Faculty of Management Technology

Business Informatics Department

The German University in Cairo

Securing Game Code using AES and Logistic Map

Bachelor Thesis

Author: Seifeldeen Eyad Harfoush

Supervisor: Dr. Wassim Alexan

Submission date: June 4$^{\text{th}}$, 2023

This is to certify that:

(i) the thesis comprises only my original work towards the Bachelor degree,

(ii) due acknowledgement has been made in the text to all other material used.

Seifeldeen Harfoush

June 4th, 2023

Abstract

The thesis paper presents a comprehensive investigation into securing game code by utilizing the advanced encryption standard (AES) and logistic map. The primary aim of this study was to contribute to the field through extensive literature research and the proposal of a two-stage encryption algorithm that combines the logistic map with AES-128. Moreover, the paper conducts an examination of contemporary technologies used for the implementation of encryption algorithms. The proposed encryption scheme was implemented using the Wolfram Language and Wolfram Mathematica. Performance metrics, including mean square error (MSE), throughput, execution time, and memory usage, were utilized to analyze and compare the proposed scheme with existing algorithms including AES-128 and logistic map. The performance evaluation yielded promising security levels, as the proposed algorithm exhibited higher MSE scores than AES-128 while maintaining comparable execution times and throughput. However, it was observed that the algorithm's efficiency in memory usage diminishes as the plaintext length increases, which presents potential challenges regarding scalability and resource availability.

Acknowledgement

I wish to express my sincere thanks to my supervisor Dr. Wassim Alexan for providing me with all the necessary facilities for the project and for their support, patience and help through this project. Finally, I would like to thank my family for there continuous support and for their great belief in me.

Seifeldeen Harfoush

June 4th, 2023

Contents

List of Figures

List of Tables

List of Acronyms

AES	Advanced Encryption Standard
MSE	Mean Square Error
RAM	Random Access Memory

Chapter 1

Introduction

With the evolution of technology in the past decades; the availability of high-speed Internet; and the rise of using peer-to-peer networking, video game companies shifted away from depending on distributing physical copies of games and are now depending on digital distribution services to sell digital copies of their games.

However, this shift led to the rise of new piracy methods that allow for illegal copying, counterfeiting, and distribution of commercial video games; threatening the video game industry as a whole as more and more users get involved in video game piracy. Thus, implementing more security measures and the usage of encryption is essential to maintain the security of such games.

1.1 Motivation

The main motive behind this paper is to assess and test the usage of Advanced Encryption Standard (AES) and Logistic map in encrypting video game code as a hybrid algorithm to find the encryption standard that best secures the data. Moreover, the proposal of the usage of a Logistic map in key generation for AES is another motive behind this paper.

1.2 Contributions

This thesis provides an in-depth literature review of different articles that discuss the usage of AES and Logistic map in the encryption of data to be able to study how previous studies implemented such algorithms. Moreover, it proposes a text encryption scheme utilizing AES-128 and Logistic map, and tests its performance with Java game code as plaintext.

1.3 Organization

Chapter 2 discusses previous works and literature. The first section of the literature review discusses Chaos theory and Logistic map that is implemented in encryption. The second section discusses Advanced Encryption Standard and the steps that are taken each round. Performance evaluation metrics for which the algorithm will be assessed are then discussed, and then the state-of-the-art in which previous works using AES and Logistic map for text encryption are discussed. Next, a technology comparison between different encryption environments will be discussed to evaluate the best technology to implement this algorithm. The next chapter will discuss the proposed encryption algorithm and explain how it operates. Next, the proposed algorithm is tested and evaluated based on different performance metrics. Finally, conclusions and discussion will be presented.

Chapter 2

Background and Literature Review

This section discusses the background information about the proposed algorithm and the recent works that have been made that relate to the proposal we have in this paper.

2.1 Chaos Theory

Previously believed to have purely random states of disorder and irregularity, dynamical systems that are very sensitive to initial conditions are now known to contain underlying patterns and deterministic rules. This field of research is known as chaos theory. According to chaos theory, chaotic complex systems appear random, but they include underlying patterns, connections, ongoing feedback loops, repetition, self-similarity, fractals, and self-organization [1].

Cryptography has long employed the chaos theory. Numerous cryptographic primitives have been created in the last several years using chaos and nonlinear dynamics. These methods include steganography, watermarking, secure pseudo-random number generators, hash functions, and image encryption. The control parameters and initial state of the chaotic maps serve as the keys for the majority of these algorithms, which are based on uni-modal chaotic maps [2].

2.1.1 Chaotic Maps

Numerous chaotic maps and associated bifurcations were created from chaos theory. Some of these maps are shown in Table 2.1.

Different chaotic maps, including Gaussian, Tent, Bernoulli, and Logistic maps, are presented in Table 2.1. These four maps are only a selection of the different chaotic maps available.

As previously noted, chaotic maps are used because of their randomness and unpredictable nature. The user would have access to numerous useful capabilities, such as a random number generator if they knew how to manipulate the chaotic map and its associated chaotic regions [3].

Throughout the next subsection, we will focus on the generalized Logistic chaotic map which will be discussed further.

Table 2.1: Different discrete chaotic maps equations and properties

Name	Equation	Space Domain	Number of Space Dimensions
Gaussian Map	$x_{n+1} = e^{-ax_n^2} + \beta$	Real	1
Tent Map	$x_{n+1} = f_\mu(x_\mu) = \begin{cases} \mu(x_n) \text{ for } x_n < \frac{1}{2} \\ \mu(1 - x_n) \text{ for } x_n \geq \frac{1}{2} \end{cases}$	Real	1
Bernoulli Map	$f(x) = \begin{cases} \text{2x for } 0 \leq x_n < \frac{1}{2} \\ \text{2x-1 for } 1 > x \geq \frac{1}{2} \end{cases}$	Real	1
Logistic Map	$x_{n+1} = rx_n(1 - x_n)$	Real	1

2.1.2 Logistic Chaotic Map

One of many chaotic maps is the Logistic map. The Logistic map equation in Table 2.1 uses (r) as the bifurcation coefficient and (x_n) as the ratio of the current population to the maximum possible population. Certain restrictions are placed on the equation's parameters based on table 2.1's equation. The bifurcation parameter has a range of [0,4] whereas (x_n) has a range of (0,1) [4].

The complete bifurcation diagram for the Logistic map in Table 2.1 obtained numerically is shown in Figure 2.1 [4]. The fixed point or 1-cycle at $x = 0$ is stable for $-1 < a < 1$, where the fixed point at $x = 1 - 1/a$ is unstable. The stable point at $x = 0$ loses stability at $a = -1$, where the period 2 attractor or 2-cycle is created as:

$$x_{\pm} = \frac{1}{2a}(a + 1 \pm \sqrt{a - 3)(a + 1)}) \tag{2.1}$$

The expression for the 2-cycle (5) is the same as that for the case of positive a. However, that $x_{\pm}$ becomes real again for $a < -1$. The stability condition for Equation 2.1 is fulfilled for

$$3 < a < 1 + \sqrt{6} \tag{2.2}$$

and

$$1 - \sqrt{6} < a < -1 \tag{2.3}$$

The period-doubling bifurcation is also seen for the negative side causing chaos. The fully developed chaos is seen for $a = -2$, where x_n is bounded in the range $[-0.5, 1.5]$. For $a < -2$, x_n goes to ∞ as n increases, hence the lower bound for a can be set to be at $a = -2$ according to the same criterion that determines the upper bound for a to be at $a = 4$ [4].

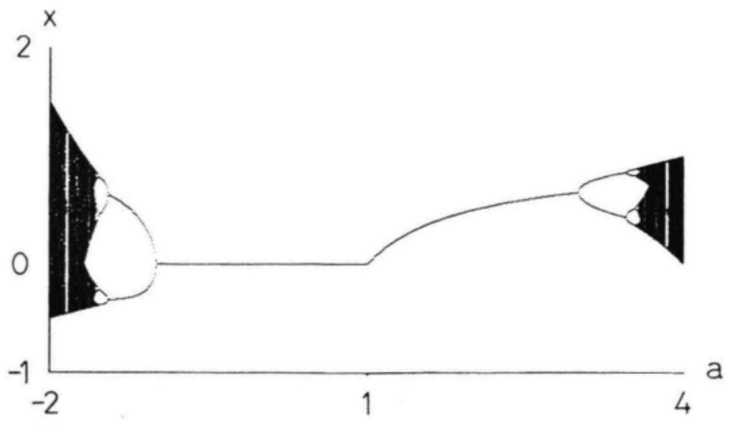


Figure 2.1: The complete bifurcation diagram for the Logistic map [4]

2.2 Advanced Encryption Standard

The Advanced Encryption Standard (AES) is a widely used encryption algorithm for securing electronic data. It is considered one of the most secure and robust encryption algorithms currently available, and was selected by the National Institute of Standards and Technology (NIST) in 2001 as the replacement for the outdated Data Encryption Standard (DES).

AES is a symmetric-key encryption algorithm, which means that the same key is used for both encryption and decryption. The algorithm uses a block cipher, which means that data is encrypted in fixed-size blocks. The most common block sizes for AES are 128-bits, 192-bits, and 256 bits.

The AES algorithm operates in rounds as shown in Figure 2.2 [5], each of which consists of several sub-steps. The overall structure of the AES algorithm is similar to other block ciphers, but its internal workings are quite different, as it is based

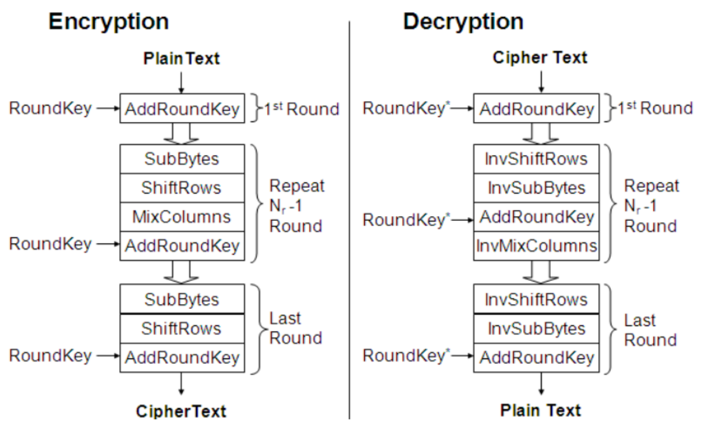


Figure 2.2: AES encryption/decryption flowchart [5]

on a substitution-permutation network (SPN) structure and uses four main operations: SubBytes, ShiftRows, MixColumns, and AddRoundKey, which are repeated multiple times in each round. The algorithm also uses a key expansion process to generate a set of round keys from a given secret key.

SubBytes is the first operation in the AES algorithm. It involves substituting each byte in the input data with another byte from a predefined S-box. The S-box used in the AES algorithm is a fixed table of 256 bytes, which is generated using the following polynomial equation:

$$P(y) = x^8 + x^4 + x^3 + x + 1 \tag{2.4}$$

The substitution operation is designed to provide confusion and non-linearity, which makes it harder for attackers to decipher the encrypted data [6].

ShiftRows is the second operation in the AES algorithm. It involves shifting the rows of the input data matrix by a certain number of bytes, which is determined by the row number. The purpose of this operation is to increase the diffusion of the input data, which helps to prevent patterns from emerging in the encrypted output [7].

MixColumns is the third operation in the AES algorithm. It involves multiplying each column of the input data matrix with a fixed matrix, which is designed to provide diffusion and non-linearity. This operation is used to further increase the complexity of the encrypted data, making it harder for attackers to decrypt [7].

AddRoundKey is the final operation in the AES algorithm. It involves XORing the input data with a set of round keys, which are generated using the key expansion process. The round keys are derived from the secret key, and each round key is used only once during the encryption and decryption process [7].

The key expansion process is used to generate a set of round keys from a given secret key. The process involves iterating a mathematical formula to generate a series of words, which are then used to derive the round keys. The number of rounds and the size of the key determine the number of words generated during the key expansion process [7].

AES has numerous benefits that make it a popular choice for encrypting electronic data. One of the key benefits of AES is its high level of security. The algorithm has been extensively analyzed and tested by cryptographers, and it has been found to be resistant to a wide range of attacks, including brute force attacks, differential cryptanalysis, linear cryptanalysis, and algebraic attacks [6].

The security of AES has been further strengthened by the use of key sizes of up to 256 bits, which makes it virtually impossible for attackers to brute force the key.

Another important benefit of AES is its efficiency. The algorithm is designed to be fast and computationally efficient, which means that it can be used to encrypt and decrypt large amounts of data quickly and without significant overhead. This makes it well-suited for use in a wide range of applications, including secure communication protocols, file encryption, and database encryption [7].

Additionally, AES has a well-defined and widely accepted standard, which makes it easy to implement and use. The algorithm has been standardized by the National Institute of Standards and Technology (NIST), and it is widely used in industry and government applications. The availability of standardized implementations of AES also makes it easy for developers to integrate the algorithm into their applications and systems, without the need for custom implementations [8].

Furthermore, AES is a symmetric key algorithm, which means that it is well-suited for use in applications that require fast and efficient encryption and decryption of data. Symmetric key algorithms are typically faster than public key algorithms, and they require less processing power and memory to operate. This makes them well-suited for use in embedded systems and other applications where resources are limited [7].

AES has a wide range of applications in various fields, including government, finance, healthcare, and the military. The algorithm is used to secure sensitive data and communications, and it is considered to be a critical component of modern information security systems.

One of the key applications of AES is in secure communication protocols. The algorithm is widely used in protocols such as SSL/TLS, which are used to secure online transactions and communications. AES is also used in Virtual Private Networks (VPNs), which provide secure remote access to networks and resources [7].

Another important application of AES is in file encryption. The algorithm is used to encrypt files and data on disk, which helps to protect sensitive information from unauthorized access. AES is widely used in file encryption software such as BitLocker and VeraCrypt, which provide high levels of security for sensitive data [8].

2.3 Performance Evaluation Metrics

Text encryption algorithm's performance can be evaluated using a variety of measures that are best suited for cryptographic methods including encryption time, decryption time, the throughput of encryption, the throughput of decryption, mean square error and memory utilization [9].

Encryption time: The entire amount of time needed to convert plain text into cipher text is known as the encryption time. The throughput of the encrypted algorithm is then determined using the computed encryption time.

Decryption time: Decryption time is the total amount of time needed to convert encrypted text into plain text. The throughput of the decrypted algorithm is then determined using the calculated decryption time.

Encryption throughput: The speed of encryption is determined by the throughput of the encryption algorithm. A reduction in the power consumption algorithm occurs as the throughput of the encryption method increases.

Decryption throughput: The speed of decryption is determined by the throughput of the decryption algorithm. The power consumption algorithm gets less efficient as the throughput of the decryption process increases.

Mean square error: The average difference of the bytes between the plaintext and encrypted text

Memory usage: The memory required for encryption and decryption.

2.4 State of the Art Review

Throughout this section, this paper discusses different research papers that have been conducted on the usage of Logistic maps and AES in text encryption as it is parallel to game code, which is text in its essence.

The authors of [10] propose a modification to the AES algorithm for text encryption. Their approach utilizes 12 rounds of the AES algorithm with dynamic key selection, where the key is chosen based on the length of the text to be encrypted. The proposed method is designed to provide better security than the standard AES algorithm, which uses a fixed key length. The authors provide a detailed description of their modified AES algorithm and present experimental results that demonstrate its effectiveness in encrypting text data. They also compare the performance of their approach with the standard AES algorithm and show that their modified algorithm achieves better results in terms of security and encryption time. Overall, the authors' work highlights the importance of developing customized versions of encryption algorithms to improve their performance and security in specific applications.

The authors of [11] propose an AES-based encryption and decryption scheme for both text and image data. The authors modify the standard AES algorithm by introducing an additional step of dynamic key scheduling, which enhances the security of the algorithm. They also implement a technique called bit-plane slicing for image encryption, which provides better security and faster processing time. The authors evaluate their proposed approach using several standard metrics for encryption quality and show that it performs well in terms of both security and efficiency. Their work highlights the importance of customizing encryption schemes to fit specific types of data and demonstrates the potential benefits of combining different techniques to achieve better performance.

The authors of [12] propose a file security system based on AES encryption algorithm. They discuss the key features of AES that make it an effective technique for data encryption, and provide a detailed description of the system design, including data flow diagram and encryption/decryption methods. The authors also evaluate the proposed system's performance and find that it provides high security with low computational overhead. The proposed system can be used in various applications that require secure file storage and transmission. The authors suggest future research directions, such as enhancing the system's user authentication and incorporating other security mechanisms. Overall, this paper provides a comprehensive analysis and design of an AES-based file security system that can offer high levels of security for various applications.

The authors of [13] present a study on the encryption and decryption of text using the Advanced Encryption Standard (AES) algorithm. The authors propose a method to encrypt and decrypt plain text using the AES algorithm with a 128-bit key. The encryption process involves four operations: substitution, shift rows, mix columns, and add round key, which are applied in a specific sequence. Decryption is done by applying the inverse operations in the reverse order. The authors evaluate the performance of the proposed method and report that it provides high security and efficiency. The study concludes that AES is a reliable and secure encryption algorithm for text data, and its implementation can be easily achieved using software libraries or programming languages.

The authors of [14] propose a modified AES algorithm to improve data security. The proposed algorithm replaces the key expansion process of AES with a more complex key generation scheme. Additionally, the SubBytes, ShiftRows, and MixColumns steps are combined into a single step to enhance the algorithm's efficiency. The proposed algorithm is evaluated by comparing its performance with the original AES algorithm in terms of encryption and decryption time and key sensitivity analysis. The experimental results demonstrate that the proposed algorithm provides better data security compared to the original AES algorithm.

The authors of [15] propose an application for file encryption and decryption using the Advanced Encryption Standard (AES) algorithm on Android devices. The application uses a graphical user interface that allows users to select files to be encrypted or decrypted. The AES algorithm is implemented using Java Cryptography Architecture (JCA) providers, and the application allows for selection of AES key size and mode of operation. The authors test the application using different file types and sizes and report successful encryption and decryption. The proposed application can be used to secure files on Android devices and ensure confidentiality of sensitive information.

The authors of [16] proposed a modified Advanced Encryption Standard (AES) algorithm that enhances the security of data communication. The modification is made by introducing additional operations to the encryption and decryption process, which increases the complexity of the encryption key. The authors evaluated the performance of the modified algorithm using the standard AES test vectors and found that the modified algorithm has a higher level of security with comparable encryption and decryption time. They also tested the algorithm using different sizes of plain text and found that the modified algorithm provides improved security with increased plaintext size. The proposed modification can be implemented in various applications that require secure data communication, including wireless sensor networks, cloud computing, and mobile devices.

The authors of [17] proposed a modified version of the Advanced Encryption Standard (AES) algorithm to enhance the security of data transmission. The proposed algorithm uses a substitution method to transform the plaintext into a more complex structure before it is encrypted using the AES algorithm. The substitution method consists of two parts, one based on bitwise operations and the other on arithmetic operations. The substitution method helps to make the data encryption more secure and robust. The proposed algorithm was tested using the Avalanche Effect Test and the results showed that the modified AES algorithm is highly secure as compared to the original AES algorithm. Moreover, the proposed algorithm has faster encryption and decryption speeds as compared to the original AES algorithm. The modified AES algorithm has potential applications in data security, secure data storage, and data transmission systems.

The authors of [18] proposed the implementation of the Advanced Encryption Standard (AES) algorithm to enhance the information security of web-based applications. They utilized AES-128, AES-192, and AES-256 to encrypt sensitive data on web-based applications. The encryption process was done on the client-side using JavaScript, and the encrypted data was sent to the server through HTTP/HTTPS protocol. The decryption process was performed on the server-side using PHP. The proposed system was tested for data integrity and confidentiality and proved to be effective in securing sensitive data in web-based applications. The authors concluded that the AES algorithm is a robust encryption technique and can provide a secure solution for web-based applications.

The authors of [19] propose a new technique for generating secure keys for AES-based IoT security using chaotic maps. The technique involves generating keys using both chaotic maps and Logistic maps, which are then used to encrypt and decrypt data transmitted between IoT devices. The authors show that their proposed method provides better security than traditional methods of key generation. They also compare their proposed technique with other well-known techniques and demonstrate its superiority in terms of security and efficiency. The paper concludes that using chaotic maps for key generation in AES-based IoT security provides a promising way to enhance the security of IoT devices and applications. Overall, this paper presents a novel approach to secure key generation for AES-driven IoT security, leveraging the power of chaos theory and Logistic maps.

The authors of [20] proposed a novel encryption method based on chaos theory and DNA computing for both text and image data. The proposed method uses a chaotic map to generate a sequence of numbers which is used as a key in DNA computing. The plain data is first converted into binary form and then encoded into DNA sequences based on a predefined rule. The key sequence is then used to shuffle the DNA sequences and generate a ciphertext. The decryption process reverses the process by extracting the DNA sequence from the ciphertext and reversing the DNA encoding to obtain the original binary data. The chaotic key sequence is then used to shuffle the binary data to obtain the original plaintext. The proposed method was tested on several text and image datasets and showed superior performance in terms of security and computational efficiency.

The authors of [21] proposed a text encryption method based on chaotic Logistic maps and nonlinear equations. The proposed method uses two chaotic Logistic maps to generate the initial conditions of nonlinear equations, which are then used to encrypt plaintext. To enhance the security of the encryption process, the key space is increased by using a set of secret parameters as input. The proposed method was tested on various plaintext samples, and the experimental results showed that it achieved high encryption efficiency and security. The proposed method has the potential to be used in various applications that require secure communication and data transmission.

The authors of [22] presented a text encryption scheme that uses a chaotic pseudo-random bit generator (CPRBG) based on a three-dimensional autonomous chaotic system. The scheme uses the C-SPN encryption algorithm, which combines substitution and permutation operations for enhanced security. The proposed CPRBG has a long key length, high correlation dimension, and satisfactory statistical properties. The encryption scheme is evaluated through several performance metrics such as encryption speed, encryption efficiency, sensitivity to the key, and resistance to attacks. The results show that the proposed scheme provides high security levels while maintaining high performance. The study highlights the potential of C-SPN encryption with a chaotic PRBG as an effective solution for secure communication applications.

The authors of [23] proposed a new chaos-based block cipher algorithm based on an enhanced Logistic map and simultaneous confusion-diffusion operations. The proposed algorithm employs the Logistic map with improved cryptographic properties, and XOR operations with the plaintext and a generated key stream to enhance the confusion stage. The diffusion stage is enhanced by performing permutation and substitution operations. The proposed algorithm is tested using statistical analysis and is found to provide better results compared to other existing algorithms in terms of key sensitivity, avalanche effect, and nonlinearity. The proposed algorithm is also shown to be resistant against common attacks such as differential and linear attacks, and brute-force attacks.

The authors of [24] provide a comprehensive review of the Advanced Encryption Standard (AES) algorithm and its various improvements based on time execution, differential cryptanalysis, and level of security. The authors discuss the importance of AES algorithm in modern-day security systems and highlight its limitations. They analyze and compare several AES variations such as Rijndael, AES-Like, and AES-Extended, and identify their strengths and weaknesses. The paper further evaluates the efficiency and security of these algorithms against various attacks. Finally, the authors propose future directions for research to enhance the security and performance of AES algorithm.

The authors of [25] proposed a cryptosystem for secure data transmission using Advanced Encryption Standard (AES) and steganography. The AES algorithm was used to encrypt the plaintext message, while steganography was employed to hide the encrypted message within the least significant bits of the image. The performance of the proposed cryptosystem was evaluated based on the encryption and decryption time, and the level of security provided by the system. The results showed that the proposed system was efficient in terms of execution time and provided a high level of security for the transmitted data. The proposed system can be applied in various fields such as military communication, banking systems, and e-commerce.

The authors of [26] presented the implementation of Advanced Encryption Standard (AES) algorithm and measures its time complexity for various inputs. The authors implemented AES algorithm in C language using different block sizes and key lengths. The experiments were conducted to analyze the time taken by the algorithm to encrypt and decrypt data of varying sizes. The results of the study showed that the time complexity of the AES algorithm was affected by the block size, key length, and input size. The study concluded that the AES algorithm is an efficient and secure method of encryption for data security.

The authors of [27] presented an effective implementation of the Advanced Encryption Standard (AES) and evaluated the avalanche effect in AES. They proposed an efficient algorithm for AES with a time complexity of O(n). The authors also analyzed the security of AES by measuring its avalanche effect, which measures how much change in plaintext affects the corresponding ciphertext. They showed that their implementation of AES achieved a high avalanche effect and hence enhanced security. The paper is a valuable resource for researchers and practitioners who are interested in the implementation and security analysis of AES.

The authors of [28] proposed an Enhanced Advanced Encryption Standard (AES) algorithm that employs a new encryption process with a substitution method to increase the encryption strength. The substitution method includes the generation of a random number matrix to replace the plain text matrix before encryption. The proposed algorithm also uses a modified key expansion process to generate a larger number of round keys. The performance of the proposed algorithm was evaluated using measures such as encryption time, avalanche effect, key sensitivity analysis, and entropy analysis. The results showed that the proposed algorithm provides better encryption strength and higher security than the original AES algorithm while maintaining reasonable encryption time.

The authors of [29] propose a technique for enhancing cloud computing security using the Advanced Encryption Standard (AES) for secure data storage. The proposed technique focuses on protecting data confidentiality and integrity by encrypting data before it is stored on the cloud. The authors implement the AES algorithm using a symmetric key for encryption and decryption of data. They also perform a comparative analysis of the proposed technique with existing encryption techniques, highlighting its superiority in terms of security and performance. The results of the analysis demonstrate that the proposed technique is effective in enhancing cloud computing security by providing strong encryption for stored data.

The authors of [30] propose a secure message transmission scheme based on AES double-layer encryption, which provides an additional layer of security for sensitive data. The scheme uses a combination of two AES encryption techniques: one uses a secret key, and the other uses the hash of the message. The scheme also includes a random initialization vector, which helps in preventing a known-plaintext attack. The authors conducted an extensive simulation to evaluate the security and efficiency of the proposed scheme. The results show that the proposed scheme provides an efficient and secure method for message transmission, as it has a high level of security, low computational cost, and low transmission overhead.

The authors of [31] proposed a double-layer image security scheme that utilizes mathematical sequences and Advanced Encryption Standard (AES). The scheme uses two layers of encryption, where the first layer is a combination of chaos-based Logistic map and Arnold's cat map, and the second layer is AES encryption. The proposed method also employs an aggregation process that enhances the security of the scheme by improving the confusion and diffusion properties. The authors conducted various experiments and analysis to validate the effectiveness and efficiency of the proposed scheme. The results showed that the proposed scheme offers improved security and robustness against various attacks, while maintaining reasonable encryption and decryption times.

The authors of [32] present a new image encryption algorithm based on a combination of the Logistic and sine maps. The proposed algorithm aims to improve the security and complexity of image encryption by applying multiple rounds of encryption with a secret key derived from the maps. The encryption process involves scrambling, diffusion, and mixing operations that increase the non-linearity of the cipher. The paper evaluates the proposed algorithm using standard measures, such as correlation coefficients and histograms, and compares it with existing image encryption methods. The experimental results demonstrate the effectiveness of the proposed algorithm in achieving high levels of security and robustness against various attacks.

The authors of [33] propose a new steganographic technique that hides secret data within sliced 3D medical images. The proposed method employs the Advanced Encryption Standard (AES) algorithm to securely encrypt the hidden message and a bit-cycling technique to embed the encrypted data into the least significant bits of the image. The use of bit-cycling helps to increase the amount of hidden data while maintaining the visual quality of the image. The experimental results show that the proposed method provides high levels of security and can hide a large amount of data within the image. The technique has potential applications in medical data transmission and storage where confidentiality and data integrity are of utmost importance.

The authors of [34] present a double-layer message security scheme based on 3DES, presented at the 2019 ICCAIS conference. It aims to enhance the security of message transmission by using a combination of encryption and decryption processes. The proposed scheme is implemented in two layers, where the first layer uses a modified form of 3DES and the second layer uses the standard 3DES algorithm. The scheme is tested against various attacks, including brute-force and statistical attacks, and is found to be highly secure. Overall, this scheme provides an efficient solution for secure message transmission in various applications.

2.5 Technology Comparison

In this section, we will discuss the different technologies that are used for the implementation of text encryption.

The authors of [35] used MATLAB for the implementation of the AES algorithm for encryption. The AES algorithm's infrastructure uses matrices as its fundamental building block. MATLAB includes robust numerical computation functions, particularly for array and matrix calculations. Therefore, it is simple to encrypt data based on the AES algorithm in the MATLAB environment.

The authors of [36] used Java programming language in the implementation of AES and RSA in securing health records. The Java programming language proved integrity, confidentiality, authentication, and non-repudiation, as it is more portable, object-oriented, multiplatform, and open source.

The authors of [33] provided a two-layer message security system using AES-18 in Wolfram Mathematica language. Which has performed according to the standards put by the authors and has provided reliable results.

The authors of [37] emphasized that one of C++AMP's strongest features is the ability to define and use templates, which enables the creation and use of highly reusable code. The built-in support for creating parallel applications is another strong point. However, there are certain limitations, such as the absence of the new and delete operators, the inability to convert pointers, and the lack of throw-try-catch.

The authors of [38] implemented a version of the Advanced Encryption Standard (AES) algorithm using the Python-based tool-flow, with block and key sizes of 128 bits. They came to the conclusion that Migen, when used in a Python environment, may create designs that perform better than those created by other high-level FPGA design tool-flows and that perform somewhat less effectively but are still fairly close to those created by hand-written RTL designs.

Chapter 3

Securing Game Code Using AES and Logistic Map

In this chapter, this paper discusses the proposed modified AES-128 and Logistic map algorithm in Wolfram Mathematica to allow for more secure encryption of data. The proposed algorithm will be implemented to encrypt basic java game code and will be tested for the metrics mentioned earlier in this paper.

3.1 Proposed Algorithm

In this section, this paper explains the proposed algorithm that will be used for the encryption and decryption of game code.

3.1.1 Encryption

The encryption process commences by taking the game code plaintext as input and selecting two initialization seeds for each encryption stage. The plaintext is then encrypted using AES-128 with a symmetric key generated from one of the initialization seeds, as depicted in Figure 3.1. The resulting encrypted text is converted into a byte-stream and subsequently transformed into a bit stream, which will be utilized in the XOR operation during the second encryption stage.

To generate encryption keys for the Logistic map encryption, the second initialization seed is employed to generate pseudo-random numbers using the Logistic map equation provided in Table 2.1. The encrypted bit stream is then XORed with the generated Logistic map keys to complete the second encryption stage. Finally, the bit stream is converted back into a byte stream and further transformed into a string using ASCII code representation.

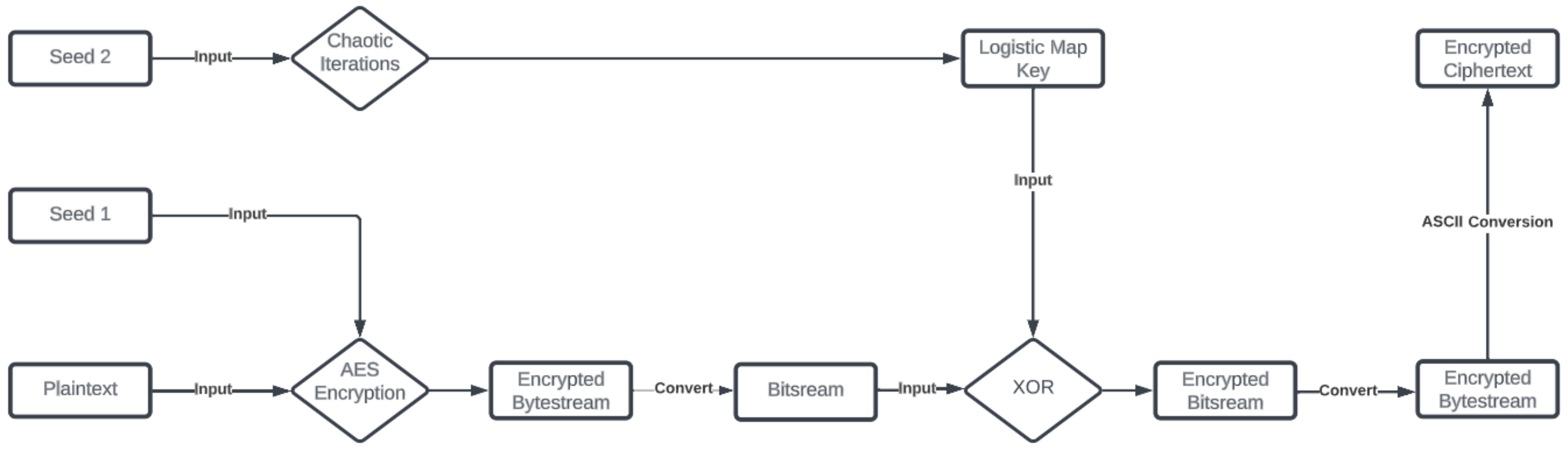


Figure 3.1: The flowchart for the proposed AES-Logistic map encryption algorithm

3.1.2 Decryption

During the decryption process, the first step involves converting the encrypted ciphertext into a byte stream, which is then transformed into a bit stream. The encrypted bit stream is subsequently subjected to an XOR operation with the Logistic map key that was used during the encryption process, as illustrated in Figure 3.2. The resulting bit stream is converted back into a byte stream and further converted into a string representation. Finally, the ciphertext undergoes decryption using AES-128 and the corresponding symmetric key, ultimately resulting in the recovery of the original plaintext.

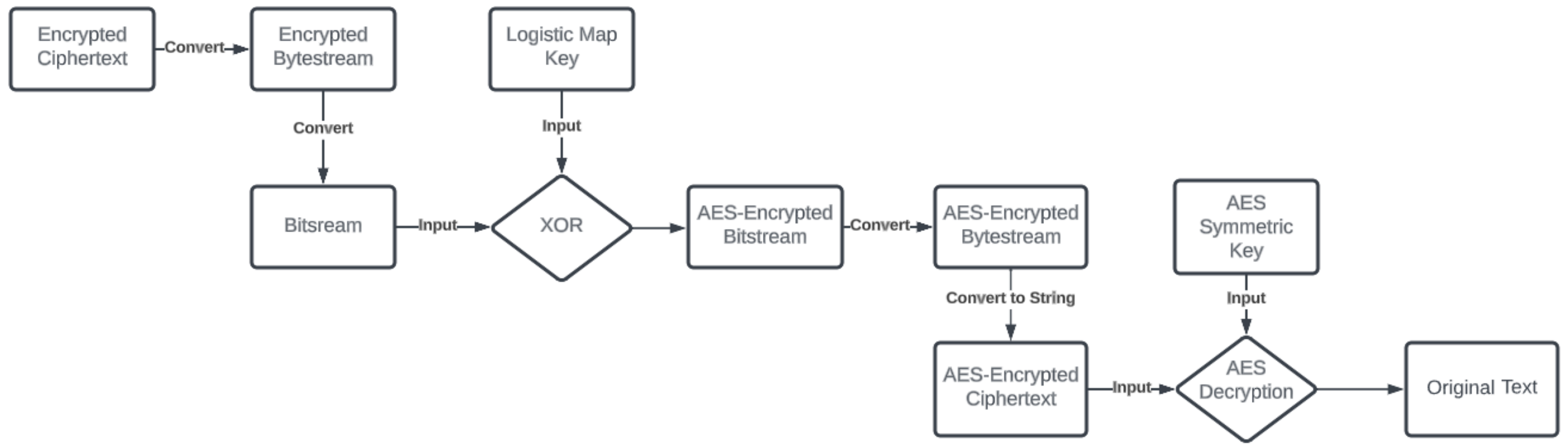
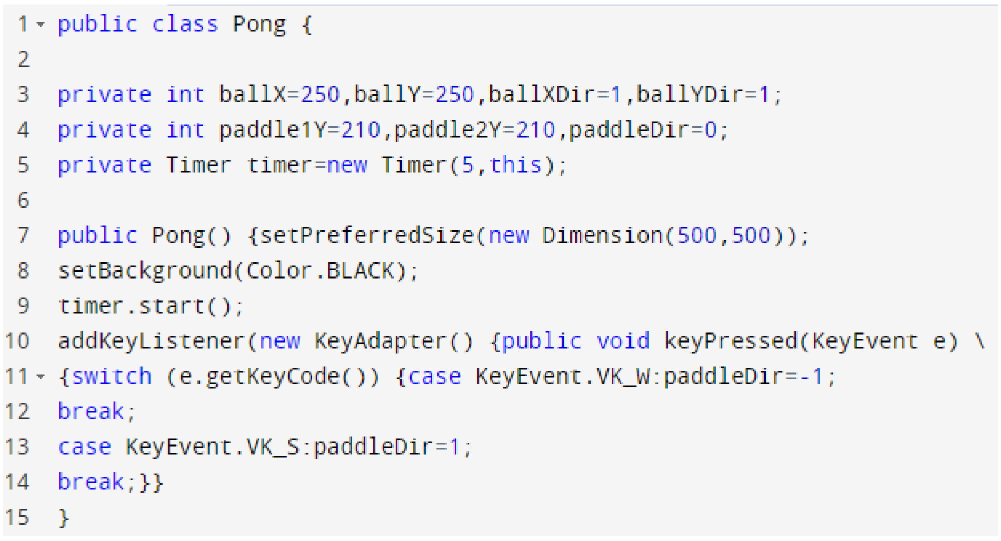


Figure 3.2: The flowchart for the proposed AES-Logistic map decryption algorithm

3.2 Results and Findings

In this section, this paper analyses and tests the proposed algorithm in accordance with the performance evaluation metrics mentioned earlier in this paper. The plain text that was used for the test is a portion of a simple Pong game code written in Java language containing 420 characters as shown in Figure 3.3, and the resulting ciphertext after encryption of the game code using the proposed algorithm is shown in Figure 3.4. The implementation and testing of the proposed algorithm was conducted using Wolfram Mathematica version 13.1 in the Wolfram language. The machine used to perform the testing and deployment is of the following specifications: 3.6 GHz AMD Ryzen 5 3600 6-Core Processor, with 16 GBs of RAM, and running on 64-bit Windows 10.

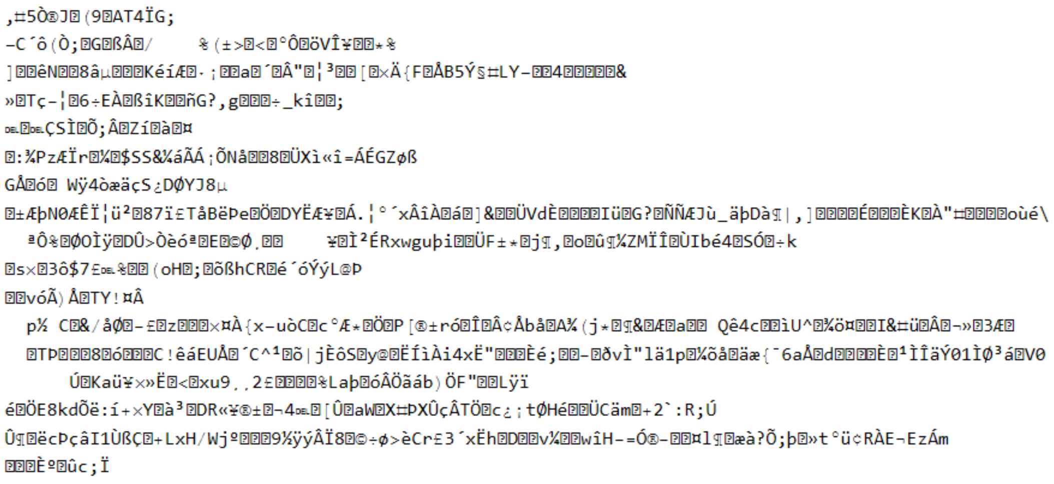


```java
public class Pong {

private int ballX=250,ballY=250,ballXDir=1,ballYDir=1;
private int paddle1Y=210,paddle2Y=210,paddleDir=0;
private Timer timer=new Timer(5,this);

public Pong() {setPreferredSize(new Dimension(500,500));
setBackground(Color.BLACK);
timer.start();
addKeyListener(new KeyAdapter() {public void keyPressed(KeyEvent e) \
{switch (e.getKeyCode()) {case KeyEvent.VK_W:paddleDir=-1;
break;
case KeyEvent.VK_S:paddleDir=1;
break;}}
}
```

Figure 3.3: Portion of the Java code for the game Pong



```
,¤5Ò®J(9AT4ÏG;
-C´ô(Ò;GßÂ/      ‰(±>< °ÔöVÎ¥*‰
]êN8âµKéíÆ·;a´Â"¦³[×Ä{FÅB5Ý§¤LY-4&
»Tç-¦6÷EÀßîKñG?,g÷_ki;
ÇSÌÕ;ÂZía¤
:¼PzÆÏr¼$SS&¼áÃÁ;ÕNå8ÜXì«î=ÁÉGZøß
GÅó Wý4òæäçS¿DØYJ8µ
±ÆþN0ÆÊÏ¦ü²87ï£TåBëÞeÖDYÉÆ¥Á.¦°´xÂîÀá]&ÜVdÈIüG?ÑÑÆJù_äþDà¶|,]ÉÈKÀ"¤oùé\
  ªÔ‰ØOÌÿDÛ>Òèóª E©Ø,      ¥Ì²ÉRxwguþiÜF±*j¶,oû¶¼ZMÏÎÙIbé4SÓ÷k
sx3ô$7£‰(oH;õßhCRé´óÝýL@Þ
vóÃ)ÅTY!¤Â
  p½ C&/âØ-£z×¤À{x-uòCc°Æ*ÖP[®±róÎÂ¢Åbâ A¾(j*¶&Æa Qê4ciU^¼ö¤I&¤üÂ-»3Æ
  TÞ8óC!êáEUÅ´C^¹õ|jÈôSy@ËÍìÀi4xË"Èé;-ðvÌ"lä1p¼õåæ{¯6aÅdÈ¹ÌÎäÝ01Ìφ³áV0
      ÚKaü¥×»Ë<xu9¸¸2£‰LaþóÂÖãáb)ÖF"LÿïÎ
éÖE8kdÕë:í+×Yà³DR«¥®±-4[ÛaWX¤ÞXÛçÂTÖc¿;tØHéÜCäm+2`:R;Ú
Û¶ëcÞçâI1ÙßÇ+LxH/Wjº9½ÿýÂÏ8©÷ø>èCr£3´xËhDv¼wîH-=Ó®-¤1¶æà?Õ;þ»t°ü¢RÀE¬EzÁm
Èºûc;Ï
```

Figure 3.4: Ciphertext after the encryption of the original game code

3.2.1 Encryption and Decryption Execution Times

In this subsection, this paper analyse the performance of the proposed algorithm in the context of encryption and decryption timings in comparison with the standard AES-128 algorithm and the standard Logistic map algorithm. Table 3.1 shows the timing evaluations after running the encryption and decryption on the game code.

From the results in Table 3.1, it is found that the proposed algorithm has a significantly higher encryption execution time than AES-128 and Logistic map, and that is due to the complexity of the proposed algorithm. It is also found that the proposed algorithm has a decryption execution time similar to AES-128.

Table 3.1: Timings comparison between the proposed algorithm, AES-128, and Logistic map

Metrics	Proposed AES-Logistic Map Algorithm	AES-128	Logistic Map
Encryption Time (ms)	115.512	72.060	43.451
Decryption Time (ms)	430.208	430.203	5.2×10^{-3}

3.2.2 Encryption and Decryption Throughput

In this subsection, this paper analyses the performance of the proposed algorithm in the context of encryption and decryption throughput in comparison with the AES-128 algorithm and the Logistic map algorithm. Throughput refers to the speed or rate at which data can be encrypted or decrypted by an encryption algorithm, and it is measured in terms of bytes processed per second. Table 3.2 shows the throughput evaluations after running the encryption and decryption on the game code.

Table 3.2 shows that the proposed algorithm has a slightly lower encryption throughput than AES-128, and a significantly higher throughput in comparison to Logistic map. The proposed algorithm has the highest decryrption throughput in comparison to both algorithms. This shows that the proposed algorithm has a higher efficiency in decryption in comparison to both algorithms, and a slightly lower efficiency when compared to AES-128.

Table 3.2: Throughput comparison between the proposed algorithm, AES-128, and Logistic map

Metrics	Proposed AES-Logistic Map Algorithm	AES-128	Logistic Map
Encryption Throughput (byte/s)	2.531×10^6	2.672×10^7	$978,122$
Decryption Throughput (byte/s)	1.317×10^7	425.010	1.056×10^{12}

3.2.3 Encryption and Decryption Memory Usage

In this subsection, this paper evaluates the performance of the proposed algorithm in the terms of encryption and decryption memory usage in comparison with the AES-128 algorithm and the Logistic map algorithm. Memory usage refers to the amount of computer memory, typically RAM, utilized during the encryption process. It represents the temporary storage required to perform encryption operations, such as storing intermediate results, encryption keys, and data buffers.

Table 3.3 shows that the proposed algorithm requires slightly higher amounts of memory compared to AES-128, and significantly higher amounts of memory in comparison with Logistic map for the encryption process. Moreover, The memory usage for the proposed algorithm in the decryption process is similar to Logistic map, and much higher than AES-128.

Table 3.3: Memory usage comparison between the proposed algorithm, AES-128, and Logistic map

Metrics	Proposed AES-Logistic Map Algorithm	AES-128	Logistic Map
Encryption Memory Usage (MB)	2.019	1.977	0.042
Decryption Memory Usage (MB)	5.703	1.84×10^{-4}	5.703

3.2.4 Mean Square Error

In this subsection, this paper analyses the performance of the proposed algorithm when it comes to the MSE in comparison with AES-128 encryption. Table 3.4 shows that the proposed algorithm has a higher MSE score than AES-128 encryption, and this shows that the proposed algorithm is more secure that the standard AES-128 encryption. However, further testing is required for a more accurate evaluation.

Table 3.4: MSE comparison between the proposed algorithm, AES-128, and Logistic map

Encryption Algorithm	Mean Square Error
Proposed Algorithm	6.45888×10^7
AES-128	6.45512×10^7

Chapter 4

Conclusions and Future Works

In this chapter, this paper briefly summarizes the extensive reading conducted in this paper and engage in a discussion about it. In conclusion, the author of this paper was driven to make a contribution to the subject of securing game code using advanced encryption standard and logistic map by thoroughly examining the relevant literature and conducting a comprehensive literature review, and proposing a two-stage encryption algorithm that uses logistic map for generating an encryption key to be XORed with the encrypted text generated by AES-128.

This paper includes an examination of the state-of-the-art as well as a comparison of different technologies used for the implementation of encryption algorithms. The encryption scheme proposed in this study was implemented in the Wolfram Language using Wolfram Mathematica. Various performance metrics, such as MSE, throughput, execution time, and memory usage, were analyzed and compared to other schemes from the literature, including AES-128 and logistic map. The performance evaluation demonstrated a promising security level for having a higher MSE score than AES-128, as well as comparable execution times and throughput. However, the algorithm is less efficient in some contexts when it comes to memory usage, with the difference increasing with the length of the plaintext. Scalability and resource availability can be significant concerns in this context, raising potential challenges and considerations.

References

[1] E. Britannica. “Chaos theory.” (2022), [Online]. Available: `https://www.britannica.com/science/chaos-theory` (visited on 12/26/2022).

[2] A. Akhavan, A. Samsudin, and A. Akhshani, “A symmetric image encryption scheme based on combination of nonlinear chaotic maps,” *Journal of the Franklin Institute*, vol. 348, no. 8, pp. 1797–1813, 2011.

[3] C. Oestreicher, “A history of chaos theory,” *Dialogues in clinical neuroscience*, 2022.

[4] T. Tsuchiya and D. Yamagishi, “The complete bifurcation diagram for the logistic map,” *Zeitschrift für Naturforschung A*, vol. 52, no. 6-7, pp. 513–516, 1997.

[5] P. T, B. Biswal, and V. Santhi, “Design and analysis of multimedia communication system,” Dec. 2011. DOI: `10.1109/ICoAC.2011.6165174`.

[6] A. Biryukov and D. Khovratovich, “Related-key cryptanalysis of the full aes-192 and aes-256,” in *Advances in Cryptology – CRYPTO 2009*, Springer, 2009, pp. 1–18.

[7] J. Daemen and V. Rijmen, *The design of Rijndael: AES - the advanced encryption standard*. Springer, 2002.

[8] National Institute of Standards and Technology (NIST), *Federal Information Processing Standards Publication 197: Advanced Encryption Standard (AES)*, `https://csrc.nist.gov/publications/detail/fips/197/final`, 2001.

[9] M. B. Bharathi, M. G. Manivasagam, and M. A. Kumar, "Metrics for performance evaluation of encryption algorithms," in *Int. Conf. Emerg. Trends Eng. Sci. Manag*, vol. 6, 2017, pp. 62–72.

[10] N. Mathur and R. Bansode, "Aes based text encryption using 12 rounds with dynamic key selection," *Procedia computer science*, vol. 79, pp. 1036–1043, 2016.

[11] K. R. Saraf, V. P. Jagtap, and A. K. Mishra, "Text and image encryption decryption using advanced encryption standard," *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)*, vol. 3, no. 3, pp. 118–126, 2014.

[12] K. Muttaqin and J. Rahmadoni, "Analysis and design of file security system aes (advanced encryption standard) cryptography based," *Journal of Applied Engineering and Technological Science (JAETS)*, vol. 1, no. 2, pp. 113–123, 2020.

[13] R Padate and A Patel, "Encryption and decryption of text using aes algorithm," *International Journal of Emerging Technology and Advanced Engineering*, vol. 4, no. 5, pp. 54–59, 2014.

[14] P. Kumar and S. S. B. Rana, "Development of modified aes algorithm for data security," *Optik*, vol. 127, no. 4, pp. 2341–2345, 2016.

[15] S. Tayde and S. Siledar, “File encryption, decryption using aes algorithm in android phone,” *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 5, no. 5, 2015.

[16] O. C. Abikoye, A. D. Haruna, A. Abubakar, N. O. Akande, and E. O. Asani, “Modified advanced encryption standard algorithm for information security,” *Symmetry*, vol. 11, no. 12, p. 1484, 2019.

[17] P. Kawle, A. Hiwase, G. Bagde, E. Tekam, and R. Kalbande, “Modified advanced encryption standard,” *International Journal of Soft Computing and Engineering*, vol. 4, no. 1, pp. 21–23, 2014.

[18] J Simarmata, T Limbong, M. Ginting, R Damanik, M. Padli, A. Nasution, *et al.*, “Implementation of aes algorithm for information security of web-based applications,” *Int. J. Eng. Technol*, vol. 7, no. 3.4, 2018.

[19] Z. Rahman, X. Yi, I. Khalil, and M. F. R. Sumi, “Chaos and logistic map based key generation technique for aes-driven iot security,” in *International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, Springer, 2021, pp. 177–193.

[20] M. Babaei, “A novel text and image encryption method based on chaos theory and dna computing,” *Natural computing*, vol. 12, no. 1, pp. 101–107, 2013.

[21] A. Akgül, S. Kacar, B. Arıcıoğlu, and I. Pehlivan, “Text encryption by using one-dimensional chaos generators and nonlinear equations,” in *2013 8th International Conference on Electrical and Electronics Engineering (ELECO)*, IEEE, 2013, pp. 320–323.

[22] C. K. Volos, I. M. Kyprianidis, and I. N. Stouboulos, "Text encryption scheme realized with a chaotic pseudo-random bit generator," *Journal of Engineering Science & Technology Review*, vol. 6, no. 4, 2013.

[23] M. Alawida, J. S. Teh, A. Mehmood, and A. Shoufan, "A chaos-based block cipher based on an enhanced logistic map and simultaneous confusion-diffusion operations," *Journal of King Saud University-Computer and Information Sciences*, 2022. DOI: https://doi.org/10.1016/j.jksuci.2022.01.019.

[24] M. E. Hameed, M. M. Ibrahim, and N. Abd Manap, "Review on improvement of advanced encryption standard (aes) algorithm based on time execution, differential cryptanalysis and level of security," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 10, no. 1, pp. 139–145, 2018.

[25] M. M. Yahaya and A. Ajibola, "Cryptosystem for secure data transmission using advanced encryption standard (aes) and steganography," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT)*, vol. 5, no. 6, pp. 317–322, 2019.

[26] S More, "Implementation of aes with time complexity measurement for various inputs," *Global Journal of Computer Science and Technology*, vol. 15, no. E4, pp. 11–20, 2015.

[27] A. Kumar and N. Tiwari, "Effective implementation and avalanche effect of aes," *International Journal of Security, Privacy and Trust Management (IJSPTM)*, vol. 1, no. 3/4, pp. 31–35, 2012.

[28] V Sumathy and C Navaneethan, “Enhanced aes algorithm for strong encryption,” *International Journal of Advances in Engineering Technology*, vol. 4, no. 2, pp. 547–553, 2012.

[29] V. R. Pancholi and B. P. Patel, “Enhancement of cloud computing security with secure data storage using aes,” *International Journal for Innovative Research in Science and Technology*, vol. 2, no. 9, pp. 18–21, 2016.

[30] W. Alexan, A. Hamza, and H. Medhat, “An aes double–layer based message security scheme,” in *2019 International Conference on Innovative Trends in Computer Engineering (ITCE)*, IEEE, 2019, pp. 86–91.

[31] M. T. Elkandoz, W. Alexan, and H. H. Hussein, “Double-layer image security scheme with aggregated mathematical sequences,” in *2019 International Conference on Advanced Communication Technologies and Networking (CommNet)*, IEEE, 2019, pp. 1–7.

[32] M. T. Elkandoz, W. Alexan, and H. H. Hussein, “Logistic sine map-based image encryption,” in *2019 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA)*, IEEE, 2019, pp. 290–295.

[33] S. Yasser, A. Hesham, M. Hassan, and W. Alexan, “Aes-secured bit-cycling steganography in sliced 3d images,” in *2020 International Conference on Innovative Trends in Communication and Computer Engineering (ITCE)*, IEEE, 2020, pp. 227–231.

[34] N. Adam, M. Mashaly, and W. Alexan, “A 3des double-layer based message security scheme,” in *2019 2nd International Conference on Computer Applications Information Security (ICCAIS)*, IEEE, 2019, pp. 1–5.

[35] Q. Zhang and Q. Ding, "Digital image encryption based on advanced encryption standard (aes)," in *2015 Fifth International Conference on Instrumentation and Measurement, Computer, Communication and Control (IMCCC)*, IEEE, 2015, pp. 1218–1221.

[36] M. A. Sadikin and R. W. Wardhani, "Implementation of rsa 2048-bit and aes 256-bit with digital signature for secure electronic health record application," in *2016 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, IEEE, 2016, pp. 387–392.

[37] S. Mocanu, G. Munteanu, and D. Saru, "Gpgpu optimized parallel implementation of aes using c++ amp," *Journal of Control Engineering and Applied Informatics*, vol. 17, no. 2, pp. 73–81, 2015.

[38] K. O. Setetemela, K. Keta, M. Nkhabu, and S. Winberg, "Python-based fpga implementation of aes using migen for internet of things security," in *2019 IEEE 10th International Conference on Mechanical and Intelligent Manufacturing Technologies (ICMIMT)*, IEEE, 2019, pp. 194–198.

Appendix A

Wolfram Mathematica Code for the Proposed AES-Logistic Map Algorithm

A.1 AES Encryption Code

```
Key = GenerateSymmetricKey[Method -> "AES128"];

M = "private int ballX=250,ballY=250,ballXDir=1,
ballYDir=1;
private int paddle1Y=210,paddle2Y=210,paddleDir=0;
private Timer timer=new Timer(5,this);\[
IndentingNewLine]
public Pong() {setPreferredSize(new Dimension(500,500))
;\
\[IndentingNewLine]setBackground(Color.BLACK);\[
IndentingNewLine]\
timer.start();\[IndentingNewLine]addKeyListener(new

KeyAdapter() \
{public void keyPressed(KeyEvent e) {switch (e.

getKeyCode()) {case \
KeyEvent.VK_W:paddleDir=-1;\[IndentingNewLine]break;\

\[IndentingNewLine]case \

KeyEvent.VK_S:paddleDir=1;\[IndentingNewLine]break;}}"


ML = StringLength[M];


RL = Mod[ML, 8];


added = 8 - RL;


If[RL != 0 , NM = StringPadRight[M, ML + added], NM = M

];
NM;


Msg = StringPartition[NM, 8];


d = {};

ini = {};


AbsoluteTiming[Table[

  eobj = Encrypt[key, Msg[[i]]];

  bin =

   eobj["Data"] // Normal // IntegerDigits[#, 2, 8] &

// Flatten;
  initvec =

   eobj["InitializationVector"] // Normal //

     IntegerDigits[#, 2, 8] & // Flatten;

  d = Join[d, {bin}];
  ini = Join[ini, {initvec}];
  , {i, 1, Length[Msg]}]];

df = Flatten[d];
dfl = Length[df];

iniF = Flatten[ini];
iniFL = Length[iniF];

dfl + iniFL;

bin = d;
initvec = ini;
```

A.2 Logistic Map Encryption

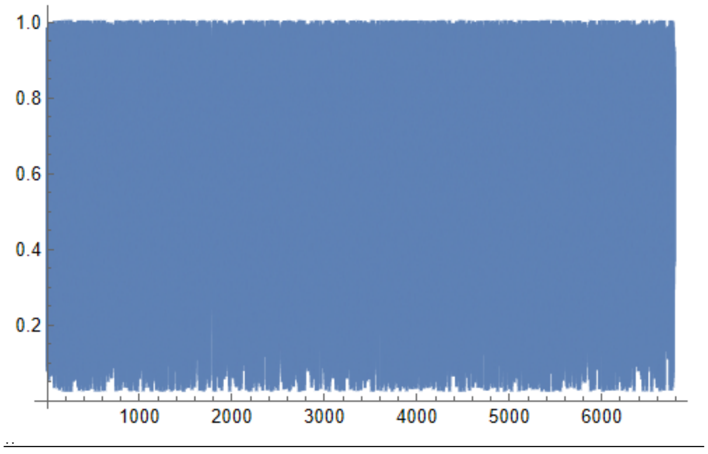
```
MsgBits = df;

MsgLength = Length[Flatten[IntegerDigits[MsgBits, 2]]];

sol = RecurrenceTable[{x[n + 1] == 3.6*Tan[x[n]]*(1 - x
[n]),
  x[0] == 0.5}, x, {n, 1, MsgLength}];

solLength = Length[sol];

ListPlot[sol, Joined -> True]
```



```
Lkey = ConstantArray[0, solLength];
For[i = 1, i < solLength + 1, i++,
If[sol[[i]] < 0.6, Lkey[[i]] = 0, Lkey[[i]] = 1]];

AbsoluteTiming[CodeEncrypted = BitXor[Lkey, MsgBits]];

ciph = CodeEncrypted;
ciph = Partition[CodeEncrypted, 8];
ciph = Table[FromDigits[ciph[[i]], 2], {i, Length[ciph
]}];
ciph =  FromCharacterCode[ciph];
```

A.3 Logistic Map Decryption Code

```
DecryptedMessage = BitXor[CodeEncrypted, Lkey];
```

A.4 AES Decryption Code

```
DeMsg = Table[
 eobj3 =
  EncryptedObject[
   Association[
    "Data" ->
     ByteArray[(FromDigits[#1, 2] &) /@
       ArrayReshape[bin[[i]], {16, 8}]],
    "InitializationVector" ->
     ByteArray[(FromDigits[#1, 2] &) /@
       ArrayReshape[initvec[[i]], {16, 8}]],
    "OriginalForm" -> String]];


        EnMSG = Decrypt[key, eobj3], {i, 1, Length[d
]}];

t = Partition[DeMsg, 1];

StringJoin[t];
```