Analysis of Chaotic-Chebyshev Polynomials using on Public Key Cryptosystems

K.Prasadh[1], R.Gnanajeyaraman[2]

[1]*Research Scholar, Dept. of CSE, Vinayaka Missions University, Salem. India, ksprasaad@gmail.com*
[2]*Research Scholar, Dept .of CSE, VMKV Engineering College, Salem .India, r.gnanajeyaraman@gmail.com*

Abstract

Due to rapid developments in limits and possibilities of communications and information transmissions, there is a growing demand of cryptographic techniques, which has spurred a great deal of intensive research activities in the study of cryptography. This paper describes a public key encryption based on chebyshev polynomials [1].We discuss the algorithm for textual data and present the cryptanalysis which can be performed on this algorithm for the recovery of encrypted data [2]. We also describe a simple hashing algorithm for making this algorithm more secure, and which can also be used for digital signature [3]. The main scope of this paper is to propose an extension of this algorithm to images and videos and making it secure using multilevel scrambling and hash. Software implementations and experimental results are also discussed in detail.

Keywords: *Choas, cryptography, public key, hash, chebyshev map*

1. Introduction

In the past few years lot of research on chaotic systems has been undertaken [4]. The chaotic systems are known to be very sensitive to initial conditions and they possess very random behavior. Due to these properties chaotic systems have become a very good candidate for their use in the field of cryptography [5]. The field of cryptography deals with providing security to one's data or files. Initially cryptography used the concept of secret key which was required to be transmitted in a very secure way.

Diffie and Hellmann showed for the first time that secret communication was possible without any transfer of a secret key between sender and receiver [6]. This new technique was named public key cryptography. Public key encryption techniques in contrast to secret key techniques, posses the following properties:

1. The encryption key is different from the decryption key.
2. The encryption key is public.
3. The calculation of decryption key from encryption key is almost impossible.

The sender uses the public key of receiver to encrypt the message; on the other end for decryption the receiver uses the corresponding secret key to decrypt the message.

In this paper we describe a public-key encryption algorithm based on chaotic maps [4]. The chaotic map used in this algorithm is Chebyshev map [1]. In section 2 we describe the existing algorithm, then its cryptanalysis and how to make the algorithm secure against the cryptanalysis. After establishing the security of this algorithm in section 2, in section 3 we propose an extension of this algorithm to images and videos, incorporating multilevel scrambling for better security and, further improve the security by adding hash to the algorithm. In the section 4 we summarize our observations and results. We close the paper with a conclusion in section 5.

2. Public Key Encryption Using Chebyshev Map

Chebyshev map is a chaotic map which is defined as follows:

$T_n(x) = 2 \cdot x \cdot T_{n-1}(x) - T_{n-2}(x)$, for any $n >= 2$, where $T_0(x) = 1$ and $T_1(x) = x$.

The algorithm described here uses a remarkable property called semi group property which is given by:
$T_r(T_s(x)) = T_{r\cdot s}(x)$. We now describe the existing algorithm [1]:

A, in order to generate the keys, does the following:

1. Generates a large integer s.
2. Selects a random number x in the interval [−1, 1] and computes $T_s(x)$.
3. A sets her public key to $(x,T_s(x))$and her private key to s.

B, in order to encrypt a message, does the following:

1. Obtains A's authentic public key $(x, T_s(x))$.
2. Represents the message as a number M in the interval [−1 1].
3. Generates a large integer r.
4. Computes $T_r(x),T_{r\cdot s}(x)= T_r(T_s(x))$ and $X =M\cdot T_{r\cdot s}(x)$.
5. Sends the cipher text $C = (T_r(x), X)$ to A.

A, to recover the plaintext M from the cipher text C, does the following:

1. Uses her private key s to compute $T_{s\cdot r}(x) = T_s(T_r(x))$.
2. Recovers M by computing $M = X/T_{s\cdot r}(x)$.

2.1. Cryptanalysis

Chebyshev polynomials can be alternatively defined as follows: Let n be an integer, and let x be a variable taking value over the interval [−1, 1]. The polynomial $T_n(x)$: [−1, 1] Æ [−1, 1] is defined as:

$T_n(x) = \cos(n\cdot \arccos(x))$.

Description of the Attack:

Let $(x, T_s(x))$ be A's public key. In order to encrypt a message M, B chooses a large integer r and computes: $T_r(x)$, $T_{r\cdot s}(x) = T_r(T_s(x))$, and $X = M \cdot T_{r\cdot s}(x)$ Then, he sends the cipher-text $C = (T_r(x), X)$ to A.

Given A's public key $(x, T_s(x))$ and the cipher text $(T_r(x),X)$, an adversary can recover M as follows:

1. Computes an r′ such that $T_{r'}(x) = T_r(x)$.
2. Evaluates $T_{r's}(x) = T_{r'}(T_s(x))$.
3. Recovers $M = XT_{r's}(x)$.

For the description of how to calculate r' one can refer to [2]. Thus Chebyshev map based encryption technique is not robust against attacks as such. Hence it needs security enhancement. The proposed security enhancement is described in the next section.

2.2. Security Enhancement

We use the hash algorithm [3] to calculate the hash value of session id and password concatenated together of any user. But instead of the XOR technique suggested we use our own encryption algorithm, as most of the XOR- ing based techniques are not robust against well known attacks such as chosen/known plain text attack. To avoid the possibilities of this attack we do the following: The hash function [3] returns a 128 bit hash value. This 128 bit hash value is divided into three parts first two of 52 bits and third of 24 bits. All of these values are then transformed into corresponding decimal representation. The value Tr(X) which is to be transmitted is then encrypted as follows:

$$Tr(X)' = ((p1 + p2) / p3)* \text{Original } Tr(X)$$

where p1,p2,p3 are decimal values calculated from the binary representation of 128 bit hash. On the receiver's end the hash is again calculated using session id and password and again the

value of the parameters p1, p2 and p3 is calculated. Using these values the correct value of Tr(x) can be calculated. Through this scheme, only the user with correct session id and password can decrypt the message. Hence the secure transmission of Tr(X) is ensured.

3. Extension of Secure Algorithm to Images and Videos

In this section we describe how we can use the above described algorithm for the encryption of images and videos. Images are composed of discrete units called pixels. A pixel is a small square representing some colour value, which when taken together form the mosaic. The image is a m x n matrix, where m represents the number of rows of pixels and n the number of columns of pixels, with each entry in the matrix being a numeric value that represents a given colour. For encryption purpose each pixel of the image can be considered as input message to the encryption algorithm. We also propose the use of two scrambling techniques to provide additional security.

3.1. Arnold Cat Scrambling

Arnold Cat Scrambling [7] is a simple and elegant demonstration and illustration of some of the principles of chaos–namely, underlying order to an apparently random evolution of a system. An image is hit with a transformation that apparently randomizes the original organization of its pixels as shown in Figure 1.

The transformation is defined as follows: If we let

$$X = \begin{bmatrix} x \\ y \end{bmatrix}$$

be a *n* x *n* matrix of some image, Arnold's cat map is the transformation

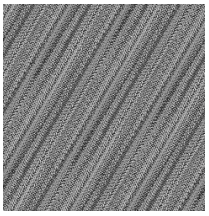
$$\Gamma\begin{bmatrix} x \\ y \end{bmatrix} \rightarrow \begin{bmatrix} x+y \\ x+2y \end{bmatrix} \bmod n$$

3.2. Phase Scrambling

This technique randomizes the phase of the r, g and b layers of an image individually [8] as shown in Figure 2. This changes the colour composition of the image significantly. It adds the same random phase structure to the existing three (rgb) phase structures in the original image. As a result, the relative phases of the r, g, and b layers in the scrambled image will be identical to their relative phases in the original image and the colour composition of the scrambled image will be as in the original image. (e.g., a gray scale image will generate a scrambled image which is also gray scale).

The contrast of all three layers in the image will, after scrambling, be identical to that of the rescaled (0-1) original image.

Original Image



Scrambled Image

Figure 1: Lena image after Arnold scrambling

Random phase can be generated by generating a random matrix whose size is the same as size as the image, taking its fourier transform, setting the magnitude to unity, and taking the inverse fourier transform. Figure 3 demonstrates the phase scrambling of an image.

3.3. Encryption using Chebyshev Polynomial

This phase takes as input the scrambled image and encrypts it using chebyshev polynomial [1] of the order as defined by the key generation process.

The input image is read pixel wise, and each pixel is given as input to the encryption function described earlier as input message. The output is the encrypted pixel value. After every pixel of the scrambled image has been encrypted the encrypted pixels are again converted back to image form hence giving the final encrypted image as shown in Figure 4

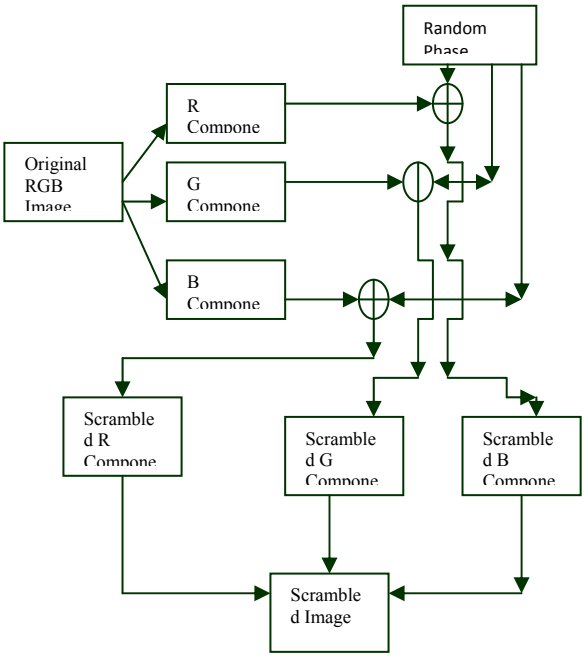
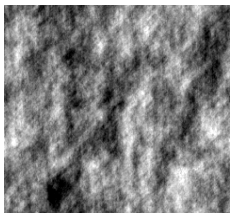


Figure 2: Various stages of Phase Scrambling technique

Original Image



Scrambled Image

Figure 3: Lena image after phase scrambling

Table 1: Time taken by both the scrambling techniques for a single frame of video

SCRAMBLING METHOD	FRAME SIZE	TIME TAKEN
ARNOLD SCRAMBLING	256X256	0.8420
PHASE SCRAMBLING	256X256	0.3120

From the observation it can be clearly seen that phase scrambling is much faster when

compared to Arnold cat scrambling and hence is better for videos.

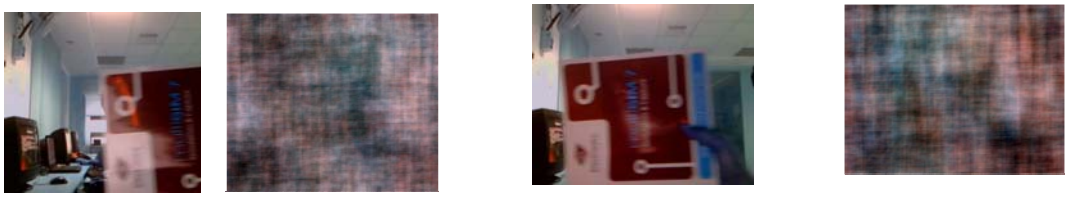


Figure 6: Results obtained on encrypting two frames of video after phase scrambling

4. Efficiency Check and Testing

A good encryption scheme should resist all kinds of attacks, such as brute-force attack, known plaintext attack, and statistical attack. We have already shown that our proposed algorithm is robust against chosen/plain text attack. Some statistical tests such as key sensitivity, correlation of adjacent pixels, mono bit test and run test are demonstrated in the following section.

4.1. Key sensitivity test

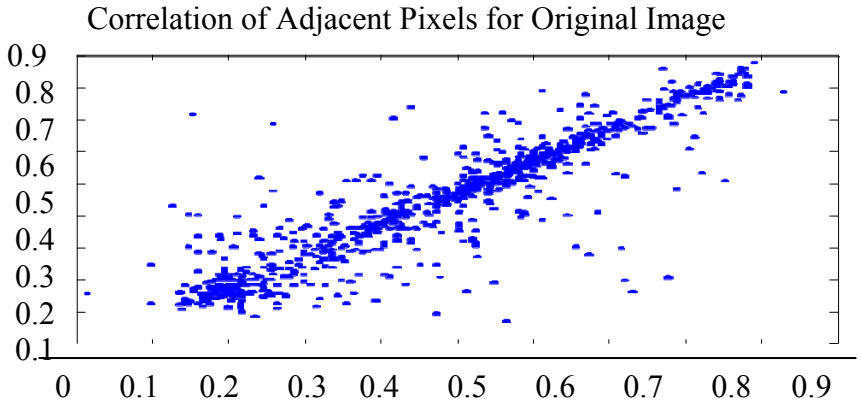
An ideal image encryption procedure should be sensitive with respect to the initial parameters. The change of a single bit in the key should produce a different encrypted image. We performed the test for r=81500, x=.5678 and the following results as shown in Figure 7 were obtained. We see that even for a slight change in S, we get an image which is 99.481% different from original one.

Original Image Encrypted image1 S=10600 Encrypted image 2 S=10601

Figure 7: Key sensitivity test

4.2. Correlation of adjacent pixels

We have analyzed the correlation between adjacent pixels in several images and their encrypted images. For an ordinary image, each pixel is usually highly correlated with its adjacent pixels. These high correlation properties can be quantified as the correlation coefficients for comparison. First we select 1000 pairs of two adjacent pixels from an image. Then we calculate the correlation coefficient. The following correlation plot was obtained when a grayscale Lena image was encrypted using the proposed cryptosystem. Correlation in original image = 0.916 Correlation in encrypted image=0.102



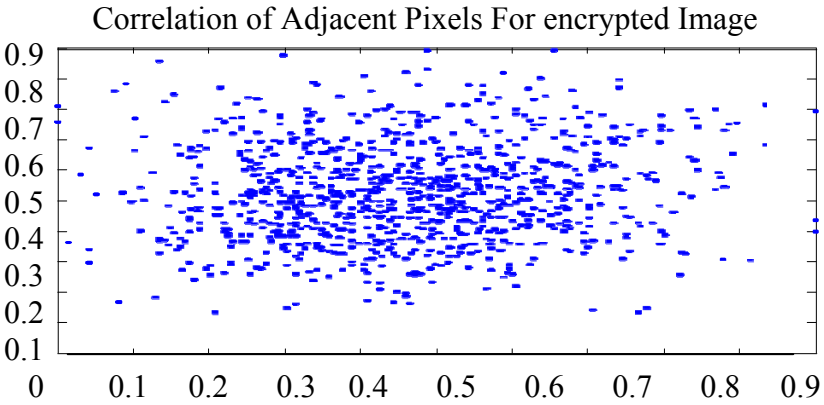


Figure 8: Correlation in original image and encrypted image

From the above findings, shown in Figure 8 it is evident that correlation in encrypted image is very less as compared to the original image, hence it is very difficult to figure out the approximate value of any pixel with the knowledge of its adjacent pixels.

4.3. Mono bit test [9]

This test counts the number of ones in the first 20,000 bit stream. The test is passed if the number of ones is greater than 9,654 and less than 10,346 Test Results of this test are shown in Table 2:

Table 2: Result of Mono bit test.

Image	Passing category	Result
Lena Color Image(256x256)	9654 < X < 10,346	10227

4.4. Long run test [9]

Testing Procedure:
1) A long run is defined to be run of length 34 and more (of either zeroes or ones)
2) On the sample of 20,000 bits the test is passed if there are no longer runs.

Table 3: Results of Long run test

Image	Passing Category	Results
Lena Colour Image(256x256)	< 34	16
Lena Grayscale Image(256x256)	<34	14

From Table 3, we can find that our chaotic sequence generates sufficiently long sequence of random bits, needed for robust encryption of images and videos.

4.5. Time Analysis

In Table 4, we have recorded the time taken by our algorithm to perform the encryption of images.

Table 4: Time analysis for various image sizes.

Size of image(Kb)	No. of pixels	Time taken(Sec)
2.01	64x64	.1400
4.76	128x128	.1710
12.8	256x256	.7170
612	512x512	4.8350

Plot of image size and time taken for encryption

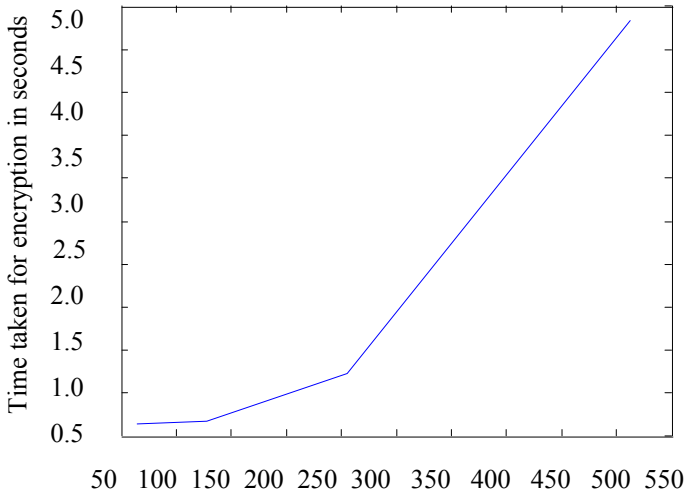


Figure 9: Plot of image size versus time taken for encryption

From Figure 9 we can see that the relationship between size and time to encrypt is almost linear and as the file size increases there is no abrupt change in the time taken for encryption and it increases proportionally.

In Table 5 we have tabulated time taken for encryption and phase scrambling for various video sizes (pixel count) for 15 frames video:

Table 5: Time analysis for videos

Pixel Count	Phase Scrambling Time(sec)	Encryption Time(sec)
64 x 64	0.2190	0.0620
128 x 128	0.8420	0.2180
256 x 256	3.9930	1.4360
512 x 512	20.8880	7.6440

In real time video streaming, encryption is efficient for 64x64 and 128x128 pixel size videos and is jerk-free. With further optimization and proper hardware implementation it can be made efficient for higher resolution videos.

5. Conclusions

In this paper, first we describe the existing algorithm to encrypt textual data using chebyshev polynomial and its cryptanalysis. Then we have also introduced a non XOR-ing technique to make the hashing algorithm more secure against the chosen plain text attacks. Further we have proposed the extension of encryption based on chebyshev polynomial from textual data to images and videos. The use of multilevel scrambling in the encryption of images makes the cryptosystem more secure and robust making it difficult for any intruder to crack the original video.

7. References

[1] Alfred J.Menezes, Paul C.Van Oorschoty, Scott A.Vanstone, "Handbook of Applied Cryptography",Pg.180-185.

[2] Diffe W. And Hellman M .E.,"New Directions in Cryptography". IEEE Transaction of Information Theory.22:644-454,1976.

[3] Di Xiao, Xiaofeng Liao, Shaojiang Deng, "A novel key agreement protocol based on chaotic maps" Information Sciences Volume 177, Issue 4, 15 February 2007, Pages 1136-1142

[4] K.Ganesan, R.Muthukumar, K.Murali. "Look-up Table Based Chaotic Encryption of Audio Files" IEEE Trans Circ Systems 2006; APCCAS 2006;Pages. 407-7

[5] L.Kocarev," Chaos Based Cryptography: A Brief Overview". IEEE Circuits and Systems Magazine,1(3):6- 21,2001.

[6] L.Kocarev, Z.Tasev, "Public key encryption based on Chebyshev maps", in: Proc. 2003 IEEE Symposium on Circuits and Systems, Bangkok, TH, vol. 3, pp. 28–31.

[7] N.K.Nishchal, J.Joseph and K.Singh, "Fully phase based encryption using fractional Fourier transform", Opt.Eng 42,1583-1588(2003).

[8] P.Bergamo, P.D'Arco, A.Santis and L.Kocarev, "Security of public key cryptosystems based on Chebyshev polynomials", *IEEE Transactions on Circuits and Systems—I* 52 (2005), pp.1382– 1393.

[9] Yuanzhi Wang, Guangyong Ren, Julang Jiang , Jian Zhang, Lijuan Sun, "Image Encryption Method Based on Chaotic Map", Second IEEE Conference on Industrial Electronics and Applications, Pg. 2557-2560, 2007.

Article received: 2009-08-05