AN IMPROVED DNA CRYPTOGRAPHY ALGORITHM USING RESIDUE NUMBER SYSTEM

HABIB LEKAN SALAUDEEN 18/27/MCS018

MARCH, 2021

SCHOOL OF POSTGRADUATE STUDIES (SPGS)

AN IMPROVED DNA CRYPTOGRAPHY ALGORITHM USING RESIDUE NUMBER SYSTEM

BY

HABIB LEKAN SALAUDEEN

18/27/MCS018

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF MASTER OF SCIENCE (M.Sc.) DEGREE IN COMPUTER SCIENCE

A THESIS SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY, KWARA STATE UNIVERSITY, MALETE, NIGERIA.

MARCH, 2021

DECLARATION

I, **SALAUDEEN, Habib Lekan** with Matriculation Number **18/27/MCS018** hereby declare that this thesis titled "**An Improved DNA Cryptography Algorithm using Residue Number System**", is a record of my research. It has neither been presented nor accepted in any previous application for higher degree.

HABIB Lekan Salaudeen Signature / Date

APPROVAL PAGE

This is to certify that this thesis was carried out by **SALAUDEEN Habib Lekan** with the matriculation number **18/27/MCS018** has been read and approved as meeting the requirements of the Department of Computer Science for the award of the degree of Master of Science (M.Sc.) in Computer Science.

Prof. K.A. Gbolagade — 05/05/2021

Main Supervisor — Signature / Date

Dr. J.F Ajao — 05/05/2021

Co-Supervisor — Signature / Date

Prof. K.A. Gbolagade — 05/05/2021

Head of Department — Signature / Date

Dr. Lambe Adesina — 16/05/2021

Internal Examiner — Signature / Date

External Examiner — Signature / Date

Prof Hamzat Abdulrahman — 17-05-21

Dean,
School of Postgraduate Studies(SPGS) — Signature / Date

Office of the Dean
R E C E I V E D

DEDICATION

This thesis is dedicated to God Almighty, the beneficent, the merciful, the fountain of all knowledge and to Him be the glory and honor, forever and ever Aameen.

ACKNOWLEDGEMENTS

All praise is due to Almighty Allah. I praise Him and seek His guidance and forgiveness. I thank Him for giving me the strength and knowledge to complete this phase of my life journey and for my continued existence on the earth. He is the Creator, the All wise, All-knowing and without him this won't have been possible.

I appreciate the utmost effort of my supervisor, Head of Department and The Dean, Faculty of Information and Communication Technology, Prof. Kazeem Alagbe Gbolagade (a great mentor and international scholar) whose patience, support and encouragements have been the driving force behind the success of this work. He took time out of his tight schedules to guide me and go through this thesis.

Special thanks go to my most loving, understanding Mum, Alhaja Sikirat Salau, for the support financially, mentally, with a lot of encouragement and immeasurable prayer that got me through this. Thanks Mum, love you. And siblings; Dr. Tijani Salau, Mrs. Habiba Salau Dauda, Mrs. Jemila Salau Ibrahim, Mrs Nafisa Salau Ismail, I love and appreciate you all.

Finally, I also convey my regards to the Post Graduate Coordinator; Dr. R.M. Isiaka and all other staff of department; Dr. J.F. Ajao, Dr. R.S. Babatunde, Dr. A.N. Babatunde, Mr. S.O. Abdulsalam, Mrs. S. Yusuf, Mrs. F. Balogun, Mr. D. Popoola and Mr. A. Kadri.

Thank you all, may Almighty Allah bless you all.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

Abstract

In this information rich era, data protection is needed to ensure swift and secure communication through a digital medium. Data need to be protected from unauthorized access and transmitted to the intended receiver with confidentiality, availability, integrity and authenticity. Several schemes have been proposed over the years toward ensuring that data sent over a digital medium is difficult to understand by an intruder by hiding or transforming the data from a plain-text to a DNA based form which is sent as the cipher-text. Although, some of the schemes performed to a certain level but high computational time is the problem of most of these techniques. Therefore, this research presented a method that incorporate cryptography in securing the data using residue number system and DNA cryptography. Firstly, residue number system is used in order to reduce the computational time by encrypting plaintext into a residue form, using the moduli set {2n-1, 2n, 2n+1}, then convert the residue to a DNA format. The DNA cryptography was done to hide the existence of the encrypted DNA by using two different reference sequence randomly generated. The proposed scheme produces the cipher text in both an encrypted and encoded DNA based form, which takes less computational time and attracts less attention of intruders.

Keywords: DNA; Data; Computational time; Cryptography; Residue number system

CHAPTER ONE

INTRODUCTION

1.1 Background to the study

Communicating digitally has evolved to be a fundamental aspect of interaction between two ends in this generation, with a lot of internet based field. Keeping communication secret is of high importance. So, the security of data passed over a network is a primary consign, which stretches to the confidentiality as well as integrity of the data, making it mandatory to protect against intruders or unauthorized access and use. In the quest to have a secured communication between two ends, the concept of cryptography and steganography came into light. Cryptography enables you to store sensitive information or transmit it across the insecure networks (like the internet) such that it cannot be read by anyone other than the intended recipient (Badoni & Jain, 2014).

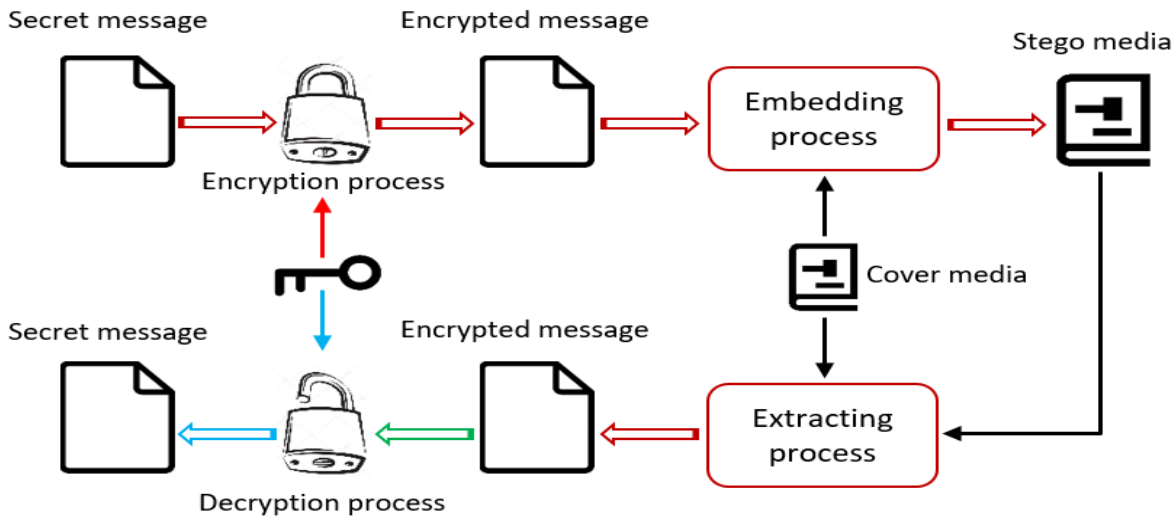

Figure 1.1: Basic diagram of combining steganography and cryptography (Taha, et.al, 2019)

Cryptography and steganography are usually interrelated and share the common aims and services of preserving the confidentiality, integrity, and availability of information, which are some of the most significant fields in computer security (Krishnan, 2017). Cryptography and steganography are methods allowing information to be sent securely (Sokół, 2005). Cryptography is an historical science that began in Egypt around 1900 B.C. with hieroglyphic writing (Siper, 2005). It uses encryption to scramble the secret information in such a way that only the sender and the intended receiver can reveal it (Selvaraj, 2017). On the other hand, steganography began in ancient Greece around 440 B.C. (Malathi, et.al., 2017). It hides the secret information in different carriers in which the visibility of private information is made unavailable to unauthorized users. This is done by concealing the sensitive information within cover mediums such as images, video, and DNA in such a way that it becomes difficult to detect (Siper, 2005).

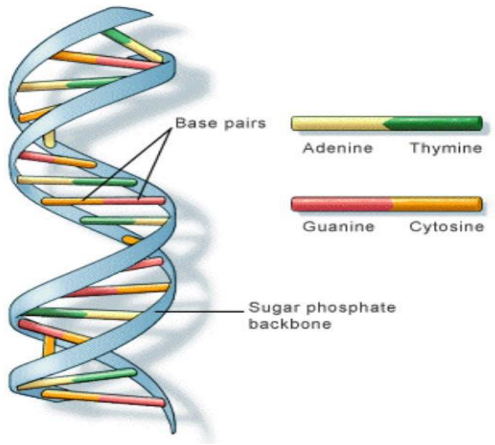

Figure 1.2: DNA Structure (Omer & Farooq, 2015)

Several algorithms have been proposed in image steganography for hiding secret information inside an image. However, the embedding capacity of the image is low, so it cannot hide a large data inside it, (Malathi, 2016). In order to overcome the deficit of the capacity, DNA steganography has been introduced. DNA steganography is a research direction of DNA

cryptography, which started in 1999. This approach uses DNA sequences as carriers to enable secure transfer of the critical data (Sokół, 2005). The principal idea is basically to encrypt and conceal messages in a large number of DNA strands to prevent adversaries from reading and deciphering the messages. This could be achieved only if the original sequences are preserved from adversaries (Sharma, 2016). Hiding data in DNA sequences is a new and evolving scientific field. This research aims to propose an enhanced approach to improve the security level and also lower the computational time to produce a cipher text.

The security is the most important factor to be considered when the confidential information is transmitted through the internet. The prevention of unauthorized access to the confidential data is achieved by the two security features known as cryptography and steganography. The cryptography deals with the encryption and decryption process of the data that is transmitted through an open network (Meera, 2015). The sender encrypts the data using key value and the key value is exchanged with the recipients to decrypt the received encrypted message. The cryptography scheme secures the text format information but fails to secure the information in image or video format. The steganography is the scheme of providing the security to the confidential information by concealing it within the cover mediums like the image, video, audio, DNA, etc. The visibility of the confidential information is made unavailable to the unauthorized users and it provides additional security by using different types of algorithm to hide it. Based on the cover mediums used the steganography varies like image steganography, DNA steganography, audio steganography, video steganography etc. The image steganography has a lot of algorithms to hide the confidential information inside the image with the variation of security it provides. The embedding capacity of the image is low and it cannot hide a huge data inside it (Malathi, 2017). To overcome the shortage of embedding capacity the DNA steganography is introduced.

The major issue to design any encryption and decryption algorithm is to improve the security level such that the implementation of such algorithm on data makes it difficult and complex for such data to be understood by an intruder.

Residue Number System (RNS) is an integer number system with the competency to support parallel, carry-free addition, borrow-free subtraction and single step multiplication without unfinished product. Data Conversion in RNS is usually based on either the Chinese Remainder Theorem (CRT) or the Mixed Radix Conversion (MRC), which can be categorized into forward and reverse conversions. The forward conversion involves converting a binary or decimal number into it RNS equivalent while the reverse conversion is the inverse operation, which involves converting RNS number into binary or decimal. Relatively, reverse conversion is more complex (Aremu & Gbolagade, 2017).

Even though a level of success is achieved when it comes to DNA cryptography algorithm by (Malathi, et.al., 2017), which modified the existing DNA insertion algorithm, but the system has high the computational time problem. Therefore, in other to ensure messages are sent securely and faster over a network, this research proposes an approach that will secure the identity of the message being sent with lower the computational time using DNA cryptography and residue number system.

1.2 Statement of Problem

Nowadays, the fastest means of transmitting data is through the internet and one of the most cogent issues is the security and privacy of the data been transmitted through an unsecure network like the internet because sensitive data like bank details, private files and confidential information are transmitted over it. Data is at a high risk of being theft or breached by the intruders, hackers or internet wanderers.

There are various modern techniques of cryptography and steganography which ensures the security attributes namely; Availability, Confidentiality, Integrity and Authenticity. But there

are memory and computational issues with these techniques. To overcome these issues, DNA computing techniques have been developed. These techniques are effective as they provide huge memory and parallel computing of DNA.

Recently, (Malathi, et.al., 2017) proposed a scheme to solve the drawback of the existing schemes by modifying the existing DNA insertion algorithm, they introduced two key values to lower the cracking probability but the problem with the system is that the longer the text to be encrypted the higher the computational time.

Furthermore, (Roy, et.al., 2017) proposed a crypto-stego system that used delayed chaotic neural network with DNA complementary pair rule but the drawback with the system is high computational time because of the complexity nature of delayed chaotic neural network.

Having understand both schemes, it is obvious that a better scheme is required, which will not only be a hybrid system but also lower the computational time and consume less memory as much as possible.

1.3 Aim and Objectives

The aim of this research is to implement an improved DNA cryptography using residue number system.

The objectives are to;

i. Formulate a model for encryption and decryption process using DNA cryptography and Residue Number System with respect to the moduli set {2n-1, 2n, 2n+1}.
ii. Implement the formulated model in (i).
iii. Evaluate and compare the performance of the system.

1.4 Scope of the Study

This research work will be limited to the use of the alphanumeric characters and keyboard symbols for the plain-text, Residue Number System will be adopted for cryptography using the moduli set {2n-1, 2n, 2n+1} for forward conversion while Chinese remainder theorem is used for the decryption process.

1.5 Significance of the study

This scheme will allow encrypted data to be sent securely and faster over a network without the knowledge of an intruder. It also provides an improvement over DNA sequence (insertion method) towards ensuring that transmission of vital information between one end to the other is difficult to detect. It will further showcase how residue number system can be applied in cryptography

1.6 Research Layout

This research is organized into five chapters. Chapter One covers the area of introduction to the research, showcasing the background of the study, the statement of problem to justify the reason behind the proposed scheme, the aim and objectives to solve the identified problem, as well as the scope and significance of the study. Chapter Two deals with review of literature, consisting of previous related works and concepts that are related to the research. Chapter Three describes the existing system as well as the mode of operation of the proposed system. Chapter Four presents the result, evaluation and comparison of the scheme. Chapter Five consists of the summary, conclusion and major contributions to knowledge, and recommendation of the proposed scheme as well as relevance for future works on the research area.

CHAPTER TWO

LITERATURE REVIEW

2.1 Related Concept

2.1.1 Data Security

Data security is the process of securing the data and protecting it from unauthorized and corrupted access. Not all data might be sensitive, but others might be private and valuable. When unauthorized access to such data is enabled, it may create problems as it can be used by people who should not be using it. Data security is the measure which is taken to prevent the loss of data through these unauthorized accesses. There are many ways to protect data, and some of them include strong user authentication, cryptography, steganography, data erasure, backup etc. A key data security technology measure is encryption, where digital data, software/hardware, and hard drives are encrypted and therefore rendered unreadable to unauthorized users and hackers. One of the most commonly encountered methods of practicing data security is the use of authentication. With authentication, users must provide a password, code, biometric data, or some other form of data to verify identity before access to a system or data is granted.

2.1.2 Cryptography

Cryptography is an ancient art; the first documented use of cryptography in writing dates back to circa 1900 B.C. when an Egyptian scribe used non-standard hieroglyphs in an inscription. Some experts argue that cryptography appeared spontaneously sometime after writing was invented, with applications ranging from diplomatic missives to war-time battle plans. It is no surprise, then, that new forms of cryptography came soon after the widespread development of computer communications. In data and telecommunications, cryptography is necessary when communicating over any untrusted medium, which includes just about any network,

particularly the Internet (Nemati & Hamid, 2010). There are five primary functions of cryptography; Privacy/confidentiality, Authentication, Integrity, Non-repudiation, Key exchange.

In cryptography, the process start with the unencrypted data, referred to as plaintext. Plaintext is encrypted into cipher text, which will in turn (usually) be decrypted back into usable plaintext. The encryption and decryption is based upon the type of cryptography scheme being employed and some form of key. For those who like formulas, this process is sometimes written as:

$$C = E_k(P) \quad (2.1)$$

$$P = D_k(C) \quad (2.2)$$

where **P** = plaintext, **C** = ciphertext, **E** = the encryption method, **D** = the decryption method, and **k** = the key.

Given this, there are other functions that might be supported by crypto and other terms that one might hear:

- Forward Secrecy (Perfect Forward Secrecy): This feature protects past encrypted sessions from compromise even if the server holding the messages is compromised. This is accomplished by creating a different key for every session so that compromise of a single key does not threaten the entirely of the communications.
- Perfect Security: A system that is unbreakable and where the cipher-text conveys no information about the plaintext or the key. To achieve perfect security, the key has to be at least as long as the plaintext, making analysis and even brute-force attacks impossible. One-time pads are an example of such a system.

- Deniable Authentication (Message Repudiation): A method whereby participants in an exchange of messages can be assured in the authenticity of the messages but in such a way that senders can later plausibly deny their participation to a third-party(Kessler, 2020).

2.1.2.1 Types of Cryptography

1. **Symmetric key cryptography:** is a type of cryptography in which the single common key is used by both sender and receiver for the purpose of encryption and decryption of a message. This system is also called private or secret key cryptography and AES (Advanced Encryption System) is the most widely uses symmetric key cryptography. The symmetric key system has one major drawback that the two parties must somehow exchange the key in a secure way as there is only one single key for encryption as well as decryption process. AES (Advanced Encryption Standard), DES, Triple DES, RC2, RC4, RC5, IDEA, Blowfish, Stream cipher, Block cipher, etc. are the types of symmetric key cryptography.

2. **Asymmetric Key Cryptography:** is completely different and a more secure approach than symmetric key cryptography. In this system, every user uses two keys or a pair of keys (private key and public key) for encryption and decryption process. Private key is kept as a secret with every user and public key is distributed over the network so if anyone wants to send message to any user can use those public keys. Either of the key can be used to encrypt the message and the one left is used for decryption purpose. Asymmetric key cryptography is also known as public key cryptography and is more secure than symmetric key. RSA is the most popular and widely used asymmetric algorithm. RSA, DSA, PKCs, Elliptic Curve techniques, etc. are the common types of asymmetric key cryptography.

3. **A Hash Function:** is a cryptography algorithm that takes input of arbitrary length and gives the output in fixed length. The hash function is also considered as a mathematical equation that takes seed (numeric input) and produce the output that is called hash or message digest. This system operates in one-way manner and does not require any key. Also, it is considered as the building blocks of modern cryptography. The hash function works in a way that it operates on two blocks of fixed length binary data and then generate a hash code. There are different rounds of hashing functions and each round takes an input of combination of most recent block and the output of the last round. Some popular hash functions are Message Digest 5 (MD5), SHA (Secure Hash Algorithm), RIPEMD, and Whirlpool. MD5 is the most commonly used hash function to encrypt and protect your passwords and private data.

2.1.3 Steganography

Steganography word is the combination of two Greek word's "stego" and "graphia". Stego means "cover" and grafia means "writing". Staganalysis is a technique to examine the existence of steganography generally practiced by staganalyst. Steganography is an approach for hiding the information or plain text into carrier medium or hiding one form of information into another form of information. Steganography is the method of secret communication generally known for creating less suspect as its beautiful feature. It provides mechanism which encode information in particular manner that ensures presence of information invisible. In this process original file is called cover file and after hiding information on that cover file is called stego file. To hide the secret information there is need of stego-key for encoding and decoding the information (secret message). In this system scenario, first select appropriate hiding medium (i.e. Text, Image, Audio/Video or Protocol). Some steganography algorithms are also available

but approach is always need to select more efficient algorithm that is able to encode information in secure manner (Dixit, et.al., 2019).

2.1.3.1 Types of Steganography

1. **Text Steganography:** In this method the cover text is explain by changing words within a text, arranging the formatting of an existing text to hide the message, generating character sequences which are random in nature, or by utilizing the concept of context free grammar.
2. **Image Steganography:** This technique is most popular than other steganographic technique. Bulk of e-image information is present on internet, that are generated from digital devices such as from digital cameras s. Mostly image contain some type of noise within itself. Noise refers to the defect inherent in the procedure of rendering an analog picture as digital image. In this type of steganography message is hidden in pixels of an image. In this secret communication scheme, hide secret message in a digital image using some hiding procedure. Another end or receiver can regenerate the original plain text from the stego image by decoding procedure. Original image is called carrier image and image obtained after hiding message into carrier image is called stego- image. There are some methods available to hide the secret information such as Data hiding, Data embedding and Data extracting method.
3. **Audio Steganography:** In this type of stego procedure message is covered using audio as cover medium.
4. **Video Steganography:** Is the method for hiding secret information inside a video. Addition of the secret sensitive plain text (secret information containing message) inside a video is done in such a fashion that it is unrecognizable by the human's eye. This can be achievable because some negligible changes of the color of the pixel. In

video steganography first extract the frame. This frame is chosen randomly to ensure the security characteristics i.e. confusion. Hide the secret message into this randomly selected frame. This stego frame is again placed in its correct location and video is reconstructed. Check the integrity of the video before sending to other side i.e. receiver. It is now ready to send to receiver side. Original video is recognized as cover video and message containing video is called stego video.

2.1.4 DNA Sequence Coding

Each DNA strand has four chemical bases: A, C, G, and T. Biologically, A is associated to T and C is associated to G. In the binary computing area, the synthesis of DNA bases can be changed by the input decisions, by assuming that T is associated with C or T is associated with G, and so on (Sureshraj & Bhaskaran, 2012). In order to store data in DNA molecules, researchers have to encode a secret message into DNA bases using a binary coding rule to combine with the DNA sequence. Researchers have the option of selecting any equivalent binary form for each base (A): the binary forms can be '00', '01', '10', or '11', and so on. This coding along with the randomness properties makes DNA a befitting application for both computing and cryptography. Therefore, the coding of DNA to binary form can give 4! = 24 different encoding ways (Singh & Singh, 2015). Which in turn makes it possible to carry out logical operations such as Addition, Subtraction, XOR, AND, OR, and NOT over the DNA bases.

F(X): X → Y, where X = {A, C, G, T} and Y = {00, 01, 10, 11}. This can be express as in Table 2.1.

Table 2.1: DNA Binary Coding

DNA Base	Binary code
A	**00**
C	**01**
G	**10**
T	**11**

Several carriers have been used in data hiding algorithms where each carrier has it is own characteristics. Depending on the specific carrier, different techniques have been used to hide secret information. The process of hiding and the mechanism of each technique is different from another. According to the techniques, different changes will happen during the hiding process. In DNA data hiding, three techniques were proposed in (Shiu, et.al., 2010), all of which were considered to be the main techniques of hiding data in DNA sequences. The three main techniques can be defined as follow:

2.1.4.1 Insertion Technique

This technique depends on the merging among the reference of DNA sequence (S) which use as a carrier and secret message. During the process of this technique, both of them are translated into a binary system according to any binary coding rule. After that, the DNA reference is separated into equal sized segments in order to insert each bit of the secret message after each segment of DNA reference and then converting back into a DNA sequence resulting in a stego DNA. Furthermore, less modification rate is considered as a great feature of this technique because it depends on inserting secret data in the DNA reference not replacing the contents of DNA reference (Bhateja & Mittal, 2015). However, the main disadvantage of this technique is the increase in redundancy during the process, and the stego DNA length will be higher than

that of the DNA reference. This implies that the use of this technique will attract the attention of unauthorized users (Hamed, et.al., 2015).

2.1.4.2 Complementary Rule Techniques

In this technique, the procedure begins with the selection of a DNA sequence in which the longest existing complementary pair is contained. This is followed by the random generation of two complementary string pairs whose length is one more than that existing in the sequence, after which these pairs are padded with a 'T' at the posterior and anterior. Afterwards, they are inserted one at a time into S while ensuring that there is no overlapping. The message is then divided into segments, each containing an even number of bits after which the data is coded back into nucleotides using the binary coding rule. For each pair of a complementary substring in the converted sequence, a message bit is inserted before TajT, where aj represents the pair of longest complementary substrings. A resultant sequence containing message S' is then obtained. This scheme results in a significantly alters the length of the DNA sequence which rouses the suspicion of a hacker to the existence of an embedded message (Bhateja & Mittal, 2015). Legally there are six main complementary rules as it is shown in Figure 1.

AT	TC	CG	GA
AT	TG	GC	CA
AC	CT	TG	GA
AC	CG	GT	TA
AG	GT	TC	CA
AG	GC	CT	TA

Figure 2.1: The Six Complementary Rules (Dilovan, Habibollah, Subhi, 2017**)**

2.1.4.3 Substitution Technique

Regarding this technique there is no merge between reference DNA sequence and the secret information. In this scheme, specific positions in the DNA reference are selected randomly as determined by the algorithm. After that, at least one complementary rule should be selected to

replace each letter of the message with the DNA contents in particular locations. Depending on the contents of the message, the process will be carried out to obtain the stego DNA. Hence, the DNA length is maintained following the embedding of the message only the replacement has done between secret message and DNA reference. This, in turn, means that in an effort to conceal the secret data, the resulting stego DNA is highly modified (Bhateja & Mittal, 2015). As a result, this technique is considered as a more efficient technique than the previous techniques because it provides more complexity and better performance (Taur, et.al., 2012).

2.1.5 Residue Number System

Residue Number System comprises of a set of moduli that are independent of each other. An integer is represented by the residue of each of the moduli and arithmetic operations are based on residues individually. The advantages of using the RNS over the conversional system include parallelism, modularity, "a carry-free" operation and fault tolerance (Gbolagade & Cotofana, 2008). These features make RNS to be widely acceptable and used in Digital Signal Processing (DSP) applications such as image processing, convolution, digital filtering and fast Fourier transform (Alhassan & Gbolagade, 2013).

Let $\{m_1, m_2, m_3,\ldots,m_n\}$ be a set of positive integers all greater than 1. mi is called a modulus, and then n-tuple set $\{m_1, m_2, m_3 \ldots,m_n\}$ is called a moduli set. Consider an integer number Y. For each of the modulus in $\{m_1, m_2, m_3 \ldots,m_n\}$, we have $y_i = Y \bmod m_i$, (which will be denoted as $|Y|m_i$). Thus the number Y in this system is represented as $Y = (y_1, y_2, y_3,\ldots,y_n)$,$0 \leq y_i < m_i$. (Siewobr, Gbolagade & Cotofana, 2014).

Given the moduli set {7,8,9}, the number 150 can is depicted in RNS as:

$$y_1 = |y|m_1 = |150|_7 = 3$$

$$y_2 = |y|m_2 = |150|_8 = 6$$

$$y_3 = |y|m_3 = |150|_9 = 6$$

Thus, the RNS representation of 150 is $(3,6,6)_{RNS(7|8|9)}$

In order to avoid redundancy, the moduli set must be pair wise relatively prime. Thus gcd (m_i ,m_j) = 1 for i≠j, where gcd stands for greatest common divisor of m_i ,m_j.

Let M = $\prod_{i=1}^{n} mi$, then the RNS representation is unique for any integer M is called the dynamic range. (Siewobr & Gbolagade, 2012).

Decimal to Residue (D/R) converter (encoder) is needed in order to convert a decimal number to RNS representation.

2.1.6 Forward Conversion

Forward conversion is done by a forward converter which decomposes a weighted binary number into a residue represented number with regards to a moduli set. It is the conversion from a conventional representation to a residue representation by dividing the number X by each of the given moduli and then collecting their remainder, (Gbolagade & Cotofana, 2008). Taking the example, moduli set {3, 4, 5}, 4 is depicted in RNS as: Therefore, the depiction of 4 in RNS as:

$$x_i = |X|m_i \qquad (2.3)$$

$$x_1 = |x|m_1 = |4|_3 = 1;\ x_2 = |x|m_2 = |4|_4 = 0;\ x_3 = |x|m_3 = |4|_5 = 4$$

therefore, the RNS representation of 4 is $(1, 0, 4)_{RNS(3|4|5)}$

2.1.7 Reverser Conversion

The conversion from residue to a conventional representation (either binary or decimal representation) is known as reverse conversion. The two most widely used techniques of reverse conversion are the Mixed Radix Conversion (MRC) and Chinese Remainder Theorem (CRT), (Gbolagade, 2009).

2.1.7.1 Conversion by Chinese Remainder Theorem (CRT)

For a moduli set with and a dynamic $\{m_1, m_2, \ldots, m_k\}$ with $\gcd(m_i, m_j) = 1$ for $i \neq j$ and a dynamic range $M = \prod_{i=1}^{n} mi$, the residue number $\{x_1, x_2, \ldots, x_k\}$ can be converted into the decimal number X if the moduli set are co-prime (Salifu & Gbolagade, 2016).

$$X = \left| \sum_{i=1}^{k} m_i \left| m_i^{-1} x_i \right|_{mi} \right|_m \quad (2.4)$$

$M = \prod_{i=1}^{n} m_i$ and $M_i = \frac{M}{mi}$ and M_i^{-1} is the multiplicative inverse of M_i with respect to m_i.

Example:

Finding the decimal equivalent of the RNS number (0, 0, 4) with respect to the moduli set {3,4,5}.

Solution:

$$M = \prod_{i=1}^{N} mi = 3 \times 4 \times 5 = 60$$

$M_1 = M/m_1 = 60/3 = 20$ then $= M_1^{-1} = 2$

$M_2 = M/m_2 = 60/4 = 15$ then $= M_2^{-1} = 3$

$M_3 = M/m_3 = 60/3 = 12$ then $= M_3^{-1} = 3$

$$X = |0 \times 20 \times 2 + 0 \times 15 \times 3 + 4 \times 12 \times 3|_{60}$$

$$= |0 + 0 + 144|_{60}$$

$$= 24$$

2.2 Review of Related Works

Al-Harbi, et.al., (2020), investigates the most recent data hiding techniques based on DNA steganography, including the highly improved DNA-based steganography technique, the data hiding using double DNA sequences method, and the enhanced DNA-based steganography technique. The strengths and weaknesses of these techniques are discussed. Additionally, the security of these techniques is analyzed based on several security parameters that measure the

quality of DNA steganography with respect to many factors, including, but not limited to, cracking probability, blindness, modification rate and expansion rate, and layers of security. The goal of the comparison between the investigated techniques is to highlight the advantages and disadvantages of the existing data hiding algorithms and to motivate future research in this field. Moreover, the paper evaluates the discussed techniques based on some parameters, including capacity, payload, and bit per nucleotide (bpn). The result shows that the enhanced DNA-based steganography technique hides 2 bpn, whereas the highly improved method can hide on average 1.46 bpn, which is higher than data hiding using double DNA sequences method can hide. The paper also presents suggestions for how each technique can be optimized to achieve a higher security level for hiding data within DNA sequences.

El-Latif, et.al., (2019), suggested a method which has two rounds of encryption. This scheme is the same as the existing technique named the Data Encryption Standard (DES) algorithm. In this method, two keys are used for encoding the plaintext. These two keys are made up of the elliptic curve cryptography (ECC), and Gaussian kernel function (GKF) and another key is created on random based injective mapping on the second characters repeated in the first key. At last, the encryption message arbitrarily hides in the second DNA sequence based on the numbers from GKF.

Sohal, et.al., (2018), introduced a new method with the cryptographic technique. In this technique, client-side data is encrypted before storing it in the cloud. This is a symmetric-key cryptography scheme which uses DNA cryptography. Apart from presenting the thorough design of this approach, and comparing it with the present symmetric-key algorithms (DNA, AES, DES, and Blowfish), the experimental results show that this method leaves behind the traditional algorithms based on ciphertext size, encryption time, and throughput. Hence this new method is much more efficient and performs better.

Tiwari, et.al., (2018), recommended a scheme in which the DNA mapping technique was offered for ECC. In this method the DNA code is random, and non-repetitive subsections are allotted to alphabets. Then these alphabets are used for encoding and decoding at the two ends. This scheme was effectively employed and used in real- time internet of things devices.

Malathi, et.al., (2017), modifies the insertion algorithm to decrease the cracking probability of the cipher DNA sequence. The algorithm uses two different keys. The first key **(K1)** is a number in the range of 0 to 255, which is used to XOR the last character in the message (**M**); the result will be XORed with the character preceding the last one in the M, and so on. Accordingly, the first key is used to encrypt the message. The second key **(K2)** is randomly generated and is used to divide the DNA sequence into same-length segments. The resulting cipher characters are inserted as binary bits one by one at the beginning of each segment. Then, the binary sequence is converted into DNA sequence. The second key is preferred to be a small number so that the DNA sequence has a minimum length while hiding the secret message.

Algorithm 2.1: Malathi, et.al., (2017).

Encoding

Step 1: Randomly generate a binary sequence which is to be used as a cipher 's'.

Step 2: The message 'M' to be modified is split onto characters M= {m1, m2, m3…mn} and each element is converted to its 8-bit binary equivalent based upon ASCII standards.

Step 3: The last element of set M is XORed with s.

Step 4: The result is the XORed with the element preceding the last one in the set M and repeated till all the elements are converted and stored in 'A'.

Step 5: The binary sequence in A is converted to the protein sequence.

Step 6: A sample DNA sequence 'S' is taken and along with a randomly generated number n less than the size of M. Step 7: Break S down into n divisions such that size of each division sums up to the size of S.

Step 8: Similarly, break down the generated DNA sequence A and insert each division of A onto the divisions of S using the insertion algorithm.

Step 9: Generate the binary form of the sample DNA sequence S by using the coding methods.

Decoding

Step 1: Convert the encoded DNA sequence(S') into binary form.

Step 2: Divide S' sequence into parts of length s1+r1, s2+r2… sP+rP.

Step 3: Now from all the small strings obtained from step 2 extract the first r (i) bits.

Step 4: Combine all the remaining s (i) strings to get the DNA sequence.

Step 5: Join all the extracted bits in step 3 to form the message in binary.

Step 6: Convert the binary message and sequence to ATGC form by using the coding rule.

Step 7: Get the binary equivalent from the insertion decryption output and store in M

Step 8: Divide M into pieces of 8 bits each {m1, m2, m3….mn-2, mn-1, mn}

Step 9: For i=n: 2 A= (XOR mn-1 with mn) +A

Step 10: A= (XOR m1 with key) +A

Step 11: This binary is converted to its ASCII equivalent to obtain the message

Roy, et.al., (2017), proposed a new method using delayed chaotic neural network with a posterior DNA cryptography. The binary sequence needed to perform XOR operation with message blocks is generated from chaotic neural network. The permutation of the plaintext and the number of epoch is also based on the chaotic neural network. It is difficult for any cryptanalyst to determine the actual parameters of the encryption method and decrypt the DNA cipher sequence. Without knowing all the parameters i.e. input, delay function of the chaotic neural network as well as the position bits for using permutation operation. The scheme is slower for use in practical applications. In case of online media file transmission, the scheme would not provide efficient solution.

Algorithm 2.2: Roy, et.al., (2017)

Step 1: From *N*0 epoch, calculate the start point *y*0 by *y*0 = *y1*(*N*0*h*).

Step 2: Equally divide the plaintext message, *k* into fixed length subsequences *kj* of length ε (ε = 4):

$$k = \frac{r0,r1,\ldots,r\varepsilon-1}{k0}, \frac{r\varepsilon,r\varepsilon+1,\ldots,r2\varepsilon-1}{k1}, r_{2\varepsilon} \quad (2.5)$$

If the length of the last subsequence is less than 4 bytes, the remaining length is filled with subsequent *Z* no of 0's. The number of zeroes needed to fill up the last subsequence can be 8, 16 or 24. After dividing the plaintext, message block *Rj* is formed by combining *rj* , *rj+1* , *rj+2* , *rj+3* of 32-bits size. $R_j = r_j + r_j+1 + r_j+2 + r_j+3$.

Step 3. To obtain the binary sequence, Random binary sequence generation method is followed which produces $Sj = si^1\ si^2 \ldots si^{42}$. Sj^1 is based upon the selection of 5 bits from si^{33} to si^{41} by sender. It is sent to the receiver along with other secret parameters. $Sj\ ^1 = si^{33}\ si^{34}\ldots si^{41}$ (Selected 5 bits), $Sj\ ^2 = si^{42}$. For $i = 4$ in Eq. (6), the binary sequences are generated after 42 epoch of the chaotic network from (3). The first 32 bits are used as the key S_j. The decimal value of S_j 1 is calculated as V_j . The chaotic neural net is iterated *Vj* times after the encryption of the message block.

Step 4. Each message block Rj is permutated with a left cyclic shift of Vj bits and message block Sj is shifted in right cyclic order by Vj bits. As a result, Message blocks Rj´and Sj´ is received.

Step 5. The trajectory of neural map is selected on the value of S_j^2. It the value is 0, the y_1 (*t*) trajectory is selected. If the value is 1, *y2* (*t*) trajectory is selected for block iterations explained in step 3. S_j^2 works as a selection functions for the trajectories.

Step 6. Perform XOR operation on R_j´ and S_j´ to obtain the 1st level encrypted ciphertext message block $\acute{C}_j$.

$$\acute{C}_j = R_j \oplus S_j' \quad (2.6)$$

The ciphertext message block $\acute{C}j$ can be divided into 8-bits subparts which are $\acute{c}\,j$, $\acute{c}\,j+1$, $\acute{c}\,j+2$, $\acute{c}\,j+3$ for the plaintext subparts rj , $rj+1$, $rj+2$, $rj+3$ respectively.

Step 7. The binary data bits in $\acute{C}j$ are now converted into 2nd level encrypted DNA sequences by following simple DNA encoding technique where consecutive '00' represents '*A*', '01' represents '*G*', '10' represents '*C*' and '11' represents '*T*'. Thus, the cipher message block is now converted into a small DNA sequence *Cj*.

Step 8. After processing all the plaintext message blocks, the plaintext has been encrypted by the proposed algorithm. If not, let $y0 = y\,B_j^2 + 1\,((38 + V_j)\,h)$, where S_j^2 is the trajectory selection operator. Then jump to step 2.

The DNA cipher-text *Cj* is transferred along with *Z*, which contains the number of zero added at the end of the message. The decryption process is the reverse of the encryption process.

After DNA decoding by the reverse technique as step 7, 1st level decrypted message block $Dj\,'$ is obtained. The following equation is the key to produce the 2nd level decrypted message block D_j'' from the decrypted message block $Dj\,'$ by the help of the binary sequence *Sj*.

By inverse operation on the permutation, D_j'' is shifted by V_j bits to obtain message block *Rj*.

All the message blocks are recovered. Z no of zeroes is removed from the end of the binary message before converting all the binary to plaintext at the receiver end. They are divided into bytes which eventually return the plaintext.

Pushpin, (2017), proposed a DNA based encryption algorithm that has several processes that are implemented to hide the information, that makes it difficult for an attacker to crack the information and get back the plain text which has been sent by the sender. The random selection of DNA nucleotide is implemented for providing secured transmission of messages over the network. There are three stages in this algorithm – Encryption, random key table generation and Decryption. In the first stage the source data is encrypted using binary coding and

complementary rules are applied. In second stage random key is generated which is used for next level of encryption. In the third stage decryption process takes place which is reverse process of encryption. This method provides more security to the information and hence it is not easy accessible for a hacker to crack the encrypted message. Only the intended receiver can extract the message by decrypting the cipher information to get the original information. The analysis of the proposed technique shows that this is more powerful against certain attacks. This method ensures data integrity and confidentiality over data transmission.

Menaka, (2014), proposed a data hiding method where the algorithm first randomly selects a DNA sequence. The message to be encoded is then taken and each letter in the faked DNA sequence. Each letter in the message is converted into its ASCII equivalent and they are then converted into equivalent binary form. Each two digits in the converted binary sequence are converted using a specific table. Then, the message index position (first position of each letter) in the faked DNA sequence is applied to each letter of the converted sequence. Each digit in the resultant sequence is replaced with its equivalent binary value and the equivalent alphabet value is replaced for the binary value. For example, if the obtained binary value is 010 011 101 …, then it will be replaced as C D F… where A has the value 000, B has 001 and so on. The resultant sequence of alphabets is transmitted over to the receiver. In the receiver side, the reverse process is done in which the original receiver knows the complementary rules and the randomly selected DNA sequence. The message to be sent is then encoded with the fake DNA sequence.

Gupta and Jain, (2014), symmetric-key encoding algorithmic rule supported the DNA approach is projected. The initial key sequence is enlarged to desire length victimization projected key growth technique guided by the pseudo random sequence. The advantage is that there's no need to send an extended key over the channel. The variable key growth in encoding method combined with DNA addition and complement makes the technique sufficiently secure. A

DNA sequence consists of four nucleic acid bases A (adenine), C (cytosine), G (guanine), T (thymine), wherever A and T are complementary, and G and C are complementary. Also use C, T, A and G to denote 00, 01, 10, 11 (the corresponding decimal digits are ―0123‖). By victimization this encoding technique every 8-bit component worth of the gray scale image is pictured as a nucleotide string of length four. Reciprocally to decrypt the nucleotide string will get a binary sequence simply. In total 4! = 24 forms of writing, there are only 8 of them will meet complementary rule, for instance, the decimal digits ―0123‖ (the corresponding binary range is ―00011011‖) will be encoded in to one of them, like ―CTAG‖, ―CATG‖, ―GATC‖, ―GTAC‖, ―TCGA‖, ―TGCA‖, ―ACGT‖ or ―AGCT‖. There are total six legal complementary rules [3] that are as follows: (AT)(TC)(CG)(GA), (AT)(TG)(GC)(GA), (AC)(CT)(TG)(GA), (AC)(CG)(GT)(TA), (AG)(GT)(TC)(CA), (AG)(GC)(CT)(TA). Any one of them for instance, (AG) (GC) (CT) (TA) is applied to projected methodology.

Gupta and Singh, (2013), has been projected a DNA Based Cryptological Techniques for an encryption algorithm based on OTP (one-time-pad) that involve data encryption using traditional mathematical operations and/or data manipulating DNA techniques. However once an encryption algorithm has been applied and therefore the data is transmitted on the transmission media: there's a clear stage that the data, although within the cipher type gets manipulated by any interceptor.

Debnath, (2013), developed an algorithm for data encryption using DNA sequencing. In their algorithm, they have used the concept of indexing the DNA Sequencing and transmitting the message to the receiver. They have not used any complementary rules.

Cheng, (2012), the hiding procedure substitutes another letter for an existing letter on a special location set by the algorithm. The embedding algorithm encompasses a conversion operates that converts a given letter with a selected letter outlined by the complementary rule. For

example, if a complementary rule is outline as (AC)(CG)(GT)(TA), then the result of θ(G) are going to be T, and therefore the result of θ(T) are going to be A. To boot, the substitution methodology can convert the letter s into s (unchanged), θ(s) and θ (θ (s)) once the secrete message is 0, 1 and no data, respectively.

Mohammad, et al., (2011), proposed an information hiding methodology wherever data was efficiently encoded and decoded following the properties of DNA sequence. Complementary combine rules of DNA were employed in their methodology.

Jin-Shiuh, et.al., (2010), proposed a way referred to as Table Lookup Substitution methodology (TLSM) that might double the capability of message activity. In TSLM, they need replaced the complementary rule with a rule table. The key plan of the TLSM is to increase the 1-bit complementary rule into a 2-bit rule table so every conversion of letters will represent 2 bits of the secret message.

CHAPTER THREE

METHODOLOGY

3.1 Existing System

From the existing literatures reviewed, it is clear that noticeable weakness exists in the previous schemes proposed. The weakness pertaining to most is the simplicity attached to the algorithms, making them easy to crack and at the same time making the encoded text easily detected. It is to be noted that the main aim for both encoding and encrypting algorithms is to prevent a confidential message from an unauthorized party. But when such purpose has not been achieved effectively, it is necessary to provide a better or more improved means of ensuring that the main goal is accomplished.

3.2 Proposed Methodology

Improving the existing system towards providing a better means through which secure communication can be ensured from one end to the other, a new scheme was proposed. The concept of residue number system was adopted to encrypt the corresponding ASCII value of the character entered, through computation of forward conversion. The scheme contains one DNA lookup table which is used for substitution when encoding the encrypted message into a DNA sequence. with the DNA sequence implementing an insertion method, the data encoded in DNA base is merge together with two different reference DNA sequence randomly chosen online to form a cipher DNA sequence which is sent to the receiver.

The reverse conversion, which is the decryption process makes use of the Chinese remainder theorem (CRT) method, which first returns the cipher-text back from RNS form to ASCII form

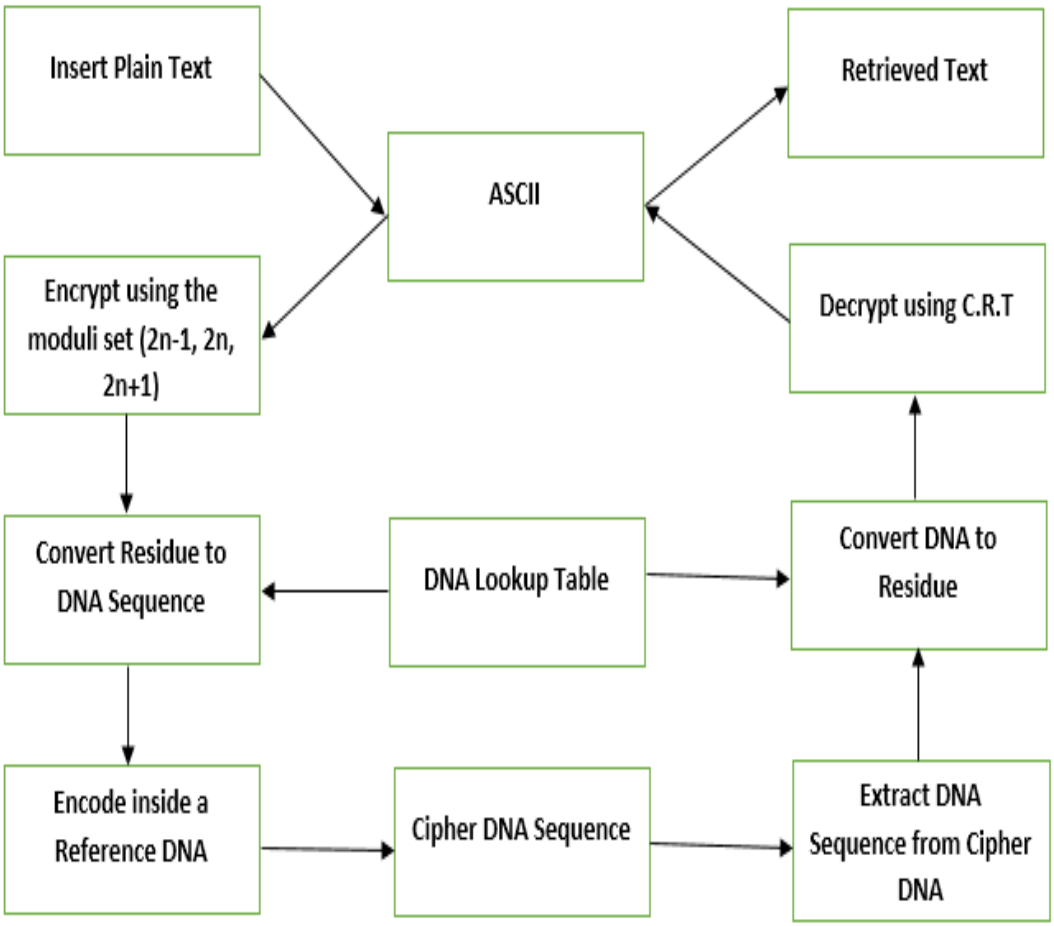

Figure 3.1: System Architecture

which is then converted to the original message after the system has make use of the lookup table to convert the DNA sequence back to it binary form and then to it equivalent residue form.

This will be achieved using the developed system as shown in Figure 3.2.

Figure 3.2: System Interface

The system accepts the text that is entered in the text field. Thereafter, the user will be required to click the “Encryption using RNS” button, which will trigger the system to perform both the conversion of the text entered to ASCII form and forward conversion. After the encryption, the user will be required to click the “DNA Key Generation” button in order to convert the encrypted message to a DNA sequence. Then the user is required to click the “DNA Insertion method” to encode DNA sequence into a reference DNA to generate a cipher DNA sequence.

To decrypt the cipher text, user only needs to click the “Decryption” button, enabling the system to reverse the process to produce the plain text.

3.2.1 Proposed Algorithm

Algorithm 3.1

Encryption

Step 1: Given Plain-text **S** and Convert **S** into their respective ASCII numbers (in decimal format)are grouped into blocks.

Step 2: The ASCII numbers are converted into residues with respect to the moduli set {2n-1, 2n, 2n+1}.

Step 3: The residues are converted into 4-bit binary number (0's and 1's).

Step 4: The sequence of binary numbers is broken in pairs. The pairs could be 00, 01, 10, 11 These pairs are converted into a DNA sequence using Table 3.1.

Step 5: Based on Step 4, the DNA sequence as **S'** is created.

Step 6: Two reference sequence **R** and **Q** are chosen randomly online out of 163 million available.

Step 7: DNA sequence **S'** is inserted in-between the two reference sequence **R** and **Q** to create Cipher DNA **G**

Step 8: Return **G** and send **G** to the receiver.

Decryption

Step 1: The Cipher DNA **G** is received from sender.

Step 2: DNA sequence **S'** is extracted from the cipher DNA **G.**

Step 3: DNA sequence **S'** is converted into binary and splitted into 4-bits.

Step 4: The binary numbers are converted into residue (decimal).

Step 5: The residues are converted into ASCII numbers (decimal) using Chinese Remainder Theorem.

Step 6: ASCII numbers are converted back to Plain-text **S**.

Step 7: Return **S**.

3.2.2 Formation of Binary to DNA Sequence table

For an encrypted message to be hidden inside a DNA sequence, a table was formed to substitute the message from binary to DNA sequence. It is understood that there are four DNA nucleotide bases; Adenine, Cytosine, Guanine and Thymine (A, C, G. T). The substitution table incorporates four DNA nucleotide bases with binary bits 00, 01, 10, 11 as shown in Table 3.1.

Table 3.1: DNA lookup table

DNA Base	Binary code
A	**00**
C	**01**
G	**10**
T	**11**

3.2.3 Transposition

The first stage of the transposition involves performing the conversion of each of the English alphabets present in the message to it ASCII equivalent. The forward conversion takes place for each of the characters using the moduli set {2n-1, 2n, 2n+1}. A value n is assigned to the moduli set which serves as a secret key. Using the moduli set it is expected that for each character, three residue numbers will be computed. From the computed residue numbers, a DNA sequence is generated and the DNA sequence is encoded in between two reference DNA sequences to generate the cipher-text.

The transposition is achieved using;

$$x = m_i[x/m_i] + |x|_{mi} \quad (3.1)$$

where m is the moduli and x is the number to be transposed.

3.2.4 Encryption process

The encryption stage which is transposition and encoding will be achieved using the procedure below.

1. Convert each of the characters to it equivalent ASCII value
2. Transpose each of the ASCII values using forward conversion based on the moduli set {2n-1, 2n, 2n+1}, to generate residue of each of the ASCII.
3. Substitute each of the residue with a DNA sequence based on DNA lookup table.
4. Encode the DNA sequence in a reference DNA sequence to form a cipher DNA sequence.

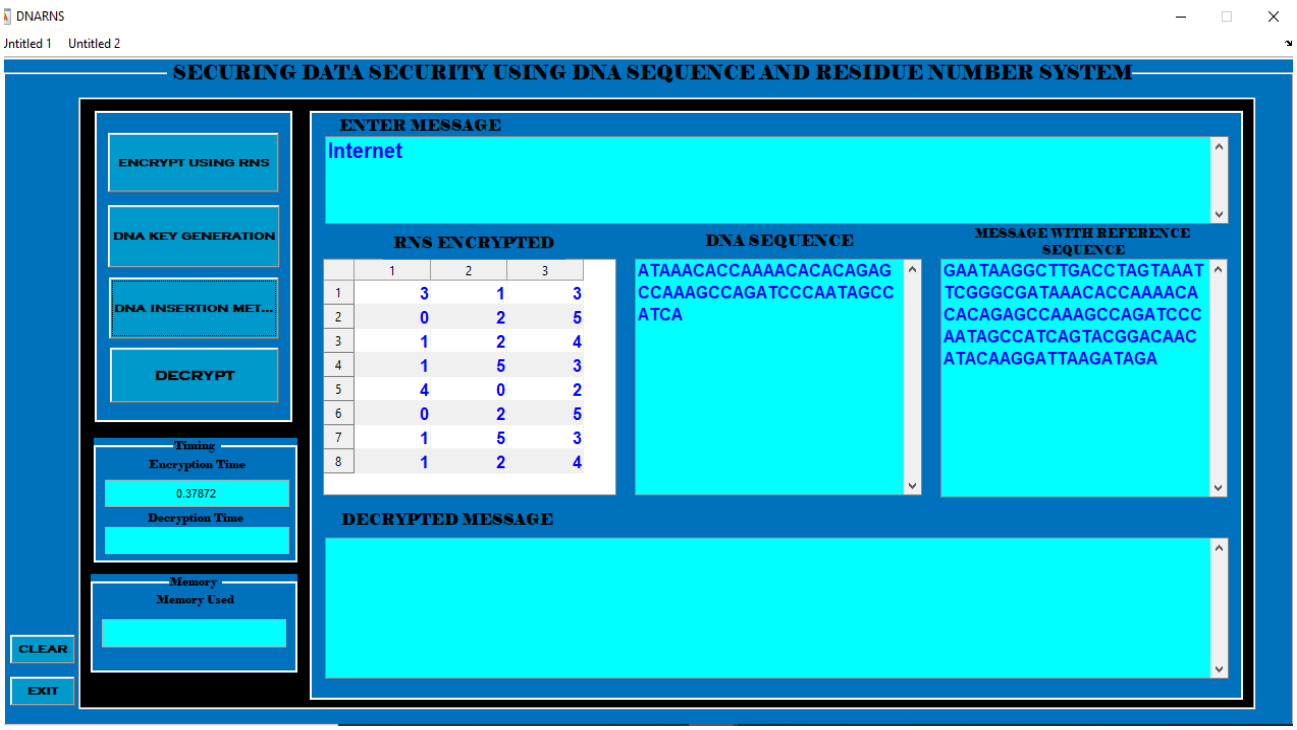

Figure 3.3: Encryption Process

3.2.5 Decryption process

For the decryption process, the DNA sequence is extracted from the cipher DNA sequence and then converted back to residues using the DNA base table. the residues generated will undergo backward conversion using Chinese Remainder Theorem (CRT).

This theorem is used in converting residue number system to decimal. Given the residue representation $\{r_1, r_2, \ldots, r_n\}$ of c, the CRT makes it possible to determine $|c|_m$, provided the gcd of any pair of moduli is 1. Such moduli are called pairwise relatively prime.

The CRT is given by

$$X = \left| \sum_{i=1}^{k} m_i \left| m_i^{-1} x_i \right|_{mi} \right|_m \qquad (3.2)$$

$M = \prod_{i=1}^{n} m_i$ and $M_i = \frac{M}{mi}$ and M_i^{-1} is the multiplicative inverse of M_i with respect to m_i.

The procedure for the decryption is as follows;

1. Extract the DNA sequence from the cipher DNA sequence.
2. Substitute each of the DNA sequence to residues base on DNA lookup table.
3. Using Chinese remainder theorem, perform a reverse conversion on the residues based on the moduli set {2n-1, 2n, 2n+1} to generate the ASCII value.
4. Convert the ASCII value back to plain text.

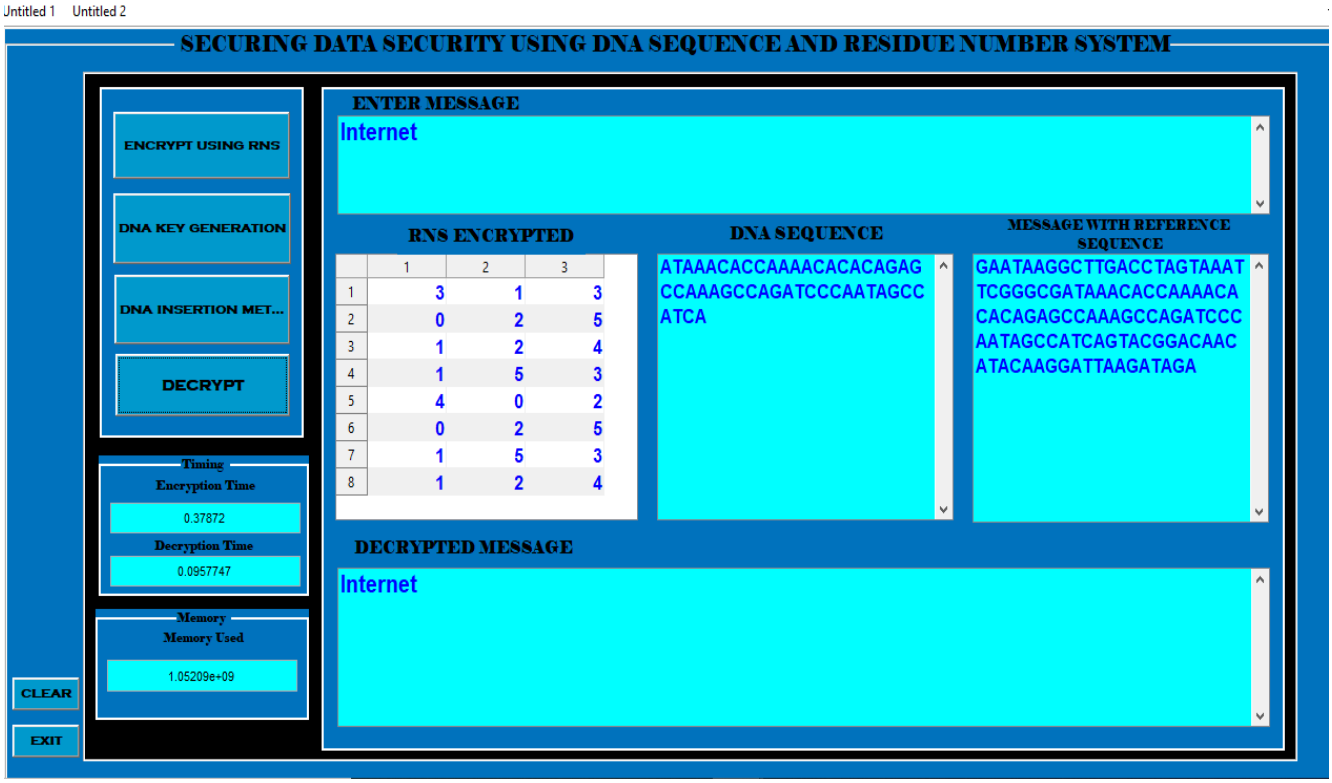

Figure 3.4: Decryption process

3.3 System Requirement

For better effectiveness and performance, the developed system requires a computer system with minimum intel core 2 processor running at 2.0 GHZ, 4 GB ram and 120 GB free hard disk space. Also, it requires operating system of minimum Window 7 with MATLAB 2015a installed on it.

3.4 Choice of Programming Language

The entire system will be developed using MATLAB. MATLAB is a high-level programming language and interactive environment for numerical computation, visualization, and programming. Using MATLAB, you can analyze data, develop algorithms, and create models and standalone applications. The language, tools, and in-built math functions enable you to explore multiple approaches and reach a solution faster than with spreadsheets or traditional programming languages, such as C/C++ or Java. MATLAB programming language is also

object-based programming language and it is integrated with MATLAB Compiler Runtime for the purpose of creating applications that will work outside of MATLAB integrated environment, thus making room for applications to run without installing MATLAB.

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 Text Conversion

The first stage of the entire process is the conversion of each character of the message to its equivalent ASCII form. The ASCII form comprises of the representation of all English alphabet a-z, A-Z, special character and numbers 0-9 in a numerical form, which is used in representing the content of the original message.

4.2 Encryption of Plain Text

Given the word *Internet* as the plain text, firstly, text conversion to ASCII form is carried out on the plain text.

Table 4.1: Text Conversion

73	**110**	**116**	**101**	**114**	**110**	**101**	**116**

Thereafter transposition of the numbers takes place, using forward conversion based on the chosen moduli set.

Given the moduli set {2n-1, 2n, 2n+1} the value of n was set to 3 and the computation of the moduli set generate 5, 6, 7 as our moduli. Such that $m_1 = 5$, $m_2 = 6$, $m_3 = 7$, which will be used to determine the RNS representation of each number in table 4.1.

For the first character where the numeric value is 73,

$$\left|X\right|m_1 = \left|73\right|_5 = 3$$

$$|X|_{m_2} = |73|_6 = 1$$

$$|X|_{m_3} = |73|_7 = 3$$

Therefore, the RNS representation of 73 is {3, 1, 3}

For the next character where the numeric value is 110,

$$|X|_{m_1} = |110|_5 = 0$$

$$|X|_{m_1} = |110|_6 = 2$$

$$|X|_{m_1} = |110|_7 = 5$$

Therefore, the RNS representation of 110 is {0, 2, 5}

For the next character where the numeric value is 116,

$$|X|_{m_1} = |116|_5 = 1$$

$$|X|_{m_1} = |116|_6 = 2$$

$$|X|_{m_1} = |116|_7 = 4$$

Therefore, the RNS representation of 116 is {1, 2, 4}

For the next character where the numeric value is 101,

$$|X|_{m_1} = |101|_5 = 1$$

$$\left|X\right|_{m_1} = \left|101\right|_6 = 5$$

$$\left|X\right|_{m_1} = \left|101\right|_7 = 3$$

Therefore, the RNS representation of 101 is {1, 5, 3}

For the next character where the numeric value is 114,

$$\left|X\right|_{m_1} = \left|114\right|_5 = 4$$

$$\left|X\right|_{m_1} = \left|114\right|_6 = 0$$

$$\left|X\right|_{m_1} = \left|114\right|_7 = 2$$

Therefore, the RNS representation of 114 is {4, 0, 2}

For the next character where the numeric value is 110,

$$\left|X\right|_{m_1} = \left|110\right|_5 = 0$$

$$\left|X\right|_{m_1} = \left|110\right|_6 = 2$$

$$\left|X\right|_{m_1} = \left|110\right|_7 = 5$$

Therefore, the RNS representation of 110 is {0, 2, 5}

For the next character where the numeric value is 101,

$$\left|X\right|_{m_1} = \left|101\right|_5 = 1$$

$$|X|_{m_1} = |101|_6 = 5$$

$$|X|_{m_1} = |101|_7 = 3$$

Therefore, the RNS representation of 101 is {1, 5, 3}

For the next character where the numeric value is 116,

$$|X|_{m_1} = |116|_5 = 1$$

$$|X|_{m_1} = |116|_6 = 2$$

$$|X|_{m_1} = |116|_7 = 4$$

Therefore, the RNS representation of 116 is {1, 2, 4}

After the completion of the forward conversion, the residue for each character is represented as a row matrix as shown in table 4.2.

Table 4.2: Residue representation of plain text

3	**1**	**3**
0	**2**	**5**
1	**2**	**4**
1	**5**	**3**
4	**0**	**2**
0	**2**	**5**
1	**5**	**3**
1	**2**	**4**

The content of Table 4.2 above is the cipher text which will be convert into binary form in four bits and later into a DNA sequence using Table 3.1 to generate Table 4.3 and Table 4.4 below.

Table 4.3: Residues in binary

0011	**0001**	**0011**
0000	**0010**	**0101**
0001	**0010**	**0100**
0001	**0101**	**0011**
0100	**0000**	**0010**
0000	**0010**	**0101**
0001	**0101**	**0011**
0001	**0010**	**0100**

Table 4.4: Binary to DNA sequence

AT	**AC**	**AT**
AA	**AG**	**CC**
AC	**AG**	**CA**
AC	**CC**	**AT**
CA	**AA**	**AG**
AA	**AG**	**CC**
AC	**CC**	**AT**
AC	**AG**	**CA**

The content of Table 4.4 is converted from a matrix to DNA sequence in a horizontal array;

ATAAACACCAAAACACACAGAGCCAAAGCCAGATCCCAATAGCCATCA

The DNA sequence generated from Table 4.4 is now hidden between two different reference DNA sequences that are randomly chosen online out of the 163 million available to generate a cipher DNA sequence.

Table 4.5: Selected Reference Sequence

SN	REFERENCE SEQUENCE
1	**GAATAAGGCTTGACCTAGTAAATTCGGGCG**
2	**GTACGGACAACATACAAGGATTAAGATAGA**

Table 4.6: Cipher DNA Sequence (cipher-text)

CIPHER DNA SEQUENCE
GAATAAGGCTTGACCTAGTAAATTCGGGCGATAAACACCAAAACACACAGAGCCAAAGCCAGATCCCAATAGCCATCAGTACGGACAACATACAAGGATTAAGATAGA

The content of Table 4.6 above is the cipher-text which is to be send to the second party.

4.3 Decryption of the Cipher-Text

For the message to be retrieved by the receiving party, they must be aware of the reference sequence used, the arrangement of the binary to DNA base table, the moduli set used, the arrangement of the moduli set and the value of n used to compute the moduli.

The real DNA sequence will be extracted from the cipher DNA sequence using Table 4.5 to identify the reference sequence used.

Then the real DNA sequence will be converted back into binary then later to residue using Table 3.1.

Thereafter, the reverse of the transposition which was used to convert the message to residue, the Chinese Remainder Theorem (CRT) is applied.

The CRT is given by

$$X = \left| \sum_{i=1}^{k} m_i \left| m_i^{-1} x_i \right|_{mi} \right|_m \qquad (4.1)$$

$M = \prod_{i=1}^{n} m_i$ and $M_i = \frac{M}{mi}$ and M_i^{-1} is the multiplicative inverse of M_i with respect to m_i.

Using the moduli $m_1 = 5$, $m_2 = 6$, $m_3 = 7$, the dynamic range is;

$$M = \prod_{i=1}^{N} m_i = 5 \text{ x } 6 \text{ x } 7 = 210$$

$$M_1 = \frac{M}{m1} = \frac{210}{5} = 42, \qquad M_2 = \frac{M}{m2} = \frac{210}{6} = 35, \qquad M_3 = \frac{M}{m3} = \frac{210}{7} = 30$$

$$M_1^{-1} = 3, \qquad M_2^{-1} = 5, \qquad M_3^{-1} = 4,$$

For the first row of the cipher-text with residue {3, 1, 3} the reverse conversion is;

$$|x|_{210} = |(3 \text{ x } 42 \text{ x } 3) + (1 \text{ x } 35 \text{ x } 5) + (3 \text{ x } 30 \text{ x } 4)|_{210}$$

$$= |378 + 175 + 360|_{210}$$

$$= |913|_{210} = 73$$

For the next row of the cipher-text with residue {0, 2, 5} the reverse conversion is;

$$|x|_{210} = |(0 \text{ x } 42 \text{ x } 3) + (2 \text{ x } 35 \text{ x } 5) + (5 \text{ x } 30 \text{ x } 4)|_{210}$$

$= |0 + 350 + 600|_{210}$

$= |950|_{210} = 110$

For the next row of the cipher-text with residue {1, 2, 4} the reverse conversion is;

$|x|_{210} = |(1 \times 42 \times 3) + (2 \times 35 \times 5) + (4 \times 30 \times 4)|_{210}$

$= |126 + 350 + 480|_{210}$

$= |956|_{210} = 116$

For the next row of the cipher-text with residue {1, 5, 3} the reverse conversion is;

$|x|_{210} = |(1 \times 42 \times 3) + (5 \times 35 \times 5) + (3 \times 30 \times 4)|_{210}$

$= |126 + 875 + 360|_{210}$

$= |1361|_{210} = 101$

For the next row of the cipher-text with residue {4, 0, 5} the reverse conversion is;

$|x|_{210} = |(4 \times 42 \times 3) + (0 \times 35 \times 5) + (5 \times 30 \times 4)|_{210}$

$= |504 + 0 + 240|_{210}$

$= |744|_{210} = 114$

For the next row of the cipher-text with residue {0, 2, 5} the reverse conversion is;

$|x|_{210} = |(0 \times 42 \times 3) + (2 \times 35 \times 5) + (5 \times 30 \times 4)|_{210}$

$= |0 + 350 + 600|_{210}$

$= |950|_{210} = 110$

For the next row of the cipher-text with residue {1, 5, 3} the reverse conversion is;

$|x|_{210} = |(1 \times 42 \times 3) + (5 \times 35 \times 5) + (3 \times 30 \times 4)|_{210}$

$= |126 + 875 + 360|_{210}$

$= |1361|_{210} = 101$

For the next row of the cipher-text with residue {1, 2, 4} the reverse conversion is;

$|x|_{210} = |(1 \times 42 \times 3) + (2 \times 35 \times 5) + (4 \times 30 \times 4)|_{210}$

$= |126 + 350 + 480|_{210}$

$= |956|_{210} = 116$

The computation carried out on each of the residue produces the initial ASCII value as shown in Table 4.1 and the ASCII values are converted back to characters to produce the plain text which is the original message sent.

4.4 Comparison with existing approaches

Several words with different character length was encrypted using the previous schemes and the proposed scheme, both the encryption and decryption time was analyzed and evaluated. After the evaluation of the schemes, as shown in Table 4.7 and Table 4.8, the proposed scheme was compared with the existing scheme to check for success attained.

Table 4.7: Comparison in term of Encryption Time

CHARACTER (size)	P. Malathi (2017) (ms)	S.S. Roy (2017) (ms)	PROPOSED SCHEME (ms)
Design (6)	**2.1321**	**3.1655**	**0.2786**
Message (7)	**2.9102**	**4.4316**	**0.3062**
Internet (8)	**3.2140**	**4.5224**	**0.3436**
Database (8)	**3.1452**	**4.9636**	**0.3049**
Cryptography (12)	**4.9801**	**6.3321**	**0.4020**
Generational (12)	**4.8431**	**6.3431**	**0.3077**

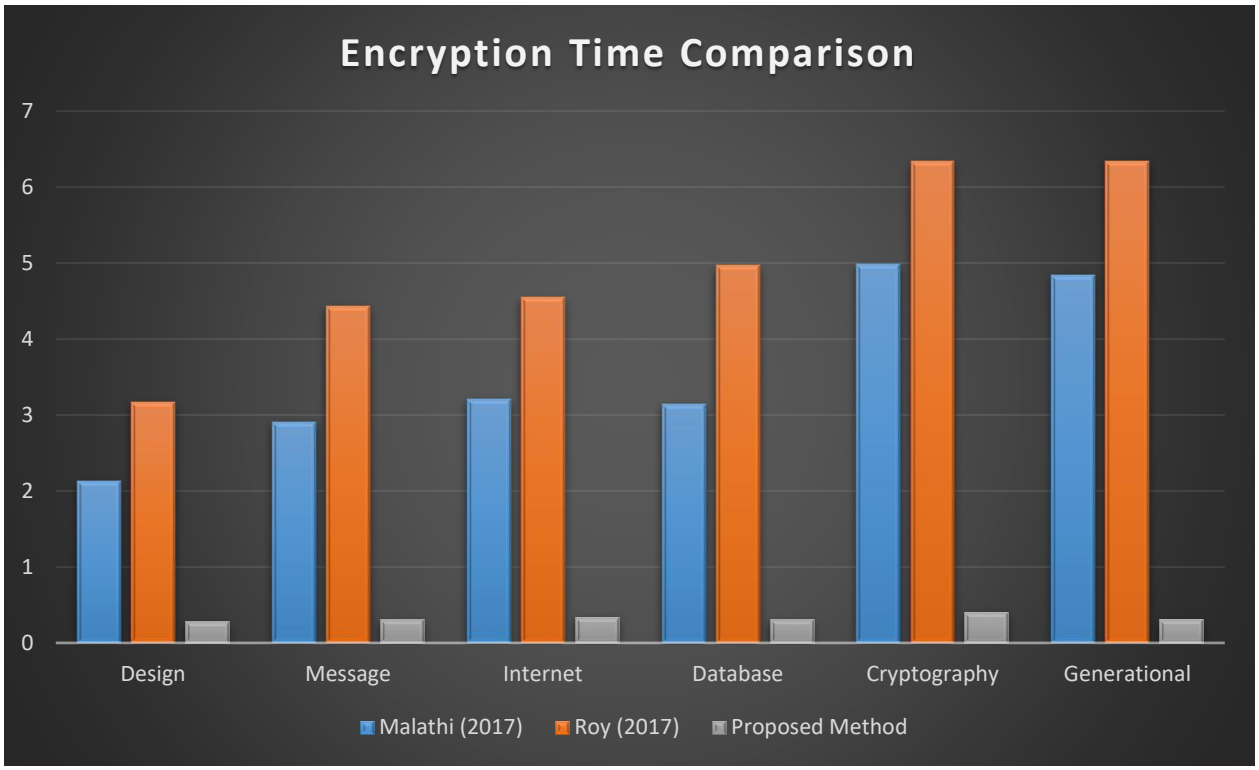

Figure 4.1: Comparison in term of Encryption Time

Table 4.8: Comparison in term of Decryption Time

CHARACTER (size)	P. Malathi (2017) (ms)	S.S. Roy (2017) (ms)	PROPOSED SCHEME (ms)
Design (6)	0.4152	1.1094	0.0851
Message (7)	0.9136	1.6641	0.1267
Internet (8)	1.4132	1.8351	0.0882
Database (8)	1.1246	2.0093	0.0912
Cryptography (12)	2.0148	3.3241	0.1387
Generational (12)	1.9241	3.4132	0.0916

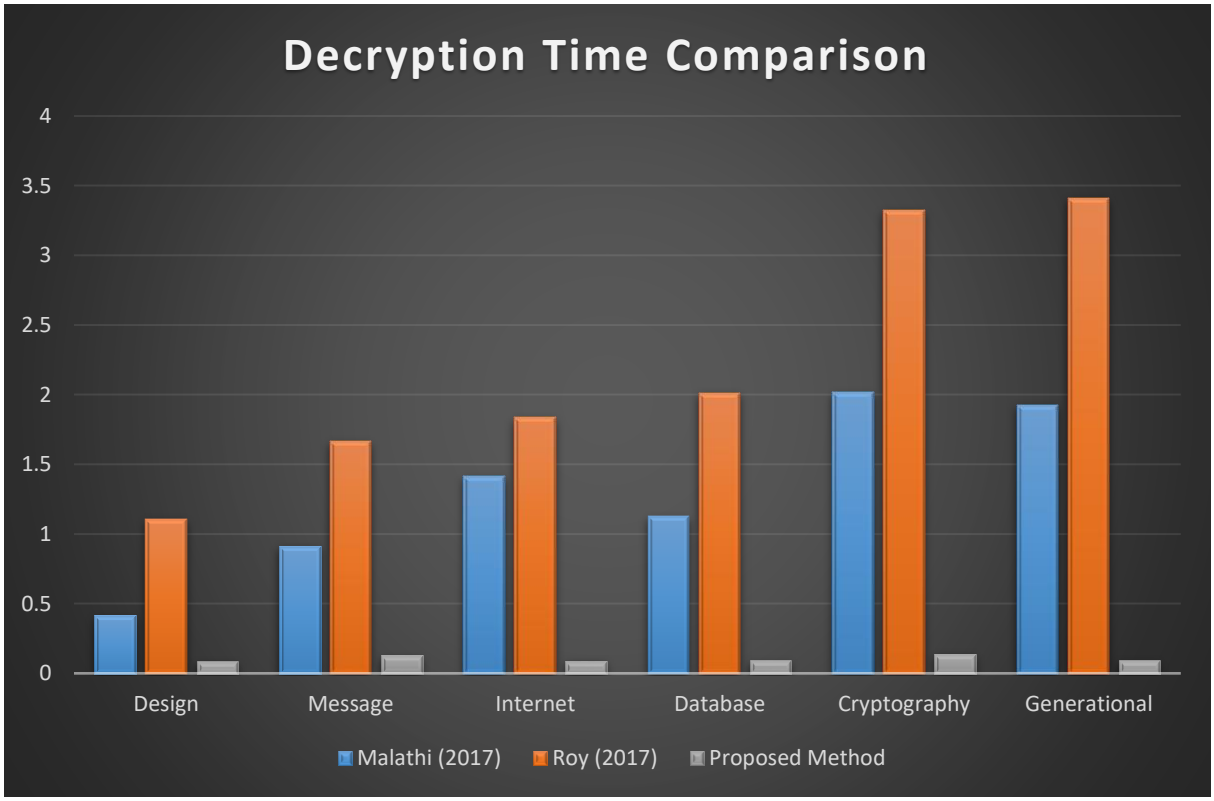

Figure 4.2: Comparison in term of Decryption Time

The results of the experiment carried out shows that the proposed system maintain a relatively low computational time regardless of the character length of the text being encrypted whereas previous schemes computational time keeps increasing base on the size of character being encrypted.

CHAPTER FIVE

SUMMARY, CONCLUSION AND RECOMMENDATIONS

5.1 Summary

This research proposed the use of Residue Number System (RNS) forward conversion and DNA cryptography (Insertion method) to encrypt characters and decrypt the characters based on the DNA lookup table and RNS reverse conversion (Chinese Remainder Theorem). The characters are first converted to their ASCII equivalent before being encrypted using RNS with respect to the moduli set {2n-1, 2n, 2n+1}, then the residue is converted into a DNA sequence and the DNA sequence is embedded between two reference sequence to form a cipher DNA sequence which will be transmitted to the receiver. The receiver on the other end must have known the moduli set being used, the value of n, the arrangement of the DNA lookup table and the reference sequence to be able to decrypt the message accurately.

5.2 Conclusion

The proposed scheme has provided a better means of transmitting message securely and faster over the network while the message maintains an extremely high level of secrecy, proving to outperform it predecessors in term of computational time to produce results and still maintaining the secrecy attached to the message. The result of the cipher-text produced in the proposed scheme is very complex and to crack it, there are many probabilities that should be tried, such as: finding the sequence of the algorithms used, determining the moduli set, determining the value of n for the moduli set, determining the arrangement of the DNA lookup table and also determining the first and the second reference sequence chosen randomly from the 163 million available online.

5.3 Major contribution to knowledge

The major contributions of this thesis are listed below. The scheme produced a system that;

i. Provides a better security with the improve in the computational time.
ii. Makes the cipher-text maintain a decent representation, making it hard for an attacker to suspect such data has been encrypted.

5.4 Recommendation and Future Work

As indicated in Chapter 4, the proposed design outperforms the current designs in terms of computational time but still does not maintain DNA identity. Thus, a new design of DNA cryptography and residue number system that preserve the DNA identity can be used to enhance the security of the system.

For the purpose of further enhancement and providing a more efficient way of sending message securely using this approach, the following are expected future works;

i. An increase in the length of the moduli set, to make the encrypted text harder to crack by an intruder. This will increase the number of possible combination an intruder will have to make in other to decrypt the message.
ii. Use of Mixed Radix conversion with DNA sequence to evaluate the efficiency of this scheme in term of computational speed in the reverse process.
iii. Combination of Residue number system with other DNA sequence methods like complimentary rule and substitution rule.

REFERENCES

Abd El-Latif E.I. & Moussa M.I. (2019). Information hiding using artificial DNA sequences based on Gaussian kernel function. *Journal of Information and Optimization Sciences* ISSN: 0252-2667. 2169-0103.

Al‑Harbi O.A., Alahmadi W.E. & Aljahdali A.O. (2020). Security analysis of DNA based steganography techniques. *Springer Nature journal, SN Applied Sciences* 2:172.

Alhassan S. & Gbolagade K.A. (2013). Enhancement of the Security of a Digital Image using the Moduli Set 2n-1, 2n, 2n+1, *International Journal of Advance Research in Computer Engineering & Technology*, 2(7).

Aremu, I.A. & Gbolagade K.A. (2017). An overview of Residue Number System. *International Journal of Advanced Research in Computer Engineering & Technology,* 6(10), 1618 1623.

Badoni V.D. & Jain A. (2014). Chip Implementation of Text Encryption and Decryption Algorithms. *IOSR Journal of Computer Engineering* (IOSR-JCE). e-ISSN: 2278-0661, p ISSN: 2278-8727, Volume 16, Issue 3, Ver. VI (May-Jun. 2014), PP 56-61.

Bhateja, A. & Mittal K. (2015). *DNA Steganography: Literature Survey on its Viability as a Novel Cryptosystem.* Journal of Computer Science and Engineering. 2(1): p. 8-14.

Bae H., et.al., (2019). DNA Steganalysis Using Deep Recurrent Neural Networks. *Pacific Symposium on Biocomputing.*

Cheng G., Chin-Chen C. & Wang Z. (2012). A New Data Hiding Scheme Based On DNA Sequence. *International Journal of Innovative Computing, Information and Control ICIC International* Volume 8, Number 1(A). ISSN 1349-4198, 139-149.

Debnath B. & Samir K.B. (2013). Hiding Secret Data in DNA Sequence, *International Journal of Scientific &Engineering Research* Volume 4, Issue 2, February-2013 ISSN 2229 5518.

Deepak B. (2017*). Steganographic Algorithms and Application of DNA*. M.Sc. Dissertation, Kaunas University of Technology.

Dhanya M.S., Asha J. & Dincy E.B. (2018). DNA Based Cryptography and Steganography in Cloud Computing using Socket Programming. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)* ISSN: 2321-9653.

Dilovan A.Z., Habibollah H., Subhi R.M.Z. (2017). Security issues in DNA based on data hiding: a review. *International Journal of Applied Engineering Research.* ISSN 0973 4562, Volume 12, Number 24. pp. 15363-15377.

Dixit P., Dey A., Agarwal A. & Pal T. (2019). Genetic algorithm for Robust Total coloring of a Fuzzy graph. *IEEE Congress on Evolutionary Computation* (CEC), 1806-1813.

Gbolagade K.A. & Cotofana S.D. (2008). A residue to binary converter for the {2n+2;2n+1;2n} moduli set. *Proceedings of 42nd Asilomar Conference on Signals, Systems, and Computers*, pp. 1785-1789.

Gupta K. & Singh S. (2013). DNA Based Cryptographic Techniques.*International Journal of Advanced Research in Computer Science and Software Engineering* Volume 3, Issue 3, ISSN: 2277 128X.

Gupta R. & Jain A. (2014). A New Image Encryption Algorithm based on DNA Approach. *International Journal of Computer Applications* Volume 85, No 18, 0975 – 8887.

Hamed G., Mohammed A. M., Safaa A., Mohamed T., (2015). Hybrid technique for steganography based on DNA with n-bits binary coding rule. *7th International conference of soft computing and pattern recognition (SoCPaR).* IEEE.

Kessler G.C. (2020). *An Overview of Cryptography. https://www.garykessler.net/library/crypto.html*

Krishnan R.B., Thandra P. K. & Baba M. S. (2017). An overview of text steganography. *2017 Fourth International Conference on Signal Processing, Communication and Networking* (ICSCN), pp. 1-6, doi: 10.1109/ICSCN.8085643.

Meera, M. & Malathi, P. (2015). An improved embedding scheme in compressed domain image steganography. *International Journal of Applied Engineering Research* ;10 (55):1933 -1937.

Malathi P. & Gireeshkumar T. (2016). Relating the Embedding Efficiency of LSB SteganographyTechniques in Spatial and Transform Domains. *Procedia Computer Science*; 93:878 85.

Malathi P., Manoaj M., Manoj R., Vaikunth Raghavan & Vinodhini R.E. (2017). Highly Improved DNA Based Steganography. *7th International Conference on Advances in Computing & Communications*, ICACC, *Procedia Computer Science* 115, 651–659.

Menaka K. (2014). Message Encryption Using DNA Sequence. *Institute of Electrical and Electronics Engineers*, 978-1-4799-2977-4.

Mohammad R. A., Azizah A., & Shahidan M.A. (2011). Data Hiding Method Based on DNA Basic Characteristics, *International Conference on Digital Enterprise and Information Systems*, 53–62.

Olga T. (2013). Contributions to DNA Cryptography Applications to Text and Image Secure Transmission. Joint PhD Thesis, Technical University of Cluj – Napoca & University of Nice Sophia Antipolis.

Omer A. & Farooq M.I. (2015). DNA Cryptography Algorithms and Applications. HITECH University.

Premkumar A.B. (1992). An RNS to binary converter in {2n+1;2n;2n-1} moduli set. *IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing*, Vol. 39, No.7, pp. 480 482.

Pushpa B.R. (2017). A new technique for data encryption using DNA sequence. *International Conference on Intelligent Computing and Control* (I2C2), DOI: 10.1109/I2C2.2017.8321834.

Roy S.S., *et.al.* (2017). A Novel Encryption Model for Text Messages using Delayed Chaotic Neural Network and DNA Cryptography. *20th International Conference of Computer and Information Technology* (ICCIT).

Selvaraj D. (2017). Development of a secure communication system based on steganography for mobile devices. p 3.

Sharma A. (2016). Security and information hiding based on DNA Steganography. *A Monthly J Comput Sci Inf Technol.* 5(3):827–832.

Shiu H. J., *et.al.* (2010). Data hiding methods based upon DNA sequences, *Information of Science*, vol.180, no.11, 2196-2208.

Siewobr H. & Gbolagade K.A. (2012). An area efficient RNS-to-binary converter for the {2n, 22n+1 −1, 2n −1} moduli set. *ICASTech*, SN - 978-1-4673-4787-7.

Siper A., Farley R. & Lombardo C. (2005). The rise of steganography *Proceedings of Student/Faculty Research Day CSIS,* Pace University.

Sohal, Sharma (2018). BDNA-A DNA inspired symmetric key cryptographic technique to secure cloud computing. *Journal of King Saud University - Computer and Information Sciences.*

Sokół B. & Yarmolik V.N. (2005) Cryptography and steganography: teaching experience. Enhanced methods in computer security, biometric and artificial intelligence systems. *Springer, Boston*, pp 83–92.

Sureshraj, D. & Bhaskaran V.M. (2012). *Automatic DNA sequence generation for secured cost effective multi-cloud storage.*

Sushma R.B., *et.al.* (2019). DNA based Steganography Using 2-3-3 Technique. *International conference on Data Science and Communication.*

Taha M.S., *et.al.* (2019). Combination of Steganography and Cryptography: A short Survey. *2nd International Conference on Sustainable Engineering Techniques (ICSET)* doi:10.1088/1757-899X/518/5/052003.

Taur J., Lin H., Lee H. & Tao C. (2012). Data Hiding in DNA Sequences based on Table Lookup Substitution. *International Journal of Innovative Computing, Information and Control*, ISSN: 1349–4198, vol. 8, Issue No. 10, pp. 6585–6598.

Tiwari H.D. & Kim J.H. (2018). Novel Method for DNA- Based Elliptic Curve Cryptography for IoT Devices. *ETRI Journal*, Volume 40, Number 3.

APPENDIX

SOURCE CODE

```
function varargout = DNARNS(varargin)
% DNARNS MATLAB code for DNARNS.fig
%      DNARNS, by itself, creates a new DNARNS or raises the existing
%      singleton*.
%
%      H = DNARNS returns the handle to a new DNARNS or the handle to
%      the existing singleton*.
%
%      DNARNS('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in DNARNS.M with the given input arguments.
%
%      DNARNS('Property','Value',...) creates a new DNARNS or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before DNARNS_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to DNARNS_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help DNARNS

% Last Modified by GUIDE v2.5 19-Mar-2020 20:54:33

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @DNARNS_OpeningFcn, ...
                   'gui_OutputFcn',  @DNARNS_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT



% --- Executes just before DNARNS is made visible.
function DNARNS_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to DNARNS (see VARARGIN)
```

```
% Choose default command line output for DNARNS
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes DNARNS wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = DNARNS_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;



function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as a double
sliderMin = get(handles.slider, 'Min');
sliderMax = get(handles.slider, 'Max');
increment = 1/(sliderMax-sliderMin);
sliderStep = set(handles.slider, 'Step', [increment, 2*increment]);
sliderValue = round(get(handles.slider, 'Value'));

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function RNS_Callback(hObject, eventdata, handles)
% hObject    handle to RNS (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of RNS as text
%        str2double(get(hObject,'String')) returns contents of RNS as a double
```

```
% --- Executes during object creation, after setting all properties.
function RNS_CreateFcn(hObject, eventdata, handles)
% hObject    handle to RNS (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as a double
global slider
sliderMin = get(handles.slider, 'Min');
sliderMax = get(handles.slider, 'Max');
increment = 1/(sliderMax-sliderMin);
sliderStep = set(handles.slider, 'Step', [increment, 2*increment]);
sliderValue = round(get(handles.slider, 'Value'));

% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%        str2double(get(hObject,'String')) returns contents of edit6 as a double
sliderMin = get(handles.slider, 'Min');
sliderMax = get(handles.slider, 'Max');
increment = 1/(sliderMax-sliderMin);
sliderStep = set(handles.slider, 'Step', [increment, 2*increment]);
sliderValue = round(get(handles.slider, 'Value'));
```

```
% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global DNAR t1 t2 t3 memor3 memor2 memor1
tic
am=memory;
      s1 = 'GAATAAGGCTTGACCTAGTAAATTCGGGCG';
      s2 = [DNAR];
      s3 = 'GTACGGACAACATACAAGGATTAAGATAGA';
       s = strcat(s1,s2,s3);

            set(handles.edit7,'string', s);

        t3=toc;
        memor3=am.MemUsedMATLAB;
          time = t1 + t2 + t3;
          memor= memor1;
            set(handles.edit8,'string', time);



% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global ff t1 memor2
tic
am=memory;
ui = double(get(handles.edit5,'string'));



      P =(ui);
      f1 = mod(P,5);
   f2 = mod(P,6);
   f3 = mod(P,7);
   ff = [f1; f2; f3];
   ff = ff';

      set(handles.uitable3,'Data',ff);

     t1=toc;
     memor2=am.MemUsedMATLAB;
```