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

An Approach to Block Based Ciphering Using Bit Wise Calculation for Representation of A Number Using Its Corresponding Perfect Square Number and Position of Prime Number

Subir Sharma[1], Prof. Pranam Paul[2]

Final Year Student of MCA, Narula Institute of Technology, Agarpara, Westbengal, India[1]

HOD, Department of Computer Application, Narula Institute of Technoloy, Agarpara, Westbengal, India[2]

ABSTRACT:-Here the main concept of the algorithm comes from cryptography, to secure any kind of file as it is implemented on bit-level. The strength of the technique is analyzed in this paper. This is a block based private key cryptographic technique. Here our idea is focused on technique by which we will calculate the nearest perfect square number of a given number along with its square root and position of prime by which the difference among the perfect square number and the given number can be represented . In parallel in each step we will also calculate the number of bits by which these perfect square number and prime number will be represented in its encrypted form. The algorithm section describes the method in brief.

KEY WORDS:-Cryptography, Encryption, Decryption, Cipher, Private key, Symmetric key, Plain text, Network security.

I. INTRODUCTION

Cryptography, not only protects data from hacking or alteration, but can also be used for user authentication. The scenario of present day of information security system includes confidentiality, authenticity, integrity, and non-repudiation. Security breaches can often be easily prevented. How? This guide provides you with a general overview of the most common network security threats and the steps you and your organization can take to protect yourselves from threats and ensure that the data travelling across your networks is safe. Each type of data has its own features; therefore different techniques should be used to protect confidential data from unauthorized access. Here the same idea of cryptography is working.

II. RELATED WORK

The author used perfect square number to calculate the difference between two numbers and calculated the number of bits required to represent them [18]. The author emphasized on division method where how many times division method will be applied is calculated [17]. Depending on the primer number, basic concept of this algorithm is obtained [7]. 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. ALGORITHM

In this section the process of encryption will be illustrated in details, in parallel the process of representing a number in binary by a proper calculation is also illustrated which is a key factor of our process. The key structure is also explained in this step.

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

1. Key structure: As per our algorithm we will have three segments of the key. These segments are illustrated in the table below.

Key Structure

1.	BLOCK SIZE
2.	NUMBER OF UNUSED BITS
3.	DUMMY BITS "0" THAT ARE ADDED TO ENCRYPTED BIT STREAM

The 1st segment is the block size which is any number selected by the sender.
The 2nd segment holds information about the number unused block which is left behind after selecting the bits as per the block size. The 3rd segment is the extra 0's that we need to append, so as to make it fully divisible by 8 with no remainder. This is done because 2^8 (using 8 bits) is 256 which is maximum ASCI value.

2. Encryption Process

STEP 1: Convert plain text to their ASCII value and then convert the ASCII value to their binary form. Let us take a binary bit stream which is to be encrypted and let's treat it as source bit stream.
STEP 2: Select "N" number of bits, where n is the block size. Convert these selected bits to their corresponding decimal values. A set of random numbers are generated as per the block size or key from the bit stream, where the block size is selected by the user. A set of decimal numbers will be obtained from this process.
STEP 3: Consider any one of the decimal number and calculate the perfect square number that is less than or equal to that decimal number. After obtaining the perfect square number lets calculate the square root of the perfect square number that we got .
Perfect square number → (2*2, 3*3, or 4*4 …)
Example : decimal number =24
Perfect square number less than 24 is 16
16 = 4 * 4 and square root of 16 is 4.
STEP 3.1: (BIT CALCULATION 1)
At this step we will calculate the number of bits in which the obtained square root number (of step 3.1) will be represented. For this, the calculation will be $2^n - 1$
where **n** is block size of key (segment 1) . Now we will calculate nearest perfect square number of the value that we will get and as the square root of the same number .Now we need to calculate the number of bits that are required to represent this number.
The number of bits required to represent the root that we will get at this step , will be our **required number of bits** in which square root of step 3.1 will be represented.
If n=5 then 2^5 -1 will be 31. Nearest perfect square number of 31 is 25 and square root of 25 is 5.To represent 5 we need(5→ 101) so 4 of step 3.1 will be represented in 3 bits.
STEP 4: At this step we will calculate the difference between the decimal number (i.e in step 3.1)
and perfect square number (i.e in step 3.1) , if there exists any difference , this difference will be our **new decimal number** on which further calculation will be performed.
24-16=8.
8 = new decimal number.
Now we will calculate the nearest prime number that is less than or equal to the new decimal number. (Nearest prime of 8 is 7). After getting the prime number we will calculate the position of the prime number starting from 0 , in our case we have taken "0" as a prime number . So now we have the **position of prime number,** we have to calculate the number of bits required to represent the position of the prime number.
Prime → 0 2 3 5 7 11 13 …….
position → 0 1 2 3 4 5 …….
STEP 4.1 : (BIT CALCULATION 2)
IN STEP 3.1 we got a perfect square number, at this step we will calculate the difference between next perfect square number and current perfect square number, suppose current perfect square number is 16 and next perfect square number is 25 and the difference between them is **8** as on 9th place we get next perfect square number itself.
(25-16)-1 = 8

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 4.2 : So now as we have the **difference** i.e 8, we will calculate the nearest prime number of the difference that we just got as well as the position of the prime starting from 0 as "0" is taken as a prime number.
Prime → 0 2 3 5 7 11 13 …….
position → 0 1 2 3 4 5 …….
Now at last we will calculate number of bits that is required for representing the position of the prime number that we just got. The result that we will get which is the number of bits required , will be the number in which the prime number of step 4.1 will be represented.
Example → Nearest prime of 8 is 7 and position is 4 as we count from "0". The result that we will get here will be the number in which the position of the prime number that is less than or equal to the new decimal number of step 4.1 will be represented.
7→ 111 (3bits)
STEP 5: At this step we will check if there is any difference left between decimal number of step 3.1 and sum of our calculation that is sum of perfect square number of step 3.1 and the prime number of step 4.1. If the sum is not equal to the decimal number of step 3.1 then the **difference** between the decimal number and sum of perfect square number and prime number (of step 3.1 and 4.1) will be calculated .
STEP 5.1: 24 – (16+7) → 24-23 = 1.
The **difference** (ie 1)that we will get will be the next number to consider. But here also calculation is to be performed alike our previous calculation ,that in how many bits this difference of number of step 5.0 will be represented.
STEP 5.2 : (BIT CALCULATION 3)
At this step we will recall the prime number that we got in step 4.1(i.e **7)**
Here we will calculate the difference between the prime number of step 4.1 and its immediate next prime number. After getting the difference we will calculate the position of the difference counting from 0. Now after getting the position we will calculate the minimum number of bits
required to represent the position of number which is the difference between the primes.
7→prime number
11→next prime number.
Difference will be 11-7=4
Position of 4 counting from 0 is 3 and to represent 3 number of bits required is 2 bits.
Difference→ 1 2 3 4 5
Position → 0 1 2 3 4
3→11 (2 bits, bit count of 3)
The number of bit count that we will get here will be number in which the difference of step 5.1 will be represented (which is the difference between the decimal number and sum of perfect square number step 3.1 and prime number of 4.1).
So 1 will be represented in 2 bits as 1→01,
Now as per our calculation the decimal number of step 3.1 is equal to the sum of perfect square number of step 3.1 prime number of step 4.1 and any difference of numbers of step 5.1. So our encryption of a single decimal number is complete.
24 – (16+7+1) = 0.
If its 0 then we don't proceed further. And this process would repeat for all decimals that we get from the binary bits.

STEP 6 : In this step we will discuss about the process , how the bit stream is to be written in the new file. Till now we discussed about the operation to be performed on the decimal number and now we have a binary bit stream that is encrypted. After getting this new bit stream we will count that the obtained encrypted bit stream that we have is divisible by 8 with 0 as remainder or not. If it is not divisible by 8 then we add "0"(dummy bits) to bit stream , to make it divisible by 8 with 0 as remainder. And if there remains any bits after selecting the bits as per block size that will be our unused bits.
1st : At the beginning of the binary bit stream we will first append the dummy bits and the information about the number of dummy bits is present in the 3rd segment of the key.
2nd :after the dummy bits there will and the unused bits, and the information about the number of unused bits that is to be selected will be stored in the 2nd segment of the key.
3rd : from now onward encrypted bit stream will be as it is, which we got after performing encryption operation.

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

3. Decryption Process

STEP 1: At first from the 3rd line of the key we will fetch information about the dummy bits that are appended at the beginning of the encrypted bit stream. After fetching this information we will first discard the dummy bits (remove added 0). Then we will check the 2nd line of the key to check how many bits to select next, as this value indicates is there any unused bits or not. If there would have been any unused bits then these bits would be saved in some place and added at end of decrypted bit stream that we will get.
Now we have a bit stream that is encrypted and this bit stream is to be decrypted. Our first target is to calculate how many bits is to be taken into consideration of which the binary to decimal conversion is to be done.
Each time how many bits are to be selected need to be calculated and will be a continuous process for each bits.

STEP 2.1 (BIT SELECTION)
From 1st segment of key we will get the block size, suppose we have block size= n.
So we will do $2^n - 1$
We will get a value from this equation where **N** is the block size. Now we will calculate the nearest perfect square number of the value. After calculating the perfect square number we will calculate square root of the perfect square number.
Now we will calculate minimum number of bits required to represent the square root number. So now we have a value which will decide how many bits is to be taken from the encrypted bit stream.
If n=5 then 2^5 = 32 and as per $2^n - 1$ the value we get is (32-1) 31.
Nearest perfect square number of 31 is 25 and square root of $\sqrt{25}$ is 5.
So we will select first 3 bits as (5→ 101) 3 bits.
STEP 2.2 : From the earlier step we got a value that how many bits we will select, now we will convert these binary bits into its corresponding decimal values. After getting the decimal value we will do the square of the decimal value that we just got and store the value in some place.
Example : →100←10001 …
Then taking 1st 3 bits we 100→4 and
4^2 = 16

STEP 3.1 : BIT SELECTION
From step 2.2 we got a decimal value that is a perfect square number. At this step we will calculate the difference between next perfect square number and current perfect square number (minus 1). Now we will calculate the nearest prime number of the number that we just got.
Example :-
(25-16)-1 = 8
So now as we have the **difference** i.e 8, we will calculate the nearest prime number of the difference that we just got as well as the position of the prime starting from 0 as "0" is taken as a prime number.
Nearest prime number of 8 is 7, and position of this number is 4.
Prime → 0 2 3 5 7 11 13 …….
position → 0 1 2 3 4 5 …….
Now at last we will calculate number of **bits** that is required for representing the position of the prime number that we just got, and select that many number of bits.
4→100 (3bits), so select 3 bits.
Now we need to calculate the minimum number of bits that is required to represent the position of prime number. This **number** that is required to represent the position of prime number, is the number that we will select next from the encrypted bit stream.
Example : 100→100←1001
100 = 4.

STEP 3.2: From the selected bits as per step 3.1 we will get a decimal value. Here this decimal number indicates the positional value of the prime number starting from 0. That means taking 0 as prime and counting from 0 the prime number of that position will be calculated.
Prime → 0 2 3 5 7 11 13 …….

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

position → 0 1 2 3 4 5 ……

This value that we will get will be added to the previous value that we got in step2.2. So now we have a new value that will be stored by replacing the earlier value.
16+7=23

STEP 4.1 (BIT SELECTION)
In earlier step 3.2 we got prime number. In this step we will recall that prime number, and we will calculate the difference between that prime number and its immediate next prime number, and also counting from 0 find the position of the value which is the difference among the prime number. Now calculate the number of bits required to represent this number. And the value that we will get in this step will be the value for selecting next number of bits , and from this bits we will get a decimal value. The result that we will get in this step now will be the next number of bits that we will select from encrypted bit stream.
Example : 7 and 11 both are prime number and the difference between them i.e 11-7=4 Counting from 0 the position of prime is 3.and to represent 3 we need 2 bits. So 2 bits will be selected 100100→01←
STEP 4.2: Counting from 0 the position of the number is 3 and to represent 3 we need 2 bits. So 2 bits will be selected 100100→01←
In step 4.1 we got how many bits is to be selected and their corresponding decimal value is obtained. This decimal value will be added the to the final value that we got in step 3.2.
01→1
So the new number will be 23+1 =24.
By adding all the values of step 2.2, step 3.2 and step 4.2 we will get a decimal value which is our decrypted value. After getting the decimal, this decimal number will be treated as ASCII value their corresponding character value is obtained.
These processes (step1 to 4.2) will repeat until all the bits are executed.

IV. RESULT ANALYSIS

Here we have shown result analysis for different file size taking block size as 6.
Size and Time Comparative Report
This algorithm has been implemented on number of data files with varying types of content and sizes of wide range. Here we compared between the plain text file size, encrypted file size, from where we got encryption time taken , and also encryption time/byte. And also the comparison is done between the encrypted file size and the decrypted file size, time taken for decryption with decryption time per byte.

.Table 1
Analysis with same key but different file size

FILE SIZE (in BYTE)	ENCRYPTED FILE SIZE IN BYTE	ENCRYPTION TIME (in sec)	ENCRYPTION TIME/BYTE
48	54	0.05494506	0.0011446887
244	273	0.10989011	0.0004503693
366	411	0.16483517	0.0004503693
486	546	0.21978022	0.0004522226
608	682	0.27472529	0.0004518508

Table 1 shows time taken for encryption for different file size and time taken for encryption for each byte.

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

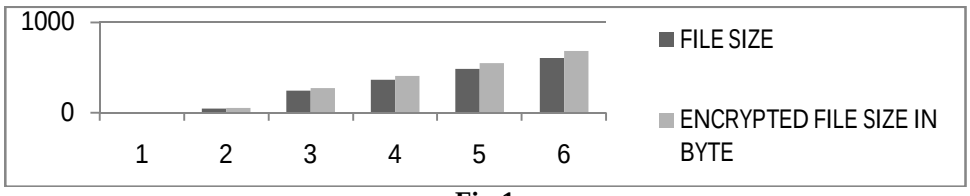

Fig:1
Figure of original file size vs encrypted file size

Fig:1 shows comparison between original file size and encrypted file size.

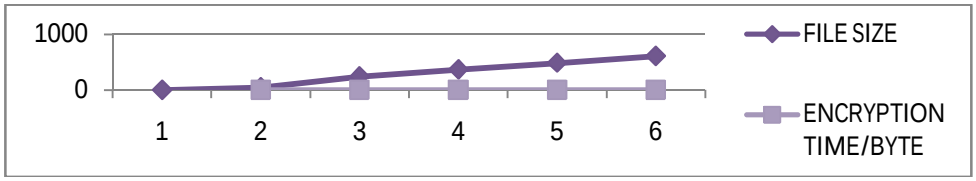

Fig:2
Figure of original file size and encryption time/byte.

Fig: 2 shows encryption time per byte in comparison to file size

Table 2
Decryption time of various file size

Table 2

File size in byte	Decrypted file size in byte	Decryption time in sec	Decryption time/byte
54	48	0.054945	0.0010175000
273	244	0.164135	0.0006012271
411	366	0.274725	0.0006684306
546	486	0.329670	0.0006037912
682	608	0.384615	0.0005639516

shows time taken for decryption for different file size and time taken for decryption for each byte.

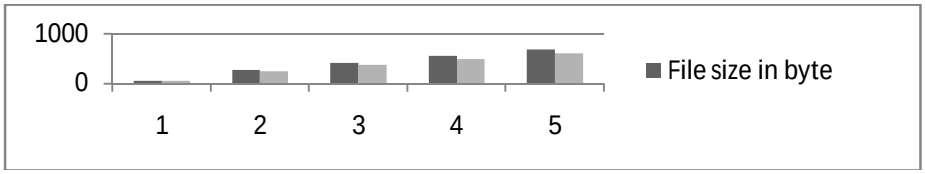

Fig:3
Figure of original file size and decrypted file size.

Fig 3 shows comparison between original file size(encrypted) vs decrypted file size

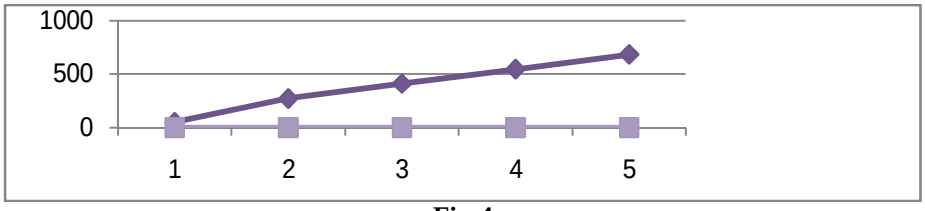

Fig:4
Figure of original file size and decryption time/byte.

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

Fig:4 shows comparison between file size and decryption time per byte.

V. SECURITY

If the size of key is n, then the possibility of correct number of bits to be selected also increases also the bits in which the numbers need to represented also increases.
From the above observation we can say that if the size of the block is increased, then the probability to choose the correct number of bits also increases exponentially.

VI. CONCLUSION

My conclusion towards this algorithm is that I have tested with 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 securityprinciples and practices", 4th edition, Pearson Education, Inc.,publishing as Prentice Hal, 2006.
[2]Pranam Paul, Saurabh Dutta, A K Bhattacharjee, "AnApproach to ensure Security through Bit-level Encryption with
Possible Lossless Compression", International Journal ofComputer Science and Network Security", Vol. 08, No. 2, pp.
291 – 299, 2008.
[3]SanjitMazumdar, SujayDasgupta, Prof.(Dr) PranamPaul, "Implementation of Block based Encryption at Bit-Level", International journal of Computer Science andNetwork Security, Vol. 11, No.2, pp. 18-23, 2011.
[4]SujayDasgupta, SanjitMazumdar, Prof.(Dr) Pranampaul, "Implementation of Information Security based on Common Divison", International journal of Computer Science andNetwork Security, Vol. 11, No.2,pp. 51-53, 2011.
[5]http://en.wikipedia.org/wiki/Symmetric-key_algorithm
[6] AsokeNath, Saima Ghosh, Meheboob Alam Mallik, "Symmetric Key Cryptography using Random keyGenerator", 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 ofInformation Security using Substitution of Bits Through PrimeDetection in Blocks", Proceeding of National Conference onRecent Trends in information Systems(ReTIS-06), Organizedby 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, Pa:r.t 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 andNetwork 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 andNetwork Security,ISSN: 0974 – 9616 vol-7,No.2,2015.

. BIOGRAPHY

Subir Sharma, is a final year student of MCA 2016 from Narula Institute of Technology (NIT), Agarpara ,Kolkata.

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

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.**