International Journal of Computer Applications (0975 – 8887)
Volume 107 – No 12, December 2014

A Stream Cipher based Bit-Level Symmetric Key Cryptographic Technique using Chen Prime Number

Sarbajit Manna
Assistant Professor
Ramakrishna Mission Vidyamandira
Belur Math, Howrah - 711202

Saurabh Dutta
Professor
Dr. B.C. Roy Engineering College
Jemua Road, Fuljhore, Durgapur - 713206

ABSTRACT
In this paper, a stream cipher based symmetric key cryptographic technique based on Chen prime number has been proposed. The proposed technique is suitable for encryption and decryption of a large number of files of almost any type. A symmetric key is formed by the sender directly from the plain text using Chen prime number. Then a symmetric key value is derived from the symmetric key which is used to form the cipher text. The symmetric key is sent to the receiver by the sender. The symmetric key value is derived by the receiver from the symmetric key which is used for decryption. To enhance security, focus is to secure the symmetric key value rather than the symmetric key.

Keywords
Chen Prime Number, Stream Cipher, Symmetric Key, Plain Text, Cipher Text.

1. INTRODUCTION
Now-a-days the Internet has become an essential mode of communication and is being increasingly used day by day. So, the information security becomes an important issue to deal with. Some algorithms are based on symmetric key, some are based on asymmetric key, but the focus is to secure the information which is the essence of communication [7]-[9].

In this paper, a new symmetric key algorithm based on Chen prime number has been proposed, where the plain text is considered as a stream of binary bits. During encryption process, a symmetric key value is generated from the symmetric key, which is used to form the cipher text. During decryption process, the plain text can be re-generated from the cipher text using the symmetric key value. The algorithm consists of following four major components [2], [6].

1.1 Symmetric Key Generation
Plain Text ⟶ Symmetric Key

1.2 Symmetric Key Value Generation
Symmetric Key ⟶ Symmetric Key Value

1.3 Encryption Mechanism
XOR

Plain Text ⟷ Symmetric Key Value ⟶ Cipher Text

1.4 Decryption Mechanism
XOR

Cipher Text ⟷ Symmetric Key Value ⟶ Plain Text

The proposed algorithm is presented in section 2. One sample implementation is shown in section 3, whereas the experimental result and analysis are shown in section 4. Finally, section 5 draws the conclusion.

2. PROPOSED ALGORITHM
A prime number p is called a Chen prime if p + 2 is either a prime or a product of two primes. 31 is a Chen prime number because 31+2=33 is a product of two prime numbers 3 and 11. 17 is a Chen prime number because 17+2=19 is a prime number. 43 is not a Chen prime number because 43+2=45 is neither a prime number nor a product of two prime numbers [6].

Section 2.1 and section 2.2 respectively describe encryption and decryption mechanisms.

2.1 Encryption Mechanism
The encryption mechanism consists of the following five steps.

Binary stream of bits formation of the input file
↓
Symmetric Key generation from the Plain Text (PT)
↓
Symmetric Key value generation from the key
↓
Binary representation of the Symmetric Key value
↓
Cipher Text (CT) generation

Steps are described in section 2.1.1 to section 2.1.5.

2.1.1 Binary Stream of Bits Formation of the Input File
The input file is read one character at a time. Each character is converted into 8-bit binary representation according to its ASCII value.

Let, the character from the input file is ‘d’. The binary stream of bits for ‘d’ (ASCII value 100) will be 01100100 which is represented in an array PT of 8-bits.

2.1.2 Symmetric Key Generation from the Plain Text(PT)
The size of the symmetric key is 24 bits having 4 blocks of size 8 bits, 7 bits, 1 bit and 8 bits respectively. The 1st block represents sum of positional weights of Chen prime bit positions of the plain text (S_{CP}). The 2nd block represents sum of positional weights of remaining bit positions of the plain text (S_{RP}). The 3rd block represents the forward (0) or backward (1) direction which is derived from the expression (ABS(S_{CP}-S_{RP}))%2 where ABS and % represents absolute value and modulus operation respectively. The 4th block represents S_{RP}th / S_{CP}thChen prime term starting from the plain text depending on forward / backward direction respectively. If the derived value of the above expression is 0, then 0 (represents forward direction) is stored in 3rd block of key and the binary equivalent of S_{RP} is stored in 4th block of key. If the derived value of the above expression is 1, then 1 (represents backward direction) is stored in 3rd block of key and the binary equivalent of S_{CP} is stored in 4th block of key.

The Chen Prime bit positions (CP) of a 8-bit number are : 2, 3, 5, 7 and the Remaining bit positions (RP) are : 0, 1, 4, 6.

So, for plain text 'd', the CP positions 2,3,5,7 and RP positions are 0,1,4,6. S_{CP} = 4+0+32+0 = 36 = 00100100 in binary, it is stored in 1st block of key and S_{RP} = 0+0+0+64 = 64 = 1000000 in binary, it is stored in 2nd block of key. (ABS(S_{CP}-S_{RP}))%2 = (ABS(36-64))%2 = 0, it is stored in 3rd block of key. As 0 represents Forward Direction (FD), So, 01000000 i.e. the binary equivalent of S_{RP} is stored in 4th block of key.

2.1.3 Symmetric Key Value Generation from the Plain Text(PT)

If the value in 3rd block of the key is 0, then forward movement is taken and S_{RP}thChen prime starting from PT is calculated. If the value in 3rd block of the key is 1, then backward movement is taken and S_{CP}thChen prime starting from PT is calculated. If the S_{CP}thChen prime starting from PT in backward direction is negative, then the absolute value is taken.

So, for plain text 'd', as the value in 3rd block of the key is 0, so forward movement is taken and 64th (S_{RP}th) Chen prime starting from 100 (ASCII value of 'd') is calculated. It is 659.

2.1.4 Binary Representation of Symmetric Key value

The symmetric key value is represented in binary form. If the number of bits in the binary form is not a multiple of 8, then it is made multiple of 8 by inserting the remaining number of 0's to the left side of MSB.

So, for plain text 'd', 659_{10} is represented in equivalent binary form. The total number of bits in binary representation of 659 is 10, which is not a multiple of 8. It is made multiple of 8 by inserting the remaining number of 0's (six) to the left side of MSB. 659_{10} = 1010010011_2 = 0000001010010011_2 (By inserting six 0's to the left of MSB to make it a multiple of 8).

2.1.5 Cipher Text (CT) Generation

Cipher text is generated by successively XORing the 8-bit blocks starting from MSB of the binary form of the symmetric key value and finally with the 8-bit binary representation of the plain text.

So, for plain text 'd', Cipher Text(CT) is generated by the following procedure where $\oplus$ represents XOR operation.

Symmetric Key Value =0000001010010011_2

00000010 $\oplus$ 10010011

10010001

PT_d(01100100) $\oplus$ 10010001

CT = 11110101

So, CT = 11110101_2 = 245_{10} = õ, CT for PT_d is õ.

2.2 Decryption Mechanism

The decryption mechanism consists of the following three steps.

Conversion of Cipher Text (CT) in binary form

Formation of symmetric key value from symmetric key

Plain Text (PT) generation

Steps are described in section 2.2.1 to section 2.2.3.

2.2.1 Conversion of Cipher Text (CT) in Binary form

The entire encrypted file is read by the receiver and each character is converted into 8-bit binary form from their ASCII values.

So, ASCII value of cipher text 'õ' is 245_{10} = 11110101_2 in binary.

2.2.2 Formation of Symmetric Key Value from Symmetric Key

As the symmetric key is known to the receiver, so the symmetric key value is generated by the receiver from the symmetric key. The symmetric key value is then represented in binary form. If the total number of bits in the binary representation of the symmetric key is not a multiple of 8, then it is made multiple of 8 by inserting remaining number of 0's to the left of MSB. Otherwise it is kept as it is.

For cipher text 'õ', the receiver generates the symmetric key value 659_{10} and it is represented in binary form (1010010011_2). The total number of bits in 1010010011_2 is 10, it is not a multiple of 8. So to make it multiple of 8, remaining 0's(6) are inserted to the left of MSB of 1010010011_2. So the binary representation of 659_{10} becomes 0000001010010011_2.

2.2.3 Plain Text (PT) Generation

Plain text is generated by successively XORing the 8-bit blocks starting from MSB of the binary form of the symmetric key value and finally with the 8-bit binary representation of the cipher text.

So, for cipher text 'õ', Plain Text(PT) is generated by the following procedure where $\oplus$ represents XOR operation.

Symmetric Key Value =0000001010010011_2

00000010 $\oplus$ 10010011

10010001

CT_d(11110101) $\oplus$ 10010001

PT = 01100100

So, PT = 01100100_2 = 100_{10} = d. So PT_d is successfully recovered from $CT_õ$.

3. IMPLEMENTATION

Following tables contain the detailed description of the encryption and decryption process of the proposed algorithm. A plain text file of size 1 KB (test.txt) is taken for encryption. The content of the plain text file is "do". Then it is encrypted to form the encrypted file ct_test.txt. From the encrypted file, the decrypted file pt_test.txt is formed by the decryption process. It is seen that the contents of the plain text file and the decrypted file is same.

Table 1. Encryption Process

Plain Text (PT)	d	o
ASCII value of each character	100	111
Binary equivalent of ASCII value	01100100	01101111
S_{CP}	36 (00100100)	44 (00101100)
S_{RP}	64 (1000000)	67 (1000011)

Direction (ABS(S_{CP}-S_{RP}))%2	0 (Forward)	1 (Backward)
S_{CP} / S_{RP}	S_{RP} = 64 (1000000)	S_{CP} = 44 (00101100)
Key Value (KV)	659 (64th Chen prime from 100 in forward direction)	101 (44th Chen prime from 111 in backward direction) absolute value is taken
Binary equivalent of key value	1010010011 (No. of bits not a multiple of 8)	01100101
Making key value multiple of 8	Required 0000001010010011 (Six 0's inserted to the left)	Not required
Cipher text (KV ⊕ PT) in binary	00000010 ⊕ 10010011 ⊕ 01100100 = 11110101	01100101 ⊕ 01101111 = 00001010
Decimal equivalent of cipher text	245	10
Cipher Text (CT)	õ	LF

Table 2. Decryption Process

Cipher Text (CT)	õ	LF
ASCII value of each character	245	10
Binary equivalent of ASCII value	11110101	00001010
Key Value generated from key (KV)	659	101
Binary equivalent of key value	1010010011 (No. of bits not a multiple of 8)	01100101
Making key value multiple of 8	Required 0000001010010011 (Six 0's inserted to the left)	Not required
Plain text (KV ⊕ CT) in binary	00000010 ⊕ 10010011 ⊕ 11110101 = 01100100	01100101 ⊕ 00001010 = 01101111
Decimal equivalent of cipher text	100	111
Plain Text (PT)	d	O

4. RESULT AND ANALYSIS

This section presents results for files of different categories namely, .EXE, .DLL, .COM, .SYS, and .TXT. Each file category contains ten different files. Each section consists of tables with information on source file size, target file size, encryption time, chi square value with the degree of freedom and avalanche percentage. Implementation of the algorithm and different types of analysis has been done using C on a computer with Intel Pentium IV 2.40 GHz processor having 512 MB RAM.

The Pearson's chi-squared test has been performed here between the source and the encrypted files to test the non-homogeneity of the two files which means whether the observations onto encrypted files are in good agreement with a hypothetical distribution. In this case, the chi square distribution is being performed with (256-1)=255 degrees of freedom, 256 being the total number of classes of possible characters in the source as well as in the encrypted files. If the observed value of the statistic exceeds the tabulated value at a given level, the null hypothesis is rejected [1].

The "Pearson's chi-squared" or the "Goodness-of-fit chi-square" is defined by the following equation:

$$X^2 = \Sigma \{(f_0 - f_e)^2 / f_e\} \quad (1)$$

Here f_e and f_0 respectively stand for the frequency of a character in the source file and that of the same character in the corresponding encrypted file. On the basis of this formula, the chi-square values have been calculated for sample pairs of source and encrypted files [1].

The avalanche effect is a very important property of cryptographic algorithms. The avalanche effect is evident if, when an input is changed slightly, the output changes drastically. If a bit in the plain text is flipped or changed, then for an evident avalanche effect, almost half of the cipher text bits are changed in the cipher text. The small change can occur either in the plaintext or in the key so that it can cause a drastic change in the cipher text. In the proposed technique, the 3rd bit of each plaintext is flipped. The key for each plaintext changes automatically with the flipping of bits in the plaintext. The avalanche percentage for each file is shown in following tables [3]-[5].

Section 4.1, section 4.2, section 4.3, section 4.4 and section 4.5 respectively describe the result for .EXE, .DLL, .COM, .SYS, .TXT files.

4.1 Result for .EXE files

Table 3 presents the result for ten .EXE files. The file sizes are in the range of 5632 bytes to 21811 bytes. Encrypted file size as well as decrypted file size is same as the source file. The encryption time ranges between 94.53 seconds to 397.82 seconds. The chi square value for each of the cases lies in the range of 14092 to 60098 with the degree of freedom ranging from 209 to 255. The achieved avalanche ranges from 15.37% to 40.56%.

Table 3. Result For .EXE Files

Source/ Target File Size (Byte)	Encryption Time (S)	Chi Square Value	Degree of Freedom	Avalanche Achieved (%)
12288	208.56	14092	209	15.37
11776	214.48	55873	252	31.19
11264	197.40	60098	252	40.56
8192	140.66	38470	253	27.68

5632	94.53	22748	240	24.21
7680	135.56	34705	245	29.06
18432	340.09	23091	255	32.00
15360	268.31	46243	254	27.26
21811	397.82	20561	255	30.35
12288	202.12	22154	223	15.98

Fig. 1 graphically compares the encryption time with the source file size for .EXE files. Each horizontal bar of color gray stands for the source file size in KB and the horizontal bar of color blackstands for the encryption time in second. It is observed from the figure that encryption time is directly proportional to the source file size.

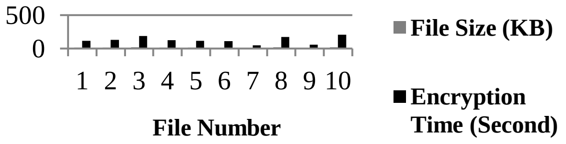

Fig. 1: A comparison between source file size with encryption time for .EXE files

4.2 Result for .DLL files

Table 4 presents the result for ten .DLL files. The file sizes are in the range of 10752 bytes to 34816 bytes. Encrypted file size as well as decrypted file size is same as the source file. The encryption time ranges between 111.39 seconds to 522.45 seconds. The chi square value for each of the cases lies in the range of 15708 to 86039 with the degree of freedom ranging from 187 to 255. The achieved avalanche ranges from 20.55% to 31.61%.

Table 4. Result For .DLL Files

Source/ Target File Size (Byte)	Encryption Time (S)	Chi Square Value	Degree of Freedom	Avalanche Achieved (%)
17408	274.84	27692	255	30.22
17408	111.39	30629	187	20.55
13312	187.68	45919	253	23.31
30208	464.07	55796	255	31.61
34816	522.45	15708	255	28.53
11264	162.30	29011	253	28.80
16896	235.69	86039	248	29.80
10752	153.62	62629	253	27.23
15872	237.33	47901	255	30.07
13312	193.67	62594	255	29.08

Fig. 2 graphically compares the encryption time with the source file size for .DLL files. Each horizontal bar of color gray stands for the source file size in KB and the horizontal bar of color black stands for the encryption time in second. It is observed from the figure that encryption time is directly proportional to the source file size.

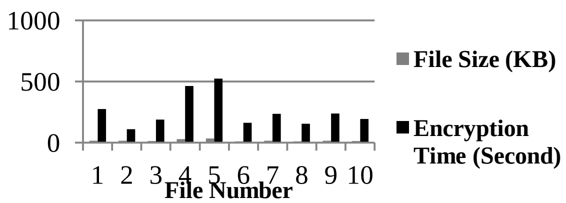

Fig. 2: A comparison between source file size with encryption time for .DLL files

4.3 Result for .COM files

Table 5 presents the result for ten .COM files. The file sizes are in the range of 1126bytes to 69836 bytes. Encrypted file size as well as decrypted file size is same as the source file. The encryption time ranges between 16.86 seconds to 1276.25 seconds. The chi square value for each of the cases lies in the range of 976 to 51245 with the degree of freedom ranging from 186 to 255. The achieved avalanche ranges from 25.31% to 34.71%.

Table 5. Result For .COM Files

Source/ Target File Size (Byte)	Encryption Time (S)	Chi Square Value	Degree of Freedom	Avalanche Achieved (%)
7680	134.41	27141	250	27.73
9216	159.12	29518	251	30.21
7168	122.10	49458	243	28.04
69836	1276.25	15219	255	30.01
29696	565.52	11644	255	34.71
26112	491.09	35335	255	28.47
19660	352.90	51245	254	28.00
7014	126.05	32029	255	30.57
14643	260.90	32093	255	28.49
1126	16.86	976	186	25.31

Fig. 3 graphically compares the encryption time with the source file size for .COM files. Each horizontal bar of color gray stands for the source file size in KB and the horizontal bar of color black stands for the encryption time in second. It is observed from the figure that encryption time is directly proportional to the source file size.

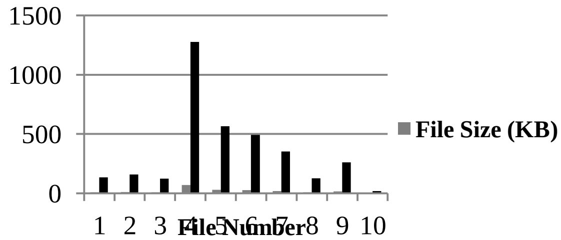

Fig. 3: A comparison between source file size with encryption time for .COM files

4.4 Result for .SYS files

Table 6 presents the result for ten .SYS files. The file sizes are in the range of 4218bytes to 17612 bytes. Encrypted file size as well as decrypted file size is same as the source file. The encryption time ranges between 57.61 seconds to 257.21 seconds. The chi square value for each of the cases lies in the range of 9837 to 70763 with the degree of freedom ranging from 229 to 255. The achieved avalanche ranges from 20.98% to 29.65%.

Table 6. Result For .SYS Files

Source/ Target File Size (Byte)	Encryption Time (S)	Chi Square Value	Degree of Freedom	Avalanche Achieved (%)
10240	147.80	47769	254	28.01
11264	161.92	47556	252	26.92
15872	237.17	70088	255	28.67
10240	155.66	36554	252	27.85
10240	147.31	56506	252	28.09

9021	132.70	48945	252	28.87
4218	57.61	9837	229	24.20
16384	232.22	44655	252	20.98
4761	71.95	37768	249	29.65
17612	257.21	70763	255	28.69

Fig. 4 graphically compares the encryption time with the source file size for .SYS files. Each horizontal bar of color gray stands for the source file size in KB and the horizontal bar of color black stands for the encryption time in second. It is observed from the figure that encryption time is directly proportional to the source file size.

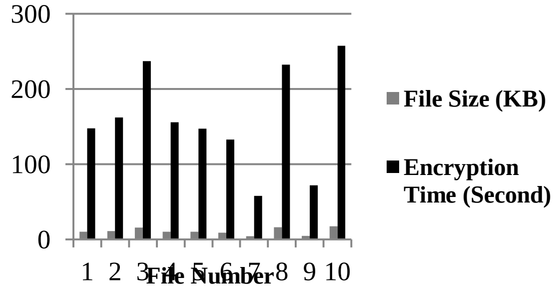

Fig. 4: A comparison between source file size with encryption time for .SYS files

4.5 Result for .TXT files

Table 7 presents the result for ten .TXT files. The file sizes are in the range of 133 bytes to 32563 bytes. Encrypted file size as well as decrypted file size is same as the source file. The encryption time ranges between 0.77 seconds to 155.60 seconds. The chi square value for each of the cases lies in the range of -161 to 16105 with the degree of freedom ranging from 53 to 134. The achieved avalanche ranges from 23.79% to 33.47%.

Table 7. Result For .TXT Files

Source/ Target File Size (Byte)	Encryption Time (S)	Chi Square Value	Degree of Freedom	Avalanche Achieved (%)
1331	7.42	3994	101	33.47
5416	31.91	10536	114	29.87
6451	35.86	16105	108	25.16
133	0.77	253	53	29.89
829	4.34	1032	88	32.70
22937	115.34	6958	121	26.82
2201	11.70	4099	104	28.64
2498	13.52	5120	103	31.40
32563	155.60	-161	94	23.79
19046	101.50	10177	134	30.62

Fig. 5 graphically compares the encryption time with the source file size for .TXT files. Each horizontal bar of color gray stands for the source file size in KB and the horizontal bar of color black stands for the encryption time in second. It is observed from the figure that encryption time is directly proportional to the source file size.

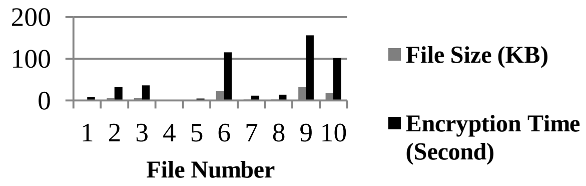

Fig. 5: A comparison between source file size with encryption time for .TXT files

5. CONCLUSION

The proposed symmetric-key algorithm has been developed keeping security factor in mind. In the proposed algorithm, the generated symmetric key is totally based on the plain text and Chen prime number. As the plain text changes, the symmetric key also changes automatically. The symmetric key value generated from the symmetric key gives an extra edge in security for the algorithm. The key space is directly proportional to the source file size. The proposed technique presented in this paper is simple to implement using any popular high level language. The size of the encrypted file and decrypted file is same as the source file size. So any additional memory is not needed to store the encrypted and decrypted file. The execution time is dependent on the source file size, not on the type of the file as the encryption and decryption has been done in bit level. For most of the files, the achieved avalanche percentage is between 25 and 35. The avalanche percentage will be better if more than one bit is flipped. The avalanche effect can also be calculated not only flipping one or more bits of the plain text, but also flipping any number of bits of the symmetric key or even both the plain text and the symmetric key. Not only XOR operation but also any one or more reversible logical operations can be applied on the plaintext to get the cipher text.

6. REFERENCES

[1] Saurabh Dutta, “An Approach towards Development of Efficient Encryption Technique”, Ph.D. Thesis, The University of North Bengal, 2004.

[2] Ramkrishna Das, Saurabh Dutta, “An Approach of Bit-level Private-key Encryption Scheme based on Armstrong Number and Perfect Number in Selective Mode”, Bulletin & Engineering Science (BES), ISSN : 0974- 7176 Vol. 5 No. 1.

[3] http://en.wikipedia.org/wiki/Avalanche_effect

[4] SriramRamanujam and MarimuthuKaruppiah “Designing an algorithm with high Avalanche Effect” IJCSNS International Journal of Computer Science and Network Security, VOL.11 No.1, January 2011.

[5] Manas Paul, Tanmay Bhattacharya, Suvajit Pal, RanitSaha, “A Novel Generic Session Based Bit Level Encryption Technique to Enhance Information Security”, (IJCSIS) International Journal of Computer Science and Information Security, Vol. 3, No. 1, 2009.

[6] http://en.m.wikipedia.org/wiki/Chen_prime

[7] Mark Nelson, Jean-Loup Gailly, The Data Compression Book. BPB Publication

[8] Atul Kahate, Cryptography and Network security, Tata McGraw Hill Publishing Company Limited.

[9] William Stallings, Cryptography and Network security: Principles and Practice (Second Edition), Pearson Education Asia, Sixth Indian Reprint 2002.