www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 6 Issue 6 June 2017, Page No. 21650-21656
Index Copernicus value (2015): 58.10 DOI: 10.18535/ijecs/v6i6.17

Symmetric Stream Cipher Based On Chebyshev Polynomial

D. Sravana Kumar[1], P. Sirisha[2], CH.Suneetha[3]
[1]Associate Professor in Physics, Dr.V.S.Krishna Government Degree,
Visakhapatnam
skdharanikota@gmail.com
[2]Faculty in Mathematics, Indian Maritime University
Visakhapatnam
sirinivas06@gmail.com
[3]Associate Professor in Mathematics, GITAM University College
Visakhapatnam
gurukripachs@gmail.com

Abstract: ***The rapid development of information technology turned internet as the basic means and wide choice for communications. Due to extensive adoption of internet for communications it is essential these days to conceal the message from unintended reader. The present paper describes a new encryption algorithm using Chebyshev polynomial of I kind. The plain text is encrypted in three rounds each round consisting of two stages with the concatenation of the previous cipher character with different keys. For making the algorithm more secure the key for the first round of encryption is generated from the main key (agreed upon by the sender and the receiver) and the subsequent round keys are concatenated with the previous round keys. The stream cipher proposed here has several advantages over conventional cryptosystems.***

Keywords: Chebyshev Polynomial, Encryption, Decryption, Concatenation

Introduction: Secure transmission of the sensitive information to the intended recipient with confidentiality is the essence of cryptography. Message encryption is one way of achieving the confidentiality [10]. Performing modular arithmetic operations and bit-wise logical XOR operations repeatedly strengthens of the cipher [4]. But simple XOR encryption is vulnerable to several types of active and passive attacks. Performing logical XOR operation along with the concatenation with the previous round cipher enhances the security levels.

Concatenation:

String concatenation is an operation used in co programming and data base theory. In conve cryptosystems like AES concatenation coding in stri called "serial concatenation". M.IsmailJabiullahEt.al.[3 concatenation technique of two or more keys to creat encryption key. Andres UHL Et.al. [1]used bit concatenation process in image and video encryption. Subhas BarmanEt.al. [11]used concatenation to generate cryptographic keys for finger print based biometric system. In the present paper concatenation technique is used in designing the cipher and also in generating the schedule round key from the agreed upon main key. The message stream is encrypted in three rounds, each round consisting of two stages. In first stage each character is encrypted using Chebyshev polynomial of I kind and in second stage concatenation technique is applied to encrypt the characters except for the first character.

Chebyshev polynomial of First kind:

The Chebyshev polynomials are orthogonal polynomials defined as the solution of the Chebyshev differential equation

$$(1-x^2)y'' - xy' + n^2 y = 0$$

The Chebyshev polynomials of the first kind are defined by the recurrence relation

$$T_0(x) = 1$$
$$T_1(x) = x$$
$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$$

The first few Chebyshev polynomials of the first kind are

$T_0(x) = 1$

$T_1(x) = x$

DOI: 10.18535/ijecs/v6i6.17

$T_2(x) = 2x^2-1$

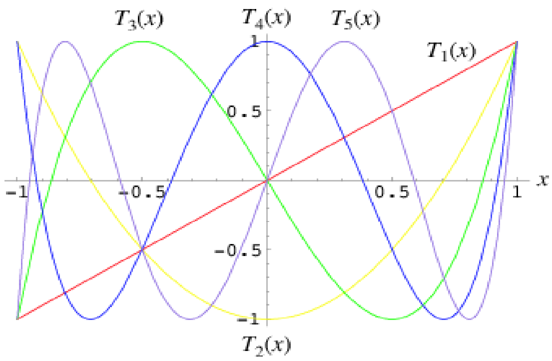
$T_3(x) = 4x^3-3x$

$T_4(x) = 8x^4-8x^2+1$

$T_5(x) = 16x^5-20x^3+5x$

$T_6(x) = 32x^6-48x^4+18x^2-1$

First few Chebyshev polynomials of I kind are shown in the following graph



Earlier work on Chebyshev Polynomial:

Chebyshev Polynomials have been recently proposed for assigning public key cryptosystems. G.J.Fee Et.al.[2] used Chebyshev Polynomial $T_n(x)$ for replacing monomial x^nin Diffie-Helman and in RSA algorithms. Chebyshev polynomial can be mapped into classical discrete log problem. Based on Chebyshev Polynomial, we get RSA algorithm. Kai-Yuen Cheong [5] used Chebyshev polynomial to compare chaotic encryption systems with one-way functions to get new insights for chaos-based cryptosystems.K. Prasadh Et.al [9] described public key encryption based on Chebyshev polynomial.

Proposed Method:

Procedure for schedule round key generation:

If two communicating parties want to communicate with each other first they agree upon to use a 8 digit decimal number to act as the secret key or main key for their communication. The 8 digit key is divided into two equal parts K_A and K_B .

$K = K_1 K_2 K_3 K_4 K_5 K_6 K_7 K_8$

$K_A = K_1 K_2 K_3 K_4$

$K_B = K_5 K_6 K_7 K_8$

The message to be communicated is encrypted in 3 rounds with different keys. The key for each round of encryption/decryption is derived from the agreed upon main key $K=K_A+K_B$ using some permutation function.

The main key $K = K_1 K_2 K_3 K_4 K_5 K_6 K_7 K_8$. The key for first round of encryption is

$K_I = K_{1I}K_{2I}K_{3I}........K_{8I}$ where

$$K_{IA} = K_{1I}K_{2I}K_{3I}K_{4I}$$
$$K_{IB} = K_{5I}K_{6I}K_{7I}K_{8I}$$

and $K_{1I} = K_1K_2, K_{2I} = K_2K_3........K_{8I} = K_8K_1$

If the products $K_1K_2, K_2K_3,........K_8K_1$ exceed 256 then modular operation of mod 256 is applied.

The key for second round of encryption is generated from the first round key using concatenation technique

$K_{II} = K_{1II}K_{2II}K_{3II}........K_{8II}$ where

$$K_{IIA} = K_{1II}K_{2II}K_{3II}K_{4II}$$
$$K_{IIB} = K_{5II}K_{6II}K_{7II}K_{8II}$$

and $K_{1(II)} = K_{1I}K_{2I}, K_{2II} = K_{2I}K_{3I},........K_{8II} = K_{8I}K_{1I}$

The key for third round of encryption is generated from the second round key using concatenation technique

$K_{III} = K_{1III}K_{2III}K_{3III}........K_{8III}$ where

$$K_{IIIA} = K_{1III}K_{2III}K_{3III}K_{4III}$$
$$K_{IIIB} = K_{5III}K_{6III}K_{7III}K_{8III}$$

and $K_{1(III)} = K_{1II}K_{2II}, K_{2III} = K_{2II}K_{3II},........K_{8III} = K_{8II}K_{1II}$

Encryption:

Suppose that the plain text stream to be communicated be M with characters $M_1M_2M_3........M_n$. All the characters $M_1M_2M_3........M_n$ are coded to equivalent binary numbers using ASCII code table. Each character of the message stream is encrypted in 3 rounds and each round consisting of two stages of encryption, the first stage using Chebyshev polynomial with the key K_A and the second stage using concatenation technique with the key K_B starting from the first round key

$K_I = K_{1I}K_{2I}K_{3I}........K_{8I}$ where

DOI: 10.18535/ijecs/v6i6.17

$$K_{IA} = K_{1I}K_{2I}K_{3I}K_{4I}$$
$$K_{IB} = K_{5I}K_{6I}K_{7I}K_{8I}$$

I Stage Encryption:

In first stage each character of the message is encrypted using Chebyshev polynomial

$$T_0(x) = 1$$
$$T_1(x) = x$$
$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$$

Compute $\left[T_1(K_{IA})\right]_{\mathrm{mod}\,256} = (K_{IA})_{\mathrm{mod}\,256}$ and the resulting number is converted to 8 bit binary number using ASCII code table. Logical XOR operation is performed between 8 bit binary equivalent numbers of M_1 and $\left[T_1(K_{IA})\right]_{\mathrm{mod}\,256}$ to get the first stage cipher character C_1^1 of the first plain text character M_1.

$$C_1^1 = M_1 XOR\left[T_1(K_{IA})\right]_{\mathrm{mod}\,256}$$

Similarly logical XOR operation is performed between the 8 bit binary equivalent numbers of the second character M_2 and $\left[T_2(K_{IA})\right]_{\mathrm{mod}\,256} = \left[2(K_{IA})^2 - 1\right]_{\mathrm{mod}\,256}$ to get the first stage cipher character C_2^1 of the plain text character M_2.

$$C_2^1 = M_2 XOR\left[T_2(K_{IA})\right]_{\mathrm{mod}\,256}$$

All the plain text characters are encrypted in first stage to get the cipher characters $C_1^1 C_2^1 C_3^1 ......C_n^1$

$$C_n^1 = M_n XOR\left[T_n(K_{IA})\right]_{\mathrm{mod}\,256}\ n =,12,3,......n$$

II stage of Encryption:

The second stage of encryption is performed using the remaining half part K_{IB} of the first round key K_I. The output of the first stage encryption is $C_1^1 C_2^1 C_3^1 ......C_n^1$. Logical XOR operation is performed between the 8 bit binary equivalent numbers of C_1^1 and $\left[(K_{IB})\right]_{\mathrm{mod}\,256}$ to get the second stage cipher character C_1^2 of the plain text character M_1

$$C_1^2 = C_1^1 XOR\left[K_{IB}\right]_{\mathrm{mod}\,256}$$

The first cipher character C_1^2 of the second stage is concatenated with the key K_{IB} and logical XOR operation is performed between the 8 bit binary equivalent numbers of C_2^1 and $\left\lceil K_{IB} * C_1^2 \right\rceil_{\mathrm{mod}\,256}$ to get the second cipher character C_2^2 of the second stage.

$$C_2^2 = C_2^1 XOR\left\lceil K_{IB} * C_1^2 \right\rceil_{\mathrm{mod}\,256}$$

[Here C_1^2 is binary number and K_{IB} is a decimal number. So, C_1^2 is converted to equivalent decimal number and then $\left\lceil K_{IB} * C_1^2 \right\rceil_{\mathrm{mod}\,256}$ is calculated]

In a similar way all the characters of the first stage encryption are second stage encrypted to get the second stage cipher characters $C_1^2 C_2^2 C_3^2 ......C_n^2$

$$C_n^2 = C_n^1 XOR[K_B * C_{n-1}^2]_{\mathrm{mod}\,256}\ n = 2,3,4......$$
$$C_n^2 = C_n^1 XOR\left[K_{IB}\right]_{\mathrm{mod}\,256}\ n = 1$$

The two stage encryption process is repeated in two more rounds with different keys K_{II}, K_{III} which are generated from the previous round key using concatenation technique. Then all the 8 bit binary numbers are coded to the text characters using ASCII code table and communicated to the receiver as the cipher text in a public channel K_{III}

Decryption:

The receive after receiving cipher text converts all the cipher characters $C_1^2 C_2^2 C_3^2 ......C_n^2$ to equivalent 8 bit binary numbers. The decryption process is done in three rounds, each round consisting of two stages using the key K_B for first stage and the key K_A for the second stage starting from the third round key

I Stage Decryption:

Logical XOR operation is performed between the 8 binary equivalent numbers of the first cipher character

$C_1^2 and \left[K_{IIIB}\right]_{\mathrm{mod}\,256}$ to get the first stage decipher character D_1^1.

$$D_1^1 = C_1^2 XOR\left[K_{IIIB}\right]_{\mathrm{mod}\,256}$$

Logical XOR operation is performed between 8 bit binary equivalents of C_2^2 and $\left[K_{IIIB} * C_1^2\right]_{\mathrm{mod}\,256}$ to get the first stage decipher character D_2^1

$D_2^1 = C_2^2 XOR\left[K_{IIIB} * C_1^2\right]_{\mathrm{mod}\,256}$ [Here also K_{IIIB} is decimal and C_1^2 is binary. So, C_1^2 is converted to equivalent binary number and $\left[K_{IIIB} * C_1^2\right]_{\mathrm{mod}\,256}$ is calculated]

$$D_n^1 = C_n^2 XOR\left[K_{IIIB} * C_{n-1}^2\right]_{\mathrm{mod}\,256} \; n = 2,3,......$$

II stage of decryption:

The second stage of decryption is performed using the first half part K_A of the key K starting from the third round key. The output decipher characters of the first stage of decryption are $D_1^1, D_2^1, D_3^1........D_n^1$. Logical XOR operation is performed between binary equivalents of the first character D_1^1 and $\left[T_1(K_{IIIA})\right]_{\mathrm{mod}\,256}$ to get the first plain text character M_1.

$$M_1 = D_1^1 XOR\left[T_1(K_{IIIA})\right]_{\mathrm{mod}\,256}$$

Logical XOR operation is performed between the second character D_2^1 and $\left[T_2(K_{IIIA})\right]_{\mathrm{mod}\,256}$ to get the second plain text character M_2

$$M_2 = D_2^1 XOR\left[T_2(K_{IIIA})\right]_{\mathrm{mod}\,256}$$

Similarly all the first stage decipher characters are decrypted to get the plain text characters $M_1, M_2, M_3........M_n$.

$$M_n = D_n^1 XOR\left[T_n(K_{IIIA})\right]_{\mathrm{mod}\,256}$$

The two stage decryption process is repeated for two more times with the keys K_{II}, K_I and all the 8 bit binary numbers are converted to text characters using ASCII code table to get the original message.

Example for one round of Encryption and Decryption:

Encryption Consider the plain text ‘mmmmmm’

Plaintext mmmmm
Key 12345678

I Stage Cipher c1 = 1 0 1 1 1 1 1 1
c2 = 1 1 1 0 1 0 1 0
c3 = 1 1 0 0 0 1 1 1
c4 = 0 0 0 0 1 1 0 0
c5 = 0 0 0 1 0 1 0 1
c6 = 0 1 1 0 1 1 0 1
II Stage Cipher h1 =1 0 0 1 0 0 0 1
h2 =1 1 1 0 0 1 0 0
h3 = 0 0 1 1 1 1 1 1
h4 = 0 1 0 1 1 1 1 0
h5 = 1 1 1 1 0 0 0 1
h6 = 0 0 1 0 0 0 1 1

Cipher text ‘ ä ? ^ ñ #

Decryption

Cipher text ‘ ä ? ^ ñ #
Key 12345678
I Stage Decipher d1 = 1 0 1 1 1 1 1 1
d2 = 1 1 1 0 1 0 1 0
d3 = 1 1 0 0 0 1 1 1
d4 = 0 0 0 0 1 1 0 0
d5 = 0 0 0 1 0 1 0 1
d6 = 0 1 1 0 1 1 0 1

II Stage Decipher m1 =0 1 1 0 1 1 0 1
m2 = 0 1 1 0 1 1 0 1
m3 = 0 1 1 0 1 1 0 1
m4 = 0 1 1 0 1 1 0 1
m5 = 0 1 1 0 1 1 0 1
m6 = 0 1 1 0 1 1 0 1

Plain Text mmmmmm

DOI: 10.18535/ijecs/v6i6.17

The two stage encryption process is repeated in three rounds using different keys as shown in the example.

Conclusions: The security of the stream cipher usually depends on the protection against correlation attacks based on linear feedback shift registers (LFSR) [6]. To resist different types of correlation attacks some authors [7] proposed Boolean functions. In the present stream cipher each character of the stream is encrypted in three rounds, each round consisting of two stages, the first stage using chebyshev polynomial of I kind with half part of the key and the second stage with the remaining part of the key. In the second stage encryption each character is concatenated with previous cipher character. The key for each round of encryption is different and is generated from the main key agreed upon by both the sender and the receiver. With the present available technology it is possible to check one million keys per second. Since in the proposed algorithm the schedule round key is different for different rounds and the round key is generated by concatenating the previous round key, brute force attacks to retrieve the key is quite impossible to execute using a one core processor.

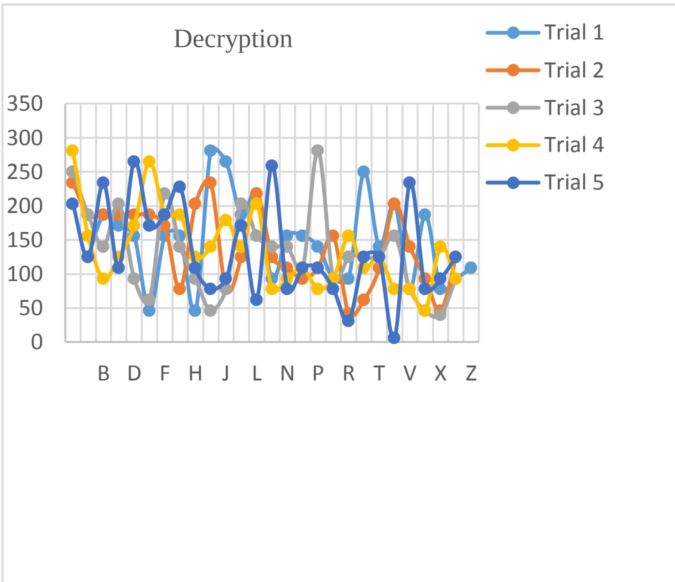
Time Analysis: The complexity of a good encryption algorithm normally depends on two properties - space complexity which attributes to the amount of memory needed to store the algorithm and the time complexity which refers to the time required to run the program. In cryptography timing attack is a side channel attack in which the intruder can break the cipher depending on the time to run the program. Timing attack enables an attacker to extract secrets in the system by observing the time of response to various queries. Kocher [8] designed a timing attack to expose the secret keys for RSA algorithm. In the present algorithm demonstrated here in this paper the time taken for encryption and decryption for different plaintext characters is different. If we consider the message a six letter word consisting of same alphabet characters from A to Z (i.e., AAAAAA, BBBBBB etc.) then the time of encryption and decryption of the same message is different for different messages and for different trials. The following graph between alphabet series and the running time (in milliseconds) for different trials shows that the timing attack is impossible to execute for the present stream cipher.

Table 1

Encryption Time for Alphabets in milli Sec	A	B	C	D	E	F	G
Trial 1	390	375	468	296	250	390	265
Trial 2	281	421	468	296	250	265	296
Trial 3	265	328	234	593	296	265	328
Trial 4	343	359	281	296	406	312	234
Trial 5	343	406	484	265	187	187	218
Encryption Time for Alphabets in milli Sec	H	I	J	K	L	M	N
Trial 1	265	390	312	265	328	265	281
Trial 2	265	250	312	296	250	187	218
Trial 3	250	312	187	281	296	343	203
Trial 4	203	328	203	265	328	296	296
Trial 5	187	312	437	234	578	187	375
Encryption Time for Alphabets in milli Sec	O	P	Q	R	S	T	U
Trial 1	250	343	250	218	328	969	328
Trial 2	390	375	234	234	265	296	250
Trial 3	359	281	359	203	187	218	250
Trial 4	218	250	218	234	296	281	203
Trial 5	234	218	250	375	328	296	187
Encryption Time for Alphabets in milli Sec	V	W	X	Y	Z		
Trial 1	328	250	218	296	296		
Trial 2	218	234	234	250	187		
Trial 3	234	281	281	171	250		
Trial 4	203	265	265	265	265		
Trial 5	203	171	265	265	265		

Table 2

Decryption Time for Alphabets in milli Sec	A	B	C	D	E	F	G
Trial 1	125	187	171	156	46	156	156
Trial 2	234	187	187	187	187	187	171
Trial 3	250	187	140	203	93	62	218
Trial 4	281	156	93	125	171	265	187
Trial 5	203	125	234	109	265	171	187
Decryption Time for Alphabets in milli Sec	H	I	J	K	L	M	N
Trial 1	46	281	265	187	218	93	156
Trial 2	78	203	234	78	125	218	125
Trial 3	140	93	46	78	203	156	140
Trial 4	187	125	140	179	140	203	78
Trial 5	328	109	78	93	171	62	359
Decryption Time for Alphabets in milli Sec	O	P	Q	R	S	T	U
Trial 1	156	140	93	93	250	140	203
Trial 2	109	93	109	156	46	62	109
Trial 3	140	109	281	093	125	125	125
Trial 4	93	109	78	93	156	109	125
Trial 5	78	109	109	78	31	125	125
Decryption Time for Alphabets in milli Sec	V	W	X	Y	Z		
Trial 1	78	187	78	93	109		
Trial 2	203	140	93	46	125		
Trial 3	156	78	62	125	93		
Trial 4	78	78	46	140	93		
Trial 5	62	234	78	93	125		



When the key is symmetric or private the security of the key should be managed. In the present algorithm the schedule round key for each round of encryption/decryption is changed which is generated from the main secret key (agreed upon key) by concatenating the previous round key. Even though a part of the key t a particular round is leaked the subsequent round keys remain secured because the key is changing from time to time for each round. This enhances the safeguard of the key because the intruder cannot perform exhaust key search.

References:

1. Andres UHL, Andreas Pommer "Image and video encryptionfrom digital rights management to secured personal communication" Advances in information technology, Springer 2005

2. Fee G. J. and MonaganM. B., "Cryptography using Chebyshev polynomials", www.cecm.sfu.ca/CAG/papers/Cheb.pdf

3. Ismail JabiullahM., ZakariaSarkerMd., AnisurRahmanand M. Lutfar Rahman, "A secured message transaction approach by dynamic hill cipher generation and message digest concatenation" , Daffodil International university journal of science and technology, Volume 5, Issue 1, January 2010

4. Islam R., "Enhanced Security in Mobile IP Communication", Masters Thesis, Department of Computer and System Sciences, Stockholm University, Royal Institute of Technology, February, 2005

5. Kay-Yung Cheong, "One-way functions from Chebyshevpolynomials "May 2012, https://eprint.iacr.org/2012/263.pdf

6. Nicholas Courtois and Willi Meier "Algebraic attacks on stream ciphers with Linear Feedback http://eprint.iacr.org/2002/087

7. Palash Sarkar, SubhamoyMaitra, "Construction of non linear Boolean functions with important cryptographic protocols", Advances in Cryptology EUROCRYPT 2000

8. Paul Kocher, "Timing attacks on implementations of Diffie-hellman, RSA, DSS,and other systems", Advances in Cryptology , pages 104–113, 1996.

9. Prasadh K., Ramar K. and Gnanajeyaraman R., "Public key cryptosystem based on chaotic Chebyshev polynomial", Journal of Engineering and Technology Research Vol.1 (7), pp. 122-128, October, 2009

10. Stallings W., "Cryptography and Network Security – Principles and Practices",3rd Edition, 4th Indian Reprint, 2004, ISBN: 81-7808-902-5.11

11. Subhas Barman, DebasisSamantaandSamiran Chattopadh**yay, "**Fingerprint-based crypto-biometric system for network security" , EURASIP Journal on information security**,** A Springer open journal 2015:3(2015) 2015:3