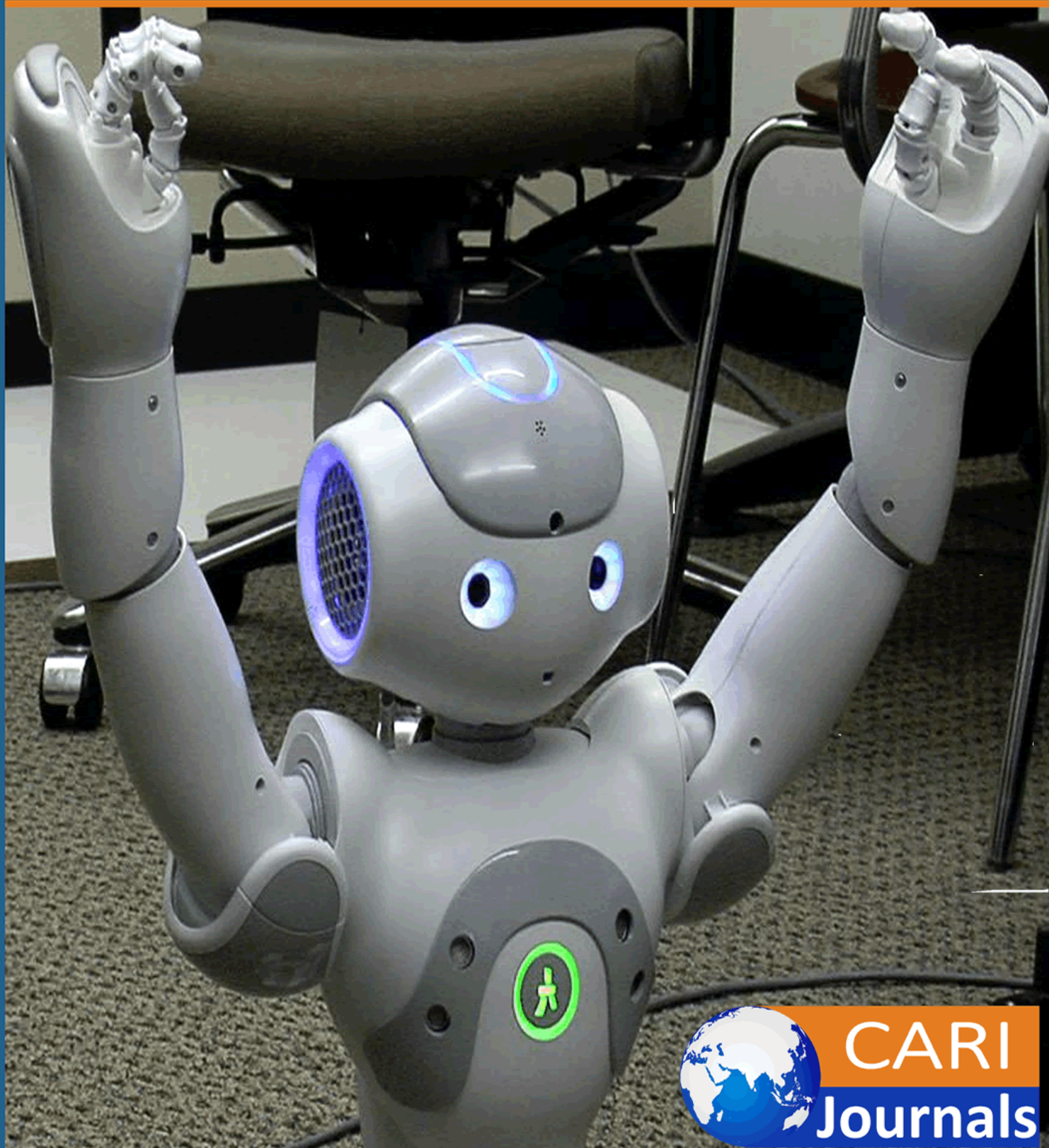
International Journal of

Computing and Engineering

(IJCE)



CARI Journals

International Journal of Computing and Engineering
ISSN 2788-6344 (Online)
Vol. 2, Issue No. 1, pp 1- 13, 2021
www.carijournals.org

Symmetric Cryptography for Confidential Communications: *Implemented by Enhancing the Caesar Cipher*

[1*]Munura Maihankali
Chevening Scholar, MSc Data Science: School of Computing
Robert Gordon University Aberdeen, United Kingdom.
Corresponding author email: maihankspinas@gmail.com
[2*]Esther Chinwe Eze
MSc Cyber Security: School of Computing
Robert Gordon University Aberdeen, United Kingdom.
Corresponding author email: esther4eze@gmail.com

Abstract

***Purpose*:** This paper identifies and rectifies the key lapses associated with the existing Caesar cryptographic algorithm and further implements the proposed solution, analyses and demonstrates the validity of the improvements effected. The achieved model was implemented as a software solution and duly demonstrated to be a good choice of encryption algorithm in modern-day cryptography.

***Methodology*:** The methodology adopted in this study is typically the waterfall model, further incorporating the Object-Oriented Design and Analysis to facilitate the deployment of a cross-platform friendly software implementation of the model solution developed. Computational analysis of the blueprint technique employed to rectify the lapses associated with the existing Caesar cipher e.g., Diffie Helman Technique is elucidated with a more comprehensive description of the architectural design and functionality of the model accessible via: https://github.com/Maihanks/EnhancedCaeserCIpher . A blend of secondary and primary data was used for experimentation and random text data was used to test the modelled solution primarily to validate its feasibility in real-time.

Results: The result obtained is a software implementation of the Enhanced Caesar Cipher developed and demonstrated in real-time to be feasible, functional, and cross-platform friendly accessible via: https://github.com/Maihanks/EnhancedCaeserCIpher. It was experimented and demonstrated to have rectified the key lapses associated with the conventional Caesar Cipher analysed and discussed in this paper.

***Unique contribution to theory, policy, and practice*:** The result arrived at by the end of the research include: (a) an improved and better version of the Caesar cipher dully implemented, rectifying the conventional Caesar cipher's lapses as described in detail in this paper (b) An automated version of the Caesar Cipher deployed as a software solution (c) A higher complexity for the ameliorated Caesar Cipher with a complexity of $O(n^2)$ was achieved which is far better than the conventional Caesar cipher's complexity of O(n).

Keywords: *Algorithm, Caesar Cipher, Complexity, Cryptography, Symmetric.*

INTRODUCTION

The need for privacy and confidentiality in the transmission, storage, and usage of information is most imminent and necessary now than ever before. Although modern technologies, tools, software, and gadgetry have emerged to facilitate convenience and ease human interactions and working situations; this convenience often comes at the expense of information security, and loss of privacy- indeed a perturbing problem. The internet has grown to the point where it plays an important and indispensable role in our daily lives, and with this merit comes the issue of data security primarily because the IT and its tools are been abused, and exploited for crimes and malicious reasons. For anyone, including organizations connected to the web, the issue of data security is now of enormous concern. The primary aim of data security is achieving confidentiality, integrity, authentication, and non-repudiation of data and cryptography plays a crucial role in achieving this. The major concern is ensuring that all forms of data on the web or local systems are protected and transmitted confidentially among communicators, an in this context cryptography is indispensable. Presently cryptography is merged with IT and various business sectors to ensure sensitive personal, financial, medical, and other forms of data are protected and kept confidential. Today, digital privacy is a highly prized commodity for individuals, firms, and governments alike. According to Piotr & Wiesław (2010), "The influence of the computer technologies on modern measurement systems (MS) is steadily increasing. Numerous contemporary solutions not only allow connecting traditional devices to computers but also require the computer network to transmit the acquired data and processing units to perform calculations on them. As the distributed measurement systems (DMS) are a common implementation (especially in industrial applications), the security of data transferred between the nodes of the system becomes a primary concern". Although numerous solution models to boost securities in the IT have been set in place; in this study, the focus is on securing and maintaining privacy in communications that involve the transmission of textual digital data packets – particularly using a symmetric cryptographic scheme (The Caesar Cipher). Key lapses associated with the conventional Caesar cipher were identified and rectified, and the achieved model was implemented as software using java programming.

1.1 The Concept of Cryptography

Point (2018), defines Cryptography as the art and science of using mathematical algorithms to provide security services with a focus on securing digital information. This is just a grand view of cryptography as in a more specific sense, Jangir (2014) explains Cryptography as the process of encrypting sensitive information to hide its content. This process involves using several secure algorithms to encrypt and decrypt data; the encryption and decryption process is based on the use of secret keys which is an important parameter for an encryption algorithm. Encryption cannot be reversed if the secret key is not known hence, the importance of the secret key. Cryptography uses various algorithms to offer different levels of data security that preserves the confidentiality, integrity, and availability (CIA) of any data.

In relation to cryptography, Monika (2012) highlights the basic cryptographic terminologies to encompass: (i) *Plain text:* the original content or data (ii) *Ciphertext*: the coded content or data

(iii)*Encryption Algorithm:* the process that involves converting plain text to ciphertext

(iv)*Decryption Algorithm*: the process that involves restoring plain text from a ciphertext

(v)*Key*: a key could either be an alphanumeric text or numeric or a set of symbols that is applied on plain text to encrypt it and on a ciphertext to decrypt it.

Munir (2005), affirms that Cryptography is crucial in the world today as it typically ensures that all confidential data like communications over telephone lines including faxes and e-mail messages, financial transactions, medical histories, e-banking, and other types of sensitive pieces of information are governed by privacy, a secure communication medium, and confidentiality. The application of cryptography provides a secure communication medium needed to transfer sensitive information reliably to protect your information from attackers, intruders, or hackers. When encryption is applied to information or message, it becomes useless to any receiver apart from the intended receiver because the information will appear in an encrypted form rather than plain text. Cryptography is one of the tools, that offers a sense of privacy and security.

Cryptography has gone through immense evolution since the time it was adopted by the Egyptians till today. Cryptography was originally adopted by Egyptians to beautify and decorate their tombs to draw attention and interest. Over the years, cryptography has established its fundamental role in ensuring secrecy, privacy, and confidentiality. Nowadays, it is incomprehensible for private or secret information to be decoded. With the latest thing of moving administrations to the web, we now have delicate information transmitted and put away all around the internet in form of data packets. They are susceptible to capture attempts and consequently completely dependent on cryptography to maintain confidentiality. Furthermore, the cryptographic framework utilized has to be solid and impervious to cryptanalysis and render a form of brute force attack that is illogical as numerous people might attempt each means to uncover the protected secret (Zulkifli, 2007). The advancement of cryptography has demonstrated that its improvement should closely follow the pace of technology. Right from ancient times till now, Cryptography has figured out how to adjust to keep being the method for ensuring the privacy of sensitive data.

Symmetric and Asymmetric Key Cryptography

According to Atish (2015), Symmetric key cryptography applies a single key for encryption and decryption which translates that a single key is shared between the sender and receiver. The sender uses the shared key and encryption algorithm to encrypt the message. The receiver uses the shared key and decryption algorithm to decrypt the message. Whereas in Asymmetric key cryptography, each user is assigned a pair of keys, a public key, and a private key. The public key is announced to all members while the private key is kept secret by the user. The sender uses the public key which was announced by the receiver to encrypt the message. The receiver uses his private key to decrypt the message.

Symmetric Cryptography

According to Rountree (2011), Symmetric encryption applies a single secret key for encryption of plaintext and decryption of ciphertext; this means that a single key is shared by two or more communicating parties (sender and recipient of a message). In its simplest form, symmetric encryption could be represented like this:

Encryption (plain text, key) = cipher text

Decryption (cipher text, key) = plain text

Rountree (2011), further explains that there are two types of symmetric encryption – stream and block encryption. Stream encryption operates by encrypting one bit or bytes at a time. In this cipher, plaintext (data stream) is usually combined with a pseudorandom digit and is then

encrypted using a key. Since bits are encrypted one at a time it means each bit is dependent on the current state of the encryption. Stream encryption uses an exclusive -or (XOR) for operation and the implementation of a stream cipher is quite fast and does not require many resources. Whereas in block encryption, several bits usually made up of 64bits or more are taken and encrypted as a block. The majority of encryption algorithms that are used currently are block encryption. In comparison, block ciphers are usually slower but more effective than stream cipher.

Data Encryption Standard (DES)

DES was the first encryption standard to be designed in 1973. Later on, in 1976 it was recommended by NIST (National Institute of Standard and Technology) as the most effective method for encryption of data and was a widely used standard all over the world (Zhang, 2009). DES is a block cipher that usually encrypts blocks of 64bits at a time and uses a key of 56bits (which is quite short). DES operates in CBC (cipher block chaining), ECB (electronic code block), CFB (cipher feedback), and OFB (output feedback) modes. DES has about 16 rounds meaning that 16 processing steps are applied on the plaintext to produce a ciphertext. Initially, 64 bits of data is passed through the initial phase and 16 rounds of processing take place and the final step of permutation is carried out on the plain text which outputs a 64-bit ciphertext. DES has only a 2^56 possible combination which is quite easy to crack. That makes DES not secure (Diaa, 2010).

According to Mansoor (2013), the strengths and weaknesses of the DES include the following: *Strengths*: (i) DES is hard to crack because of the number of rounds that are used to encrypt any message (ii) Hardware implementation of DES is faster than in software. *Weaknesses:* (i) it has a relatively small key length which makes it easy to crack (ii) It is relatively slow when it is run on software because it was not designed to run on software (iii) DES is not flexible meaning that it doesn't allow any form of modification (iv) Prone to brute force attacks.

Triple DES (TDES or 3DES)

TDES was designed as a replacement to replace the DES algorithm as a result of advances in key searching (Javed, 2005). It uses successive three rounds of DES encryption (Encrypt, Decrypt, Encrypt, or EDE) and has a key length of 168bits i.e 56*3. TDES uses three different keys (EDE) for the encryption algorithm to generate a ciphertext on plain text message t.

$$C(t) = Ek1\ (Dk2\ (Ek3\ (t)))$$

where C(t) is the ciphertext of plaintext message t, Ek1 is the encryption method using key k1, Dk2 is the decryption method using key k2 and Ek3 is the encryption method using key k3. Another option is to use two different keys for the encryption algorithm. This reduces the memory requirement of keys in TDES. These three keys require 2^168 possible combinations to be tried out for brute force attack which is practically not possible. This provides TDES as the strongest encryption algorithm which gives its application in the banking industry. The disadvantage of this algorithm is that it is too time-consuming (Verma, 2011). Although the Tripple DES is better off the DES, Verma, (2011) further identifies its key strengths and weaknesses to include: *Strengths:* (i) It is safer than DES (ii) It uses a longer key length, hence the reason it is safer than DES. *Weaknesses:* (i) It is slow and complex to implement compared to AES (ii) It is prone to related-key attacks (ii) It is also prone to certain variation of meet-in-the-middle attack (MITM).

Advance Encryption Standard (AES)

The AES was recommended by the US National Institute of Standards and Technology (NIST) to replace DES in 1998. AES is a variable bit block cipher and uses a variable key length of 128, 192, and 256 bits (the key length does not have a theoretical maximum). If both the block length and key length are 128 bits, AES will perform 9 processing rounds. If the block and key are 192 bits, AES performs 11 processing rounds. If the block and key are of length 256 bits then it performs 13 processing rounds (Sharma, 2010). Each processing round involves four steps as illustrated by Aamer (2005) which include: (i) Substitute bytes – Uses an S-box to perform a byte by byte substitution of the block (ii) Shift rows – A simple permutation(iii) Mix column – A substitution method where data in each column from the shift row step is multiplied by the algorithm's matrix and (iv) Add round key – The key for the processing round is XORed with the data.

Sharma (2010) further points out that AES encryption is quite fast and flexible; it can be implemented on various platforms especially in small devices. According to Mansoor (2013), its key strengths and weaknesses encompass- Strengths: (i) Implementing AES is not complex, it is also highly effective when compared to others(ii) AES is very secure (it cannot be hacked), fast and flexible to use (iii) It is more robust because it supports both hardware and software (iv) Less prone to cryptanalysis. Weaknesses: (i) Although it supports both hardware and software, implementing it on software is hard (ii) When AES is in counter mode, implementing it in software is complex (iii) Blocks in AES are encrypted in the same way.

Blowfish

The Blowfish algorithm emanated in 1993 by Bruce Schneier (Zhang, 2009). Blowfish is a 64-bit block cipher with a variable-length key from 32 bit (4 bytes) to 448 bits (56 bytes). The key strength of this algorithm is that it is highly secure and has not been cracked yet. It is suitable and efficient for hardware implementation. Blowfish algorithm has two parts- Key expansion and Data Encryption. The key expansion step converts a 448-bit key into 4168 bytes. A P array of size 18 and four S boxes whose size is 256 each of which is initialized to hexadecimal digits of π. XOR each entry in P array and S boxes with 32 bits of the key (Desoky, 2008).

According to Mousa (2015), the Blowfish algorithm typically consists of a total of 16 rounds of data encryption; in each round, a 32-bit subkey is XORed with leftmost 32 bits of plaintext and the result is then passed to the F function of Blowfish. This result becomes rightmost 32 bits for the next round and the output of the F function is XORed with the original rightmost 32 bits of plaintext becomes leftmost 32 bits for the next round and so on.

Mansoor (2013) further points that the key strengths and weaknesses of the Blowfish algorithm include- *Strengths:* (i) Blowfish is highly rated in terms of security compare to other algorithms because of its added functionality and structure (ii) It is also invulnerable to related-key attacks because it uses many independent round keys (iii) Implement is simple. *Weaknesses:* (i) It is slow but faster than DES (ii) Blowfish has some classes of a large number of weak keys, this makes it questionable because of the weak keys it has. Weak keys make encryption vulnerable and easy to crack.

The Caesar Cipher

The Caesar cipher is one of the oldest and simplest examples of a substitution cipher that evolves encrypted text (SINKOV, 1996). It is named after Julius Caesar, who, according to

Suetonius, used shift cipher with a constant left shift of 3 to communicate with his Army during the war to encrypt important military messages. Hence the name shift cipher, Caesar's cipher, or Caesar shift (Senthil, 2013). Arguably, Julius Caesar is believed to have been one of the first people to use encryption to transmit information securely. Caesar informed his Generals of his standard algorithm that shifting each letter three places down the alphabet in the message would serve as a means of sending secure messages (Savarese, 2010).

How the Caesar Cipher Works

According to Inan (2019), the Caesar Shift Cipher operates by shifting the letters of a message along the alphabetical sequence by a number of times referred to as key; e.g. key =3 and when the recipient of the message receives it, they would then shift the letters back by the same number (3) to decrypt and reveal the original message. Encryption and decryption can be represented in modular arithmetics. Mathematically, if a letter (x) is to be encrypted, it is expressed as:

$$E_n(x) = (x + n) \bmod 26.$$

Decryption is performed similarly:

$$D_n(x) = (x - n) \bmod 26$$

Each letter in the original message is a format in which the alphabet is replaced by a letter corresponding to a particular letter with a shift key. Let's assume the message is 'SECRET' and the key count is 3. Each letter in the message is shifting 3 letters forward, we find the encrypted message as 'VHFUHW'. To solve this cipher, we get 3 letters back from that letter and we obtained a 'SECRET' message (Inan, 2019).

Table 1: *The Caesar Cipher Algorithm using key 3. Source: (Inan, 2019)*

Key: 3

Alphabet	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Cipher Alphabet	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
Message	S	E	C	R	E	T																				
Cipher Text	V	H	F	U	T	W																				

As shown in Table 1, Caesar Cipher is considered easy and simple to use which is one of its strengths because it does not require using any form of a complicated coding system and a few computing resources.

Problem Domain (Lapses of the Caesar Cryptographic Algorithm)

The simplicity of the Caesar cipher is its primary weakness basically because little work is required to decode the algorithm. Ochoche (2012), affirms that the Caesar Cipher has a huge gap in its algorithm that makes it a thing of concern; its simple form and ease of use make the Caesar Cipher encryption and decryption algorithm weak and less secure; another pressing security concern is the fact that if one letter is deciphered the entire message can easily be known and most times the system can be deciphered even without knowing the encryption key.

Consequently, the research undertaken in this study has pinpointed the following key lapses associated with the conventional Caesar cipher:

(I) The Caesar cipher is limited to only ***n*** key possibilities, where ***n*** corresponds to the number of characters in the alphabets for the language which it is implemented(i.e English). This implies ***1>= k <= (n -1)***, where ***n***=26 for the English alphabet set and **k** = key value.

(II) The conventional Caesar cipher is restricted to only the uppercase characters of the English alphabets which is not a comprehensive representation of the English Language characters; as it ignores the lowercase alphabet versions: ***a,b,c…z,*** the numbers: **0,1…9** and special characters/symbols(!,;'… /).

(III) The alphabet sequence used for encryption retains the conventional alphabetic order i.e in the order a, b, c, d,e…x,y,z. And this contributes significantly to the algorithm's weakness and predictability

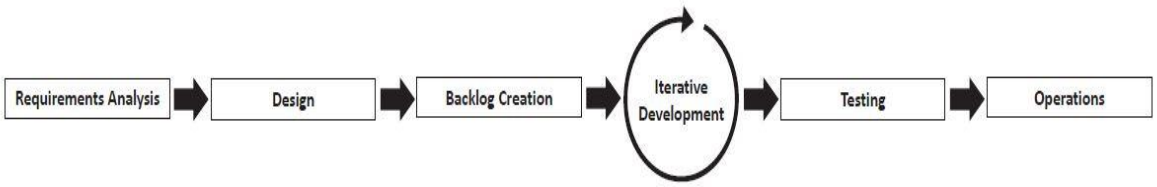
(IV) The encryption key is generated by one party of the duo involved; This presents a weakness in the scheme. i.e the generated key can be intercepted during transmission by third parties.

(V) The computational complexity of the conventional Caesar cipher is of order: ***O(n)*** - a linear and low complexity that implies little work to decipher and reverse-engineer.

Research Methodology and Data

This research identifies the core lapses of the conventional Caesar cipher as aforementioned, therein an enhanced version is developed that curtails the identified lapses. The methodology adopted was the waterfall model primarily because it shapes and captures the requirements needed to develop the Enhanced Caesar Cipher software; encompassing the following phases - requirements, design, implementation, testing, and deployment/maintenance as represented by Nils et al (2020) alternatively as Requirements Analysis, Design, Backlog Creation, Iterative Development, Testing, Operations.

Figure 1: *The Waterfall Model.*



Source: (Nils et al, 2020)

Tools

In this study, the typical requirements for this model encompass the following software tools/technologies: (i) Java Development Kit (JDK) (ii) Java Virtual Machine(JVM) (iii) Javac compiler (iv) Netbeans IDE (v) Java programming language to develop the software in an Object-Oriented fashion. The Data used for experimentation and testing the model was sourced from a blend of primary and secondary sources. This data constitutes random texts from the individuals, internet, books, materials that were to test the algorithm's performance on a wide variety of data so as to attest its efficacy.

Model Design

Table 2: *Techniques Employed for Improving the Conventional Caesar Cipher*

Case	**Improvement Technique Employed**
1. The Caesar cipher is limited to only ***n*** key possibilities, where ***n*** corresponds to the size of the alphabets(Upper case): **A, B, C, D…Z** for the language in which it is implemented(i.e English). This implies ***1*>= *k* < = *(n -1)***, where ***n***=26 for the English alphabet set and **k** = key value. Therefore ***k*<= *25.***	The lower case alphabet set : *a,b,c,d…x,y,z* with size ***x = 26*** ; the special character set : *`¬!,"£…/* with size ***y =36;*** the numbers set: *0,1,2,3…9* with size ***z = 10*** have been added to form the new English character set i.e ***n* = n_i + *x* + *y* + *z* = 98.** Hence the range of key possibilities has increased from 26 to 98. With ***1*>= *k* <= *(n -1)***, where ***n***=98. i.e ***k*< =*97.*** Although the key value range has been increased from 25 to 97; more complexity and abstraction have been added as the key to be used at run time has been set to be autogenerated using key components contribution(alongside intense mathematical computations) from the parties involved as shown in case 4.
2. The context of the Caesar cipher encompasses only the uppercase characters of the English alphabets which is not a comprehensive representation of the English Language characters; it ignores the lowercase alphabet versions: ***a,b,c…z,*** the numbers: **0,1…9** and special characters/symbols(!,;'… /).	The omitted characters are been incorporated into the character set as depicted in techniques employed in case 1 thereby solving the problem described in case 2.
3. The alphabet sequence used for encryption retains the conventional alphabetic order	The alphabetical sequence for encryption is restructured via the following steps:

i.e in the order a, b, c, d,e…x,y,z. And this contributes significantly to the algorithm's weakness and predictability.	**STEP 1** – The alphabetical sequence is inverted i.e reversed with *z,y,x* retaining positions 1,2,3. i.e the new alphabetical order is in the form *z,y,x,w…d,c,b,a* **STEP 2** – A permutation is performed on the new alphabet order. For every ***a*** $\subset$ ***n*** The permutation function swaps the character at the ***a*** th position with the character at the ***(a+1) th*** position. Consequently yielding an alphabetical sequence in the order: ***y,z,w,v…b,a,d*** in the basic sense but realistically, this new order is blended with a mix of special characters say: -,.;'…etc **STEP 3** – Another permutation is executed on the new alphabet order such that For ***a*** $\subset$ ***n,*** *where the* ***a = a+2*** at every iteration, the permutation function swaps the character at the ***a*** th position with the character at the ***(n+1- a) th*** position. Consequently yielding a an alphabetical sequence in the order: ***d,z,b,v…w,a,y*** And with this, a more complex alphabetical sequence is obtained
4. The Encryption key is decided by one party of the duo involved; This presents a weakness in the scheme. i.e the generated key can be intercepted during transmission by third parties.	Contributions are made by both parties to generate the encryption key. i.e party 1 supplies a numeric value of **a** and party B supplies a numeric value of **b**. The Diffie Helman technique is used to generate the encryption key via mathematical computations on the ***p*** and ***q.*** ***Diffie helman Technique:*** 1. Alice and Bob publicly agree to use a modulus $p = 23$ and base $g = 5$ (which is a primitive root modulo 23). 2. Alice chooses a secret integer ***a*** = 4, then sends Bob $A = g^a \bmod p$ ○ $A = 5^4 \bmod 23 = 4$ 3. Bob chooses a secret integer ***b*** = 3, then sends Alice $B = g^b \bmod p$ ○ $B = 5^3 \bmod 23 = 10$ 4. Alice computes $s = B^a \bmod p$ ○ $s = 10^4 \bmod 23 = 18$ 5. Bob computes $s = A^b \bmod p$ ○ $s = 4^3 \bmod 23 = 18$ 6. Alice and Bob now share a secret (the number 18).

	Both Alice and Bob have arrived at the same values because under mod p, $A^b \bmod p = g^{ab} \bmod p = g^{ba} \bmod p = B^a \bmod p$ More specifically, $(g^a \bmod p)^b \bmod p = (g^b \bmod p)^a \bmod p$ Source: (Wikipedia, 2021)
5. The conventional Caesar Cipher encrypts a single character as a single cipher character. say: **A** is encrypted as C using a key, k = 3.	The Enhanced Caesar Cipher encrypts each character with an equivalent of two cipher characters; if the mathematical computations have yielded a final key, **k = 1**, **A** is encrypted as **&***. thereby providing greater abstraction and complexity.
6. The computational complexity of the Caesar cipher is of order: ***O(n)***. A linear and low complexity which implies little work to decipher and reverse-engineer.	From analysis of the aforementioned steps, $\boldsymbol{O(n) = n + n^2 + n^2}$ I.e the computational complexity of the Enhanced Caesar Cipher is of the order $\boldsymbol{O(n^2)}$**.**

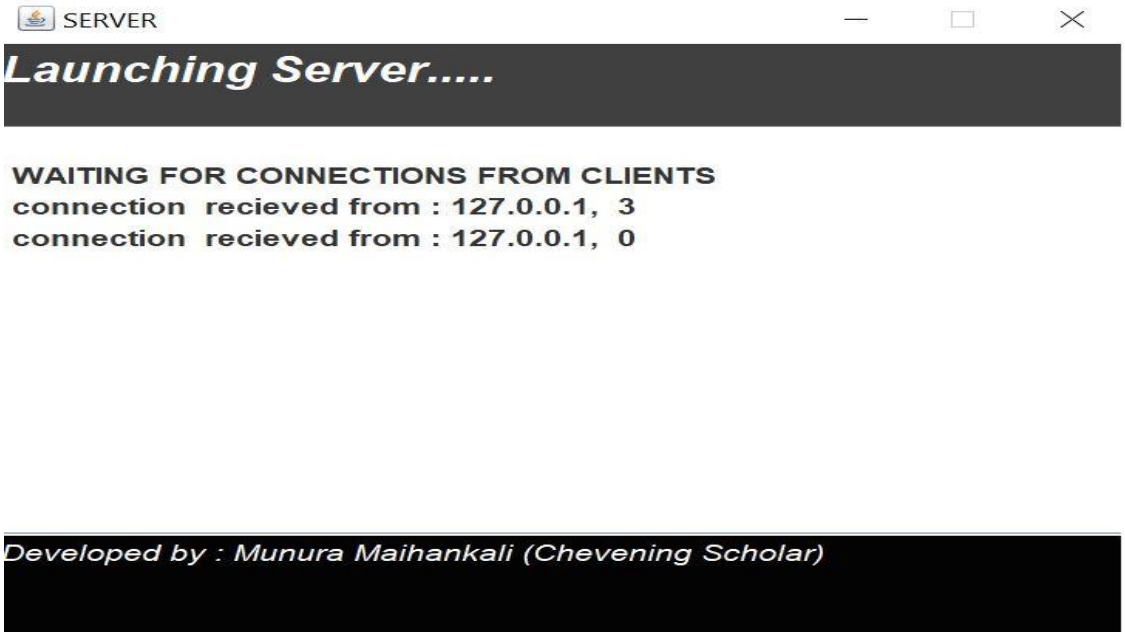
Results and Discussions

The actual implementation of the developed model with complete source code and in-depth description of its usage can be found on GitHub via:

https://github.com/Maihanks/EnhancedCaeserCIpher

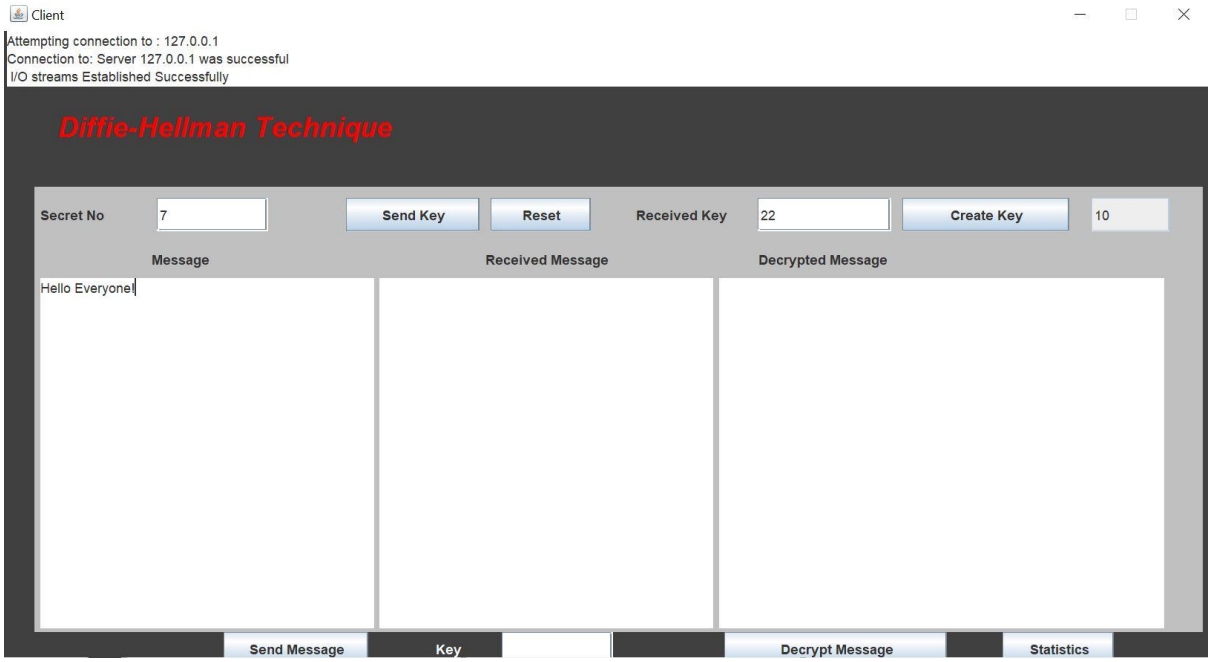
The Enhanced Caesar Cipher in usage at real-time is illustrated below:

Fig1: The Server



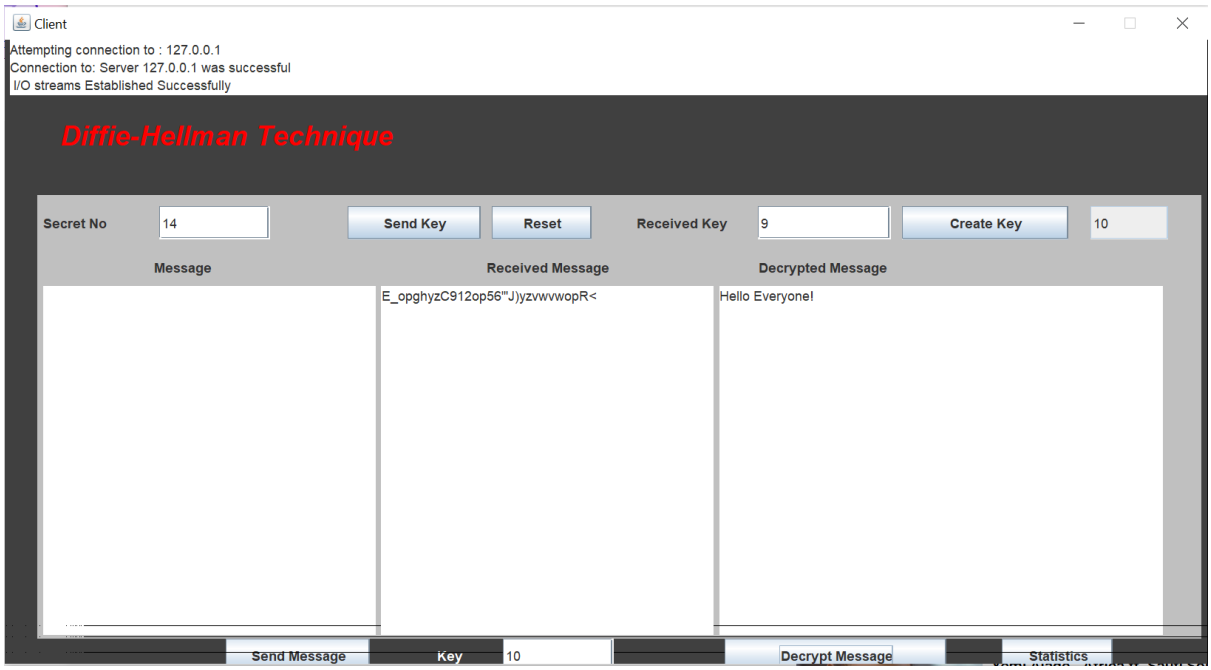
The server version of the program is launched and client programs connect via sockets using the server's IP address, further incorporating parallelization for concurrency in communication between the clients. As shown above, two clients systems have connected to the server using the streams 0 and 3 via host(Server) IP = 127.0.0.1

Fig 2: *Clients Connected to the Server*



A client system launched, successfully connected to the server, exchanged contributory key components by clients resulting in a generated key of 10 with the message to be encrypted: "*Hello Everyone*".

Fig 3: *Message encrypted, exchanged, Decrypted*



The encrypted message was encrypted and exchanged with the second client as shown above; and using key 10, the message was decrypted on the other end. The dissimilarity between the original message and the cipher message can be observed and analysed.

5.0 Recommended Improvements and Research

The enhanced Caesar Cipher as elucidated in this study is tailored to the encryption of textual data only, however, there are variant forms in which data exist such as multimedia comprising images, audios, and video files. The model developed in this study does not make provisions for the encryption/decryption of these multimedia data types, hence it is the intention of the researchers/authors of this work that the study carried out and result obtained severs as a basis for incorporating/integrating the capability of encrypting the multimedia files into the Enhanced Caesar cryptographic scheme. A comprehensive and detailed description of the architecture, design, functionalities of the software developed can be found on Github via this link: *https://github.com/Maihanks/EnhancedCaeserCIpher*

References

Aamer, N., & Muhammad, Y. (2005). A performance Comparison of Data Encryption Algorithms. IEEE. DOI: 10.1109/ICICT.2005.1598556

Atish, J., et al. (2015). Enhancing the Security of Caesar Cipher Substitution Method using a Randomized Approach for more Secure Communication. *International Journal of Computer Applications*, *129* (13), 6-11. DOI: 10.5120/ijca2015907062

Desoky, et al. (2008). An Implementation of the Blowfish Cryptosystem. *IEEE*.

Diaa, S., et al. (2008). Performance Evaluation of Symmetric Encryption Algorithms. *International Journal of Computer Science and Network Security*.

Diaa, S., et al. (2010). Evaluating the Effects of Symmetric Cryptography Algorithms on Power Consumption for Different Data Types. *International Journal of Network Security*, 78-87.

Inan, Y. (2019). Analyzing the Classic Caesar Method Cryptography. *4th International Conference on Computational Mathematics and Engineering Sciences* (pp. 213-220). Turkey: ResearchGate.

Jangir, N. (2014, April). Cryptography Security System. Kota.

Javed, A. N. (2005). A Performance Comparison of Data Encryption Algorithms. *IEEE*.

Mansoor, E., et al. (2013). Symmetric Algorithm Survey: A Comparative Analysis. *Internal Journal of Computer Applications*, 0975-8887.

Monika, A., & Pradeep, M. (2012). A Comparative Survey on Symmetric Key Encryption Techniques. *International Journal on Computer Science and Engineering (IJCSE)*, 877-882.

Mousa, A. (2005). Data Encryption Performance Based on Blowfish. *47th International Symposium ELMAR.*

Munir, M. W. (2005). Cryptography. *ResearchGate.*

Nils, P., et al. (2020). How are Hybrid Development Approaches Organized? – A Systematic Literature. Review. *IEEE/ACM International Conference on Software and*

System Processes (ICSSP)

Ochoche Abraham, G. O. (2012). An Improved Caesar Cipher (ICC) Algorithm.

International Journal of Engineering Science & Advanced technology,

1198 – 1202.

Piotr, B., & Wiesław, W. (2010). Multi-core Implementation of the Symmetric

Cryptography algorithms in the measurement system. *Measurement*, *43*(8), 1049-1060. https://doi.org/10.1016/j.measurement.2010.03.002

Point, T. (2018). Java Cryptography. *Tutorials Point (I) Pvt. Ltd*.

Rountree, D. (2011). Security for Microsoft Windows System Administrators. *ScienceDirect*.

Savarese, C., & Hart, B. (2010). The Caesar Cipher. Retrieved from http://www.cs.trincoll.edu /~crypto/historical/caesar.html

Senthil, K., et al. (2013). A modern avatar of Julius Caesar and Vigenere cipher. *IEEE International Conference.* Computational Intelligence and Computing Research (ICCIC).

Sharma, H., et al . (2010). Implementation and analysis of various symmetric cryptosystems. *Indian Journal of Science and Technology*.

Sinkov, A. (1996). Elementary Cryptoanalysis – A Mathematical Approach.. *The Mathematical Association of America*. Washington, D.C.

Verma, O.P et al. (2011). Performance Analysis Of Data Encryption Algorithms. *IEEE.* Delhi

Technological University India.

Wikipedia. (2021). Retrieved from:

https://en.wikipedia.org/wiki/Diffie%E2%80%93Hellman_key_exchange.

Zhang, T. N., & Teng. (2009). A Study of DES and Blowfish Encryption Algorithm. *IEEE*.