A Cryptosystem of Skewed Affine Cipher of Multiple Keys

ECS Transactions, 107 (1) 15071-15080 (2022)
10.1149/10701.15071ecst ©The Electrochemical Society

A Cryptosystem of Skewed Affine Cipher of Multiple keys

Jayanti Sravani[a], K Chittibabu[a], and Prof. A Chandra Sekhar[a]

[a] Department of Mathematics, GITAM, Visakhapatnam 530045, Andhra Pradesh, India

In this era where communication over technology has become vital, the reliability of the same is of utmost need. Cryptography ensures confidentiality, user authentication, and integrity of data. One of the techniques is the Elliptic Curve Cryptography (ECC). Several classical ciphers are designed based on mathematical backgrounds. In this paper, we focus on combining Affine Cipher and ECC to magnify the security provided by an Affine cipher. Hence a skewed Affine cipher that uses multiple keys over Elliptic curves is proposed. The keys chosen are derived from the points on the specified Elliptic curve, which forms a cyclic group or a cyclic subgroup.

Introduction

Cryptology is the study of writing codes and cracking them. The system used to intensify the security over communication uses cryptology for its effectiveness. There is much advancement in this field where the combination of Hill cipher and Elliptic curve cryptosystem (ECC), the combination of Affine and Hill cipher(6)(10) has been witnessed to ensure that the security is magnified (2)(4)(5)(7)(8)(9). Elliptic Curve Cryptography (ECC) is one of the fields in cryptography that include Elliptic curves and points on it in a defined manner for performing encryption and decryption. This cryptosystem is famous for its smaller key size (1). The ElGamal algorithm is one of the most used techniques for encrypting encryption and decryption over Elliptic curves.

In this paper, we have proposed a cryptosystem that works on two keys which are the points over the chosen Elliptic curve. The encryption and decryption algorithms are similar to that of Affine Cipher.

The concepts involved in designing the cryptosystem are:

Elliptic curve

An elliptic curve (3) over real numbers is of the form:

$$y^2 = x^3 + ax + b \qquad [1]$$

If $4a^3 + 27b^2 \neq 0$, then the curve is a non-singular Elliptic curve. Else, it is a singular Elliptic curve. Generally, the points on an Elliptic curve form an abelian group for the point addition defined over the points on an Elliptic curve.

Elliptic curves modulo a prime (p>3)

The points satisfying

$$E_p(a,b) : y^2 = x^3 + ax + b(\text{mod } p) \quad [2]$$

together with the point at infinity(say O) form an Elliptic curve where $x, y, a, b \in \{0,1,\dots,p-1\} \ni 4a^3 + 27b^2 \neq 0(\text{mod } p)$.

Point addition on Elliptic curve For $(a_1, b_1), (a_2, b_2) \in E$,

1. If $a_1 \neq a_2$ then $(a_1, b_1) + (a_2, b_2) = (a_3, b_3)$ where

$$a_3 = (h * h - a_1 - a_2)\text{mod } p, \quad [3]$$

$$b_3 = (h * (a_1 - a_3) - b_1)\text{mod } p \quad [4]$$

$$\text{for } h = \frac{(b_1 - b_2)}{(a_1 - a_2)} \text{ mod } p. \quad [5]$$

2. If $a_1 = a_2$ and $b_1 = b_2$ then $(a_1, b_1) + (a_2, b_2) = (a_3, b_3)$ where

$$a_3 = (h * h - a_1 - a_2)\text{mod } p \text{ and} \quad [6]$$

$$b_3 = (h * (a_1 - a_3) - b_1)\text{mod } p \quad [7]$$

$$\text{for } h = \frac{(3a_1^2 + a)}{2b_1} \text{ mod } p. \quad [8]$$

3. If $a_1 = a_2$ and $b_1 = -b_2$ then $(a_1, b_1) + (a_2, b_2) = O(\text{point at infinity})$.

If the total number of points on the curve (including the point at infinity) is prime or product of distinct primes, then the EC $E_p(a,b)$ forms a cyclic group concerning point addition on Elliptic curves where each point is a generator point (except the point at infinity)(3).

If the total number of points on the curve (including the point at infinity) is composite, the Elliptic curve doesn't form a cyclic group. However, all points generated by a single point on the Elliptic curve form a cyclic subgroup of the Elliptic curve, and hence each point of the subgroup so formed will be a generator point of that subgroup.

The number of points on an Elliptic curve modulo a prime q (say n) satisfies the following inequality:

$$q + 1 - 2\sqrt{q} \leq n \leq q + 1 + 2\sqrt{q}. \quad [9]$$

This inequality can roughly estimate the number of points on an Elliptic curve.

ElGamal algorithm over Elliptic curve E to exchange a key k between a Sender(S) and a Recipient:
Arbitrary integer chosen by Recipient(R): d
Random integer chosen by Sender(S): r
Step 1:
R chooses a point p_1 on E

Then R computes $q_1 = p_1 * d$
R sends (E, p_1, q_1) to S.
Step 2:
After receiving(E, p_1, q_1), S computes p_2 and q_2 as:
$p_2 = r * p_1; q_2 = k + r * q_1$
S sends (p_2, q_2) to R.
Step 3:
Finally, R retrieves the key by computing: $k = q_2 - d * p_2$

Diffie Hellman Key exchange algorithm to exchange a key between a Sender(S) and a Recipient(R)
Elliptic curve used: E
Initially, a point $P(x, y)$ on E is made public.
S chooses random integer h and sends $h * (x, y)$ to R.
R, on the other hand, chooses some arbitrary integer t and sends $t * (x, y)$ to S.
S on receiving $t * (x, y)$ computes key value as $h * (t * (x, y))$, and R on receiving $h * (x, y)$ computes key value as $t * (h * (x, y))$.
Finally, the key exchanged between them is $h * (t * (x, y))$, which is the same as $t * (h * (x, y))$ as $h * t = t * h$.

Proposed Method

The following is the proposed method for securely sending the information where Skewed Affine Cipher and ECC are applied.

Step 1

Assign numeric values to the alphabets by defining a bijective map from the set of alphabets to the set $\{1,2, \ldots ,26\}$ as $A \leftrightarrow 1, B \leftrightarrow 2, \ldots, Z \leftrightarrow 26$.

Step 2

Choose an Elliptic curve $E_p(h, k)$: $y^2 = x^3 + hx + k (\mathrm{mod}\ p)$ and a generator point on the curve in a way that the point generates more than 26 points on the curve, which would form a cyclic group with respect to the point addition on the curve.

Step 3

Choose two points from the points on the curve in step 2, say A and B, which serve as the secret keys exchanged between the sender and the recipient using Diffie- Hellman and Elgamal key exchange algorithms, respectively.

Step 4

For each plain text(P) character, Ciphertext (C) is computed as

$$C = A * P + B \qquad [10]$$

where $*$ and $+$ are point multiplication and addition over Elliptic curve points. This Ciphertext is communicated to the recipient over a secure channel which is a point on the chosen Elliptic curve.

Step 5

Computing $A * P = C - B$ retrieves the plain text character for the received Ciphertext (C) character. To retrieve P, one must add 'A' P times to obtain $C - B$. Then the number of times 'A' has been added to yield $C - B$ gives the value of the required P.

Pseudo code for computing the value of P :

Define f(a, b) $\left(\text{a function which adds a, b } \left(\text{points on } E_p(h, k)\right)\right)$

Initialise count $= 1$
Set result $= A$
while(result! $= C - B$)
{
result $=$ f(result, A)
count $++$
}
$P =$ count
Output P

Implementation

Code for the implementation of the above-designed algorithm

Header files included

```
#include<iostream>
#include<cmath>
using namespace std;
```

Figure 1

Function to list the points on the Elliptic curve

```
void ECCpoints(int fp,int fa,int fb)
{
    int x=0;
    double w;
    while(x<fp)
    {
        w=(((x*x*x)+(fa*x)+fb)%fp);
        for(int i=0;i<fp;i++)
        {
            if((i*i)%fp==w)
                cout<<x<<","<<i<<endl;
        }
        x++;
    }
}
```

Figure 2

Function to perform point addition and multiplication over a given Elliptic curve

```
void Add(int &x1,int &y1,int x2,int y2,int fp,int fa)
{
    int s;
    int x3,y3,q,q1;
    if(x1!=x2||y1!=y2)
    {
        q=(x1-x2)%fp;
        while(q<0)
        {
            q=q+fp;
        }
        for(int i=0;i<fp;i++)
        {
            if(((q*i)%fp)==1)
            {
            q1=i;
            }
        }
        s=(((y1-y2)*q1)%fp);
        while(s<0)
        {
            s=s+fp;
        }
        x3=(((s*s)-x1-x2)%fp);
        while(x3<=0)
            x3=x3+fp;
        y3=((s*(x1-x3)-y1)%fp);
        while(y3<=0)
            y3=y3+fp;
    }
```

Figure 3

```
    else
    {
        q=((2*y1)%fp);
    while(q<0)
    {
        q=q+fp;
    }
    for(int i=0;i<fp;i++)
    {
        if(((q*i)%fp)==1)
        {
        q1=i;
        }
    }
    s=((((3*x1*x1)+fa)*q1)%fp);
    while(s<0)
    {
        s=s+fp;
    }
    x3=((s*s)-(2*x1))%fp;
    while(x3<0)
    {
        x3=x3+fp;
    }
    y3=(s*(x1-x3)-y1)%fp;
    while(y3<0)
    {
        y3=y3+fp;
    }
    }
    x1=x3;
    y1=y3;
}
```

Figure 4

Encryption

```
main()
{
        int a,b,p,k1x,k2x,k1y,k2y,c,a1,a2,b1,b2;
        char PT;
        cout<<"Enter the details of the Elliptic curve you wish to work with: p = ";
        cin>>p;
        cout<<"a = ";
        cin>>a;
        cout<<"b = ";
        cin>>b;
        cout<<"Enter the Plain text character to be encrypted:";
        cin>>PT;
        if(PT>='A'  && PT<='Z')
          c=PT-64;
        cout<<"Enter the key values:\nA = ";
        cin>>k1x>>k1y;
        cout<<"B = ";
        cin>>k2x>>k2y;
        if(c==1)
        {
            a1=k1x;
            a2=k1y;
        }
        else
        {
            a1=k1x;
            a2=k1y;
            for(int i=2;i<=c;i++)
            {
                b1=k1x;
                b2=k1y;
                Add(a1,a2,b1,b2,p,a);
            }
        }
        Add(a1,a2,k2x,k2y,p,a);
        cout<<"Cipher text for the plain text "<<PT<<"is: ("<<a1<<","<<a2<<")";
}
```

Figure 5

Output

```
Enter the details of the Elliptic curve you wish to work with: p = 41
a = 1
b = 1
Enter the Plain text character to be encrypted:S
Enter the key values:
A = 24
14
B = 32
40
Cipher text for the plain text Sis: (34,26)
--------------------------------
Process exited after 11.33 seconds with return value 0
Press any key to continue . . .
```

Figure 6

Decryption

```
main()
{
    int p,a,b,c1,c2,c,a1,a2,b1,b2;
    char PT;
    cout<<"Enter the details of the Elliptic curve you wish to work with:\np = ";
    cin>>p;
    cout<<"a = ";
    cin>>a;
    cout<<"b = ";
    cin>>b;
    cout<<"Enter the cipher text : (c1,c2) = ";
    cin>>c1>>c2;
    cout<<"Enter the key values:\nA = ";
    cin>>a1>>a2;
    cout<<"B =";
    cin>>b1>>b2;
    b2=p-b2;
    cout<<"Cipher text is: ("<<c1<<","<<c2<<")"<<"\nKey values are: A=("<<a1<<","<<a2<<")"<<"B=("<<b1<<","<<b2<<")";
    Add(c1,c2,b1,b2,p,a);
    int count=1;
    int d1=a1;
    int d2=a2;
    while(c1!=a1 || c2!=a2)
    {
        Add(a1,a2,d1,d2,p,a);
        count++;
    }
    if(count>=1 && count<=26)
    {
        PT=(char)(64+count);
        cout<<"\nThe plain text character encrypted is:"<<PT;
    }
}
```

Figure 8

Output

```
Enter the details of the Elliptic curve you wish to work with:
p = 41
a = 1
b = 1
Enter the cipher text : (c1,c2) = 34
26
Enter the key values:
A = 24
14
B =32
40
Cipher text is: (34,26)
Key values are: A=(24,14)B=(32,1)
The plain text character encrypted is:S
--------------------------------
Process exited after 12.45 seconds with return value 0
Press any key to continue . . .
```

Figure 9

Let the Elliptic curve be $E_{41}(1,1) : y^2 = x^3 + x + 1 (\mathrm{mod}\ 41)$. Then $o(E) = 35$, a product of two distinct primes 5 and 7. Hence the chosen elliptic curve is a cyclic group where each of its points generates all the other points on the curve.

TABLE I. Points on $E_{41}(1,1)$

(0,1)	(0,40)
(3,20)	(3,21)
(5,7)	(5,34)
(6,10)	(6,31)
(7,8)	(7,33)
(9,1)	(9,40)
(11,20)	(11,21)
(19,14)	(19,27)
(24,14)	(24,27)
(27,20)	(27,21)
(28,13)	(28,28)
(31,4)	(31,37)
(32,40)	(32,1)
(34,15)	(34,26)
(35,5)	(35,36)
(39,14)	(39,27)
(40,9)	(40,32)
O = point at infinity	

For the plain text character ‘S’ and the key values A = (24,14) , B = (32,40), the cipher text obtained is (34, 26) (11).

Cryptanalysis

Cryptanalysis is cracking the codes to retrieve the secret key involved in encrypting the information (7). The brute force attack is carried out to trace the keys used in the algorithm. The total number of possible keys for the designed cryptosystem

$$= \begin{cases} o(E) * o(E) & , E \text{ is a cyclic Elliptic curve} \\ o(CE) * o(E) & , CE = \text{Cyclic subgroup of Elliptic curve E} \end{cases}$$

Here, the number of possibilities for the first key 'A' is $o(E)$ or $o(CE)$, and for the second key 'B' is $o(E)$, respectively. In the example discussed in the implementation section, the number of possible keys is 35*35 =1225. By sufficiently increasing the order of the cyclic group formed from the points on the Elliptic curve, the algorithm's security can be magnified.

Conclusions

A cryptosystem that looks like an Affine cipher termed as a Skewed Affine Cipher where encryption is performed over points on an Elliptic curve is proposed.

The necessary condition for the developed cryptosystem to work is that the first key must be chosen from the set of points on an Elliptic curve that form a cyclic subgroup.

Also, the cyclic group's order must be greater than the order of the character set (which is 26 in the example discussed). These conditions cannot be relaxed as there would be a possibility of deriving the same cipher texts for different characters in the plain text. Example: Consider an Elliptic curve $E_{29}(2,3)$. Let $A = (5,15)$ and $B = (14,7)$. We obtain the cipher text (16,19) for the plaintext characters 'D'=4 and 'M'=13. This is because $9A = 0$(the point at infinity).

The operations point addition and multiplication over the points of an Elliptic curve are difficult to perform for solving a system of equations to find an unknown point. Therefore it is computationally difficult to trace the keys used in our algorithm by the method of frequency analysis. Thus intensifying the security given by an Affine cipher.

The security of the ElGamal algorithm used to exchange the first key lies in the difficulty of solving the Elliptic curve logarithm problem. Although the same situation may arise in the decryption algorithm of the proposed method, the number of characters when confined to 26(alphabets) or 256(ASCII characters) is comparatively small and hence will eradicate the possibility of errors. The working of the proposed algorithm is tested in C++ programming language for English alphabets using the DEV C++ compiler.

The other key is exchanged using the Diffie Hellman key exchange algorithm, allowing two unknown people to agree over a shared key. The encryption and decryption algorithms are simple, but the decryption algorithm is desirable only when a limited number of characters are present.

The security of the proposed method can be enhanced by not displaying the Elliptic curve involved in the algorithm publicly. This can be done by privately sharing the cyclic Elliptic curve or a generator point of a non-cyclic Elliptic curve which forms a cyclic subgroup. This will allow only the sender and the recipient to know the set of points on an Elliptic curve from where keys are chosen, eradicating the brute force attack. Hence, there is a need for an efficient key exchange protocol that would increase the method's security.

Acknowledgments

We extend our sincere gratitude to GITAM for supporting our work by providing Dr. M.V.V.S. Murthi Research Fellowship.

References

1. Behrouz A Forouzan and Debdeep Mukhopadhyay, *Cryptography and network security*, p.283-290, Third edition, McGraw Hill Education.
2. Dewi Sartika Ginting, Kristin Sitompul, Jasael Simanulang, Rahmat Widia Sembiring, and Muhammad Zarlis, *ASTESJ*, **2**(5), p.6-12 (2017).
3. Douglas R Stinson and Maura B Paterson, *Cryptography Theory and Practice*, Fourth Edition, p.278-286, CRC Press, Taylor & Francis group.
4. Inam, S and Ali R, *Neural Comput & Applic*, **29**, p.1279–1283 (2018). https://doi.org/10.1007/s00521-016-2745-2

5. Kalika Prasad and Hrishikesh Mahato, *Journal of Discrete Mathematical Sciences & Cryptography*,(2021). DOI: 10.1080/09720529.2020.1838744
6. K.P. Satyam, P. Sundarayya, and M.G. Vara Prasad, *OJATM*, **2**(4), p.88-98,2016.
7. Laiphrakpam Dolendro Singh and Khumanthem Manglem Singh, *Procedia Computer Science,* **54**, p.73-82 (2015).
8. Somdip Dey, SD-AREE : A New Modified Caesar Cipher Cryptographic Method Along with Bit-Manipulation to Exclude Repetition from a Message to be Encrypted,(2012). https://arxiv.org/ftp/arxiv/papers/1205/1205.4279.pdf
9. Sriramoju Ajay Babu, *IJRSE*, **3**(2), (2017).
10. Wulandari,S.Y, *Proceeding ICSE*, **3**, p.741-744 (2020). https://doi.org/10.14421/icse.v3.595
11. http://www.christelbach.com/ECCalculator.aspx