IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

Twin Encryption System: Approach to Self-Changing Encryption and Decryption Key

Suhail Sayeed Bukhari[1], Meru Prabhat[2]

Department of Software Engineering, SRM University, Chennai, Tamil Nadu, India [1,2]

Abstract: Currently all the encryption and decryption methods, in communication depend upon the use of a key, which by itself remains constant and needs a manual interaction to bring forward a change. In order to avoid this problem, we introduced a method which will almost eliminate the need for human element. This concept has been established by developing a key generation system that does not need the key to be constant or pre declared. A type of key that is not only self-changeable but also unique for every message that is being sent. This key will remain applicable to a particular message only for a set period of time. This dual concept of time and uniqueness for every message makes up the Twin Encryption System (TES). TES develops a key based on unique identities of two communicating machines and the current time, which creates a time function that can encrypt and decrypt the file or message without the need for fixing the key and then sharing it.

Keywords: TES, Cryptography, encryption, time-stamp, self-changeable key, Symmetric key, Asymmetric key, security, secure communication.

I. INTRODUCTION

In today's world security and privacy in communication is of a major concern. A lot of ways have been developed to create such an environment where messages can be sent, encrypted. This process, of encryption and decryption of data is called cryptography [1]. Cryptography effectively includes two things: encryption and decryption and a key is required for both the processes. This key can be classified into symmetric key and asymmetric key. In symmetric-key cryptography two keys are used, one for encryption and other for decryption. They can be implemented either as block ciphers [2] or stream ciphers [3]. And in asymmetric key cryptography only one key is used for, both, encryption as well as decryption and a system of public key [4] and private key [5] is used to ensure further security. The terms Bob, Alice and Eve are much used in understanding cryptography, where Alice stands for the encryption side and Bob stands for decryption side and Eve represents the third party which can steal the codes and messages, which is depicted in Fig.1.

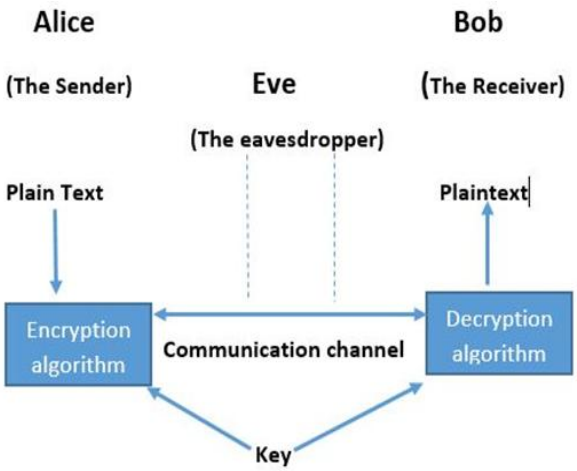

Fig.1 Alice and Bob, communication channel

The threat of Eve is possible in both the types of cryptography as they all have a prefixed key which can be stolen. In order to resolve this flaw new methodologies, need to be developed which can protect the key from being leaked.

Our contribution: - We have tried to contribute as follows: - First, we introduced a method of developing the key which includes MAC addresses and other types of identifications in combination with the time- stamp which will include the current year, month, date, time up to the current minute.

Secondly, we have proposed a way of using the key which will use the same key for encryption and decryption, but, the key will create three different parameters for encryption as well as decryption.

Third, we have used the length of the text in the process of encryption and decryption in order to further increase the security and uniqueness of the key and lastly we have introduced a method for keeping the messages private by making the key changeable with respect to the time. The key will change itself every minute and for every different length of text. Many algorithms have been developed for the purpose of encryption and decryption. Such as Data Encryption Standard (DES), the Advance Encryption Standard (AES) [6], RSA algorithm [7], and SSL/ TLS [8], but they all have a constant key or keys which need to be changed manually. All these algorithms mentioned use complex mathematical analysis and mathematical functions to get the desired key which takes time in processing while our methodology uses simple mathematical solutions and functions to achieve the result and the complexity of the key and the cipher is only created by the manipulation of alphabets and numbers.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

II. PROPOSED METHODOLOGY

In this section all the technicalities will be explained and various procedures and pseudo algorithms will be given so that, they could be later, easily written in any programming language, desired. This section will include the overview along with other major concepts of TES.

A. Overview

We have created a method of encryption and decryption of plain text which we named as Twin Encryption System (TES). Our proposed TES works in three stages: The first stage is called key keeper. The second phase is called the Run phase and the third phase is called the Vine phase. TES can currently be only used for communication between two thick clients [9] which have a MAC address and global time synchronization or synchronization with the Precision Time Protocol (PTP) [10] [11]. In the TES system the run phase and the key keeper will run on both the thick clients while as the vine phase will run only once which will be used for the communication between the two thick clients, and it will be also used for sending the message and sharing the MAC addressees with both the thick clients.

B. The Key Keeper

The basic purpose of the key keeper is to run the algorithms in order to create the key as well as the ciphers. The key keeper in our proposed methodology will combine the MAC address and the time stamp and the length of the text into a, x + 12-digit key, (where x is the length of the original text and the symbol "+" stands for concatenation), which will be then split into three parts which are named as cursors. All the three phases will have a cursor associated with it: the memory cursor, the reference cursor and the encryption cursor respectively.

By splitting the key in three cursors we get two four digit cursors and one x + 4-digit cursor, out of which two, 4 digit cursors will be responsible for setting the characters into the Reference array and the Encryption array and x + 4-digit cursor will be used to set characters into the Memory array. The cursors are illustrated in Fig.2

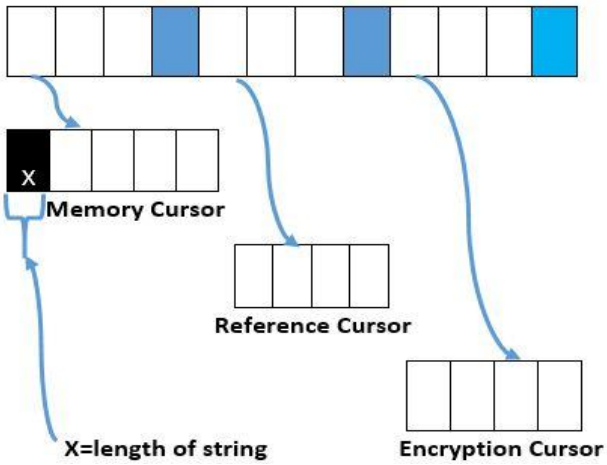

Fig.2 depiction of the key and the three cursors

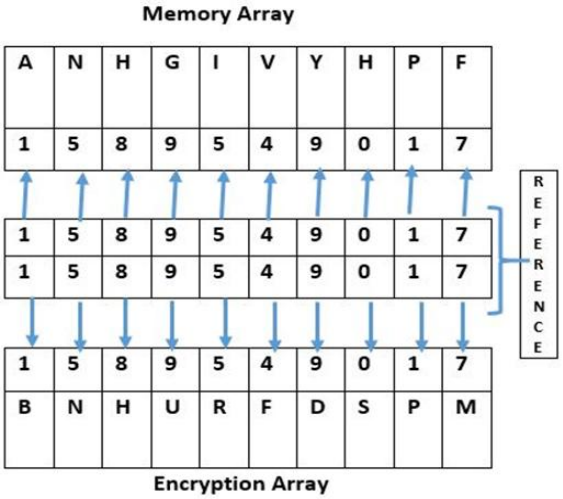

Fig.3 sample of three arrays and their association with each other

C. The Three Arrays

In our system we use three Arrays which are classified as the Memory array, the Reference array and the Encryption array. The TES will encrypt only one word at a time but due to its simplicity it will still be a quick process. The characters, which are to be encrypted, are treated as a queue and the encryption will take place as a first come first serve basis.

The Memory array will basically be set with Alphanumeric characters into different blocks which will be acquired with the help of the key, then it will match the characters with the character that is currently being processed and with that particular character there will be a number assigned to it which will be generated in the Reference array.

The Reference array works in two parts; one attached with the Memory array and the other will be associated with the Encryption array. In the Reference array using the key, a series of numbers will be created which will be put into the wheel and then split into two parts. One part will assign numbers to the blocks of the Memory array and the other part will assign numbers to the Encryption array, identically. The Encryption array will have alphanumeric characters filled in the blocks just like Memory array, but in this case the characters will be attached to the numbers, of the Reference array, while as in the Memory array the numbers of the Reference array were attached to the characters. The Encryption array will create a final cipher which will be sent to the receiver or Alice. The Memory array and the Encryption array will contain all the English alphabets both upper case and lower case and it will also include numbers from 0 to 9 and a space and period including other symbols which are present on a standard keyboard [12].

D. The Run Phase

This phase actually works on converting the plain text into a cipher text. The number of alphanumeric characters in the cipher text will be greater than the number of characters in the plain text. The run phase will use the cursors from the key keeper. The memory cursor will be

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

used to set the characters into the Memory array in a sequence which will be same when generated by the receiver thick client or the sender thick client.

The run phase will take the Reference cursor and the Encryption cursor to generate the setting for the Reference array and the Encryption array respectively. After all the three arrays have been set. The Reference array splits into two identical parts, one part gets attached with the Memory array and the Encryption array gets attached to the second part. Now the actual encryption takes place, which is shown in Fig.4.

Original Text: - THIS IS A BOOK

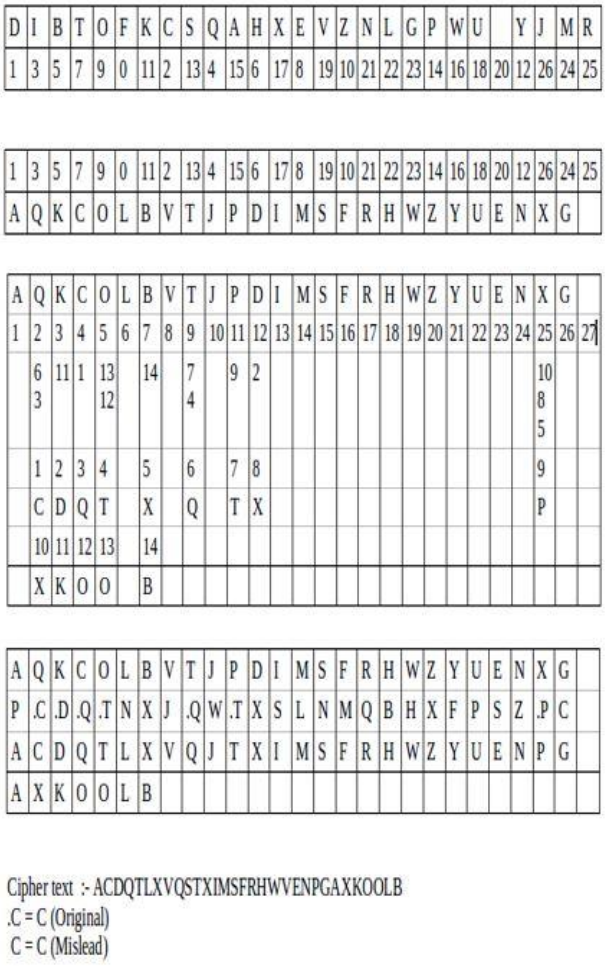

Fig.4 illustration of the encryption process

(I) First it will take a character from the actual text and match it in the Memory array.

(II) With that particular character there will be a number attached to it.

(III) This number will be matched with other part of the Reference array which is attached to the Encryption array.

(IV) In the Encryption array there will be a character attached to that unique number, which will be called for processing.

(V) Depending upon the number and the sequence, a character is called from the Encryption array and numbers will be assigned to that character. For example: - a character "A" is called 2 times on the 4th and 7th instance of the sequence. The numbers 4 and 7 will be associated with the character "A".

(VI) In the Encryption array each character will also have an index number, which starts from 0. In this step a new array will be created which will reserve index positions depending upon the index of the characters which were called earlier.

(VII) Each reserved position will be filled by the numbers associated to the characters, which were called earlier, in an ascending order filling the unreserved positions with the exact character from the Encryption array, which lies in the same index as that of the empty positions. If the reserved positions are greater than the actual number of blocks they will follow a repetition in the same pattern as mentioned above.

(VIII) In the last step of the process the reserved positions will be occupied by the characters associated with the numbers in those particular positions. The repetitions will also follow the said pattern. The Memory array works on a first in first out (fifo) principle.

E. The Vine Phase

This phase will be used for communication between the sender and the receiver. At the senders side it will fetch the MAC address of the receiver and the time stamp to the minute which should be same for both the thick clients and in the receiver's side it will fetch the MAC address of the sender. The Vine phase will also be used to send and receive the cipher text. The address resolution protocol (ARP) is a telecommunication protocol which is used for resolution of network link layer addresses (IP) into link layer addresses (MAC). ARP is used to map a network address (like IPv4) into a physical address (MAC). ARP can be implemented with many combinations of the networks and data link layers. But here we will only use it to extract the target Think Client's MAC address to generate a unique structure to create our key. [15]

The main part of the communication will be taken over a common shared server where on a successful login by both the thick clients, the time and MAC address will be stored and entered in a log table which will further be shared by both the thick clients. Now to prevent misuse a final crosscheck will be made by the respective thick clients by checking their own MAC addresses.

To extract MAC address we will use ARP and nmap, their MAC address will be extracted with the help of the IP addresses. The scanning will be done by nmap. Nmap is a port scanning tool which can scan all the connection made by both TCP and UDP which again is an advantage for us as we are not bound by single standard of protocol. On scanning and obtaining the IP addresses of both the thick client their respective MAC addresses will be shared. Now comes the use of ARP, ARP (Address Resolution Protocol) works on the layer 2 of OSI model and is used to fetch the MAC Address with the help of the IP address. After acquiring both the MAC addresses, for security reasons ARP will authenticate by accessing its own MAC address where the algorithm is running and if one MAC address out of two matches the thick client's MAC

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

address, it will proceed. The working of the ARP is shown in Fig.5.

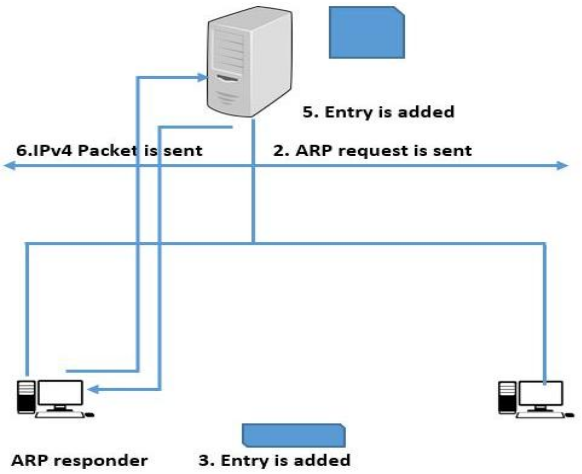

Fig.5 ARP request pattern

III.THE ALGORITHMS

A. The Key Keeper Algorithm

Key Keeper Algorithm (KKA) has two stages and they will run differently on the receiver side and the sender side. On the sender side, the KKA will simply create a 12 + x digit key as all the necessary inputs are acquirable. In Equation (1), the formula will generate the 12 digits for the sender side and **x** is the total number of characters present in the original text. ID_a and ID_b are identities for the two clients and $P_{time,}$ is the current time stamp. It can be a MAC address or any other type of identity, depending upon the technology used to connect the two thick clients.

$$f\left\{\left(ID_a \sim ID_b\right) \sim \left(g\left(P_{time}\right)\right)\right\} \quad (1)$$

Algorithm pseudo code: -

```
Convert the alphabets into numbers
 11-> A,12-> B, 13> C, 14-> D, 15->E, 16->F.
fetch MACadd(M1)
fetch MACadd(M2)
fetch Time(T1)
fetch original string (x)
for (i=n to 0; i--)
        num=add(M1[i], M2[i])
if (num=(M1[i]+M2[i]))
        num=M1[i]
else if (num>=10)
        Convert (num=A, B, C, D, E, F, G)
split till len(num)=12
        if n < 12 then n + x = 12
        if x = 3
                then n-1 * x, n-2 * x, n-3 * x
loop
        key=add (num, T1)
        if add>10
                Convert (add=A, B, C, D, E, F, G)
loop
        len(x)
        k. append(x%10)
        x=x/10
len(key)=12
        for i 0 to len(x)
                k1=Key[[concat(k)]  [i]  #  Memory Cursor
        for i 4 to 8
                k2=Key[i]          #          Reference Cursor
        for i 8 to 12
                k3=Key[i]          #          Encryption Cursor
```

This will create the x + 12-digit key and split the key into three cursors which is shown in Fig.6.

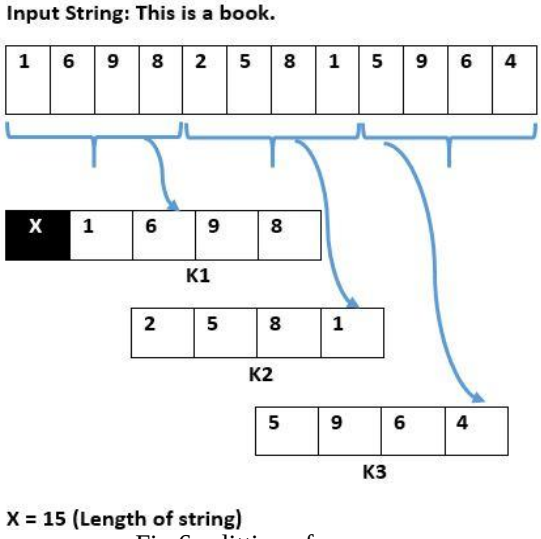

Fig.6 splitting of cursors

Stage two: -The stage two will run on the receiver side. This will work exactly the same as "stage one", but the only difference is, the length of the original text will not be known. The Reference cursor and the Encryption cursor will proceed normally while as the Memory cursor will be kept on standby until we acquire the length of the actual text. So in stage two the algorithm will start from the Encryption array

(I) The Encryption array will be overlapped by the cipher text and all the characters which are not identical will create a string of the remaining characters. The length of this string will be equal to the length of the original text. Now the Memory cursor can also proceed.

(II) This string of characters will be matched with the same characters in the Encryption array and the numbers associated with those characters will be fetched.

(III) The fetched numbers will be matched with the numbers in the Memory array and the characters associated with those numbers will be pushed and stored in a string. This string will be the actual text.

B. The Cursor Algorithm

This is the stage two of the KKA. It will take the three cursors which were created in the key keeper algorithm and concatenate the total number of characters used in the original plain text to the beginning of the string. The cursor algorithm is responsible for setting up the numbers

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

and characters in all the three arrays. Below are the steps which are undergone in the cursor algorithm:

1. Take all characters and symbols in standard keyboard and put them in actual sequences in an array or a list.
2. For Memory array take the first cursor, the memory cursor and take one number at a time from it, depending upon the value of the number shift the characters to the left or right. In the array index if the character is even take it to right if it is odd then take it to left. repeat this process until all the numbers of the first cursor have undergone this system and then repeat the whole process N times (N stands for the sum of characters in the key).
3. For Encryption array the process will be exactly the same, so read above. The only difference is that third cursor will be used here.
4. For Reference array fill the array with numbers from 0 to 9 and then use the same process to mix up the numbers by using the second cursor.
5. By using these three cursors in the Memory array, the Encryption array and the Reference array respectively we get three different strings which will be put into the arrays identically and in that particular sequence. It is shown in Fig.7.

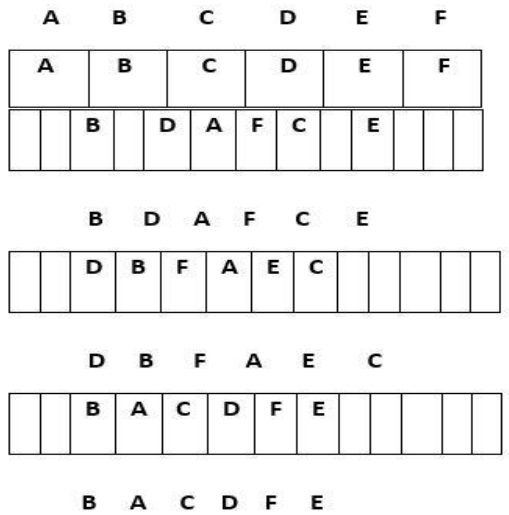

Fig.7 cursor algorithm

pseudo cursor algorithm: -

```
d=length of original string
a= [d, [4]], b= [4], c= [4]
#The four digits inside every key decides the arrangement of the characters in each and every array memory= [], reference= [], encryption= []
#>>[x] and <<[x] are referred to shift right and left by the number of x places in the list
#setting up the memory array when x is in a
 i from 0 to 95
      memory. append((>>[x]) a[i])
          i=i+2
 j from 95 to 0
              memory. append((<<[x]) a[i])
          j=j-2
#setting up the reference array when x is in b
 i from 0 to 95
      reference. append((>>[x]) a[i])
          i=i+2
 j from 95 to 0
              reference. append((<<[x]) a[i])
          j=j-2
#setting up the encryption array when x is in c
 i from 0 to 95
      encryption. append((>>[x]) a[i])
          i=i+2
 j from 95 to 0
              encryption. append((<<[x]) a[i])
          j=j-2
```

C. Limitations

As the concept the basic level there are certain limitations:

1. At the present level the TES system can only be used between two thick clients.
2. The uniqueness of the system is, the self-changeable key process which is done every minute so it gives only one minute for encryption and decryption.

IV. CONCLUSION

The TES system can give more secure method of communication. A server based system could be developed which will take the TES system to next level of secure communication.

REFERENCES

[1] Rivest, Ronald L. (1990). "Cryptography". In J. Van Leeuwen. Handbook of Theoretical Computer Science 1. Elsevier.
[2] Cusick, Thomas W. & Stanica, Pantelimon (2009). Cryptographic Boolean functions and applications. Academic Press. pp. 158–159. ISBN 9780123748904.
[3] Matt J. B. Robshaw, Stream Ciphers Technical Report TR- 701, version 2.0, RSA Laboratories, 1995
[4] Diffie, Whitfield; Hellman, Martin (8 June 1976). "Multi-user cryptographic techniques". AFIPS Proceedings 45: 109– 112.
[5] Diffie, Whitfield; Hellman, Martin (November 1976). "New Directions in Cryptography".IEEE Transactions on Information Theory. IT-22: 644–654. doi:10.1109/tit.1976.1055638.
[6] FIPS PUB 197: The official Advanced Encryption Standard". 5y Computer Security Resource Center. National Institute of Standards and Technology. Retrieved 26 March2015.
[7] Rivest, Ronald L.; Shamir, A.; Adleman, L. (1978). "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". Communications of the ACM (Association for Computing Machinery) 21 (2): 120–126. doi:10.1145/359340.359342.
[8] Schneier, Bruce (1996). Applied Cryptography (2nd ed.). Wiley. ISBN 0-471-11709-9.
[9] http://www.techterms.com/definition/thi kclient
[10] "IEEE 1588 Systems". National Institute of Standards and Technology (NIST).
[11] IEEE 1588-2002, IEEE, 8 November 2002, doi:10.1109/ IEEESTD.2002.94144
[12] man ASCII(7), "American Standard Code for Information Interchange"
[13] Kruse, Robert L. (1987) [1984]. Data Structures & Program Design (second edition).Joan L. Stone, Kenny Beck, Ed O'Dougherty (production process staff workers) (second (hc) textbook ed.).Englewood Cliffs, New Jersey 07632: Prentice-Hall, Inc. div. of Simon & Schuster. p. 150. ISBN 0- 13-195884-4.
[14] Chappell, Laura A. and Tittel, Ed. Guide to TCP/IP, Third Edition. Thomson Course Technology, 2007.
[15] Bruen, Aiden A. & Forcinito, Mario A. (2011). Cryptography information theory and error correction : a handbook for the 221 century \John Wiley & Sons. p. 21. I