*ACCENTS Transactions on Information Security, Vol 1(2)*
*ISSN (Online): 2455-7196*

Research Article

# Key generation technique by data diffusion through prism

**Aashijit Mukhopadhyay***
Department of Computer Science, St. Xavier's College (Autonomous), Kolkata, India



## Abstract
***One of the main challenges in cryptography is key generation for a particular any symmetric key algorithms. The foremost idea in key generation lies in using very small passwords which can easily be remembered by the users and creating keys of large size for the symmetric key encryption procedure [1][2][3] to be quite strong. The particular key generation technique forms a huge sized key of 64 bytes even when the password provided by the user comprises of a single byte. The main procedure makes use of theory of white light beam when made to fall on prism gives rise to a spectrum of light rays as velocity of light depends on the wavelength of light in a particular medium. Data given as password from the user has been converted to its hexadecimal equivalent and stored in a list. This list is then created as input to Fisher Yates Shuffling Algorithm and a pseudo-random shuffled list is extracted as output. Then random positions are extracted from this list and we get our particular angle of incidence, the refractive indices of the three mediums and the different angles of the prism. Thus we place data in a two dimensional data matrix in the particular angle as calculated. The data is made to fall on the first surface of the prism. As soon as the data falls, the data breaks into numbers of a lower base. Each byte is xor-ed with each other and both the resultant byte and an original byte are stored. The bytes are then stored in another matrix in an angle where the angle is calculated using the Snell's law. Data is placed in a two dimensional matrix inside the prism according to the given angle of refraction. Then the data is made to fall on the second inner face of the prism. After refraction the data again spits to numbers of lower base and thus creates a larger matrix. At all the time of positioning of data in the matrix according to given calculated angles there will always remain some empty spaced cells. These cells are filled up with random data of the particular number system the other part of the data is in. This matrix can then be used for any Symmetric Key encryption algorithm that needs to be much secured with a large key size as well as very user-friendly with a very small sized password.***

## Keywords
***Prism, Key generation, Symmetric key cryptography, Small passwords, Data diffusion.***

## 1.Introduction

### 1.1Background

As time is flying over, systems are taking birth that has intense processor speed to run on. This has made the work of Symmetric Key Cryptographic algorithms very hard. Algorithms that run on small key sets can easily be broken down by brute force attacks even when the passwords entered by the user tend to be quite large. In the present Key generation algorithm, the author has introduced a new method which can help symmetric key cryptographic algorithms to ask for small passwords from the users and work on huge key sizes to enhance security and add a strong barrier against brute force attack.

### 1.2Method Used

The present algorithm uses theory of white light being broken into constitutive components when made to fall on a surface of prism.

Data taken as password from the user is first converted to hexadecimal number system and stored in a list. The list is used as input to the Fisher Yates Shuffling Algorithm and then a randomly shuffled list is received as output. Then extracting data bytes from random parts of the list the algorithm calculates the angle of incidence to the prism, the three refractive indices of the three mediums respectively and the different angles of the prisms.

The angle of incidence is used by the algorithm to calculate the angle at which data must reside in a two dimensional matrix and data is stored in the matrix. The empty cells are filled up by random bytes of the particular number system. This data is made to fall on the first surface of the prism. Data divides into a lower number system. Each byte is xor-ed with each other and both the resultant byte and an original byte are stored. The bytes are then stored in another matrix in an angle where the angle is calculated using the Snell's law. This matrix is again made to fall on the next surface of the prism. Data in a similar

*Author for correspondence

fashion broken up into a lesser number system and stored in another matrix in the angle calculated from Snell's Law. The matrix is much larger in size as compared to the entered password and can be used to encrypt useful data using any Symmetric Key Algorithm. The present algorithm has been successfully tested in driving out Brute Force attacks over any Symmetric Key algorithm that takes the help of this algorithm as it derives a large sized key from a small password given by the user.

### 1.3Objective

The main objective of the system has been to create a key generation procedure which can help Symmetric Key Cryptography algorithms to be more secured by using large sized key bytes. This large key can be generated using a small password given by the user. So, the user needs to remember a smaller password and can easily secure their highly important data using a large sized key with any Symmetric Key Cryptography algorithm.

## 2.Block diagram

The equations are an exception to the prescribed specifications of this template. You will need to determine whether or not your equation should be typed using either the times new roman or the symbol font (please no other font). To create multileveled equations, it may be necessary to treat the equation as a graphic and insert it into the text after your paper is styled.

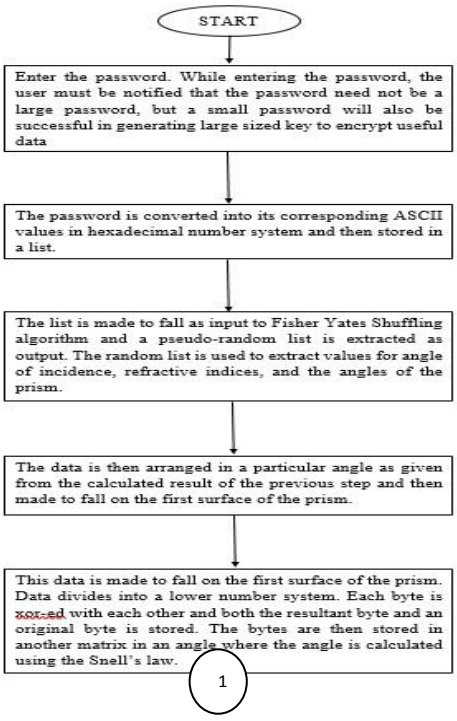


**Figure 1** Block diagram

**Equations**

Snell's Law Equation:

$$n_1 \sin\theta_1 = n_2 \sin\theta_2 \qquad (1)$$

where n1 is the refractive index of the first medium, n2 is the refractive index of the second medium. $\theta_1$ is the angle of incidence and $\theta_2$ is the angle if refraction respectively.

## 3.Algorithm

**Step 1:** Input a small password to the system.

**Step 2:** The password is converted into its corresponding ASCII values in hexadecimal number system and then stored in a list.

**Step 3:** The list is made to fall as input to Fisher Yates Shuffling algorithm and a pseudo-random list is extracted as output. The random list is used to extract values for angle of incidence, refractive indices, and the angles of the prism.

**Step 4:** The data is then arranged in a particular angle as given from the calculated result of the previous step and then made to fall on the first surface of the prism.

**Step 5:** This data is made to fall on the first surface of the prism. Data divides into a lower number system. Each byte is xor-ed with each other and both the resultant byte and an original byte is stored. The

bytes are then stored in another matrix in an angle where the angle is calculated using the Snell's law.
**Step 6:** The data that comes out is then stored inside the prism in the two dimensional matrices in the calculated angle of refraction given by Snell's Law.
**Step 7:** This beam if data is again made to fall on the third surface of the prism. Data is similarly broken into numbers of smaller base and each byte was xor-ed with each other and then stored on a matrix in angle calculated using the Snell's law.
**Step 8:** This resultant matrix can be used by any other Symmetric Key Cryptographic algorithm.

## 4.Results and discussions

**Test Case 1:** A trivial single byte password is used to test the strength of the key generation procedure for the first time.

```
Password Entered: 1
Resultant data Array:
4   5   1   0   2   7   7   6
4   1   0   5   2   6   7   0
1   5   5   2   0   6   7   3
2   1   5   1   0   3   4   5
3   3   2   1   4   1   1   3
0   2   0   2   3   2   5   4
6   7   4   1   4   0   4   124
6   1   1   1   5   61  77  74
```

**Figure 2**Key driven from the given small password

A 64-byte key array is the output even for a small password of a single byte size.

**Test Case 2:** A trivial single byte password is used to test the strength of the key generation procedure for the second time.

```
Password Entered: 1
Resultant data Array:
0   3   2   3   1   0   2   3
3   0   1   6   7   6   7   1
0   7   5   3   1   1   1   1
1   2   1   4   6   3   2   4
2   0   5   0   5   1   6   0
1   0   33  24  1   5   7   7
133 3   4   5   7   33  51  42
31  2   7   6   6   7   142 34
```

**Figure 3** Key driven from the same given small password for the second time

**Test Case 3:** A different trivial single byte password is used to test the strength of the key generation procedure for the first time.

```
Password Entered: 0
Resultant data Array:
6   2   0   7   7   1   7   5
1   0   7   0   2   3   0   6
7   3   3   7   2   0   1   1
2   6   4   1   1   124 5   135
44  113 5   5   4   1   4   33
127 112 7   1   5   0   5   1
54  136 6   5   4   6   0   7
73  63  6   7   3   1   6   4
```

**Figure 4** Key driven from a different small password for the first time

**Test Case 4:** A different trivial single byte password is used to test the strength of the key generation procedure for the second time.

```
Password Entered: 0
Resultant data Array:
5   4   5   5   4   4   5   1
1   0   1   0   2   2   3   1
0   3   3   7   3   7   1   1
1   0   1   7   1   2   6   116
47  103 0   6   7   7   130 23
70  105 1   3   6   1   1   20
65  45  2   0   3   4   0   5
41  114 1   3   7   6   1   4
```

**Figure 5** Key driven from a different small password for the first time

**Test Case 5:** A different trivial small password is used to test the strength of the key generation procedure.

```
Password Entered: 0000
Resultant data Array:
2   1   3   2   3   3   4   4
1   5   4   0   1   5   3   4
1   4   5   5   2   0   4   5
0   6   1   1   6   4   3   7
7   3   1   4   1   1   6   0
4   5   1   0   1   1   6   1
142 43  13  141 73  1   1   4
14  132 140 105 44  143 1   1
```

**Figure 6** Key driven from four consecutive zeroes taken as password

**Test Case 6:** A different trivial small password is used to test the strength of the key generation

procedure. This has a single byte difference with the last password used for test.

```
Password Entered: 0001
Resultant data Array:
6   1   1   1   5   1   1   3
1   2   7   6   4   5   1   5
1   6   7   0   1   0   0   1
1   5   6   1   2   0   0   7
1   4   1   1   6   1   1   5
3   0   0   1   2   1   2   3
4   7   1   0   1   3   0   1
111 15  114 30  11  35  1   1
```

**Figure 7** Key driven from a password which has got a single byte difference with the last password

**Test Case 7:** A small password is used to test the strength of the key generation procedure. This comprises of consecutive alphabets.

```
Password Entered: ABCD
Resultant data Array:
4   1   1   1   1   5   4   7
3   0   2   5   1   7   1   6
4   6   4   2   3   2   0   1
5   5   2   1   6   7   0   1
0   4   1   7   0   7   2   3
6   7   6   1   1   4   2   4
73  66  17  101 34  4   0   6
51  51  55  32  133 16  5   2
```

**Figure 8** Key driven from four consecutive alphabets taken as password

**Test Case 8:** A normal 8-byte password used by the key generator algorithm to generate the key array.

```
Password Entered: COMP1009
Resultant data Array:
4   5   0   1   1   0   4   7
5   2   7   7   4   1   5   2
0   7   7   3   2   0   1   2
0   1   5   5   5   7   1   0
5   4   40  1   4   1   1   113
76  25  27  17  6   0   5   7
36  7   1   3   1   103 143 143
51  4   3   4   1   72  131 40
```

**Figure 9** Key driven from a common 8-byte password taken as input to the system

## 5.Conclusions and future work

The present algorithm has been tested on all required data that can be considered as the most trivial password for securing useful data. The procedure worked successfully in producing a result that can be used by any other symmetric key cryptographic algorithm to encrypt useful data.

The present algorithm has been thought to implement with the author's previous works of NDEA-1 [1], NDEA-2 [2] and NDEA-3 [3] which actually requires a larger key set in order to encrypt very useful information within a given amount of stipulated time. The algorithm can be also used in fields of steganography to store very useful and delicate information in a picture in a diffused manner. Due to the virtue of the theory used, in this algorithm, the incident data array can be brought back once again using a reverse process. This helps to use the algorithm as a data diffusion process too as an extra layer of security in some less secured symmetric key cryptographic algorithm.

### Acknowledgment

The author acknowledges the Department of Computer Science, St. Xavier's College (Autonomous), Kolkata and to Father Principal Dr. J. Felix Raj for giving support to do research in Network Security.

### Conflicts of Interest

The author has no conflicts of interest to declare.

### References

[1] Mukhopadhyay A, Saha S, Tagala NA, Nath A. Noise driven encryption algorithm (NDEA) version-1. International Journal of Innovative Research in Advanced Engineering. 2015; 2(12): 28-37.

[2] Nath A, Mukhopadhyay A, Saha S, Tagala NA. Noise Driven Encryption Algorithm Version-2 (NDEA – 2). IEEE international conference CSNT 2016 (pp. 5-7). IEEE.

[3] Mukhopadhyay A. Noise Driven Encryption Algorithm-3 (NDEA-3). https://gist.github.com/Aashijit. Accessed 26 May 2016.