ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

Implementing Cryptography on the Concept of Returning Back Its Own Nest of a Bird

Anupam Mondal[1], Prof. Dr Pranam Paul[2]

MCA Final Year Student, Narula Institute of Technology, Agarpara, Kolkata, West Bengal, India[1]

HOD, Department of Computer Application, Narula Institute of Technology, Agarpara, Kolkata, West Bengal, India[2]

ABSTRACT-Cryptography is a Greek word that's means is Hidden Secret. In any communication, security is the most important issue in today's world. Lots of data security and data hiding algorithms have been developed in the last decade, which worked as motivation for the research. The scenario of present day of information security system includes confidentiality, authenticity, integrity, non-repudiation.

This present work focus is enlightening the technique to secure data or message with authenticity and integrity. With the growth of internet and network, the need for secure data transmission become more and more essential and important, as security is a major concern in the internet world. Data likely to be kept hide from all people except from the authorized user cannot be sent in plain text. So the plain text should be codified by the process of encryption.

Each type of data has its own features; therefore different techniques should be used to protect confidential data from unauthorized access. Here we introduced a new algorithm which is based on simple mathematical operation. In this algorithm encryption is done on binary file so it can be applicable for any type of data such a text as well as multimedia data. Here the same idea of cryptography is working (i.e. using key, conversion of plain text into cipher text called encryption and the reverse, means cipher text to plain text called decryption).

KEYWORDS: Cipher Text, Cryptography, Encryption, Decryption, Plain Text, Symmetric Key.

I. INTRODUCTION

The rapid growth of computer networks allowed larger files, such as digital image, text to be easily transmitted over the internet. Data encryption is widely used to ensure security of those data. Here we introduce a Block based symmetric key encryption algorithm. For encryption a key is to be generated. Key length and bit stream is chosen at random. At first we take block size from key and get all possible number under the bit. The all number will module with number/s which is/are consists in key. Subtract the module result each other's and arrange the numbers in ascending order and get the original numbers. Indexing the numbers and traverse with a number of time that's consists in key. That's our encrypted format. In decryption we traverse all number until the number back again and count the traverse numbers. Find the LCM of all traverse number and subtract it from encrypted traverse number and the number of time we traverse it and get the original file.

In section **III**, algorithm is defined. While section **IV** shows the example of whole process. An analysis has been done in section **V**, along with conclusion in **VI**.

II. RELATED WORK

In [18] the author used perfect square number to calculate the difference between two numbers and calculated the number of bits required to represent them. In [17] the author emphasized on division method where how many times division method will be applied is calculated. In [7] author used primer number from where basic concept of this algorithm is obtained. Each author has shown different ways of strengthening security to data. . In this algorithm encryption and decryption process are performed on binary data. All data which is under stable by the computer is finally converted into binary bits. So it can be implemented for any data type encryption process. Therefore that encryption technique can be used for text encryption, image encryption etc.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

III. ALGORITHM

3.1. Key Structure:-

The Key used to be in Encryption and Decryption process can be choose any integer randomly. IT is content block size, how many numbers we want to module and the module numbers, how many numbers of times we want to traverse.

3.2. Encryption:-

Step 1: We can fetch block size from key file and we get from 0 to 2^block size numbers and store them (example- suppose block size is 4 then we get 0 to 2^4=16 number).
Step 2: We fetch how many module number and what are the numbers.
Example-how many number are 2.Numbers are 4 and 5.
Step 3: We module 0 to 2^block size each number by the module number which we fetch from key file.
Example-0%2=0, 1%2=1,
Step 4: Then we subtract the module result each other. Example: 0-1=-1
Step 5: We arrange the 0 to 2^block size number according to ascending order of subtract result. Example-suppose we consider block size 2 and we get 0,1,2,3 number from the bit size.
Step 6: Then we are indexing the new arranged number. Example:-

0	1	2	3
3	0	2	1

Step 7: We fetch ASCII value from a file which will be encrypted and convert the ASCII value to binary number and store it to file1.
Step 8: We block size number of bit from file 1 and convert into decimal value. The decimal values will the index number of arranged numbers and this index there is a value and the value will the next index. This is one round and we continue the round number which we fetch the number from key file.
Example-:- Supposes the series is 3, 0, 2, 1.and indexing it that's 0(3), 1(0), 2(2), 3(1). Traversing 0→1→0→3.
Step 9: We convert the number to binary number with block size number bit size and store into file4.
Step 10: The file 4 is the cipher text or encrypted file.

3.3. Decryption:-

Step 1: We can fetch block size from key file and we get from 0 to 2^block size numbers and store them.
Example-suppose block size is 4 then we get 0 to 2^4=16 number.
Step 2: We fetch how many module number and what are the numbers.
Example- how many number are 2.Numbers are 4 and 5.
Step 3: We module 0 to 2^block size each number by the module number which we fetch from key file.
Example-0%2=0, 1%2=1
Step 4: Then we subtract the module result each other. Example-0-1=-1
Step 5: We arrange the 0 to 2^block size number according to ascending order of subtract result.
Example- supposes we consider block size 2 and we get 0,1,2,3, number from the bit size. Suppose, after the operation we get the series- 3, 0, 2, 1.
Step 6: Then we are indexing the new arranged number. Example-

0	1	2	3
3	0	2	1

Step 7: We go to 1st index number and that number will replace by corresponding index's value that is one traversing process and the value is the next index and the process is continue until the 1st indexing number is not return and we store how many process is require to find the number again and store the number.
Example: - Supposes the series is 3, 0, 2, 1.and indexing it that's 0(3), 1(0), 2(2), 3(1). Traversing 0→1→0→3.
Step 8: Then we find the LCM of stored number (which is the counting of process required).
Example:-suppose there are 2, 5, 10.The LCM are 10.
Step 9: We subtract the LCM number from the terminate value from the key file.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

Step 10: We fetch block size number of bit stream from encrypted binary bit stream and convert the numbers to decimal number.
Step 11: The decimal values will the index number of arranged numbers and this index there is a value and the value will the next index. This is one round and we continue the round number which is equal to subtraction of LCM and terminate number.
Step 12: The numbers will convert into binary stream with block size number bits.
Step 13: We convert the binary stream with 8bit decimal converter and that's the ASCII value of the original text.

IV. EXAMPLE

4.1. *Key Structure:-*

Supposes in key file block size-4,2 numbers we want to module and they are 2,3. The encrypted traversing number is 6.

4.2. Encryption:-

Step 1: In 4 bit block size we have 0-15 maximum number we can represent and store in array.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Step 2: We fetch the module number from key file that's 2, 3 and module the numbers, shown in Table 4.2.1

Table 4.2.1

No.	Mod2	Mod3	No.	Mod2	Mod3	No.	Mod2	Mod3	No.	Mod2	Mod3
0	0	0	4	0	1	8	0	2	12	0	0
1	1	1	5	1	2	9	1	0	13	1	1
2	0	2	6	0	0	10	0	1	14	0	2
3	1	0	7	1	1	11	1	2	15	1	0

Step 3: Subtract the module result from each other, shown Table 4.2.2.

Table 4.2.2

No.	Mod2	Mod3	Sub	No.	Mod2	Mod3	Sub	No.	Mod2	Mod3	Sub	No.	Mod2	Mod3	Sub
0	0	0	0	4	0	1	-1	8	0	2	-2	12	0	0	0
1	1	1	0	5	1	2	-1	9	1	0	1	13	1	1	0
2	0	2	-2	6	0	0	0	10	0	1	-1	14	0	2	-2
3	1	0	1	7	1	1	0	11	1	2	-1	15	1	0	1

Step 4: Arrange the number according to subtract value and indexing it, shown in table 4.2.3.

Table 4.2.3

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
No's	2	8	14	4	5	10	11	0	1	6	7	12	13	3	9	15

Step 5: Suppose in a text file content is NIT and ASCII value of each characters are.

N→78 I→73 T→84

Step 6: Convert the ASCII value of character to 8bit binary number.

78→01001110 73→01001001 84→01010100

Then the bit streams are 010011100100100101010100 and save to file 1.

Step 7: Pick 4 bit (Block size) number from file 1 and convert in to binary number and these numbers are the index of the arrange numbers and next the index value is the next index that is one traversing. These traversing number is equal to 6(6 is the traversing number from key file).

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

0100→	4→	10→	12→	7→	1→	8→	0
1110→	14→	3→	4→	10→	12→	7→	1
0100→	4→	10→	12→	7→	1→	8→	0
1001→	9→	6→	5→	11→	13→	9→	6
0101→	5→	11→	13→	9→	6→	5→	11
0100→	4→	10→	12→	7→	1→	8→	0

Step 8: Convert the decimal number to 4bit binary numbers and store in File 2.That is the encrypted binary stream.
File2, 000000010000011010110000.
Step 9: Convert the 8bit binary to decimal value that is the ASCII value of encrypted Character and store in Encry.txt file. Content of encrypted chars are "°+"

4.3. Decryption:-
STEP:-1 In 4 bit block size we have 0-15 maximum number we can represent and store in array.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Step 2: We fetch the module number from key file that's 2, 3 and module the numbers, that is same as table 4.2.1
Step 3: Subtract the module result from each other, that is same as 4.2.2.
Step 4: Arrange the number according to subtract value and indexing it, that is same as 4.2.3.
Step 5: We traverse all the arrange number until the number get back and count how many numbers of traversing we back the same number and store it in a array.

10	10	10	10	10	5	5	10	10	5	10	5	10	5	10	1

Step 6: Find the LCM of all traversing number**.** The value is 10.
Step 7: Pick the ASCII value of each character from encrypted file and convert into 8 bit binary number. The stream is-
000000010000011010110000
Step 8: Pick 4 bit binary number and convert into decimal value. These numbers are the index of the arrange numbers and next the index value is the next index that is one traversing. The traversing number is equal to LCM number – Traversing number in key file. 10 – 6 = 4

0000→	0→	2→	14→	3→	4
0001→	1→	8→	0→	2→	14
0000→	0→	2→	14→	3→	4
0110→	6→	5→	11→	13→	9
1011→	11→	13→	9→	6→	5
0000→	0→	2→	14→	3→	4

Step 9: Convert the number into 4bit binary number and store in file5. **i.e.** 010011100100100101010100.

V. RESULT ANALYSIS

5.1. Choose key:-As before we say the Key used to be in Encryption and Decryption process can be choose any integer randomly. But some integers we cannot be choose, and that is a limitation of this algorithm. Because if we choose 4 as our key then 2^4=16 and also 4^2=16.

5.2. Size and Time Comparative Report:-We analysis the plain text file size, total time for encryption to create encrypted file and in the same way the total time for decryption and the encrypted file size to create a decrypted file. In this section we compare file size with the time for clear observation.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

Table: 5.1
Comparison of original file and time of encryption

Original File Size (Byte)	Encrypted File Size (byte)	Encryption time (Sec.)	Encryption Time/Byte
348	345	0.164835	4.73663791e-4
578	574	0.274725	4.78619826e-4
232	230	0.109890	4.73663793e-4
1259.76	1259.52	0.549451	4.36155299e-4

Table.5.1 shows time taken for encryption for different file size (with fixed key)i.e. Original file size and time taken for encryption for each byte, encrypted file size. From the above table data we draw two following figures.

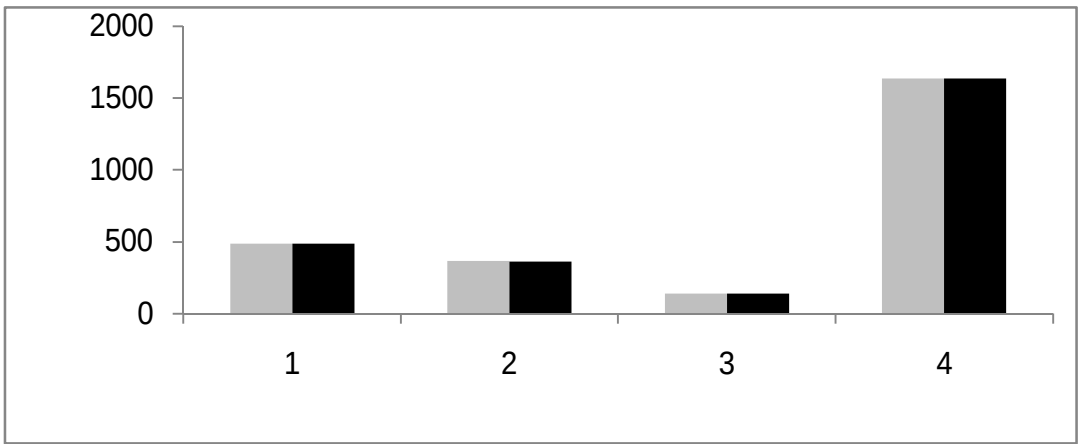

Fig: 5.1
Figure of original file size and Encryption file size.
Black-Encrypted file size. Grey- Original file size.

In Fig:-5.1 we notice that the original file size and encryption file size are almost same. So we said that after encryption there are no extra bits add and file size both are almost same.

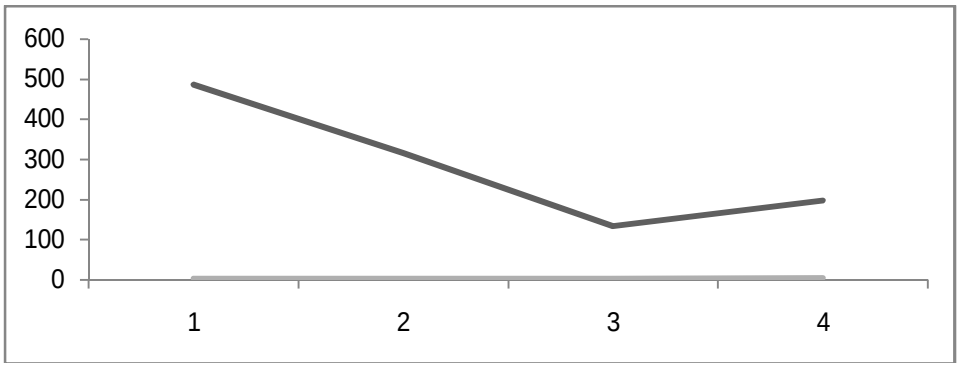

Fig: 5.2
Figure of original file size and encryption time/byte
Black: - Original File size. Grey: - Encryption Time.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

In Fig: 5.2 we have calculated the encryption time taken/ Encrypted file size and show that the all values are almost same. So we can decide that content a file does not effect on encryption time.

TABLE -5.2
Size and Time Comparative Table of decryption

Original File Size (byte)	Decrypted File Size (Byte)	Decrypted Time (Sec.)	Decryption Time/Byte
487	487	0.16483516	3.384705544147844e-4
364	364	0.10989011	3.035638397790055e-4
137	137	0.05494505	4.010587591240876e-4
1638.4	1638.4	0.27472527	1.676789978027344e-4

In Table 5.2 shows time taken for decryption for different file size (with fixed key value) i.e. Original file size and time taken for decryption for each byte, decrypted file size. From the above table data we generate the following figure.

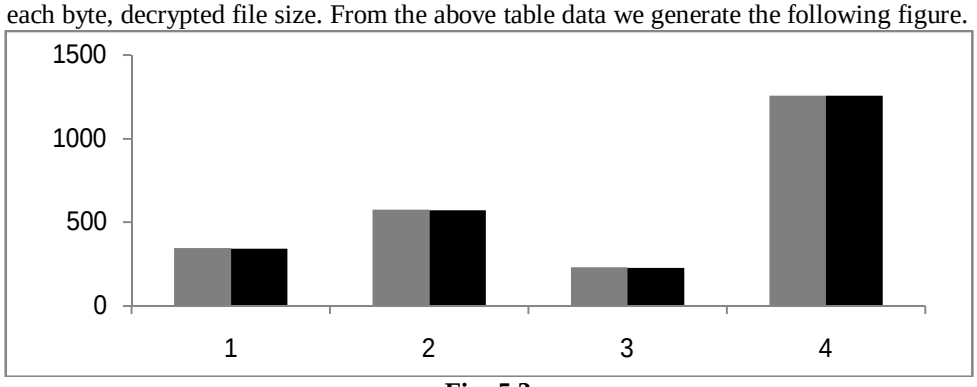

Fig: 5.3
Figure of encrypted file size and decryption file size
Grey-Original File Black-Decrypted File size

In fig:-5.3. We notice that we back the same file size in decryption that is equal to original file size. We notice that the original file and decrypted file size same and also said that the contents of original file and decrypted file are same.

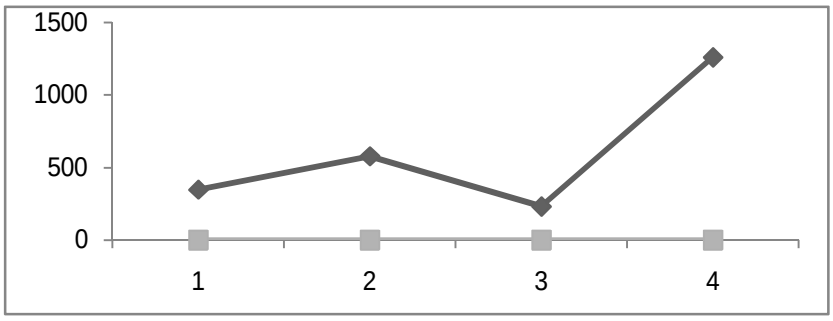

Fig: 5.4
Figure of encrypted file size and decryption time/byte.
Black: - Original File Size. Grey: - Decryption time.

In fig: 5.4 we calculate the decryption time taken/ Decrypted file size and show that the all values are almost same. So we can decide that the file content does not effect on decryption time.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

5.3. Security: -If the size of key is n, then the possibility of number of key generation is 2^n.Therefore possibility of choosing the correct key among 2^n number is $2^{(n-1)}$ possible numbers. From the above observation we can say that if the size of the key is increased, then the probability of choose the correct key is also increasing exponentially.

VI. CONCLUSION

My conclusion towards this algorithm is that I have tested the implementation of this algorithm and this algorithm worked correctly for the above set of values. From this we can assume that algorithm can correctly be implemented for various type and size of file. It will be secured.

REFERENCES

[1] William Stallings, "Cryptography and network security principles and practices", 4th edition, Pearson Education, Inc. publishing as Prentice Hal, 2006.
[2] Pranam Paul, Saurabh Dutta, A K Bhattacharjee, "An Approach to ensure Security through Bit-level Encryption with Possible Lossless Compression", International Journal of Computer Science and Network Security", Vol. 08, No. 2, pp.291 – 299, 2008.
[3] Sanjit Mazumdar, Sujay Dasgupta, Prof.(Dr) Pranam Paul, "Implementation of Block based Encryption at Bit-Level", International journal of Computer Science and Network Security, Vol. 11, No.2, pp. 18-23, 2011.
[4] Sujay Dasgupta, Sanjit Mazumdar, Prof.(Dr) Pranam Paul, "Implementation of Information Security based on Common Division", International journal of Computer Science and Network Security, Vol. 11, No.2,pp. 51-53, 2011.
[5] http://en.wikipedia.org/wiki/Symmetric-key_algorithm
[6] Asoke Nath, Saima Ghosh, Meheboob Alam Mallik, "Symmetric Key Cryptography using Random key Generator", Proceeding of International conference on security and management (SAM'10" held at Las Vegas, USA Jull 12-15,2010), P-Vol-2, pp. 239-244,2010.
[7] Pranam Paul, Saurabh Dutta, "An Enhancement of Information Security using Substitution of Bits Through Prime Detection in Blocks", Proceeding of National Conference on Recent Trends in information Systems(ReTIS-06), Organized by IEEE Gold Affinity Group, IEEE Calcutta Section,Computer Science & Engineering Department, CMATER &SRUVM Project-Jadavpur University and Computer Jagat.
[8] Oded Goldreich, "Foundation of Cryptography (A primer)", July 2004.
[9] Bruce Schneier, "Applied Cryptography", ISBN 0-471-12845-7
[10] John Talbot, Dominic Welsh; "Complexity and Cryptography An introduction". ISBN-10: 0521852315
[11] Denise Sutherland, Mark Koltko-Rivera "Cracking Codes and Cryptograms For Dummies"; ISBN: 978-0-470-59100-0;October 2009
[12] Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone "Handbook of Applied Cryptography"; CRC Press; ISBN: 0-8493-8523-7
[13] WILLIAM 'F. FRIEDMAN; "MILITARY CRYPTANALYSIS, Part I, MONOALPHABETIC SUBSTITUTION SYSTEMS"
[14] Henk C.A. van Tilborg, Sushil Jajodia; "Encyclopedia of Cryptography and Security", 2nd edition; 2011; ISBN: 144195905X
[15] Wenbo Mao; "Modern Cryptograph"..
[16] Wels Chenbach; "Cryptography in C and C++".
[17]Ayan Banrjee, Prof. Dr.Pranam Paul, "Bock Based Encryption and Decryption", International journal of Computer Science and Network Security, ISSN: 0974 – 9616 vol-7,No.2,2015.
[18]Shibaranjan Bhattacharyya, Prof. Dr.Pranam Paul, "An Approach to Block Ciphering using Root of Perfect Square Number", International journal of Computer Science and Network Security,ISSN: 0974 – 9616 vol-7,No.2,2015.

BIOGRAPHY

Anupam Mondal, he is a student of MCA from Narula Institute of Technology and former student of BCA from B.P.Poddar Institute of Management & Technology under WBUT.

Dr Pranam Paul, ***Assistant Professor and Departmental Head, CA Department, Narula Institute of Technology (NIT), Agarpara*** had completed MCA in 2005. Then his carrier had been started as an academician from MCKV Institute of Technology, Liluah. Parallel, At the same time, he continued his research work. At October, 2006, National Institute of Technology (NIT), Durgapur had agreed to enroll his name as a registered Ph.D. scholar. Then he had joined Bengal College of Engineering and Technology, Durgapur. After that Dr. B. C. Roy Engineering College hired him in the MCA department at 2007. At the age of 30, he had got Ph.D. from National Institute of Technology, Durgapur, and West Bengal. He had submitted his Ph.D. thesis only within 2 Years and 5 Months. After completing the Ph.D., he had joined Narula Institute of Technology in Computer Application Department. Parallel he continues his research work. For that, he has 39 International Journal Publications among 54 accepted papers in different areas. He also reviewer of International Journal of Network Security (IJNS), Taiwan and International Journal of Computer Science Issue (IJCSI); Republic of Mauritius**.**