An Efficient Binary Playfair Algorithm Using a 4×4 Playfair Key Matrix

Saswati Mukherjee, Matangini Chattopadhyay, Ayan Lahiri, Samiran Chattopadhyay

An Efficient Binary Playfair Algorithm using a 4×4 Playfair Key Matrix

Saswati Mukherjee[1], Matangini Chattopadhyay[1], Ayan Lahiri[1], Samiran Chattopadhyay[2],

[1]School of Education Technology, Jadavpur University, Kolkata, India
sash_cal@rediffmail.com,
ayanlahiri007@gmail.com,chttpdhy@yahoo.com
[2]Department of Information Technology, Jadavpur University, Kolkata, India
samiranc@it.jusl.ac.in

Abstract. Playfair cipher is a digraph cipher which is not preferred now a day for two main reasons. Firstly, it can be easily cracked if there is enough text and secondly, frequency analysis of digraph is anyway possible. This paper proposes a new solution, which encrypts / decrypts each byte by applying the Playfair on its nibbles with the help of a reduced 4×4 Key matrix. This byte by byte encryption supports any character (even multilingual character), number (of any base), symbol and any type of media file and thereby ensures flexibility. Randomness of the algorithm is achieved by rotating the key matrix randomly after encryption / decryption of each byte. Several operations are performed to support the mechanism of lightweight cryptography. The proposed method is implemented and compared with other popular ciphers on the basis of certain parameters, like Avalanche Effect, Time Complexity, and Space Requirement. The result obtained demonstrates efficiency of the proposed algorithm.

Keywords: Playfair cipher, Digraph, Encryption, Decryption, Avalanche effect.

1 Introduction

Playfair cipher was invented by Charless Wheatstone [1] in 1854 but was named after Lord Playfair who prompted the use of the cipher. In Playfair cipher, the alphabets are arranged in a 5×5 key matrix based on secret key. Though there are 26 alphabets in English language, Playfair cipher can handle only 25 alphabets. So either Q is to be discarded or any one of i/j can be used. Despite its proven efficiency the algorithm lacks on several areas. Over the years several attempts have been made to modernize the algorithm [2], [3], [4], [5], [6], [7], [8], [9], [10] to increase its acceptances by eliminating its limitations. This paper provides a new solution approach to overcome the shortcomings of the Playfair algorithm.

1.1 Algorithm

Playfair cipher is a symmetric encryption technique which uses digraph substitution [1]. This cipher encrypts alphabets based on a reference 5×5 key matrix which is formed from the given key- PASSWORD.

Table 1. Key Matrix

P	A	S	W	O
R	D	B	C	E
F	G	H	I/J	K
L	M	N	Q	T
U	V	X	Y	Z

Now two alphabets are taken at a time from the source plaintext file and then with the reference of this key matrix a new pair of cipher text are obtained. Despite of its proven efficiency, the main drawback of Playfair Cipher lies in its limitations. The limitations of the cipher are as follows.

- Only 25 alphabets of English language are supported.
- No support for numeric characters.
- Only either upper cases or lower cases are supported.
- No special characters (viz. blank space, new line, punctuations etc.) can be used.
- Unable to deal with languages other than English.
- Any type of media files cannot be encrypted.

Beside these limitations playfair is a poly alphabetic cipher. So by testing the frequency of occurrence the plain text can easily be tracked.

1.2 Related Work

The main limitation of playfair algorithm lies in its support strictly limited to 25 alphabets of English language. Over the years several attempts have been made to increase the character limit of its dataset. Some modifications increase the matrix size to enhance the character set. A 6×6 matrix supports 36 characters, which include 26 alphabets of English language and all 10 decimal numbers (0-9) [2]. But it needs more character support in order to be able to work over a large range of text file. To increase the character set the matrix size is further increased to 8×8 to allow 64 characters, which includes 26 English alphabets, 10 decimal numbers (0-9) and a selected set of 28 symbols [3]. Though these modifications increases the character set, but still they were limited. Especially 64characters are not at all sufficient in modern day encryptions. Keeping this in mind some modifications focus on the use of ASCII values in playfair. The use of 7-bit ASCII values increased the character support to 128 characters. To deal with the matrix manipulation of playfair the concept of interweaving is introduced. It actually jumbles the binary values of the ASCII codes of a set of characters. Use of multiple iteration and character substitution further increases the secu-

rity. These interweaving and iteration actually leads to lots of confusion and diffusion [4] [5]. These modifications are useful for text encryption, but for modern day encryption of multilingual character or media files, the algorithm must be modified to support binary file encryption. To achieve the desired output some adaptations are made on traditional playfair cipher. The binary values of 7-bit ASCII codes corresponds colors of ARGB color model. To utilize the maximum color limit some further calculations are made before choosing the next color of the cipher based on the key/password [6]. But still a 7-bit ASCII support limits the scope to English language only. Multilingual characters are not supported. Keeping this in mind another modification focuses on the incorporation of DNA coding in the playfair cipher. Binary 8-bit ASCII values are initially replaced by DNA bases, and then converted into amino acid groups before applying normal playfair algorithm [7]. This actually solves the problem of limited character support but due to its complicated and lengthy process it takes more time to encrypt / decrypt. These modifications actually treat the plaintext file as binary data stream and thus enrich the character set. Another major problem of the traditional playfair is the predictability of the cipher by using frequency testing of character occurrences. To overcome this a new approach was introduced which keeps track of the frequency of occurrences of each and every character in English language and replaces the every next occurrence of the character with a character of least frequency of use [10]. Though this provides a variation in case of repetitive characters but it also takes extra time as it searches for the frequency table for each and every replacement. Another efficient way to deal with the problem is to use Random Numbers [3] [8] is a popular technique. Unpredictable different random sequences are produced from Linear Feedback Shift Register by varying logic functions and taps based on key. But the use of random numbers dose not supersedes probability of breaking it on the basis of the frequency test. Some research is also done on integration of several encryption algorithms. To be precisely if suggests use of a hybrid technique blending of both classical encryption technique as well as modern techniques to provide better security [9].

1.3 Contribution

This paper provides a new solution approach to overcome the shortcomings of the Playfair algorithm. Enhanced Binary Playfair Algorithm uses a reduced 4×4 key matrix to encrypt each byte of the plaintext file. Two nibbles of each byte are applied on the reference key matrix and a new pair of nibbles are obtained, which form the cipher byte. This algorithm also uses a dynamic matrix rearrangement to incorporate randomness in playfair instead of conventional poly alphabetic block cipher technique. The proposed Binary Playfair Algorithm is a stream cipher which uses dynamic byte substitution. This algorithm also is also efficient in respect to time complexity and space complexity and power consumption as it uses simple XOR and assignment operations. For these features this lightweight encryption algorithm can be used, where resources are limited in terms of- memory, computing time, computing power, battery supply, especially in the case of encryption/decryption in mobile device.

2 The Modified Binary Playfair Cipher

The new approach to overcome the limitations of the playfair cipher, which is discussed in this paper, treats each file as a binary file and applies the playfair algorithm on each byte of the file. The nibbles of each byte are used to encrypt / decrypt with the help of a reduced 4×4 reference key matrix. A nibble consists of 4 bits having a value of range 0-15. On the other hand, the 4×4 reference key matrix also contains 16 values, in the range 0-15. So, each pair of the nibbles of a byte is replaced with a new pair of nibbles and thus new byte is obtained. This algorithm encrypts / decrypts the file byte wise and hence the reminder offset or odd length word problem doesn't arise.

2.1 Algorithm

This algorithm consists of two phases: Key Matrix Formation and the main Encryption / Decryption phase.

Password / Key Matrix formation:

Step 1: Read the key file K_f.
Step 2: XOR both nibbles of the byte and put the result value in a key-buffer.
Step 3: If the XOR value already exists in the buffer, put the next value.
Step 4: After putting the values if the buffer-size is less than 16 put the remaining values in the buffer, so that no repetition occurs.
Step 5: Put the values of the key-buffer in the key-matrix in any specific arrangement order, based on the key value.

Encryption / Decryption:

Step 1: Read the plaintext file P_f.
Step 2: Take a byte of the plaintext file to encrypt/decrypt.
Step 3: Take two nibbles of each byte as reference and apply Playfair to get two resultant nibbles.
Step 4: Combine these two nibbles to get the encrypted byte.
Step 5: Write the byte in cipher text file C_f.
Step 6: Rearrange the key matrix on the basis of plaintext value and present key matrix arrangement.
Step 7: Repeat the steps 2-6 for next byte value until the plaintext is empty.

2.2 Step by step illustration for an example string

Now to understand the algorithm more clearly, let us take a more detailed illustration of the algorithm.

1. **Create Key-Buffer:** Let us assume that the key file contains text “IT(cwe)”.

Table 2. Creating unique key buffer

Key	ASCII	HEX-Code	Nibble 1	Nibble 2	XOR	Buffer
I	73	49	4	9	D (13)	13
T	84	54	5	4	1	1
(	40	29	2	9	B (11)	11
c	99	63	6	3	5	5
w	101	65	6	5	3	3
e	119	77	7	7	0	0
)	41	30	3	0	3	4

Now, the remaining values (that are not already placed) are placed serially starting from 0. So the total key buffer is given below.

13 1 11 5 3 0 4 2 6 7 8 9 10 12 14 15

2. **Key Matrix Arrangement:** Initially key matrix can be arranged in different variations. Some of the possible options are as follows.

Table 3. Some possible key arrangements

8	0	9	1
4	12	5	13
10	2	11	3
6	14	7	15

12	11	4	3
13	10	5	2
14	9	6	1
15	8	7	0

0	1	5	6
2	4	7	12
3	8	11	13
9	10	14	15

0	1	2	3
11	12	13	4
10	15	14	5
9	8	7	6

3. **Encrypt / Decrypt:** After the arrangement of key-matrix the main encryption/ decryption is performed. Let us assume the first byte of the plaintext is K (HEX value: 4B). And the present state of Key Matrix is as follows.

Table 4. Key Matrix

8	0	9	1
5	12	5	13
10	2	11	3
6	14	7	15

The nibble values 4 & 11 of K are applied on the key matrix and the output nibbles are 5 & 10. So the output cipher is 5A which resembles Z.

4. **Key Matrix Rotation:** The size of the reference key matrix is reduced in this algorithm. So to prevent frequency based tracking this paper suggests a compulsory rotation / rearrangement of the reference key matrix.

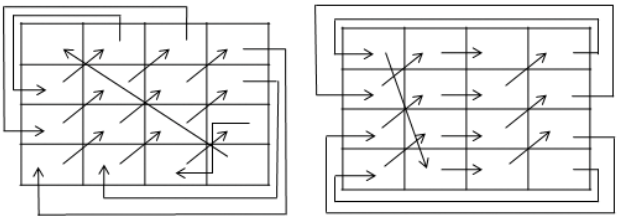

Figure 1: Some possible key re-arrangements

2.3 Illustration with an example file

To understand the algorithm more clearly, let us take an example and see its illustration. Let us assume the key / password is ➔ IT(cwe).

And the plaintext is ➔ *Switch to prudential home insurance and you could get a "25%" introductory discount for the first year of your policy*.

1. **Create Key-Buffer:** According to Table 2 of the previous example, following key buffer is formed from the password IT(cwe).

   13 1 11 5 3 0 4 2 6 7 8 9 10 12 14 15

2. **Key Matrix Arrangement:** Initially key matrix is arranged as follows.

Table 5. Key Matrix

9	12	14	15
4	6	7	8
0	3	11	2
13	1	10	5

3. **Encryption/Decryption:** The first byte of the plaintext is S (HEX value: 53). The nibble values 5 & 3 of S are applied on the key matrix and the output nibbles are 1 & 2. So the output cipher is of HEX value 12 which resembles DC2 (Device Conctol 2).

4. **Matrix Rotation:** Based on the key matrix of Table-5 and previous plaintext character S (Ascii value: 83) the matrix is rotated. So the current key-matrix is as follows.

Table 6. Key Matrix

3	9	12	14
0	4	6	15
13	11	7	8
1	10	5	2

5. **Encryption/Decryption:** The second byte of the plaintext is *w* (HEX value: 77). The nibble values 7 & 7 of *w* are applied on the key matrix and the output nibbles are 8 & 8. So the output cipher is of HEX value 88 which resembles the symbol '?'.

   These two steps of Encryption/Decryption and Matrix Rotation will be repeated for each bytes of the plaintext / ciphertext. For the plaintext of this example it will repeated for 118-times.

   And the corresponding cipher is ➔ *ı?$:-B??þ9H?#øÖB?äç®œı??}ıìRı ¼úÖı¦Ç©¦é?ı4ÔÊ?¬v?¬?ı©¤-;ª?qÑªQóEıø[ÆÛv?¦Þ£b]Æ+É~¿Öbıw¦lı œAðo tä5 ùL_ ııUâX?öÂ\'¶§¬ık*

3 Experimental Results

The proposed Enhanced Binary Playfair algorithm is implemented on java platform [11] and a number of tests are considered to observe the encryption efficiency in terms of certain parameters like avalanche effect, key randomness and time. Two more algorithms DES (Data Encryption Standard) [12] and DNA-Playfair [7] algorithms are considered for comparative encryption analysis with the proposed algorithm. In respect of the above mentioned parameters some experimental results are given below.

3.1 Avalanche Effect

The avalanche effect refers to a desirable property of cryptographic algorithms. The avalanche effect is evident if, when an input is changed slightly (for example, flipping a single bit) the output changes significantly (e.g. - 35% of the output bits flip).

In the proposed enhancement of playfair the avalanche effect is tested on a number of randomly generated 100 files. The result of Avalanche Effect lies between 46% - 55%. The ideal case or strict avalanche criterion (SAC) is the probability of 50% bits change. So the obtained result proves the efficiency of the algorithm very strongly.

Though in this algorithm the type or varity of rotation is fully dependent on ‘key’ and on the 'content of the plaintext file’, the result may vary for different files.

A details comparison of Avalanche effect between different algorithms is given in the chart below.

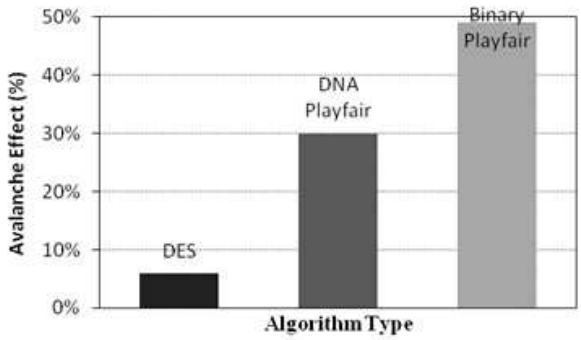

Figure 2: Avalanche Effect of different algorithms

Figure 2 shows the avalanche effect of different encryption algorithms. It can be observed that the proposed Binary Playfair algorithm gives avalanche effect of almost 49% change, which satisfies nearly the strict avalanche criteria of 50%. Thus, it can be inferred that the Enhanced Binary Playfair algorithm gives comparatively much better result than other two encryption techniques DES and DNA Playfair which show 6% and 30% avalanche effects respectively.

3.2 Repetitions of characters

The chance of repetition of a character is very low. So, it is safe from frequency tracking. The repetition a of character in the process of encryption of a 16 KB file on its every occurrence a particular byte “#” with hexadecimal value of 23 is encrypted as follows.

37	1F	6C	68	3A	50	C1	DF	0D	9D	95	BD	03	A1
C7	35	01	DB	1B	42	1C	10	84	75	B2	CE	F9	74
35	52	06	7D	37	58	D6	8D	C4	B2	1B	37	DF	BA
FC	71	8F	78	52	05	3D	E2	6B	DC	F6	30		

The above box shows that it is almost impossible to break the cipher using frequency of the character occurrence testing.

A frequency count is also computed on the cipher key matrix to analyze what are the most and least common character occurrences in the cipher. Experimental analysis for character repetition has been conducted on a large set of files of varying sizes and the

proposed algorithm has been compared with DNA Playfair and DES. Two sizes of file, 4 KB and 12 KB are chosen. The percentage values are obtained from the average of large set of small size and large size files. The results are shown as below.

Table 7. Comparison of Character Occurrences

File Size	Algorithm	Percentage of Repetition
4 KB	Binary Playfair	15 %
	DES	24%
	DNA Playfair	40%
12KB	Binary Playfair	29%
	DES	32%
	DNA Playfair	79%

It is observed in Table 7, that for small size file, the frequency tracking of the proposed algorithm is almost difficult and byte repetition is around 15% in comparison with DNA Playfair and DES of values 24% and 40% respectively. Although for large file size, the chances of byte repetition is likely to be more and hence percentage of repetition gets higher. Still Binary Playfair is proved to be more lightweight and efficient as claimed in our algorithm in comparison to DNA Playfair and DES algorithm.

3.3 Random Rotations

After encryption of each byte the matrix can be rotated in n! (i.e. 2092278988800) where n = row×column = 16) number of different ways. Among these rotations, in some cases repetitions may occur in the cipher. If row and col represents the number of rows and columns of the key matrix, and Nr represents possible number of unique rearrangements of the matrix. Then in each case of rotation/rearrangement, the actual number of variations where no repetition will occur can be expressed as follows.

$$N_r = n! - [\ ({}^{row}C_2 \times 2!) \times ({}^{col}C_2 \times 2!)\ ] \times (n-4)!\ ] \quad (1)$$

In case of this modified version of playfair a 4×4 key matrix is used. So, row=4, col=4, n = (row×col) = 16

$$N_r = 16! - [\ ({}^{4}C_2 \times 2!) \times\ ({}^{4}C_2 \times 2!)\ ] \times (16-4)! = 43536 \quad (2)$$

This 43536 is a huge number of variations, for a cryptanalyst to search in each byte of the cipher in order to break it. Hence the cipher is secured.

3.4 Low Space Requirement

The space requirement of this modified Playfair algorithm is very low. It uses a 4×4 matrix. Only a buffer of 16 nibble is required. In addition, two bytes are required one for file read/write buffer and the other for matrix manipulation buffer.

3.5 Low Time Requirement

The time requirement for encryption of this modified Playfair algorithm is substantially low. The proposed algorithm has been applied randomly on a set of 100 files of different in order to calculate the average time requirement for encryption / decryption.

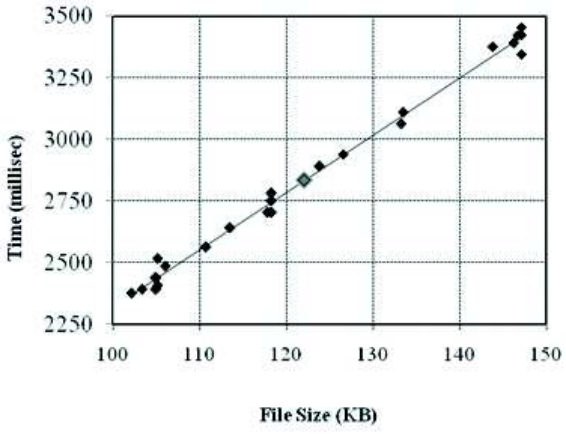

Figure 3: Average Encryption / Decryption Time

In Figure 3, the average time is depicted to encrypt 24 files of size ranging from 100 KB to 150 KB. It is observed that on an average 2 sec 832 millisecond time is required to encrypt a file of size 122 KB. As a matter of fact, it can be well inferred that average time required to encrypt increased number of large size files will be more.

A detailed comparison of different algorithms regarding time requirement to encrypt / decrypt a file of size 103 KB is given in the chart below.

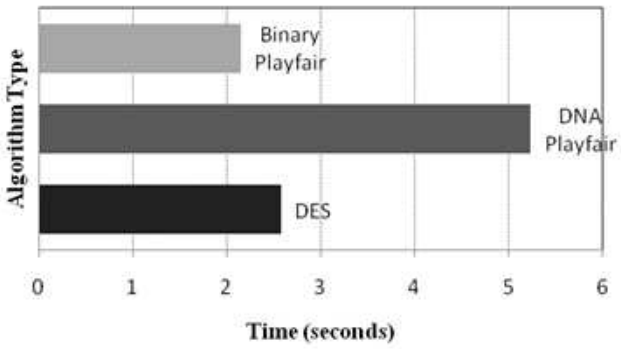

Figure 4: Time requirement versus different encryption algorithms

Figure 4 depict the average time taken by different algorithms to encrypt files of size 100 KB. A large number of randomly generated text and binary files of size 100 KB was used in the experimentation. It can be clearly observed that the proposed Enhanced Binary Playfair algorithm is taking least time of 2.150 seconds to encrypt the file. For other algorithms like DES and DNA Playfair, the time requirement for encrypting the same file are more, which are around 2.580 seconds and 5.234 seconds respectively.

4 Conclusion And Future Work

The original Playfair cipher uses a digraph substitution technique to encrypt/decrypt alphabets based on a reference 5×5 key matrix which is formed from the given key. The algorithm is strictly restricted to "English Alphabet", that to either in uppercase or in lowercase character. No numbers, punctuations and other characters are supported. Several modification attempts have focused on elimination of several limitations. Some methods have increased the character-set of the key matrix, others have used the ASCII values and others have incorporated randomness. But these modifications stand strong in their own purposes. The overall limitations of the cipher were not eliminated by these individual modifications.

This paper proposes a modification of the Playfair algorithm which strongly increases the security of the cipher. The proposed algorithm uses the 8-bit ASCII values of the plaintext file to encrypt/decrypt in the binary mode. Thus it supports any type of plaintext be it any alphabet of any language, any symbol, any number system, any type of media file or anything else. This algorithm acts as a stream cipher rather than conventional poly alphabetic block cipher. There is no need to adjust reminder offset or odd length word. Randomness is incorporated by means of random specific rearrangement of key matrix. The algorithm uses a 4×4 key matrix. So space efficiency is achieved. It uses simple XOR, shift and assignment operations straightway. The time complexity of the algorithm is O (n), where n is the file size. For these features, this

lightweight encryption algorithm can be used, where security requirement is high but resources are limited in terms of memory, computing time, computing power, battery supply.

The future work can focus on a larger key matrix which might enhance the security further and also reduce the time complexity. An additional indexing can be introduced to sort out the order and total number of rearrangement techniques based on the key / password. This algorithm can further be implemented in chip level to embed it in mobile sensors networks.

Acknowledgement

Authors acknowledge UPE-II program of Jadavpur University for partially supporting this work.

References

1. Playfair Cipher, http://en.wikipedia.org/wiki/Playfair_cipher, 21/10/11.
2. Babu, Ravindra. Uday Kumar, S. Vinay Babu, A. Aditya I. V. N. S. Komuraiah, P.: An Extension to Traditional Playfair Cryptographic Method, International Journal of Computer Application. Vol. 17, No. 5, March 2011, pp. 34-37
3. Srivastava, Shiv Shakti. Gupta, Nitin.: Security aspects of the Extended Playfair cipher, International Conference on Communication Systems and Network Technologies, 2011, pp. 144-147
4. Umakanta Sastry, V. Shankar N. Ravi, Durga Bhabani, S.: A Modified Playfair Cipher Involving Interweaving and Iteration, International Journal of Computer Theory and Engineering, Vol. 1, No. 5, December 2009, pp.597-601
5. Umakanta Sastry, V. Ravi Shankar N. Durga Bhabani, S.: A Modified Playfair for a Large Block of Plaintext, International Journal of Computer Theory and Engineering, Vol. 1, No. 5, December 2009, pp. 592-596
6. Kallam, Ravindra Babu. Udaya Kumar S. Reddy, M. Thirupathi .: A Block Cipher Generation Using Color Substitution, 2010 International Journal of Computer Applications (0975 - 8887), Vol.1 – No. 28, 2010, pp.25-27
7. Sabry, Mona. Hashem, Mohamed. Nazmy, Taymoor. Khalifa, Mohamed Essam.: A DNA and Amino Acids-Based Implementation of Playfair Cipher, (IJCSIS) International Journal of Computer Science and Information Security, Vol. 8, No. 3, 2010, pp. 129-136
8. Murali, Packirisamy. Kumar, Gandhidoss Senthil.: Modified Version of Playfair Cipher using Linear Feedback Shift Register, 2009 International Conference on Information Management and Engineering, 2009, pp. 488-490
9. Saeed, Fauzan. Rashid, Mustafa. Integrating Classical Encryption with Modern Technique, IJCSNS International Journal of Computer 280 Science and Network Security, Vol.10 No.5, May 2010, pp. 280-285
10. Mondal, Uttam Kr. Mandal, Satyendra Nath Pal Choudhury, J. : A Framework for the Development Playfair Cipher Considering Probability of Occurrence of Characters in English Literature, 2009 International Journal of Computer Science and Network Security, Vol. 8, No. 8, August 2008.
11. Bishop, David.: Introduction to cryptography with Java applets, 2005.
12. Stallings W.: Cryptography and network security: principles and practice, Fifth Edition, Prentice Hall International, 2010.