Journal of Advances in Mathematics and Computer Science

Volume 37, Issue 12, Page 29-40, 2022; Article no.JAMCS.94337

ISSN: 2456-9968

(Past name: British Journal of Mathematics & Computer Science, Past ISSN: 2231-0851)

Encryption with Generalised Balancing Sequences and Generalised Lucas Balancing Sequences

P. Anuradha Kameswari [a*] and K. Anoosha [a]

[a] *Department of Mathematics, Andhra University, India.*

Authors' contributions

This work was carried out in collaboration between both authors. Both authors read and approved the final manuscript.

Article Information

DOI: 10.9734/JAMCS/2022/v37i121726

Open Peer Review History:
This journal follows the Advanced Open Peer Review policy. Identity of the Reviewers, Editor(s) and additional Reviewers, peer review comments, different versions of the manuscript, comments of the editors, etc are available here: https://www.sdiarticle5.com/review-history/94337

Received: 12/10/2022
Accepted: 08/12/2022
Published: 21/12/2022

Original Research Article

Abstract

In this paper we propose a key stream for encryptions obtained by concatenation of solutions of Diophantine equations which are generated by using some properties of Generalised Balancing Sequences.

Keywords: Encryption; key stream; Diophantine equation; Balancing Sequences.

2010 Mathematics Subject Classification: 94A60, 11T71.

1 Introduction

Cryptosystems are of two types, namely Symmetric Key Cryptosystem (SKC) and Public Key Cryptosystem (PKC). For enciphering and deciphering messages a SKC uses same key or one can easily obtain the deciphering key with the knowledge of enciphering key. This makes maintaining the secrecy of the key difficult which is the main disadvantage in SKC. In order to avoid this PKC has been introduced. PKC is the cryptosystem with different enciphering and deciphering keys. The main drawback of PKC is that it is not efficient in

**Corresponding author: E-mail: panuradhakameswari@yahoo.in;*

communicating large messages. So to construct a crysptosystem that is strong, efficient and devoid of the above mentioned drawbacks both PKC and SKC are employed, where PKC is used for key exchange and SKC for communication of large messages.

Diophantine Equations with infinitely many soluitons play a vital role in cryptography. In [1, 2, 3] we described the solutions of Diophantine Equations $x^2 \pm pxy + y^2 \pm x = 0$ and $x^2 \pm pxy + y^2 \pm (\frac{p^2-4}{4})x = 0$ that are generated from Generalised Balancing Sequences and Generalised Lucas Balancing Sequences respectively. In [4] P.Anuradha Kameswari and B.Ravi Theja came forward with algorithms for fast computation of Lucas sequences [5, 6]. In this paper we extend these algorithms to Generalised Balancing Sequences. Then a key stream for encryption is generated by concatenation of solutions of the Diophantine equation $x^2-pxy+y^2-x = 0$, for encryption.

In this paper, we first discuss some preliminaries on Generalised Balancing Sequences and Diophantine Equation $x^2 - pxy + y^2 - x = 0$ whose solutions expressed in terms of Generalised Balancing Sequences.and algorithms of fast computation to calculate these terms are discussed. We extend the algorithms discussed in [4] for all computations of Generalised Balancing Sequences in section 3. In section 4 we generate a key stream which is a concatenation of solutions of Diophantine Equations. In section 5 and 6 the efficiency and cryptanalysis of the cryptosystem are discussed respectively.

2 Preliminaries

2.1 Generalised balancing numbers, their properties and associated diophantine equation

Balancing Numbers B_n are defined by Behera and Panda [7] as the natural numbers that satisfy the recurrence relations $B_{n+1} = 6B_n - B_{n-1}$ with $B_0 = 0$ and $B_1 = 1$. In [2], Generalised Balancing Sequences are introduced as the numbers satisfying the recurrent relation $B_{n+1} = pB_n - B_{n-1}$ with $B_0 = 0$, $B_1 = 1$ where p is a positive integer. It also discusses the following properties of Generalised Balancing Sequences and Diophantine Equation $x^2 - pxy + y^2 - x = 0$ whose solutions are in terms of Generalised Balancing Sequences.

Notation 2.1. B_n,the nth term of Generalised Balancing Sequence, from now on is denoted by $B_n(p)$ i.e

$$B_{n+1}(p) = pB_n(p) - B_{n-1}(p)$$

Theorem 2.2. For $Q_{B_p}=\begin{bmatrix} p & -1 \\ 1 & 0 \end{bmatrix}$, $Q^n_{B_p} = \begin{bmatrix} B_{n+1}(p) & -B_n(p) \\ B_n(p) & -B_{n-1}(p) \end{bmatrix}$, $n \geq 1$.

Theorem 2.3. $B_{-n}(p)=-B_n(p)$ for all $n \geq 1$.

Theorem 2.4. $pB_n(p)B_{n+1}(p) - B_n(p)^2 + 1 = B_{n+1}(p)^2$ $\forall$ integers n.

Theorem 2.5.
$$\begin{aligned} B_{m+n}(p) &= -B_{m+1}(p)B_n(p) + B_m(p)B_{n-1}(p) \\ &= B_m(p)B_{n+1}(p) - B_{m-1}(p)B_n(p) \\ B_{2m}(p) &= B_m(p)B_{m+1}(p) - B_{m-1}(p)B_m(p) \\ B_{2m-1}(p) &= B_m(p)^2 - B_{m-1}(p)^2 \\ B_{2m+1}(p) &= B_{m+1}(p)^2 - B_m(p)^2 \ \forall \text{ integers } m \text{ and } n. \end{aligned}$$

Theorem 2.6. *For a positive integer p, all the solutions of $x^2 - pxy + y^2 - x = 0$ are*
1) $\left(B_{2n}(p)^2, B_{2n-1}(p)B_{2n}(p)\right)$
2) $\left(B_{2n}(p)^2, B_{2n}(p)B_{2n+1}(p)\right)$
3) $\left(B_{2n+1}(p)^2, B_{2n}(p)B_{2n+1}(p)\right)$
4) $\left(B_{2n+1}(p)^2, B_{2n+1}(p)B_{2n+2}(p)\right)$ *for all integers $n \geq 0$.*

2.2 Generalised Lucas balancing numbers, their properties and associated Diophantine equation

Lucas Balancing Numbers C_n are defined by Behera and Panda [7] as the natural numbers that satisfy the recurrence relations $C_{n+1} = 6C_n - C_{n-1}$ with with $C_0 = 1$ and $C_1 = 3$. In [1], Generalised Balancing Sequences are introduced as the numbers satisfying the recurrent relation $C_{n+1} = pC_n - C_{n-1}$ with $C_0 = 1$, $C_1 = p/2$ where p is a positive integer. It also discusses the following properties of Generalised Lucas Balancing Sequences.

Notation 2.7. C_n,the nth term of Generalised Balancing Sequence, from now on is denoted by $C_n(p)$ i.e

$$C_{n+1}(p) = pC_n(p) - C_{n-1}(p)$$

Theorem 2.8. $C_{-n}(p) = C_n(p)$ for all $n \geq 1$.

Theorem 2.9. $C_{m+n}(p) = C_m(p)C_n(p)+(\frac{p^2-4}{4})B_m(p)B_n(p)$ and $C_{m-n}(p) = C_m(p)C_n(p)-(\frac{p^2-4}{4})B_m(p)B_n(p)$ for all integers m, n.

Remark 2.10. We observe that $C_n(p) = (\frac{p}{2})B_n(p) - B_{n-1}(p) = \frac{B_{n+1}(p)-B_{n-1}(2)}{2}$

We define Balancing R-Matrix as

$$R_B = \begin{bmatrix} \frac{p}{2} & -1 \\ 1 & -\frac{p}{2} \end{bmatrix}$$

Remark 2.11. $R_B Q^n_{B_p} = \begin{bmatrix} C_{n+1}(p) & -C_n(p) \\ C_n(p) & -C_{n-1}(p) \end{bmatrix}$

Theorem 2.12. $C_n(p)^2 - pC_n(p)C_{n-1}(p) + C_{n-1}(p)^2 + \frac{p^2-4}{4} = 0$ for all integers n.

Theorem 2.13. For $S = \begin{bmatrix} \frac{p}{2} & \frac{p^2-4}{4} \\ 1 & \frac{p}{2} \end{bmatrix}$, $S^n = \begin{bmatrix} C_n(p) & \frac{p^2-4}{4}B_n(p) \\ B_n(p) & C_n(p) \end{bmatrix}$

Theorem 2.14. $C_n(p)^2 - \frac{p^2-4}{4}B_n(p)^2 = 1$ for all integers n.

Theorem 2.15. $C_{mn}(p) = C_m(2C_n(p))$

3 Fast Computation Algorithms For Evaluating Generalised Balancing Sequences

Fast computation method for computations of Generalised Balancing Sequences, B_n using the formulas in Theorem(2.4) above are described in the following. We extend the addition chains used for computations of lucas sequences in [4] in the following.

Definition 3.1. For any integer n the binary expression from left to right is given as follows
$n = \sum_{i=0}^{k} u_i 2^{k-i}$, $u_i = 0$ or 1 for $i \geq 0$ and for any integer $1 \leq j \leq k$
we have $v_{j+1} = \begin{cases} 2v_j & \text{if } \mathrm{u_{j+1}} = 0 \\ 2v_j + 1 & \text{if } \mathrm{u_{j+1}} = 1 \end{cases}$

Theorem 3.2. For any integer $j \geq 0$, if $v_j = \sum\limits_{i=0}^{j} u_i 2^{j-i}$ for $u_i = 0$ or 1 then

$$v_j = \begin{cases} 2v_{j-1} & \text{if } \mathrm{u_j} = 0 \\ 2v_{j-1} + 1 & \text{if } \mathrm{u_j} = 1 \end{cases}$$

Note 3.3. B_{v_j} may be computed using the formulas B_{2v_j+1}, B_{2v_j} and B_{2v_j-1} and proceed so on we have the computation of B_{v_k} giving B_n.

3.1 Addition chains

Addition Chains [7, 8] for n gives a scheme and division of n leading to computation of B_n. In this section we describe the fast computation algorithms for computing the Generalised Balancing Sequences B_n using addition chains [9, 10].

Definition 3.4. An addition chain [11, 12] for n is a sequence of positive integers $\{1 = a_0, a_1, \ldots, a_r = n\}$ with the property that $a_i = a_j + a_k$, for some $k \leq j < i$ for all $i = 1, 2, 3 \ldots, r$.

Definition 3.5. A Lucas addition chain for a positive integer n is a sequence $\{0 = a_{-1}, 1 = a_0, a_1, \ldots, a_r = n\}$ such that

1. $a_{-1} = 0, a_0 = 1 \;\; and \;\; a_r = n$
2. $a_i = a_j + a_k$, for some $k \leq j < i$ for all $i = 1, 2, \ldots, r$
3. $a_j - a_k \in \{a_{-1}, a_0, a_1, \ldots, a_r\}$.

Definition 3.6. A Lucas addition chain $C = \{0, 1, a_1, a_2, \ldots, a_r = n\}$ is called degenerate if one element in the chain is not necessary.

For any positive integer n if the left-to-right binary representation is considered for n then n is given as $n = u_0 2^k + u_1 2^{k-1} + \ldots + u_{k-1}^2 + u_k$ for $u_i = 0$ or 1 and $v_j = \sum_{i=0}^{j} u_i 2^{j-i}$, for any integer $1 \leq j \leq k$ with $v_k = n$. To generate the addition chains we first note the following theorems:

Theorem 3.7. For any j such that $1 \leq j \leq k, v_j$ is the sum of two of its previous values $v_j - 1$ and v_{j-1}

Theorem 3.8. For any j such that $1 \leq j \leq k, v_j$ is the sum of two of its previous values $v_{j-1} + 1$ and $v_{j-2} + 1$

Theorem 3.9. All the addition chains [13]generated by left-to-right binary method for any positive integer n yield the Generalised Balancing Sequence B_n

The addition chains generated by left-to-right binary method
for $n = u_0 2^k + u_1 2^{k-1} + \ldots + u_{k-1}^2 + u_k$ for $u_i = 0$ or 1 and $v_j = \sum_{i=0}^{j} u_i 2^{j-i}$ are given as:

1. $\{v_0, v_1 - 1, v_1, v_1 + 1, \ldots v_{k-1} - 1, v_{k-1}, v_{k-1} + 1, v_k\}$
2. $\{v_0, v_1 - 1, v_1, \ldots v_{k-1} - 1, v_{k-1}, v_k\}$
3. $\{v_{-1}, v_0, v_1, v_1 + 1, \ldots v_{k-1} - 1, v_{k-1}, v_k\}$.

Now we have the following algorithms for computing B_n using the above three addition chains for n.

Algorithm 1 Evaluates B_n using degenerate addition chain

Step 0:(Initialize) Set $M \leftarrow \frac{n}{2^{k-i}}$ where $k = \lfloor \log n \rfloor, i = 0, 1, 2, \ldots, k$
$P \leftarrow 0, Q \leftarrow 1, R \leftarrow Q + 1$

step 1:(Value M) $M \leftarrow \frac{n}{2^{k-i}}$ and determine whether M is even or odd,
if M is even skip to step 4.
step 2: set $P \leftarrow 2Q, Q \leftarrow P + 1$ and $R \leftarrow 2R$
step 3: $[M = n]$, if $M = n$ the algorithm terminates with Q as the answer.
step 4: set$P \leftarrow P + Q, Q \leftarrow 2Q,\ R \leftarrow Q + 1$ and return to step 1.
step 5: **[initialize B_n]** set $B_0 = 0, B_1 = 1$
step 6: For i from 0 *to* k set $B_n \leftarrow B_P, B_Q$ and $\mathrm{B_R}$
set $n \leftarrow 2n$ and compute $B_{2n} \leftarrow pB_n^2 - 2B_n B_{n-1}$
set $n \leftarrow 2n + 1$ compute $B_{2n+1} \leftarrow (pB_n - B_{n-1})^2 - B_n^2$
set $n \leftarrow 2n - 1$ and compute $B_{2n-1} \leftarrow B_n^2 - B_{n-1}^2$

This algorithm uses the addition chain $\{v_{j-1}, v_j, v_{j+1}\}_{j=0}^{k}$ and evaluates B_{v_j} for all $j = 0, 1, \ldots, k$, $\{B_{v_0}, B_{v_1 - 1}, B_{v_1}, B_{v_1 + 1} \ldots B_{v_{j-1} - 1}, B_{v_{j-1}}, B_{v_{j-1} + 1}, B_{v_j}\}$ by using the formulas B_{2n}, B_{2n+1} and B_{2n-1}.

Example 1: In the evaluation of B_n for $n = 171, p = 6$ the above algorithm proceeds according to the steps in the following table:

Table 1. Algorithm ratio

M	P	Q	R
O	0	1	2
E	1	2	3
O	4	5	6
E	9	10	11
O	20	21	22
E	41	42	43
O	84	85	86
E	170	171	-

The values of Generalised Balancing Sequences B_n are given in the following table.

Table 2. Generalised Balancing Sequences

S.No.	$n = 171$	B_n	Values of B_n for $p = 6$
1	0	B_0	0
2	1	B_1	1
3	2	B_2	6
4	3	B_3	35
5	4	B_4	204
6	5	B_5	1189
7	6	B_6	6930
8	9	B_9	1372105
9	10	B_{10}	7997214
10	11	B_{11}	46611179
11	20	B_{20}	361786555939836
12	21	B_{21}	2108646576008245
13	22	B_{22}	12290092900109634
14	41	B_{41}	4315500870452487274745056273129
15	42	B_{42}	25152582330211071001119327984510
16	43	B_{43}	146599993110813938731970911633931
17	84	B_{84}	35788224053879696869571001462142 04796279034658098867281517177020
18	85	B_{85}	208589055822126481067321546712319 74305187342623207117345350572661
19	86	B_{86}	125261979484216042368013288122895 248417300208156863992087744667756
20	170	B_{170}	246126301522698219853497095686129 30923720256387764613708259178106594 92478281149672058239908352660965110 959820251558818153624825086
21	171	B_{171}	1525546956221649872331047785876515496772944 28154645000842208417881651218147096860852176 33430407855805564935497550135912330185454615

The second column gives the degenerate addition chain of length 21 that is used in the computation of B_{171} for $p = 6$.

Algorithm 2 Evaluates B_n using non-degenerate addition chain

step 0:(Initialize) Set $M \leftarrow \frac{n}{2^{k-i}}$ where $k = \lfloor \log n \rfloor, i = 0, 1, 2, \ldots, k$
$P \leftarrow 0, Q \leftarrow 1$
step 1:(Value M) $M \leftarrow \frac{n}{2^{k-i}}$ and determine whether M is even or odd,
if N is even skip to step 4.
step 2: set $P \leftarrow 2Q$ and $Q \leftarrow P + 1$

step 3: $[M = n]$, if $M = n$ the algorithm terminates with R as the answer.
step 4: set $P \leftarrow Q + P$, $Q \leftarrow 2Q$ and return to step 1.
step 5: **[initialize B_n]** set $B_0 = 0, B_1 = 1$

step 6: For i from 0 *to* k set $B_n \leftarrow B_P$ *and* B_Q
set $n \leftarrow 2n$ and compute $B_{2n} \leftarrow pB_n^2 - 2B_nB_{n-1}$
set $n \leftarrow 2n + 1$ and compute $B_{2n+1} \leftarrow (pB_n - B_{n-1})^2 - B_n^2$
set $n \leftarrow 2n - 1$ and compute $B_{2n-1} \leftarrow B_n^2 - B_{n-1}^2$

This algorithm uses non degenerate addition chain $\{v_j - 1, v_j\}_{j=0}^{k}$ and evaluates B_{v_j} for all $j = 0, 1, \ldots, k$, $\{B_{v_0}, B_{v_1-1}, B_{v_1}, \ldots B_{v_{t-1}-1}, B_{v_{t-1}}, B_{v_t}\}$ by using the formulas $B_{2n}, B_{2n-1}, B_{2n+1}$.

Example 2: In the evaluation of B_n for $n = 171, p = 6$ with the above algorithm proceeds according to the steps in the following table:

Table 3. Algorithm ratio and frequency

M	P	Q
O	0	1
E	1	2
O	4	5
E	9	10
O	20	21
E	41	42
O	84	85
O	170	171

This algorithm may be used in the cryptosystem with Generalised Balancing Sequences. The values of Generalised Balancing Sequences B_n are given in the following Table 4.

Table 4. values of Generalised Balancing Sequences in cryptosystem

S.No.	$n = 171$	B_n	Values of B_n
1	0	B_0	0
2	1	B_1	1
3	2	B_2	6
4	3	B_3	35
5	4	B_4	204
6	5	B_5	1189
7	9	B_9	1372105
8	10	B_{10}	7997214
9	20	B_{20}	361786555939836
10	21	B_{21}	2108646576008245
11	41	B_{41}	4315500870452487274745056273129
12	42	B_{42}	25152582330211071001119327984510
13	84	B_{84}	35788224053879696869571001462142 04796279034658098867281517177020
14	85	B_{85}	208589055822126481067321546712319 74305187342623207117345350572661
15	170	B_{170}	246126301522698219853497095686129 30923720256387764613708259178106594 92478281149672058239908352660965110 959820251558818153624825086
16	171	B_{171}	1525546956221649872331047785876515496772944 28154645000842208417881651218147096860852176 33430407855805564935497550135912330185454615

The second column gives the non degenerate addition chain of length 16 that is used in the computation of B_{171}.

Algorithm 3: This algorithm uses Lucas addition chain $\{v_j, v_{j+1}\}_{j=0}^{k}$ and evaluates B_{v_j} for all $j = 0, 1, \ldots, k$, $\{B_{v_0}, B_{v_1}, B_{v_1+1} \ldots B_{v_{j-1}}, B_{v_{j-1}+1}, B_{v_j}\}$ by using the formula B_{x+y}.

Algorithm 3 Evaluates B_n using Lucas chain

step 0:(Initialize) Set $M \leftarrow \frac{n}{2^{k-i}}$ where $k = \lfloor \log n \rfloor, i = 0, 1, 2, \ldots, k$
$Q \leftarrow 1, R \leftarrow 2$

step 1:(Value M) $M \leftarrow \frac{n}{2^{k-i}}$ and determine whether M is even or odd,
if M is even skip to step 4.

step 2: set $Q \leftarrow 2Q + 1$ and $R \leftarrow 2R$

step 3: $[M = n]$, if $M = n$ the algorithm terminates with Q as the answer.

step 4: set $Q \leftarrow 2Q$, $R \leftarrow Q + 1$ and return to step 1.

step 5: **[initialize B_n]** set $B_0 = 0, B_1 = 1$

step 6: For i from 0 *to* k set $B_n \leftarrow B_Q$ *and* B_R
set $n \leftarrow x + y$ and compute $B_{y+z} \leftarrow B_y B_{z+1} - B_{y-1} B_z$

Example 3 In the evaluation of B_n for $n = 171, p = 6$ with the above algorithm is proceeds according to the steps in the following Table 5.

Table 5. Algorithm ratio

M	Q	R
O	2	2
E	2	3
O	5	6
E	10	11
O	21	22
E	42	43
O	85	86
O	171	-

The values of Generalised Balancing Sequences B_n for $p = 6$ are given in the following table.

Table 6. Generalised Balancing Sequences Bn for p = 6

S.No.	$n = 171$	B_n	Values of B_n
1	0	B_0	0
2	1	B_1	1
3	2	B_2	6
4	3	B_3	35
5	5	B_5	1189
6	6	B_6	6930
7	10	B_{10}	7997214
8	11	B_{11}	46611179
9	21	B_{21}	2108646576008245
10	22	B_{22}	12290092900109634
11	42	B_{42}	25152582330211071001119327984510
12	43	B_{43}	146599993110813938731970911633931
13	85	B_{85}	208589055822126481067321546712319 74305187342623207117345350572661
14	86	B_{86}	125261979484216042368013288122895 248417300208156863992087744667756
15	171	B_{171}	1434529211908353815216618270928136371147 6313391622299866483575686232240636013 160008774791903510965216588443108573471 834951124409995

Remark 3.10. All the algorithms discussed above can be extended for a computations of Generalised Lucas Balancing sequences.

4 Key Stream for Encryption with solutions of Diophantine Equation $x^2 - axy + y^2 - x = 0$

In this section we construct a key stream for encryption in classical cryptosystems which can be exchanged regularly. In the following first a shared secret key a called key exchange is generated by using Diffe-Hellmann and then a shared key stream is developed using concatenation of solutions of Diophantine Equation $x^2-axy+y^2-x = 0$ that are in terms of Generalised Balancing Numbers.This Key Stream is such that all the parties in the communication each generate the key stream using the Key Exchange. In the construction of the proposed cryptosystem we adapt the discrete log problem of Lucas Sequences to Generalised Balancing Sequences and extend the Diffe-Hellmann protocol with Generalised Balancing Sequences.

4.1 Key Stream with solutions of Diophantine Equation $x^2 - axy + y^2 - x = 0$

The key stream proposed is based on solutions of Diophantine Equation $x^2-axy+y^2-x = 0$. For all the parties in this communication note a in the Diophantine Equation is the shared key and is secret. So first this shared secret is comminicated through Diffe-Hellman key exchange using Generalised Lucas Balancing Sequences as follows.

4.1.1 Generating the shared secret key

Now we generate the shared key using Generalised Balancing Sequence and Generalised Lucas Balancing Sequences.

4.1.2 Diffe-Hellman with generalised balancing sequence and generalised Lucas balancing sequences

Now with the knowledge of p, all the parties generate the coefficient a of the diophantine equation $x^2 - axy + y^2 - x = 0$ as a secret key using Diffe-Hellmann protocol for Generalised Balancing Sequence and Generalised Lucas Balancing Sequences follows:

- The sender chooses random Generalised Lucas Balancing Sequence $C_m(p)$ and makes $(C_m(p), p)$ public.
- The receiver chooses random Generalised Balancing Sequence $C_n(p)$ and makes $(C_n(p), p)$ public .
- The sender and receiver compute $(C_{nm}(p))$ using theorem(2.15), as the secret key.
- The secret key $C_{mn}(p)$ with the sender and receiver is selected as the coefficient a of the diophantine equation $x^2 - axy + y^2 - x = 0$.

4.1.3 Generating the common key stream using Diophantine Equation $x^2 - pxy + y^2 - x = 0$

A Key Stream using a solution of a particular class of Diophantine Equation is generated as follows. Both sender and receiver agree upon positive integer α and compute all the Generalised Balancing sequences from $B_{2\alpha-1}$ using fast computation algorithm.Considers a class of solutions $(B_{2n}(p)^2, B_{2n}(p)B_{2n-1}(p))$ of the Diophantine

Equation $x^2 - pxy + y^2 - x = 0$ as in [2] for $\alpha \leq n$. Then concatenate solutions $(B_{2n}(p)^2, B_{2n}(p)B_{2n-1}(p))$ of Diophantine equation for all $\alpha \leq n$ and obtain key stream as

$$k = B_{2\alpha}(p)^2 B_{2\alpha}(p)B_{2\alpha-1}(p)\cdots$$

Encryption:

The encryption with Key stream is as follow.

- The sender convertes the plaintext message units into the numerical equivalent in $\mathbb{Z}_N$ and arrange them in a matrix M of order of his choice (say $u \times v$) $M = [m_{ij}]_{u\times v}$ for m_{ij} is the numerical equivalent of each character.
- The key stream is fragmented as units where each unit is of $\log n$ digits from left to right, these units are arranged as a key matrix $K = [k_{ij}]_{u\times v}$ of order same as that of M.
- The message unit matrix M is encrypted as enciphered matrix C using te Key matrix K as follow. $C = M + K$, that is, each entry of the matrix is computed as, $[c_{ij}] = [m_{ij}] + [k_{ij}]$ in $\mathbb{Z}_n$ and sends the matrix C to the receiver.

Decryption:

- After receiving the enciphering matrix, C the key matrix K of order same as that of C, whose each entry is $\log n$ digits taken at a time from the key stream is constructed. Then note $K = [k_{ij}]_{u\times v}$, that is, K is exactly the same as the key matrix constructed by the sender.
- The receiver retrieves the message by computing the message matrix $M = C - K$, that is each entry is computed as, $[m_{ij}] = [c_{ij}] - [k_{ij}]$ in $\mathbb{Z}_n$. Now, writing the entries of the matrix in a single row and writing its alphabetic equivalents, the receiver reads the message.

Example: Suppose sender **A** wants to send the message "YOUR PIN NUMBER IS FOUR ONE TWO SIX" to receiver **B**, in an encrypted manner and 27-letter alphabet is used with numerical equivalents of $A - Z$ are 0-25 and that of blank space is 26.A makes $(17, 6)$ public and B makes $(99, 6)$ and agree on α=2.

The numerical equivalent of message is "24-14-20-17-26-15-08-13-26-13-20-12-01-04-17-26-08-18-26-05-14-20-17-26-14-13-04-26-19-22-14-26-18-08-23-26". When converted to matrix

form we get $M=\begin{bmatrix} 24 & 14 & 20 & 17 & 26 & 15 \\ 08 & 13 & 26 & 13 & 20 & 12 \\ 01 & 04 & 17 & 26 & 08 & 18 \\ 26 & 05 & 14 & 20 & 17 & 26 \\ 14 & 13 & 04 & 26 & 19 & 22 \\ 14 & 26 & 18 & 08 & 23 & 26 \end{bmatrix}$.

With the knowledge of $(17, 6)$ and $(99, 6)$ A and B calculate $C_{mn}(p)$= 19601=a

Encryption: Diophantine Equation $x^2 - axy + y^2 - x = 0$ has solutions of the form $(B_{2n}(p)^2, B_{2n}(p)B_{2n-1}(p))$. The sender and receiver concatenate $B_{2\alpha}^2 B_{2\alpha} B_{2\alpha-1}$..... to generate the key stream.

S.No	n	B_{2n}	$(B_{2n}^2, B_{2n}B_{2n-1})$
1	2	$B_4(19601)$	(567112692779926378231608 01,289328449699513612080 0)
2	3	$B_6(19601)$	(83710951369755492045407747394319923225600 00, 4270749022378436526411333284719376784 00)
3	4	$B_8(19601)$	(12356491872681302882917194759285551576227576025112211186776 01, 6304010971298904824603442381050883855282297514751617960 1)

Keystream:56711269277992637823160801289328449699513612080083710951369755492045407747394319923225
60000427074902237843652641133328471937678400123564918726813028829171947592855515762275760251122
1118677601630401097129890482460344238105088385528229751475161796O1

Now, construct key matrix K by considering $\log 27 = 2$ digits taken at a time from k as an entry of the matrix K of same order as matrix M i.e., 6×6, we have

$$K = \begin{bmatrix} 56 & 71 & 12 & 69 & 27 & 79 \\ 92 & 63 & 78 & 23 & 16 & 08 \\ 01 & 28 & 93 & 28 & 44 & 96 \\ 99 & 51 & 36 & 12 & 08 & 00 \\ 83 & 71 & 09 & 51 & 36 & 97 \\ 55 & 49 & 20 & 45 & 40 & 77 \end{bmatrix}$$

Now, the sender computes the cipher matrix C, as $(M+K) \mod 27$

$C = (M+K) \mod 27$

$$= \begin{bmatrix} 24 & 14 & 20 & 17 & 26 & 15 \\ 08 & 13 & 26 & 13 & 20 & 12 \\ 01 & 04 & 17 & 26 & 08 & 18 \\ 26 & 05 & 14 & 20 & 17 & 26 \\ 14 & 13 & 04 & 26 & 19 & 22 \\ 14 & 26 & 18 & 08 & 23 & 26 \end{bmatrix} + \begin{bmatrix} 56 & 71 & 12 & 69 & 27 & 79 \\ 92 & 63 & 78 & 23 & 16 & 08 \\ 01 & 28 & 93 & 28 & 44 & 96 \\ 99 & 51 & 36 & 12 & 08 & 00 \\ 83 & 71 & 09 & 51 & 36 & 97 \\ 55 & 49 & 20 & 45 & 40 & 77 \end{bmatrix} \mod 27$$

$$= \begin{bmatrix} 26 & 04 & 05 & 05 & 26 & 13 \\ 19 & 22 & 23 & 09 & 09 & 20 \\ 02 & 05 & 02 & 00 & 25 & 06 \\ 17 & 02 & 23 & 05 & 25 & 26 \\ 16 & 03 & 13 & 23 & 01 & 11 \\ 15 & 21 & 11 & 26 & 14 & 22 \end{bmatrix}$$

Now, the sender sends his enciphered message C, to the receiver.

Decryption: Then, the receiver decrypts C and retrieves M as below:
$M = (C-K) \mod 27$

$$= \begin{bmatrix} 26 & 04 & 05 & 05 & 26 & 13 \\ 19 & 22 & 23 & 09 & 09 & 20 \\ 02 & 05 & 02 & 00 & 25 & 06 \\ 17 & 02 & 23 & 05 & 25 & 26 \\ 16 & 03 & 13 & 23 & 01 & 11 \\ 15 & 21 & 11 & 26 & 14 & 22 \end{bmatrix} - \begin{bmatrix} 56 & 71 & 12 & 69 & 27 & 79 \\ 92 & 63 & 78 & 23 & 16 & 08 \\ 01 & 28 & 93 & 28 & 44 & 96 \\ 99 & 51 & 36 & 12 & 08 & 00 \\ 83 & 71 & 09 & 51 & 36 & 97 \\ 55 & 49 & 20 & 45 & 40 & 77 \end{bmatrix} \mod 27$$

$$= \begin{bmatrix} 24 & 14 & 20 & 17 & 26 & 15 \\ 08 & 13 & 26 & 13 & 20 & 12 \\ 01 & 04 & 17 & 26 & 08 & 18 \\ 26 & 05 & 14 & 20 & 17 & 26 \\ 14 & 13 & 04 & 26 & 19 & 22 \\ 14 & 26 & 18 & 08 & 23 & 26 \end{bmatrix} = M$$

Now, writing the entries of the matrix in a single row, we obtain "24-14-20-17-26-15-08-13-26-13-20-12-01-04-17-26-08-18-26-05-14-20-17-26-14-13-04-26-19-22-14-26-18-08-23-26" whose corresponding alphabets reads as "**YOUR PIN NUMBER IS FOUR ONE TWO SIX**".Thus, the receiver retrieves the message.

We would discuss efficiency and time analysis of our proposed algorithm in the coming sub-section.

5 Efficiency Analysis

The encryption with key stream generated are based on computations of the Generalised Balancing Sequences and Generalised Lucas Balancing Sequences using the three algorithms. Addition chain as in algorithm 1 is of length $3logn - 1$ as it is the composition of the basic chains $\{v_{t-1} - 1, v_{t-1}, v_{t-1} + 1\}$for all $t =, 1, 2, 3, ...k$ for $k = logn$.Addition chain as in algorithm 2 is of length $2logn$ as it is the composition of the basic chains $\{v_t - 1, v_t\}$ for all $t = 1, 2, 3, ...k$ for $k = logn$.And finally addition chain as in algorithm 3 is of length $2logn - 1$ as it is the composition of the basic chains $\{v_t, v_t + 1\}$ for all $t = 1, 2, 3, ...k$ for $k = logn$.

6 Cryptanalysis

The cryptanalysis depends on the computations of $C_n(p)$ and $B_m(a)$ where a is the coefficient of the Diophantine Equation considered. Note $C_n(p)$ and a are both based on discrete log with Generalised Lucas Balancing Sequences. The discrete log for Generalised Balancing Sequences is same as the discrete log on Lucas Balancing Sequences. In [14] the discrete log on Lucas Sequences is $O(log^2(N))$ for all $p \in \mathbb{Z}_N$ for N with small primes and is exponential for N with large primes.

7 Summary

In this paper we proposed an encryption using a keystream which concatenation of a class of solutions of diophantine equation $x^2 - axy + y^2 - x = 0$, where a is a shared secret that is computed by both the parties by using the Diffe-Hellmann protocol in terms of Generalised Lucas Balancing Sequences. The fast computations algorithms are used for calculation of terms of Generalised Balancing and Generalised Lucas Balancing Sequences. By using different diophantine equations $x^2 \pm axy + y^2 \pm 1 = 0$, $x^2 \pm axy + y^2 \pm x = 0$, $x^2 \pm pxy + y^2 \pm (\frac{p^2-4}{4}) = 0$ and $x^2 \pm pxy + y^2 \pm (\frac{p^2-4}{4})x = 0$ and their corresponding classes of solutions as in [1] and 4 we can also construct other keystreams for encryption.

Competing Interests

Authors have declared that no competing interests exist.

References

[1] Anuradha Kameswari P, Anoosha K. Solutions of Diophantine Equations Generated By Generalised Lucas Balancing Sequences. Int. J. Math. And Appl. 2021;9(4):75-88.

[2] Anuradha Kameswari P, Anoosha K. A Generalised Balancing Sequence and Solutions of Diophantine Equations $x^2 \pm pxy + y^2 \pm x = 0$. Communications of Mathematics and application. 2021;12(4).

[3] Brauer A. on addition chains. Bull. Amer. Math. Soc. 1939;45:736-739.

[4] Anuradha Kameswari P, Ravitheja B. Addition Chain For Lucas Sequences With Fast Computation Method. International Journal of Applied Engineering Research. 2018;13(11):9413-9419.

[5] Smith PJ, Lennon GJJ. VLUC: A new public key cryptosystem. Ninth IFIP Symposium on Computer Science Security, Elsevier Science Publication. 1993;103-117.

[6] Petre L. Montgomery. Evaluating recurrences of the sum the form $X_{m+n} = f(X_m, X_n, X_{m-n})$ via Lucas chains; 1992.

[7] Behera A, Panda GK. On the square roots of triangular numbers. The Fib. Quart. 1999;37:98–105.

[8] Zulkarnain Md Ali, Othman M, Said MRM, Sulaiman MN. Computations of cryptosystems based on Lucas functions using addition chain. IEEE. 2010;1082-1086.

[9] Lehmer DH. An Extendeds theory of Lucas functions. Annals of Math. 1930;31:419-448.

[10] Downey P, Loney B, Sethi R. Computing Sequences with addition chains. SIAM Journal of Computing. 1981;10(3):638-646.

[11] Dickson LE. History of Theory of Numbers. Chelsea Publication Company, New York.

[12] Knuth DE, LU. The art of computer programming, Volume-II: Seminumericals Algorithms, Third Edition, Addison-Wesley.

[13] Daniel Bleinchenbacher. Efficiency and Security of Cryptosystems based on Number Theory. Ph.D thesis; 1964.

[14] Anuradha Kameswari P, Surendra T, Ravitheja B. Shank's Baby-step Gaint-step Attack extended to discrete log with Lucas sequences. IOSR Journal of Mathematics. 2016;12(1):09-16.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
https://www.sdiarticle5.com/review-history/94337