International Journal of Advanced Technology & Engineering Research (IJATER)
www.ijater.com

THE SYMMETRIC ENCRYPTION TECHNIQUE BY USING SUBSTITUTION MAPPING, TRANSLATION AND TRANSPOSING OPERATIONS

Shadma Fazal, Email: khanzshadma@gmail.com
Prof B.P.S Sengar, Email: bpssengar@gmail.com

Abstract

Now a days with the rapid growth in the electronic media and internet, demand of secure communication are also increases. Many of the secret data is transmitted through this unsecure transmission channel. To provide security over transmitted data many cryptographic algorithms has been proposed. An encryption/decryption algorithm like AES and DES are widely used for this purpose. In this paper, authors have proposed a new symmetric encryption/decryption algorithm based on substitution mapping, translation and transposing operation. Implemented results shows that proposed work is not only simple but also fast compare to AES and DES algorithms. Also security level of proposed work is higher due to the inherent poly-alphabetic nature of the substitution mapping method.

Keywords— Cipher text; Decryption; Encryption; Plaintext; Secret key

INTRODUCTION

In open networked systems, information is being received and misused by adversaries by means of facilitating attacks at various levels in the communication. Data encryption is sought to be the most effective means to counteract the attacks. There are two classes of encryption in use, which are referred to as i) Symmetric-key encryption using secret keys and ii) Asymmetric-key encryption using public and private keys[1]. Public-key algorithms are slow but Symmetric-key algorithms generally run 1000 times faster. Symmetric-key cryptography has been -- and still is extensively used to solve the traditional problem of communication over an insecure channel [2]. The encryption standards such as DES (Data Encryption Standard), AES (Advanced Encryption Standard), and EES (Escrowed Encryption Standard) are used in Government and public domains [6,8]. With today's advanced technologies these standards seem not to be as secure and fast as one would like[5]. Time Dependant Multiple Random Cipher Code is a non- Feistel Symmetric-key encryption algorithm using random numbers[3]. Performance comparison of popular symmetric-key encryption algorithms found in literature indicates that Blowfish is faster compared to DES and AES. High level encryption and decryption are becoming increasingly important in the area of high-speed networking[4]. Fast encryption algorithms are needed these days for the secure communication of high volume information through insecure channels.

In this paper, a new symmetric-key encryption algorithm for secured message communication over insecure channels is presented[7]. It is a direct mapping poly alphabetic Symmetric-key encryption algorithm. Here, we use direct substitution mapping and subsequent translation and transposition operations using X-OR logic and circular shifts that results in higher conversion speed[9]. The block size is 128 bits (16 characters) and the key size is also 128 bits (16 characters)[10]. A comparison of the proposed encryption method with DES and AES is shown in table. 2. The rest of the paper is organized in the following sections. In section 2, the encryption process is explained and in section 3, decryption process is explained. Performance is evaluated in section 4 and conclusions are made in section 5.

THE ENCRYPTION PROCESS

A. *Nomenclature:*

P – Plaintext
C – Cipher text
K – Secret key
C_{L1} – Level-one cipher text
$P(i)$ – ith plaintext character in input plaintext character block
$C(i)$ – ith cipher text character in a block
$C_{L1}(i)$ – ith level-one cipher text character in a block $M[i][j]$ – Element of matrix M with row i and column j
$A(i)$ – Element of Array A with index i $K(i)$ – ith character of secret key, K
K_{ts_n} -- Sub-key for translation in nth round
K_{tp_n0}, K_{tp_n1}, K_{tp_n2}, K_{tp_n3} -- Sub-keys for transposition in the nth round

B. *ENCRYPTION STEPS*

The encryption, $C = E(K,P)$, using the proposed encryption algorithm consists of three steps. The first step involves initialization of a matrix with ASCII code of characters, shuffled using a secret key, K. This initialization is required only once before the beginning of conversion of a plaintext into corresponding cipher text. The second step involves mapping

by substitution using the matrix, each character in every block of 16 characters into level-one cipher text character. The third step involves translation and transposition of level -one cipher text characters within a block, by X-OR and circular shift operations, using arrays, in 8 rounds. Fig.1 shows simplified block diagram of the encryption scheme.

C. *MATRIX FOR SUBSTITUTION MAPPING.*

A matrix *M* with 16 rows and 95 columns initialized with ASCII codes of characters using secret key is used for mapping the plaintext characters into level-one cipher text characters. During encryption, a block of 16 plaintext characters in the message is taken into a buffer. The ASCII code of the character *P(i)* is obtained. From this ASCII code, 32 is subtracted. The resulting integer is used as column number j of i^{th} row of the matrix *M*. The element contained in this cell which is an ASCII code of a character, is taken as the level-one cipher text character $C_{L1}(i)$ corresponding to the plaintext character *P(i)*. In this way all the characters in a block are mapped into level-one cipher text characters and all plaintext character blocks are mapped into level-one cipher text character blocks.

D. *MATRIX INITIALIZATION.*

A matrix *M* with sixteen rows and ninety five columns is defined. Columns in every row of the
matrix is filled with ASCII codes of characters starting from BLANK (ASCII = 32) in column zero to '~' (ASCII = 126) in column ninety-four representing elements of the matrix.

A 16 character (128 bits) secret key *K*, with key characters *K(0)* through *K(15)*, is used for encryption and decryption. The i^{th} row of the matrix is given an initial right circular shift, as many number of times as equal to the ASCII code of $(i+1)^{th}$ key character to shuffle the contents of the matrix *M*, for i = 0 to 14. For example, if *K(1)*, is 'a' whose ASCII code is 97, row 0 of the matrix *M* is right circular shifted 97 times. If *K(2)* is 'h' whose ASCII code is 104, the second row of the matrix *M* is right circular shifted 104 times and so on. The row 15 of matrix *M* is right circular shifted as many number of times as equal to ASCII value of the key character *K(0)*.

Further, the i^{th} row of the matrix is given a second right circular shift as many number of times as equal to ASCII (*K(i)*) to shuffle the contents of the matrix *M*, for i = 0 to 15. For example, the row 0 of M is right circular shifted as many number of times as equal to the ASCII value of key character *K(0)*. The row 1 of the matrix *M* is given a right circular shift as many number of times as equal to the ASCII value of the key character *K(1)* and so on. Code.3 shows this second circular shift operation applied to the rows of matrix *M*.

E. *SUBSTITUTION MAPPING PROCEDURE.*

A given message is broken into blocks of sixteen plaintext characters *P(0)* through *P(15)*. Plaintext character *P(i)* is taken and a number j is calculated such that j = (ASCII code of plaintext character *P(i)* – 32). This number, j, is used as column number of the matrix *M*. Using j as column number we proceed to find the element in the i^{th} row of the matrix *M*. This element (ASCII code of a character) is used as level-one cipher text character $C_{L1}(i)$ for a given plaintext character *P(i)*. For example, for the plaintext character *P(0)* in a block, i = 0, j = (ASCII code of plaintext character *P(0)* – 32) is used as column number of row 0 of the matrix *M* to obtain level-one cipher text character corresponding to *P(0)*. Similarly for character *P(1)* in the plaintext character block, i = 1 and j = (ASCII code of plaintext character *P(1)* – 32) where j is used as column number of the row 1 of the matrix to obtain level-one cipher text character corresponding to *P(1)*. In this way, all the 16 plaintext characters in a block are mapped into 16 level-one cipher text characters denoted by $C_{L1}(i)$, i = 0 to 15. The characters of level 1 cipher text character block ($C_{L1}(0)$ through $C_{L1}(15)$) are transferred to a 16 element array A_1.

F. *SUB-KEY SET GENERATION.*

One set of eight sub-keys K_{ts_0}, K_{ts_1}, K_{ts_2}, …. K_{ts_7} are generated using the secret key *K* such that: K_{ts_n} = characters in columns 0 through column 15 in row *n* of matrix *M* concatenated. These keys are used in translation rounds. Another set of sub-keys K_{tp_n0}, K_{ps_n1}, K_{tp_n2} and K_{tp_n3} are generated such that K_{tp_n0} = character of matrix *M* with row number n and column number 0. Here, each key, is a character represented by the corresponding element in the matrix *M*. These keys are used in transposition rounds.

G. *TRANSLATION AND TRANSPOSING.*

Eight rounds of translation and transposition operations are performed on the level 1 cipher text character block. The translation operations are done using X-OR operation performed on the cipher text character block using sub key, K_{ts_n} in the n^{th} round. The translated cipher text character block is transposed using four arrays whose elements are circular shifted using sub-keys K_{tp_n0}, K_{tp_n1}, K_{tp_n2}, K_{tp_n3} used in that round. These operations make the resulting output cipher text characters extremely difficult to decrypt by any adversary without having the secret key. The translation and transposition produce the effect of diffusion.

1) TRANSLATION OF CIPHER TEXT CHARACTERS.

The contents of array A_1 is X-ORed with sub key K_{ts_n} in the n^{th}

round. The 16 characters of each block of cipher text are X-ORed with 16 characters of sub key K_{s_n}.

2) TRANSPOSING TO CIPHER TEXT CHARACTERS.

The X-ORed level -one cipher text characters available in array A_1 are bifurcated and transposed using four arrays. For the n^{th} round, array A_1 is right circular shifted as many number of times as equal to the integer value of K_{tp_n0}. After this operation, the first eight elements of A_1 (left most elements) are transferred to another array A_2 having 8 element positions. Then, A_2 is right circular shifted as many number of times as equal to the integer value of K_{tp_n1}. The other eight elements of the array A_1 (rightmost elements) are transferred to another 8 element array A_3 which is left circular shifted as many number of times as equal to integer value of K_{tp_n2}. Then A_2 and A_3 are concatenated and transferred to the 16 element array A_1. This 16 element array, A_1, is right circular shifted as many number of times as equal to the integer value of K_{tp_n3} . After this operation, the contents of A_1 represent the cipher text characters in a given block. The elements of array A_1 are moved to the cipher text block *C(0)* through *C(15)*. The cipher text blocks are used to create the output cipher text message file.

THE DECRYPTION PROCESS

The decryption algorithm performs the reverse operations of encryption such that *P = D(K,C)*. It is done in three steps. Here, cipher text character *C(i)*, in blocks of 16 are processed using arrays and matrix. The first step involves initialization of a matrix with ASCII codes of characters, shuffled using the secret key. In the second step, the cipher text characters are de-transposed using circular shift operation of array and de-translated by X-OR logic using sub-keys in multiple rounds. With this operation we get back the level-one cipher text characters. In the third step, these level-one cipher text characters are inverse-mapped into plaintext characters using the matrix. In the decryption algorithm, sub-keys are generated from the secret key in the same way as in the case of encryption algorithm.

H. *MATRIX INITIALIZATION.*

An identical matrix *M*, used for mapping the plaintext characters into level-one cipher text characters, is used here for inverse mapping of the level-one cipher text characters into plaintext characters during decryption. At the decryption site, this matrix is created using the secret key *K* in the same way as in the case of encryption.

I. *DE-TRANSPOSING OF CIPHER TEXT CHARACTERS.*

The cipher text character block from the cipher text file is brought in to a 16 element array A_1. For the n^{th} round, array A_1 is left circular shifted as many number of times as equal to the integer value of K_{tp_n3}. After this operation, the first eight elements of A_1 (left most elements) are transferred to another array A_2 having 8 element positions. Then, A_2 is left circular shifted as many number of times as equal to the integer value of K_{tp_n2}. The other eight elements of the array A_1 (rightmost elements) are transferred to another 8 element array A_3 which is right circular shifted as many number of times as equal to integer value of K_{tp_n1}. Then A_2 and A_3 are concatenated and transferred to the 16 element array A_1. This array is left circular shifted as many number of times as equal to the integer value of

K_{tp_n0}

J. *DE-TRANSLATION OF CIPHER TEXT CHARACTERS.*

The contents of array A_1 is X-ORed with the bits of sub key K_{ts_n} in the nth round. After this operation, the contents of the array *A* corresponds to the level-one cipher text character block corresponding to the one obtained after the mapping operation done at the encryption side using the matrix. The contents of array A_1 is moved to level 1 cipher text block, C_{L1}.

K. *INVERSE MAPPING USING MATRIX.*

If $C_{L1}(i)$ is the level-one cipher text character in a block, the inverse mapping is such that *P(i)* = char((column number j of i^{th} row of matrix *M* where $C_{L1}(i)$ is the element) + 32). For example, let the 1^{st} level-one cipher text character, $C_{L1}(1)$, in a block be '#'. We proceed to search '#' in the matrix *M* to find the column number j in the 1^{st} row where $C_{L1}(1) = M[1][j]$. Then we determine the character whose ASCII = (j + 32) which gives the plaintext character *P(1)* corresponding to $C_{L1}(1)$. Let the 2^{nd} level-one cipher text character, $C_{L1}(2)$, in a block be '%'. We proceed to search '%' in the matrix *M* to find the column number j in the 2^{nd} row where $C_{L1}(2) = M[2][j]$. Then we determine the character whose ASCII = (j + 32) which gives the plaintext character *P(2)* corresponding to $C_{L1}(2)$. In this way we can inverse map every cipher text character in every block into plaintext characters to get back the original message file.

PERFORMANCE EVALUATION

Performance comparison of various popular secret key algorithms, such as DES, AES and Blowfish running on a Pentium-4, 2.4 GHz machine, discussed in the literature [9] shows that Blowfish is the fastest among these algorithms. The throughputs of these algorithms are respectively 7,988 bytes/sec, 5,326 bytes/sec and 10,167 bytes/sec.

The proposed Symmetric-key Encryption algorithm is subjected to performance evaluation using a Pentium-4, 2.4 GHz machine. Execution time taken by the algorithm was measured using a plaintext file and the throughput calculated. The time between two test points in the algorithm during execution was measured with the help of system clock. The number of bytes (in the plaintext file) required for an execution time of one second during encryption was ascertained. Table.1 shows the comparison of performance of this encryption algorithm with the performance of popular secret key algorithms given in the throughput of Blowfish algorithm is only 10,167 bytes per second whereas this encryption algorithm provides 81,674 bytes per second. Thus this Encryption algorithm is 8 times faster than Blowfish algorithm.

Table 1: Comparison of throughput of proposed encryption algorithm with popular encryption algorithms on a Pentium-4, 2.4 GHz machine

Encryption Algorithm	DES	AES	Blow fish	Proposed Encryption
Throughput Bytes/sec	7,988	5,326	10,167	81,674

Plate.1 shows a plaintext message file used for encryption. Plate.2 shows the cipher text message file generated using the proposed encryption scheme. Plate.3 shows the plaintext message recovered from the cipher text message using the proposed decryption scheme.

AAAAAAAAAAAAAAA
aaaaaaaaaaaaaa
BBBBBBBBBBBBBBB
bbbbbbbbbbbbbbb
111111111111111
222222222222222
$$$$$$$$$$$$$$$

Plate.1:Plaintext message used for encryption

\rm]a i5sNz{z < <R.-aC(2 Z4ZU}gCsn^a h4rM{t{ =
#/aB+ 5[5 T~h b}Ma0yEC~
Kj VSc~Na3xDB}Dk W
mHx*-JRTk Ve7I

Plate.2: Cipher text generated from the message

AAAAAAAAAAAAAAA
aaaaaaaaaaaaaa
BBBBBBBBBBBBBBB
bbbbbbbbbbbbbbb
111111111111111
222222222222222
$$$$$$$$$$$$$$$

Plate.3: Recovered message after decryption

PERFORMANCE EVALUATION

Performance comparison of various popular secret key algorithms, such as DES, AES and Blowfish running on a Pentium-4, 2.4 GHz machine, discussed in the literature shows that Blowfish is the fastest among these algorithms. The throughputs of these algorithms are respectively 7,988 bytes/sec, 5,326 bytes/sec and 10,167 bytes/sec.

The proposed Symmetric-key Encryption algorithm is subjected to performance evaluation using a Pentium-4, 2.4 GHz machine. Execution time taken by the algorithm was measured using a plaintext file and the throughput calculated. The time between two test points in the algorithm during execution was measured with the help of system clock. The number of bytes (in the plaintext file) required for an execution time of one second during encryption was ascertained. Table.1 shows the comparison of performance of this encryption algorithm with the performance of popular secret key algorithms given in. The throughput of Blowfish algorithm is only 10,167 bytes per second whereas this encryption algorithm provides 81,674 bytes per second. Thus this Encryption algorithm is 8 times faster than Blowfish algorithm.

CONCLUSION

The Encryption algorithm is a simple, direct mapping algorithm avoiding the Fiestel structure and multiple round operations used in popular symmetric ciphers. Consequently, it is very fast and suitable for real-time applications. Two of the advantages are as First, the encryption and decryption procedures are much simpler, and much faster. Second, the security is quite higher as the inherent poly-alphabetic nature of the substitution mapping method which is used here together with the translation and transposition operations.
The matrix based substitution results in poly alphabetic cipher text generation which is followed by multiple round arrays based transposing gives strength to this encryption algorithm.

The combination of the poly-alphabetic substitution and translation and also the transposition makes the decryption of the algorithm extremely difficult without having the secret key. The cipher text generated by this algorithm does not have one to one correspondence in terms of position of the characters in plaintext and cipher text. This also makes the decryption extremely difficult even by brute force attack.

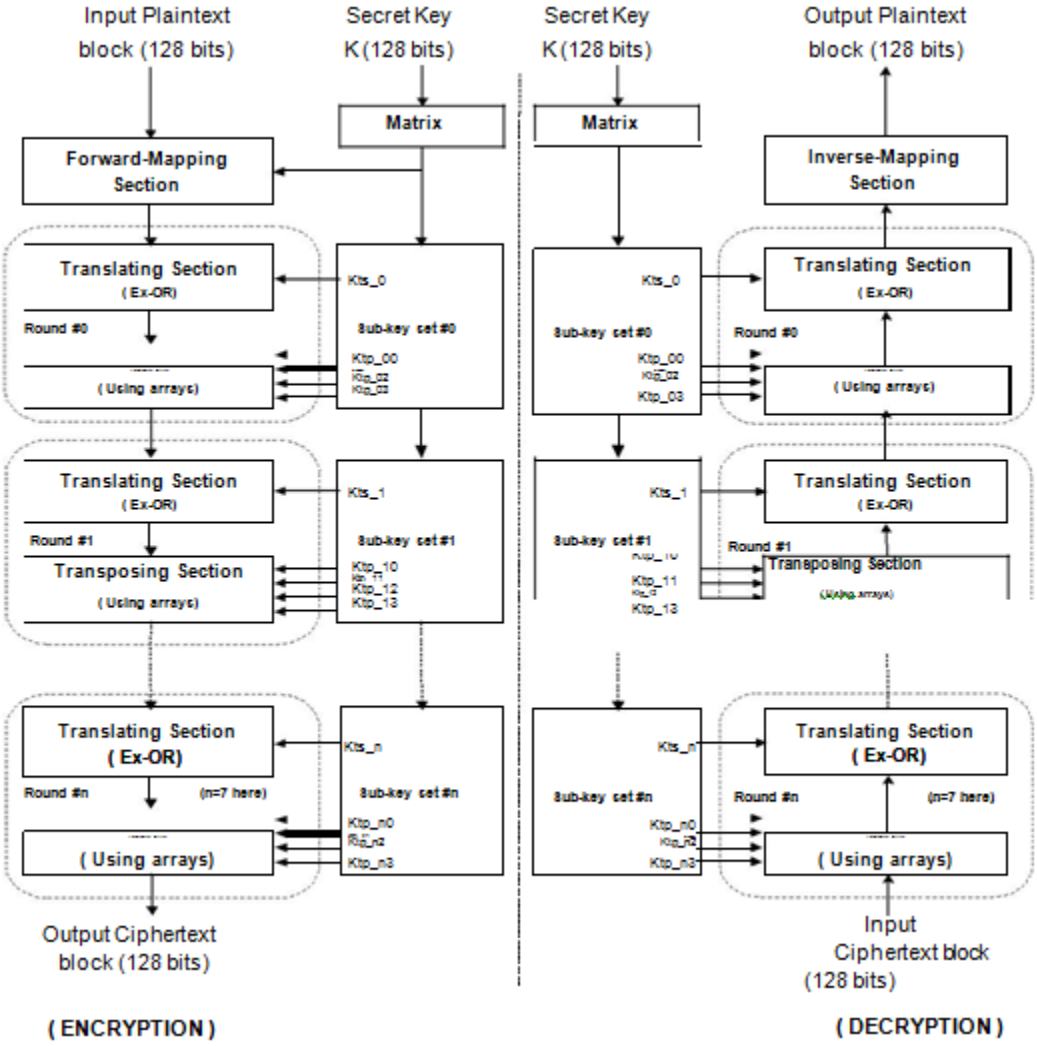

Fig. 1: Simplified Block diagram of the Encryption Scheme

Table.2: Comparison of proposed encryption algorithm with DES and AES

Parameter	AES	DES	PROPOSED
Block Size	128 bits	64 bits	128 bits
Key Length	128, 192, 256 bits	56 bits	128 bits
Encryption primitives	Substitution, Shift, Bit mixing	Substitution Permutation	Substitution, Translation, Transposing
Cryptographic primitives	Confusion Diffusion	Confusion Diffusion	Confusion Diffusion
Performance	Slow	Fast	Faster

REFERENCES

[1] William Stallings, "Network Security Essentials (Applications and Standards)" Pearson Education, 2004, pp. 2–80.
[2] Charles P. Pfleeger, Shari Lawrence Pfleeger. "Security in computing" Pearson Education 2004 – pp. 642-666
[3] Jose J. Amador, Robert W. Green, "Symmetric-Key Block Ciphers for Image and Text Cryptography", International Journal of Imaging System Technology, Vol. 15 – pp. 178-188, 2005.
[4] Dragos Trinca, "Sequential and Parallel Cascaded Convolution Encryption with Local Propagation: Toward Future Directions in Cryptography", Proceedings of The third International Conference on information Technology-New Generations. (ITNG'06), 0-7695-2497- 4 / 2006, IEEE Computer Society.
[5] Data Encryption Standard :http://csrc.nist. gov/publications/fips/fips46-3/fips-46-3.pdf
[6] Advanced Encryption Standard http://csrc.nist.gov /publications/fips/fips197/fips-197.pdf
[7] Dr. Varghese Paul, "Data Security in Fault Tolerant Hard Real-time Systems: Use of Time Dependant Multiple Random Cipher Code". Ph.D dissertation, Cochin University of Science and Technology, April, 2003.
[8] Escrowed Encryption Standard http://csrc.nist.gov /publications/fips/fips1185/fip185.txt Aameer Nadeem, Dr. M. Younus Javed, "A Performance Comparison of Data Encryption Algorithms", 0-7803-9421-6 /2005 IEEE.
[9] Designing The New Security Protocol Computer Science Essay: http://www.ukessays.com/essays/computer-science/designing-the-new-security-protocol-computer-science-essay.php#ixzz31ByT7E7b