ISSN: 2321-7782 (Online)
e-ISJN: A4372-3114
Impact Factor: 7.327

Volume 6, Issue 4, April 2018

International Journal of Advance Research in Computer Science and Management Studies

Research Article / Survey Paper / Case Study

Available online at: www.ijarcsms.com

Symmetric Key Encryption Algorithm using DNA Sequence

Dr. Asoke Nath[1]
Department of Computer Science
St. Xavier's College(Autonomous)
Kolkata, India

Abedin Dodia[2]
Department of Computer Science
St. Xavier's College(Autonomous)
Kolkata, India

Abstract: Security measures must be incorporated into computer systems whenever they are potential targets for malicious or mischievous attacks. This is especially for systems which handle financial transactions or confidential, classified or other information whose secrecy and integrity are critical. With the need to protect the integrity and privacy of information belonging to individuals and organizations, we have developed this system. The objective of encryption algorithms is to allow the encryption of a message by one person and the decryption of the message by another person .In this project, the plain text is being divided into bits and multiple mathematical functions have been used on it and then converted to DNA form so that even if the intruder intercepts the cipher text ,it will be computationally infeasible to detect the key and convert the DNA sequence back to plain text .That is the encrypted message can be decrypted only if the algorithm and the key are known .In the present paper the authors aims at converting the plaintext into a form unreadable form and then it can be readily transferred across the web and decrypted at the recipient side only by authorized people. It provides an algorithm to encrypt , decrypt or transfer encrypted files without compromising with the integrity and privacy of critical information .In the era of wide area , open distributed systems , this system will help resolve various security issues.

Keywords: Security; encryption; decryption; intruder; DNA; integrity.

I. INTRODUCTION

The current scenario is such that the assurance of security in large open networks has become the need of the hour. With increase in the rate of crimes, one needs to take precautions to protect the data in an efficient manner from all possible attacks. This application plays an important role in providing security for military communications, financial transactions, corporate and political issues. Cryptography is one of the major concerned areas of computer and data security and a very promising direction in cryptography research is known as DNA Cryptography. DNA computational logic can be used in cryptography for encrypting, storing and transmitting the information, as well as for computation. Although in its primitive stage, DNA Cryptography is shown to be very effective. In this the concept of DNA is being used in the encryption and decryption process. The theoretical analysis and implementations shows this method to be efficient in computation, storage and transmission and it is very powerful against certain attacks. This also proposes a unique cipher text generation procedure. In cryptography, cipher text is the result of encryption performed on plaintext using an Algorithm, called a cipher. Cipher text is also known as encrypted or encoded information because it contains a form of the original plaintext that is unreadable by a human or computer without the proper cipher to decrypt it. Decryption, the inverse of encryption, is the process of turning cipher text into readable plaintext.Substitution and transposition are ways of encrypting the plain text. Poly-alphabetic substitution is useful and is less vulnerable to cryptanalysis attacks-both Brute Force and Statistical attacks.   In Brute Force attacks, the cryptanalyst tries to break into the cipher text by trying out all possible keys on the cipher text to generate the plain text. In Statistical attack, the cryptanalyst uses some inherent characteristics of the language to interpret the cipher text. For ex: if the language used is English language, then the most frequently appearing letter is the letter 'E' in an English writing. Other types of attacks may be

Known cipher text, Chosen Plain text, Chosen cipher text, Cipher text only. In contrast, our algorithm proposes to find a solution to these problems of cryptanalysis attacks. We device the algorithm that converts the plain text file into bits and then uses encryption mechanisms on it. This provides the advantage of hiding the inherent characteristics of the language. Thus, even if the same characters appear in the plain text, using the same key also, the cipher text generated is unique in all cases.

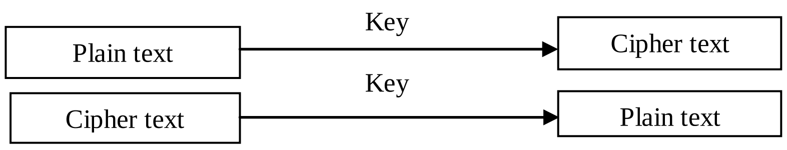

Fig-1: Encryption and Decryption procedure

To provide a highly secured cryptographic algorithm which is practically impossible to break by any intruder even if he is able to intercept it. In the present work the plain text is initially converted to bits and after that bit-wise complement is done on prime positions of the entire bit stream. The entire bit stream is then reversed and again bit complement operation is done on the prime positions. The bit complement is followed by bit-wise XOR operation and then the modified bit streams placed in a 2-dimesional array and perform some bit operations such as left-shift, up-shift, diagonal shift, cycling, right-shift 'n' number of times to make the bit patterns random and to exhaust all the bits. The bit operations are performed a number of times and finally bits were converted into a DNA sequence followed with mutation and then transferred to some output file. The results show that the present method is very much effective to encrypt password, sms of any other confidential message. The addition of DNA cryptography in this project makes the project more secured. The method may also be used to encrypt data in any cell phone/ Smartphone also.

II. DNA Based Cryptography

DNA cryptography is a relatively new paradigm that has attracted great interest in the field of information security. DNA coding technology is used to convert binary data to DNA strings. Since scientists found that binary computers have many physical limitations, especially in data storage and computation, they have concentrated on DNA computers and tried to implement this new science in the information security field. DNA cryptography is a new concept that needs many improvements. Although there are still problems with DNA cryptography, many scientists are trying to solve them because they believe that, with the characteristics of DNA computers they have more advantages than conventional cryptography. DNA coding technology is another concept in cryptography that is intended to encode binary data to a DNA strand and vice versa. Binary data can be encoded in DNA by using sequence of alphabet. It is known that DNA sequences contain four basic letters Adenine (A), Cytosine (C), Guanine (G) and Thymine (T) .

00 converts to A.	01 converts to C.
10 converts to G.	11 converts to T.

For example, a binary string like '00011011' is converted to 'ACGT'. Here the authors have first implemented few bit level encryption techniques and then they converted the bits to DNA sequences to apply genetic operations like mutation.

Mutation:-

After crossover process, mutation process is applied on the chromosome population. Mutation is the alteration of string elements. Two types of mutation are used. In the first one, two mutation points are selected between the entire lengths of the bit string. Then the bits in between these two points are complemented that is, single point mutation changes a 1 to a 0, and vice versa.

Example: Before mutation:

………… 11 0110 010010010010 11 ………….

After mutation:

………… 10 0011 0001110001001 …………..

In the second mutation type, four bits are converted to two bases of DNA

Example: Before mutation:

………… G T A C T G C G A T ……………

After mutation:

………… T G C A G T A T C G…………..

Here each DNA base is treated as two bits and the second bit is complemented for mutation. Thus G='10' is mutated to '11'=T and vice versa.

III. Proposed Algorithm

III.1 ENCRYPTION ALGORITHM :

Step-1: Start

Step-2:Input the file to be encrypted file.

Step-3: Input the key.

Step-4: Convert the bytes of input file into bits and find the size of the bit pattern(n).

Step-5: Perform bit-wise XOR operation on the first and second bit and substitute it in the first bit.

Step-6: Perform bit-wise XOR operation on the first and second bit and substitute it in the second bit.

Step-7: move to next two bits and repeat Step-5 and Step-6 till all the n-bits exhaust.

Step-8: Complement the Prime position bits of the entire bit stream.

Step-9: Reverse the entire bit stream.

Step-10: Again complement the Prime position bits of the entire bit stream.

Step-11: Perform bit-wise XOR operation on bit-1 with bit-n and substitute in n-th bit and so on.

Step-12: Perform bit-wise XOR operation bit-1 with bit-n and substitute in 1-st bit and so on.

Step-13: Repeat step-11 and step-12 till you exhaust all bits.

Step-14:Calculate Encryption Number(e) and Randomisation Number(r) .

Step-15: Store the bits into 2 dimensional array (eXe) and the residual bits into a 1-dimensional array and perform the following shifting operations on the 2 dimensional arrays.

Step-16: Perform bit-wise leftshift(); // To shift all bits in each row by 1 unit on LHS

Step-17: Perform bit-wise upshift(); //To shift all bits in each column by 1 unit in upward direction

Step-18: Perform bit-wise diagonalshift(); // To exchange bits along two diagonals

Step-19: Perform bit-wise rightshift(); // To shift all bits in each row by 1 unit on RHS

Step-20: Perform bit-wise downshift (); // To shift all bits in each column by 1 unit in downward direction

Step-21: After that shift the residual bits into 2-D array and shift the same number of bits from 2-d array to the residual array,the 1-D array is to used in a circular fashion.

Step-22: Repeat step-12 to step-18 as per maxibite(=n/e*e).

Step-23: Perform bitwise XOR on 1st and n/2th bit and substitute in bit 1.

Step-24: Perform bitwise XOR on 1st and n/2th bit and substitute in bit n/2 and then move to next bits from both sides.

Step -25:Repeat above two steps for n/2 times to cover all the entire bit stream.

Step-26: Repeat Step 8 and 9.

Step-27: Take 2 bits at a time and convert it into DNA sequences : 00 $\rightarrow$A, 01$\rightarrow$C, 10$\rightarrow$G, 11$\rightarrow$T,and Store the dna sequence into 2 dimensional array (rXr) and residual bits into a 1-dimensional array and perform the following shifting operations on the 2 dimensional arrays.

Step-28: Perform leftshift();

Step-29: Perform upshift();

Step-30: Perform diagonalshift();

Step-31: Perform rightshift();

Step-32: Perform downshift ();

Step-33: After that shift the residual bits into 2-D array and shift the same number of bits from 2-d array to the residual array,the 1-D array is to used in a circular fashion.

Step-34: Repeat step-26 to step-33 for maxibitr(=n/2*r*r) no.of times.

Step-35:Perform mutation().//mutating the DNA by complementing the second bit of every DNA.

Step-36: Convert the DNA sequence into bits.

Step-37: Perform bit-wise XOR operation on bit-1 with bit-n and substitute in n-th bit.

Step-38: Perform bit-wise XOR operation bit-1 with bit-n and substitute in 1-st bit and then move to 2nd bit and n-1th bit and so on..

Step-39: Repeat step-37 and step-38 till you exhaust all bits.

Step-40: Convert bits to bytes and store it into a file as cipher text.

Step 41: End.

III.2 DECRYPTION ALGORITHM :

Step-1: Start

Step-2: Input the file to be decrypted.

Step-3: Input the key.

Step-4: Convert the bytes of input file into bits.

Step-5: Calculate the size of the bit stream(n).

Step-6: Perform bit-wise XOR operation on bit-1 with bit-n and substitute in n-th bit and so on.

Step-7: Perform bit-wise XOR operation bit-1 with bit-n and substitute in 1-st bit and so on.

Step-8: Repeat step-5 and step-6 till you exhaust all bits.

Step-9: Take 2 bits at a time and convert it into DNA sequences: 00→A, 01→C, 10→G, 11→T.

Step-10:Perform mutation().//To reverse the effect of mutation() in encryption.

Step 11:Calculate Encryption Number and Randomization Number.

Step-12: Store the dna sequence into 2 dimensional array (rXr) and residual bits into a 1-dimensional array and perform the following shifting operations on the 2 dimensional arrays.(r=Randomisation Number)

Step 13: Perform downshift();

Step-14: Perform rightshift();

Step-15: Perform diagonalshift ();

Step-16: Perform upshift();

Step-17: Perform leftshift ();

Step-18: After that shift the residual bits into 2-D array and shift the same number of bits from 2-d array to the residual array,the residual array is to used in a circular fashion.

Step-19: Repeat step-12 to step-18 , ,maxibitr(=n/2*r*r) number of times.

Step 20:Convert DNA sequence to bits with A→00, C→01, G→10, T→11.

Step-21:Perform bitwise XOR on n/2th bit and bit 1 and substitute in bit n/2.

Step-22:Perform bitwise XOR on n/2th bit and bit 1 and substitute in bit n/2th bit,then move to next bits from both sides.

Step:23:Repeat Step-21 and Step-22 till all bits exhaust (or to be precise n/2 no.of times).

Step 24: Store the bits into 2 dimensional array (nXn) and the residual bits into a 1-dimensional array and perform the following shifting operations on the 2 dimensional arrays.

Step-25: Perform bit-wise downshift();// To shift all bits in each column by 1 unit in downward direction

Step-26: Perform bit-wise rightshift();// To shift all bits in each row by 1 unit on RHS

Step-27: Perform bit-wise diagonalshift();//To exchange bits along two diagonals.

Step-28: Perform bit-wise upshift ();//To shift all bits in each column by 1 unit in upward direction

Step-29: Perform bit-wise leftshift();// To shift all bits in each row by 1 unit on LHS

Step-30: After that shift the residual bits into 2-D array and shift the same number of bits from 2-d array to the residual array,the 1-D array is to used in a circular fashion.

Step- 31: Repeat step-18 to step-24 , maxibite(=n/e*e) no.of times.

Step- 32 :Reverse the entire bitsream and then compliment its prime positioned bits.

Step-33:Perform bit-wise XOR operation on bit-1 with bit-n and substitute in n-th bit and so on.

Step-34: Perform bit-wise XOR operation bit-1 with bit-n and substitute in 1-st bit and so on.

Step-35: Repeat step-33 and step-34 till you exhaust all bits.(or n/2 no.of times to be precise.)

Step-36: Complement the Prime position bits of the entire bit stream.

Step-37: Reverse the entire bit patterns.

Step-38: Complement the Prime position bits of the entire bit stream.

Step-39: Perform bit-wise XOR operation on the first and second bit and substitute it in the first bit.

Step-40: Perform bit-wise XOR operation on the first and second bit and substitute it in the second bit.

Step-41: move to next two bits and repeat Step-39 and Step-40 till all the n-bits exhaust.

Step-42:Convert the bit stream into bytes.

Step-43:Store the bytes into the file.

Step-44:End.

III.3 CALCULATION OF ENCRYPTION NUMBER

Let us take an example,say ABCD is our shared secret key,then

Bit form of key=01000001010000100100001101000100.

No.Of bits in key=32

Calculate sum of all even position bits ,to this sum add all the bits in the first half of the key bitstream and store it in Se.

Se=0+0+0+0+0+0+0+1+0+0+0+1+0+0+0+0+0+1+0+0+0+0+0+1+0+1+0

Se=6

Se=Se mod 8=6 %8 =6

Encryption Number=6

III.4 CALCULATION OF RANDOMISATION NUMBER

Let us take an example,say ABCD is our shared secret key,then

Bit form of key=01000001010000100100001101000100.

No.Of bits in key=32

Calculate sum of all odd position bits, to this sum add all the bits in the second half of the key bitstream and store it in Se.

Sr=1+0+0+1+1+0+0+0+1+0+0+1+1+0+1+0+0+1+0+0+0+0+1+1+0+1+0+0+0+1+0+0

Sr=12

Sr=Se mod 8=12 %8 =4 , Encryption Number=4

IV. Results and Discussions

PLAIN TEXT	KEY	CIPHER TEXT
AAAAAAAAAAAAAAAA	AAAA	ã¿« • È□ºØE‘k’
BBBBBBBBBBBBBBBB	Aeiou	f*Á(ì§@ÒÀ«- • »□ýÊ
AAAAAAAAAAAAAAAB	AAAA	ç¿« • È_ºØE‘k²
ABABABABABABABABABAB	ASD	?S›@ºþ´÷«.Ï£àùêR@ •
HE IS A GOOD BOY.	CAPTAIN	¿€$÷ž¢%T€ìµ,Àé,÷
Will this ever get completed ?	abbyman	æ†º:{Â×ÈU854šrÈñžÚ´JŒA/sN
1111222233334444	Guns	Ë[ãBÈpÓûà%µð0äMî

V. Conclusion and Future Scope

In this decade information is being considered to be most valuable entity. So the proper protection of information is highly needed. Huge amount of sensitive information are stored in computer which are now available and accessible through internet. As more information is made available electronically, it can be assumed that threats and vulnerabilities to the integrity of that information will increase as well. We need to protect our vital information from adversaries or any person who may use our important data to benefit them directly. Using this algorithm we secure our information so that anybody cannot hack it easily. This method can be applied on various types of files such as .txt, .doc, .exe etc. In the present method encrypted text cannot be decrypted without knowing the initial key. The present method is free from any kind of brute-force attack or known-plaintext attack and is flexible in terms of key range. If the encrypted text gets modified by the attacker by just one bit then he or she will not be able to get the proper original plain text as the algorithm is sensitive. Every application has its merits and demerits. The project has covered almost all requirements. Further requirements and improvements can easily be done since coding is mainly structured or modular in nature.

By implementing another layer of transposition methods we can extend this algorithm to 3 dimensions. The present algorithm **c**an be applied to files like .txt, .png, .jpg, .dll, .exe etc. But the authors have implemented the method in basic text files and the results were quite satisfactory .This method can also be modified to send only the DNA stream as output of the plain text which will reduce the transfer file size to half of the original.

Acknowledgement

The authors are very much grateful to Prof. Shalabh Agarwal, Head, Department of Computer Science, St. Xavier's College(Autonomous), Kolkata for giving opportunity to in Cryptography and Network Security. One of the authors AN wants to express his sincere gratitude to Rev. Fr. Dr. Dominic Davio, Principal, St. Xavier's College(Autonomous), Kolkata for his all time his support to do research work in Computer Science and Engineering.

References

1. Dr. Asoke Nath, Madhumita Santra, Supriya Maji and Kanij Fatema Aleya,"3-Dimensional Bit Level Encryption Algorithm Ver-1(3DBLEA-1)". International Journal of Innovative Research in Computer and Communication Engineering(IJIRCCE), Vol. 4, Issue 5, MAY 2016.
2. Dr. Asoke Nath,Ayan Ghosh,Enakshi Ghosh and Kanij Jayisha Saha,"3-Dimensional Bit Level Encryption Algorithm Ver-2(3DBLEA-2)". International Journal of Innovative Research in Computer and Communication Engineering(IJIRCCE), Vol. 5 Issue 5, MAY 2016.
3. CRYPTOGRAPHY AND NETWORK SECURITY,2E by Atul Kahate Published By Tata McGraw-Hill Publishing Company Limited.

AUTHOR(S) PROFILE

Dr. Asoke Nath, is the Associate Professor in the Department of Computer Science, St. Xavier's College (Autonomous), Kolkata, Kolkata. Currently he is involved in research work in the field of Cryptography and Network security, Visual Cryptography, Steganography, Green Computing, E-learning and Distance learning, MOOCs, Mathematical modeling of Social Networks, Big data Analytics, data Science, Li-Fi technology. He has published 230 research papers in Journals and Conference proceedings.

Abedin Dodia, is a final year student of B.Sc. Computer Science Honours, Department of Computer Science,St. Xavier's College(Autonomous), Kolkata. Currently he is involved in doing research work in Bit level encryption algorithm.