# Symmetric Key Cryptography Using Random Key Generator.

Asoke Nath
St. Xavier's College, Kolkata
**278** PUBLICATIONS **2,061** CITATIONS

# Symmetric key cryptography using Random key generator

**Asoke Nath[1], Saima Ghosh[2] and Meheboob Alam Mallick[3]**

**[1,2,3]Department of Computer Science**
**St. Xavier's College(Autonomous)**
**30, Park Street**
**Kolkata-700 016**
**West Bengal**
**India**

**Abstract -** *In the present work we have introduced a new symmetric key cryptographic method for encrypting as well as decrypting any file such as binary file, text file or any other file. In our method we have modified the idea of Play fair method into a new platform where we can encrypt or decrypt any file. We have introduced a new randomization method for generating the randomized key matrix to encrypt plain text file and to decrypt cipher text file. We have also introduced a new algorithm for encrypting the plain text multiple times. Our method is totally dependent on the random text_key which is to be supplied by the user. The maximum length of the text_key can be of 16 characters long and it may contain any character(ASCII code 0 to 255). We have developed an algorithm to calculate the randomization number and the encryption number from the given text_key. The size of the encryption key matrix is 16x16 and the total number of matrices can be formed from 16 x 16 is 256! which is quite large and hence if someone applies the brute force method then he/she has to give trail for 256! times which is quite absurd. Moreover the multiple encryption method makes the system further secured. We propose that our method could be appropriate in sensor network where the massive computation is not possible but the security of data is important at the same time.*

## 1 Introduction

Symmetric key cryptography is well known concept in modern cryptography. The plus point of symmetric key cryptography is that we need one key to encrypt a plain text and the same key can be used to decrypt the cipher text. There are various methods which 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 main problem is that the same key is used for encryption as well as decryption process. Hence the key must be secured. Because of this problem we have introduced public key cryptography such as RSA method, AES method etc. These 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. Some times these methods may not be also suitable such as in sensor networks. As we know that in sensor network the main problem is the problem of power of the sensors. The battery of the sensor node can not be rechargeable and hence for encrypting data if the sensor nodes remains open for long time then ultimately the sensors nodes will be fully discharged and will not be able send any signal to other nodes. 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 for the sensor nodes. In the present work we are proposing a symmetric key method where we have used a random key generator for generating the initial key and that key is used for encrypting the given source file. Our method basically a substitution method where we take 2 characters from any input file and then search the corresponding characters from the random key matrix and store the encrypted data in another file. For searching characters from the random key matrix we have used a different algorithm from Play fair method. In our method we have the provision for encrypting message multiple times which is not possible in Play Fair method. The key matrix contains all possible characters(ASCII code 0 to 255) in a random order. The pattern of the key matrix will depend on text_key entered by the user. Here we are proposing our own algorithm to obtain randomization number, encryption number and the shift parameter from the initial text_key. We have given a long trial run on text_key and we found that it is very difficult to match the three above parameters for 2 different Text_key which means if some one wants to break our encryption method then he/she has to know the exact pattern of the text_key otherwise it will not be possible to obtain two sets of identical parameters from two different text_key. We have given several trial runs to break our encryption method but we found it is almost unbreakable. For pure text file we can apply brute force method to decrypt small text but for any other file such any binary file we can not apply any brute force method and it not work.

## 2 Random Key Encryption Algorithm

To create Random key Matrix of size(16x16) we have to take any key. The size of 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 , the encryption number and the relative shift of characters in the starting key matrix. We take an example how to calculate randomization number, the encryption number and relative shift from a given key. Here we are demonstrating our method:

Suppose key=AB

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

Table-1:

| 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 |

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

Example-1:

Now we calculate the sum for key="**AB**" using equation(1)

$\text{Sum}=65*16^{1} + 66 * 16^{2}$

$=17936$

Now we have to calculate 3 parameters from this sum (i) Randomization number(n1), (ii) Encryption number(n2) and (iii)Relative shift(n3) using the following method:

(i) Randomization number(n1):

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

n1=sum mod num1=17936 mod 84=**44**

**Note: if n1=0 then n1=num1 and n1<=128**

(ii) Encryption number(n2):

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

n2=sum mod num2 =17936 mod 72 =**8**

**Note: if n2=0 then n2=num2 and n2<=64**

(iii) Relative shift(n3):

n3= Σall digits in sum=1+7+9+3+6=**26**

Example-2:

Now we will calculate the sum for key="**AC**" using equation(1)

$\text{Sum}=65*16^{1} + 67 * 16^{2}$

$=18192$

Let us now calculate 3 parameters from this sum (i) Randomization number(n1), (ii) Encryption number(n2) and (iii)Relative shift(n3) using the following method:

(iv) Randomization number(n1):

num1=1*1+8*2+1*3+9*4+2*5=66

n1=sum mod num1=18192 mod 66=**42**

**Note: if n1=0 then n1=num1 and n1<=128**

(v) Encryption number(n2):

num2=2*1+9*2+1*3+8*4+1*5=60

n2=sum mod num2 =18192 mod 60 =**12**

**Note: if n2=0 then n2=num2 and n2<=64**

(vi) Relative shift(n3):

n3= Σall digits in sum=1+8+1+9+2=**21**

Now we show the original key matrix(16 x 16) which contains all characters(ASCII code 0-255):

Table –2 : The original Matrix:

☺ ☻ ♥ ♦ ♣ ♠
♫ ☼
► ◄ ↕ ‼ ¶ § ▬ ↨ ↑ ↓ ← ∟ ↔ ▲
▼
! " # $ % & ' ( ) * + , - . /
0 1 2 3 4 5 6 7 8 9 : ; < = > ?
@ A B C D E F G H I J K L M N O
P Q R S T U V W X Y Z [ \ ] ^ _
` a b c d e f g h i j k l m n o
p q r s t u v w x y z { | } ~ ⌂
Ç ü é â ä à å ç ê ë è ï î ì Ä Å
É æ Æ ô ö ò û ù ÿ Ö Ü ¢ £ ¥ ₧ ƒ
á í ó ú ñ Ñ ª º ¿ ⌐ ¬ ½ ¼ ¡ « »
░ ▒ ▓ │ ┤ ╡ ╢ ╖ ╕ ╣ ║ ╗ ╝ ╜ ╛ ┐
└ ┴ ┬ ├ ─ ┼ ╞ ╟ ╚ ╔ ╩ ╦ ╠ ═ ╬ ╧
╨ ╤ ╥ ╙ ╘ ╒ ╓ ╫ ╪ ┘ ┌ █ ▄ ▌ ▐ ▀
α ß Γ π Σ σ µ τ Φ Θ Ω δ ∞ φ ε ∩
≡ ± ≥ ≤ ⌠ ⌡ ÷ ≈ ° ∙ · √ ⁿ ² ■

We will now describe our Randomization method
The following are the operations we execute serially one after another.
Step-1: Function cycling()
Step-2: Function upshift()
Step-3: Function downshift()
Step-4:Function leftshift()
Step-5:Function rightshift()
Step-6:Function random()
Step-7:Function random_diagonal_right()
Step-8:Function random_diagonal_left()

Now we describe the meaning of all above functions when we apply on a 4x4 matrix as shown below:

Table-4: Original table

| | | | |
|---|---|---|---|
| A | B | C | D |
| E | F | G | H |
| I | J | K | L |
| M | N | O | P |

Relative shift( by 1)

| | | | |
|---|---|---|---|
| .A | B | C | D |
| E | F | G | H |
| I | J | K | L |
| M | N | O | P |

For I=1 to n1

Call Leftshift()

Table-3 : The matrix after relative shift(n3=26) is:

4 N h é £ ╢ ╨ Ω ☺ ← 5 O i â
¥ ╖ ╤ δ ☻ ∟ 6 P j ä ₧ ╕ ╥ ∞ ♥ ↔
7 Q k à ƒ ╣ ╙ φ ♦ ▲ 8 R l å á ║
╘ ε ♣ ▼ 9 S m ç í ╗ ╒ ∩ ♠ : T
n ê ó ╝ ╓ ≡ ! ; U o ë ú ╜ ╫ ±
" < V p è ñ ╛ ╪ ≥ # = W q ï
Ñ ┐ ┘ ≤
$ > X r î ª └ ┌ ⌠ ♂ %
? Y s ì º ┴ █ ⌡ ♀ & @ Z t Ä ¿ ┬
' A [ u Å ⌐ ├ ▌ ≈ ♫ ( B \
v É ¬ ─ ▐ ° ☼ ) C ] w æ ½ ┼ ▀ ∙
► * D ^ x Æ ¼ ╞ α · ◄ + E _ y ô
¡ ╟ ß √ ↕ , F ` z ö « ╚ Γ ⁿ ‼ -
G a { ò » ╔ π ² ¶ . H b | û ░ ╩
Σ ■ § / I c } ù ▒ ╦ σ ▬ 0 J d
~ ÿ ▓ ╠ µ ↨ 1 K e ⌂ Ö │ ═ τ ↑ 2
L f Ç Ü ┤ ╬ Φ ↓ 3 M g ü ¢ ╡ ╧ Θ

| | | | |
|---|---|---|---|
| C | L | D | M |
| A | J | B | K |
| G | P | H | I |
| E | N | F | O |

Call cycling()

| | | | |
|---|---|---|---|
| A | C | L | D |
| G | B | H | M |
| E | J | P | K |
| N | F | O | I |

. Call upshift()

| | | | |
|---|---|---|---|
| E | A | J | C |
| O | H | I | M |
| N | G | F | B |
| D | P | L | K |

Call rightshift()

| | | | |
|---|---|---|---|
| K | E | N | A |
| G | J | F | C |
| B | O | D | H |
| P | I | L | M |

Call downshift()

| | | | |
|---|---|---|---|
| M | F | L | C |
| K | B | E | O |
| N | D | A | H |
| G | P | J | I |

After finishing above shifting process we perform
(i)column randomization
(ii)row randomization and
(iii)diagonal rotation and
(iv)reverse diagonal rotation.

Each operation will continue for i,i-1,i-2 till the value becomes 1.

Next I

Now we apply encryption process on any text file. Our encryption process is as follows:

We choose a 4X4 simple key matrix:

Table-5:

| | | | |
|---|---|---|---|
| A | B | C | D |
| E | F | G | H |
| I | J | K | L |
| M | N | O | P |

Case-**I** : Suppose we want to encrypt **FF** then it will taken as **GG** which is just one character after F in the same row.
Case –**II** : Suppose we want to encrypt **FK** where **F** and **K** appears in two different rows and two different columns. **FK** will be encrypted to **KH (FK→GJ→HK→KH).**
Case-**III**: Suppose we want to encrypt **EF** where **EF** occurs in the same row. Here **EF** will be converted to **HG.**

## 3 Results and Discussion

Here we are giving a real live solution for our encryption method:

(i) Key used:1234
(ii)Randomization number created by our method : 98
(iii)Encryption number generated by our method : 40
(iv) Relative shift generated by our method: 23
The above values were used for encryption and decryption of a given text file(readme.txt) which is given in the next page:

**Table-6: The Key Matrix generated after relative shift:**

```
 ↨ . E \ s è í ╕ ╧ µ ² ☺ ↑ / F
] t ï ó ╣ ╨ т ■ ☻ ↓ 0 G ^ u î ú
║ ╤ Φ  ♥  1 H _ v ì ñ ╗ ╥ Θ ♦
← 2 I ` w Ä Ñ ╝ ╙ Ω ♣ ∟ 3 J a
x
Å ª ╜ ╘ δ ♠ ↔ 4 K b y É º ╛ ╒
∞
 ▲ 5 L c z æ ¿ ┐ ╓ φ ▼ 6 M d
{ Æ ⌐ └ ╫ ε      7 N e | ô ¬ ┴
╪
∩
 ! 8 O f } ö ½ ┬ ┘ ≡ ♂ " 9 P
g ~ ò ¼ ├ ┌ ± ♀ # : Q h ⌂ û ¡ ─
 $ ; R i Ç ù « ┼ ▄ ≤ ♫ % <
S j ü ÿ » ╞ ▌ ⌠ ☼ & = T k é Ö
░
╟ ▐ ⌡ ► ' > U l â Ü ▒ ╚ ▀ ÷ ◄
(
? V m ä ¢ ▓ ╔ α ≈ ↕ ) @ W n à
£
│ ╩ ß ° ‼ * A X o å ¥ ┤ ╦ Γ · ¶
+ B Y p ç ₧ ╡ ╠ π · § , C Z q
ê
ƒ ╢ ═ Σ √ ▬ - D [ r ë á ╖ ╬ σ ⁿ
```

**Table-7: The Final Randomized Key Matrix after we apply random generator program**:

```
╦ ; J u x ♦ > ⌠ └ ? Φ ╛ ║ ≥ ƒ ⌡
$ ↓ ├ Σ , ▓ K ╟ 8 U Ü ú æ ╙ )
╘
± # ╨ V α @ ' ∟ ╧ ┬ r ¡ ╒ Ç ¬
«
 ╥  \ ó * ε ² ⌐ ♣ c
O ë Θ : Ö v " ▄ Q  ö § ┼ ┐ 2 i
1 É 7 P Γ  ▐ ■ F ◄ ╢ ≈ ▬ a ≤
ü
f ∩ Z B g ╠ Ñ z ╔ t √ Ω ç o ╓ ╤
° { X G ← ☻ ` » ╞ 0 ↨ T m l │
y
▼ á ⁿ ∞ E l j n S ê í [ N ì D <
 e ↑  W δ ╪ σ π  ┌ b ┴  ░ â
♀ ╜ s h ▀ % ~ / µ ¼ ─ è · q 4
w
╕ ╝ ╖ · ¿ 6 · ╬ ⌂ ► 5 ò ♥ p å
₧
▲ ╢ ╚ ! ä k ♂ - _ Y ▌ é ^ ☼ ÿ
9
M ♠ φ Å ≡ ¶ d ñ ª ↔ L
 ╩ î = C
Ä т 3 + R & ░ A ù û ┤ } | ☺ ‼ à
] º ÷ ╗ ♫ ↕ ¥ £ ß H ┘ ô ═ ╫ ( ï
```

**Original File(Readme.txt):**

The society of Jesus, a Christian Religious Order founded by Saint Ignatius of Loyola in 1540, has been active in the field of education through out the world since its origin. In the world, the Society of Jesus is responsible for over 1865 Educational Institutions in 65 countries.These Jesuit Educational Institutions
Institutions in 65 countries.These Jesuit Educational Institutions engage the efforts of approximately 98,000 teachers.They educate approximately 17,92,000 students.

Size of the original file:425 bytes

```
Y6ÙÌ_ÜY /kÌ□ë¹´¾´‡ëlkœ
üÜÙï¯lž6Õ       pz_Ül
ÙëÉ7)  üë□Ìw)
          ªë!/BuŽ_zžek__ž_e¯¾´kë□ë'Ì/ëp_k
¯ž6Æ3>{‡ë
_Ùë.Y_wkl_ïzlY6zžke,_ë□zY□)kë□«»Y)¾_
_e¯lžkeŠXl
ùŠkë
 ke,_k¤Ìü□)ëÙzžü         k¯ïÙkëüÜ_Üw
k%ž6
Y6¤ëX□(Òke,_BuŽÌ_ÜY /kÌ□ë¹  ´¾´k¯ÙëX
          Ù6lžÙÜ.pY6□Ìüëll
          XëæfpÃëù(¾üle¯ëž_pPu%<´e¯
e¯lžÙëzž6_Ãë_ÌwïüzY´•_
          ´Y6"     ´¾¯eëÊ)¾__e¯lž_pk%<´e¯
e¯lžÙ«»Y<ù__Y6
Y6
          æ□ÌüïÙëÌ□kl&&üÌ_Üð_ YpåkœOÒ
½½½k Y__,_7Þ}_,_/B»Y)¾__ Ykl&&üÌ_Üð_
Ypåëæ_Ò-~‡{½½ëÙ
)          ž ´•
```

engage the efforts of approximately 98,000 teachers.They
educate approximately 17,92,000 students.

**Size of the original file:425 bytes**

**Size of the original file:425 bytes**
**Encrypted file(X1.txt)**:

_

**Size of encrypted file: 425 bytes**

**Decrypted file(X2.txt)**:

The society of Jesus, a Christian Religious Order founded by Saint Ignatius of Loyola in 1540, has been active in the field of education through out the world since its origin. In the world, the Society of Jesus is responsible for over 1865 Educational Institutions in 65 countries.These Jesuit Educational Institutions engage the efforts of approximately 98,000 teachers.They educate approximately 17,92,000 students.

**Size of decrypted file: 425 bytes**

## 4 Conclusion

In the present work we use the maximum encryption number=64 and maximum randomization number=128. The present work is a substitution method and can be used to replace a character by any of the 256 characters. In the present work the key matrix may be generated in 256! Ways. So in principle it will be difficult for any one to decrypt the encrypted text without knowing the exact key matrix. Our method is essentially stream cipher method and it may take huge amount of time if the files size is large and the encryption number is also large. The merit of this method is that if we change the key_text little bit then the whole encryption and decryption process will change. This encryption method can be applied for data encryption and decryption in banks, railway reservation systems, ATM, in defense, in sensor nodes. We have already started to work on pair of characters to make the encryption process more secured and that will be almost impossible to break encryption code.

## 5 Acknowledgement

We are very much grateful to Department of Computer Science to give us opportunity to work on Cryptography. One of the authors (AN) sincerely expresses his gratitude to Prof. S.Agarwal, Prof. S.Das of Dept. of Computer Science for giving constant inspiration to complete this work. We are also thankful to all 3rd year Computer Science Hons. Students (2010 batch) for their encouragement to finish this work.

**References:**

[1] Abhijit Das and C.E. Madhavan : Public-Key Cryptography Theory and Practice, Pearson Education

[2] William Stalings : Cryptography and Network SecurityData Encryption method and Network security, PHI

[3] C. Kaufman, R. Perlman and M. Speciner, Network Security Private Communication in a PUBLIC World, PHI

[4] Atul Kahate, Network Security and Cryptography, PHI