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.2, Special Issue 4, September 2014

Advanced Cryptographic Technique Using Double Point Crossover

P. Lakshmi Devi[1] , G. Sai veena[2]

Associate professor[1], Annamacharya Institute of Technology & Sciences, Rajampet, A.P, India

M.Tech scholar[2], Annamacharya Institute of Technology & Sciences, Rajampet, A.P, India

ABSTRACT*:* In this paper a cryptographic algorithm is introduced. This technique uses two keys for encryption and decryption. The technique is generated intermediate cipher followed by genetic function double point crossover to produce final cipher. Square matrix is used to put the input stream in a unique manner. Left diagonal's positional value will be the key 1 and with that key intermediate cipher text will be produced. Double point crossover is applied on the binary field of intermediate cipher text. Before doing the crossover 5 digit random number will generate as key2. According to the digit of key 2 block division process and crossover point is finalized to produce final cipher. Reverse procedure with keys will generate plaintext.

KEYWORDS: Substitution, Encryption, decryption, Key, Plain-text, Cipher-text, Crossover.

I.INTRODUCTION

The demand for effective internet security is increasing exponentially day by day . So for high protection, maintaining integrity of the data a robust and secure security system is needed. With wide application of Internet, especially the increasing adoption of the cloud computing paradigm, storing sensitive user data to un-trusted, remote hosts on Internet has been popular. The public availability of information about individuals has led to natural privacy concerns, even though the true identities of participants in traces are made anonymous. Cryptography is the science of making communication unintelligible to everyone except the intended receiver(s). It is the study of methods of sending messages in disguised form so that only intended recipients can remove the disguise and read the message.

A cryptosystem is a set of algorithm, indexed by some keys(s), for encoding messages into cipher text and decoding them back into plaintext. Many genetic algorithm based encryption have been proposed. A Tragha et al., describe a new symmetrical block ciphering system named ICIGA (Improved Cryptography Inspired by Genetic Algorithms) which generates a session key in a random process. The block sizes and the key length are variable and can be fixed by the user at the beginning of the ciphering. ICIGA is an enhancement of the system (GIC) “Genetic algorithms Inspired Cryptography”.

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. 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. Two Point Crossover: In this type of crossover, two crossover points are selected and the crossover operator is applied. Uniform Crossover: In this type, bits are copied from both chromosomes uniformly. In this paper a new algorithm for encryption and decryption is introduced. The algorithm is based on the process of substitution and genetic function. In this proposed model at first count the number of letters present in plain text and each letter is placed into a square matrix in a specific manner, and calculating a key value, by using this key, the plain text transformed into intermediate cipher text. Genetic function is used in the binary form of intermediate cipher using another key which is randomly generated. Finally, the cipher text is obtained and just in the inverse way (using all keys) plain text will be achieved from the cipher text.

II. HISTORY AND BACKGROUND

The Data Encryption Standard (DES) is a block cipher (a form of shared secret encryption) that was selected by the National Bureau of Standards as an official Federal Information Processing Standard (FIPS) for the United States in 1976 and which has subsequently enjoyed widespread use internationally. It is based on a symmetric-key algorithm that uses a 56-bit key. The algorithm was initially controversial with classified design elements, a relatively short key length, and suspicions about a National Security Agency (NSA) backdoor. DES consequently came under intense academic scrutiny which motivated the modern understanding of block ciphers and their cryptanalysis.

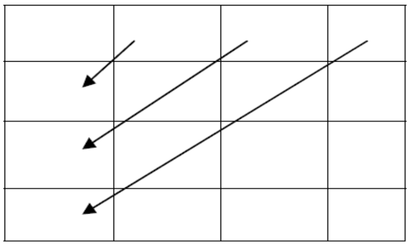
DES is a block cipher, which means that during the encryption process, the plaintext is broken into fixed length blocks and each block is encrypted at the same time. One Block is 64 bits and the key is 64 bits wide (but only 56 bits are used)

The algorithm is designed to encipher and decipher blocks of data consisting of 64 bits under control of a 64-bit key1. Deciphering must be accomplished by using the same key as for enciphering, but with the schedule of addressing the key bits altered so that the deciphering process is the reverse of the enciphering process. A block to be enciphered is subjected to an initial permutation IP, then to a complex key-dependent computation and finally to a permutation which is the inverse of the initial permutation IP-1. The key-dependent computation can be simply defined in terms of a function f, called the cipher function, and a function KS, called the key schedule. A description of the computation is given first, along with details as to how the algorithm is used for encipherment.Next; the use of the algorithm for decipherment is described. Finally, a definition of the cipher Function f is given in terms of primitive functions which are called the selection functions Si and the permutation function P.major drawbacks of this algorithm are this can be attacked by brute force attack and this suffers from mathematical as well as differential attacks.

The drawbacks of des algorithm is over by single point crossover but it provides less security and also it uses only single key while performing communication in between source and destination through encryption as well as decryption process

III. THE SCHEME

In this proposed technique placed each letter of input stream into a square matrix. Each letter is placed diagonally in the matrix. Squire matrix is selected according to the size of input stream. Arrangement of the letters of input stream into square matrix.

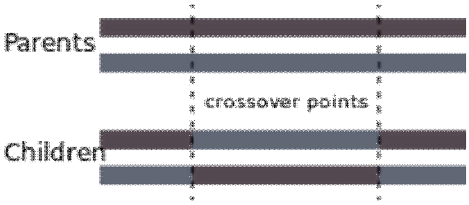


The key will be generated by adding the position value (A=1,B=2,……Z=26) of letters present in left diagonal position of the square matrix. Read all the letters serially from the square matrix and add the key value with positional value of the each letter. By this way substitute all the characters of the square matrix to generate intermediate cipher text. If the number of letter is not a square number then padding 0 to make it a square number, before placing the text into the box. All the letters of intermediate cipher text are converted into its binary code and generate a 5 digit random number. The first digit of random number is the section number by which all the bits are divided into small sections. If there is any remainder part, then will be discarded for future use. Each section is divided into blocks according to the last 4 digits of random number. This 5 digit random number is Key – 2. . Genetic function double point crossover is followed on blocks of bits of each section. The total block number of a section/partition is even, each block crossed over with the next block and produce Level 1 child blocks otherwise first block, N block (where N is odd) and second block, fourth block is crossed over and so on to produce Level 1 child blocks. Total number of block size of each section/partition, will be the Key – 3 of the proposed technique and point crossover point on binary field.

IV. FLOW CHART REPRESENTATION OF DOUBLE POINT CROSSOVER

In genetic functions, crossover is a genetic operator used to vary the programming of a chromosome or chromosomes from one generation to the next. It is analogous to reproduction and biological crossover, upon which genetic algorithms are based. Cross over is a process of taking more than one parent solutions and producing a child solution from them.

Double point crossover calls for two points to be selected on the parent strings. Everything between the two points is swapped between the parent organisms, rendering two child organisms:



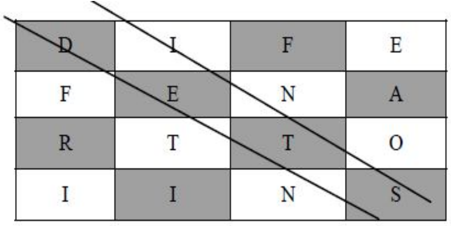
.

This function is one type of swapping process. We have to select bits from both MSB as well as LSB side to crossover two blocks. This selected bit's are based on number of blocks that we have partitioned from total number bits. Now the middle remaining bit string should be swapped for getting child blocks.

V.EXAMPLE

Encryption:

Let the Plain text is DIFFERENTIATIONS Size of the plain text is 16. Squire Matrix has been taken 4*4 and all letters of plaintext is placed into the box of the matrix according to the proposed technique which is shown below



D	I	F	E
F	E	N	A
R	T	T	O
I	I	N	S

Key 1= Position Value of 'D' + Position Value of 'E' +Position Value of 'T' + Position Value of 'S' = 4 + 5 + 20 +19 = 48 [Where, A=1…………….z=26]
Now the Key 1 will be added with each letter's position value of matrix to generate the intermediate cipher text.
Intermediate cipher text: **ZEBBANAJPEWPEKJO**

Z	E	B	A
B	A	J	W
N	P	P	K
E	E	J	O

Each letter of matrix is represented 64 bits binary code of ASCII value.
So that total bits number of intermediate cipher text will be: 16*64 bits=1024 bits this is given below.

Letter	Binary code(64 bit)
Z	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0101 1010
E	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 0101
B	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 0010
B	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 0010
A	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 0001
N	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 1110
A	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 0001
J	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 1010
P	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0101 0000
E	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 0101
W	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0101 0111
P	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0101 0000
E	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 0101
K	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 1011
J	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 1010
O	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 1111

Binary representation of intermediate cipher is:

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0101 10100000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 01010000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 0010
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 0010 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 00010000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 1110
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 00010000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 1010 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0101 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 0101 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0101 0111 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0101 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 01010000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 1011 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 1010
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 1111

5digit constant is taken as Key – 2. The number is: 54623 (Key – 2)
Depending on the Key – 2, 1024 bits will be partitioned into 5 sections, because the first digit of Key – 2 is 5 and each section will be divided into 4, 6, 2, 3 blocks respectively as
because 4, 6, 2, 3 are the next digits of Key – 2.
Section 1 is divided into 4 blocks.
Section 2 is divided into 6 blocks.
Section 3 is divided into 2 blocks.
Section 4 is divided into 3 blocks
Section 5 divided into 4 blocks.
Each section contains (1024 / 5) bits = 204 bits.
Discard the remainder (1024 % 5) bits = 4 bits for future use.
Last 4 bits '1111' will be discarded from intermediate cipher.

Partition 1 (Each Block contain (204 / 4) bits or 51 bits)

Block1	Block2	Block3	Block4
000000000000 000000000000 000000000000 000000000000 000	000000101101 000000000000 000000000000 000000000000 000	000000000000 000000010001 010000000000 000000000000 000	000000000000 000000000000 000000001000 010000000000 000

Partition 2 (Each Block contain (204 / 6) bits or 34 bits)

Block1	Block2	Block3	Block4	Block5	Block6
0000000000 0000000000 0000000000 00 00	00000000000 10000010000 0000000000 00	0000000000000 0000000000000 000000 00	00000001000 10100000000 0000000000 00	000000000000000 000000000000000 00 00	000100000100000 000000000 00000000 00

Partition 3 (Each Block contain (204 / 2) bits or 102 bits)

Block1	Block2

00000000000000000000000000000000010010
10000000000000000000000000000000000000
00000000000000000000 010101
	11000000000000000000000000000000
	00000000000000000000000000010011
	100000000000000000000000 00

Partition 4 (Each Block contain (204 / 3) bits or 68 bits)

Block1	Block2	Block3

00000000000000
00000001010000
00000000000000
00000000000000
000000000000
	00000000000000
	00010100000000
	00000000000000
	00000000000000
	000000000000
		000000000000
		010010110000
		000000000000
		000000000000 0000000000

Partition 5 (Each Block contain (204 / 4) bits or 51 bits)

Block1	Block2	Block3	Block4
0000000001 0001010000 0000000000 0000000000 0000000000 0	0000000000 0000000000 0001000101 0000000000 0000000000 0	0000000000 0000000000 0000000000 0000010010 1000000000 0	0000000000 0000000000 0000000000 0000000000 0000000010 0

Total block number of each partition will be treated as Pivot Point or Crossover Point. "X" symbol is represented crossover in the following.
In encryption process the crossover is operated using **left shift operation** as shown below.

Partition 1: Number of Blocks 4 (Even)

Block1 X Block2

000**0**00000000000000000000000000000000000000000**0**0000
X

000**0**001011010000000000000000000000000000000000**0**0 00

000000101101000000000000000000000000000000000000 0000 (**Block 1.1**)

0000000000000000000000000000000000000000000000000000 (**Block 1.2**)

Block3 X Block4

000**0**0000000000000001000101000000000000000000000**0**00 0
X
000**0**0000000000000000000000000000100001000000000**0**000

0000000000000000000000000000000010000100000000 000(**Block 1.3**)

0000000000000000000100010100000000000000000000000 000(**Block 1.4**)

Partition 2: Number of blocks 6 (Even)

Block1 X Block2
00000**0**0000000000000000000000**0**00000

X
00000**0**000000100000010000000000**0**000000

000 00000000010000001000000000000000000(**Block 2.1**)
0 0000000000000000000000000000000 00000(**Block 2.2**)

Block3 X Block4

0000 0**0**00 0000 0000 0000 0000 0000 **0**000 00

00 000**0** 0100 0101 0000 0000 0000 0000 0000

00000000000010000001000000000000000000000(**Block 2.3**)

00100000 0000 0000 0000 0000 0000 0 00000(**Block 2.4**)

Block5 X Block6

0000 0**0**00 0000 0000 0000 0000 0000 **0**000 00
X
00 010**0** 0001 0000 0000 0000 0000 00**0**0 0000

0000000 0001 0000 0000 0000 0000 000 00010(**Block 2.5**)

000010000 0000 0000 0000 0000 0000 0 00000(**Block 2.6**)

Partition 3: Number of Blocks 2 (Even)

Block1 X Block2

0**0**00 0000 0000 0000 0000 0000 0000 0000 0100 1010 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0101 **0**1
X

1**1** 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 1110 0000 0000 0000 0000 0000 0000 0000 0000 00**0**0

01 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 11100000 0000 0000 0000 0000 0000 0000 0000 000
1(**Block 3.1**)

1000 0000 0000 0000 0000 0000 0000 0000 0100 1010 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0101 0 0(**Block 3.2**)

Partition 4: Number of Blocks 3 (Odd)

Block1 X Block3

00**0**0 0000 0000 0000 0000 0101 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0**0**00
X

00**0**0 0000 0000 0100 1011 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0**0**00

0000 0000 0000 0100 1011 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 00 00(**Block 4.1**)

0000 0000 0000 0000 0000 0101 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 00 00(**Block 4.3**)

Block 2 will remain same.
Partition 5: Number of Blocks 4 (Even)

Block1 X Block2
000**0** 0000 0100 0101 0000 0000 0000 0000 0000 0000 0000 000**0** 000
X

0 00**0**0 0000 0000 0000 0000 0100 0101 0000 0000 0000 0000 00**0**0 00

000 00 0000 0000 0000 0000 0100 0101 0000 0000 0000 0000 000000(**Block 5.1**)

0000 0000 0100 0101 0000 0000 0000 0000 0000 0000 0000 0000000(**Block 5.2**)
Block3 X Block4
00 0**0**00 0000 0000 0000 0000 0000 0000 0000 0100 1010 0000 0**0**00 0
X

000 **0**000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 **0**100

000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000(**Block 5.3**)

000000 0000 0000 0000 0000 0000 0000 0000 0100 1010 0000 00000**ock 5.4**)

Concatenate all child blocks section/partition wise and discarded bits '0101' to produce cipher text in binary form. Length of the binary field will be 1024 bits. 1024 bits will be divided into 16 blocks of each 64 bits.
Final Cipher text will be:

2d 228 21 82 84 82 94 ae 9c 14 50 12c 228 228 25 20f

Decryption:

The decryption process is similar to the above encryption process but follows the reverse order.

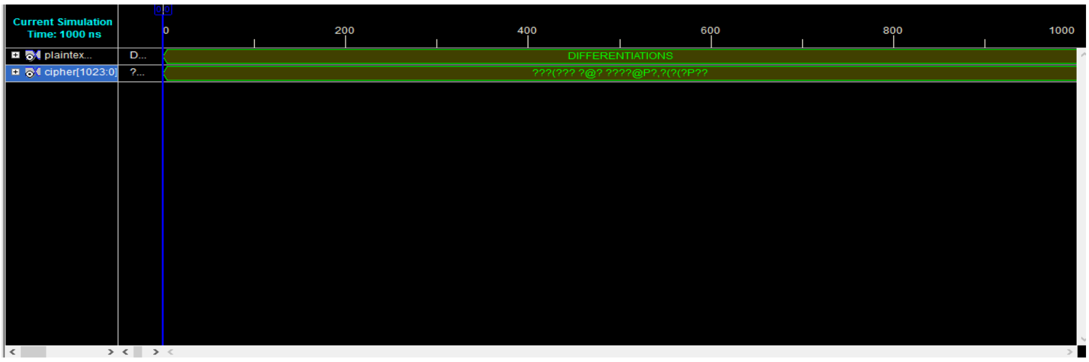
First we have to calculate the binary equivalent for the cipher text.
Total number of bits=1024
Key1=48
Key2=54623
1024/5=204
1024%5=4
Discard the last four bits '1111' and store it for future use.
According to key2, 1024 bits will be divide into 5 sections/partitions. After partitioning the bits into blocks then perform double point crossover genetic function laterconcatenate all child blocks section wise and discarded bits '1111' to produce intermediate cipher text in binary form.length of the binary field will be 1024 bits.1024 bits will be divided into 16 blocks of each 64 bits.

Intermediate cipher text is: **ZEBBANAJPEWPEKJO**
Substitute intermediate cipher by positional value (where A=1.B=2…Z=26)and key1(48)
Z=26-(48-26)=4=**D**, E=5-(48-26)=9=**I**, B=2-(48-26)=6=**F**,

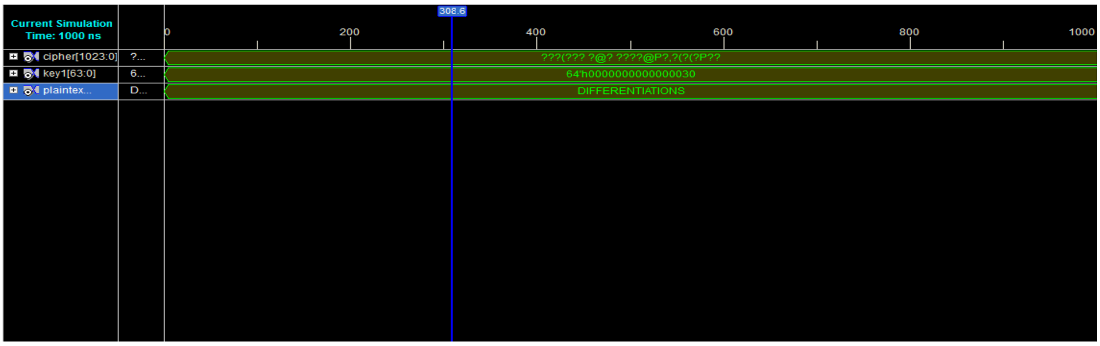
B=2-(48-26)=24=**F**, A=1-(48-26)=23=**E**, N=14-(48-26)= 18=**R,** A=1-(48-26)=5=**E**, J=10-(48-26)=14=**N**, P=16-(48-26) =20=**T**, E=5-(48-26)=9=**I**, W=23-(48-26)=1=**A**, P=16-(48-26)= 20=**T**, E=5-(48-26) =9=**I**, K=11-(48-26)=15=**O**, J=10-(48-26) =14=**N**, O=15-(48-26) =19=**S**.

VI. EXPERIMENTAL RESULTS

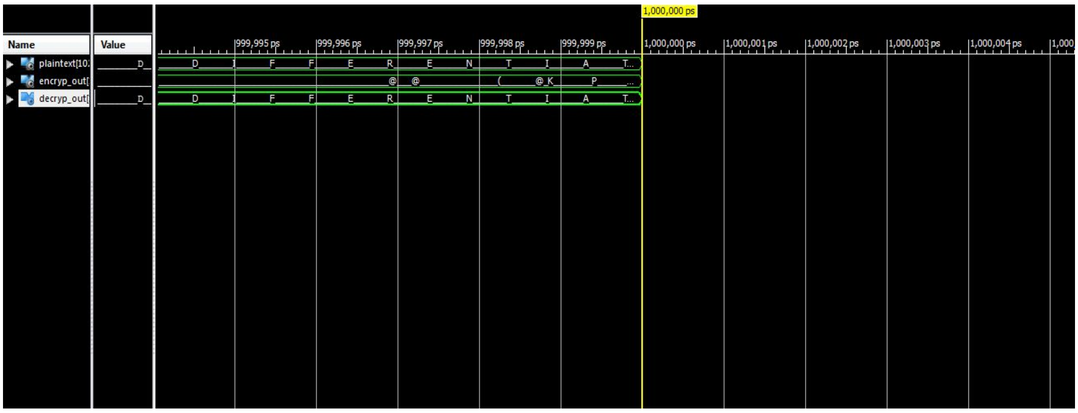
Encryption:



Decryption:



Overall results:



VIII. CONCLUSION AND FUTURESCOPE

The objective of this project is to facilitate the development of applications that include advanced cryptography through above said technique for secured transmission of the messages [1]. Genetic function double point crossover is used to make the technique susceptible from the attacker. Different block division process in binary field of intermediate Cipher confirms the more security of the algorithm. The proposed square matrix model also makes the proposed technique unique. Double point crossover in binary field of intermediate cipher can be made in different point. From the different child blocks fitness test can be applied to take fittest child block to produce more complex cipher text. Distribution of character frequencies will be analyzed for proposed algorithms. Some testing like non-homogeneity between source and encrypted file, chi-squire value test, has to be done to measure the security of proposed technique with well known existing techniques.

Repeated characters in the message will be taken only once while transmission of messages during encryption as well as decryption will be performed with a new algorithm in future. Time complexity for different category of files with existing algorithm in the market will be performed in future. All above said parametric test will confirm the good security in the present age of global communication system.

REFERENCES

[1] S. Som, M. Banerjee, “Cryptographic Technique Using Substitution through Circular Path Followed By Genetic Function”, CCSN-2012, 1st International conference on Computing, Communication and Sensor Network, November 22nd and 23rd, 2012, Roukela, India. Accepted

[2] Poonam Garg, “Genetic algorithms and simulated annealing: a comparison between three approaches for the crypto analysis of transposition cipher” IMT, INDIA-2004.

[3] A.J.Bagnall, “The Applications of Genetic Algorithms in Cryptanalysis”, School of Information Systems, University Of East Anglia, 1996.

[4] N.Koblitz, “A Course in Number Theory and Cryptography‟, Springer-Verlag, New York, Inc., 1994.

[5] Menzes A. J., Paul, C., Van Dorschot, V., Vanstone, S. A., “Handbook of Applied Cryptography”, CRS Press 5th Printing; 2001.

[6] National Bureau Standards, “Data Encryption Standard (DES),” FIPS Publication 46; 1977.

[7] Tragha A., Omary F., Mouloudi A.,”ICIGA: Improved Cryptography Inspired by Genetic Algorithms”, Proceedings of the International Conference on Hybrid Information Technology (ICHIT'06), pp. 335-341, 2006.

[8] Melanie Mitchell, “An introduction to Genetic Algorithms”. A Bradford book.

[9] H. Bhasin and S. Bhatia, “Application of Genetic Algorithms in Machine learning”, IJCSIT, Vol. 2 (5), 2011.

[10] Pisinger D (1999). "Linear Time Algorithms for Knapsack Problems with Bounded Weights". Journal of Algorithms, Volume 33, Number 1, October 1999, pp. 1–14.

[11] Harsh Bhasin, “Use of Genetic Algorithms for Finding Roots of. Algebraic Equations”, IJCSIT, Vol. 2, Issue 4.

[12] *Yu Tak Ma, David K. Y. Yau, Nung Kwan Yip and Nageswara S. V. Rao* “Extended Abstract: Cipher Techniques to Protect Anonymized Traces from Privacy Attacks”, 10th International
Conference, ACNS 2012, Singapore, June 26-29, 2012.