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 5, May 2016

Developing a Cryptography Protocol Depending on the Concept of Playing Football

Nabendu Paul[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: In recent days, for secure information transmission through internet, Cryptography is used. Here for secure data communication the plain text would be encrypted into cipher text using encryption process. This encrypted text along with the key or information would be send by the sender at receiver s end. Then using the key or information, the receiver would able to decrypt the encrypted text. Using this base idea there exist different algorithm for encryption and decryption using key. Here we are implementing the concept of playing football to encrypt the plain text. A player passes the ball with some force to a player having a unique number. The strength of the technique is analyzed in this algorithm. This is a block based private key cryptographic technique. Here the block of plaintext is converted the journey of ball for a particular team during the encryption. The process is later discussed in details in this algorithm.

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

I. INTRODUCTION

Cryptography is a concept of data conversion for securing the data from the unauthorized access with an algorithm which is open to all. That means we need the conversion of data which is nothing but an encrypted text. Here we are implementing the concept of playing football to encrypt the plain text. A player passes the ball with some force to a player having a unique number. Like This way this passing football will be continued until and unless ball will be reached to the opponent s goal or players. Which are respectively treated here as 0 and 1.Both the cases playing of these particular team is temporally stopped. Here the block of plain text is converted the journey of ball for a particular team during the encryption. Here we treated plain text as a ball and a divisor as a player initially who passes the ball with some quotient value as a force to the player noted as remainder value. Again this remainder valued player pass the quotient as a ball to another player.

II. RELATED WORK

In [23] the author used perfect square number to calculate the difference between two numbers and calculated the number of bits required to represent them. In [22] the author emphasized on division method where how many times division method will be applied is calculated. In [9] 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.

III. PROPOSED ALGORITHM

In this section, Key generation is discussed in section 2.1. In the section 2.2 and 2.3 discussed about the encryption process and decryption process respectively.

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 5, May 2016

2.1. Key Generation

The Key used to be in Encryption and Decryption process can be choose any integer randomly. We take two numbers for block size and divisor which are also kept into the two segment of the key respectively.

2.2. Encryption Process

Step 1: Convert the plain text into its binary form and we get source bit stream.

Step 2: Decompose source bit stream into some blocks with given block size which is already kept into the first segment of key.

Step 3: We just take the first block and the decimal value of that block is divided by the divisor which is already kept into the second segment of key.

Step 4: After division the coming remainder value will be represented by the number of bits such that maximum possible remainder in this step can be represented. Represented bit stream format of the remainder will be appended in to the resent target bit stream.

Step 5: In the next step the remainder and quotient of the previous step will be treated as a divisor and dividend respectively and continue Step 4 and Step 5 until and unless the remainder will be 0 or 1.

Step 6: Now represent the quotient in this step with such number of bit stream through which the maximum possible quotient in this step can be represented in the binary form. Represented bit stream format of the quotient will be appended into the resent target bit stream.

Step 7: The same process, Step 3 to 5, will be continued for the next block until and unless all blocks are processed. At the end the final target or encrypted bit stream will be gotten.

Step 8: There may be some bits in source bit stream are not involved directly into the encryption process, number of those bits should be less than block size. These unused bits are also kept into the third segment of key.

Step 9: Now generated encrypted bit stream is converted into the cipher text. During this conversion process if some bits are left, these bits are kept into the fourth segment of key.

2.3. Decryption Process

Step 1: Convert the cipher text into its binary form and we get bit stream.

Step 2: Now we have to append those bits which are already kept into the fourth segment of the key.

Step 3: We take number of bits in such a manner where maximum possible remainder of the division in this step can be represented. This step will be continued until and unless the decimal value of the taken bits is 0 or 1.

Step 4: Now we take such number of bit stream through which the maximum possible quotient in this step can be represented. During reaching in this step we have to keep all the values of the intermediate step define in step 3 and 4.

Step 5: Now we get some values and the last value should be quotient value and 2^{nd} last is remainder value and 3^{rd} last is divisor in this step but it is also a remainder value of the previous step.
By calculating these 3 values we get dividend value of this step as well as the quotient value of the previous step during the encryption. This step will be continuing till, when we did not get the first remainder and quotient value of the encryption. Now we take the divisor which is already kept into the second segment of key. By calculating these last 3 values we get final dividend which is the decrypted value of this block.

Step 6: Now convert this value into its binary form with block size which is already kept into the first segment of key. This is the decrypted bit stream of that encrypted block.

Step 7: The same process, Step 3 to 6, will be continued until and unless we reach end of the encrypted bit stream. Generated decrypted bit stream of a block of each iteration will be appended sequentially to generate entire decrypted bit stream.

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 5, May 2016

Step 8: Now we have to append those bits which are already kept into the third segment of the key, at the last of the recently generated decrypted bit stream.

Step 9: Now generated decrypted bit stream is converted into the text, which is the decrypted text that is similar to the plaintext.

IV. EXAMPLES

3.1. Key Generation

Our algorithm is based on private key operation. We can choose any number as key.
For examples here we choose our block size as 7 and key as 12.

3.2. Encryption Process

Consider the text “ENCRYPTION “as plain text.

Step 1: Fist each character of the plain text is converted into its corresponding ASCII value.
E → 69
N → 78
C → 67
R → 82
Y → 89
P → 80
T → 84
I → 73
O → 79
N → 78
Now each ASCII value converted into its binary form of 8 numbers of bits. And we get a binary stream for the plain text as below—
01000101 01001110 01000011 01010010 01011001 01010000 01010100 01001001 01001111 01001110

Step 2: Decompose source bit stream into some blocks with given block size, here that is 7, which is already kept into the first segment of key.

Step 3: Now we take the first block that is-
0100010 → 34
This decimal value is divided by the divisor, which is 12 and it s already kept into the second segment of key.

Step 4: So, the divisor is 12 and dividend is 34 and the remainder comes 10 and the quotient is 2. This 10 remainder is represented as a 4 bits binary form because maximum possible remainder in this step can be represented by 4 bits.
10 → 1010

Step 5: Now, divisor = 10, dividend = 2 so, the remainder comes 2 and the quotient is 0. Now, remainder 2 is represented as a 4 bits binary form.
2 → 0010
Again repeat the same but now,
Divisor = 2, dividend = 0 so, the remainder comes 0 and the quotient is 0. Now, remainder 0 is represented as a 1 bit binary form.
0 → 0

Step 6: Here we stop the division process because the remainder comes 0. Now represent the quotient in this step that is 0, with 1 bit binary form, through which the maximum possible quotient in this step can be represented.
0 → 0

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 5, May 2016

Represented bit stream is appended in to the resent target bit stream as below-
1010001000

Step 7: At the end the final target or encrypted bit stream will be gotten as below-
1010001000101101100000000000110010110000001100010001010110101000010000000011100000011101101100000100110000000

Step 8: There some bits in source bit stream are not involved directly into the encryption process, these are-
110
These unused bits are also kept into the third segment of key.

Step 9: Now decompose the generated encrypted bit stream into 8 numbers of bits and then convert to its corresponding decimal value as below-

10100010 → 162
00101101 → 45
10000000 → 128
00011001 → 25
01100000 → 96
01100010 → 98
00101011 → 43
01010000 → 80
10000000 → 128
01110000 → 112
00111011 → 59
01100000 → 96
10011000 → 152
During this conversion process if some bits are left, these are-
0000
These bits are kept into the fourth segment of key.
Each decimal value is now converted to its corresponding ASCII character and produces the cipher text against that plain text.

PLAIN TEXT → ENCRYPTION
CIPHER TEXT → ¢-€├ `b+P€p;`˜

3.3. Decryption Process
Consider the text "¢-€├ `b+P€p;`˜" as cipher text.

Step 1: Fist each character of the cipher text is converted into its corresponding ASCII value.
¢ → 162
- → 45
€ →128
├ → 25
` → 96
b → 98
+ → 43
P → 80
€ → 128
p → 112
; → 59

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 5, May 2016

` → 96
˜ → 152
Now each ASCII value converted into its binary form of 8 numbers of bits. And we get a binary stream for the cipher text as below—
10100010 00101101 10000000 00011001 01100000 01100010 00101011 01010000 10000000 01110000 00111011 01100000 10011000

Step 2: Now we have to append those bits which are already kept into the fourth segment of the key, that is-
0000

Step 3: At first we have to take the divisor, which is 12 and it s already kept into the second segment of key.
Now we take 4 numbers of bits through which maximum possible remainder, 11 of the division in this step can be represented as below-
1010 → 10
Now, divisor = 10
Again we take 4 numbers of bits as below-
0010 → 2
Now, divisor = 2
Again we take 1 number of bits as below-
0 → 0
Here we stop taking bits because the remainder comes 0.

Step 4: Now we take 1 bit through which the maximum possible quotient in this step can be represented as below-
0 → 0
We have to keep all the decimal values of the intermediate step define in step 3 and 4 those are-
10 2 0 0

Step 5: Now we should know that the last value should be quotient value and 2nd last is remainder value and 3rd last is divisor in this step so,
0 * 2 + 0 = 0 is a dividend value of this step as well as the quotient value of the previous step during the encryption.
Again we calculate between 10 2 0 so,
0 * 10 + 2 = 2
Now we have 10 and 2 those are the first remainder and quotient value of the encryption. Now we take the divisor which is 12 and it is already kept into the second segment of key. Now we calculate between 12 10 2, so
12 * 2 + 10 = 34
Now we get final dividend which is the decrypted value of this block.

Step 6: Now convert this value, which is 34 into its binary form with block size which is 7 and it is already kept into the first segment of key. So the bit stream is-
34 → 0100010

Step 7: Generated decrypted bit stream of a block, during each iteration will be appended sequentially to generate entire decrypted bit stream as below-
0100010101001110010000110101001001011001010100000101010001001001010011110100111 0

Step 8: Now we have to append those bits which are already kept into the third segment of the key, those are-
110
at the last of the recently generated decrypted bit stream.

Step 9: Now decompose the generated decrypted bit stream into 8 numbers of bits and then convert to its corresponding decimal value as below-
01000101 → 69
01001110 → 78

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 5, May 2016

01000011 → 67
01010010 → 82
01011001 → 89
01010000 → 80
01010100 → 84
01001001 → 73
01001111 → 79
01001110 → 78

Each decimal value is now converted to its corresponding ASCII character and produces the decrypted text that is similar to the plaintext.

CIPHER TEXT → ¢-€ ├ `b+P€p;`˜
PLAIN TEXT → ENCRYPTION

V. SIMULATION RESULTS

In this algorithm encryption is perform 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 kind of file. Therefore that encryption technique can be used for text encryption, image encryption i.e., multimedia encryption process etc. In this algorithm we cannot predict the encrypted block size because it s totally depends on division process and the first divisor value of the key. On the other side in the decryption process we cannot predict how many bits we take first and where we stop for a particular value. This kind thing makes this algorithm very strong.

Let's concentrate about keys. Our first segment of the key is block size. If we consider **i** as number of digits of block size, then we have $\mathbf{9*10^{i-1}}$ number of combination of block size value. Where $\mathbf{9*10^{i-1} -1}$ number of values are wrong. Our second segment of the key is divisor. If we consider **j** as number of digits of divisor, then we have $\mathbf{9*10^{j-1}}$ number of combination of divisor value. Where $\mathbf{9*10^{j-1} -1}$ number of values are wrong and $\sum\mathbf{(9*10^{j-1} -1)}$ number of possible reindeer can be generated. Our third segment is stored unused bits which should be less than block size. So, $\mathbf{(2 \wedge (9*10^{i-1} -1)) -1}$ number of combination of bit stream can be generated. Our fourth segment is also stored unused bits which should be less than 8. So, $\mathbf{2^7}$ number of combination of bit stream can be generated.

So, we have $\mathbf{(9*10^{i-1} + 9*10^{j-1} + 2\wedge(9*10^{i-1} -1) + 2^7) - 1}$**,** this number of combination can be generated for each segment and it s too hard to find, for that reason it makes this algorithm too strong.

4.1 Size and Time Comparative Report

This algorithm has been implemented on number of data files varying types of content and sizes of wide range, shown in

Table: 4.1
Size and Time Comparative Table of encryption

SL. No.	Original File Size(byte)	Encrypted File Size(byte)	Encryption Time	Encryption Time /byte
1	10	13	0	0
2	21	29	0	0
3	65	87	0	0
4	131	176	0.054945	0.000312188
5	219	292	0.10989	0.000376336

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 5, May 2016

6	309	418	0.164835	0.000394342
7	619	833	0.274725	0.000329802
8	840	1,092	0.494505	0.000452843
9	3802	4672	1.373626	0.000294012
10	2342	2878	0.494505	0.000171822

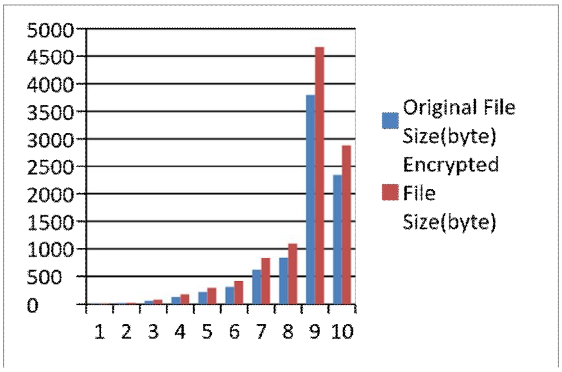


Fig: 4.1 of original file size and Encrypted file size.

Figure is shown that the compare between Original File Size as a blue bar and Encrypted File Size as a red bar.

Table: 4.1 shows time, taken for encryption for different file size i.e. Original file size and time taken for encryption for each byte and encrypted file size. From the above table data we draw two following figures.

Table: 4.2
Size and Time Comparative Table of decryption

SL. No.	Encrypted File Size(byte)	Decrypted File Size(byte)	Decryption Time	Decryption Time /byte
1	13	10	0	0
2	29	21	0	0
3	87	65	0.054945	0.000845308
4	176	131	0.10989	0.000838855
5	292	219	0.10989	0.000501781
6	418	309	0.265845	0.00086034
7	833	619	0.32967	0.000532585
8	1092	840	0.494505	0.000588696
9	4672	3802	2.857143	0.000751484
10	2878	2342	1.153846	0.000492675

Table: 4.2 shows time, taken for decryption for different file size i.e. Encrypted file size and time taken for decryption for each byte and decrypted file size. From the above table data we draw two following figures.

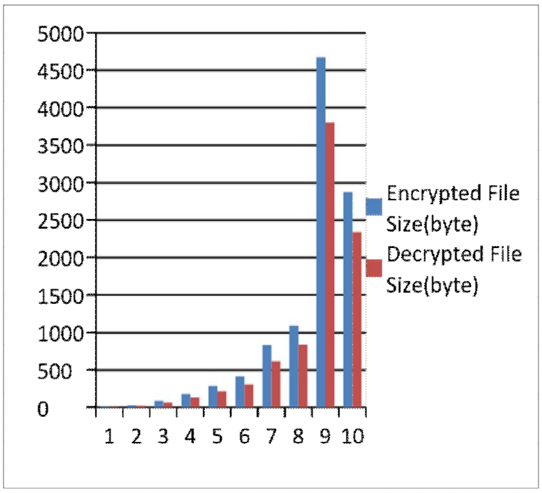


Fig: 4.2 of Encrypted file size and Decrypted file size.

Figure is shown that the compare between Encrypted File Size as a blue bar and Decrypted File Size as a red bar.

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 5, May 2016

VI. CONCLUSION AND FUTURE WORK

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] A. Kahate,"Cryptography and Network Security", (2nd ed.). New Delhi: Tata McGraw Hill, 2008.
[2] William Stallings, "Cryptography and network security principles and practices", 4th edition, Pearson Education, Inc. publishing as Prentice Hal, 2006.
[3]India2 Zirra Peter Buba, Gregory Maksha Wajiga– "Cryptographic Algorithms for Secure Data Communication" ,International Journal of Computer Science and Security, Vol. 5, Issue 2, 2011.
[4] 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.
[5] 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.
[6] 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.
[7]http://en.wikipedia.org/wiki/Symmetric-key_algorithm
[8] 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.
[9] 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.
[10] Oded Goldreich, "Foundation of Cryptography (A primer)", July 2004.
[11] Bruce Schneier, "Applied Cryptography", ISBN 0-471-12845-7
[12] John Talbot, Dominic Welsh; "Complexity and Cryptography An introduction". ISBN-10: 0521852315
[13] Denise Sutherland, Mark Koltko-Rivera "Cracking Codes and Cryptograms For Dummies"; ISBN: 978-0-470-59100-0;October 2009
[14] Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone "Handbook of Applied Cryptography"; CRC Press; ISBN: 0-8493-8523-7
[15] WILLIAM F. FRIEDMAN; "MILITARY CRYPTANALYSIS, Part I, MONOALPHABETIC SUBSTITUTION SYSTEMS"
[16] Henk C.A. van Tilborg, Sushil Jajodia; "Encyclopedia of Cryptography and Security", 2nd edition; 2011; ISBN: 144195905X
[17] Wenbo Mao; "Modern Cryptograph"..
[18] Wels Chenbach; "Cryptography in C and C++".
[19] Koblitz, N.,"A Course in Number Theory and Cryptography, 2nd ed. New York: Springer-Verlag, 1994.
[20] A. Menezes, P. Van Oorschot, S. Vanstone,"Handbook of Applied Cryptography", CRC Press, 1996.
[21] Mark Adler, Jean-Loup Gailly,"An Introduction to Cryptography", released June 8, 2004. [Online] Available: http://www.pgp.com.
[22]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.
[23]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.
[24]Moinak chowdhury and Prof. Dr. Pranam Paul, **"**BLOCK BASED DATA ENCRYPTION AND DECRYPTION USING THE DISTENCE BETWEEN PRIME NUMBERS"
[25] Anupam Mondal, Prof. Dr Pranam Paul, Implementing Cryptography on the Concept of Returning Back Its Own Nest of a Bird, IJIRCCE
[26] Sukanya Chakravarty, Prof. Dr. Pranam Paul, Approach Based on Finding the Difference between Consecutive Numbers, IJIRCCE

BIOGRAPHY

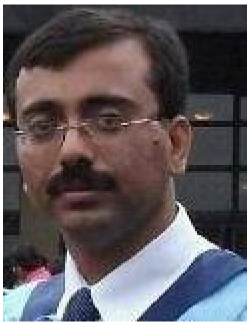
Nabendu Paul, he is a student of MCA from Narula Institute of Technology and former student of BCA from techno India Institute Technology under WBUT.

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 5, May 2016



Author 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**.**