Advanced Symmetric Key Cryptosystem using Bit and Byte Level Encryption Methods with Feedback

A. Prabal Banerjee[1] **and B. Asoke Nath**[2]
[1,2] Department of Computer Science, St. Xavier's College (Autonomous), Kolkata, India
[1]mail.prabal@gmail.com, [2]asokejoy1@gmail.com

Abstract - *In the present paper the authors have introduced a new symmetric key cryptographic method where the authors have applied bit level and byte level generalized modified vernam cipher method followed by bit-wise transposition method. Nath et al already developed method which was a combination of generalized bit level and byte level encryption methods. In the present method the authors have added one more encryption method that is bit-wise columnar transposition method. Nath et al also developed bit level encryption standard(BLES) Ver-I and Ver-II where they have used extensive bit level permutation, bit exchange, bit xor and bit shift encryption method. In the present study the authors have used both bit level generalized vernam cipher method and after that byte level vernam cipher method using feedback and finally the output is passed through bit-wise columnar transposition method to make the whole system more secured. The introduction of feedback in both bit level as well as byte level vernam cipher method prevents from standard attacks such as differential attack or known plain text attack. In the present paper the authors have used random key generator to construct the keypad for vernam cipher method. The present method will be most effective for encrypting short message, password, any confidential key etc.*

Keywords: BLES, bit-wise columnar transposition, differential attack, vernam cipher method

1 Introduction

In Internet when a person sends some confidential data from one computer to another computer then there is no guarantee that the confidential message can not be intercepted by any unwanted intruder. This is because the internet is now so open that any body can access any information and sometimes he/she can divert/forward to anyone also. So the security of data is now has a big question mark. Any kind of private data should not be sent in raw form from one computer to another. The private/confidential data must be encrypted first and then it should be sent over the internet. In the modern days e-mail is one most important method to send data from one machine to another or from one person to another person. But the question is how secured is this method. The hackers have made many packages and they have uploaded in various websites to break any password. So it is not at all a difficult task to break any password of any e-mail especially if it is very weak password. Once the password is hacked then anything can be done from that e-mail. So the e-mail must not contain any confidential information in raw form. The hackers are always try hack the password of e-mail. Anytime the disaster may come. So if the data is confidential/private then it must be encrypted first with some good encryption method and then it can be sent to someone. The security or the originality of data has now become a very important issue in data communication network. It is now a common practice in any academic institution to send marks, attendance or question papers, bank statement over e-mail. But this method is not fully secured as anybody can intercept the data from internet and misuse it. It is not at all difficult task for a hacker to intercept an e-mail and retrieve the confidential data especially if it is not encrypted. It must be ensured that in any kind of e-business, air or railway reservation system or in credit card or debit card system the data should not be tampered or intercepted by an unauthorized person. The disaster may happen in any corporate sector, business house when the data is sent from one computer to other computer in an unprotected manner. To overcome this problem one has to send the encrypted text or cipher text from client to server or to another client instead of sending in unencrypted form. To protect data from intruder or hacker now network security and cryptography is an emerging research area where the programmers are trying to develop some strong encryption algorithm so that no intruder can intercept the encrypted message. The cryptography methods can be divided into two categories : (i) symmetric key cryptography where one key is used for both encryption and decryption purpose. (ii) Public key cryptography where two different keys are used one for encryption and the other for decryption purpose. In symmetric key we have to maintain only one key and hence the key management is simple . In public key cryptography we maintain two keys one is public key which is known to everybody and that can be used for encryption purpose and there is another key called private key which is a secret key and that is used for decryption purpose only. In the present work the authors are proposing a symmetric key method where they have used bit level and byte level modified generalized vernam cipher method using feedback method followed by randomized bit level columnar transposition method The present method can be applied in corporate sectors, business house, academic institutions, Defense network etc. The present method performs the following:

The user has to enter some secret key and which is used to generate MSA matrix.
The program then generates all the required anagrams sufficient to encrypt all of the plaintext.
Then Bit level vernam cipher with feedback is applied , reverse file, apply again.
After that Byte level vernam cipher is applied with feedback. Reverse the file and again applied the same method.
Finally randomized Bit level transposition method applied.

The final bits were converted to bytes and write on to some output file.

The multiple key generation from a set of random characters and both bitwise and bytewise encoding make the system very secure.

2 Encryption Algorithm

The present method is dependent both on the text-key and the plaintext file size. From the text-key a randomization matrix is generated using the method developed by Nath et al(1). The algorithm of bit-level and byte level generalized vernam cipher method and bitwise columnar transposition is given as follows:

Step 1: Call Bitwise_Encrypt()

Step 2: Call Bytewise_Encrypt()

Step 3: Call Transpose_Encrypt()

Step 4: Exit

2.1 Function Bitwise_Encrypt ()

Step 1: Input a key string K

Step 2: Generate a 16x16 matrix (mat[][]) using the MSA algorithm for the key string K

Step 3: Input Filename P which is the plaintext on which the encryption is to be applied

Step 4: size=no. of bytes in file P, rand_no=1

Step 5: If size>=factorial of rand_no, rand_no=rand_no+1, repeat step 5

Step 6: Take 'rand_no' amount of characters from mat[][] and put in string buf

Step 7: Find all anagrams of buf and put in file F

Step 8: Call Encrypt_byte(P,F,mat)

Step 9: Reverse the contents of A into which function Encrypt_byte has written

Step 10: Call Encrypt_bit(A,mat)

Step 11: limit=number of bytes in file B

Step 12: i=0

Step 13: if i>=limit/8, goto step 23

Step 14: add=j=0

Step 15: if j>=8, goto step 20

Step 16: Read a character from B and store into ch

Step 17: add=add+(ch-48)*power(7-j)

Step 18: j=j+1

Step 19: Goto step 15

Step 20: Convert add to character and print into file C

Step 21: i=i+1

Step 22: Goto step 13

Step 23: Return control to calling function

2.2 Function Bytewise_Encrypt (File C)

Step 1: limit=number of bytes in File C, k=carry=0

Step 2: if k>limit , goto step 11

Step 3: Read a character from file C and store to ch

Step 4: ch=ch+mat[i][j]+carry

Step 5: Write ch to file D

Step 6: carry= ch % 256

Step 7: j=j+1, k=k+1

Step 8: if j=16, i=i+1 and j=0

Step 9: if i=16, i=0

Step 10: Goto step 2

Step 11: Exit

2.3 Function Encrypt_byte (File P, File F, mat[16][16])

Step 1: Find the number of bytes in the plaintext file P on which the encryption is to be applied. Let it contain no_of_bytes.

Step 2: carry=0

Step 3: Read a character from file F and store to ch

Step 4: Call char_to_bit(ch,key_bit)

Step 5: Read a byte ch from P

Step 6: Call char_to_bit(ch,text_pattern)

Step 7: k=0

Step 8: if k>=8, goto step 16

Step 9: add=text_pattern[k]+key_bit[k]+carry

Step 10: if add=1 or add=3, cipher_bit=1

else cipher_bit=0

Step 11: if add>=2, carry=1

else carry=0

Step 12: If carry=0, carry=cipher_bit

Step 13: Print cipher_bit into file A

Step 14: k=k+1

Step 15: Goto Step 8

Step 16: no_of_bytes=no_of_bytes-1

Step 17: If no_of bytes>0, goto step 3

Step 18: Return control to calling function

2.4 Function Encrypt_bit (File A, mat[16][16])

Step 1: Find the number of bytes in A on which the encryption is to be applied. Let it contain no_of_bytes.

Step 2: carry=0

Step 3: Read a character from file F and store to ch

Step 4: Call char_to_bit(ch,key_bit)

Step 5: n=0

Step 6: if n>=8, goto step 11

Step 7: Read a char from A

Step 8: text_pattern[n]=ch-48

Step 9: n=n+1

Step 10: Goto step 6

Step 11: k=0

Step 12: if k>=8, goto step 16

Step 13: add=text_pattern[k]+key_bit[k]+carry

Step 14: if add=1 or add=3, cipher_bit=1

else cipher_bit=0

Step 15: if add>=2, carry=1

else carry=0

Step 16: If carry=0, carry=cipher_bit

Step 17: Print cipher_bit into file B

Step 18: k=k+1

Step 19: Goto Step 8

Step 20: no_of_bytes=no_of_bytes-8

Step 21: If no_of bytes>0, goto step 3

Step 22: Return control to calling function

2.5 Function power (integer p) -- Function returns 2 to the power p

Step 1: ans=2

Step 2: if p!=0, return 1

Step 3: p=p-1

Step 4: if p=0, goto step 7

Step 5: ans=ans*2

Step 6: Goto step 4

Step 7: return ans

Step 8: Return control to calling function

2.6 Function char_to_bit (integer c, integer a[]) --Function changes a character to its corresponding bit pattern

Step 1: i=0

Step 2: if i>=8, goto step 4

Step 3: if ((ch)AND(1<<i))>0, a[7-i]=1

else a[7-i]=0

Step 4: Return control to calling function

2.7 Function Transpose_Encrypt()

Step 1: Take a file A. Say it has n characters.

Step 2: Define a n x 8 table. i=0.

Step 3: Read a character ch from file A.

Step 4: Convert ch into its corresponding bit pattern and save it in i^{th} row of the table.

Step 5: Take 8 numbers from MSA table such that each number modulo 8 is unique and covers whole of range 0 to 7. Let the numbers be $M_1, M_2, \ldots. M_8$

Step 6 : For each of i from 1 to 8, choose M_i^{th} column of the table and save the contents into a temporary file T.

Step 7: Read 8 integers from file T. Compute its equivalent binary. Save into final file F.

Step 8: Repeat step 7 until whole of file T is read.

Step 9: Return control to calling function.

3 DECRYPTION ALGORITHM

Step 1: Call Transpose_Decrypt()

Step 2: Call Bytewise_Decrypt()

Step 3: Call Bitwise_Decrypt()

Step 4: Exit

3.1 Function Bitwise_Decrypt (File P)

Step 1: Input a key string K

Step 2: Generate a 16x16 matrix (mat[][]) using the MSA algorithm for the key string K

Step 3: size=no. of bytes in file P, rand_no=1

Step 4: If size>=factorial of rand_no, rand_no=rand_no+1, repeat step 4

Step 5: Take 'rand_no' amount of characters from mat[][] and put in string buf

Step 6: Find all anagrams of buf and put in file F

Step 7: Call Encrypt_byte(P,F,mat)

Step 8: Reverse the contents of B into which function Encrypt_byte has written

Step 9: Call Encrypt_bit(B,mat)

Step 10: limit=number of bytes in file B

Step 11: i=0

Step 12: if i>=limit/8, goto step 22

Step 13: add=j=0

Step 14: if j>=8, goto step 19

Step 15: Read a character from B and store into ch

Step 16: add=add+(ch-48)*power(7-j)

Step 17: j=j+1

Step 18: Goto step 14

Step 19: Convert add to character and print into file C

Step 20: i=i+1

Step 21: Goto step 12

Step 22: Exit

3.2 Function Bytewise_Decrypt (File P)

Step 1: Input Filename P which is the plaintext on which the encryption is to be applied

Step 2: limit=number of bytes in File P, k=carry=0

Step 3: if k>limit , goto step 12

Step 4: Read a character from file P and store to ch

Step 5: ch=ch - mat[i][j] - carry

Step 6:if ch<0, ch=ch+255

Step 7: carry= ch , Store ch in File A

Step 8: j=j+1, k=k+1

Step 9: if j=16, i=i+1 and j=0

Step 10: if i=16, i=0

Step 11: Goto step 3

Step 12: Exit

3.3 Function Transpose_Decrypt()

Step 1: Take file A on which transposition is to be applied and decrypted. Let it contain n characters

Step 2: Take tables T1 and T2 of size n x 8.

Step 3: Read a character from file A. Take its corresponding bit pattern and save in T1 by filling it column wise. Repeat the process until all the characters are read of file A.

Step 4: Take 8 numbers from MSA table such that each number modulo 8 is unique and covers whole of range 0 to 7. Let the numbers be $M_1, M_2, \ldots. M_8$

Step 5: For each of i from 1 to 8, choose M_i^{th} column of the table and copy the contents of the column into $(i-1)^{th}$ column of T2.

Step 6: Starting from row 0, take all rows, one at a time. Compute the corresponding byte for the bit pattern (each row is a bit pattern) and save in file F.

Step 7: Return control to calling function.

4 Randomixzation Of Matrix Using Meheboob, Saima & Asoke(Msa) Randomization Method

We first create a square matrix of size n x n where n can be 4, 8, 16 and 32. First we store numbers 0 to (n*n-1). We apply the following randomization techniques to create a random key matrix. The detail description of randomization methods is given by Nath et.al[1].

The following Randomization methods were applied on initial key matrix to obtain a randomized key matrix:
Step-1: call Function cycling()
Step-2: call Function upshift()
Step-3: call Function downshift()
Step-4: call Function leftshift()
Step-5: call Function rightshift()

5 Results And Discussion

The present method applied on various text files such as all characters ASCII ‘0’, all characters ASCII ‘1’ and the frequency distribution of the encrypted text is shown the below spectral graph.

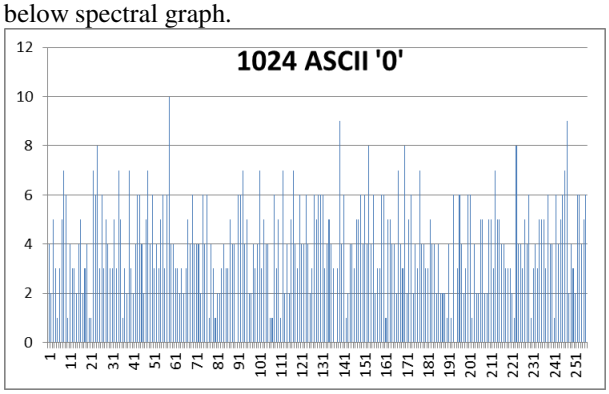

Fig-1: Frequency Spectral analyses of Plain Text file containing 1024 ASCII ‘0’ and Key=’a’.

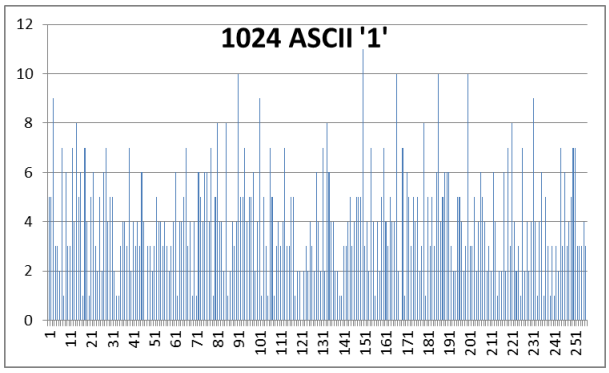

Fig-2: Frequency Spectral analyses of Plain Text file containing 1024 ASCII ‘1’ and Key=’a’.

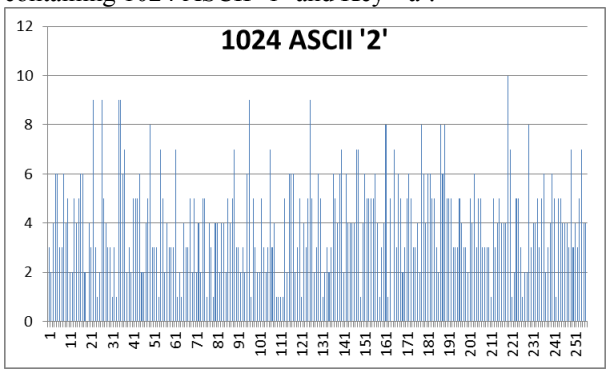

Fig-3: Frequency Spectral analyses of Plain Text file containing 1024 ASCII ‘2’ and Key=’a’.

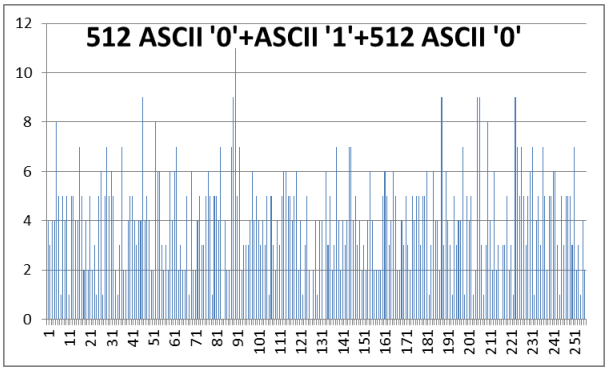

Fig-4: Frequency Spectral analyses of Plain Text file containing 512 ASCII ‘0’,then 1 ASCII ‘1’ and then 512 ASCII ‘0’, encrypted using Key=’a’.

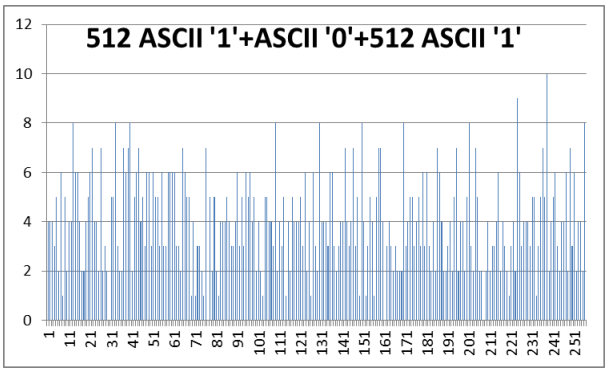

Fig-5: Frequency Spectral analyses of Plain Text file containing 512 ASCII ‘1’,then 1 ASCII ‘0’ and then 512 ASCII ‘1’, encrypted using Key=’a’.

Table 1: A Plain Text file containing a paragraph and its encrypted file after bit and bytewise encryption.

St. Francis Xavier is the Patron of St. Xavier's College. There is not an educated Indian who has not heard the name of Francis Xavier. It is to India that Ignatius of Loyola, the Founder of the Society of Jesus, sent his greatest son, Francis Xavier in 1542. Xavier was a zealous "missionary on the move". He sailed to Malacca and Japan in 1549 where he spent two and a half years. In April 1552 he set sail to China via Malacca from Goa, never to return alive. He died at Sancian, a small island facing china, on 2	淋□六뱄朕 엟뽄□⌘طي녡□ ᵑ붨洲2Ⅱ溦鰆韕 **省柾礹**헄楢 슮옕갾瘁ɔᵊᵣɳ㊟ □ 銮鹹惶껔 鶴몦聂굓术砍律□**裰**剔坡ᵠ□ᵒe ᵇᶜ뫓**薟**辱먓솄啶뭬ᵑ̃ 辙” 될壏□ □**佈郴**叉乩荟꾖匴뒀몹ɱ**良**□ 챘↩泺 冯 乸쾘**蚡**ʜ**熱**⁙갇悰 □ʂ孇鼯剚澡핦병ᵴ̂ 攎葷ᵃᵚ爆ᔆ 썴繃穭 **邵奸崇**뎼걆 □ᵚ늅嬇 孫썽⁙**微**彙ᵀ 긻蹜뙷ᵀ鑣غ· ❽胎ᵴ迪**蒞** ;퐍**疊**□턍옏聜ᵩ □**淌**ɯ홑棤蔌÷ ᵠ 垒瀎口聂筴 荐록舸繿 鞡!ᅇ璞斀麟챐□□ 鲔□**檬**쀗潔砭鑀퐨焚 ᵬ 膹□ **鹿**鷂릠 **嚮**儡횘**踐**졋置夺ϖ爇 **漸**哑屧ㄴ愙奚嫂▧聂樸ص늲 玑썿洛좙렚쏦늜□**晢蘘**□**醭梓** 嶬摠棱썬벍薂 琲 魆7□峬靼

December 1552. Wherever he went, he plunged himself into charitable and pastoral work preaching the message of God's love to people. He worked in India for 10 years from 1542 to 1552, called the Xaverian decade.	藝呶□캤浢法뒨曬홱□蜆멮 煁楑鬸 复σ嗡黫ﻸ聊礵ﾞ □– 됆듯敎□□褆汽坺□ 藒□犷十 □ 牡 搚:弓妬□뻺媧 밞힃 씣□□¡□꾖浊1Ⅱ를 □餟懂紕ъ □ᥒ폳⁙粖헧◣ 嶕챰꺫샥ᄃ鈁ᄆ 瓁棎□욘ʃ□竷쵪칂孃 獝뀌歞 꾁柴 □öz麭□□뤘貑 췔礼ᇹ 兀◌얼渺

6 Conclusion And Future Scope

The encrypted text cannot be decrypted without knowing the exact initial random matrix. The size of random matrix taken is 16x16. The numbers in 16x16 may be arranged in 256! Ways. To complete the whole process the authors have chosen any of the random matrix to perform bit exchange method and there is no similarity between any two matrices and even if there is then it is very hard to find out the similar ones. The order of encryption applied in this paper is bit-level first and then byte level. Multilevel applications of these techniques in different order will yield considerably different results. The spectral analysis shows that the present method is free from standard cryptography attacks namely brute force attack, known plain text attack and differential attack. The present method will be most effective to encrypt short message such as SMS in mobile phone, password encryption and any type of confidential message. If the file size is large then the present method will take more time to encrypt. So therefore our proposed method may be used in defense systems, Banking systems, Sensor networks, Mobile computing etc. The present method may be further upgraded by introducing bit level bit exchange method which is used in BLES Version-I.

7 Acknowledgement

The authors are very much grateful to the Department of Computer Science for giving the opportunity to work on symmetric key Cryptography. One of the authors (AN) sincerely expresses his gratitude to Fr. Dr. Felix Raj, Principal of St. Xavier's College (Autonomous) for giving constant encouragement in doing research in cryptography.

8 References

[1] Symmetric Key Cryptography using Random Key generator: Asoke Nath, Saima Ghosh, Meheboob Alam Mallik: "Proceedings of International conference on security and management(SAM '10)" held at Las Vegas, USA July 12-15, 2010), Vol-2, Page: 239-244(2010).

[2] Advanced Symmetric key Cryptography using extended MSA method: DJSSA symmetric key algorithm: Dripto Chatterjee, Joyshree Nath, Soumitra Mondal, Suvadeep Dasgupta and Asoke Nath, Journal of Computing, Vol 3, Issue-2, Page 66-71,Feb(2011).

[3] A new Symmetric key Cryptography Algorithm using extended MSA method: DJSA symmetric key algorithm, Dripto Chatterjee, Joyshree Nath, Suvadeep Dasgupta and Asoke Nath : Proceedings of IEEE International Conference on Communication Systems and Network Technologies, held at SMVDU(Jammu) 03-06 June,2011, Page-89-94(2011).

[4] New Symmetric key Cryptographic algorithm using combined bit manipulation and MSA encryption algorithm: NJJSAA symmetric key algorithm: Neeraj Khanna, Joel James,Joyshree Nath, Sayantan Chakraborty, Amlan Chakrabarti and Asoke Nath : Proceedings of IEEE CSNT-2011 held at SMVDU(Jammu) 03-06 June 2011, Page 125-130(2011).

[5] Symmetric key Cryptography using modified DJSSA symmetric key algorithm, Dripto Chatterjee, Joyshree Nath, Sankar Das, Shalabh Agarwal and Asoke Nath, Proceedings of International conference Worldcomp 2011 held at Las Vegas 18-21 July 2011, Page-306-311, Vol-1(2011).

[6] An Integrated symmetric key cryptography algorithm using generalized vernam cipher method and DJSA method: DJMNA symmetric key algorithm: Debanjan Das, Joyshree Nath, Megholova Mukherjee, Neha Chaudhury and Asoke Nath: Proceedings of IEEE International conference: World Congress WICT-2011 held at Mumbai University 11-14 Dec, 2011, Page No.1203-1208(2011).

[7] Symmetric key cryptosystem using combined cryptographic algorithms- generalized modified vernam cipher method, MSA method and NJJSAA method: TTJSA algorithm – Trisha Chatterjee, Tamodeep Das, Joyshree Nath, Shayan Dey and Asoke Nath, Proceedings of IEEE International conference: World Congress WICT-2011 t held at Mumbai University 11-14 Dec, 2011, Page No. 1179-1184(2011).

[8] Symmetric key Cryptography using two-way updated Generalized Vernam Cipher method: TTSJA algorithm, International Journal of Computer Applications (IJCA, USA), Vol 42, No.1, March, Pg: 34 -39(2012).

[9] Ultra Encryption Standard(UES) Version-I: Symmetric Key Cryptosystem using generalized modified Vernam Cipher method, Permutation method and Columnar Transposition method, Satyaki Roy, Navajit Maitra, Joyshree Nath, Shalabh Agarwal and Asoke Nath, Proceedings of IEEE sponsored National Conference on Recent Advances in Communication, Control and Computing Technology -RACCCT 2012, 29-30 March held at Surat, Page 81-88(2012).

[10] An Integrated Symmetric Key Cryptographic Method – Amalgamation of TTJSA Algorithm, Advanced Caeser Cipher Algorithm, Bit Rotation and reversal Method: SJA Algorithm., International Journal of Modern Education and Computer Science, Somdip Dey, Joyshree Nath, Asoke Nath,(IJMECS), ISSN: 2075-0161 (Print), ISSN: 2075-017X (Online), Vol-4, No-5, Page 1-9,2012.

[11] An Advanced Combined Symmetric Key Cryptographic Method using Bit manipulation, Bit Reversal, Modified Caeser Cipher(SD-REE), DJSA method, TTJSA method: SJA-I Algorithm, Somdip dey, Joyshree Nath, Asoke Nath, International Journal of Computer Applications(IJCA 0975-8887, USA), Vol. 46, No.20, Page- 46-53,May, 2012.
[12] Ultra Encryption Standard(UES) Version-IV: New Symmetric Key Cryptosystem with bit-level columnar Transposition and Reshuffling of Bits, Satyaki Roy, Navajit Maitra, Joyshree Nath, Shalabh Agarwal and Asoke Nath, International Journal of Computer Applications(IJCA)(0975-8887) USA Volume 51-No.1.,Aug, Page. 28-35(2012)
[13] Bit Level Encryption Standard(BLES) : Version-I, Neeraj Khanna, Dripto Chatterjee, Joyshree Nath and Asoke Nath, International Journal of Computer Applications(IJCA)(0975-8887) USA Volume 52-No.2.,Aug, Page.41-46(2012).
[14] Bit Level Generalized Modified Vernam Cipher Method with Feedback, Prabal Banerjee, Asoke Nath, Proceedings of International Conference on Emerging Trends and Technologies held at Indore, Dec 15-16,2012.
[15] Cryptography and Network Security, William Stallings, Prentice Hall of India