Volume 1 No. 8, December 2011 ISSN-2223-4985
International Journal of Information and Communication Technology Research

http://www.esjournals.org

Enhanced key Generation Scheme based Cryptography with DNA Logic

[1]Bibhash Roy, [2]Gautam Rakshit, [3]Ritwik Chakraborty
[1, 3]Department of Computer Science and Engineering
Tripura Institute of Technology, Narsingarh, Tripura, India
[2]Department of Computer Science and Engineering
National Institute of Technology, Agartala, Tripura, India

ABSTRACT

DNA Cryptography is the promising methodology of data security in the concerned areas of computer communication and its cryptographic computations can be used for encrypting, storing and transmitting the information. DNA cryptography has shown its effectiveness in the field of secured data transmission and much research work is going on to make the computational process more complex to the unauthorized users. In this paper, an improved key generation based proposal is given using DNA synthesis concepts and the encryption and decryption process is being optimized. The proposed methods are being implemented and analyzed and it shows its efficiency in computation, storage and transmission; and it is more powerful during the decryption process. This paper includes the procedures like key generation, encryption and decryption that are continued and enhanced from our previous proposed work. The performances are finally demonstrated and its implementations are explained and analyzed.

Keywords: *security, encryption, decryption, key generation, cipher text, DNA cryptography*

I. INTRODUCTION

DNA concepts have become a major area of discussion in recent research trends where importance is given on obtaining complex computations in the process of achieving the strong cipher text. The main inherent property of DNA cryptography i.e. massive-parallel computations are being used during evaluation of encryption and decryption of the public and private keys. The resulting complex encryption algorithm used in the cipher text generation is much more complex and stronger than the conventional encryption methods [10]**.** Public Key Cryptography is one set of cryptographic techniques for providing confidentiality, preventing data compromise, detecting alteration of data and verifying its authenticity [6]. By the use of DNA computing, the Data Encryption Standard (DES) cryptographic protocol can be broken [4]. In [1], one-time pad cryptography with DNA strands, and the research on DNA steganography (hiding messages in DNA) was proposed. Essential parts of what we may call data security, specifically confidentiality and authentication, are achieved using cryptography [5]. In our previous work [22],[23] the proposed algorithm takes its basic idea from the way DNA encodes the genetic information in the codons (i.e. each codon holds the information of a particular protein to be synthesized). So using this idea any plaintext can be encoded with a one-time code book. The scheme is principally a symmetric key algorithm, except that the sender initially has only part of the keys, and he generates the rest part of the keys. Symmetric key based encryption is the only way for secure communication between nodes. In this paper, strong procedures for symmetric key generations and sender and receiver side computations are proposed in an optimized way. However, to do that, two nodes should agree upon a common key first. For this, various key distribution schemes have been proposed in the literature. Eschenauer and Gligor [7] proposed a random key pre distribution scheme, referred to as the basic scheme or EG scheme. Based on this scheme, various improvements have been proposed in the literature [2], [3], [8], [9], [11]. The proposed enhanced procedures are being implemented using java platform the results are analyzed both in tabular and graphical form.

II. RELATED STUDIES

As a basic storage unit for living cells, DNA's main functionality is to absorb and transmit the data of life for billions years. Due to its advantage of being occupied in very large numbers, near about 10 trillions in a marble sized space, DNA molecules can process large amount of data simultaneously; theoretically, they can perform massive parallel computations in a small space at one time. DNA computing, also known as molecular computing, offers a completely new scenario for computation. The idea behind DNA computing technique is to simulate the arithmetic and logic operations in a DNA strand form. The main operations of DNA Synthesis are designing and restructuring information in DNA sequence form. In this process, designing and synthesizing information in the DNA sequence form is an important process where wrong design might leads to wrong result.

Large number of researchers took an initiative to implement DNA computing in the solutions of applications like cryptography, scheduling, clustering, encryption, forecasting and even in the field of wireless sensor networks. Some other researchers employed DNA Computing in information security technology [14]. For example, Boneh et al.[19] and Adleman et al. [18] have proposed a model to break a Data Encryption Standard(DES) as a alternative way for encryption data

technology. DNA cryptography has been proposed by Gehani et al. [15], Kartalopoulos [16] and Tanaka et al. [17] as a new born cryptography field. Beside DNA cryptography and DES, there are some development in DNA steganography and DNA certification. Recently, DNA is employed as an intrusion detection model for computer and telecommunication systems by Boukerche et al. [21]. Among all DNA computing models proposed in this research area DNA certification is most matured and the application is most widely studied [18].

III. PROPOSED STRONGER KEY BASED CRYPTOGRAPHY

In this proposed paper, the format of cipher text is kept as our previous proposal [22], where the primary cipher text obtained after encoding is being divided into three unequal parts and then extra parameters such as primer code, file type code, integrity code, and authentication code are added in between parts of the cipher text to obtain the final cipher text.

Format of Cipher Text[22]

From plain text (PT) the primary cipher text (CT) is obtained by using the encryption algorithm and the 1st level key (PK1). The following steps are to be followed to obtain the final cipher text.

Step-1: Encrypt the plain text with 1st level key (PK1).
Step-2: Divide the primary cipher into three unequal parts.
Step-3: Attach AUT, INTR, FT, SPM, EPM with the above CTB's. After encrypting CTB's using level2 private keys(which include the information about the introns(AUT,FT ,ETC) positions and the length of the SPM and EPM).

Sender's side computation:

Procedure to prepare the substituent list (array):

Input: a. A byte array substituent of size = 256.
b. Key1 (0 to 255).
Output: Filling the array with 256 different byte value using key1 .These bytes will be used for substituting purpose.
Begin
Step-1: Let a Byte variable J= -128.
Step-2: Let Integer variable L=0.
Step-3: Repeat through step8 for I=0 to 255.
Step-4: L=I+Key1;
Step-5: If (L<256) then substituent [L] =J.
Step-6: Else substituent [L-256] =J.
Step-7: J=J+1.
Step-8: I=I+1.
End

Procedure for Encryption:

Input: a. plain text file.
b. Key2 =400(range is 255 to 9999999).
c. A List of auto generated substituent using procedure A.
Output: cipher text.
Begin
Step-1: Repeat through Step 8 for each individual byte of the plain text.
E.g Let ByteValue= 82 be the Original Byte before Encoding.
Step-2: ByteValue=ByteValue+128(to ensure that negative bytes are also converted into positive values).
E.g ByteValue=82+128=210.
Step-3: Add +128 to byte value 82 so as to ensure that negative bytes are also converted into positive values.
Step-4: Calculate q= (integer) ByteValue/16 ,e.g q=210/16=13. And r= ByteValue %16,e.g r=210%16=2.
Step-5: Calculate index = r*16+q
e.g. index=2*16+13=45.
Using 'index' we initiate another round by fetching the data at substituent [index] from
Step-6: Let PreFinalIndex= array [index] = -27 and add +128 again to ensure Positive PreFinalIndex = 101.
Step-7: Calculate a new index, we name it as the 'FinalIndex' whose content will be the cipher byte as:
i.finalindex= (prefinalindx + key2) % 256.
e.g finalindex= (101 + 400) % 256=245.
ii.Cipherbyte= substituent [finalindex].
e.g. Let Cipherbyte = -83.
Step-8: Cipher byte is ready to be transmitted across.
End

Receiver's side computation:

Procedure to prepare the substituent list (array):

Input: a. A byte array substituent of size = 256.
b. Key1 (0 to 255).
Output: Filling the array with 256 different byte value using key1 .These bytes will be used for substituting purpose.
Begin
Step-1: Let a Byte variable J= -128.
Step-2: Let Integer variable L=0.
Step-3: Repeat through step8 for I=0 to 255.
Step-4: L=I+Key1;
Step-5: If (L<256) then substituent [L] =J.
Step-6: Else substituent [L-256] =J.
Step-7: J=J+1.
Step-8: I=I+1.

End

Procedure for Encryption:

Input: a. cipher text file.
b. Key2 =400(range is255 to 9999999).
c. A List of auto generated substituent using procedure A.
Output: plain text file.
Begin
Step-1: Repeat through Step 11 for each individual byte of the plain text. e.g Let the cipher byte read is -83
Step-2: Repeat through Step for I=0 to 255
Step-3: If (CipherByte =Substituent[I]) then prefinlindx = I-key2, e.g. here I will be 245 therefore new index be PreFinlindex=245 - 400= -155.
Step-4: Repeat through step 5 while(prefinlindx<0)
Step-5: Prefinlindx = prefinlindx + 256.
e.g. Prefinlindx= -155+256=101.
Step-6: Prefinlindex= Prefinlindex-128
e.g. Prefinlindx= 101-128= -27.
Step-7: Repeat through step 11 for K=0 to 255
Step-8: if(prefinlindx=Substituent[K])then index=K
e.g. Let the index of -27 be 45
Step-9: Evaluate r=index/16 e.g r=45/16=2.
and q=index%16 e.g q=45%16=13.
Step-10: Evaluate PreOriginalbyte=q*16+r,
e.g. PreOriginalbyte=13*16+2=210
Step-11: Originalbyte=Originalbyte-128,
e.g. Originalbyte=210-128=82
End

IV. KEY STRENGTH ANALYSIS

The level-2 private key gives the strength on cipher text by adding the primers & the positions of introns [22] which are inserted within the cipher text at positions as described by the Level-2 key. The sender's file length is chosen as level-2 key. Introns positions are taken on the basis of the individual digits of the file length. But the sender should not send the raw file length(i.e level2 key) to the receiver even though it is sending through secret channel so it has to be encoded using the following procedure:

Step-1: First the receiver will send a number through private channel or public channel. This key should be any positive number between the ranges 1 to 255. Now the sender will perform the following task using this number.
Step-2: Let P be an array which will hold the secondary level of keys.
Step-3: Take a variable and initialize it with a number which is the file length.
Step-4: Repeat through the following steps for 1 to number of digits in N.
Step-5: Perform digit wise X – OR of N from left to right (i.e. from MSB to LSB) (Fig-4). It is done in the following manner:
Step-6: N = rn-1 rn – 2…….r1.
Step-7: PK = rn
Step-8: P[i++] = rn.
Step-9: Send the array P (level2 private key) to the receiver to get the file length by applying reverse procedure of the above (*Note:* The sender will send both level1 and level2 private keys to the receiver in a digest form).

A. Analysis for Brute-force attack only

The key generation scheme uses three different keys at different levels of computations.
Key1 ranges from 0 to 255 i.e. require 8 bits to represent.
Key2 ranges from 255 to 999999999 i.e. it requires 30 bits for representation.
key3 (8 bits) is used for left and right padding of the encrypted text to apply diffusion.
So in total the key size is 46 bit i.e. is there are 2^46 = 7.0368744 x 10^13 possible keys.
Now assume that a hacker have a very fast computer using which our decryption algorithm can be executed in 1 micro second for all possible key trials. Even if he tries half the set of keys then also he is quite successful in decrypting.
But then also the hackers require more than half year decrypting the cipher text which is show as below:
In one second = 1000000 possible key trials
In one hour = 36 x 10^8 possible key trials
In one day = 864 x 10^8 possible key trials
In one year = 3.1536 x 10 ^13 possible key trials (less than half of the total key set).

V. PERFORMANCE EVALUATION

The proposed procedures are implemented in java platform for its platform independent property and available in-built cryptography functionalities. The procedures are implemented successfully for the specified sized input plain texts. Initially, there were constraints for large files such as images or videos where the required primary memory of the system could create a problem in the execution and conversion of the plain text into cipher text. But, later on that problem are also resolved by simply dividing the large file into fixed sized sub-files and then performing the swapping while encoding and decoding. The following table represents the data sets that are obtained during the testing and analysis of the proposed procedure.

TABLE 1: Datasets for Performance Analysis

TESTS	File size (KB)	Cipher size (KB)	Encrypt time (sec)	Decrypt time (sec)
doc	370	370	7	8
pdf	524	524	9	9

zip	210	210	4	4
txt	89	89	3	1
gif	2750	2750	44	54
jpeg	3930	3930	62	62

A. Length Analysis

Based on the datasets obtained in table-I, the following graph indicates that the size of the cipher text is almost same in every test cases. It also shows that a large file can also be encrypted using the proposed procedure.

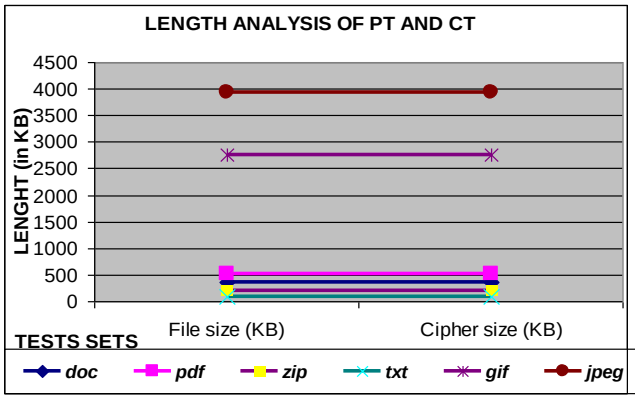

Figure 1: Length analysis of PT and CT

B. Time Analysis

The following graph obtained from the datasets of table-1 represents the comparisons between the encryption and decryption time of a specified file size. This graph shows that the decryption time is much lesser than the encryption time. It can also be inferred that there are no drastic changes in the encryption and decryption time even if the file size increases to a large extent.

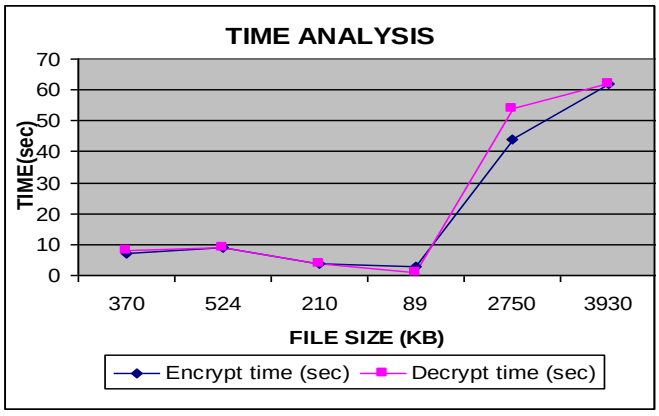

Figure 2: Encrypt time vs Decrypt time based on file size

VI. CONCLUSION

The proposed cryptography procedure is the enhanced proposal of our previous research work where the concept DNA cryptography and DNA computing was implemented in our proposed procedures. In our proposal, the key strength is increased to three level keys where key2 is the strongest one. The proposal can surely be enhanced with much more advanced concepts such as realization in several security technologies of mobile ad hoc networks where limited bandwidth constraint can be overcome as in our proposed procedure the plaint text and cipher text are of same size. There are a lot of opportunities in expanding and manipulating DNA characteristics and operations to solve real application especially industrial engineering and management engineering problems. Although this method is efficient, and it is powerful against certain attacks; the two different rounds of algorithms makes it stronger and faster in execution. The partial information contained in the cipher text makes the method much stronger. Small amount of diffusion is added to make it stronger, but a percentage of increase of cipher size will be there.

REFERENCES

[1] Ashish Gehani, Thomas LaBean and John Reif. *DNA-Based Cryptography*. DIMACS DNA Based Computers V, American Mathematical Society, 2000.

[2] R. Blom. An optimal class of symmetric key generation systems. *Advances in Cryptology: Proceedings of EUROCRYPT 84 Springer- Verlag*, 209/1985:335 – 338, 1985.

[3] C. Blundo, A.D. Santis, A. Herzberg, S. Kutten, U. Vaccaro and M. Yung. Perfectly-secure key distribution for dynamic conferences. *Lecture Notes in Computer Science*,, 740:471–486, 1993.

[4] Dan Boneh, Cristopher Dunworth, and Richard Lipton. *Breaking DES Using a Molecular Computer*. Technical Report CS-TR-489-95, Department of Computer Science, Princeton University, USA, 1995.

[5] Garfinkel Simson, Web Security, Privacy & Commerce, 2nd Edition, O'Reilly Publisher, November 2001

[6] Kahn D., The Codebrakers, McMillan, New York, 1967

[7] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. *Proceedings of the 9th ACM conference on Computer and communications security, Washington, DC, USA*, pp. 41–47, November 18-22 2002.

[8] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. *Proc. of the 9th ACM conference on computer and communications security(CCS'03)*, Oct. 2003.

[9] S. Zhu, S. Xu, S. Setia and S. Jajodia. Establishing pairwise keys for secure communication in ad hoc networks: a probabilistic approach. *Proceedings of the 11th IEEE International Conference on Network Protocols (ICNP'03)*, Nov. 2003.

[10] Gehani Ashish, La Bean, Thomas H. Reif, JohnH, "DNA-Based Cryptography", Department of Computer Science, Duke University, June 1999

[11] W. Du, J. Deng, Y. S. Han and P. K. Varshney. A pairwise key predistribution scheme for wireless sensor networks. *Proceedings of the Tenth ACM Conference on Computer and Communications Security (CCS 2003)*, pp. 42–51, October 2003.

[12] Guangzhao Cui Limin Qin Yanfeng Wang Xuncai Zhang. An encryption scheme using DNA technology. *Bio-Inspired Computing: Theories and Applications,2008. BICTA 2008. 3rd International Conference on Publication Date: Sept. 28 2008-Oct. 1 2008 ISBN: 978-1-4244-2724-6, page(s):37-42;Adelaide, SA.*

[13] Guangzhao Cui Limin Qin Yanfeng Wang Xuncai Zhang; Information Security Technology Based on DNA Computing; *Anti-counterfeiting, Security, Identification, 2007 IEEE International Workshop on 16-18 April 2007,page(s): 288-291, ISBN: 1-4244-1035-5, Location: Xiamen, Fujian.*

[14] William Stallings. *Cryptography and Network Security, Third Edition, Prentice Hall International, 2003.*

[15] A. Gehani, T. LaBean and J. Reif, DNA-based cryptography, *Lecture Notes in Computer Science, vol.2950, pp.167-188, 2004.*

[16] S. V. Kartalopoulos, DNA-inspired cryptographic method in optical communications, authentication and data mimicking, *Proc. of the IEEE on Military Communications Conference, vol.2, pp.774-779, 2005.*

[17] K. Tanaka, A. Okamoto and I. Saito, Public-key system using DNA as a one-way function for key distribution, *Biosystems, vol.81, no.1, pp.25-29, 2005.*

[18] G. Cui, L. Qin, Y. Wang and X. Zhang, Information security technology based on DNA computing, *Proc. of the 2007 IEEE International Workshop on Anti-counterfeiting, Security, Identification, Xiamen, China, pp.288-291, 2007.*

[19] D. Boneh, C. Dunworth and R. Lipton, Breaking DES using a molecular computer, *Technical Report, CS-TR-489-95, Princeton University, 1995.*

[20] G. Xiao, M. Lu, L. Qin and X. Lai, New field of cryptography: DNA cryptography, *Chinese Science Bulletin, vol.51, no.12, pp.1413-1420, 2006.*

[21] A.Boukerche, K. R. L. Juca, J. B. Sobral and M. S. M. A. Notare, An artificial immune based intrusion detection model for computer and telecommunication systems, *Parallel Computing, vol.30,no.5-6, pp.629-646, 2004.*

[22] Bibhash Roy, Gautam Rakshit, Pratim Singha, Atanu Majumder, Debabrata Datta, "A DNA based Symmetric key Cryptography"- ICSSA-2011, 24-25 Jan'11, G H Patel College of Engineering & Technology, Gujarat, India.

[23] Bibhash Roy, Gautam Rakshit, Pratim Singha, Atanu Majumder, Debabrata Datta, "An improved Symmetric key cryptography with DNA Based strong cipher"- ICDeCom-2011, Feb' 24-25'2011, pp.1-5,2011 Birla Institute of Technology, Meshra, Ranchi, Jharkhand, India.