applied sciences

MDPI

Article

A Dynamic Hybrid Cryptosystem Using Chaos and Diffie–Hellman Protocol: An Image Encryption Application

Rolando Flores-Carapia [1], Víctor Manuel Silva-García [1] and Manuel Alejandro Cardona-López [2,*]

[1] Instituto Politécnico Nacional, Centro de Innovación y Desarrollo Tecnológico en Cómputo, Ciudad de México 07738, Mexico; rfloresca@ipn.mx (R.F.-C.); vsilvag@ipn.mx (V.M.S.-G.)
[2] Instituto Politécnico Nacional, Centro de Investigación en Computación, Ciudad de México 07738, Mexico
* Correspondence: mcardonal2022@cic.ipn.mx

Abstract: Images with sensitive content require encryption for storage and transmission. Symmetric schemes can cipher them, while an asymmetric cryptosystem can distribute the secret key safely. For this reason, we propose a dynamic hybrid cryptosystem, which ciphers images and transfers its private keys. It has a symmetric algorithm that applies the Lorenz equations for generating different boxes and permutations in every encryption process and round. Since the secret key concatenates two private numbers, an asymmetric algorithm is included for its key distribution. The proposal uses the Diffie–Hellman protocol with ElGamal for obtaining a seed and building 128 strings. Then, the SHA-512 is applied in each of them a number of times associated with the secret key value in its blockchain representation. The resultant strings are concatenated to conform to the public key. Finally, the tests indicate that the cryptosystem resists differential, linear, algebraic, and brute-force attacks. Its cipher quality is high according to the entropy, correlation, DFT, NPCR, UACI, AC, texture analysis, and goodness of fit test. Additionally, occlusion, additive, multiplicative, and the proposed χ^2 noise attacks are simulated on encrypted images. Finally, the sharpness loss is measured with the Similarity Parameter and improved with a filter 5×5.

Keywords: Lorenz equations; ElGamal system; chaos; Hash SHA functions; dynamic *S*-box; dynamic permutation

Citation: Flores-Carapia, R.; Silva-García, V.M.; Cardona-López, M.A. A Dynamic Hybrid Cryptosystem Using Chaos and Diffie–Hellman Protocol: An Image Encryption Application. *Appl. Sci.* **2023**, *13*, 7168. https://doi.org/10.3390/app13127168

Academic Editor: Mostafa Fouda

Received: 23 May 2023
Revised: 11 June 2023
Accepted: 13 June 2023
Published: 15 June 2023

1. Introduction

Images are a representative source of information [1] in different fields. Some of them require privacy and security because of their sensitive content, such as medical registers and personal identification. In this sense, Cryptography provides symmetric and asymmetric schemes for protecting data before storing and transmitting it. For this reason, many robust cryptosystems have emerged to cipher images [2–6]. Furthermore, the study of hybrid algorithms has increased since the benefits of including the strengths of symmetric and asymmetric cryptosystems simultaneously. In this way, symmetric cryptosystems will have a less drastic security impact in the age of quantum computers [7], while an asymmetric scheme permits key distribution and digital signatures.

For these reasons, this research proposes a hybrid cryptosystem, which generates a seed with the Diffie–Hellman protocol based on ElGammal. This seed is sent to the receiver according to the Block-Chain technique as a string of SHA-512 strings. Subsequently, a fourteen-round symmetric cryptosystem is built using the seed over the Lorenz equations solution and a bijective function. The substitution boxes, permutations, and round keys are dynamic. This means that they are all different in each encryption process and round. The authors call this cryptosystem an Algorithm of Image Cipher using Lorenz equations and the Diffie–Hellman protocol (AICLDH). It is compatible with 256 grayscale levels and color images and the schedule keys bytes size is equal to the image pixel size.

On the other hand, related works with hybrid algorithms, which include symmetric and asymmetric algorithms have been proposed. For instance, Elliptic Curve Cryptography

(ECC) with Hill Cipher (HC), ECC with Advanced Encryption Standard (AES), and ElGamal with Double Playfair Cipher [8]. Guodong et al. propose a hybrid cryptosystem applying the RSA cryptosystem for the generation of a pair of public and private keys [9] that precede the diffusion and confusion process. Its security depends on the RSA algorithm. Moreover, the Diffie–Hellman protocol and SHA-256 function were used to cipher images. The first 128 bits verify confidentiality and the remaining 128 bits for authentication [10].

Some image encryption algorithms compute their security results based on similar evaluations and parameters to this proposal, and part of the numerical results achieved in this proposal is superior to them. For instance, Ye et al. reported a maximum entropy of 7.99930 [11], which indicates high-quality encryption, and this work achieved a superior one equal to 7.99938. A comparative table is included in the discussion section.

The number of elements in the key space, and the cipher image resistance to noise attacks, are other differences with other developments [12–15]. In this order of ideas, these proposals have a number of elements in the key set less than ours because it is smaller than the 2^{1024} elements reached by AICLDH. In addition, they do not analyze the noise attack on encrypted images, and the developments that include it do not quantify the loss of sharpness [11,16] with any instrument. In contrast, this work proposes the SP parameter to measure it.

All cipher quality evaluations examine at least two parameters: entropy and correlation. Although compressed images with loss of information, such as JPEG, do not include them. Just the correlation parameter is considered [17], and the entropy achieves a value near 7.8 [18]. In contrast, the entropy of cipher images with the proposal is close to 7.999. Due to the absence of encryption measures, the present work does not consider lossy compressed images. Another argument is that the normativity of some countries does not allow lossy compression for images. One example is Mexico [19]. In this case, the proposal suggests and works over the possibility of storing images using BMP files or others without loss of information.

Later, the security analysis is made when linear, differential, algebraic, and brute-force attacks are applied [20]. To evaluate the resistance of AICLDH to the differential attack, we employ the Number of Pixels Change Rate (NPCR), Unified Average Changing Intensity (UACI), and Avalanche Criteria (AC) parameters. Furthermore, since the AICLDH construction uses an ElGamal asymmetric cryptosystem, the discrete logarithm problem is considered for its security analysis [21]. It tries to find the a value in equation $\beta = \alpha^a \mod p$ when β, α y p values are known. In this sense, many algorithms have been developed, called generic algorithms, to find the value a [22], as well as the Pohlig—Hellman attack [23]. In Section 6, we analyzed the complexity of these algorithms.

For evaluation purposes, cipher images are damaged with the additive, multiplicative, and occlusion noises [24] and one proposed using the χ^2 distribution. As a result of these attacks, a sharpness loss is produced over the images. This work proposes measuring it with the Similarity Parameter (SP) for each color: red, green, and blue. The cipher quality evaluation includes the encryption of totally black and totally white images, as well as other works measuring it [25].

The distribution of this work begins with Section 1, which corresponds to the introduction and some related works. Section 2 provides a theoretical description of the materials and methods applied in the proposal. Section 3 shows the building elements that conform to the proposed cryptosystem as well as the quality test. Section 4 presents all the noise attacks considered for a damage simulation to the encrypted images. In addition, it includes the median filter 5×5 application to improve the image quality after the attack and its measure with the SP parameter. The results are shown in Section 5, while their discussion and analysis are presented in Section 6. Finally, Section 7 contains the conclusions and future works.

2. Materials and Methods

The theoretical tools required for the proposal are described below, starting with the Lorenz equations.

2.1. Lorenz Equations

The Lorenz system of differential equations is shown in Equations (1)–(3) [26], where $\sigma, r, b \in \mathbb{R}^+$.

$$\frac{dx}{dt} = \sigma(-x + y) \tag{1}$$

$$\frac{dy}{dt} = rx - y - xy \tag{2}$$

$$\frac{dz}{dt} = -bx + xy \tag{3}$$

If the Lorenz Equations (1)–(3) are equal to zero, the critical points are found. In this case, the critical points are $P_1 = (0, 0, 0)$, $P_2 = (\sqrt{b(r-1)}, \sqrt{b(r-1)}, r-1)$, and $P_3 = (-\sqrt{b(r-1)}, -\sqrt{b(r-1)}, r-1)$.

The Lorenz equations describe the convection phenomenon in the Earth's atmosphere, in this case, $\sigma = 10$ and $b = \frac{8}{3}$. Moreover, the solution to the Lorenz system of equations follows the form $\vec{X} = \vec{\xi} e^{\lambda t}$ where $\vec{\xi}$ represent the eigenvectors, and λ the eigenvalues. On the other hand, for the calculation of the solutions in the neighborhood of the point P_2, we start from $X^{'} = AX$, where the matrices A, X, and $X^{'}$ are described in Equations (4)–(6).

$$A = \begin{pmatrix} 10 & 10 & 0 \\ r & -1 & -\sqrt{8/3(r-1)} \\ \sqrt{8/3(r-1)} & \sqrt{8/3(r-1)} & -8/3 \end{pmatrix} \tag{4}$$

$$X = \begin{pmatrix} \mathrm{x} \\ \mathrm{y} \\ \mathrm{z} \end{pmatrix} \tag{5}$$

$$X^{'} = \begin{pmatrix} x^{'} \\ y^{'} \\ z^{'} \end{pmatrix} \tag{6}$$

The eigenvalues come from the characteristic polynomial and are obtained from Equation (7).

$$|A - \lambda I| = 0 \tag{7}$$

Equation (8) expresses the characteristic polynomial with $r = 28$.

$$3\lambda^3 + 41\lambda^2 - 50\lambda + 2160 = 0 \tag{8}$$

From Equation (8), one real root and two complex ones are obtained; these are written in Equations (9)–(11).

$$\lambda_1 = -22.558424 \tag{9}$$

$$\lambda_2 = 4.445878 + 3.485904i \tag{10}$$

$$\lambda_3 = 4.445878 - 3.485904i \tag{11}$$

Regarding the eigenvectors, it is only necessary to generate two of them to obtain the general solution. Equations (12) and (13) show the eigenvectors $\vec{\xi_1}$ and $\vec{\xi_2}$, respectively.

$$\vec{\xi_1} = \begin{pmatrix} 9.163288 \\ -11.507650 \\ 1 \end{pmatrix} \tag{12}$$

$$\vec{\xi_2} = \begin{pmatrix} 0.359510 + 0.116796i \\ 0.478680 + 0.294040i \\ 1 + 0i \end{pmatrix} \tag{13}$$

$$\vec{u(t)} = \begin{pmatrix} 0.3595\cos(3.4859)t - 0.1167\sin(3.4859)t \\ 0.4786\cos(3.4859)t - 0.2940\sin(3.4859)t \\ \cos(3.4859)t \end{pmatrix} e^{dt} \tag{14}$$

$$\vec{v(t)} = \begin{pmatrix} 0.1167\cos(3.4859)t - 0.3595\sin(3.4859)t \\ 0.2940\cos(3.4859)t - 0.4786\sin(3.4859)t \\ \sin(3.4859)t \end{pmatrix} e^{dt} \tag{15}$$

Note that d = 4.445878, and the solution $\vec{\xi_2} e^{(d+3.485904i)t}$ contains a real part and a complex part. The real part is presented as $\vec{u}$ and the complex one as $\vec{v} \times i$, where $\vec{u}$ and $\vec{v}$ are expressed in Equations (14) and (15). Furthermore, if we set $\vec{w} = \vec{\xi_1} e^{-22.5584t}$, the general solution is then written in Equation (16).

$$\vec{X(t)} = e^{-22.5584t} C_1 \vec{\xi_1} + C_2 \vec{u}(t) + C_3 \vec{v}(t) \times i \tag{16}$$

The function $\varphi_z(t)$ is composed of the third coordinate of the vectors in Equation (16). On the other hand, in this development, $C_1 = 0$ and $t_0 = 1/(4.445878)$. The result is given in Equation (17).

$$\varphi_y(t_0) = (0.999924)C_2 e + (0.012315)C_3 e \tag{17}$$

In addition, the units of the argument θ in $\sin\theta$ and $\cos\theta$ functions are degrees.

2.2. ElGamal System

AICLDH has a symmetric cryptosystem using the key share of the ElGamal system and the Diffie–Hellman protocol [27] that works with prime numbers. This study suggests building a prime p^* in the following way: $p^* = 2k \times p_1 p_2 + 1$. Where p_1 and p_2 are numbers of high primality with a bit size of 2^{512}, and $k = 1, 2, \cdots$. Finally, the proposed α has to satisfy Equation (18) [28] for all the prime factors of $p^* - 1$ and $0 < \alpha < p^* - 1$.

$$\alpha^{p^*-1/q} \neq 1 \mod p^* \tag{18}$$

where q are the prime factors of $(p^* - 1)$ different from 1. It is pointed out that the high primality of p_1, p_2 is verified using the Miller-Rabin algorithm [29].

On another topic, both the sender A, and the receiver B, share the following value: $\beta = (\beta_{a_B})^{a_A} \mod p^*$ = $(\beta_{a_A})^{a_B} \mod p^*$. Considering that $\beta_{a_B} = \alpha^{a_B} \mod p^*$ and $\beta_{a_A} = \alpha^{a_A} \mod p^*$. The sender receives the value β_{a_B} from the receiver, and the receiver obtains the integer β_{a_A} from the sender. For increasing security, the integers a_A and a_B can change in each communication between the sender and the receiver. Although, they could also remain fixed for a while. Furthermore, both integers satisfy the following condition: $0 < a_A, a_B < p^* - 1$.

In this work, we calculate the values $\beta_i = \alpha^i \times \beta \mod p^*$ to send a pair of constants using the Hash SHA-512 algorithm. In addition, the authors consider it convenient to present an example with particular values. If we assume that $p_1 = 101$, $p_2 = 103$, it is not difficult to verify that the prime we are looking for is $p^* = 2k \times p_1 p_2 + 1 = 20{,}807$, when $k = 1$.

In this order of ideas, if we propose α = 17,742, it can be verified that $\alpha^{p^*-1/q} \neq 1 \mod p^*$ when the primes q are equal to: 2 , 101, and 103. On the other hand, if we assume that sender A and receiver B have the private keys $a_A = 600$ and $a_B = 1500$, it is easy to check that the public keys of the sender and receiver are: b_A = 11,147 and b_B =16,143. Therefore, β = 11,425 can be computed by the sender and the receiver. To finish this example, β_1 = (17,742)(11,425) $\mod$ 20,807 $\equiv$ 556.

2.3. The Hash SHA Function

In this research, we will use the Hash standard, particularly the SHA-512 algorithm [30]. In this sense, the SHA-512 algorithm results in a fixed-length string of 512 bits. Furthermore, the input message can have a maximum of $2^{128} - 1$ bits. Its implementation in the present work is by means of the Java programming language to execute it on a PC. Although it is not the unique option to work SHA functions, some proposals have developed a Hash accelerator in an FPGA [31]. In the same way, it is pointed out that the SHA-512 algorithm defines a function that is not one-to-one. The above means that, given a result, going in the reverse direction is difficult. In this case, obtaining the message. In fact, this type of problem is called "Preimage". Furthermore, we can say that the SHA-512 algorithm defines a one-path procedure [32]. This property is relevant since the results are public during transmission. Regarding security, in a sample of 2^{256} messages, the probability of a collision is at most 0.5 [33].

2.4. Entropy

In most image encryption works, a parameter that measures the encryption quality is entropy [34]. It is calculated according to Equation (19) [35].

$$E(x) = -\sum_{x \varepsilon X} P_r(x) log_2 P_r(x) \tag{19}$$

One byte can represent the pixel of an image with 256 gray levels, and three bytes a pixel of a color image, one for each basic color (red, green, and blue). On the other hand, images with 256 levels with an entropy equal to 8 indicate a well-encrypted image. In other words, this means that the insensitive color levels are uniformly distributed, and the associated histogram is close to a uniform distribution.

Despite obtaining an entropy value equal to 8, the color distribution is not necessarily random. It is possible to build a theoretical distribution with entropy equal to 8 and not randomly. For this reason, the distribution randomness is measured using different instruments and verified for each color. However, an entropy value close to 8 indicates an acceptable degree of randomness [36].

2.5. Correlation Coefficient

The correlation coefficient is another classical parameter to measure encryption quality [37,38], where a correlation close to zero indicates secure encryption. Its procedure follows the next steps. Suppose we analyze the horizontal direction and the color red; first, randomly, a pixel of the encrypted image is chosen. It has three basic colors, and z_r represents the red color. Subsequently, the adjacent pixel to it in the horizontal direction is selected, and w_r denotes its red component. Therefore, it is possible to calculate the correlation between z_r and w_r for n pairs of pixels in the horizontal direction and the red color with Equation (20). In addition, the means computing $\overline{z}_r$, $\overline{w}_r$ are indicated in Equations (21) and (22). The steps are applicable to compute the correlation in other directions and colors.

$$r_{h;z_r,w_r} = \frac{\frac{1}{n}(\sum_{i=1}^{n}(z_{i,r} - \overline{z}_r)(w_{i,r} - \overline{w}_r))}{\sqrt{(\frac{1}{n}\sum_{i=1}^{n}(z_{i,r} - \overline{z}_r)^2(\frac{1}{n}\sum_{i=1}^{n}(w_{i,r} - \overline{w}_r)^2))}} \tag{20}$$

$$\overline{z}_r = \frac{1}{n}\sum_{i=1}^{n} z_{i,r} \tag{21}$$

$$\overline{w}_r = \frac{1}{n}\sum_{i=1}^{n} w_{i,r} \tag{22}$$

2.6. Discrete Fourier Transform

Another instrument used in this research is the Discrete Fourier Transform (DFT), which is a hypothesis test that measures the randomness in a bit string. The goal is to analyze that there is no repeating pattern in the binary string [39]. Additionally, it is part of the NIST 800-22 standard. Finally, the variables involved in the test are expressed in Equations (23)–(25), which are written below.

$$N_0 = \frac{(0.95) \times n}{0.05} \tag{23}$$

In Equation (23), n is the length of the string. Moreover, using this data, a bound l is calculated according to Equation (24).

$$l = \sqrt{\mathrm{Ln}\frac{1}{0.05}(n)} \tag{24}$$

Additionally, the g_j functions of the procedure are calculated according to Equation (25). Where $x_k = \{-1, 1\}$, $i = \sqrt{-1}$ and $j = 1, 2, \ldots, \frac{n}{2} - 1$, in our case the value n is always even. Another key point in Equation (25) is that g_j is a complex function. The variable N_1 in Equation (26) starts equal to zero. Subsequently, $\|g_j\|$ is determined. In case of a value less than l, then one is added to N_1. Otherwise, the N_1 value is not modified.

$$g_j = \sum_{k=1}^{n} x_k e^{\frac{2\pi(i)(k-1)j}{n}} \tag{25}$$

Once N_1 has been calculated, it is possible to obtain d using Equation (26). From here, the decision rule is: if the P-value, (defined in Equation (27)), is less than 0.01, then the hypothesis that the string is random is rejected. Otherwise, it is accepted.

$$d = \frac{N_1 - N_0}{\sqrt{\frac{n(0.95)(0.05)}{4}}} \tag{26}$$

$$P\text{-}value = \mathrm{erfc}\frac{|\, d \,|}{\sqrt{2}} \tag{27}$$

The erfc function is calculated according to Equation (28).

$$\mathrm{erfc}\frac{|\, d \,|}{\sqrt{2}} = 2(1 - \Phi(|\, d \,|)) \tag{28}$$

2.7. Parameters to Measure the Strength of AICLDH against the Differential Attack

The strength of AICLDH against differential attack is tested based on Number of Pixel Change Rate (NPCR), Unified Average Changing Intensity (UACI), and Avalanche Criterion (AC).

The NPCR parameter requires two images ciphered to compare their values in the same position (i, j). A function $D(i, j)$ measures the differences between the bytes of the encrypted image number 1 and belongs to encrypted image 2. That is, given the position (i, j), it is observed that bytes are different or equal. If they are identical, the function $D(i, j)$ takes the value 0. Otherwise, it takes the value 1. The NPCR parameter is defined in Equation (29). The subscript indicates the analyzed color, and variables W, H are the width and height of the encrypted images. Convenient values to avoid a differential attack are around 99.6% [40].

The UACI measure also evaluates byte differences between the encrypted images 1 and 2 by means of Equation (30). Although, it considers the intensity levels of each color, from 0 to 255. The desired value of UACI to resist the differential attack is 33.4% [41].

$$\mathrm{NPCR}_c = \frac{\Sigma_{i,j} D(i,j)_c}{W \times H} \times 100\% \tag{29}$$

$$\mathrm{UACI}_c = \frac{1}{W \times H}[\Sigma_{i,j} \frac{\mid C_{1,c} - C_{2,c} \mid}{255}] \times 100\% \tag{30}$$

Finally, the AC parameter is calculated according to Equation (31). T indicates the number of bits from images 1, 2, and the subscript c is the color analyzed. This measure evaluates the bitwise differences between encrypted images 1 and 2. In this sense, the function $b(i,j)$, which appears in Equation (31), is described in Equation (32). The appropriate value for this parameter is 50%.

$$\mathrm{AC}_c = \frac{\Sigma_{i,j} b(i,j)_c}{T} \times 100\% \tag{31}$$

$$b(i,j)_c = \begin{cases} 0 \text{ otherwise} \\ 1 \text{ if bits are diffent} \end{cases} \tag{32}$$

2.8. Energy, Contrast, and Homogeneity

Other encryption quality measures based on texture analysis are energy, contrast, and homogeneity. Equation (33) defines energy, where $f(i,j)$ represents the energy value at the point (i,j). It measures the degree of information disorder in an encrypted image based on the bytes of each basic color. In this case, if the energy value is close to zero, it indicates a high disorder level in the encrypted image. In other words, it means that the image is well encrypted. In consequence, it is sought that the energy values are close to zero [42].

$$Energy = \sum_{i,j} f(i,j)^2 \tag{33}$$

While Equation (34) calculates contrast. As in the previous indicator, the function $f(i,j)$ is the value that f takes on the point (i,j). It is important to say that contrast measures the differences between neighboring points. Moreover, it is expected to have high values [43], which indicates high-quality encryption.

$$Contrast = \sum_{i,j} \mid i-j \mid^2 f(i,j) \tag{34}$$

Homogeneity is defined according to (35). A well-encrypted image should return small values [44].

$$Homogeneity = \sum_{i,j} \frac{f(i,j)}{1+ \mid i-j \mid} \tag{35}$$

2.9. Goodness-of-Fit Test

Different from the previous tests, this gives a hypothesis test. The null one is that the basic color data fits a uniform distribution. Since it is a statistical test, it demands a statistic and a threshold. This research uses a statistic with a χ^2 distribution and $k-1$ degrees of freedom. Equation (36) shows the instrument.

$$\chi^2 = \sum_{i=1}^{k} \frac{(o_i - \exp)^2}{\exp} \tag{36}$$

According to the central limit theorem [45], the random variable χ^2 can approximate a normal distribution with a mean $\mu = 255$ and variance $\sigma = 22.58$. Consequently, if $\chi^2 \leqq 308$, the null hypothesis is accepted; otherwise, it is rejected for a significance level $\alpha = 0.01$. It is important to realize that this test is absent in the NIST 800-22 standard for measuring the randomness of each basic color using the corresponding bit string [39].

2.10. The Median Filter

We will use the Median Filter as a complementary tool. Since the encrypted images are damaged, there is a loss of sharpness in the decrypted images. For this reason, in the present research, the median filter 5×5 is proposed to improve its sharpness. The application of this instrument proceeds as follows: given a point (x_1, x_2) of the pixel map, an adjustment is performed on the neighboring pixels of size (w, h). In summary, the median filter 5×5 builds a mask of 25 pixels from a pixel (x_1, x_2). In such a way that the point (x_1, x_2) is in the center, and the others are around it. Figure 1 illustrates it. In this sense, the point $(0, 0)$ represents (x_1, x_2).

(x_1-2, x_2+2)	(x_1-1, x_2+2)	(x_1, x_2+2)	(x_1+1, x_2+2)	(x_1+2, x_2+2)
(x_1-2, x_2+1)	(x_1-1, x_2+1)	(x_1, x_2+1)	(x_1+1, x_2+1)	(x_1+2, x_2+1)
(x_1-2, x_2)	(x_1-1, x_2)	(x_1, x_2)	(x_1+1, x_2)	(x_1+2, x_2)
(x_1-2, x_2-1)	(x_1-1, x_2-1)	(x_1, x_2-1)	(x_1+1, x_2-1)	(x_1+2, x_2-1)
(x_1-2, x_2-2)	(x_1-1, x_2-2)	(x_1, x_2-2)	(x_1+1, x_2-2)	(x_1+2, x_2-2)

Figure 1. A mask of size 5×5 for the median filter.

Afterward, values are ordered according to their intensities. Let us denote the medians of each color as $M_{r,(x_1,x_2)}$, $M_{g,(x_1,x_2)}$ and $M_{b,(x_1,x_2)}$. The median per color is greater than or equal to $\lceil \frac{25}{2} \rceil - 1$ and less than the other values. Subsequently, these nine values are ordered according to their intensity. Then, the median is chosen as the value that meets the following: it is greater than or equal to the first; in other words, 50%, and less than the remaining points.

3. Building Elements

The section presents the algorithm for permutation generation and the Similarity Parameter based on the UACI parameter, among other elements. For the stages that require random numbers, a random number generator such as [46] can be used.

3.1. Algorithm for the Generation of Permutations

For a set $Z_m = \{n \in N \mid 0 \leq n \leq m! - 1\}$ any of its elements can be written in factorial basis as in Equation (37), where the constant $D_{m-1} = 0$.

$$n = D_0(m-1)! + D_1(m-2)! + \ldots D_{m-2}(1)! + D_{m-1}(0)! \tag{37}$$

The constants D_i, shown in Equation (37), are unique according to Euclid's division algorithm [47] and satisfy Equation (38).

$$0 \leq D_i < (m-i) \text{ for } 0 \leq i \leq (m-2) \tag{38}$$

The development of a permutation algorithm is possible with this information. Using the constants, D_i of Equation (37) can be scrambled with an array of m different elements [48]. Further, this algorithm has a significant property defining a one-to-one function [48]. This is because two different integers $n_1 \neq n_2$ produce two distinct permutations.

3.2. Similarity Parameter

The additive, multiplicative, occlusion, and proposed χ^2 noises damage the encrypted images. Subsequently, the difference between the damaged decrypted image and the original one is quantified to know the impact of this. The parameter UACI is suggested by authors in Equation (39). For a SP_c range between 0% and 100%, the constant 2.994 appears.

$$SP_c = |100 - UACI_c(2.994)|\% \tag{39}$$

SP_c measures the similarity between two images. If the compared images are equal, then UACI = 0, and SP_c = 100%. Otherwise, if the original image is compared with its well-encrypted image, then $UACI_c \approxeq 33.4$, and $SP_c \approxeq 0\%$. Additionally, it is possible to quantify how much the median 5×5 filter improves the sharpness of a damaged image.

3.3. Encryption Procedure

A high-level description of the encryption procedure is presented, and later the generated elements in the encryption: boxes, permutations, and schedule keys.

AICLDH has a fourteen-round symmetric cryptosystem [49]. Each round uses a different box of type $S - 8 \times 8$ *box* [50], which is dynamic since it changes in each process. Before starting the first round, a permutation with a number of elements equal to the image size is executed. It is also dynamic and works over all the image pixels, altering the pixel positions in the entire image. The schedule keys involved have a size equal to the image size. It is important to note that boxes, permutation, and schedule keys change in each image encryption, even if it is the same image but at a different time.

Below, the steps in every round of the encryption algorithm are described.

1. First, the permutation P is applied to the original image. Afterward, the xor operation is applied over the permuted image and the first round key. Later, the resulting chain is divided into blocks of one byte. Immediately, the substitution operation is carried out with the first box. The substitution operation is carried out in the same way as in AES [50].
2. From rounds 2 to 13, the xor operation is applied between the output of the previous round and the corresponding schedule key. Subsequently, the substitution is operated with the corresponding $S - 8 \times 8$ *box*.
3. In the last round, the xor operation is executed using the output string of round 13 and the schedule key 14. Then, this previous result is divided into blocks of one byte, and the substitution operation is applied to the last box. Finally, a xor is performed between the string resulting from the substitution operation and the schedule key 15. This final result is the encrypted image.

Once the high-level description has been made, a detailed description of the elements involved in the encryption process is explained. Before this, we will see the pre-processing that must be performed.

3.4. Pre-Processing

1. The Sender (A) generates a random integer a_A such that $1 < a_A < p^* - 1$; considering that $p^* \approxeq 2^{1024}$. Consequently, the calculation $\beta_A = \alpha^{a_A} \bmod p^*$ is performed. Finally, β_A is sent to the receiver (B). Similarly, the receiver (B) randomly obtains an integer a_B with $1 < a_B < p^* - 1$, then computes the value $\beta_B = \alpha^{a_B} \bmod p^*$ and transmits it to the sender. From here, the sender has β_B and performs the following calculation: $\beta = \beta_B^{a_A} \bmod p^*$. Note that the receiver can also generate the integer β. Consequently,

both the sender and receiver have the same information. As can be seen, β_A, β_B are public; although, β is private.

2. At this moment 128 strings $\beta_i = \alpha^i \times \beta \bmod p^*$ are built, with $i = 1 \cdots 128$. Note that each β_i is taken as a 1024-bits message. While if a β_i has a length less than 1024 bits, then zeros are added to the left until the string reaches that magnitude.
3. The sender randomly generates two positive integers that satisfy $1 < C^1, C^2 < 2^{512}$ for sending them to the receiver. In addition, using the Block-Chain technique, two 512-bit binary strings are associated with each integer C^1, C^2. Afterward, they are divided into blocks of 8 bits, resulting in 64 blocks of 8 bits for each integer. In fact, the result of the joint of both strings is 128 strings of one byte. The number associated with each block is b_i for $i = 1, 2, \cdots, 64$, and the range of b_i is from 0 to 255.
4. This research proposes to apply the Hash SHA-512 function to the chain β_i. If $b_i = 0$, the Hash-SHA algorithm is applied once to the chain β_i. In contrast, if $0 < b_i \leq 255$, the Hash-SHA algorithm is executed $b_i + 1$ times in a chained manner [51], starting from the block β_i. Therefore, as a result of this process, there are 128 strings of 512 bits, and each of them is associated with an integer. It is important to say that the sender publishes these strings. On the other hand, the receiver can replicate the procedure to know the associated values with each block of the 512-bit strings. Consequently, the receiver (B) knows the integers C^1, C^2.

3.5. Generation of the $S - 8 \times 8$ Boxes

Once the integers C^1, C^2 are known, the sender and receiver proceed as follows to build the boxes since the boxes $S - 8 \times 8$ are a permutation of 256 elements; that is, from 0x00 to 0xff. In this sense, C_2 from Equation (17) takes the constant value C^1. Subsequently, the operation $(0.999924)C_2e$ is carried out; considering that the constant $e = 2.7182 \cdots$ [52]. Afterward, the string of bits that is to the right of the decimal point is taken, and subsequently, we proceed as follows:

1. The bits from the decimal point to the right are divided into blocks of one byte that simultaneously compose an integer. In this way, let us denote c_i as the associated integer with the i-th byte. For the first box $i = \{0, 1, \cdots, 255\}$.
2. The constants D_i, to build the first box, are obtained by the mean of $D_i = c_i \bmod. 256 - i$. Where the D_i corresponds to the constants of the algorithm developed in Section 3.
3. Through the constants, D_i, it is possible to generate a box $S - 8 \times 8$, considering that a box is a permutation of 256 elements. While to obtain the second box, the bytes from 256 to 511 are taken. Following this method, 14 boxes can be built, one for each round.

3.6. Generation of the Permutation and Schedule Key

The permutation generation is the following:

1. The integer C_3 of Equation (17) is replaced by the value C^2. Then, it is operated $(0.012315)C_3e$; where, as before, the constant $e = 2.7182 \cdots$. From this result, we will only take the bits from the decimal point to the right.
2. The bit string after the decimal point is divided into 8-bit blocks, that is, in bytes. In addition, the number of pixels of the image is l. We proceed as follows: building a string with the first three bytes, that is, taking bytes number 0, 1, and 2. In this way, let be c_0, the associated integer to this string. Then, the calculation $D_0 = c_0 \bmod. l$ is performed. For obtaining D_1, a byte shift to the right is performed. Now, bytes 1, 2, and 3 are selected to make a new string. As before, this 24-bit string has another associated integer called c_1. Subsequently, the calculation $D_1 = c_1 \bmod l - 1$ is performed. In general, to calculate D_i, the procedure is to compute $D_i = c_i \bmod. l - i$. It is important to remember that for obtaining c_i, i-shifts of one byte are made to the right of the decimal point.

3. Once the D_i has been calculated for $i = 0, 1, \cdots, l-2$, it is possible to use the algorithm indicated in Section 3 to permute the l pixels of the image. Let us denote this permutation as P.

On the other hand, the round keys procedure is the following:

The value of C_3 in Equation (17) is replaced by C^2. Then, the $(0.012315)C_3e$ operation is computed; as before, the constant $e = 2.7182\cdots$. Henceforth from the product, the bits to the right of the decimal point are taken. To generate the first key, a string of bits is selected whose length is equal to the image bits size; in this case, for 512 × 512 pixels. The present work proposes this chain as k_1, the first round key. In this way, to generate the other round keys, for example, the i-th round key, the process is similar. Apply $i-1$ shifts of one byte after the decimal point; from there to the right, a string takes the number of bytes equal to the size of the image. Finally, the resulting string would be the key k_i.

At this point, the authors consider it pertinent to show the hexadecimal values used in this research: p_1, p_2, k, p^*, and α.

p_1 = e13e0ceb0f0798be0a5d5f8ad448524b7c1fe933440f
6fa2923f440ce749a9290927d1a3eeb44abe41258be7cf2ef
17b715a2be5a6a542eb9ea6a7da039d4dab

p_2 = 9dffc7e2f4ed8acf7dac5ad69c88b44e5a36f47a567
7eb29e7d71b1e656f652eb490a46360931402c050671245
bc5ca9c95dee3b14f3b4121d46b31c95a9ef29

k = 1734

Similarly, the prime p^* and generator α are shown:

p^* = 3ad9dcc2ec1c5e61c2449c2e8907bac8233b13fe221
41d9cf989ba34b0c387bf2058b247c3bc9fcaa3f3daaefa49
728b2d976a47b3769ca2101744cab23355e3a949c0f6262
bcde56dea2bbc969f03f2a558f871b0a7e05492761057ef6
70723a849dafec51c62187c5f1da8dc523382cbee124d12e
ebfe510ad1c1d41c675adc93

α = 326ff52fdb98aaf27594369b807e538921805f86e563
712c7616cb432ea6ae480e772dbc521782508af64e67573
17ab6818d598ed168c14ed2952c7db0042980b277b6949
fa475f53c8a7b2387344eb64b770909f4c2ec68e3f8eda66
05912f872566c02c0e06953b5b24e23c119d135068fa5c3
960fa59e55f7b1a04afd2450

The values a_A, a_B can be chosen randomly or fixed for a period in each communication between the sender and the receiver.

3.7. Elements for Testing

The images used to evaluate the AICLDH cryptosystem are presented in Figure 2. In addition, two images are examined: one completely in black and another in white. It is pointed out that this research uses images of 512 × 512 pixels. However, it is possible to work with images of different dimensions. For doing so, the size of the binary string would only increase or decrease after the decimal point.

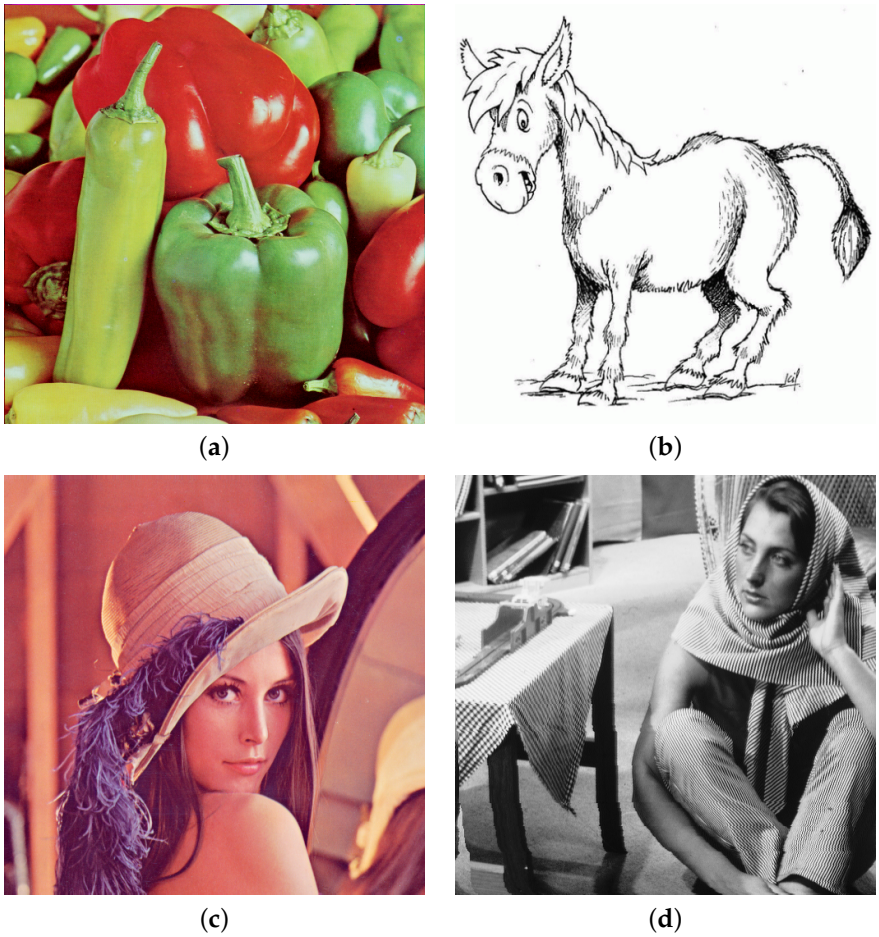

Figure 2. Images used for testing AICLDH: (**a**) Peppers; (**b**) Donkey; (**c**) Lena; (**d**) Barbara.

In the case of the Lena image, it is widely tested in image encryption developments. In addition, the results of the AES-CBC image encryption algorithms are compared with AICLDH [53] when noise is applied to the encrypted figures. Later, it is shown that the impact of noise on encrypted images is less drastic in the AICLDH algorithm than in AES-CBC.

4. Applying Noise to Encrypted Images

This work applies four noises to the encrypted images: occlusion, additive, multiplicative, and one proposed using the χ^2 distribution. The goal is to test the resistance of the AICLDH encryption algorithm to damage attacks on encrypted images. In this sense, the noise produced by the random variable χ^2 is described below.

4.1. The χ^2 Noise

For the χ^2 noise, Equation (36) is used and previously described in Section 2. This random variable approximates a normal distribution with $\mu = 255$ and standard deviation $\sigma = 22.58$. The operation starts choosing n pixels randomly of the encrypted image in the spatial domain. Each of them is associated with a frequency $f_c(x, y)$, where c indicates the color, and (x, y) is the pixel position. The range of this variable is $0 \leq f_c(x, y) \leq 255$.

Later, a value $z_c(x, y) \sim \mathrm{N}(0, 1)$ is chosen randomly for each pixel and color. Then, the variable $f'_c(x, y)$ is calculated according to Equation (40). In the case of $z_c < -3$, the value of -3 is assigned to the variable. If $z_c > 3$, it is given the value of 3. In such a way, $-3 \leq z_c(x, y) \leq 3$.

$$f'_c(x, y) = 255 + z_c(x, y)22.58 \tag{40}$$

Afterward, the variable $f'_c(x,y)$ is discretized using floor and ceiling functions. The symbol $\lfloor\ \rfloor$ is applied if the decimal part of $f'_c(x,y)$ is less than or equal to 0.5. Then, the integer part of $f'_c(x,y)$ is taken, and the discrete value of the frequency, $f'_{dc}(x,y)$, is obtained according to Equation (41). When the decimal part is greater than 0.5, the second symbol $\lceil\ \rceil$ is applied. In this case, one unit is added to the integer part, and the discrete value of the frequency is computed according to Equation (42).

$$f'_{dc}(x,y) = \lfloor 255 + z_c(x,y)22.58 \rfloor \mod 256 \tag{41}$$

If a decimal fraction is greater than 0.5, then $f'_{dc}(x,y)$ is calculated according to Equation (42).

$$f'_{dc}(x,y) = \lceil 255 + z_c(x,y)22.58 \rceil \mod 256 \tag{42}$$

In this paper, it is proposed to substitute the value of $f_c(x,y)$ for $f'_{dc}(x,y)$ as the case may be. Additionally, this type of noise mostly replaces values located at the level extremes. In other words, values from the range 0–64 or 191–255.

4.2. The Occlusion Noise

The occlusion noise application in this paper is as follows: pixels of the encrypted image are selected to form a concentric parallelogram. Subsequently, the frequencies from pixels of this parallelogram region are replaced by the ones in the cherry color. In general, it can be another color. This procedure is applied in recent research [54].

4.3. The Additive and Multiplicative Noises

A high-level description of additive and multiplicative noise is given below. Similarly to the other noises, n points of the encrypted image are chosen randomly. Furthermore, each point has an associated frequency $f_c(x,y)$ such that $0 \leq f_c(x,y) \leq 255$ and c indicates the basic color.

For the additive noise, an integer $\tau_c(x,y)$ is chosen randomly for each point and color. The discrete value $f'_{dc}(x,y)$ is calculated according to Equation (43); subsequently, $f_c(x,y)$ is changed by the discrete value of $f'_{dc}(x,y)$.

$$f'_{dc}(x,y) = [f_c(x,y) + \tau_c(x,y)] \mod 256 \tag{43}$$

To apply the multiplicative noise in the same way as additive noise, an integer $\tau_c(x,y)$ is selected randomly. Therefore, the frequency $f'_{dc}(x,y)$ is calculated, taking into account Equation (44). Later, the value of $f_c(x,y)$ is replaced by $f'_c(x,y)$.

$$f'_{dc}(x,y) = [f_c(x,y) \times \tau_c(x,y)] \mod 256 \tag{44}$$

5. Results

This Section begins showing Figure 3. Figure 3a shows the original image of Lena, while Figure 3b shows Lena encrypted with AICLDH. Figure 3c–e are the histograms of the basic colors of Figure 3b. AICLDH was made in Java programming language, using its BigInteger library for numerical operations. The cipher execution was 0.4 s on a computer with Windows 11 over an i9-10900K CPU with ten cores.

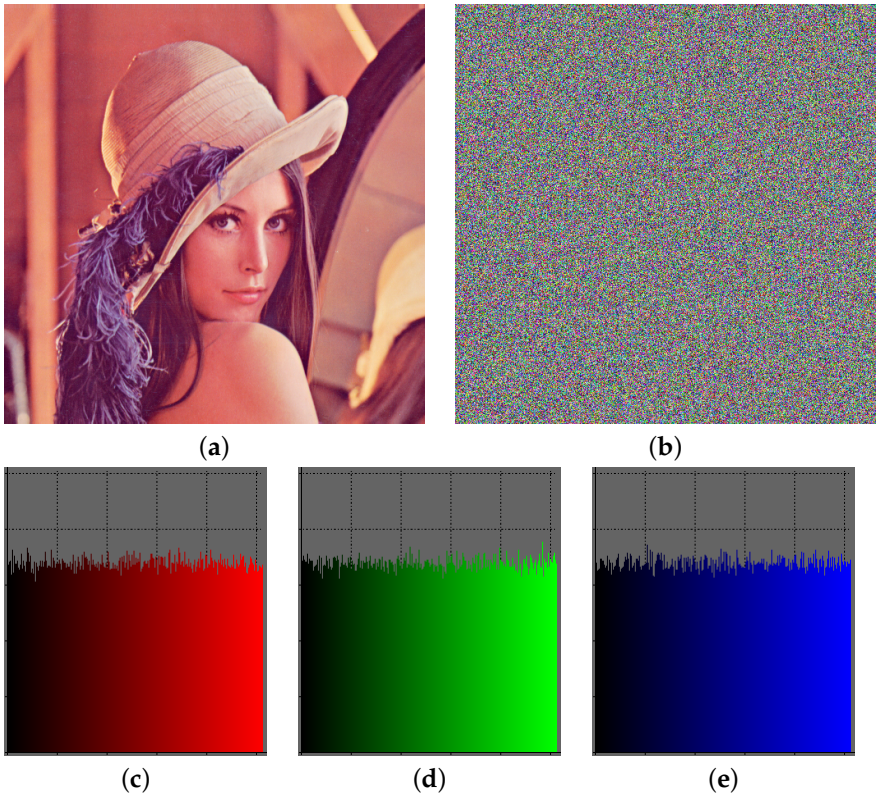

Figure 3. Lena image: (**a**) Original Lena image; (**b**) Lena ciphered image with AICLDH; (**c**) Red color histogram of (**b**); (**d**) Green color histogram of (**b**); (**e**) Blue color histogram of (**b**).

The numerical results begin the evaluation of encrypted images without damage. In addition, first, the results of the instruments that are not statistical hypothesis tests are shown: Correlation, Entropy, UACI, NPCR, AC, Homogeneity, Energy, and Contrast. Later, the results that use the statistical hypothesis test are presented: the Goodness-of-Fit test and the Discrete Fourier Transform.

5.1. Correlation and Entropy

The correlation and entropy results are shown in Tables 1 and 2 to measure the randomness of the encrypted images. This corresponds to the encrypted images from Figure 2. While the results of the NPCR, UACI, and AC parameters appear in Tables 3–5, respectively. In the same sense, the energy, contrast, and homogeneity results are presented in Tables 6–8.

Table 1. Correlation coefficient C of the encrypted test images of Figure 2.

Direction	Color	Peppers	Donkey	Lena	Barbara
Horizontal	Red	−0.00448	−0.00134	0.00116	−0.00351
	Green	0.00020	0.00043	0.00104	−0.00301
	Blue	0.00005	0.00601	0.00635	−0.00414
Vertical	Red	0.0043	−0.00086	−0.00128	−0.00369
	Green	0.00068	−0.00322	0.00523	0.00194
	Blue	0.00067	−0.00760	0.00088	−0.00646
Diagonal	Red	0.00777	−0.00138	−0.00152	−0.00211
	Green	−0.00152	0.00136	−0.00503	−0.00344
	Blue	0.00220	−0.00103	0.00011	−0.00550

Table 2. Entropy of test images of Figure 2 after encryption.

Color	Peppers	Donkey	Lena	Barbara
Red	7.99929	7.99937	7.99932	7.99927
Green	7.99926	7.99930	7.99938	7.99934
Blue	7.99935	7.99930	7.99938	7.99918

Table 3. NPCR of the test images after encryption.

Color	Peppers	Donkey	Lena	Barbara
Red	99.623	99.598	99.609	99.632
Green	99.605	99.617	99.614	99.613
Blue	99.597	99.603	99.625	99.618

Table 4. UACI of the test images after encryption.

Color	Peppers	Donkey	Lena	Barbara
Red	33.581	33.482	33.438	33.522
Green	33.496	33.462	33.486	33.471
Blue	33.443	33.496	33.516	33.453

Table 5. AC of the test images after encryption.

Color	Peppers	Donkey	Lena	Barbara
Red	50.01	49.98	49.96	50.04
Green	50.01	49.97	50.01	49.97
Blue	49.94	49.96	50.00	49.99

Table 6. Energy of encrypted images of Figure 2.

Color	Peppers	Donkey	Lena	Barbara
Red	0.01563	0.01563	0.01563	0.01563
Green	0.01563	0.01563	0.01563	0.01563
Blue	0.01563	0.01563	0.01563	0.01563

Table 7. Contrast of encrypted images of Figure 2.

Color	Peppers	Donkey	Lena	Barbara
Red	10.54102	10.42625	10.46554	10.47795
Green	10.49049	10.49013	10.49560	10.45892
Blue	10.51685	10.50117	10.52990	10.47287

Table 8. Homogeneity of Figure 2 images after encryption.

Color	Peppers	Donkey	Lena	Barbara
Red	0.38886	0.39047	0.38974	0.389900
Green	0.38924	0.38935	0.38920	0.389762
Blue	0.38892	0.38923	0.38895	0.389790

5.2. Discrete Fourier Transform and the Proposal Test

The results after applying the DFT instrument to the AICLDH-encrypted images of Figure 2 are shown in Table 9. Likewise, an additional test based on the χ^2 distribution is proposed. The results of this parameter are in Table 10.

Table 9. The randomness measurement using the Discrete Fourier Transform (✓ Accept, x Reject), with $\alpha = 0.01$.

Color	Peppers	Donkey	Lena	Barbara
Red	0.296/✓	0.707/✓	0.674/✓	0.759/✓
Green	0.072/✓	0.064/✓	0.864/✓	0.551/✓
Blue	0.151/✓	0.602/✓	0.256/✓	0.174/✓

Table 10. Results of the Goodness-of-Fit test (✓ Accept, x Reject), with $\alpha = 0.01$.

Color	Peppers	Donkey	Lena	Barbara
Red	272.9/✓	274.2/✓	276.3/✓	239.9/✓
Green	275.2/✓	241.1/✓	220.7/✓	273.1/✓
Blue	240.2/✓	251.6/✓	262.3/✓	274.2/✓

5.3. Images Black and White

Since AICLDH has a symmetric cryptosystem, the following experiment is detailed below: a black image and a white image of size 512 × 512 are encrypted. Subsequently, the encrypted figures are evaluated according to the NPCR, UACI, and AC parameters. The results are in Table 11.

Table 11. AC, NPCR, and UACI values for the completely black and completely white images.

Parameter	Color	Black Image	White Image
AC	Red	50.004	49.957
	Green	50.027	49.952
	Blue	49.990	49.968
NPCR	Red	99.617	99.599
	Green	99.595	99.603
	Blue	99.590	99.598
UACI	Red	33.522	33.421
	Green	33.418	33.452
	Blue	33.506	33.407

5.4. Encrypted Images with Noise

Below, the results explore the images encrypted with damage after noise application. The experiment considers Figures 4 and 5. While Figure 4 shows the result of the following procedure. The AICLDH algorithm encrypts Peppers' image in Figure 2. Then, the encrypted image is applied with the multiplicative noise of size 40% and the figure encrypted with damage is decrypted with AICLDH. Finally, Figure 5 is the result of the same previous procedure, with only one difference. The original image and the one encrypted with damage are encrypted and decrypted with the AES-CBC algorithm, respectively.

In Section 2, the median filter 5 × 5 was described to improve the encryption process with noise. In this way, Figure 6a presents the Peppers image encrypted and decrypted with AICLDH after the χ^2 noise application at 40%. In contrast, Figure 6b shows the Peppers image after the median filter 5 × 5 application to the deciphered image with damage. Consequently, it is possible to observe how the sharpness of the damaged image improves.

On the other hand, Table 12 shows the SP parameter evaluations. It is when the images of Figure 2 encrypted with AICLDH were damaged with the χ^2 noise of different sizes. Finally, Table 13 shows the following results. The images were encrypted with AICLDH in Figure 2, after a 40% size damage applied using the different types of noise proposed in this research, and then the decrypted images with noise after the 5 × 5 filter application are shown.

(**a**) (**b**)

Figure 4. Peppers image deciphered with AICLDH: (**a**) Original image; (**b**) Deciphered image when a multiplicative noise of 40% size was applied after encryption.

(**a**) (**b**)

Figure 5. Peppers image deciphered with AES-CBC: (**a**) Original image; (**b**) Deciphered image when a multiplicative noise of 40% size was applied after encryption.

(**a**) (**b**)

Figure 6. Peppers image deciphered: (**a**) Just deciphered with noise and AICLDH; (**b**) Deciphered and filtered using median filter of 5×5.

Table 12. SP for different damage sizes of the testing images after encryption, using χ^2 noise damage.

Color	Size Noise	Peppers	Donkey	Lena	Barbara
Red	20%	82.77	72.24	80.19	82.06
	30%	73.70	57.74	70.60	73.20
	40%	66.17	44.65	60.42	64.31
	50%	57.69	30.30	50.11	55.78
Green	20%	79.91	71.95	81.57	82.10
	30%	69.16	57.35	72.70	73.34
	40%	60.37	44.00	63.46	64.28
	50%	50.20	29.44	53.63	55.76
Blue	20%	79.37	72.14	83.38	82.06
	30%	68.27	57.66	75.34	73.26
	40%	59.38	44.42	67.00	64.30
	50%	49.10	30.16	58.20	55.79

Table 13. SP after a 5×5 median filter was applied to encrypted images with 50% damage from different noise sources.

Color	Noise Type	Peppers	Donkey	Lena	Barbara
Red	Occlusion	91.30	69.41	88.18	83.45
	Additive	91.35	69.78	88.30	83.20
	Multiplicative	91.51	69.76	89.12	83.95
	Chi-square	91.56	68.53	87.57	83.66
Green	Occlusion	86.54	68.84	87.83	83.38
	Additive	86.52	69.02	87.78	83.24
	Multiplicative	87.16	69.22	88.45	83.94
	Chi-square	87.26	67.67	87.50	83.57
Blue	Occlusion	86.11	69.46	90.70	83.45
	Additive	86.04	69.71	90.75	83.33
	Multiplicative	86.87	70.00	90.94	84.02
	Chi-square	86.95	68.40	90.48	83.58

6. Discussion

Since security is vital for any cryptosystem, it starts with the AICLDH resistance against different known attacks. With this objective in mind, attacks are divided into three categories. Those that apply to the ElGamal cryptosystem, those that impact the proposed symmetric cryptosystem, and those that damage encrypted images.

The attack on the ElGamal cryptosystem aims to obtain the private key, a, when the public key, β, is known. In this sense, some generic algorithms have been realized, which have a complexity of $O(\sqrt{p})$, to obtain a solution. In addition, there is another well-known procedure to obtain the private key known as the Pohlig–Hellman attack, and also, in this case, the complexity is $O(\sqrt{p})$ [55]. On the other hand, the value of p used in this investigation is approximately 2^{1024}. As a result, the attack complexity would be $O(2^{512})$. In conclusion, this type of attack, at least for the moment, does not affect the proposed cryptosystem [56].

The security benefits of the proposed AICLDH symmetric algorithm are described below. The substitution boxes, 8×8 $S-box$, used in the encryption procedure are dynamic. In other words, they are unknown before encryption. In addition, each round employs a different box. In conclusion, it is impossible to accomplish linear and algebraic attacks; at least as we know it [57]. On the other hand, a brute-force attack requires knowing C^1, C^2. However, that problem has a complexity of 2^{1024}. Hence, the brute-force attack cannot be performed successfully. It is important to remember that AES has at most 2^{256} and has not been broken yet. In addition, it has been reported that in private key algorithms, the impact due to the appearance of quantum computers will be less drastic [7]. Moreover, because

the proposed symmetric cryptosystem was built using the ElGamal cryptosystem, it makes it possible for ElGamal to distribute its keys.

In the same way, Table 11 shows that the values of the NPCR, UACI, and AC parameters are adequate to avoid the differential attack. Other attacks tested on encrypted images are the noises: additive, multiplicative, occlusion, and the χ^2 proposed. Moreover, the affected area is available in four sizes; 20%, 30%, 40%, and 50%.

In addition, the SP parameter evaluates the sharpness loss of decrypted images with damage. The result appears in Table 12. Furthermore, Figure 5a shows the decrypted image when the original was encrypted with the proposed algorithm and then χ^2 noise application of size 50%. While Figure 5b considers the same procedure, although the original image is encrypted with AES-CBC. As can be noticed, the image encrypted with the AICLDH algorithm offers better clarity than the one encrypted with AES-CBC. Under those circumstances, Table 13 shows the results when a median filter of 5×5 is applied to the decrypted images with damage. In this way, the SP parameter evaluates the increase in sharpness, which can be higher than 91% in some cases. With attention to black and white images are more affected than others by the noise. One example is the Donkey image.

The image encryption quality is explored in two directions: according to the DFT result and the proposed goodness-of-fit test. The second includes correlation, entropy, NPCR, UACI, AC, homogeneity, contrast, and energy parameters. As can be seen, in both directions, the results show that the encryption of the images is robust. In fact, Table 14 presents a comparison of the entropy in the images encrypted with AICLDH and other works.

Table 14. Entropy comparison between the AICLDH algorithm and other works.

Image	Algorithm	Entropy
Peppers	AICLDH	7.9993
	Ref. [58]	7.996
	Ref. [59]	7.9938
Lena	AICLDH	7.99935
	Ref. [60]	7.9975
	Ref. [14]	7.9953

7. Conclusions

The present work proposed a hybrid cryptosystem based on the Diffie–Hellman protocol. For this reason, it can distribute its keys and sign. As part of the hybrid cryptosystem, using the shared key of the Diffie–Hellman protocol and Lorenz Equations was possible to build a fourteen-round private key cryptosystem. Its substitution boxes, permutation, and round keys are dynamic; they are all different in each encryption process. The most relevant reason for choosing a symmetric system is that this type of development will be less affected in the quantum era, with a less drastic impact. This hybrid algorithm is robust because it resists differential, linear, algebraic, and brute-force attacks. The encryption quality evaluation considered: entropy, correlation, DTF, goodness-of-fit test, NPCR, UACI, AC, homogeneity, energy, and contrast, and the results were satisfactory in all, which is why the AICLDH is safe. To conclude this work, comparing the encrypted images with noise, it can be observed in Figures 4 and 5 that AICLDH is superior to AES-CBC. Furthermore, we say that future work intends to distribute the seed using post-quantum algorithms.

Author Contributions: Conceptualization, methodology, formal analysis, investigation, visualization, writing—review and editing, data curation, software, validation, writing—original draft preparation V.M.S.-G., R.F.-C. and M.A.C.-L.; resources, supervision, project administration, funding acquisition, R.F.-C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded in part by the economic support program of the Comisión de Operación y Fomento de Actividades Académicas (COFAA) and the Secretaría de Investigación y Posgrado (SIP) of the Instituto Politécnico Nacional under grant 20221263.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon request.

Acknowledgments: The authors would like to thank the Instituto Politécnico Nacional of México (Secretaría Académica, Comisión de Operación y Fomento de Actividades Académicas COFAA, SIP, and CIDETEC), and the CONACyT (SNI) for their support in the development of this work.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AICLDH	Algorithm of Image Cipher using Lorenz equations and Diffie–Hellman protocol
AES	Advanced Encryption Standard
DFT	Discrete Fourier Transform
NPCR	Number of Pixels Change Rate
UACI	Unified Average Changing Intensity
AC	Avalanche Criteria
SP	Similarity Parameter

References

1. Park, B.; Korbach, A.; Brünken, R. Does thinking-aloud affect learning, visual information processing and cognitive load when learning with seductive details as expected from self-regulation perspective? *Comput. Hum. Behav.* **2020**, *111*, 106411. [CrossRef]
2. Nadhir, N.; Ali, S.; Abdelkrim, G. Medical Image Watermarking Scheme in Transform Domain based on Asymmetric crypto-system and Arnold Chaotic Map. In Proceedings of the 2021 44th International Conference on Telecommunications and Signal Processing (TSP), Brno, Czech Republic, 26–28 July 2021; pp. 267–272. [CrossRef]
3. Habib, M.A.; Md. Rokibul Alam, K.; Morimoto, Y. A Secure Medical Record Sharing Scheme Based on Blockchain and Two-fold Encryption. In Proceedings of the 2022 25th International Conference on Computer and Information Technology (ICCIT), Cox's Bazar, Bangladesh, 17–19 December 2022; pp. 78–83. [CrossRef]
4. Ahmad, A.; Abuhour, Y.; Younisse, R.; Alslman, Y.; Alnagi, E.; Al-Haija, Q.A. MID-Crypt: A Cryptographic Algorithm for Advanced Medical Images Protection. *J. Sens. Actuator Netw.* **2022**, *11*, 24. [CrossRef]
5. Banik, A.; Laiphrakpam, D.S.; Agrawal, A.; Patgiri, R. Secret image encryption based on chaotic system and elliptic curve cryptography. *Digit. Signal Process.* **2022**, *129*, 103639. [CrossRef]
6. Qobbi, Y.; Abid, A.; Jarjar, M.; Kaddouhi, S.E.; Jarjar, A.; Benazzi, A. Adaptation of a genetic operator and a dynamic S-box for chaotic encryption of medical and color images. *Sci. Afr.* **2023**, *19*, e01551. [CrossRef]
7. Stinson, D.R.; Patterson, M. *Cryptography: Theory and Practice*, 4th ed.; CRC Press: Boca Raton, FL, USA, 2018; pp. 116–122.
8. Chowdhary, C.L.; Patel, P.V.; Kathrotia, K.J.; Attique, M.; Perumal, K.; Ijaz, M.F. Analytical Study of Hybrid Techniques for Image Encryption and Decryption. *Sensors* **2020**, *20*, 5162. [CrossRef]
9. Ye, G.; Jiao, K.; Wu, H.; Pan, C.; Huang, X. An Asymmetric Image Encryption Algorithm Based on a Fractional-Order Chaotic System and the RSA Public-Key Cryptosystem. *Int. J. Bifurc. Chaos* **2020**, *30*, 2050233. [CrossRef]
10. Khalid, I.; Shah, T.; Eldin, S.M.; Shah, D.; Asif, M.; Saddique, I. An Integrated Image Encryption Scheme Based on Elliptic Curve. *IEEE Access* **2023**, *11*, 5483–5501. [CrossRef]
11. Ye, G.; Liu, M.; Wu, M. Double image encryption algorithm based on compressive sensing and elliptic curve. *Alex. Eng. J.* **2022**, *61*, 6785–6795. [CrossRef]
12. Alohali, M.A.; Aljebreen, M.; Al-Mutiri, F.; Othman, M.; Motwakel, A.; Alsaid, M.I.; Alneil, A.A.; Osman, A.E. Blockchain-Driven Image Encryption Process with Arithmetic Optimization Algorithm for Security in Emerging Virtual Environments. *Sustainability* **2023**, *15*, 5133. [CrossRef]
13. Lu, Q.; Yu, L.; Zhu, C. Symmetric Image Encryption Algorithm Based on a New Product Trigonometric Chaotic Map. *Symmetry* **2022**, *14*, 373. [CrossRef]
14. Hazzazi, M.M.; Attuluri, S.; Bassfar, Z.; Joshi, K. A Novel Cipher-Based Data Encryption with Galois Field Theory. *Sensors* **2023**, *23*, 3287. [CrossRef]
15. Chen, Z.; Ye, G. An asymmetric image encryption scheme based on hash SHA-3, RSA and compressive sensing. *Optik* **2022**, *267*, 169676. [CrossRef]
16. Yousif, S.F.; Abboud, A.J.; Radhi, H.Y. Robust Image Encryption with Scanning Technology, the El-Gamal Algorithm and Chaos Theory. *IEEE Access* **2020**, *8*, 155184–155209. [CrossRef]

17. Singh, K.N.; Singh, A.K. Towards Integrating Image Encryption with Compression: A Survey. *ACM Trans. Multimed. Comput. Commun. Appl.* **2022**, *18*, 89. [CrossRef]
18. He, J.; Huang, S.; Tang, S.; Huang, J. JPEG Image Encryption with Improved Format Compatibility and File Size Preservation. *IEEE Trans. Multimed.* **2018**, *20*, 2645–2658. [CrossRef]
19. Archivo General de la Nación. Manual de digitalización de documentos. *Boletín Arch. Gen. Nación* **2022**, *9*, 41–117.
20. Deb, S.; Behera, P.K. Design of key-dependent bijective S-Boxes for color image cryptosystem. *Optik* **2022**, *253*, 168548. [CrossRef]
21. Kumari, A.; Pranav, P.; Dutta, S.; Chakraborty, S. Empirical and Statistical Comparison of RSA and El-Gamal in Terms of Time Complexity. In Proceedings of the International Conference on Intelligent Cyber Physical Systems and Internet of Things (ICoICI 2022), Coimbatore, India, 11–12 August 2022; pp. 111–120.
22. Akhmetzyanova, L.; Alekseev, E.; Babueva, A.; Smyshlyaev, S. On the (Im)possibility of ElGamal Blind Signatures. Cryptology ePrint Archive, Paper 2022/1128. 2022. Available online: https://eprint.iacr.org/2022/1128 (accessed on 8 May 2023).
23. Manz, O. *Encrypt, Sign, Attack*, 1st ed.; Springer: Berlin/Heidelberg, Germany, 2022; pp. 53–86.
24. Drăgulinescu, A. Optical Correlators for Cryptosystems and Image Recognition: A Review. *Sensors* **2023**, *23*, 907. [CrossRef]
25. Malallah, F.L.; Abduljabbar, A.I.; Shareef, B.T.; Al-Janaby, A.O. QR Code Encryption for improving Bank information and Confidentiality. In Proceedings of the 2023 27th International Conference on Information Technology (IT), Zabljak, Montenegro, 15–18 February 2023; pp. 1–4. [CrossRef]
26. Li, T.; Yan, W.; Chi, Z. A new image encryption algorithm based on optimized Lorenz chaotic system. *Concurr. Comput.* **2022**, *34*, e5902. [CrossRef]
27. Zhao, L.; Zhang, J.; Jing, H.; Wu, J.; Huang, Y. A Blockchain-Based cryptographic interaction method of digital museum collections. *J. Cult. Herit.* **2023**, *59*, 69–82. [CrossRef]
28. Panario, D.; Perin, L.P.; Stevens, B. Comparing balanced sequences obtained from ElGamal function to random balanced sequences. *Cryptogr. Commun.* **2023**, *15*, 675–707. [CrossRef]
29. Zhao, F.; Guo, J.; Zhang, L.; Han, W. Research on Improved Double RSA Algorithm Based on RSA. In Proceedings of the International Conference on Computer Engineering and Networks (CENet 2022), Haikou, China, 4–7 November 2022; pp. 1204–1211. [CrossRef]
30. Adeniyi, E.A.; Falola, P.B.; Maashi, M.S.; Aljebreen, M.; Bharany, S. Secure Sensitive Data Sharing Using RSA and ElGamal Cryptographic Algorithms with Hash Functions. *Information* **2022**, *13*, 442. [CrossRef]
31. Nannipieri, P.; Bertolucci, M.; Baldanzi, L.; Crocetti, L.; Di Matteo, S.; Falaschi, F.; Fanucci, L.; Saponara, S. SHA2 and SHA-3 accelerator design in a 7 nm technology within the European Processor Initiative. *Microprocess. Microsyst.* **2021**, *87*, 103444. [CrossRef]
32. Srivastava, V.; Baksi, A.; Debnath, S.K. An Overview of Hash Based Signatures. Cryptology ePrint Archive, Paper 2023/411. 2023. Available online: https://eprint.iacr.org/2023/411 (accessed on 8 May 2023).
33. Zhou, S.; He, P.; Kasabov, N. A Dynamic DNA Color Image Encryption Method Based on SHA-512. *Entropy* **2020**, *22*, 1091. [CrossRef] [PubMed]
34. Erkan, U.; Toktas, A.; Lai, Q. 2D hyperchaotic system based on Schaffer function for image encryption. *Expert Syst. Appl.* **2023**, *213*, 119076. [CrossRef]
35. Shannon, C. A Mathematical Theory of Communication. *Bell Syst. Tech. J.* **1948**, *27*, 379–423. [CrossRef]
36. Parida, P.; Pradhan, C.; Alzubi, J.A.; Javadpour, A.; Gheisari, M.; Liu, Y.; Lee, C.C. Elliptic curve cryptographic image encryption using Henon map and Hopfield chaotic neural network. *Multimed. Tools Appl.* **2023**, *86*, 1–26. [CrossRef]
37. Jirjees, S.W.; Alkalid, F.F.; Shareef, W.F. Image Encryption Using Dynamic Image as a Key Based on Multilayers of Chaotic Permutation. *Symmetry* **2023**, *15*, 409. [CrossRef]
38. Zou, H.; Chen, G. Reversible data hiding in encrypted image with local-correlation-based classification and adaptive encoding strategy. *Signal Process.* **2023**, *205*, 108847. [CrossRef]
39. Bassham, L.; Rukhin, A.; Soto, J.; Nechvatal, J.; Smid, M.; Leigh, S.; Levenson, M.; Vangel, M.; Heckert, N.; Banks, D. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*; Technical Report; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2010.
40. Zhu, Y.; Wang, C.; Sun, J.; Yu, F. A Chaotic Image Encryption Method Based on the Artificial Fish Swarms Algorithm and the DNA Coding. *Mathematics* **2023**, *11*, 767. [CrossRef]
41. Rani, N.; Mishra, V.; Sharma, S.R. Image encryption model based on novel magic square with differential encoding and chaotic map. *Nonlinear Dyn.* **2023**, *111*, 2869–2893. [CrossRef]
42. Rashmi, P.; Supriya, M.; Hua, Q. Enhanced Lorenz-Chaotic Encryption Method for Partial Medical Image Encryption and Data Hiding in Big Data Healthcare. *Secur. Commun. Netw.* **2022**, *2022*, 9363377. [CrossRef]
43. Ghadi, Y.Y.; Alsuhibany, S.A.; Ahmad, J.; Kumar, H.; Boulila, W.; Alsaedi, M.; Khan, K.; Bhatti, S.A. Multi-Chaos-Based Lightweight Image Encryption-Compression for Secure Occupancy Monitoring. *J. Healthcare Eng.* **2022**, *2022*, 7745132. [CrossRef]
44. Alghamdi, Y.; Munir, A.; Ahmad, J. A Lightweight Image Encryption Algorithm Based on Chaotic Map and Random Substitution. *Entropy* **2022**, *24*, 1344. [CrossRef]
45. Kumar, C.M.; Vidhya, R.; Brindha, M. An efficient chaos based image encryption algorithm using enhanced thorp shuffle and chaotic convolution function. *Appl. Intell.* **2022**, *52*, 2556–2585. [CrossRef]

46. Crocetti, L.; Di Matteo, S.; Nannipieri, P.; Fanucci, L.; Saponara, S. Design and Test of an Integrated Random Number Generator with All-Digital Entropy Source. *Entropy* **2022**, *24*, 139. [CrossRef]
47. Eder, C.; Pfister, G.; Popescu, A. Standard bases over Euclidean domains. *J. Symb. Comput.* **2021**, *102*, 21–36. [CrossRef]
48. Silva-García, V.M.; Flores-Carapia, R.; Rentería-Márquez, C.; Luna-Benoso, B.; Chimal-Eguía, J.C. Image cipher applications using the elliptical curve and chaos. *Int. J. Appl. Math. Comput. Sci.* **2020**, *30*, 377–391. [CrossRef]
49. Asharov, G.; Segev, G.; Shahaf, I. Tight Tradeoffs in Searchable Symmetric Encryption. *J. Cryptol.* **2021**, *34*, 9. [CrossRef]
50. Lin, C.H.; Hu, G.H.; Chan, C.Y.; Yan, J.J. Chaos-Based Synchronized Dynamic Keys and Their Application to Image Encryption with an Improved AES Algorithm. *Appl. Sci.* **2021**, *11*, 1329. [CrossRef]
51. Ravi, R.V.; Goyal, S.B.; Verma, C.; Raboaca, M.S.; Enescu, F.M.; Mihaltan, T.C. Image Encryption Using Block Chain and Chaos for Secure Communication. In Proceedings of the 2022 14th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), Ploiesti, Romania, 30 June–1 July 2022; pp. 1–6. [CrossRef]
52. Tahir, Ö. Is There Any Meaning of Planck's Constant Numbers as Regards to Quantum Superposition via the Chemical Atomic Masses of Nucleotide Bases? *Open Access Libr. J.* **2022**, *9*, e9482. [CrossRef]
53. Edwar, J.G.; Holman, M.A.; Martínez S, F.H. Enhanced Security: Implementation of Hybrid Image Steganography Technique using Low-Contrast LSB and AES-CBC Cryptography. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*, 899–905. [CrossRef]
54. Silva-García, V.M.; Flores-Carapia, R.; Cardona-López, M.A.; Villarreal-Cervantes, M.G. Generation of Boxes and Permutations Using a Bijective Function and the Lorenz Equations: An Application to Color Image Encryption. *Mathematics* **2023**, *11*, 599. [CrossRef]
55. Kusuma, C.; Sheetal, V.; Bhuvankumar, P. An authenticated University file system using EdDSA. In Proceedings of the 2022 Second International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering (ICATIECE), Bangalore, India, 16–17 December 2022; pp. 1–4. [CrossRef]
56. Boudot, F.; Gaudry, P.; Guillevic, A.; Heninger, N.; Thome, E.; Zimmermann, P. The State of the Art in Integer Factoring and Breaking Public-Key Cryptography. *IEEE Secur. Priv.* **2022**, *20*, 80–86. [CrossRef]
57. Burek, E.; Wroński, M. Quantum Annealing and Algebraic Attack on Speck Cipher. In Proceedings of the 2022 22nd International Conference on Computational Science (ICCS), London, UK, 21–23 June 2022; pp. 143–149. [CrossRef]
58. Ponuma, R.; Amutha, R. Compressive sensing based image compression-encryption using novel 1D-chaotic map. *Multimed. Tools Appl.* **2018**, *77*, 19209–19234. [CrossRef]
59. Hu, G.; Xiao, D.; Wang, Y.; Xiang, T. An image coding scheme using parallel compressive sensing for simultaneous compression-encryption applications. *J. Vis. Commun. Image Represent.* **2017**, *44*, 116–127. [CrossRef]
60. Asgari-Chenaghlu, M.; Balafar, M.A.; Feizi-Derakhshi, M.R. A novel image encryption algorithm based on polynomial combination of chaotic maps and dynamic function generation. *Signal Process.* **2019**, *157*, 1–13. [CrossRef]