DATA ENCRYPTION AND DECRYPTION WITH CHAOTIC RANDOM SEED

BY

Md. Alamin-Ul –Islam

ID: 161-15-944

Md. Rabiul Islam

ID: 161-15-945

AND

Md Jahidul Islam Sany
ID: 161-15-913

This Report Presented in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science and Engineering

Supervised By

Saif Mahmud Parvez
Lecturer
Department of CSE
Daffodil International University

Co-Supervised By

Tajim Md. Niamat Ullah Akhund
Lecturer
Department of CSE
Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

DECEMBER 2019

APPROVAL

This Project titled "DATA ENCRYPTION AND DECRYPTION WITH CHAOTIC RANDOM SEED", submitted by Md. Alamin-Ul–Islam, Md. Rabiul Islam and Md. Jahidul Islam Sany to the Department of Computer Science and Engineering, Daffodil International University, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 26 December, 2019.

BOARD OF EXAMINERS

DECLARATION

We hereby declare that, this project has been done by us under the supervision of **Saif Mahmud Parvez, Lecturer, Department of CSE** Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

Supervised by:

____

Saif Mahmud Parvez
Lecturer
Department of CSE
Daffodil International University

Co-Supervised by:

____

Tajim Md. Niamat Ullah Akhund
Lecturer
Department of CSE
Daffodil International University

Submitted by:

____

Md. Alamin-Ul-Islam
ID: 161-15-944
Department of CSE
Daffodil International University

____

Md. Rabiul Islam
ID: 161-15-945
Department of CSE
Daffodil International University

____

Md Jahidul Islam Sany
ID: 161-15-913
Department of CSE
Daffodil International University

ACKNOWLEDGEMENT

First we express our heartiest thanks and gratefulness to almighty God for His divine blessing makes us possible to complete the final year project/internship successfully.

We really grateful and wish our profound our indebtedness **Saif Mahmud Parvez, Lecturer,** Department of CSE Daffodil International University, Dhaka. Deep Knowledge & keen interest of our supervisor in the field of "*Data Security*" to carry out this project. His endless patience ,scholarly guidance ,continual encouragement , constant and energetic supervision, constructive criticism, valuable advice, reading many inferior draft and correcting them at all stage have made it possible to complete this project.

We would like to express our heartiest gratitude to **Dr. Syed Akhter Hossain, Professor and Head,** Department of CSE, Daffodil International University, Dhaka **Dr. S M Aminul Haque, Associate Professor & Associate Head,** Department of CSE Daffodil International University, Dhaka for his kind help to finish our project and also to other faculty member and the staff of CSE department of Daffodil International University.
We would like to thank our entire course mate in Daffodil International University, who took part in this discuss while completing the course work.

Finally, we must acknowledge with due respect the constant support and patients of our parents.

Abstract

In neoteric time, network and data security has become most relevant concern because of the volume of exchange hypersensitive data over internet flourishing day by day. Many approaches are obtainable in the field of data security, one is that “Cryptography”. In cryptography, when data is transferred from sender to receiver data is encrypted by encryption algorithm and inversely when it received by the receiver it is decrypted using decrypted algorithm to view the exact and real data that send by the sender before. There are many procedure for data encryption .Such as: AES, DES, RSA algorithm. In this paper a new algorithm proposed which is CRSA (Chaotic random seed algorithm) technique for data encryption and decryption. Also compared CRSA with existing algorithms and their performance. This paper also provides experimental result to analyze those algorithms and our propose CRSA algorithm.

TABLE OF CONTENTS

CONTENTS	PAGE
Board of examiner	i
Declaration	ii
Acknowledgements	iii
Abstract	iv
CHAPTER	
CHAPTER 1: INTRODUCTION	**1-3**
1.1 Introduction	1
1.2 Motivation	1
1.3 Rationale of study	2
1.4 Expected output	2
1.5 Report layout	3
CHAPTER 2: BACKGROUND	**4-8**
2.1 Introduction	4
2.2 Related works	4-6
2.3 Research Summary	6
2.4 Scope of the problem	7
2.5 Challenges	8
CHAPTER 3: RESEARCH METHODOLOGY	**9-23**
3.1 Introduction	9

3.2 Research Subject and Instrumentation 9

3.3 Implementation of Algorithm 9-22

3.4 Implementation requirements 23

CHAPTER4: EXPERIMENTAL RESULTS AND DISCUSSION **24-29**

4.1 Introduction 24

4.2 Statistical Analysis 24-28

4.3 Descriptive Analysis 28

4.4 Experimental Results 29

CHAPTER5: SUMMARY,CONCLUSION, RECOMMENDATION AND IMPLICATION FOR FUTURE RESEARCH **30-31**

5.1 Introduction 30

5.2 Summary 30

5.3 Conclusions 30

5.4 Recommendation 31

5.5 Implication for future study 31

REFERENCES **32**

LIST OF FIGURES

FIGURES	PAGE NO
Figure 3.3.1: The probability to find a random number with different digit.	12
Figure 3.3.2: Encryption process flowchart for CRSA.	14
Figure 3.3.3: Decryption process flowchart for CRSA.	15
Figure 3.3.4: The process (Class diagram) of sending data from sender to receiver for CRSA.	16
Figure 3.3.5: Graphical User Interface (GUI) for sending a message using CRSA (Homepage).	17
Figure 3.3.6: Graphical User Interface (GUI) for sending a message using CRSA from USER 1.	17
Figure 3.3.7: Graphical User Interface (GUI) for Encrypting and Decrypting a message using CRSA.	18
Figure 3.3.8: Graphical User Interface (GUI) after sending a message and receiving by USER 2 using CRSA.	19
Figure 3.3.9: Pseudo code for generating Random Seed.	21
Figure 3.3.10: Pseudo code for Encryption and Decryption.	22
Figure 4.4.1: Encryption Time Comparison for different algorithm.	26
Figure 4.4.2: Decryption Time Comparison for different algorithm.	26
Figure 4.4.3: Memory Utilization Comparison for different algorithm.	27
Figure 4.4.4: Throughput Comparison for different algorithm.	27

LIST OF TABLES

TABLE	**PAGE NO**
Table 1: Comparison test data and their performance for various algorithm.	25

CHAPTER 1
INTRODUCTION

1.1 Introduction

Data communication technology getting more advance and more susceptible to handle huge amount of data within a short time. So data security is most important issue to maintain these data to protect any fault or harm. To protect data encryption and decryption methods are most popular and used method in now a days. There are two types of encryption and decryption: Symmetric and Asymmetric keys. They are also known as Private and Public keys encryption. In asymmetric algorithm's both public and private key is used. [1].For encryption public keys are generally used as well as private keys used for decryption. [2]. There are various types of encryption procedure like as AES, AES-256, DES, 3DES and RSA. All of them are designed based on symmetric or asymmetric keys where some of them used both that means public keys for encryption and private keys for decryption. Example: RES.

1.2 Motivation

Now a days, the availability of fast data transmission ways, process, devices are increasing. It's essential to ensure secure transmission, secure the secrets. E-commerce, E-governance, Social media (ex: WhatsApp, Twitter, Facebook, Viber, Instagram, LinkedIn), agencies, institutions, industries, need to be secured against any kind of data loss or steal data, unauthorized use and modification of any data. To secure the secret's secret, cryptography maybe the best solution. Using encryption methods cryptosystem provides data integrity, authentication and secure transmission. So it's a big deal to provide security in data transmission network. . Predictors believes that the third world war will be a cyber-war. That's why data licking, stolen are common issues. This reason motivated us to increase our knowledge on data security methods.

1.3 Rationale of study

There are major three challenges in data security or cryptosystem. The main challenges is to ensure data privacy from end to end data transaction. Also protect any types of hacking and malware problems. Another is to identify and removal unauthorized access. Advanced Encryption Standard (AES), Data Encryption Standard (DES), and Rivest, Shamir Adleman (RSA) are the most common encryption method. It is extremely important to access, store and manipulate safe and unthreatened way. For the data privacy and control generally data is encrypted before. So it is a big challenge to provide secure services to the customers which also need to be efficient. Many researcher proposes many schemes and models with different systems. All of these research paper showed great effort and hard work to design solutions like as: high scheme efficiency, stateless verification, unbounded use of queries and irretrievability of data etc. An efficient decentralized multi-authority attribute-based scheme in [3]. Some of them introduced a hybrid algorithm's which is a mix algorithm of P-AES algorithm and RSA algorithm [4]

1.4 Expected output

In this paper, we propose a new encryption and decryption method using chaotic keyword and also discuss our method efficiency with key length, algorithm's complexity, best method of attack. There are some encryption algorithms like SHA256 and PKDBF2, these are used for save user information by encryption with a specific salt. And these algorithms are not reversible. Beside apart from user password we can't save user's other information with encryption. So if anyone have access of database, then he can steal a big amount of user's data in very short time. So we need a reversible encryption and decryption method to store all data into database. Data transaction is also an important things for user privacy and security. So we can use a method, which can assure the data protection that user transect with another users and third party can't retrieve the actual data. We know about SALT in existing encryption algorithms. This SALT can't be changed but in our algorithm we have used one time password that directly involve with encryption and decryption process. Generating of this one time password is a chaotic process. That means it's not static like salt. Encryption and decryption process isn't possible without those chaotic keywords. The paper discussed about the full process of encryption and decryption chaotic keyword and also discuss our method efficiency with key length, algorithm's complexity, best method of attack. .[5] In this paper, we have briefly discussed about different types of data security issues, procedure and data encryption and decryption Algorithm's. Our proposed algorithm will increase data security more and solves random seeds issues using millisecond as random seed.

1.5 Report layout

This paper is organized into five chapters. In which the first chapter is Introduction about the research work and the motivation which motivated to take up this research work. It also gives the introduction about the concepts of cryptography: data encryption and decryption algorithms and expected output.

The second chapter is about the Background which represents comprehensive review of the published and unpublished work in the field of data security. This helped us in reviewing the literature on the research area and focuses more on the research work.

The third chapter presents about Research Methodology. Research subject and instrumentation, data collection procedure, statistical analysis are represent there

The fourth chapter is about the Experimental Results and Discussion, it presents the strength of our proposed CRSA algorithm. Also discussed the Descriptive analysis of experimental analysis and at the last of that chapter a summary of the project is given.

The fifth and last chapter is about Summary, Conclusion, Recommendation and implication for future. In this chapter we discuss about the summary of study, conclusion of the report and project and implication of further study.

Apart from all these chapter, the report also consist of list of figures, diagrams, tables etc. that used in the report and also list of references used to carry out this project.

CHAPTER 2
BACKGROUND

2.1 Introduction

Encryption is one if the best solution and plays an important role in the field of information security. Public keys and private keys or combination of public keys and private keys are used to hide sensitive data and provide them best security. Advanced Encryption Standard (AES), Data Encryption Standard (DES) and Rivest-Shamir-Adleman (RSA) are the most usable algorithm in recent past.

2.2 Related works

Now in this report, we will briefly discuss About AES, DES and RSA algorithm which are the well-known methods of encryption.[6]

2.2.1 Advanced Encryption Standard (AES)

Advanced Encryption Standard (AES) is most popular data encryption method for its faster speed and faster hardware-software installation. [7] Another advantage of this algorithm is that, it can easily implemented in different types of platform especially is small size devices. And it is now used in various security system. AES encrypts data blocks of 128 bits in 10, 12 and 14 round depending on key size[6].

*Encryption***:** Each round have 4 steps: Sub-bytes, Shift Rows, Mix column and Add round key.

i) Sub-bytes: The first step is Sub-bytes which is used in encryption site. To interpret the byte as two hexadecimal digit for substituting a byte.

ii) Shift Rows: Transmission is known as Shift Rows in encryption.

iii) Mix column: Mix column transforms each column of the step from a column to a new column at the column level.

iv) Add Round Key: Add Round Key Execute one column at a time. Add Round Key adds a round key word with each column matrix.

The last state provides XO ring the output of the previous three steps with four words from the key schedule as well as the last round for encryption do not involve the "Mix columns" step. [8]

*Decryption***:** Decryption is the reverse process of encryption using inverse function that used in encryption state. Such as:

i) Inverse shift rows,
ii) Inverse substitute bytes,
iii) Add round key and
iv) Inverse-mix columns.

The last state provides XO ring the output of the previous three steps with four words from the key schedule as well as the last round for encryption don't involve the "Inverse mix columns" step.

2.2.2 Data Encryption Standard (DES*)*

Data Encryption Standard **(**DES)generally used for providing security to commercial and unclassified data. Similar key is used for DES encryption as well as decryption. [9]

The main steps for encryption are:

1. Data Encryption standard (DES) consists 64 bit size plaintext and 56 bit key in which 8 bit parity and output is 64 bit.
2. Plaintext blocks shift bits and parity 8 bits are removed from the key to its permutation.
3. Then need to divide the key into 28 part each and every half/part rotate by 1 or 2 bits.
4. All the partition reduce the key 56 bit (half) gathered and compress permutation 56 bits to 48 bits and the compressed key is generally used for encryption at this round Plaintext box.

5. Now the rotate key halves which are from step 4used in text round and data block divide into two 32bit half. One half is an expansion permutation to increase the size to 48 bits.
6. Output is XOR with 48 compressed key and its result is fed into S-box which reduces 48bit block to 32 bit and P-box to permute the bits.
7. The output from step 7(P-box) is XOR with another half of data-block, which are swapped and used for next round input.

2.2.3 Rivest-Shamir-Adleman (RSA)

RSA which is known as First public key cryptosystem (Symmetric) proposed by Ronald Rivest, Adi Shamir and Leonad Adleman in 1978. [10] This algorithm is classified into 3 steps. These are:

i) Primary key Generating
ii) Encryption step
iii) Decryption Step

Encryption process: In RSA both public and private key is used, public key can be known by all which is used for encryption. The plain text including public key is called chipper text.

Decryption process: The chipper text decrypted into plain text using decryption algorithm. A private key is used for decryption.

2.3 Research summary

After analyzing AES, DES algorithms we can identify their working procedure, encryption and decryption approaches and measure efficiencies.

Advance Encryption standard (AES) properties:

i. Developed in 2000 and key size is 198,192,256 bits, block size is 128 bits.
ii. Low power consumption.
iii. AES is faster than DES (Data Encryption standard) and other method.

iv. S-Box eliminate the symmetric.
v. Very Difficult to hack or attacking secure data especially hyper sensitive data.
vi. Same keys used for encryption and decryption.
vii. Faster Software and hardware installation.
viii. Excellent Security system.

Data Encryption Standard (DES) properties:

i. Developed in 1977. Key size is 56 bits and block size is 64 bits.
ii. Low power consumption.
iii. Moderate speed but not faster than AES.
iv. Brute force attack is impossible in DES and timing attack is difficult.
v. Same keys used for encryption and decryption.
vi. Better in hardware than software installation.
vii. Not secure enough.

RSA algorithm properties:

i. Developed in 1978 and key size is 1024 bits, block size is minimum 512 bits.
ii. High power consumption.
iii. Slower than other's (Encryption and Decryption time).
iv. No one can crack the files and very difficult to brute force and oracle attack.
v. Different keys used for encryption and decryption.
vi. Not efficient hardware and software installation.
vii. Provide least security.

2.4 Scope of the problem

Reviewing existing algorithms and approaches of data security we can see that, RSA (Rivest-Shamir-Adleman) process is slower than others while encrypting long data. It's the main fact for not so many application of that techniques. Also complex hardware software and hardware installation. To solve these drawback DES (Data Encryption Standard) comes. DES is faster than RES and also efficient to installation but it is better in hardware, software installation still complex and tough. Both of them are same in the point higher power consumption. When Advanced Data standard (AES) introduced, it solves almost maximum drawback that consists in DES and RSA. It is faster than others to encrypt and decrypt, easy and efficient hardware and software installation and low power consumption. But still AES implementations with software installation is also complex, and security system is not so strong we expect. So we propose our Chaotic Random Seed algorithm (CRSA) to provide best security approach. We introduce random seed (millisecond) with OTP (One Time Password) that is most difficult to break. And we will try to make it efficient while implementing in hardware and software installation. We found some positive result that will discuss on the next chapter.

2.5 Challenges

We are working with Cryptography that means encryption and decryption. The challenges to provide best security in cryptosystem. We can point out the major challenges like as:

i. Faster encryption and decryption speed.
ii. Low power consumption.
iii. Easy and low cost implementation.
iv. Efficient to the users.
v. Excellent and strong security so that any type of hacking approach can hardly break the security.
vi. Safe and secure process that no one can crack the file and easy fraud detection.
vii. Fastest encryption for long and heavy data.
viii. Ensure randomness in technique’s to make it upgrade and more secure.

CHAPTER 3
RESEARCH METHODOLOGY

3.1 Introduction

This paper’s proposing algorithm will increase data security more and solves random seeds issues using millisecond as random seed. For encryption and decryption random seed generation is the first priority for implementing our algorithm. So in our research, first we have generated a very strong random seed as a key for our encryption and decryption process. We know that there are no truly random number that a computer can generate. So we use millisecond from real time system as a string of number and use various types of operation for make that number very complex and randomness behavior. We discuss about the probability to find out a random number based on its digit unit and implement our algorithm step by step with algorithm, implementation, flow chart and class diagram.

3.2 Research subject and Instrumentation

This paper is about encryption and decryption. Comparing and analyzing various method we are trying to develop a new better procedure. Our research subject is the implementation of an algorithm which is use random seed as a key for encryption and decryption. For sending some text data between end to end users our algorithm handle the data security. We have used different types of language and platform for implementation our algorithm and testing. We used HTML5, CSS3, Bootstrap 4 and JQuery for GUI (Graphical user interface). In backend we used JavaScript programming language for develop our algorithm. We use Windows Operating system and hardware configuration is Intel(R) Core(TM) i3-4150 CPU @ 3.50GHz. For simulating and compare with other algorithms we use C++ language with Borland C++ compiler.

3.3 Implementation of Algorithm

3.3.1 Generating of Random number with random seed

When a crave to encrypt and decrypt some data, we demand some random number or string to carry through an operation based on a specific algorithm for alternation the substantive data. But how can we attain the random number? There are abundant way to trace a random number. Random number propagation isn't truly "random". It is deterministic, and the conformity it spawn is injunction by the seed value you pass into random function.

$$random.seed(109)$$
$$x = random.random()$$
$$y = random.random()$$
$$1st\ random\ value = 0.27958303860586786$$
$$2nd\ random\ value = 0.45927897430988984$$

Now 109 is a static value. In our algorithm we propound a way that we can pick these random seed from our system's time by millisecond. There are 1000 millisecond in 1 second. So there are 999 numbers in 0 to 1000. The possibility of getting a random millisecond is not so robust, science there are only 999 numbers. But when we run iterations for numerous times, then we attain an immensely depth random numbers consequent to its former seed. So the step is,

Step 1. Carry a millisecond number as a seed.

Step 2. Conduct that seed and generate a new seed by algorithm.

Step 3. Run Iteration of step 2 numerous times for getting a very complex random number.

More length of seed matter more complexity in algorithm but it's enhance entire security. There are all 3 digit of numbers from 0 to 999. But we can increase the number of digits by multiplying some another random numbers.

3.3.2 Algorithm process with Random Number

This system work with a chaotic random number and a specific data. Random number appears from random seed. And data is the massage which is sent by a user to another user.

The procedure of algorithm is as follows:

i. A number R (0 < R <1000) taken as a seed from millisecond.

ii. Using that seed a specific number of random digit P (99999 < P <10^5, $range\ can\ be\ decrase\ or\ increase$) is generated from an algorithm.

iii. A massage or string is a source S. Let assume first character of the string is S_0 and the last character is S_n. Length of string is S_l

iv. Length of random Numbers = L

v. M = ASCII value (S_n) + ASCII value (i % L) (where 0 <= i <= S_l)

vi. Q = Convert M into Character according to ASCII value.

vii. Put the value in a JSON.

viii. Repeat the step (5-7) until the operation of S_n

3.3 Implementation

Encryption and decryption key, that means Random number is more significant than encryption scheme. Because when we work with security, our first priority is ascertain the protection of the key or random number. Generating of random number with random seed is not a complex process. And the probability of attack on this process is impossible to success because of its true randomness behavior.

We take 6 digit number as a Random Number. There are 900000 unit of number we can generate. The probability to take a random number from 900000 unit of numbers is

$$(999999 - 100000) + 1 = 900000$$

$$\frac{1}{900000} = 0.00000111111$$

But if we take 12 digit or 24 digit of random number, then the probability of find a random number is more decrease that showed in fig 3.3.1.

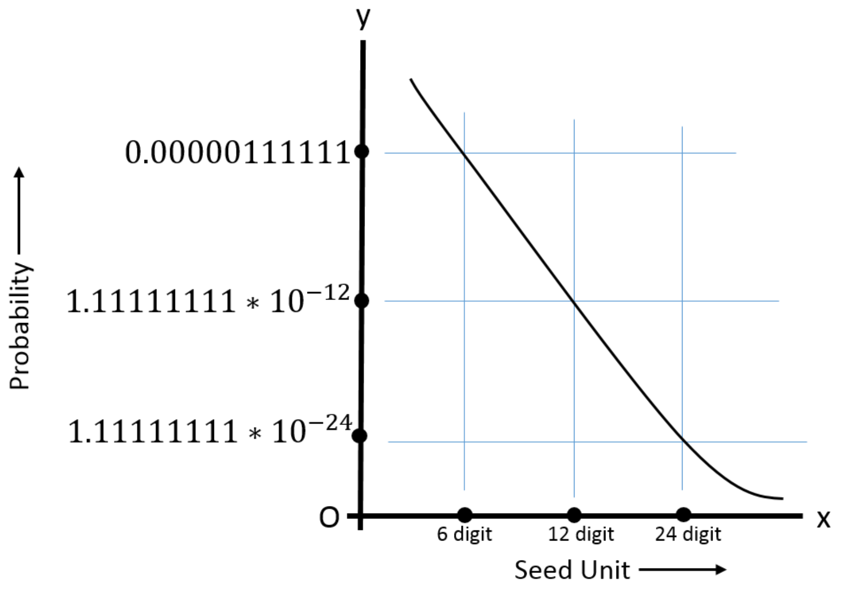


Figure 3.3.1: The probability to find a random number with different digit.

The efficiency in generate a specific number of prime numbers is an important factor for probabilistic algorithm. In cryptosystem the security depends on the key value or Seed. Find out a correct seed between 24 digits of random number is almost impossible for attackers. In fact our algorithm of generate random number doesn't maintain any sequence or any repetitive way to for generate a number.

After generating a random number we use that for encrypt a data. The formula is as follows,

$$L = \{x \in L \mid floor: x > 0 \qquad (3.3.1)$$

$$E_n = S_{n-1} + R_{\{(n-1)\ mod\ L\}} \qquad (3.3.2)$$

While encrypting a word "W" then we have to take a random number. Suppose our random number is a 6 digit number. And the number $R = 814673$. ASCII value $W = 87$, String-length $= 1$. $i = \{(n-1) mod\ 6\}$. ASCII value of $R_i(8) = 58$. So by equation (3.3.2)

$$Result: 87 + 58 = 145$$

$$\boldsymbol{Chipper\ text}: \boldsymbol{Charecter\ of\ Result}(\mathbf{145}) = \text{Å}$$

For Decryption,

$$D_n = S_{n-1} - R_{\{(n-1)\ mod\ L\}} \qquad (3.3.3)$$

By equation (3.3.3)

$$Result: 145 - 58 = 87$$

$$Original\ text: Charecter\ of\ Result(87) = W$$

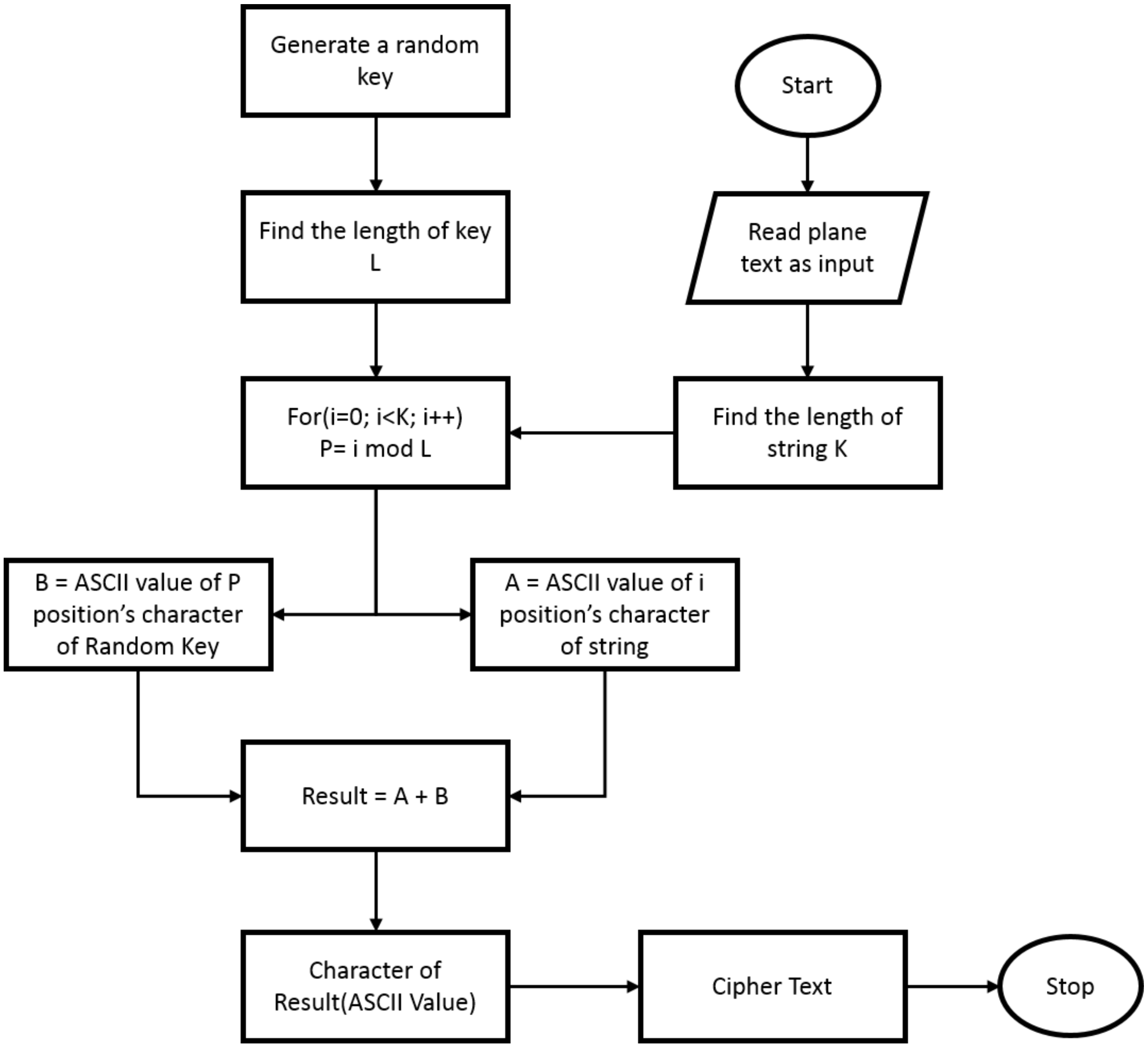

Generate a random key
Start
Find the length of key L
Read plane text as input
For(i=0; i<K; i++)
P= i mod L
Find the length of string K
B = ASCII value of P position's character of Random Key
A = ASCII value of i position's character of string
Result = A + B
Character of Result(ASCII Value)
Cipher Text
Stop

Figure 3.3.2: Encryption process flowchart for CRSA

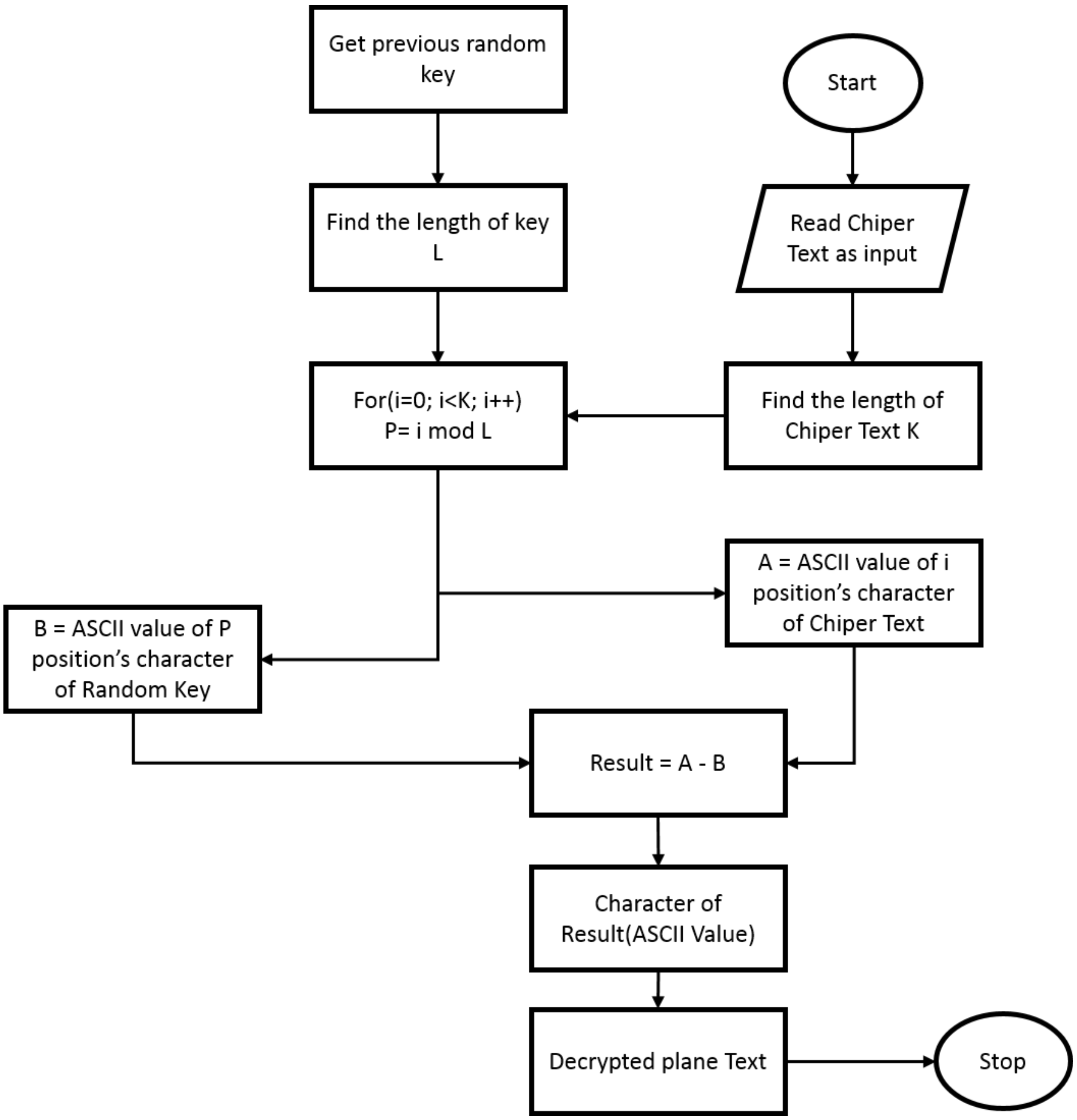

Get previous random key
Start
Find the length of key L
Read Chiper Text as input
For(i=0; i<K; i++)
P= i mod L
Find the length of Chiper Text K
A = ASCII value of i position's character of Chiper Text
B = ASCII value of P position's character of Random Key
Result = A - B
Character of Result(ASCII Value)
Decrypted plane Text
Stop

Figure 3.3.3: Decryption process flowchart for CRSA

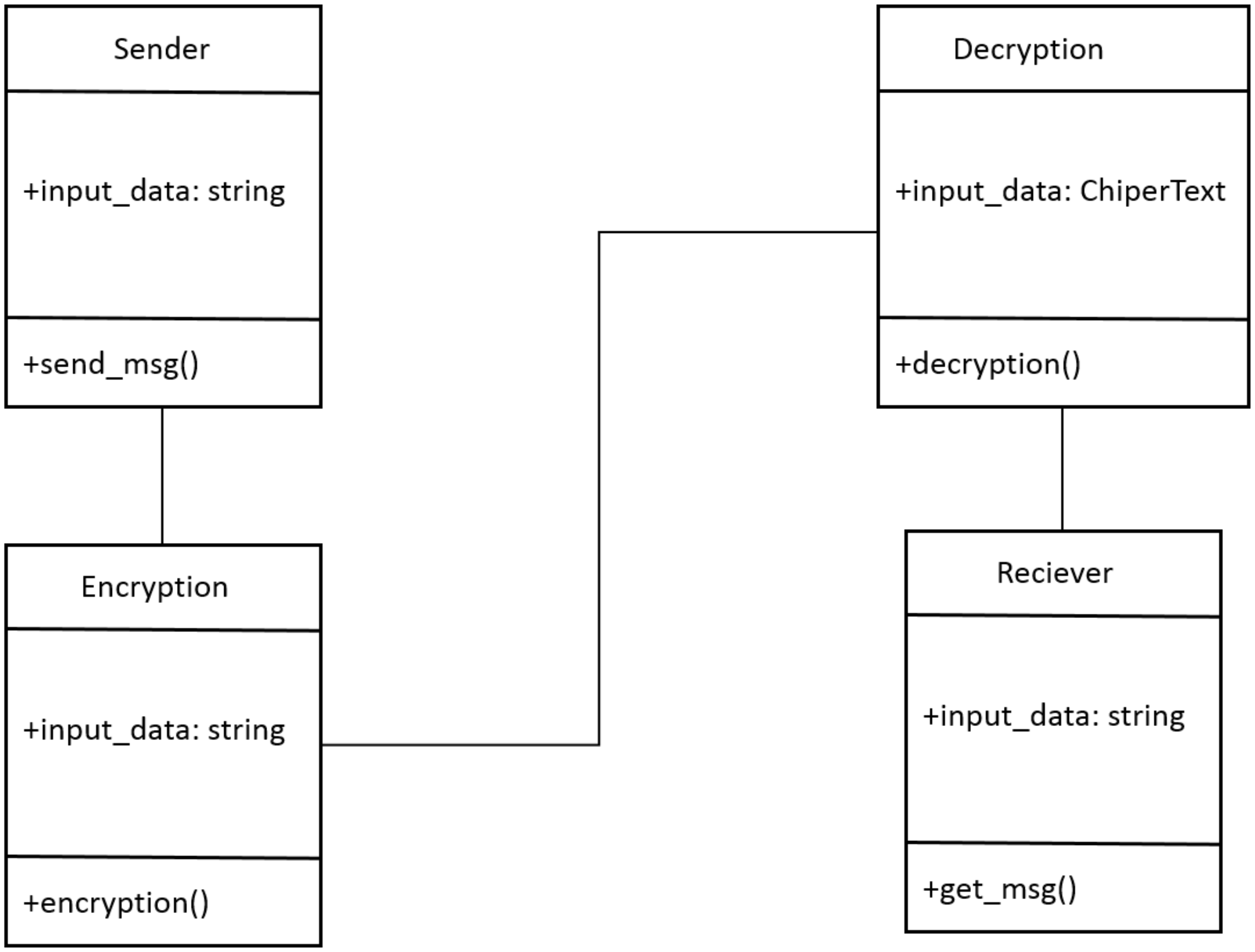


Figure 3.3.4: The process (Class diagram) of sending data from sender to user for CRSA

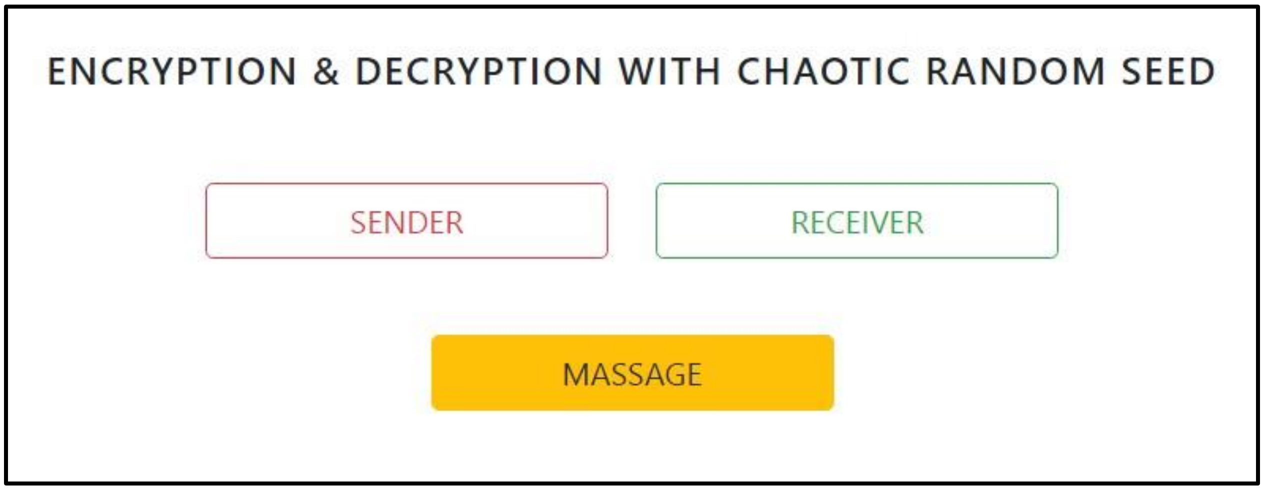
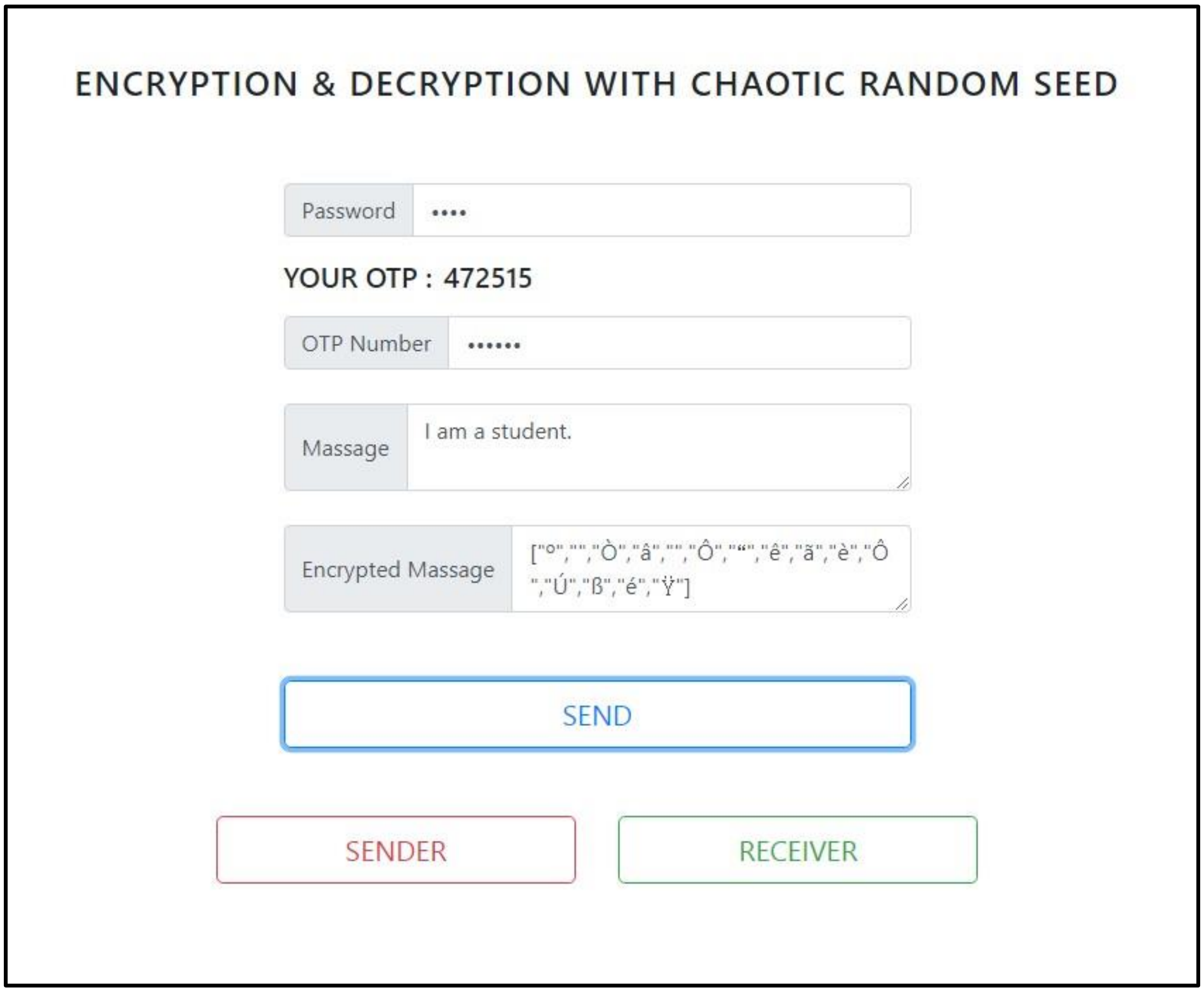


Figure 3.3.5: Graphics User Interface (GUI) for sending a message using CRSA. (Homepage)



ENCRYPTION & DECRYPTION WITH CHAOTIC RANDOM SEED

Password ••••

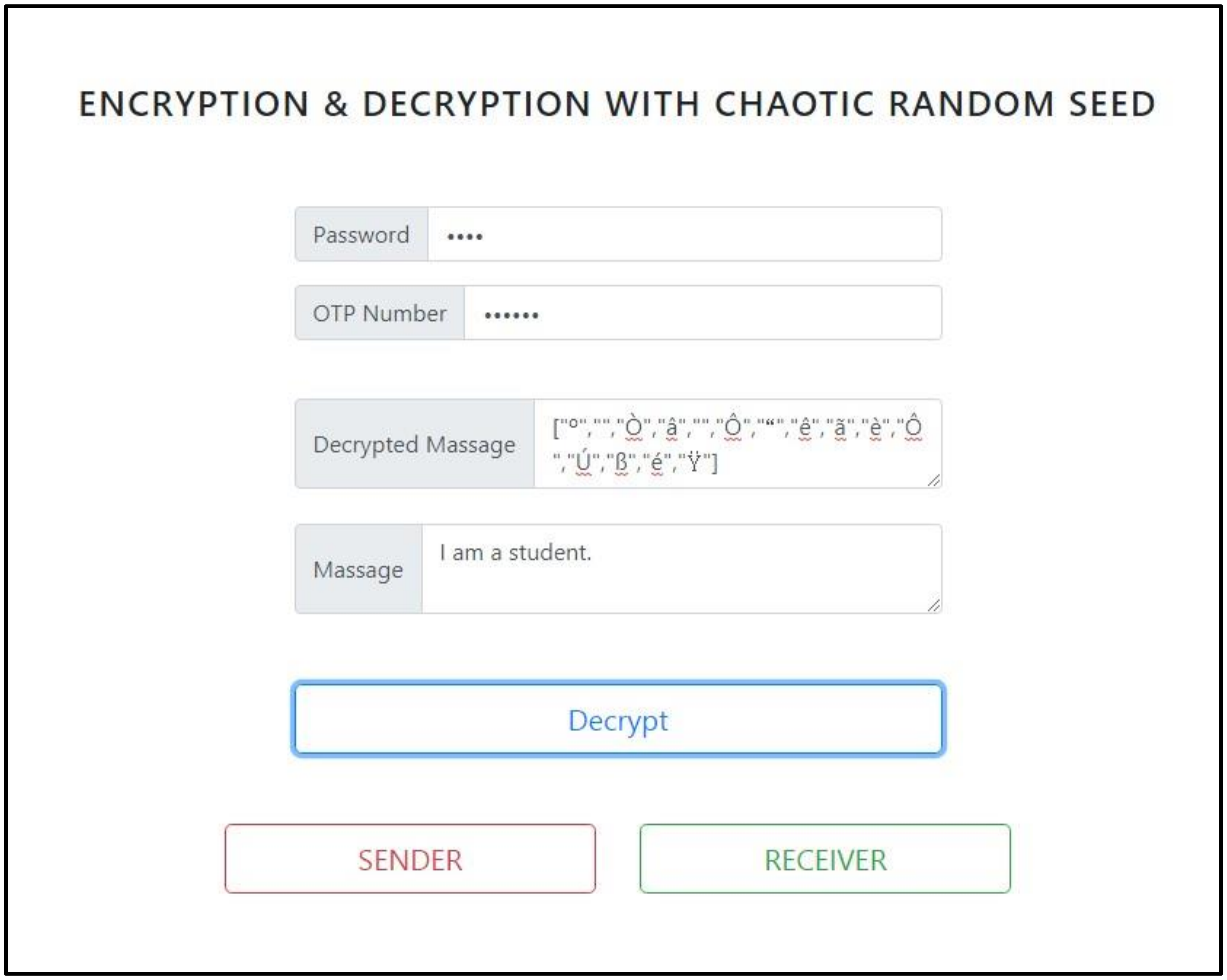
YOUR OTP : 472515

OTP Number ••••••

Massage: I am a student.

Encrypted Massage: ["°","","Ò","â","","Ô","“","ê","ã","è","Ô
","Ú","ß","é","Ÿ"]

Figure 3.3.6: Graphical User Interface (GUI) for sending a message using CRSA (USER1)



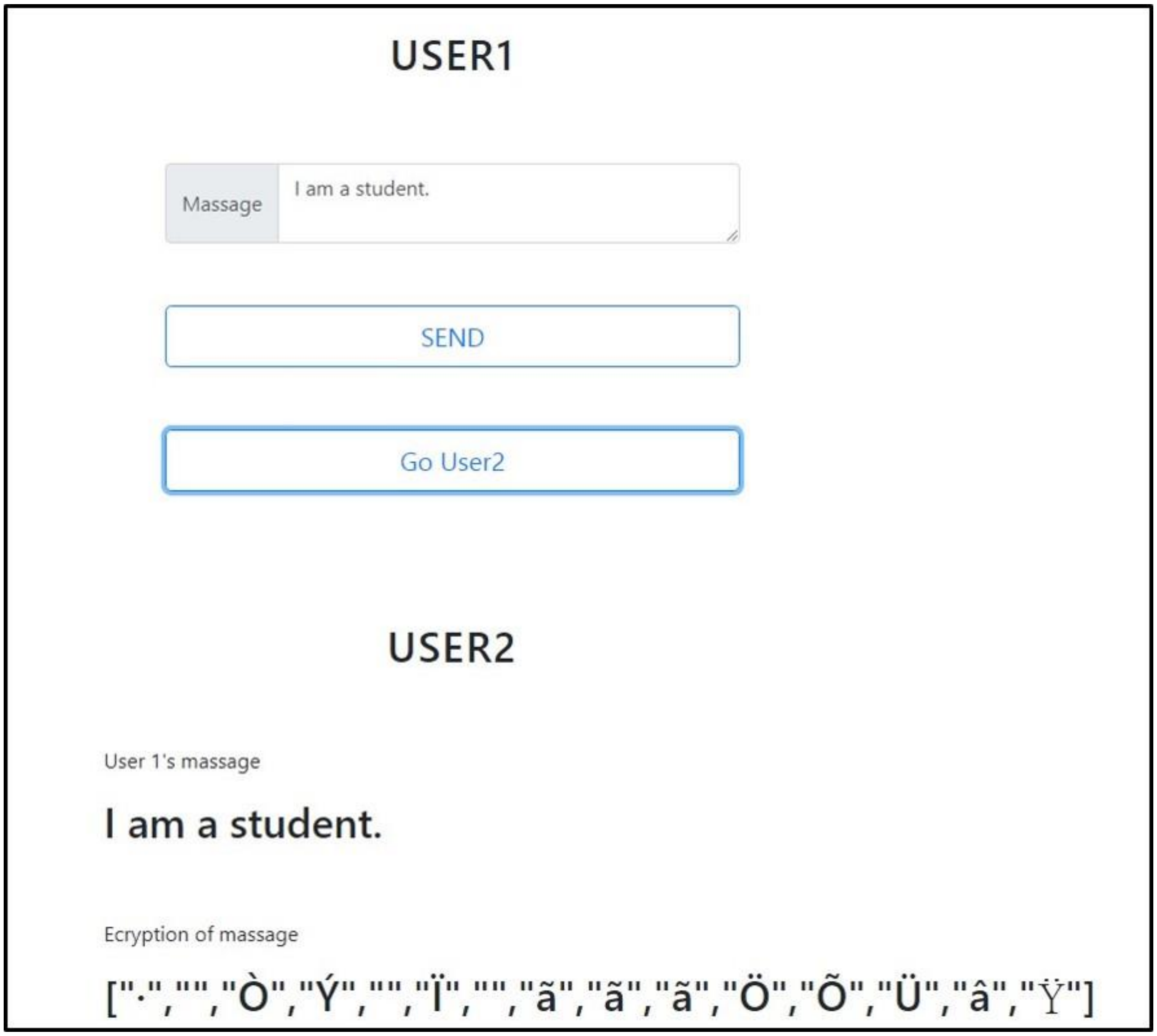
Password ••••

OTP Number ••••••

Decrypted Massage ["º","","Ò","â","","Ô","“","ê","ã","è","Ô
","Ú","ß","é","Ÿ"]

Massage I am a student.

Figure 3.3.7: Graphical User Interface (GUI) for Encrypting and decrypting a message using CRSA


USER1

Massage | I am a student.

USER2

User 1's massage

I am a student.

Ecryption of massage

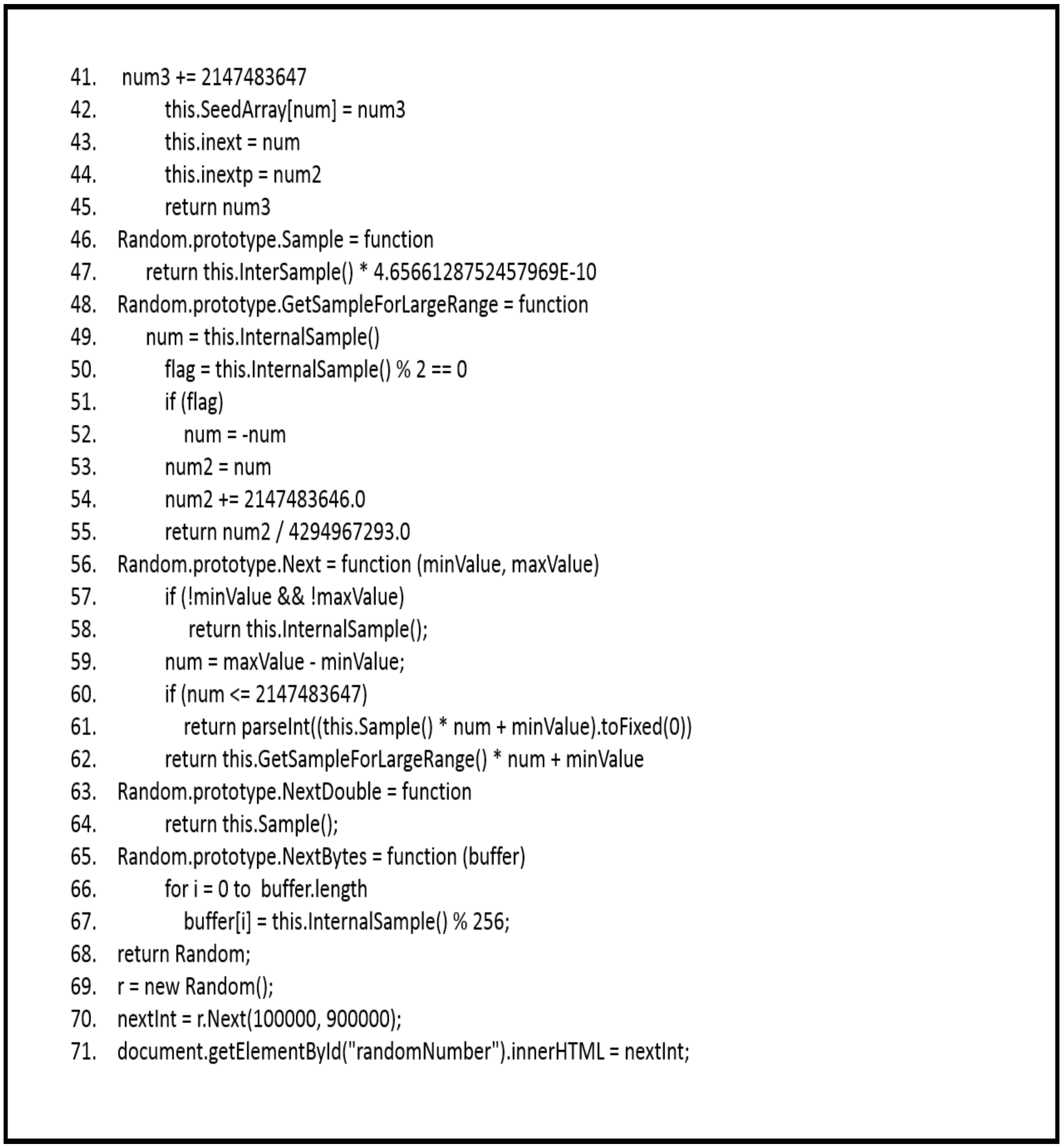
["·","","Ò","Ý","","Ï","","ã","ã","ã","Ö","Õ","Ü","â","Ẏ"]

Figure 3.3.8: Graphical User Interface (GUI) after sending a message and receiving by USER2 using CRSA

3.3.1 Pseudo code

Here the pseudo code for our proposed Chaotic Random Seed Algorithm (CRSA).

```
Random := function
    function Random(Seed)
        if (!Seed)
            Seed := this. Milliseconds * Math.floor( Math.random( ) * 50 ) + 20 )
        this.SeedArray := [ ]
        for i = 0 to 56
            this.SeedArray.push(0)
        num := (Seed == -2147483648) ? 2147483647 : Math.abs(Seed)
        num2 := 161803398 – num
        this.SeedArray[55] := num2
        num3 := 1
        for k = 0 to 55
            var num4 := 21 * k % 55
            this.SeedArray[num4] := num3
            num3 := num2 - num3
            if (num3 < 0)
                num3 += 2147483647
            num2 := this.SeedArray[num4]
        for  j = 1 to 5
            for k=1 to 56
                this.SeedArray[k] -= this.SeedArray[ 1+ (k+30) % 55 ]
                if (this.SeedArray[k] < 0)
                    this.SeedArray[k] += 2147483647
        this.inext = 0
        this.inextp = 21
        seed = 1;
Random.prototype.milliseconds = function
    str = date to String
    return parseInt(str.substr(str.length - 6))
Random.prototype.InternalSample = function
    num = this.inext
      num2 = this.inextp;
      if (++num >= 56)
         num = 1
      if (++num2 >= 56)
         num2 = 1
      num3 = this.SeedArray[num] - this.SeedArray[num2];
      if (num3 == 2147483647)
         num3--
      if (num3 < 0)
```



```
 num3 += 2147483647
        this.SeedArray[num] = num3
        this.inext = num
        this.inextp = num2
        return num3
Random.prototype.Sample = function
     return this.InterSample() * 4.6566128752457969E-10
Random.prototype.GetSampleForLargeRange = function
     num = this.InternalSample()
        flag = this.InternalSample() % 2 == 0
        if (flag)
           num = -num
        num2 = num
        num2 += 2147483646.0
        return num2 / 4294967293.0
Random.prototype.Next = function (minValue, maxValue)
        if (!minValue && !maxValue)
            return this.InternalSample();
        num = maxValue - minValue;
        if (num <= 2147483647)
           return parseInt((this.Sample() * num + minValue).toFixed(0))
        return this.GetSampleForLargeRange() * num + minValue
Random.prototype.NextDouble = function
        return this.Sample();
Random.prototype.NextBytes = function (buffer)
        for i = 0 to  buffer.length
           buffer[i] = this.InternalSample() % 256;
return Random;
r = new Random();
nextInt = r.Next(100000, 900000);
document.getElementById("randomNumber").innerHTML = nextInt;
```

Fig 3.3.9: Pseudo code for generating Random Seed.

```
cloud = setEncrypt: function(source,destination,passcode,random)
    document.getElementById(destination).innerText =
    this.encryptCodes(document.getElementById(source).value,
    document.getElementById(passcode).value,document.getElementById(random).value)
cloud = setDecrypt: function()
    document.getElementById('decryptedContent').innerText =
    this.decryptCodes(document.getElementById('originalContent').value,
    document.getElementById('passcode').value,document.getElementById('random').value);
  encryptCodes: function(content,passcode,random)
    result = []
    passLen = passcode.length
    otplen = random.length
    for i = 0  to content.length
      passOffset = i%passLen
      otpOffset = i%otplen
      calAscii = (content.charCodeAt(i)+passcode.charCodeAt(passOffset)+random.charCodeAt(otpOffset)+12);
      res = String.fromCharCode(calAscii)
      result.push(res)
    return JSON.stringify(result)
  decryptCodes: function(content,passcode,random)
    result = []
    str = ''
    codesArr = JSON.parse(content)
    passLen = passcode.length
    otplen = random.length
    for i = 0  to codesArr.length
        res = codesArr[i]
      res1 = res.charCodeAt(0)
      passOffset = i%passLen
      otpOffset = i%otplen
      calAscii = (res1-passcode.charCodeAt(passOffset)-random.charCodeAt(otpOffset)-12)
      result.push(calAscii)
    for i = 0 to result.length
      ch = String.fromCharCode(result[i])
      str += ch
    return str
```

Fig 3.3.10: Pseudo code for Encryption and Decryption.

3.4 Implementation requirements

For implementing this algorithm we used HTML5, CSS3, Bootstrap4 and JQuery for graphical user interface and JavaScript programming language for backend process. Our project is suitable for any kind of platform or computer device. It’s also responsive for good user’s interaction.

CHAPTER 4

EXPERIMENTAL RESULTS AND DISCUSSION

4.1 Introduction

When we crave to encrypt and decrypt some data, we demand some random number or string to carry through an operation based on a specific algorithm for alternation the substantive data. But how can we attain the random number? There are abundant way to trace a random number.

4.2 Experimental results

In the time of experiment we saw different types of result for different types of data set. We worked with text data file for encryption and decryption. Each encryption and decryption process successfully done by random seed. And the chipper text is totally unpredictable. We changed text data file size and random seed length and ran our algorithm for observe the impact on encryption and decryption process. The outcome of the experiment was successfully complete with real life web based application. Data communication process should be fast and secure. Our algorithm give fast output about execution time and give very strong protection of data.

4.3 Statistical analysis

For compare the performance with other algorithms, we have simulated in C++ language. Our proposed CRSC (Chaotic Random Seed Cryptography) showed different types of performance value with other algorithms. We measure encryption time, Decryption time, throughput, and memory utilization for each algorithms.

***Encryption Time*:** The encryption time is the time for an encryption algorithm to produce a chipper text from a normal plaintext. We measured it in milliseconds.

Decryption Time: The decryption time is the time for an algorithm to re-produce an original plaintext from its chipper text. We measured it in milliseconds.

***Throughput*:** In encryption and decryption process throughput means the speed of total encryption process. The equation of throughput-

$$Throughput = \frac{Tp(Kilobytes)}{Et(Second)}$$

$$where\ Tp = Total\ plain\ text\ (Kilobytes)$$
$$Et = Encryption\ Time(Second)$$

***Memory Utilization*:** Memory Utilization means how much memory is being consumed by encryption and decryption process. We measured it in kilobytes.

Description of simulation scenario table given below

Algorithms	**Test data**	**Performance**	**Configuration of Hardware**
DES (Data Encryption Standard)	Text File(2mb, 1mb, 50kb)	measure encryption time, Decryption time, throughput, and memory utilization	Intel(R) Core(TM) i3-4150 CPU @ 3.50GHz
AES (Advance Encryption Standard)			
CRSA (Chaotic Random Seed Algorithm)			

Table 1: Comparison test data and their performance for various algorithm

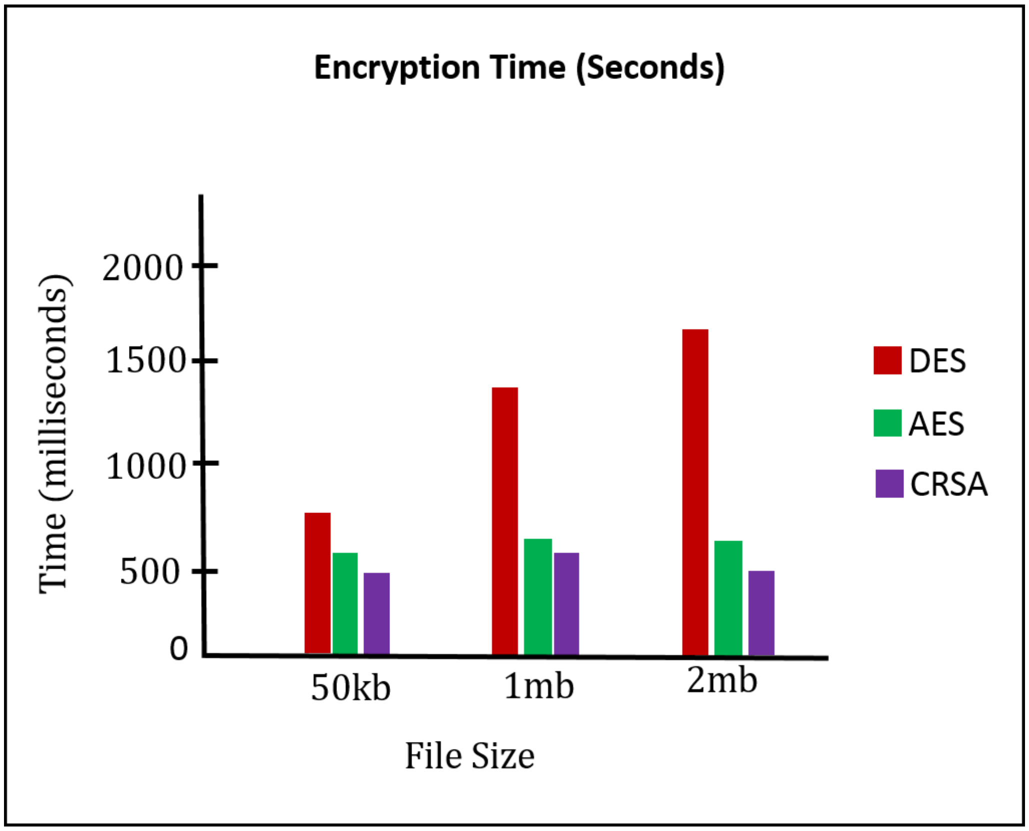


Figure 4.1.1: Encryption Time Comparison for different algorithms

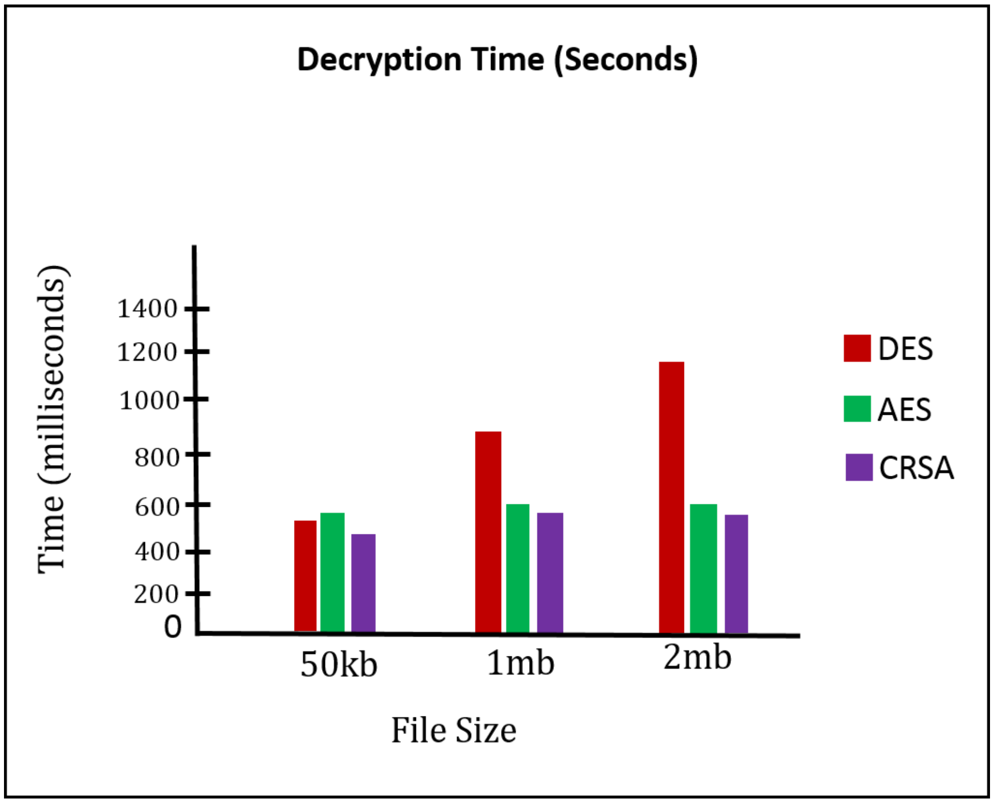


Figure 4.1.2: Decryption Time Comparison for different algorithms

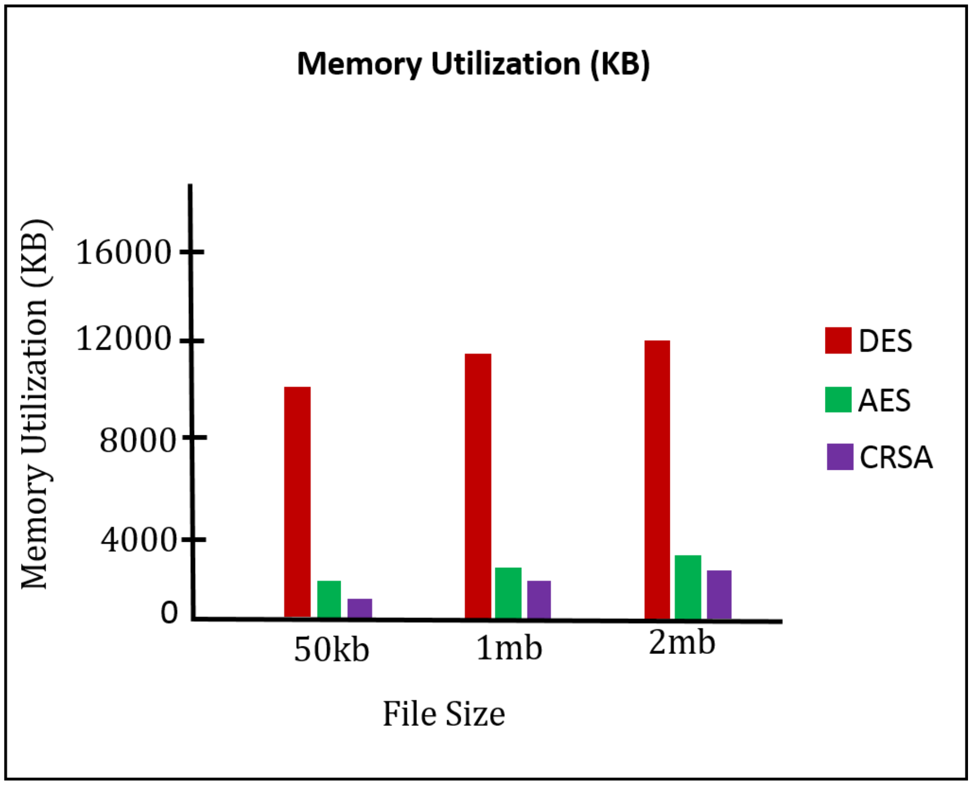


Figure4.1.3: Memory Utilization Comparison for different algorithms

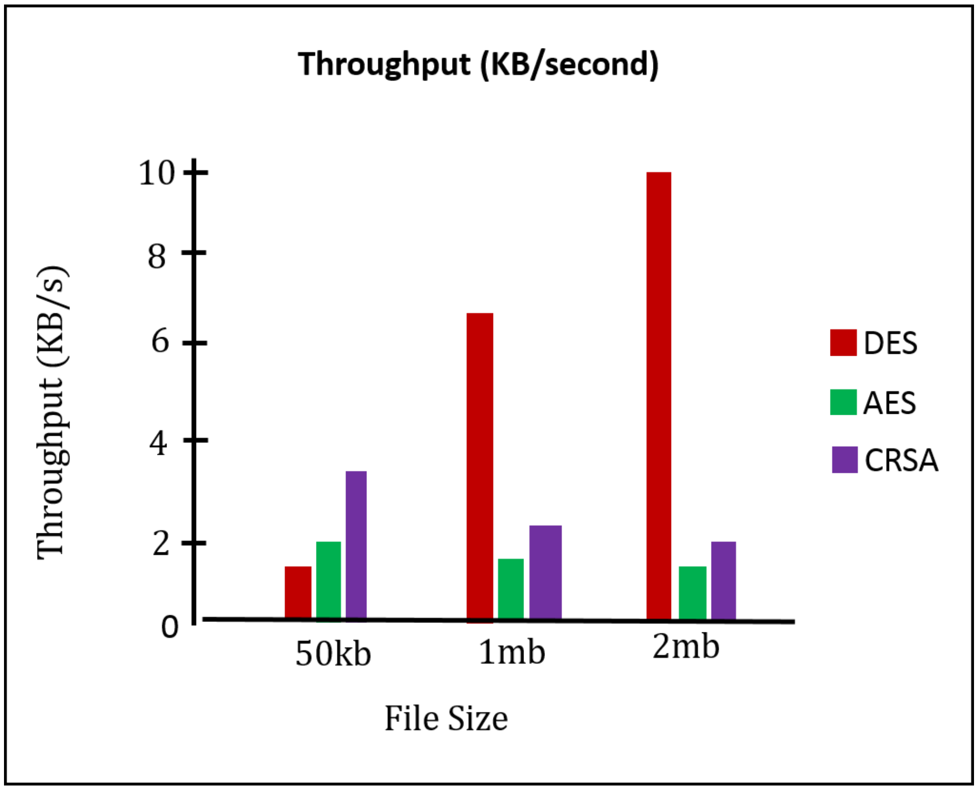


Figure 4.1.4: Throughput Comparison for different algorithms

4.4 Descriptive Analysis

After analyzing, the simulation shows that for various types of text sizes, for 50kb text file our algorithm take 98 millisecond less encryption time than AES and 254 millisecond less encryption time than DES algorithm. For 1mb text file our algorithm take 87 millisecond less encryption time than AES and 880 millisecond less encryption time than DES algorithm. For 2mb text file our algorithm take 102 millisecond less encryption time than AES and 1241 millisecond less encryption time than DES algorithm.

For 50kb text file our algorithm take 129 millisecond less decryption time than AES and 97 millisecond less decryption time than DES algorithm. For 1mb text file our algorithm take 20 millisecond less decryption time than AES and 318 millisecond less decryption time than DES algorithm. For 2mb text file our algorithm take 23 millisecond less encryption time than AES and 591 millisecond less decryption time than DES algorithm.

The throughput value for 50kb file for CRSA (3.5 KB/s) is more than DES (1.87 KB/s) and AES algorithm (2.3 KB/s). When we take 1Mb text file for comparison, it shows that, CRSA throughput (2.37 KB/s) is more than AES (1.85 KB/s) and less than DES algorithm (6.4 KB/s). For 2Mb file, it gives feedback like as less throughput value (2.2 KB/s) than (10 KB/s) DES algorithm and more than AES(1.8 KB/s) algorithm.

The memory utilization comparison show that, for 50kb file, the graph of CRSA is shorter than DES algorithm (9512 kb) and also shorter than (1053kb) AES algorithm. When we use 1 Mb size files, it takes 2235KB memory for CRSA algorithm 11500KB for DES and 3747KB for AES algorithm.

Overall, the statistics show that, our proposed CRSA algorithm is much better than DES and AES algorithm. Our securities are stronger than both of these comparing AES and DES algorithm. So we are optimistic with CRSA algorithm.

4.5 Experimental Results

In the time of experiment we saw different types of result for different types of data set. We worked with text data file for encryption and decryption. Each encryption and decryption process successfully done by random seed. And the chipper text is totally unpredictable. We changed text data file size and random seed length and ran our algorithm for observe the impact on encryption and decryption process. The outcome of the experiment was successfully complete with real life web based application. Data communication process should be fast and secure. Our algorithm give fast output about execution time and give very strong protection of data.

CHAPTER 5
SUMMARY, CONCLUSION, RECOMMENDATION AND IMPLICATION FOR FUTURE RESEARCH

5.1 Introduction

In this project, we have made a new algorithm for data security with encryption and decryption process. Security is a big issue for cloud computing and data communication. So our main goal was to improvement of security of data.

5.2 Summary

By analyzing the experiment by text data file with different size, we saw that CRSA is faster than other algorithms and the security of data is very strong. The security key or seed is totally random and not provided by anyone, so our experiment give successfully positive result with every test and data file size

5.3 Conclusions

In data encryption and decryption, security key or secret key is a very important thing for encryption and decryption of the data. Computer cannot generate a secret key itself. We have to put our own key for encryption and decryption. But there is a risk to be leaked the secret key from us. That why we need a very unique key from computer that is not predictable by hacker. So we have worked with millisecond in computer system and add different types of security key and produce a very unique key what is called random seed. Random seed is 100% random for every device and every time. We have given more importance on generating a random seed than encryption and decryption algorithm. Because if key was weak or predictable then the total security system is at risk. Our algorithm for generating random seed and encryption and decryption is take little more time than AES and similar algorithm, but the security of our algorithm is much more better than other algorithms.

5.4 Recommendations

We have worked with random seed generating process and encryption decryption process for only text file. Our experimental results show the comparison with other algorithms for text file. This is a limitation. But it possible to use random seed in other data types like image, audio etc. So in future we will use random seed in image and audio file encryption and decryption.

5.5 Implication for Further Study

In future we will work with image and audio data type for improvement of security. Our algorithm efficiency is already better than DES and almost similar with AES algorithm. In future we have plan to improve it and become more efficient than AES and also than RES algorithms. After that we have plan to implement it in cloud base security system. Now a days cloud computing is getting more popular day by day. Cloud computing is transforming information through internet. Many Corporate and academic institutions are investing in this technology as well as it changes working environment in Information technologies. So it’s very important to ensure safe transferring of data, safe data storage and safe data-ownership.

Reference:

[1]. Mr Anup Bhange, Dr. Harsh Mathur “Comparative analysis of several cryptography algorithm with its effectiveness towards the security and its performance” 2019 Journal of The Gujarat Research Society, vol. 21 ISSN: 0374-8588

[2]. Jay Alan Carlson, “Method for secure communication using asymmetric and symmetric encryption over insecure communication” 2019 Patent No: US 10,187,361 B2 available at https://patentimages.storage.googleapis.com/69/04/10/4d4cc7419b204c/US10187361.pdf last accessed on 23-11-19 at 10:37 PM.

[3]. Dr. Prema Mahajan, Abhishek Sachdeva “A study of encryption algorithms AES, DES, RSA for security” Vol: 13 Version: 1.0 2013 Global Journals. Inc(USA)

[4]. M. Kannan, Dr. C. Priya, S. VaishnaviSree. “ A comparative of DES, AES and RSA algorithms for network security in cloud computing” 2019 JETIR March 2019, Volume 6, Issue 3 www.jetir.org (ISSN-2349-5162)

[5]. Mohammed Nazeh Abdul Wahid, Abdulrahman Ali, Babak Esparham and Mohamed Marwan,” A Comparison of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish for Guessing Attacks Prevention” 2018 Journal of Computer Science Applications and Information Technology

[6]. Shaify Kansal, Meenakshi Mittal, “Performance evaluation of various symmetric encryption algorithms” 2014 International Conference on Parallel, Distributed and Grid Computing

[7]. Mark Steven DetrickJohn Edward FetkovichGeorge William Wilhelm, Jr.” Encryption/decryption of stored data using non-accessible, unique encryption key”

[8]. Learn about Wikipedia, available at https://en.wikipedia.org/wiki/Advanced_Encryption_Standard, last accessed on 24-11-2019 at 8.47 AM.

[9]. Learn about Wikipedia, available at https://searchsecurity.techtarget.com/definition/Data-Encryption-Standard, last accessed on 24-11-2019 at 9.51 AM.

[10]. Learn about Wikipedia, available at https://en.wikipedia.org/wiki/RSA_(cryptosystem) , last accessed on 24-11-2019 at 11..47 AM.