ISSN:2231-0711
www.ijcset.net

Advanced Symmetric Key Cryptography Using Extended MSA Method: BLZ Symmetric Key Algorithm

Abhilash G, Sudhakar K N , Jitendranath Mungara

abhilashcta@gmail.com
sudhukn@yahoo.com
jmungara@yahoo.com

Abstract - The present work deals with new advanced symmetric key cryptographic method for multiple encryption and decryption of any file especially image file, sound file, video file, text file, executable file or any other file. Nath et. al. (1) developed an algorithm called MSA for encryption and decryption of any file using a 16x16 random key matrix. Recently Nath et.al (2) proposed a method using a randomized key of size 256x256 containing all possible 2 lettered words. The proposed method was an extension of MSA algorithm(1). In the present work we have extended the MSA algorithm one step further. The present work uses a randomized key of size 65536x256 which contains all possible 3 lettered words. The MSA algorithm is very much suitable for small system as the total number of random matrix can be generated is 256! and which may not very difficult task for the hacker to hack the actual key matrix. Nath et al.(2) recently communicated a paper where they have claimed that they defined a key matrix of size 256x256 where each element is a two lettered word. So the complexity to find the actual matrix will be 65536!. In the present work we propose a key matrix of size 65536x256 which contains all possible 3-lettered words. We use our own randomized method to make this key matrix random. So here the complexity of finding the actual key matrix will be 16777216! trial runs and which is intractable. So the current advanced BLZ method may be taken as the ultimate symmetric key method which can not be broken by using any brute force method. Moreover we have also introduced multiple encryption here to make our system more secured. This method will be suitable in any business house, government sectors, communication network, and defense network system. In the present work we have introduced a key matrix of size 65536x256 where in each cell we store all possible 3-lettered words (ASCII code 0-255). The total number of words possible is 65536x256 or 16777216. We randomize that matrix using the method proposed by Nath et. al(1). The user has to enter some secret text-key. The maximum length of the text key should be 16 characters long. To calculate the randomization number and the number of encryption to be done is calculated from the text-key using a method proposed by Nath et.al(1). The present method will be most suitable for encryption of a small file such as digital signature or watermarking etc. To encrypt/ decrypt unlike MSA method we don't need to store the random key in a file and randomize it for substitution, using our formula we can dynamically generate the array block and randomize it and do the encryption or decryption. This saves time for generating the random key of length (65536x256x3) and provides much flexibility. With our encryption method(which works like ENIGMA(6))one can easily confuse the hacker and makes it hard to break.

1 INTRODUCTION

Due to tremendous progress in internet now a days the security of data has become a very important topic in communication network. Specially when we are sending some open mail which may contain some personal data or confidential matter then there is no guarantee that no one is intercepting our message/data. A common problem we face is that we are getting lot of junk mails through spam mail and if reply to it knowingly or un-knowingly then the hacker will immediately know the IP address of my computer from my returned mail and after that he can do anything from his machine as all data in my machine are not encrypted. Consider another situation where the sender "A" is sending a message to receiver "B". There may be some hacker who is sitting in between to intercept the message. If sender "A" is sending the plain or clear message then the hacker can hack that message and can work on it and can send something else to receiver „B". When we send some confidential matter from one client to another client or from client to server then that data should not be intercepted by someone. We assume a situation where a bank manager is giving some instruction to his subordinate to credit some amount to a particular account and some intruder intercepted that message and do the reverse process means debit the amount from a particular Bank account. This may be further worse when a captain in the war field is sending some message to a soldier over the mail to move to some place to attack the enemy and assume that the message is been intercepted by some enemy. This type of unpleasant phenomena may happen if we send plain text or clear text from one place to another place. To avoid this

one has to send the encrypted text or cipher text from client to server or to another client. Cryptography is now an emerging research area where the scientists are trying to develop some good encryption algorithm so that no intruder can intercept the encrypted message. The modern classical cryptographic methods are of two types (i) symmetric key cryptography where the same key is used for encryption and for decryption purpose. (ii) public key cryptography where we use one key for encryption and one key for decryption purpose. Symmetric key algorithms are well accepted in the modern communication network. The plus point of symmetric key cryptography is that the key management is very simple one key is used for both encryption as well as for decryption purpose. There are various symmetric key methods are already established such as DES method, double DES method, Play fair method are some important symmetric key cryptographic methods. In case of symmetric key cryptography the key should remain as secret key. In public key cryptography the encryption key remains as public but the decryption key should be kept as secret key. The public key methods have got both merits as well as demerits. The problem of Public key cryptosystem is that we have to do massive computation for encrypting any plain text. Moreover in some public key cryptography the size of encrypted message may increase. In RSA public crypto system we cannot use the same key for multiple encryption and hence we have to define multiple encryption keys and similarly multiple decryption keys which may be a tedious jobs for maintaining keys by the user. Due to massive computation the public key crypto system may not be suitable in security of data in sensor networks. So the security problem in sensor node is a real problem. However, there are quite a number of encryption methods have came up in the recent past appropriate.

In the present work we are proposing a symmetric key method where we have used a ran-dom key generator for gene-rating the initial key and that key is used for encrypting the given source file. The present method is basically a substitution method where we read 6 characters as 2 blocks each contains 3 characters from any input file and then search the corresponding blocks in the random key matrix file to get the corresponding encrypted pattern and then we write the encrypted message in another file. For searching characters from the random key matrix we have used a method which was proposed by Nath et.al(1) in MSA algorithm. In the present method we have the provision for encrypting message multiple times. The key matrix contains all possible words comprising of 3 characters each generated from all characters whose ASCII code from 0 to 255 in a random order. The pattern of the key matrix will depend on text key entered by the user. To make the key matrix random we have used our own randomization algorithm which we generate from initial text_key. Nath et.al(1) proposed a simple algorithm to obtain the randomization number and encryption num-ber from the text_key. To decrypt any file one has to know exactly what is the key matrix and to find the ran-dom matrix theoretically one has to apply 16777216! trial run and which is intractable. We apply our method on possible files such as executable file, Microsoft word file, excel file, access database, foxpro file, text file, image file, pdf file, video file, audio file, oracle database and we found in all cases it gave 100% correct solution while en-crypting a file and decrypting a file. The present method can be used for encrypting digital signature, watermark before embedding in some cover file to make the entire system full secured.

2 GENERATION OF 65536X256X3 RANDOM KEY

To create Random key of size(65536x256x3) we have to select any text-key which is a secret key. The size of text-key must be less than or equal to 16 characters long. These 16 characters can be any of the 256 characters (ASCII code 0 to 255) The relative position and the character itself is very important in our method to calculate the randomization number and the encryption number. Here we take an example how to calculate randomization number, the encryption number from a given text-key. Now we will show how really we calculate the above two parameters:

We choose the following table for calculating the place value and the power of characters of the incoming key:

Table-1 Length of text-key and the corresponding value of base for the sensor nodes

Length of key(n)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Base value(b)	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2

Suppose key entered by the user is “AB”. The length of the text-key is 2.

To calculate the randomization number and encryption number we follow the following steps:

$$\text{Step-1: Take Sum}=\sum_{m=1}^{n}(\text{ASCII Code of Character}) * bm \text{ ----(1)}$$

So if our text-key is "AB" then from equation (1) we get Sum=$65*16^1 + 66 * 16^2 = 17936$

Now we have to calculate 2 parameters from this sum (i) Randomization number(n1) and (ii) Encryption number(n2) using the following method:

(i) To calculate Randomization number(n1): Calculate sum of product of each digit in Sum and its place value in that sum as follows:

num1=1*1+7*2+9*3+3*4+6*5=84

Now n1=Mod(sum, num1)=Mod(17936 , 84)=44

Note: if n1=0 then we set n1=num1 and if n1>64 then n1=n1-64

(ii) To calculate Encryption number(n2): Calculate the product of each digit in the Sum by its position in the sum in reverse order as follows:

num2=6*1+3*2+9*3+7*4+1*5=72

Now calculate n2=Mod(sum, num2) =Mod(17936,72) =8

Note: if n2=0 then we set n2=num2 and if n2>64 then we set n2=n2-64

Now we explain how we have made the random key of size 65536x256x3 which is used for encryption as well as for decryption purpose. We create a random key file in a step by step manner as follows. As it is difficult to store 50331648 (=256x256x3x256) elements in array hence we store the entire key in a file. But we do in a step by step manner. We divide the entire key into 4096 blocks where each block contains 64x64 words and each word contains3 characters. We create each block separately in computer in random order and then we apply randomization methods one by one on it and then write on to an external file again in some random order. The basic idea of randomization process is to make the key matrix totally random so that no one can guess it in advance.

We can also use the following formula to identify the position of the value in the file and get the array dynamically to randomize and substitute.

$$\text{Index}=\sum_{n=2}^{0}\text{ASCII value of char in word}*256^n$$

Now we show the original key matrix(256x256x3x256) which contains 1024x4 blocks of size 64x64x3 characterss:

Table -2: The original Key Matrix:

Block-1(64X64X3)	Block-2(64X64X3)	4	Block-1024(64X64X3)
Block-1025(64X64X3)	Block-1026(64X64X3)	4	Block-2048(64X64X3)
Block-2049(64X64X3)	Block-2050(64X64X3)	4	Block-3072 (64X64X3)
Block-3073(64X64X3)	Block-3074(64X64X3)	4	Block-4096(64X64X3)

We generate each block in a 3-dimensional array of size(64x64x3) where we store 3 lettered words starting from a word 000 to final word 255255255 in some random order. The words in each block we generate in computer internal memory and then apply 5 randomization methods one after another in a random order and then write onto key file again in random order.

The following randomization process we apply serially on each block internally. We apply the below methods in random to make the elements in each block as random as possible.

The following are the operations we execute serially one after another.

Table-3

Randomization steps
Step-1: Function cycling()
Step-2: Function upshift()
Step-3: Function rightshift()
Step-4: Function downshift()
Step-5: Function leftshift()
Step-6: Repeat Function down-shift()
Step-7: Repeat Function rightshift() Step-8: Repeat Function upshift() Step-9: Repeat Function cycling()

Now we describe the meaning of 5 above functions(Step-1 to step-5) when we apply on a 4x4x3 matrix as shown below:

Table-4 : Original table

AAA	ABA	ACA	ADA
BAA	BBA	BCA	BDA
CAA	CBA	CCA	CDA
DAA	DBA	DCA	DDA

Table-5:Called cyclin

BAA	AAA	ABA	ACA
CAA	BCA	CCA	ADA
DAA	BBA	CBA	BDA
DBA	DCA	DDA	CDA

Table-6:Called upshift

DAA	BAA	BBA	AAA
DDA	CCA	CDA	ADA
DBA	CAA	DCA	BCA
ACA	CBA	ABA	BDA

Table-7:Called rightshift

BDA	DAA	DBA	BAA
CAA	BBA	DCA	AAA
BCA	DDA	ACA	CCA
CBA	CDA	ABA	ADA

Table-8:Called downshift

ADA	DCA	ABA	AAA
BDA	BCA	DAA	DDA
DBA	ACA	BAA	CCA
CAA	CBA	BBA	CDA

Table-9:Called leftshift

ABA	CCA	AAA	CAA
ADA	ACA	DCA	BAA
DAA	CDA	DDA	DBA
BDA	CBA	BCA	BBA

The above randomization process we apply for n1 times and in each time we change the sequence of operations to make the system more random. Once the randomization is complete we write one complete block in the output key file. Now we show how we apply encryption process on a particular file. For this we choose our last randomized 4x4 matrix(table-9). We apply the following encryption methods:

While substituting data back to the output file, the first and second chars of each word is been added and negated by a integer value which is a step factor(which changes in loop and has range from 1 to n2).

For example if we have text in source file as abcabcabcabcabcabcabcabcabc and encryption number is 2, the output will be bacbacc`cc`cbacbacc`cc`cbacbac

A special care is taken to reset the value of the loop back to 1 if it reaches the calculated encryption value.

3 RESULTS AND DISCUSSION:

Here we are giving result which we obtain after we apply encryption method on a text file and also the decryption method on the decrypted file to get back original

(i) Text-key used=12

(ii)Randomization number created by our method : 5

(iii)Encryption number generated by our method : 9

The above values were used for encryption and decryption on a given text file(mydoc1.txt):

Group-A:

1. (i) Input n(1-20). Calculate s=1+2+3+4+...+n. Print n,s

   (ii)Input n(1-20).Calculate s=(1)+(1+2)+(1+2+3)+(1+2+3+4+n). Print n,s

   (iii)Input n(1-20). Calculate s=1-2+3-4+5-6+n. Print n,s.

   (iv)Input n(1-999999999). Print the sum of digits in the given number and print it.

2. Input n(2-3000). Print all fibonacci numbers <=n.Where f(n)=f(n-1)+f(n-2) for n>2 and fib(n)=1 for n<=2.

Use (a) non-recursive method , (b) Use recursive method.

3. Implement Towers of Hanoi problem by shifting all disks from Peg-1 to Peg-2 using Peg-3 one at a time where all the disks have different diameters and are arranged in sequential order. It means disk-1 is on the top position and disk-n is in the bottom position. The arrangement of disks should not be altered at any moment. Finally print how many movements have been taken to transfer all disks. Use recursive call.

4. Input 'n' different numbers and its index. Sort them in ascending order using any sorting method such as

(i) Bubble Sort or (ii) Insertion Sort or (iii) Merge Sort method and print the sorted list.

5. Input „n" numbers and also the number to be searched.

Use linear search method to search the number in the list.

If it is found then print where it is found and if the number not found then also print that the number not found.

6.. Solve linear simultaneous equations using matrix inversion method :

5x1 + x2 + x3 + x4 = 14
x1 + 5x2 + x3 + x4 = 18
x1 + x2 + 5x3 + x4 = 22
x1 + x2 + x4 + 5x4 = 26

7. Write a program to copy the content of one file to another file.

8. Write a program to split the content of one file to 2 or more files.

9. Write a program to search a pattern in any file and replace it by a new pattern. Print how many times the pattern has been modified.

10. Write a program to convert all small letters to capital letters in any file.

Group-B:

1. Input any image file and extract all pixels from the image and store in an array. Display the image on the screen using that array.

2.Input a color image and change it B/W and display on the screen.

3.. Input an image and rotate it by 1800.

4.Input 2 images and try to mix the two images.
5.Input an image and extract some arbitrary portion of the image and display it.
6. Input any sound file(.wav) and play it.
7. Input a sound file play only a few portion of it.
8. Input two sound files and mix those two files.
9. Input any B/W image and try to try to add colors into it.
10. Input a sound file and play it in reverse order.

Size of the original file:3256 bytes Encrypted file(output.c):
Size of encrypted file: 3256bytes

%' %' %' %' %' %' %' %' _ V
%'t€}X§(i '" ßrC_cl_le# u†Anš{oj)º %' %' %' %' %' %' %' %' %'
deQOr_{ bc ... D0_
sÇog;ae\f _{e÷r 9d 1HT IB†-o«ltYd ŒorÎuha~ _I 9
dTQ•B1HoÌkeòce%h 9d w=/º</Ù(º<Tÿfs ... dsÇog;ae\f _{e†r
b~ denur†b cy ... D.,,ys+Ce_daQ
O,Âoe*f*h._~fÙto_F.Ã-.NbS[1]_^B lo "K:ũqa%Wi_`p<`t¥ea '
1/2jq¥eslo•da«llIqrãleÂoi
šu 1HT_ IB2&/_VaÂo _ke | zt•* n` #•lJ | z`e9dizssjr«l 'iÂoa
fih~m _ vt)gh§(tc cm ÷{l
/2& _ a älsRdb;a n` #•l'o z`lûxu...tt a~r hhòC.Îuoßrvt'l
%weÂoa<`iũqp 't_`are2re*f*
ls1/2jdQ oQooct ... dtLhoòc.†-o«LaŒo ¿plvy |
_vnÊneK(
kcmúll,,+oBk º ÿ_Gé!uũjAº ø1ád
§()kpnCtø %(+c2_F.N*asluëltC:s".+283.,+f..$a.Ícr%
Wt"vn%'%' Ù((pa)ŸcpÜ$ 5}1~,0z:
xllbzl veŢw=D&)†11D&)À21À2+Õ%+.,+†13¿ +28.Ícr%Wt
×,+cò# %'ipa)ŸcpÜ$ 5}1~,0z: xllbzl_veŢw=Ŧ62ø%• 75[2]*+§(n
· P | jnŸ*nº ."v ád %'vkpn-Ntø
%(+c9ò79ò79§()ò7PŸ*n | jtp ŒomQof*f*hi 'tŦbiŒot•*)gvŦb
älmâpru_n ... d%w %'rÍctct! kpn
Ct• %(+c3~,0À? _{i3w «ll'oi_cnŠec_`n Ť,bh=sßr=Ù(¥&
ŒoW%'r‰cf".)Ç/(Òo1‰cfŦ, -Pf _*rV+>
+ca e _ kbC:)‰c +crPf<º .Ï8 _y %'e aò#n¥e9dc&•s>~eÂo
doò#,[2]-b`U
Âo4I0blÙte%b _{w*f*hsßrf Ť gaòCiar%wl0b cy \fiÃ~i'oL
n _cPÖ/-â`o * K(Ţwtšuebbh...deßrlŒot1/2&
a~sÛl".vûmdßrf&oeÙ(_{ ãld%'m£ze†r ... dd_`r".a_r
a2-eLjiÒys•*uãlt)gl*f*hr\erŸ*I§(m 'nÒid‰0kb~ 9d b~%' Ù(t#
Œop;ao%wtöfn... dnb~d[2]-k(o Ä~ Â
o ÍceÐkotwo~zpÙ$i¥&oûj ".hQ_aLja_re/nLh*f*h%'f
`kiQosvh_vl •n[2]- 'o älŧ[2]-eßra¿paŸ* )gm_mtÍcF• _{aäl `e'at •
*e • *tÎ%t~hnËfe3/4ca†-
`lf4I(. •zI%'uL*'Ÿ* ågfÛlr+ct)gu£ze<h cd_`t_ki 'e^` _
!n... dsÍctqgÂoe_došks[2]-hò#s... di>nB b^` Rdr"oŸ*ò# %'ipa
Ÿ#söftßrnũqo_V ád qi` n Ÿ#r^` Ť cr†rm
Ÿ*h_` ¸od | jp%'nŒotŸ* lrn`dzsihm.§(5º 5}ppa +c"xiu£ze<h
...dd `lÂo n` e<hu+ceÎ%t†-bÔ
is".r['e o%' Ù(UÂf `n_vrÒieÑ&c_t
`ltLhd voQ eÑ&c_ttâp Œomcräln".hQ 1 tb~C %' i ...
©`o«ld)ghQ• _{inu ¥eeùo 4v Ä~
Cd `f".hQ•nßrb_` «lt | goº d©` Âo %'eë | a•*oÍcr%
wtŢwhQ +ceÂou†-e<hnPf ¸ uº ._k6^`z(lÂfrÒiiÏnlLxn
'u¡deûjaJ | oš{ 1/4ui_b Ljt
cx | jn;arz`o†rm•*hø ¸oº %' Ÿ; %'x$ º,xòs+V+3/+ `1 [a]= _4Ù(
%' %' ç9 æ45$ ~*x$+%'4òs ~_1H!
%' %' º,x%'+/+2òs /+3_p %' `1=º 2Ì; $
ç9xòs+V+4 = GH! `12ºl_ 7 ·s §(iK(lpË
f,gÞjmQ•o_foÂo Çze_{o_fe_{ Ûm «le9diRd Aoo ` eIlf†-
e_" 4vršUenu ...doÙ$aâtt!fsÎ%iŒotŸ * ‡hn lnŸ*oå,o e
â`lÎ%t".2_dr*f*ho e ¥elº .ûx9 ·s §(iK(
lpËfgÞjm_voQ•eÓ&c_taÝwa%
%'eŢwp •#t§(nßrPŸ*n | jh_b LwnbbtA!eŒot ' ltdr_sh•* •*
eälm ei oeţ2['0 | jW§(t%wa".r"frLg ‡h #•n `a×k
3/4caRd 3/4ct'r ltKxcÎ%ihmlÏne£zeÝw _` Ícyº
%')'l e,v[1]_o?9- .p*f* <[1]_.kpnCtA!n... di".g_bf... deIln` qe[2]-r...
dt_clEvp1/2&e¢r ë | oŒotÙ$ Ť cabb
_vd_`tº e_q %' Íc _t _`rQO.ksiksl w
ceÂom*f*heOfn".hQ s)ge×u Ljiš{ a r§(2ţ2IŸ*u
•zaŨfo_frOfm
ce[2]-n... dc Ť cnûm ÐK 4v/_ka2& !fšÛla •*oA!thn Œor
ncC.o;n...dtkpnOfmce[2]-n...dr la ¸ • 4
v cy Ý-0N=§(4ţ2IŸ*u•z2Ofmce_ka ' Q•yÑxoE#i_dt1/4r
ŒooOfmceţ2_)5•zI§(u•*aŸ*i".g_ba×w _ktŸ*
cËfs...dejrËftcir_qpA!t*f*hnöff".hQ i".g_baÙ(_k Ä~d
%'pcysltŒ7 .kpnCtA!n... ds[2]-nvf‰ #eIlw... d)€}nslp2-ytc
C.[3]6n... dtkp _kun` ".l ep*f*hyslnK(Âvf_a _srûjo •*o"-
iå,§(8ţ2IŸ*u•ztn` _hu e _kl_` ûxdE#i_dt".s`lt e _hlº .ûx1pa
¤,p_` 5}y2&/ÐKi".g_baÑx _ky t • t,, Î%tA!a‡h
^dlcs_qn4v #•.Ý-2º 5}ppa vsK(nIIf[2]-e[2]- n ...
dpcysltÌ{nce`rz` ßrd_q.

4 CONCLUSION:

In the present work we use the maximum encryption number=64 and maximum randomization number=64. We have used the key matrix of size 65536x256x3 . This key may be generated in 16777216! ways. So in principle it is not possible for anyone to decrypt the encrypted text without knowing the exact key. Our method is essentially block cipher method and it will take more time if the files size is large and the encryption number is also large. The merit of this method is that it is almost impossible to break the encryption algorithm without knowing the ex-act key matrix. We propose that this encryption method can be applied for data encryption and decryption in banks, in defense, in government sectors for sending confidential data. This work may be extended up to 4 lettered words and then one has to create a key matrix of size 16777216 X 256 X 4 for encryption and decryption process and that will be the most effective method for encrypting small file because it will be almost unbreakable.

REFERENCES

[1] Symmetric key cryptography using random key generator, A.Nath, S.Ghosh, M.A.Mallik, Proceedings of International conference on SAM-2010 held at Las Vegas(USA) 12-15 July,2010, Vol-2,P-239-244

[2] A new Symmetric key Cryptography Algorithm using extended MSA method :DJSA symmetric key algorithm , Dripto Chatterjee, Joyshree Nath, Suvodeep Dasgupta and Asoke Nath, communicated for publication at IEEE conference to held at Singapore from 14/06/2011 to 17/06/2011.

[3] Data Hiding and Retrieval, A.Nath, S.Das, A.Chakrabarti,

Proceedings of IEEE International conference on Computer Intelligence and Com-puter Network held at Bhopal from 26-28 Nov, 2010.

[4] Cryptography and Network , William Stallings , Prectice Hall of India

[5] Modified Version of Playfair Cipher using Linear Feedback Shift Register, P. Murali and Gandhidoss Senthilkumar, UCSNS International journal of Computer Science and Network Security, Vol-8 No.12, Dec 2008.

[6] "Enigma Variations: an Extended Family of Machines", Hamer, David H.; Sullivan, Geoff; Weierud, Frode (July 1998)., Cryptologia, 22(3)

1

AUTHORS

Abhilash G is currently pursuing his Master of Technology with a major of Computer Network Engineering at CMRIT, Bangalore.

Sudhakar K N is currently working as associate professor in department of computer science at CMRIT, Bangalore.

Dr. Jitendranath Mungara is currently working as Dean of PG Studies at CMRIT, he is having double doctorate and has published numerous international papers in field of mobile ad hoc networks.