ISSN 2320 - 2602
Volume 3, No.5, May 2014
International Journal of Advances in Computer Science and Technology
Available Online at http://warse.org/pdfs/2014/ijacst04352014.pdf

A Cryptography Algorithm Using the Operations of Genetic Algorithm & Pseudo Random Sequence Generating Functions

Suvajit Dutta[1], Tanumay Das[2], Sharad Jash[3], Debasish Patra[4], Dr.Pranam Paul[5]

[1]Student, MCA, Narula Institute of Technology, Kolkata, West Bengal, INDIA, suvajt.brj@gmail.com
[2]Student, MCA, Narula Institute of Technology, Kolkata, West Bengal, INDIA, tanumay.das@gmail.com
[3]Student, MCA, Narula Institute of Technology, Kolkata, West Bengal, INDIA, sharad99jash@gmail.com
[4]Student, MCA, Narula Institute of Technology, Kolkata, West Bengal, INDIA, debnit90@gmail.com
[5]Computer Application, Narula Institute of Technology, Kolkata, West Bengal, INDIA, pranam.paul@gmail.com

ABSTRACT

In today's computer world security, integrity, confidentiality of the organization's data is the most important issue. This paper deals with the confidentiality of electronic data which is transmitted over the internet. Data encryption is widely used to ensure security of the data. Genetic algorithms (GAs) are a class of optimization algorithms. Many problems can be solved using genetic algorithms through modeling a simplified version of genetic processes. In our approach we are using the concept of genetic algorithms with pseudorandom function to encrypt and decrypt data stream. The encryption process is applied over a binary file so that the algorithm can be applied over any type of txt as well as multimedia data.

KEYWORDS

Encryption, Decryption, Genetic algorithm, Mutation, Pseudorandom Sequence, symmetric Key, Cryptography.

1. INTRODUCTION

In the advent of greater demand in digital signal transmission in recent time, the problem of huge losses from illegal data access has become a burning issue. Accordingly, the data security has become a critical and imperative issue in multimedia data transmission applications. In order to protect valuable information from undesirable users or against illegal reproduction and modifications, various types of cryptographic schemes are needed.

There are two types of cryptographic schemes: symmetric cryptography and asymmetric cryptography. The symmetric scheme uses the same key for encryption and decryption. Two keys is used In asymmetrical cryptography, one for encryption, known as the public key, and the other for decryption, known as the private key. Asymmetric cryptography is often used in key distribution and digital signature for its slow processing speed. The symmetric cryptography is normally used to encrypt private data for its high performance. There have been various data encryption techniques on multimedia data proposed. Genetic Algorithms (GAs) are among such techniques.

The genetic algorithm is a search algorithm based on the mechanics of natural selection and natural genetics. The genetic algorithm belongs to the family of evolutionary algorithms, along with genetic programming, evolution strategies, and evolutionary programming. The set of operators usually consists of mutation, crossover and selection. In our work we are using crossover and mutation operation.

Crossover: Crossover operator has the significance as that of crossover in natural genetic process. In this operation two chromosomes are taken and a new is generated by taking some attributes of first chromosome and the rest from second chromosome. In GAs a crossover can be of following types:

1. *Single Point Crossover:* In this crossover, a random number is selected from 1 to n as the crossover point, where n being the number of chromosome. Any two chromosomes are taken and operator is applied.

2. *Two Point Crossover:* In this type of crossover, two crossover points are selected and the crossover operator is applied.

3. *Uniform Crossover:* In this type, bits are copied from both chromosomes uniformly.

Mutation: Mutation is a genetic operator used to maintain genetic diversity from one generation of population to the next. It is similar to biological mutation. Mutation allows the algorithm to avoid local minima by preventing the population chromosomes from becoming too similar to each other. GAs involves string based modifications to the elements of a candidate solution. These include bit-reversal in bit-string GAs.

Pseudorandom number generator: A pseudorandom number generator is an algorithm for generating a sequence of numbers that approximates the properties of random numbers. The sequence is not truly random in that it is completely determined by a relatively small set of initial values, called the PRNG's state. A PRNG suitable for Cryptographic applications is called a cryptographically secure PRNG (CSPRNG).

Some PRNG files are Blum Blum Shub, Wichmann-Hill, Complementary-multiply-with-carry, Inversive congruential generator, ISAAC (cipher), Lagged Fibonacci generator, Multiply-with-carry, Naor-Reingold Pseudorandom Function, Park–Miller random number generator, RC4 PRGA, Well Equidistributed Long-period Linear, Xorshift, Mersenne twister, Sophie Germain prime.

In this paper, we propose a new approach for encrypting binary files.

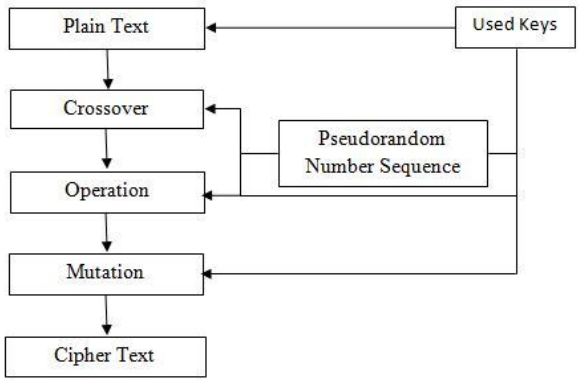

Figure 1: Flowchart of the encryption process

We have shown the encryption process briefly in the Figure.1.

The rest of the paper is organized as follows. In Section 2, we introduce the proposed method. Section 3discusses the experimental results. Section 4, discusses about the analysis of the proposed method. Section 5 concludes the paper.

2. PROPOSED METHOD

The block diagram of Figure1 shows the encryption process. The diagram consists of Crossover, Mutation and Pseudorandom Number Sequence. Each module will be discussed in the following sub section:

2.1 Pseudorandom Sequence

PRNG used to generate a Pseudorandom Sequence of numbers for our encryption and decryption method. In our method we are using two sequence and the values needed to generate those sequence are stored as Secret key.

2.2 Crossover

In our method we are using three type of crossover operation (*Single point, Two Point, Uniformed*). Modulating the Pseudorandom Sequence by 3 the crossover operations will executed according to the sequence. The process of this crossovers is showed in the figures:

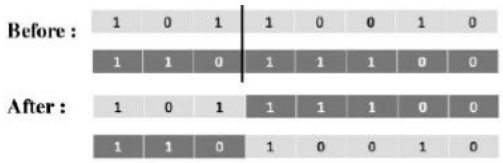

Figure 2: *Single Point Crossover*

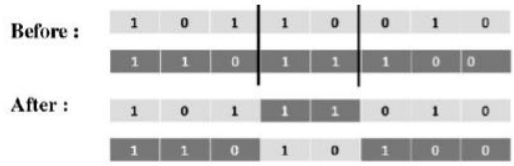

Figure 3: *Two Point Crossover*

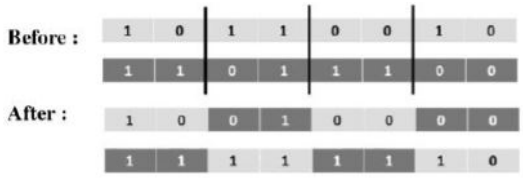

Figure 4: *Uniformed Crossover*

2.3 Mutation

For Mutation operation we are using flip bit mutation technique (i.e. if the genome bit is 1,it is changed to 0 and vice versa). Mutation operation will be applied on one of the binary data block. The index number of that data block is stored as Secret key.

2.4 Key Structure

In the algorithm we have basically used five keys: one is for dividing the plane text into blocks. Second is for generating pseudorandom sequence for crossover operation. Third is for generating another pseudorandom sequence, fourth key is for modulate the sequence (Range 16 - 255) and last one is for mutation operation.

2.5 The Encryption Process

1) Transform the file into binary sequences and divide the sequence containing 'n' bits in each block. The value of 'n' is a Secret Key.

2) Generate the first pseudorandom sequence of numbers and modulate the sequence by 3 to select any of the three crossover operations, from that sequence we can select:

0. Single point
1. Uniformed
2. Two point

The crossover operation will be applied on each block of binary digits.

3) Generate another pseudorandom sequence of numbers and modulate the sequence with a secret key (range from 16-255)
We denote the sequence as:
Z1, Z2, Z3….Zn
Find out the decimal value of each crossover blocks i.e.:
C1, C2, C3….Cn
Now do the following operation,

$$Xi=Zi \oplus Zi<< (n/2)$$
$$Ei=Ci \oplus Xi,\ i=i+1 \quad (i = 1, 2, 3, 4\ldots n)$$

4) Repeat step 3 until end of the data.

5) Now between E1, E2, E3….En block select a block for mutation. Perform mutation operation on the mentioned block, the number will be then saved into the key file,
Ei = (255-Ei)
i.e. the encrypted block looks like:
E1, E2,(255-E3),….En.
6) By printing the E1, E2, E3…., En values into a file (any file format) we can get our cipher text.

2.6 The Decryption Process

The steps for decryption are just reversal of the encryption process.

1) Read the key file and read the values form reverse direction. Read the encrypted text file to get the encrypted text and divide it into 'n' bit per sequence mentioned in the key.

2) (Mutation)
Do Mutation of the mentioned block number in the key:
Ei =255-(255-Ei)

3) Generate a pseudorandom sequence and modulate the sequence with the secret key,
We denote the sequence as:
Z1, Z2, Z3….Zn
Find out the decimal value of each crossover blocks i.e.:
E1, E2, E3…. En
Now do the following operation,

$$Xi=Zi \oplus Zi<< (n/2)$$
$$Ci=Ei \oplus Xi,\ i=i+1 \quad (i = 1, 2, 3, 4\ldots n)$$

4) (Crossover Operation)
Generate the pseudorandom sequence by reading the keys from the key file and modulate the sequence by 3 to select the three cross over operations:

0. Single point
1. Uniformed
2. Two point

5) After the reverse crossover process convert the plane binary bit blocks into the desired output file format as the decrypted file.

3. EXPERIMENTAL RESULTS

In our experiment over the algorithm for the pseudorandom sequence we are here using Blum Blum Shub PRNG which provides a strong security proof.

3.1. Encryption:

Consider a simple text file – **Narula**

1) Binary equivalent of this text is-
01001110011000010111001001110101011011000110000
Length is: 48

2) Now, we divide the binary string into 6 blocks with 8byte/ block.

- *8 is the 1st key value.*

B1=01001110 B2=01100001
B3=01110010 B4=01110101
B5=01101100 B6=01100001

3) Now by generating a pseudorandom sequence and modulate the value to 3 we get the sequence: 0 2 2 2 1
So, for the crossover between B1 & B2, 0th operation *single point crossover* being selected,

B1=0100 1110 B2=0110 0001
C1=0100 0001 B2=0110 1110

C1 will store as crossover string.

Then, between B2 & B3, 2nd operation *two point crossover* being selected.

B2= 01 101 110 B3=01 110 010

C2= 01 110 110 B3=01 101 010

C2 will store as crossover string.

After the crossover process is being finished, the crossover string will look like:

C1= 01000001 C2= 01110110
C3 = 01110010 C4= 01101101
C5 = 01101101 C6= 01100000

4) Generate the next pseudorandom sequence by using the prime values.
By using secret key value modulation of the sequence will become:

Z1=78 Z2=60 Z3=114
Z4=159 Z5=108 Z6=144

By finding decimal value of each block we get ,

C1=65 C2=118 C3=114
C4=109 C5=109 C6= 96

Now perform the following operation,

X1=78 $\oplus$78<<4
E1= 65 $\oplus$X1

Z2=60 $\oplus$60<<4
E2= 118 $\oplus$X2

After finishing the operation,
We get,

E1=1263 E2=906 E3=1824
E4=2306 E5=1729 E6=2544

5) Perform mutation operation on the

4th block which is a secret key,

E4 = (255-2306) = -2051

After finishing the operation we get,

E1=1263 E2=906 E3=1824
E4=-2051 E5=1729 E6=2544

6) Finally transforming E1, E2, E3, E4, E5, and E6 into character & putting it into a file we get:

Encrypted String: **ïŠ ýÁð**

3.2. Decryption:

1) Read the key file:

Key: 8 47 37 62 47 57 84 180 3

Read the Encrypted text file:

Encrypted String: ïŠ ýÁð

Transform into corresponding integer values:

E1=1263 E2=906 E3=1824
E4=-2051 E5=1729 E6=2544

2) (Mutation)
By using the key we performed mutation operation to the block: E4

E4 = (255-(255-2306)) =2306

After finishing the operation we get,

E1=1263 E2=906 E3=1824
E4=2306 E5=1729 E6=2544

3) Next by generating Pseudorandom Sequence and modulate the sequence by the key value we get the sequence like:

Z1=78 Z2=60 Z3=114
Z4=159 Z5=108 Z6=144

The value of each encrypted blocks are:

E1=1263 E2=906 E3=1824
E4=2306 E5=1729 E6=2544

Now by the following operation,

X1=78 $\oplus$8<<4
C1= 1263 $\oplus$1

X2=60 $\oplus$60<<4
C2= 906 $\oplus$2

After finishing the full operation we get the seq. like:

C1=65 C2=118 C3=114
C4=109 C5=109 C6= 96

4) (Cross over)
By generating a pseudorandom sequence with the key value & modulate the value with 3 we get the sequence: 0 2 2 2 1 which are to be used for operation.

The Crossover blocks with the integer values are transformed into equivalent binary numbered blocks.

And at first C5 & C6 get crossover with 1st operation (Uniformed crossover).

C5=01 10 11 01 C6=01 10 00 00

C5=01 10 11 00 B6=01 10 00 01

Then, B6 will be stored as plane bit block.

Now, the crossover will executed between C4 & C5 with 2nd operation,

C4=01 10 1 101 C5=01 10 1 100

C4= 01 10 1 101 B5=01 10 1 100

B5 will store as plane bit block.

After the crossover process is being finished, the bit blocks will look like:

B1= 01001110 B2 = 01100001
B3= 01110010 B4 = 01110101
B5= 01101100 B6 = 01100001

5) Then, the full string will look like:

01001110011000010111001001110101011011000110000 1

By transforming binary values into integer value containing 'n' bit per block key value and by printing the integer blocks to the decrypted file we get:

Decrypted String: **Narula**

4. ANALYSIS and RESULT

This encryption process is performed on binary data. Hence we conclude that, this method can be applied to encrypt any formats of data like text, image, sound and many data since they are all stored in binary form in the computer. The binary data is divided into blocks, the encryption and decryption procedure can be carried out as was discussed above.

4.1 Size Comparative Report

We have tested the algorithm on various formats and sizes of data and the results are shown in the **Table 1.**

Table 1: Size Analysis Table

SL. NO	Source File Name	Original Plain Text (bytes)	Decrypted Plain Text (bytes)
1.	ABC.txt	48	48
2.	school.txt	564	564
3.	trs.tif	238543	238543
4.	Final.py	32768	32768
5.	think.cpp	693	693
6.	Suva.jpg	8960	8960
7.	chvost.bat	17360	17360

4.2. Time Comparative Report

Table 2, shows the time analysis of the above algorithm, time analysis is performed on the **Table 1** values.

Table 2: Time Analysis Table

Sl. No	File Size (bytes)	Encryption Time	Decryption Time	Encryption / Byte	Decryption / Byte
1	48	0.000565	0.000075	0.00001177	0.00000156
2	564	0.002236	0.000694	0.00000396	0.00000123
3	238543	0.452387	0.359647	0.00000190	0.00000151
4	32768	0.025486	0.017845	0.00000078	0.00000054
5	693	0.005843	0.004127	0.00000843	0.00000596
6	8960	0.065413	0.008413	0.00000730	0.00000094
7	17360	0.235649	0.228631	0.00001357	0.00001317

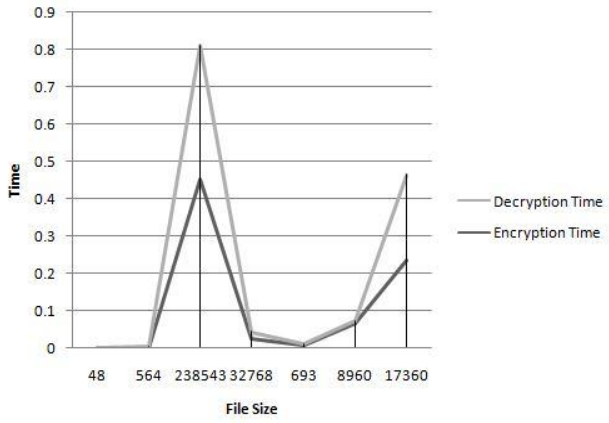

Figure 5: Time Analysis Graph

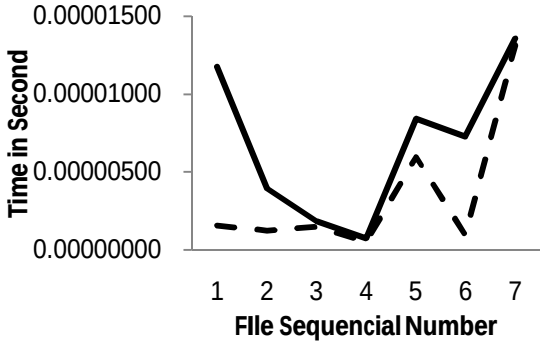

Figure 6: Time per unit size graph

Figure 2 represents the graph which compares between encryption and decryption time for per byte. Continuous line represents encryption time for 1 byte whereas decryption time for 1 byte is represented by discontinuous line. From the nature of these lines, it is cleared that

different times are taken for encrypting 1byte content of the different files as line is not parallel to the X axis. It is same for decryption also. So from this, it can be concluded that encryption as well as decryption depends on the content of the text.

5. CONCLUSION AND FUTUREWORK

In this paper, we propose a genetic algorithm based secret key encryption method. After the examinations of the proposed method, it is clear that this encryption method is satisfied the goals that are required in any encryption method for encrypt binary files.

The algorithm is being implemented on C++ language and it works out perfectly. For Pseudorandom Sequence generation we have used Blum Blum Shub PRNG function proposed in 1986 by Lenore Blum, Manuel Blum and Michael Shub. Blum Blum Shub takes the form:

$$x_{n+1} = x_n^2 \bmod M$$

where $M = pq$ is the product of two large primes p and q. At each step of the algorithm, some output is derived from x_{n+1}; the output is commonly either a bit parity of x_{n+1} or one or more of the least significant bits of x_{n+1}.

The seed x_0 should be an integer that is co-prime to M (i.e. p and q are not factors of x_0) and not 1 or 0.

In the future work, we plan to design a sophisticated software based on this technique which will targeted to use in highly secure multimedia data transmission applications.

REFERENCES

[1] Pranam Paul, Saurabh Dutta, **"A Private Key Storage- Efficient Ciphering Protocol for Information Communication Technology"**, National Seminar on Research Issues in Technical Education (RITE), March 08-09, 2006, National Institute of Technical Teachers' Training and Research, Kolkata, India

[2] David E Goldberg, **"Genetic algorithms in search, optimization and machine learning"**, Addision-Wesley Pub.Co.1989.

[3] Garg Poonam, "**Genetic algorithm Attack on Simplified Data Encryption Standard algorithm"**, International journal Research in Computing Science, ISSN1870-4069, 2006.

[4] Tragha A., Omary F., and A. Kriouile, **"Genetic Algorithms Inspired Cryptography"** A.M.S.E Association for the Advancement of Modeling & Simulation Techniques in Enterprises, Series D: Computer Science and Statistics, 2005.

[5] Douglas, R. Stinson, **"Cryptography - Theory and Practice"**, CRC Press, 1995.

[6] C.E. Shannon, **"Communication Theory of Security System"**, Bell, System Technical Journal, 28, 1949

[7] Ivars Peterson's book, **"The Jungles of Randomness".**

[8] Michael Luby, **"Pseudorandomness and Cryptographic Applications"**, Princeton Univ Press, 1996.

[9] Goyat, S., **"GENETIC KEY GENERATION FOR PUBLIC KEY CRYPTOGRAPHY."** International Journal of Soft Computing and Engineering (IJSCE), Volume 2, Issue 3, July 2012.

[10] H.Bhasin and S. Bhatia, **"Application of Genetic Algorithms in Machine learning"**, IJCSIT, Vol. 2 (5), 2011.

[11] Delman, B., **"Genetic Algorithms in Cryptography."** Master of Science in Computer Engineering, Rochester Institute of Technology, Rochester, New York, July 2004.