) **Key-space analysis**

In our example simulation we took as encryption key

\[ u_0 = 0.7655412001, \mu = 3.89231541, \]

for logistic map,

\[ v_0 = 0.865421331, v_1 = 0.563215, b = 1.3561 \]

for Henon map

\[
\text{The global Key space } \approx 2^{180} \gg 2^{100}.
\]
2) Secret key’s sensitivity Analysis

The high sensitivity of the encryption keys used in our system indicates that a very slight degradation of the encryption key automatically leads to an image that is so different from the original image. This confirmation can be viewed below the scheme in the next figure:

![Secret key's sensitivity](image)

Fig. 6. Secret key’s sensitivity.

We note that a $10^{-15}$ change in a single encryption parameter of this technology is incapable of restoring the clear image by the same decryption process.

2) Strength analysis of the new generation

Our design has given a new opportunity to survive and to partner with the strongest members in the hope of rebuilding a new population more adapted to intruder aggression. To do that, we randomly selected an image and studied the strength of the original populations and the new generation, with the following results:

3) Statistics attack security

a) Histogram analysis

The histogram gives the distribution of the pixel intensity level of any original image passed under our algorithm, showing the concentration near certain intensity values and sometimes the maximum value, while the histograms of all encrypted images are uniformly distributed. Yes, this eliminates any statistical histogram attacks.

b) Entropy Analysis

Entropy information is very important in measuring the randomness of the encrypted image. It is defined by the following equation: $(MC)$ image of size $(n, m)$, we pose $t = nm$, so

$$H(MC) = \frac{1}{t} \sum_{i=1}^{t} -p(i) \log_2(p(i))$$

where $p(i)$ is the probability of occurrence of level $(i)$ in the image of the selected database. If $H(MC)$ is close to the value $8$ (8-bit coded image), the completely random aspect of the encrypted image is ensured. The following table illustrates the entropy of some reference images tested by our method:

<table>
<thead>
<tr>
<th>Image</th>
<th>Size</th>
<th>Cypher</th>
<th>Entropy</th>
</tr>
</thead>
<tbody>
<tr>
<td>256x256</td>
<td>7,9993</td>
<td></td>
<td></td>
</tr>
<tr>
<td>512x512</td>
<td>7,9998</td>
<td></td>
<td></td>
</tr>
<tr>
<td>512x512</td>
<td>7,9997</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1024x1024</td>
<td>7,9999</td>
<td></td>
<td></td>
</tr>
<tr>
<td>256x256</td>
<td>7,9991</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
c) Correlation analysis

The correlation is given by:

\[ r = \frac{c(x,y)}{\sqrt{V(x)V(y)}} \]  

(30)

The following table illustrates the entropy of some reference images tested by our method:

Table 5. Correlation of some tested images.

<table>
<thead>
<tr>
<th>Image</th>
<th>Original Image</th>
<th>Encrypted Image</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Size H:C V:C D:C H:C V:C D:C</td>
<td></td>
</tr>
<tr>
<td></td>
<td>512x512 0.9047 0.0526 0.0230 -0.0037 -0.0004 0.0003</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1024x1024 0.9774 0.9823 0.0643 0.0031 0.0002 0.0010</td>
<td></td>
</tr>
<tr>
<td></td>
<td>512x512 0.9036 0.9824 0.0642 -0.0021 0.0006 0.0003</td>
<td></td>
</tr>
<tr>
<td></td>
<td>512x512 0.9774 0.9811 0.9986 0.0023 -0.0001 -0.0005</td>
<td></td>
</tr>
</tbody>
</table>

5) Differential analysis

Let be two encrypted images, whose corresponding free-to-air images differ by only one bit, from \((C_1)\) and \((C_2)\), respectively. The expressions of these two statistical constants \((NPCR)\) and \((UACI)\) are given by equations below

\[ NPCR = \left( \frac{1}{nm} \sum_{i,j=1}^{nm} D(i,j) \right) \times 100, \]  

(31)

with

\[ D(i,j) = \begin{cases} 1 & \text{if } C_1(i,j) \neq C_2(i,j), \\ 0 & \text{if } C_1(i,j) = C_2(i,j) \end{cases} \]

The \(UACI\) mathematical analysis

\[ UACI = \left( \frac{1}{nm} \sum_{i,j=1}^{nm} \text{Abs}(C_1(i,j) - C_2(i,j)) \right) \times 100. \]  

(32)

a) Signal-To-Peak Noise Ratio (PSNR)

i. \(MSE\)

Mean Square Error (MSE): This is the cumulative square deviation between the original image and other noisy images. When the MSE level decreases, the error also decreases. This constant measure the distance between the pixels of the clear image and the encrypted image. Calculated by the next equation.

\[ MSE = \sum_{i,j}(P(i,j) - C(i,j))^2, \]  

(33)

where \((P(i,j))\): pixel of the clear image and \((C(i,j))\): pixel of the cypher image.

The following table illustrates some differential parameters.

Table 6. Differential parameters.

<table>
<thead>
<tr>
<th>Image</th>
<th>Size</th>
<th>NPCR</th>
<th>UACI</th>
<th>PSNR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>256x256</td>
<td>99.92</td>
<td>33.35</td>
<td>8.36</td>
</tr>
<tr>
<td></td>
<td>512x512</td>
<td>99.67</td>
<td>34.23</td>
<td>8.65</td>
</tr>
<tr>
<td></td>
<td>1024x1024</td>
<td>99.96</td>
<td>33.37</td>
<td>8.10</td>
</tr>
</tbody>
</table>

b) Avalanche effect

Our algorithm uses a strong link between encrypted pixels and pixels with clear policies. As a result, as data propagates through the structure of the algorithm, gradual changes become increasingly important. The avalanche effect is the number of bits that have been changed if a single bit in the original image is changed. The mathematical expression of this avalanche effect is given by
\[ AE = \left( \frac{\sum_{i} \text{bit change}}{\sum_{i} \text{bit total}} \right) \times 100. \]  
(35)

Table 7. Avalanche effect.

<table>
<thead>
<tr>
<th>Original Image</th>
<th>Cypher Image</th>
<th>AE</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td>78.25</td>
</tr>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td>71.04</td>
</tr>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td>76.26</td>
</tr>
</tbody>
</table>

c) Performance time

In our technique, the encryption and decryption times are very similar and vary in the interval [0.05, 0.1].

Table 8. Performance time.

<table>
<thead>
<tr>
<th>Original Image</th>
<th>Size</th>
<th>Enc</th>
<th>Decr</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td>512x512</td>
<td>0.04</td>
<td>0.032</td>
</tr>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td>256x256</td>
<td>0.011</td>
<td>0.008</td>
</tr>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td>512x512</td>
<td>0.01</td>
<td>0.009</td>
</tr>
</tbody>
</table>

d) Speed analysis

To approve and document the quality of our methodology in a timely fashion. And finally, thanks to these properties, we have selected the "Lena" grayscale image with three different sizes (256 x 256) (512 x 512) and (1024 x 1024). The results are presented in Table 9.

Table 9. Execution time (in second).

<table>
<thead>
<tr>
<th>Image</th>
<th>Our method</th>
<th>DES</th>
<th>AES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena(256x256)</td>
<td>0.09647</td>
<td>0.639777</td>
<td>5.687244e-02</td>
</tr>
<tr>
<td>Lena(512x512)</td>
<td>0.27443</td>
<td>7.449005</td>
<td>0.347306</td>
</tr>
<tr>
<td>Lena(1024x1024)</td>
<td>0.20134</td>
<td>29.11388</td>
<td>1.152980</td>
</tr>
</tbody>
</table>

We compared the results with two classic algorithms, AES and DES, and can determine that the execution time is reasonable. The test was conducted on other images of different sizes, and we obtained an acceptable score. This is due to the low algorithm complexity of the algorithm implemented in our strategy.

IX. MATH SECURITY

Our algorithm uses a large symmetric key that is extremely sensitive to initial conditions and control parameters. This ensures that small interference in the key will regenerate a new subject and a new calculation table. In addition, the complexity of using discrete logarithms in calculations increases the difficulty of attacking our systems. The construction of the key matrix is closely linked to the chaotic maps used, which eliminates any brutal attacks.

X. CONCLUSION

Hill's conventional system is very easy to install in the color image encryption system, as long as the inversion matrix is determined in the carefully selected ring. But due to linearity, this technique is still vulnerable to statistics and brute force attacks. Carried on instead of the classic \( \mathbb{Z}/256\mathbb{Z} \) ring. Similarly, the construction of a large-sized invertible matrix has been introduced based on the involution block, and the non-zero eigenvalue matrix has been described in detail. The large number of matrices built in this way ensures better protection against any brutal attack. Using logarithms and discrete exponents and translation vectors to overcome linear problems will increase the complexity of our method.

Acknowledgement

This article is not subsidized by any public or private
organization. It is a personal work.

REFERENCES


Authors

Mr. Abdellatif JarJar is the alone author of this article, and therefore no conflict. To finalize this document, I did not receive any assistance funds from any organization. This document does not contain any studies or experiments on animals. This article does not contain any studies with animals performed by any of the authors. This article does not contain any studies with human participants or animals performed by any of the author.