PERFORMANCE ANALYSIS OF CHAOTIC ENCRYPTION USING A SHARED IMAGE AS A KEY

Alem Haddush Fitwi* and Sayed Nouh
Department of Electrical and Computer Engineering
Addis Ababa Institute of Technology, Addis Ababa University

ABSTRACT

Most of the secret key encryption algorithms in use today are designed based on either the feistel structure or the substitution-permutation structure. This paper focuses on data encryption technique using multi-scroll chaotic natures and a publicly shared image as a key.

A key is generated from the shared image using a full period pseudo random multiplicative LCG. Then, multi-scroll chaotic attractors are generated using a hysteresis switched, second order linear system. The bits of the image of the chaotic attractors are mixed with a plaintext to obtain a ciphertext. The plaintext can be recovered from the ciphertext during the deciphering process only by mixing the cipher with a chaos generated using the same secret key. As validated by a functional, NIST randomness, and Monte Carlo simulation tests, the cipher is very much diffused and not prone to statistical or selected cipher attacks.

In addition, the performance is measured and analyzed using such metrics as encryption time, encryption throughput, power consumption and compared with such existing encryption algorithms as AES and RSA. Then, the performance analysis and simulation results verify that the chaotic based data encryption algorithm is valid.

***Key Words**: Secret key encryption, shared image, hysteresis switched second order system, multiplicative LCG, chaotic attractors, randomness.*

INTRODUCTION

At present when the Internet provides essential communication for tens of millions of people and is being increasingly used as a tool for commerce, security becomes a tremendously important issue to deal with. There are many aspects to security and many applications, ranging from secure commerce and payments to private communications and protecting passwords. The fast expansion of computer connectivity necessitates protecting data and messages from unauthorized tampering or reading. Even the US courts have ruled that there exists no legal expectation of privacy for email. It is thus up to the user to ensure that communications which are expected to remain private actually do so. One of the techniques for ensuring privacy of files and communications is Cryptography [1].

In general, there are three types of cryptographic schemes: secret key (or symmetric) cryptography, public-key (or asymmetric) cryptography, and hash functions. In all cases, the initial unencrypted data is referred to as plaintext. It is encrypted into cipher-text, which will in turn be decrypted into usable plaintext [1-3].

The paper is organized as follows: Firstly, related works and progresses in the areas of cryptography and chaos generation and applications are examined. This is followed by the design, analysis and testing of the chaotic encryption algorithm. Performance measurements of the design and the corresponding results are then presented. Finally the conclusions that are drawn from the investigation are given.

RELATED WORKS

Pertinent works and progresses in the areas of cryptography and chaos are surveyed as follows: Data Encryption Standard (DES) is a feistel structure, block cipher that was selected by the National Bureau of Standards as an official Federal Information Processing Standard (FIPS) for the United States in 1976 and which had subsequently enjoyed widespread use internationally. DES is now considered to be insecure for many applications chiefly due to the 56-bit key size being too small. In January, 1999, Distributed.net and the Electronic Frontier Foundation collaborated to publicly break a DES key in 22 hours and 15 minutes. Consequently, DES has been withdrawn as a standard by the National Institute of Standards and Technology and was finally superseded by the Advanced Encryption Standard (AES) on 26 May 2002 [1, 4 - 9].

Advanced Encryption Standard (AES) is an encryption standard adopted by the US Government. It was announced by National Institute of Standards and Technology (NIST) as U.S. FIPS PUB 197 (FIPS 197) on November 26,

*E-mail: alemh29@gmail.com

2001 after a 5-year standardization process. The AES ciphers have been analyzed extensively and are now used worldwide, as was the case with its predecessor, DES. Until May 2009, the only successful published attacks against the full AES were side-channel attacks on some specific implementations. The input and output for the AES algorithm each consist of sequences of 128 bits. The Cipher Key for the AES algorithm is a sequence of 128, 192 or 256 bits. Other input, output and Cipher Key lengths are not permitted by this standard [1, 4, 10, 11].

RSA (which stands for Rivest, Shamir and Adleman who first publicly described it) is an algorithm for public-key cryptography. It is believed to be secure given sufficiently long keys and the use of up-to-date implementations. As of 2010, the largest (known) number factored by a general-purpose factoring algorithm was 768 bits long, using a state-of-the-art distributed implementation. RSA keys are typically 1024–2048 bits long. Some experts believe that 1024-bit keys may become breakable in the near term (though this is disputed); few see any way that 4096-bit keys could be broken in the foreseeable future. Therefore, it is generally presumed that RSA is secure if n, called modulus which is the product of two large random prime numbers, is sufficiently large. If n is 300 bits or shorter, it can be factored in a few hours on a personal computer, using software already freely available. As the key size increases, it becomes more expensive computationally [12, 13]

Elliptic curve cryptography (ECC) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. An ECC with a key-length greater than 112-bit said to be secure but slow when used for bulky data encryption. As the key size increases, encryption using ECC becomes computationally more expensive [12-14].

“Chaos" means "a state of disorder", but the adjective "chaotic" is defined more precisely in chaos theory. For a dynamical system to be classified as chaotic, it must be sensitive to initial conditions, and topologically mixing. Over the last two decades, chaotic oscillators have been found to be useful with great potential in many technological disciplines such as information and computer sciences, biomedical engineering, power systems protection, encryption and communications, etc. Recently, there has been some increasing interest in exploiting chaotic dynamics for real-world engineering applications, in which much attention has been focused on effectively generating chaos from simple systems by using simple controllers. Then a survey has been made on a number of techniques which have been developed for generating chaotic attractors and their application in papers [15-19].

The motivation to design and evaluate a chaotic based encryption algorithm is, therefore, because cryptographic algorithms play an astronomical role in information security systems, and in recent years, as the importance and the value of exchanged data over the Internet or other media types have been increasing alarmingly, there has been a search for the best solution to offer the necessary protection against the data thieves’ attacks. On the other side, cryptographic algorithms consume a significant amount of such computing resources as CPU time, memory, and battery power. As a consequence, there has been a great interest of designing cryptographic algorithms which are secure (or reliable), faster, efficient and with no known method of attacks.

DESIGN, ANALYSIS, AND TEST OF THE CHAOTIC ENCRYPTION ALGORITHM

Design overview

In the abstract, the design of a chaotic based crypto-system comprises five major tasks as delineated in Fig. 1. The tasks include image processing, key generation, generation of chaotic attractors, enciphering process, and deciphering process. In addition, the design is tested using a sample plaintext to verify if it can function as designed and required, and it is validated using statistical randomness and Monte Carlo simulation tests. Eventually, the type of techniques used to manage the secret key of the designed chaotic crypto-system, and to provide a digital finger print of the shared image to check its integrity are presented.

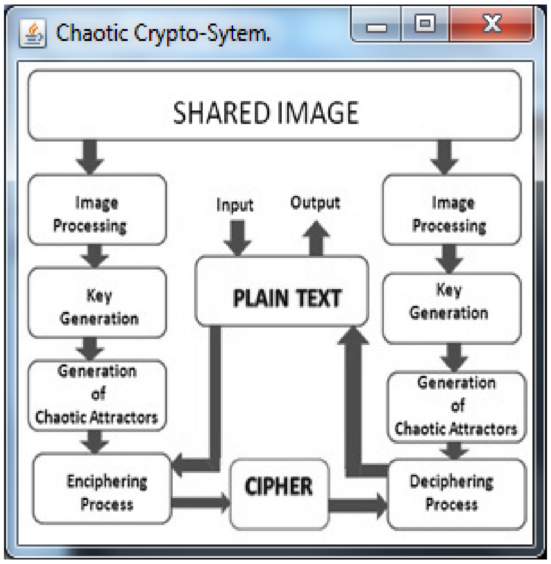


Figure 1 Chaotic crypto-system

Shared Image

In this crypto-system, the same image, in lieu of the secret key itself, is shared amongst all communicating (sending and receiving) parties from which the secret key is extracted. It is publicly shared by communicating parties just like a public key of a public-key encryption, only the information required to extract the key from the image is communicated secretly.

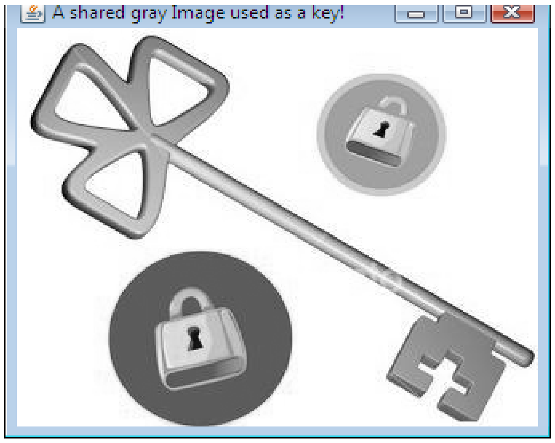


Figure 2 Grayscale image.

Keys having lengths less than the image size (width*length) are extracted from this shared image. The shared image used in this paper and from which a secret key is extracted is the one portrayed in the Fig. 2. But also it is possible to use any other image which is not completely black or white as a key! The minimum key length allowed is 128 bits for it is the minimum secure key length used in today's popular secret key encryption algorithms. Above it, it can be of any length as long as it is less than the size of the shared image.

The shared image is then processed to make it convenient to extract the secret key from its pixel values. The image processing here comprises such processes as image reading, converting to grayscale, and grabbing the pixel values of the grayscale image. If the image is RGB, it is first converted to a grayscale, as portrayed in Fig. 2, using the method convertTogray() from which pixel values, ranging from 0 to 255, are grabbed into a two dimensional array. Then, such important attributes as width (w), height (h), and pixel values (image Pixels) are accessed from the grayscale image in Fig. 2 as follows:

w = image.getWidth() (1)
h= image.getHeight() (2)

Image Pixel [w] [h]= readGrayImage Pixel (grayImage) (3)

Key Generation

Any secret key of length less than the size of the shared image can be extracted from the two dimensional pixel values of the shared image stored in the 2D array, ImagePixel[w][h], in Eq. 3.

Key. length<= w*h (4)

Where the values of w, and h are obtained in Eqs.1, and .2, respectively.

In this paper, the key is extracted from the 2D pixel values of the grayscale image using a full period pseudo random generator called linear congruential generator, LCG, constructed using defined values in GF (m) with a period of m-1. Then, the extracted key, keyExtract, is converted to binary values, and finally substituted using a seven-bit input and five-bit output S-Boxes to obtain the final enciphering and deciphering key, keyFinal.

The pseudo random generator used to extract a key from the grayscale image is given in Eq. 5, where 69,621 is the multiplier, and 2^{31}-1 is the modulus. It is called multiplicative LCG.

$$X_n = (69{,}621 X_{n-1}) \bmod (2^{31}-1) \quad (5)$$

The random numbers generated using the above algorithm [20] are used as ***indices*** of the 2D array of pixels, ImagePixel[][], to extract a key from the 2D pixel values of the grayscale image as follows, where Xo and X_1 are seed values.

```
For i=1:key.length do

    idx1=((69,621X_o) mod (2^31-1))mod w;
    idx2=((69,621X_1) mod (2^31-1))mod h

keyExtract[i]=ImgPixel[idx1][idx2];
end
```

Then, the keyBinary[] is divided into blocks of size 49 bit each, in turn, each block is divided into seven 7-bit pieces before being processed by the substitution boxes. Each of the seven S-boxes replaces its seven input bits with five output bits according to a non-linear transformation, provided in the form of a look up table. The S-boxes strengthen the security of the key; i.e substituted bits are used instead of the actual bits randomly extracted from the shared image thereby increasing the efforts of cryptanalysts who try to infer the key using brute force analysis or selected cipher attack.

The S-Boxes in this algorithm serve more or less the same purpose as the S-Boxes used in DES and AES; they are however different from those used in DES and AES. Here seven S-Boxes are used. Each of them is constructed using defined transformation of values in GF (2^5) comprising 4 unique rows and 32 columns. Each raw comprises 32 elements starting from 0 to 31 in a thoroughly random sequence. And the rows are numbered from 0 through 3.

The input bits are used as addresses in tables of the S-boxes. Each group of seven bits will give us an address in a different S-box. The first and last bits of the 7-bit input indicate row number, and the other 5 bits give the number of columns. Located at that address will be a 5-bit number. This 5-bit number will replace the original 7 bits. The net result is that the seven groups of 7 bits are transformed by the seven S-Boxes into seven groups of 5 bits for 35 bits total to obtain keyFinal[]= S-Boxes(keyExt).

Generation of Chaotic Attractors

In this paper, the required chaotic attractors are generated using a hysteresis switched second order linear system. The generation process comprises calculation of initial conditions from keys generated earlier, and solving the second order linear system using the concept of second order homogeneous differential equations.

Hysteresis Switched Second Order Linear System

There are many techniques of generating chaos; in this paper a system called "Hysteresis Switched second order linear system" is used. It is a chaotic oscillator triggered only by initial conditions. It has no inputs except the initial conditions, X_o and Y_o.

Then once triggered by the initial values, it keeps on oscillating and generating chaotic attractors for a time t, and moves from one scroll to another depending on the value of n (number of scrolls) provided due to the feedback hysteresis series as depicted in Fig 3.

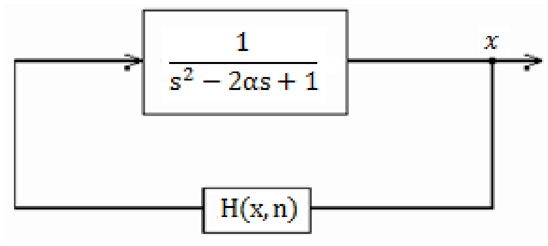


Figure 3 Chaotic oscillator [15].

The mathematical description of the hysteresis switched system in Fig 3 is given by:

$$\begin{cases} \dot{x} = y \\ \dot{y} = -x + 2\dot{\alpha} y + H(x,n) \end{cases} \tag{6}$$

where X_o, and Y_o are the initial conditions, α is a positive constant, x and y are state variables, H(x, n) is a hysteresis series described in Eq. 7 and 8, and n is the number of scrolls.

$$H(x,n) = \sum_{i=1}^{n} h_i(x) \tag{7}$$

and

$$h_i = \begin{cases} 1 & for\ x > i-1 \\ 0 & for\ x < i \end{cases} \tag{8}$$

Solution of Second Order Linear System

Differentiating both sides of the first Equation of system 6 produces the following:

$$\frac{d^2x}{dt^2} = \frac{dy}{dt} \tag{9}$$

Then the differentiated y in Eq. 9 is substituted by the equivalent expression given in the second equation of system (3.6). Hence, Eq. 3.9 becomes:

$$\frac{d^2x}{dt^2} = \frac{dy}{dt} = -x + 2\alpha\frac{dx}{dt} + H(x,n) \qquad (10)$$

Rearranging Eq. 10, gives out a homogeneous equation of the form

$$\frac{ad^2x}{dt^2} + b\frac{dx}{dt} + cx = f(x) \qquad (11)$$

Where f(x)=0. Therefore Eq. 11 is solved as follows, letting $x = e^{mt}$, and $D = \frac{dx}{dt}$

$$e^{mt}(aD^2 + bD + c) = 0 \qquad (12)$$

Then, if e^{mt} is to be a solution,

$$am^2 + bm + c = 0$$

$$m = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \qquad (13)$$

Using the values of m1 and m2, obtained from Eq. 13, the solutions are $x_1 = e^{m_1t}$ and $x_2 = e^{m_2t}$, then combining both solutions to obtain the general solution

$$x = c_1x_1 + c_2x_2$$
$$x = c_1e^{m_1t} + c_2e^{m_2t} \qquad (14)$$

If the system is to generate chaos, its solution must be complex; i.e for a=1, b=2α, and c=1 (obtained from Eq. 10, $m = \alpha + i\beta$. Then,

$$m = \alpha \pm \sqrt{\alpha^2 - 1} = \alpha + i\beta \qquad (15)$$

Hence, $\beta = \sqrt{1-\alpha^2}$, obtained by solving

Equation 13.

Euler's formulae,

$$e^{i\theta} = \cos\theta + i\sin\theta$$
$$e^{-i\theta} = \cos\theta - i\sin\theta \qquad (16)$$

If m1= $\alpha + i\beta$ and m2= $\alpha - i\beta$, then using the Euler's formulae in 16, the solution in 15 becomes:

$$x(t) = e^{\alpha t}[A\cos\beta t + B\sin\beta t] \qquad (17)$$

The solution for the state variable y is therefore obtained as follows,

$$y(t) = \frac{dx}{dt} = \frac{d\left(e^{\alpha t}[A\cos\beta t + B\sin\beta t]\right)}{dt} \qquad (18)$$

Solving for y and x in Equation 18, produces

$$x(t) = e^{\alpha t}\left[X_o\cos\beta t + i\left(\frac{Y_o - \alpha X_o}{\beta}\right)\sin\beta t\right]$$
$$y(t) = e^{\alpha t}\left[Y_o\cos\beta t + \frac{\alpha(Y_o - \alpha X_o)}{\beta}\sin\beta t\right] \qquad (19)$$

Calculation of Initial Conditions

In this paper, the initial conditions of system 6 are calculated using the bit values of the keyFinal array, which is the output of the S-Boxes, as follows.

$$X_o = var + keyFinalX$$
$$Y_o = var + keyFinalY \qquad (20)$$

Where var is a user defined value, kept secret as part of the key, keyFinalX and keyFinalY are two different keys generated using different seed values for the LCG.

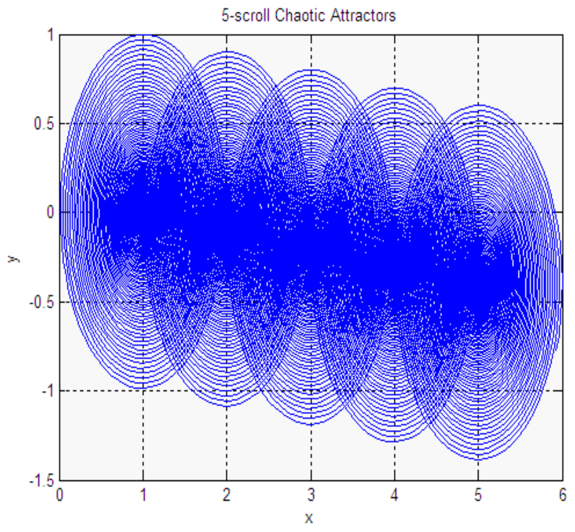


Figure 4 Five-scroll chaotic attractors

Figure 4 portrays the 5-scroll chaotic attractors generated using the solutions of system 6 given in Eq. 19. The chaos is generated using a key of length 136 bits, α=0.0049, and $\beta = \sqrt{1-\alpha^2}$ =0.999976. The chaos is sensitive to the α values, and secure range of α values are determined using an algorithm that makes use of the monobit test, described later

Enciphering Process

During the enciphering process, the plaintext is mixed with the generated chaos using a logical bitwise XOR operator as depicted in Fig 5.

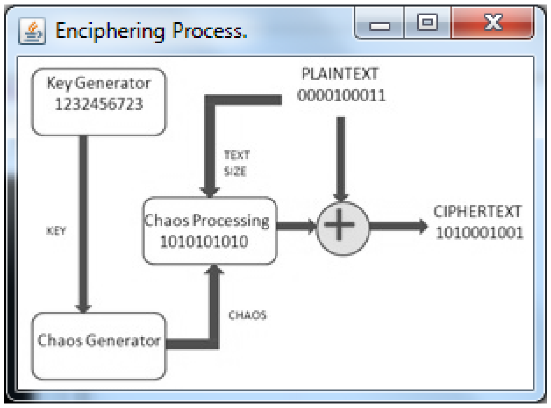
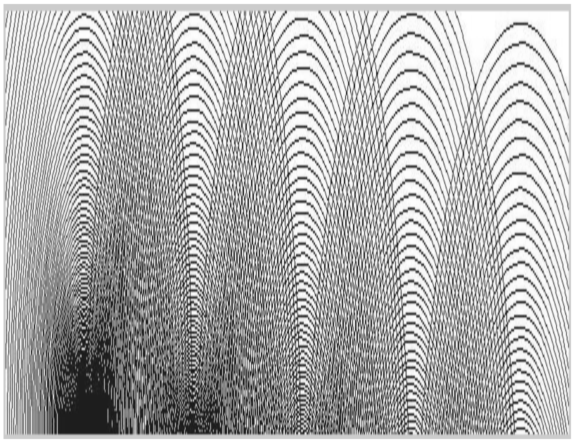
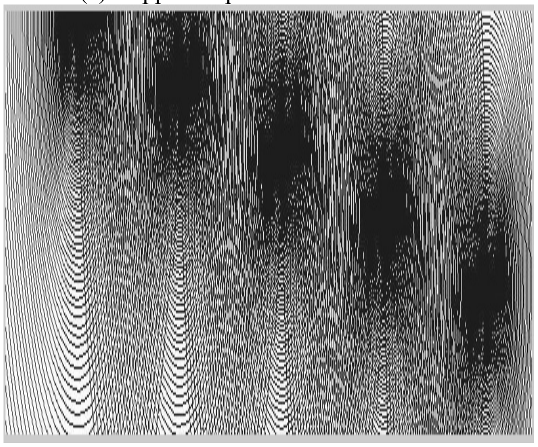


Figure 5 Enciphering process

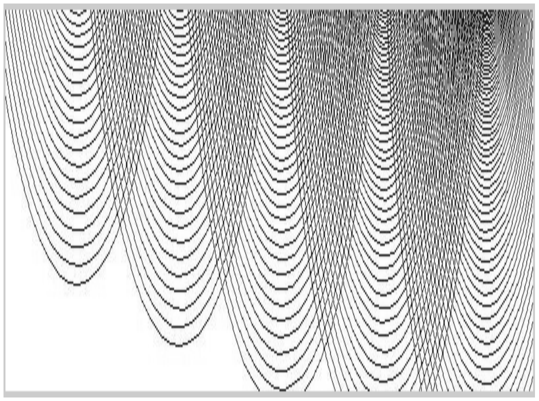
During chaos processing, three equal sections namely upper, middle and lower are cropped from the overall balanced chaos shown in Fig 4. and converted to grayscales as depicted in Figs 6 (a), (b), and (c), and XORed together to further increase the probability of the balance of 1's and 0's by avoiding localized imbalances. Then, the combination of the three crops is resized as per the size of the input plaintext as portrayed in Fig. 7. Through this process, all the security properties which include one way encryption, semantic security, and indistinguishability are achieved.



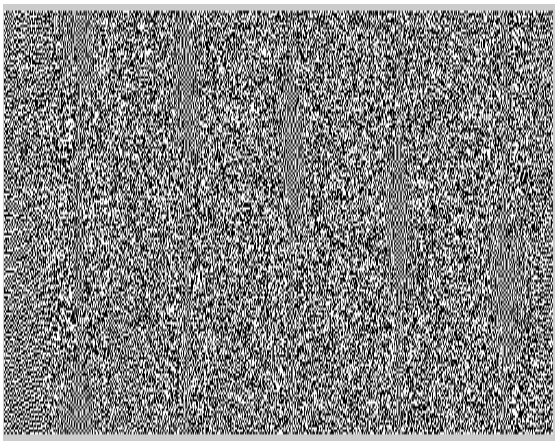
(a) Upper crop



(b) Middle crop



(c) Lower crop



(d) Mixed=a XOR b XOR c

Figure 6 Chaos cropping and mixing.

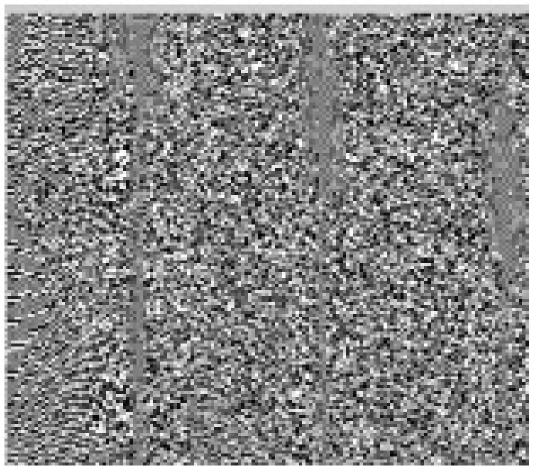


Figure 7 Resized balanced chaos.

Deciphering Process

The process of deciphering in this chaotic algorithm is essentially the same as the enciphering process. The rule is as follows: the same key used during the enciphering process is used in the deciphering process, but the cipher text, in lieu of the plaintext, is used as input to the chaotic algorithm. The ciphertext, which is the output of the enciphering process, is mixed with the appropriately sized chaos generated using the same key and the XOR operator.

Design Test

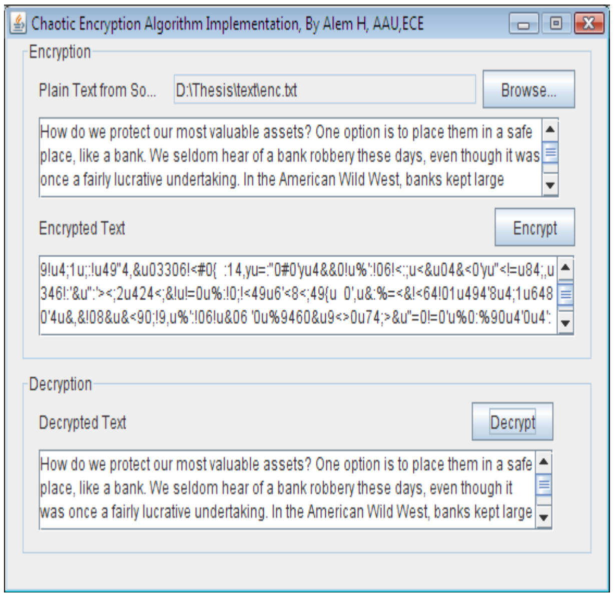
To verify if the designed chaotic algorithm can function as required, a sample plaintext was enciphered and then deciphered. A plaintext is browsed using the GUI depicted in Fig. 8, and is enciphered to convert it into a form which is unintelligible. Eventually, the ciphertext is deciphered to check if the chaotic deciphering process can fully recover the clear text (or plaintext) back from it. As copied from the text areas of the GUI in Fig. 8, portions of the plaintext, ciphertext, and decrypted text are respectively displayed below verifying that the chaotic algorithm works as designed and required. i.e the plaintext and recovered (or decrypted text) are the same.

Plaintext=How do we protect our most valuable assets?

Ciphertext=:"u1:u"0u%':!06!u:'u8:&!u#494790u4& &0!&ju;

Deciphered text=How do we protect our most valuable assets?

What is more, a number of NIST statistical tests and Monte Carlo Simulation test were performed on the chaotic sequence to attempt to compare and evaluate the sequence to a truly random sequence as depicted in Table 1. Then, the results validate the algorithm.



Chaotic Encryption Algorithm Implementation, By Alem H, AAU,ECE

Encryption

Plain Text from So... D:\Thesis\text\enc.txt

How do we protect our most valuable assets? One option is to place them in a safe place, like a bank. We seldom hear of a bank robbery these days, even though it was once a fairly lucrative undertaking. In the American Wild West, banks kept large

Encrypted Text

```
9!u4;1u;:!u49"4,&u03306!<#0{  :14,yu=:"0#0'yu4&&0!u%':!06!<:;u<&u04&<0'yu"<!=u84;,u
346!:'&u":'><;2u424<;&!u!=0u%:!0;!<49u6'<8<;49{u  0',u&:%=<&!<64!01u494'8u4;1u648
0'4u&,&!08&u&<90;!9,u%':!06!u&06 '0u%9460&u9<>0u74;>&u"=0!=0'u%0:%90u4'0u4':
```

Decryption

Decrypted Text

How do we protect our most valuable assets? One option is to place them in a safe place, like a bank. We seldom hear of a bank robbery these days, even though it was once a fairly lucrative undertaking. In the American Wild West, banks kept large

Figure 8 Chaotic crypto system

Table- 1: Summary of test results

NIST Test	Objective	P-value	Decision Rule
Monobit	Proportion of 1's and 0's	0.9491	0.01
Block Frequency	Proportion of 1's and 0's	0.9998	
Run	Oscillation b/n 1 and 0	0.9542	
Spectral	Reveal periodicities	0.0695	
Linear Complexity	Length of LFSR*	0.8030	

**LFSR=Linear Feedback Shift Register*

Key Exchange

In general, in a secret key encryption, as the number of communicating parties increase the exchange of the secret key becomes insecure. i.e n users who want to communicate in pairs need n * (n - 1)/2 keys [2]. The number of keys needed increases at a rate proportional to the square of the number of users! So a property of symmetric encryption systems is that they require a means of key distribution.

In this paper, all the information required for the extraction of the key from the publicly shared image and other constants is encrypted using a public-key RSA, and then sent to the recipient. Let INFO=Seeds + n + var + α + HKey + LenMul; then, it is sent as follows:

$$E\ (K_{PUB\text{-}R},\ E(K_{PRIV\text{-}S},\ INFO))$$

where E stands for RSA encryption, Kpub=public key, Kpriv=private key, R stands for Receiver, S stands for Sender, n is the number of scrolls, and LenMul is the number of digits contained in the multiplier of the PRNG used for key extraction. Besides, an HMAC that serves two purposes namely shared image integrity check, and origin authentication is generated using keyed SHA-1 as HMAC=H (HKey, Shared image).

PERFORMANCE ANALYSIS

In this paper, relevant metrics are identified, performance of the chaotic algorithm is measured, and eventually, the chaotic encryption algorithm is compared with existing one public key, RSA and one secret key, AES, encryption algorithms.

Metrics and Performance

In this paper, five pertinent metrics are used to evaluate the performance of the chaotic encryption. The metrics include encryption/decryption time, power consumption, encryption throughput, CPU time and cipher size. Then the experiment was conducted and performance results were collected using a laptop having processor of Intel (R) Pentium (R) Dual CPU T2370 @ 1.73GHz, 1.73GHz, and a RAM of 1GB.

Encryption Time

In most encryption algorithms, the encryption time is dependent on the computational complexity of the algorithm, key length, and the size of the plaintext to be encrypted. Here, in this paper, a number of encryption times for various plaintext sizes and key length are collected and analyzed as follows. The encryption time and text size are measured using a timer and bit length reader method built as part of the crypto system.

Effect of Key Length on the Encryption Time

Unlike other encryption algorithms, the key length does not affect significantly the computation time of the chaotic algorithm designed in this paper. This is due to the fact that the key is used only to calculate the initial conditions of the system used to generate chaos. The initial conditions are calculated in ways described in Eq. 20.

Effect of Plaintext Size on the Encryption Time

Various data sizes ranging from 7.84 Kb to 500 Kb are enciphered, and their respective encryption times are collected.

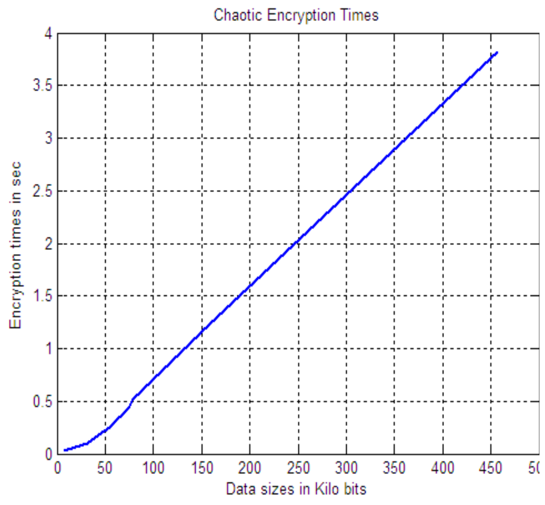


Figure 9: Data size versus encryption time

Figure 9 depicts that the enciphering time increases as the data size to be enciphered is increased. The data size and the enciphering time are linearly related.

Encryption Throughput, X_e
It is the measure of the number of bytes of ciphertext completed (enciphered) during an observation period (enciphering time). Mathematically represented:

$$x_e = \frac{number\ of\ \ completions\ in\ bytes}{Encryption\ Time(s)} \qquad (21)$$

The Effect of Changing the plaintext size on the Encryption Throughput

The graph in Fig. 10 shows that for initially small size data the throughput is not affected; rather it increases with increase in data size. However, once the data size gets large enough, further increase in the data size keeps on diminishing the encryption throughput.

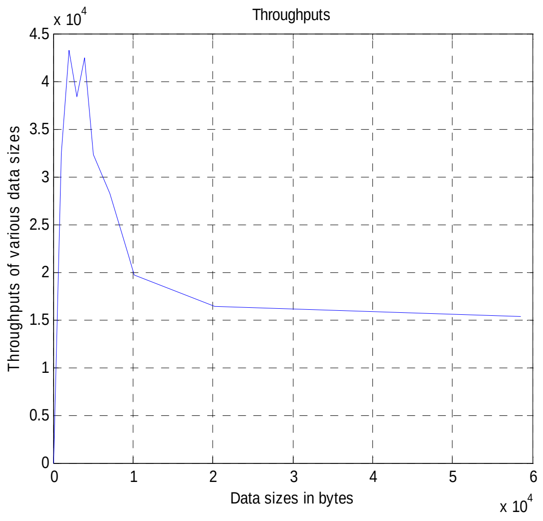


Figure 10 Encryption throughput verses data sizes

Power Consumption

Such technologies as CPU and memory are growing faster, and so is their need for power. However, battery technology is increasing at a much slower rate, forming a battery gap. Because of this, battery capacity plays a major role in the usability of devices and algorithms [21]. Hence, it is worthwhile to analyze the power consumption of the designed chaotic algorithm.

For computation of the energy cost of encryption, we use the same techniques as described in [21]. We present a basic cost of encryption represented by the product of the total number of clock cycles taken by the encryption and the average current drawn by each CPU clock cycle. The basic encryption cost is in unit of ampere-cycle. To calculate the total energy cost, we divide the ampere-cycles by the clock frequency in cycles/second of a processor; we obtain the energy cost of encryption in ampere-seconds. Then, we multiply the ampere-seconds with the processor's operating voltage, and we obtain the energy cost in Joule. That's, by using the cycles, the operating voltage of the CPU, and the average current drawn for each cycle, we can calculate the energy consumption of cryptographic functions. Then, the amount of energy consumed by the chaotic algorithm C to achieve its goal (encryption or decryption) is given by:

$$E = V\ cc*I*T \text{ joules} \qquad (22)$$

Where for a given hardware Vcc is fixed. The encryption time, T, is considered the time that an encryption algorithm takes to produce a cipher text from a plaintext, and I is the average current consumed per CPU cycle.

In this paper, the experiment was conducted and performance results were collected using a laptop with Pentium Dual 1.73GHz CPU. Therefore, the approximate average current consumed is 100mA when it is busy, and the CPU voltage is Vcc=1.25 volts (both collected from Intel Manual). The power consumption performance analysis results for various data sizes are then collected based on these current and voltage ratings.

Effect of Plaintext Size on Power Consumption

Figure 11 clearly shows the variation of energy consumption with different data sizes. It can also be inferred from the graph that it is similar to the data size versus encryption time graph in Fig. 9 which implies that the energy consumed during an enciphering process is directly proportional to the encryption time.

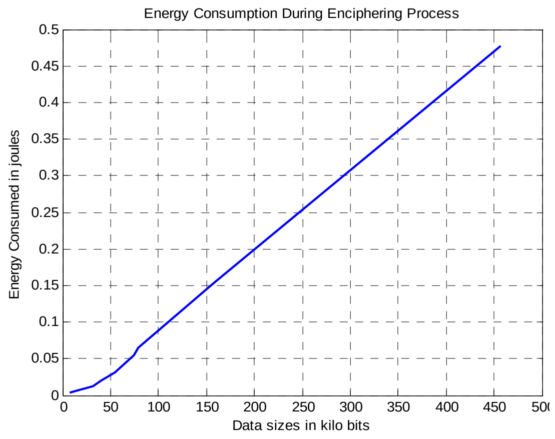


Figure 11 Data size versus energy consumption

CPU Time

The CPU process time is the time that a CPU is committed only to the particular process of calculations. It reflects the load of the CPU. The more CPU time is used in the encryption process, the higher is the load of the CPU.

In this paper, the CPU time is calculated using the technique described in [29] as follows:

$$CPU\ busy\ time, Tcpu = \frac{\text{CPU utilization}}{Observation\ Period} \qquad (23)$$

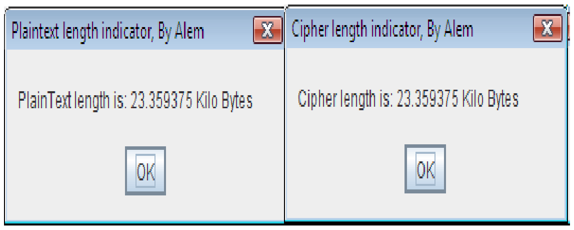
Where the observation period is equal to the enciphering time and CPU utilization for the encryption process is obtained from the task manager.

It is found out that the longer the encryption time is the busier the CPU becomes. That's, if the time required to encipher a certain text is longer, the CPU load (or busy time) is proportionally higher.

Cipher Size

One of the most integral Shannon's Characteristics of "Good" Ciphers is that the size of the enciphered text should be no larger than the plaintext of the original message [2].

As it is the case with other secret key encryption algorithms, in this work the size of the plaintext and the ciphertext are found to be the same fulfilling the Shannon's size Characteristics of "Good" Ciphers. The two merged pop-up message boxes portrayed in Fig. 12 verify that the plaintext and ciphertext, in this algorithm, are of the same size.


Plaintext length indicator, By Alem

PlainText length is: 23.359375 Kilo Bytes

Cipher length indicator, By Alem

Cipher length is: 23.359375 Kilo Bytes

Figure 12 Length measure of a clear and cipher texts

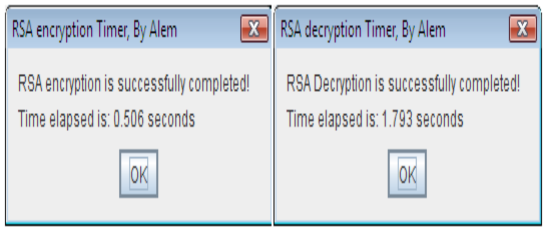
Comparison with AES and RSA

The performance of the designed algorithm is compared with the popular current-in-use secret key encryption algorithm, AES, and with a public key encryption, RSA. The data sets, used in the chaotic encryption, are encrypted using both AES and RSA, and their performance is evaluated for the same metrics used above. Here, while analyzing the performance of AES and RSA, only secure key lengths are used, 128 bits for AES and 1024 bits for RSA.

Encryption Times

Encryption times for same set of various data sizes, used to analyze the performance of the chaotic enciphering, are collected the ways depicted in Fig 13. Eventually, they are put to graphs as delineated in Fig. 14. It is found that the decryption times of RSA algorithm are higher than its encryption times. This is due the fact that the enciphering comprises modular computation of ***(plaintext)^{pubkey} mode n***, whereas the deciphering process involves the modular computation given as ***(plaintext)^{pubkeyprivkey} mode n.*** PrivateKey *PubKey is much larger than PubKey, and hence requires higher computational time.

Likewise, encryption times for AES, for the various data sizes enciphered, are collected. The deciphering times for AES are more or less close to the enciphering times.



RSA encryption Timer, By Alem

RSA encryption is successfully completed!

Time elapsed is: 0.506 seconds

OK

RSA decryption Timer, By Alem

RSA Decryption is successfully completed!

Time elapsed is: 1.793 seconds

OK

Figure 13: RSA crypto timers

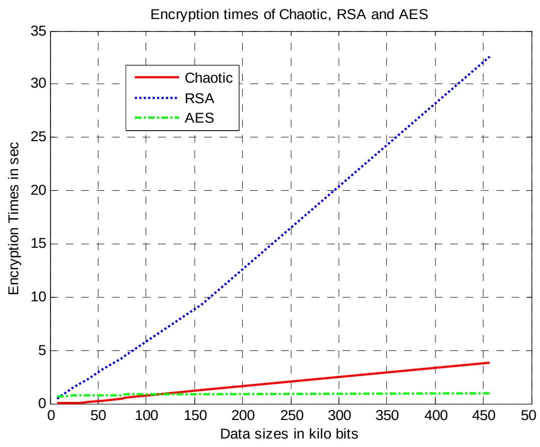


Figure 14: Encryption times of chaotic, RSA and AES

Figure 14 shows that the chaotic encryption is much faster than RSA algorithm for any data size. Besides, it has better time performance than AES, too, but for smaller data sizes (<125 Kb).

Encryption Throughput

Figure 15 illustrates that the throughput performance of the chaotic encryption is very much high for smaller size of data, and it keeps on decreasing as the data size increases. It has higher performance than AES for smaller data sizes, but it is very much superior to RSA for any data size.

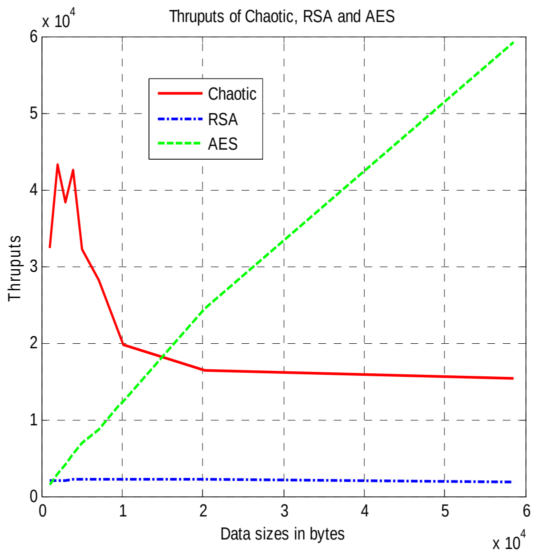


Figure 15 Chaotic, RSA, and AES throughputs

Encryption Power Consumption

The power consumptions of the Chaotic, RSA and AES are depicted in Fig. 16.

The figure demonstrates that the designed algorithm has less power consumption than AES for relatively small data sizes, and much better performance than the RSA for any data size. The graphs in this figure are similar to the ones on Fig. 14 proving that the energy consumed by an enciphering process is proportional to the time consumed by that process.

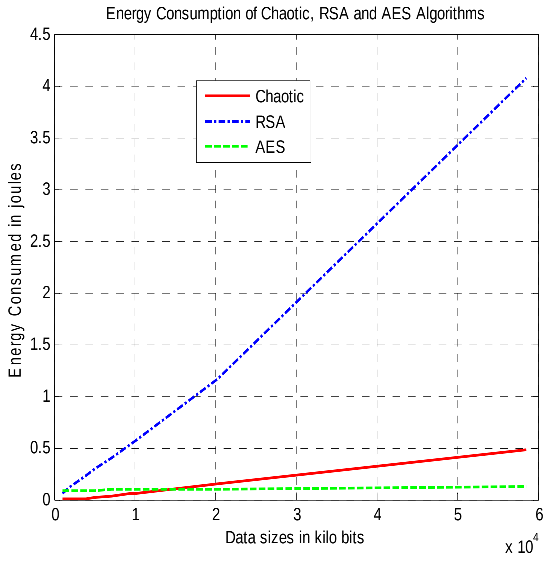


Figure 16 Power consumption of chaotic, RSA, and AES algorithms

Memory Cost

In secret key encryption, the plaintext and cipher text are of the same size. Similarly, the plaintext and ciphertext of the designed algorithm are of the same size. But, considering the RSA, the cipher size is greater than the size of the input plaintext. As a result, the cipher of an RSA algorithm occupies more memory as compared to that of chaotic and AES.

Security

The security of the chaotic encryption scheme lies on the difficulty of obtaining the exact key combination from amongst the very large set of possible combinations of the key due to limitations in the computational power of today's computers.

Table 2: Summary of security comparison [5]

Algorithm	Number of operations required	Examples
Chaotic	$2^{128}(2^{k+1}-1)$	Key=136 bits→2^{145} operations
AES	2^{n}	Key=128 bits→2^{128} operations
RSA	$\frac{e^{\sqrt{2*lnp*\ln(lnp)}}}{lnp}$	N=1024 bits →2^{90} operations

Table 2 summarizes the security comparison of the Multi-Scroll chaotic, AES and RSA encryption algorithms. The chaotic algorithm is superior to both. It meets such security properties as one way encryption, semantic security, and indistinguishability. Besides, it is prone to the three models of attacks namely total break, CCA1, and CCA2.

CONCLUSION

At present, Internet and network applications are growing very fast, so the need to protect such applications has increased. Encryption algorithms play an immense role in information security systems. This paper has presented the optimization of multi-scroll chaotic attractors for text encryption and its performance analysis. All essential and collateral parameters are systematically determined to satisfy the specific cryptographic security requirements. The algorithm solves at least some of the drawbacks suffered by existing cryptographic algorithms such as AES and RSA.
In this paper, a multi-scroll chaotic enciphering algorithm is fully described and validated via functional and randomness tests. Then, appropriate metrics for performance measurements are identified; the performance of the algorithm was measured and compared with such existing cryptographic algorithms as RSA and AES. The test results show that the designed algorithm works as required, i.e, the data enciphered by the enciphering process is fully recovered by the deciphering process. The test and security analysis prove that the cipher is not prone to cipher or statistical attack (including total break, CCAI, and CCA2), and the key is secure. It meets the three properties of security namely: one way encryption, semantic security, and indistinguishability.

The performance of the chaotic encryption algorism is by far better than the RSA. It has less encryption time, less power consumption, and higher throughput than RSA. The size of the plaintext and ciphertext is also the same in the chaotic algorism which is not the case in RSA. RSA cipher is longer than the plaintext causing more resource consumptions and congestion in memory and bandwidth. Comparing it with AES, it has better performance only for relatively smaller data sizes. The chaotic encryption has another advantage over RSA and AES in that it is key length independent. The key length can be made longer without significantly affecting the computational time. In addition, there are no known methods of attack so far for chaotic encryption!

REFERENCES

[1] Smart, N., "*Cryptography: An Introduction*", CRC press, pp.11-387, June 2010.

[2] Pfleeger, C. P., "*Security in Computing*", Pfleeger Consulting Goup, Shari Lawrence Pfleeger - RAND Corporation, October, 2006.

[3] Stallings, W., "*Data and computer Communications*", Prentice Hall, New Jersey, 1996.

[4] Menezes P., Oorschot, V., and Vanstone, S., "*Hand book of Applied Cryptography*", CRC press, 1996.

[5] Stallings,W. "*Lecture Notes for use with Cryptography and Network Security*", May 2010.(Online).

[6] FIPS PUB 46-3 Federal Information Processing Standards Publication October 25, 1999.

[7] FIPS PUB 46-2 Federal Information Processing Standard Publication December 30, 1993.

[8] FIPS PUB 46 Federal Information Processing Standard Publication January 15, 1977.

[9] FIPS PUB 46-1 Federal Information Processing Standard Publication January 22, 1988.

[10] Federal Information Processing Standards Publication 197, Announcing the Advanced Encryption Standard (AES), November 26,2001.

[11] Rhee, M.Y., "*Internet Security, Cryptographic Principles, Algorithms and Protocols*", John Wiley & Sons 2003.

[12] Gura, N., Patel, A., Wander, A.,Eberle, H. and Shantz, S.C. "*Comparing Elliptic Curve Cryptography and RSA on 8-Bit CPUs*", International Association for Cryptologic Research, 2004.

[13] Wenbo Mao Hewlett-Packard Company, "*Modern Cryptography:Theory and Practice*" John Wiley & Sons July 2003.

[14] Anoop,M.S., "*Elliptic Curve Cryptography an Implementation Guide*", 2006.(Online)

[15] Han, F., Lu, J., Yu, X., Chen,G. and Feng, Y.,*Generating Multi-Scroll Chaotic Attractors Via a Linear Second Order Hysteresis System*", Watam Press, 2005.

[16] L' ua , J., Hanb, F., Yub, X. and Chenc, G., "*Generating 3-D multi-scroll chaotic attractors: A hysteresis series switching method*" , Elsevier 18 May 2004

[17] Han, F., Hua, J., Yub, X. and Wang, Y., "*Fingerprint Images encryption via multi-scroll chaotic attractors*", Elsevier, 2007.

[18] Lü, J., Murali, K., Sinha, S., Leung, H. and Aziz-Alaoui, M. A., "*Generating multi-scroll chaotic Attractors by thresholding*", Elsevier , 30 January, 2008.

[19] Fengling Han, Xinghuo Yu, and Jiankun Hu, "*A New Way of Generating Grid-Scroll Chaos and its Application to Biometric Authentication*", Melbourne VIC Elsevier 2001.

[20] Jain, R., "*Art of Computer Systems Performance Analysis Techniques For Experimental Design Measurements Simulation and Modeling*", 2004. (Online)

[21] Naik, K. and Wei, D. S. L. "*Software implementation strategies for power-conscious systems,*" Mobile Networks and Applications, Vol. 6, pp. 291-305, 2001.