Secret Hiding Over Net Algorithm (S.H.O.N.A) for Security Enhancement In Data Communication

Husaini Akbar Ali[1] and Dr. Sunita Bansal[2]

[1]*B,.B.D. University, Lucknow, India*

[2]*BBDNITM, Lucknow, India*

Abstract-These In this work, we present an encryption algorithm that can be used to for textual exchange applications to secure data communications. This encryption algorithm is based on hiding a number of bits from simply encrypted plain text message into a random vector of bits. The locations of the hidden bits are determined by a key known to the sender and receiver. We call this Secure Hiding Over Net Algorithm (SHONA). The name demonstrates the two basic operations of this algorithm. These are operations include inserting part of the cipher text bits into a cover of a matrix to hide it from recognition. The process is quit similar to Steganography." The distinctive features of this algorithm are as follows:

• Key length is variable: the key length can be varied from 8 character up to any larger value depending on the security level required.

• Word length is variable: According to the security level required, we can go for a single character encryption at a time or set of character in one step of encryption.

• The algorithm, therefore, provides variable degrees of security. However, this increased security level will be at the cost of increased size of the cipher-text.

• The cipher algorithm is simple and way of arrangement in matrix is liner format, but easy is for only that person who knows the key.

• Application of this process is really easy for any type of security system.

Keywords: Encryption, data security, steganography, data hiding, data communication, SHONA.

I. INTRODUCTION

In this work, we present an encryption algorithm that can be designed for to secure data communications. The name demonstrates the two basic operations of this algorithm. These operations are based on inserting part of the text bits into a cover too hide it from recognition. That is there are simple conventional operations on the plain text, then hiding cipher text in a random bit string. We call this algorithm Secret Hiding Over Net Algorithm (SHONA) as it hides the cipher text in a matrix of (9X9) which already has a random bit stored in it.

In the following sections, we go through existing algorithms with some comparative study. We then going to describe our algorithm. The first algorithm that we discuss is RC4 [4]. This algorithm is a variable-key-size stream cipher. The general problem with RC4 is that almost every statement depends immediately on the previous statement, including the table index computation and the associated table accesses, thus limiting the amount of parallelism. On the other hand, SEAL [2][4] is a software-efficient stream cipher.

The major problem with this algorithm is the strong dependency between consecutive operations, allowing only minimal parallelism. In block-encryption algorithms, DES [1][5] recognized world-wide, it set a precedent as the first commercial-grade modern algorithm with openly and fully specified implementation details. The problem with DES is that the size of the key space is too small to be really secure. Khufu algorithm [2, 4] can be simply implemented in hardware at one clock per round. IDEA [2, 4] is used for message encryption in Pretty Good Privacy (PGP). In fact, the speed of hardware depends dramatically on how much cost could be absorbed due to the need for dedicated multipliers which are not cheep. Skipjack [2] is NAS-developed algorithm for the Clipper and Capstone chips and it is considered secure. REDCO II [2, 3] is secure because using the brute force attack, 2160 operations are required to recover the key. A summary of block encryption algorithms is shown in Table 1.

Table 1

Block-Encryption Algorithms			
	Key Length	**Block Length**	**Shortcoming**
DES	56 Bits	64 bits	Small size key
REDCO II	160 bits	80 bits	Sufficiently Security
IDEA	128 bits	64 bits	Patented
Skipjack	80 bits	64 bits	Secret

II. DESIGN METHODOLOGY OF THE PROPOSED ALGORITHM

On designing this algorithm, we have considered that the crypto analyst knows all details of the algorithm. This conforms to "Kerckhoffs' Principle" in cryptography, which holds that "the security of a cryptographic system should rely only on the key material". The basic idea of our proposed encryption algorithm is hiding a number of bits from cipher text message into a 9X9 vector of bits. The location of the hiding bits are determined by a pre agreed-upon key by the sender and the receiver. The following subsection gives more details about our algorithm.

Encryption Algorithm

Encryption process of SHONA has 3 basic steps:

1) ASCII Conversion.

2) Cipher Logic Shift and Binary conversion.

3) Matrix Replacement

In first step we get the message characters ASCII code. Then we apply Shift_Logic on the key K to get a shift value. Then we apply circular shift in the range of 32 to 126. In this range, all the characters commonly used in textual format are present. This new value is then converted to binary value. Then we make a matrix of 9x9 and find out the first number KF and last number KL from the key and then put the first character on the matrix on location (KF,KL) and then if KF<KL put the next coming ciphered message characters linearly forward way. If KF>=KL then put the upcoming ciphered message characters linearly backward way in matrix. An explained procedure of the process is written below:

1) ASCII Conversion:

In this step we get the character from the message and then convert it to ASCII code. As this is a numeric code so encryption logic can be easily applied. ASCII code is accepted world wide and it is a basic methodology so this is the first step as well as the message conversion is also easy.

2) Cipher Logic Shift and Binary Conversion:

In this phase, we do a circular shift on the ASCII code of the message. Shift value (SV) is decided by a logic which works in following way.

- For first character of message, we take alternate numbers from the key and add all of them then we subtract both the values to get the shift value (SV).

SV=(K1+K3+K5+K7+…… till last) – (K2+K4+K6+…..till last)

Ex: let us say if the key is 32456143 then the shift value SV will be

SV= (3+4+6+4)-(2+5+1+3)

- For second character of message, we do the same process as above, just that we now take a set of 2 digits to do the same. So hear Shift value will be

SV= (K1K2+K5K6+K7K8+…till last)- (K3K4+K7K8+…till last)

Ex: let us say if the key is 32456143 then the shift value SV will be

SV= (32+61)-(45+43)

- For third character we will repeat the same procedure similarly and this time we take the key in a block of 3 values.
- This process is repeated till half of the key size. After that if message is still not completed, the we repeat the process from the top with single value and follow the whole procedure.
- Where high security is required, and the key value is larger (say 128 characters or more) then we may go till 32 value block and then repeat from the top.
- The SV can be both positive and negative, so the shift will be on both sides.
- After the shift is done, the value is stored in a temporary array.

3) Matrix Replacement:

In this final process, we make a matrix of 9X9 with random values.

- We then take the first and last number from the key KF and KL. And taking (KF, KL) as a location put the first character in that location.
- Now if KF<KL, the next coming elements of array are arranged in linearly forward way.
- If KF>KL, the next coming elements of array are arranged in linearly backward way.

➢ If all elements are not able to record in a single matrix, we make another matrix and for that, we just reverse the order of arrangement than previous matrix.

Decryption Algorithm

Decryption process is almost similar to the encryption process. BT just in reverse order with some changes in the procedure.

It also has three steps:

1) Matrix Replacement:

Hear, we take the received encrypted matrix, and we already know the key, so take the first and last value of the key to fine the location where actual message is stored. From that point, if KF<KL we move linearly forward and store the values in a temporary array, else move linearly backward and do the same.

2) ASCII Conversion.

Now the values are converted in ASCII code and these codes are made ready to be input in the decryption logic shift process.

3) Cipher Logic Shift and Binary conversion.

We repute the same logic we have used in the encryption process , the only change we do hear is that we take the Shift Value SV with a – Sign i.e. now we use the negative of SV so that the changes will be roll baked and the actual value will be received as the output.

III. KEY VALUE

The key value can be anything and the size of key depends on the level of security required. Hear user can also take alphanumeric keys. It that case, the values of alphabet will be converted to ASCII code when encryption process will be executed.

Ex: if the key is 49AD478P then the encryption logic will take the key as 49656847880 as 65,68 and 80 are the ASCII code for A,D and P.

IV. ALGORITHMS ANALYSIS

The worst case, regarding storage requirements, occurs when replacing one bit only from message to the 9X9 vector. Hence, cipher text can be equal to mas_size+81 size of the plain text.

But commonly the messages are not of a single character, so any message of 10-15 character will be an efficient cipher text, and not just this increasing message size will also maintain the efficiency, as the encrypted output will be msg_size+81 in worst case, so let us say if a message has 100 character, then it's first 80 characters will be in one matrix set, and the rest 20 will be in the next matrix set. So overall the encrypted message will be having 162 characters in which 100 will be the cipher text entities and the rest 62 are random characters and similarly for a message of 1000 character, 1000 cipher entities will be in 13 matrix of 9x9 and the rest 53 will be random characters. So we can easily see that the SHONA works in a much more efficient way

Moreover, we have studied the Following:

- Key length is variable: the key length can be varied from 8 up to any larger value depending on the security level required.
- Word length is variable: we can put single character on cipher algo or can be made in blocks but the best result
- The algorithm, therefore, provides variable degrees of security. Yet it dose not result in a much increased size of encrypted output file so making the process more efficient on resource requirement.
- The number of rounds is variable: the whole process can be repeated r times using the same key.

V. SUMMARY & CONCLUSIONS

For higher degrees of data security, we perform the following:

1. The process can be repeated on the resulting ciphered file. That is, using more than one round of encryption cycles. The number of rounds should be agreed upon by sender and receiver before transmission begins.

2. Encryption can be performed on character blocks too depending on the requirement, block size can be varied whatever the user want to, but the best case will be use of single character at a time.

3. key value can be made alphanumeric as well as symbols can also be used in the key like AP46&#<g6>X&7 will also be a valid key for this process.

ACKNOWLEDGEMENT

For this paper , the ides is to make the data more secure and the process must be easy so that no one else can know what I want to say, what I fell, what I express , there are always some things which meant to be just for you or some one special whom you want to tell. it's between me and

the person whom I have allowed access to my world. SHONA will always do this in an efficient ways. I would like to thanks Dr. Manoj Darbari . Our Head Of Department for his kind and valuable support and suggestions throughout the work.

REFERENCES

[1] Menezes, A.. Handbook of Applied Cryptography. Boca Raton in CRC Press 1997.

[2] Schneier, B. Applied Cryptography. New York, at Wiley, 1996.

[3] T.W. Cusick and M.C. Wood, “The REDOC-II Cryptosystem,” Advances in Cryptology—CRYPTO ‘90 Proceedings, Springer-Verlag, 1991, Page No 545-564.

[4] Bruce Schneier and Doug Whiting. "Fast software encryption: Designing Encryption Algorithms for Optimal Software Speed on the Intel Pentium processor," In Biham . Page No 240-260, 1997.

[5] Stinson, D. Cryptography: Theory and Practice. Boca Raton in CRC Press, 1995

[6] New Approach of Data Encryption Standard Algorithm ,Shah Kruti R., Bhavika Gambhava

[7] ENCRYPTION TECHNIQUES: A TIMELINE APPROACH , T Morkel , JHP Eloff Information and Computer Security Architecture Research Group, The Department of Computer Science University of Pretoria at 0002, Pretoria in South Africa