Elliptical Curve Cryptography for Image Encryption

Ramandika Pranamulia / 13512078
Computer Science Department
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
ramandika@students.itb.ac.id

***Abstract*—The needs to encrypt digital data are pretty high nowadays, not only text data but also multimedia data such as images. One of the algorithm that's well known for its cipher strength is Elliptical Curve Cryptography which is claimed to be stronger than RSA. One of the major problem to encrypt image is the size of the image which is usually pretty large for high resolution images. In this paper the author will try to reduce the computation complexity of encryption as well as minimizing the memory consumption. At the end the author will test the performance after the optimization process and comparing with the naive algorithm.**

Keywords—encrypt; RSA; elliptical curve; multimedia; computation complexity, memory consumption

I. Introduction

An image is not only about aesthetic and art but can also be a media to prove validity, for instance as a traffic violation proof. In some countries traffic violations are monitored using CCTV and fine claim is directly send to the offender's house to be paid. If the offender chooses to undergo trials rather than paying fine than judges will need the proofing images or videos. This proofing image have to be original and valid. Original means that the image hasn't been changed since first captured at CCTV, while valid means that the issuer of the image is a registered one in this case government's CCTV. In cryptography there is a method to guarantee originality and validity of digital object using encryption-decryption and digital signature.

Encryption is a process to make informative data become meaningless, makes unauthorized party unable to perceive the true meaning of the data. A lot of encryption algorithm has been developed like DES, AES, RSA, and ECC. Each has its own benefits and drawbacks. In this paper the algorithm that is going to be used is the ECC. ECC is an asymmetric encryption algorithm which is an encryption algorithm that use two different keys for encrypt and decrypt. ECC is claimed to be more powerful than RSA in term of key length correspond to the cipher strength. The problem with asymmetric key cryptography including ECC is longer time needed to encrypt and decrypt data compare to symmetric cryptography. In this paper we will try to reduce the time complexity as well as memory consumption for ECC algorithm.

II. Literatures Study

A. Elliptical Curve Cryptography

Elliptical Curve is a curve with a special attribute which makes it one of candidates for encryption and decryption. That special attribute mentioned is basically called point addition on the curve. Adding here is not adding correspond to each x and y coordinate of the two points. Adding means if we draw a straight line that pass the two points than this line will pass exactly another point on the curve and the mirrored result to x axis of this point is the result of addition. The crypto system takes advantage of ECDLP which stands for elliptical curve discrete logarithm problem to do encryption and decryption.

ECDLP is a problem where we are asked to find a value k, given a base point and another point which is the result of multiplication between base point and k. The strength of ECC is based on number of points in the ECC system or usually called as order. National institute of standard and technology released some standard curve for elliptical curve cryptography. These curves have high order which makes their cipher result being hard to decipher.

When a party wants to send encrypted data to another party, they first agree upon an elliptic curve equation, a modulus prime p which define the size of the group, and a base point P. These information is not secret, after agree upon these parameters they then choose their own private key K1 and K2 which is less than order and multiply them with the base point. The result of the multiplication will be their own public key. The complete scheme is described below

Key generation: (party A)

1. Select a random integer k_A from [1,n-1] where n is the order of the curve and k_A will be the private key of A.
2. Compute PKA = k_AP, where PKA is the public key of party A

Encryption: (party B)

1. Select random integer k_B from [1,n-1]
2. Compute PKB = k_BP such that $S_{AB} = k_B(k_AP)=(X_S,Y_S)$
3. If $x_P = 0$ (mod p) and $y_P = 0$ (mod p) then go to step 2

4. Compute C_{M1}=X$_S$M$_1$ and C_{M2}=Y$_S$M$_2$, where M$_1$ and M$_2$ is the result of to be encrypted message which is encoded to a point on elliptic curve
5. Send (B,C_{M1},C_{M2}) to party A

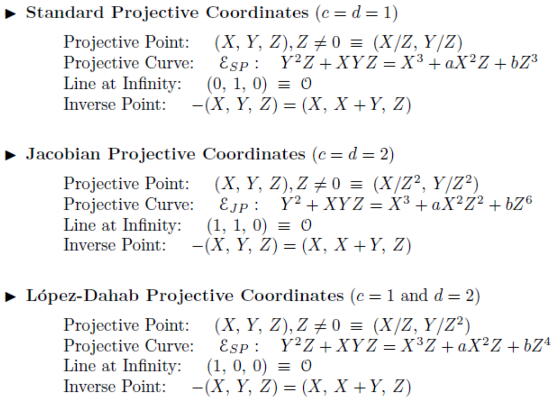
Decryption: (party A)

1. Compute $S_{AB} = k_A B = (X_S, Y_S)$
2. Find encoded point PE = (C_{M1}/X_S,C_{M2}/Y_S)
3. Decode the point back into plain message

B. Elliptical Curve Optimization

Like other asymmetric cryptography algorithm, *elliptical curve* is slower compare to symmetric cryptography algorithm. Cryptographer has developed some method to make the ECC work faster than the original ECC. These are some of these methods

1. Projective Coordinate is a method to transform affine/Cartesian coordinates to other coordinate that contains point at infinite. Here are some examples of projective coordinates where ≡ indicates the correspondence to affine coordinate.



- **Standard Projective Coordinates** ($c = d = 1$)

  Projective Point: $(X, Y, Z), Z \neq 0 \equiv (X/Z, Y/Z)$
  Projective Curve: $\mathcal{E}_{SP}: \quad Y^2Z + XYZ = X^3 + aX^2Z + bZ^3$
  Line at Infinity: $(0, 1, 0) \equiv \mathcal{O}$
  Inverse Point: $-(X, Y, Z) = (X, X + Y, Z)$

- **Jacobian Projective Coordinates** ($c = d = 2$)

  Projective Point: $(X, Y, Z), Z \neq 0 \equiv (X/Z^2, Y/Z^2)$
  Projective Curve: $\mathcal{E}_{JP}: \quad Y^2 + XYZ = X^3 + aX^2Z^2 + bZ^6$
  Line at Infinity: $(1, 1, 0) \equiv \mathcal{O}$
  Inverse Point: $-(X, Y, Z) = (X, X + Y, Z)$

- **López-Dahab Projective Coordinates** ($c = 1$ and $d = 2$)

  Projective Point: $(X, Y, Z), Z \neq 0 \equiv (X/Z, Y/Z^2)$
  Projective Curve: $\mathcal{E}_{SP}: \quad Y^2Z + XYZ = X^3Z + aX^2Z + bZ^4$
  Line at Infinity: $(1, 0, 0) \equiv \mathcal{O}$
  Inverse Point: $-(X, Y, Z) = (X, X + Y, Z)$

Addition and multiplication of points then will be done in the correspondent coordinate used.

2. Montgomery is the optimization for arithmetic points operations in elliptical curve. Montgomery use the fact that nP - (n-1)P = P where n is the multiplier and P is the point to be multiplied with. Below is the basic idea of Montgomery algorithm

Montgomery Algorithm

```
Input : Point P, Integer multiplier K = $(1,k_{l-1},k_{l-2},\dots,k_0)_2$
Output : Point Q = kP
Q←P
for i = l-1 to 0 do
    Q←2Q
    If $k_i$ = 1 then
        Q←Q+P
    end
end
return Q
```

III. Analysis and Solution

In term of encrypting an image the common way to encrypt an image using ECC is to map the RGB value of the image into a number usually large number. That number will be then encoded to a point on the elliptical curve and that point will be encrypted using ECC algorithm. This is not the efficient and effective way since every pixel with the same RGB value will be map to the same number and there are too many pixels on an image. The algorithm concept to reduce the complexity is

A. Grouping The Pixels Value

A pixel consists of channels depending on the image type. These channels are binary, grayscale, RGB, and RGBA. Binary only consist of black and white pixels, while grayscale is the gradient of black and white represented by integer value ranging from 0 to 255. RGB is a colorful image consist of red, green, and blue channel, while RGBA is just the same as RGB with additional opacity channel alpha.

Images that will be tested in this paper are RGB images. These images have three channels and each channel is represented as a byte with integer value ranging from 0 to 255. It means that each pixel will consist of three bytes of data each correspond to a channel. We can combine these tree bytes of channel data to form a representation for the pixel value, furthermore we can also combine each pixel until the maximum number value of the ECC. The maximum number value depends on how many integer points on the elliptic curve.

Suppose we have three RGB pixels with the first pixel's RGB value is 20 224 10, the second pixel's RGB value is 123 87 125, and the third pixel's RGB value is 22 11 8 respectively. Now we are going to transform these three pixels into one integer representing these three values. The representing value is $020224010123087125022011008_{255}$, since maximum value of a pixel is 255 so these representing value is in base 255 and we have to convert it to base 10. The conversion result is $3732696362276319814 88_{10}$. That value then will be converted to a point lay on the ECC curve chosen.

B. Encoding Pixels Value

After pixels are grouped together so they can be represented as single value, the value representation has to be mapped to a point on the elliptical curve which called as encoding process. Encoding process is one of the most time consuming process in ECC system. There are some methods for the encoding process but we only use one process called kolbitz encoding process with some optimization. The basic process of kolbitz encoding is

1. Pick an elliptical curve E with parameter a and b
2. The curve E has N integer points on it
3. Let us say that our number is ranging from 0 to 2^n-1 which need n bits representation
4. Now choose an auxiliary base parameter for example k = 1000
5. For each unique mk, take x=mk+1. M is the representation number of the message with k the auxiliary base parameter.

6. Substitute x to the elliptical curve equation and try to solve for y. If no y fulfils the equation, then increment x until there is y that fulfils the equation.

7. In practice, there will be y before x= mk + k-1 but if still no y then increments the value k and try to start from the sixth step above.

The step above is a time consuming process when we have to find y that fulfill the equation. We can't just use the square root to find y solution for the equation since it is modulus of prime p and we have to try y from 0 to p-1. One method to optimize this process is by using properties below

1. Choose prime modulus of the group p≡ 3 mod 4

2. y^2 = x^3+ax+b = t mod p will have solution if t^(p-1/2) mod p = 1

3. The solution for y is t^(p+1/4) mod p

C. Encryption

Encryption is done using NIST curve recommendation P-192. The encryption processes are as follows

1. Get the RGB pixel value of each pixels and grouped up to 192 bits
2. Convert the pixels to a representation number
3. Find the point mapping of the number Pm
4. Select a random number k where 1<k<order and compute kG and kPb. G is the agreed base point and Pb is the public key of the receiver.
5. Perform point addition of each Pm wih kPb resulting Pc
6. Convert each Pc to a number. If the size of the number is less than 192 bits padding 0 at the left end of that number.
7. Divide the number into groups with size of 4 bytes and convert each group to integer value.
8. Send Pair of [kG,Pc]

D. Decryption

The decryption process is basically the inverse of the encryption process. The complete processes are as follows

1. Combine the pixels up to 192 bits into a number
2. Encode that number to a point on the curve
3. Multiply kG send by the sender with receiver private key.
4. Subtract the encoded point with the multiplying result
5. Decode the result point from step 4 to a number
6. Padding that number with zero until the total bytes is 24 bytes
7. Chunk these 24 bytes into groups with size of 4 bytes each and convert each group to integer value.

IV. Implementation and Experiment

A. Implementation

Implementation is done using BufferedImage class in Java with assumption the image being tested is RGBA format type. Therefore there is no special handling for binary image or grayscale image. Key is different for every group of image even though we can make it all the same to speed up the encryption as well as decryption. The number of pixels in each group is always six pixels, based on the 192 bit curve that is used. If the image pixels is not dividable with six then additional pixels with value zero will be added. The ECC class itself is implemented in java big integer class.

There is two specific algorithm that is used here to encrypt image. The first algorithm is to combine some pixels into a group and compute the value that represent that group. The second algorithm is the inverse of the first algorithm. Here is the codes

```
public BigInteger[] convertToBigInt(int bits){
        int size=4;
        int groupMember;
/*         if(isAlpha()) size = 4;
        else size = 3;*/
        groupMember =  (bits/8)/size;
        BigInteger[] result = new
BigInteger[(int)Math.ceil((double)pixels.length/(double)groupM
ember)];
        int idx=0;
        for(int i=0;i < pixels.length;i+=groupMember){
            int j = i;
            byte[] groupPixels=new byte[1];
            Arrays.fill( groupPixels, (byte) 0 );
            do{ //combine n pixels together
                if(j<pixels.length) {
                    byte[] tempbytes =
ByteBuffer.allocate(size).putInt(pixels[j]).array();
                    groupPixels =
appendByteArray(groupPixels,tempbytes);
                    String binary = toBinary(groupPixels);
                    System.out.print("");
                }else {
                    byte[] tempbytes = new byte[4];
                    Arrays.fill( tempbytes, (byte) 0 );
                    groupPixels =
appendByteArray(groupPixels,tempbytes);
                }
                j++;
            }while(j%groupMember!=0);
            result[idx]= new BigInteger(groupPixels);
            idx++;
        }
        return result;
    }
```

Program Code 1 Combine Pixels to Group

```
public int[] convertToPixels(BigInteger[] bi, int numMember){
   int[] arrInt = new int[bi.length*numMember];
   int iterator =0;
   for(int i=0;i<bi.length;i++){
      byte[] byteArray = bi[i].toByteArray(); //each byteArray
contains numMember of pixel
      int complete = 4 - byteArray.length %4;
      byte[] addition = new byte[complete+byteArray.length];

System.arraycopy(byteArray,0,addition,complete,byteArray.lengt
h);
      for(int idx=0;idx<numMember;idx++){
         byte[] temp = new byte[4];
         System.arraycopy(addition,4*idx,temp,0,4);
         ByteBuffer wrapped = ByteBuffer.wrap(temp);
         int pixelVal = wrapped.getInt();
         System.out.println(pixelVal);
         arrInt[iterator] = pixelVal;
         iterator++;
      }
   }
   return arrInt;
}
```

Program Code 2 Array of BigInt to Pixel Value

B. *Eperiment*

Experiment is done by encrypting images with different resolution from small, medium, to high resolution. The key are also changed in respect of value an position. Aspects that is being concerned in this experiment is the time spend to encrypt and decrypt an image and also the result of the cipher image. The computer environment and specs to do the experiment are as follows

1. Intel core-i7 4770 @ 3.4 GHz
2. RAM DDR3 PC12800 8GB
3. Programming language : JDK version 8

C. *Figures and Tables*

First experiment are done to two images with different color density. The first image is an ITB logo image which has a lot of equal pixels value and the second one is a lena image which has a broad histogram value. The results are so much different. As we can see with the ITB image the encrypted image result can still be perceived with human's eyes while the lena image is somewhat randomize pixels.

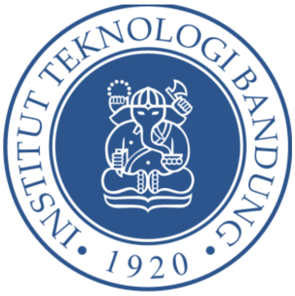


Figure 1 Plain ITB Image

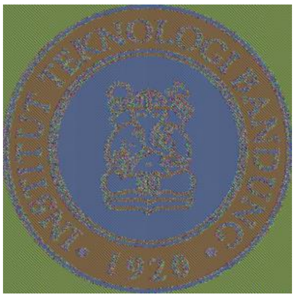


Figure 2 Encrypted ITB Image

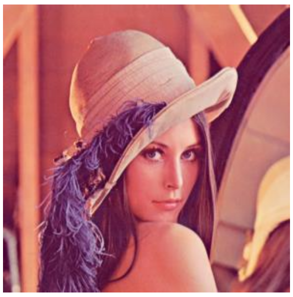


Figure 3 Lena Plain Image

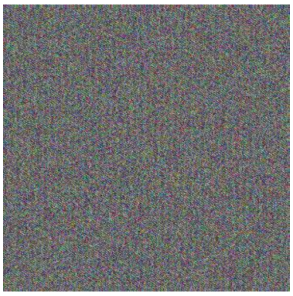


Figure 4 Lena Encrypted Image

Other tests show that colorful image with broad histogram function will result a random pixels value image like lena encrypted image. Table below will show the size and encrypted image's entropy result.

Image Name	Size	Entropy
Lena	512x512	7.823
ITB	590x590	5.221
Flowers	700x466	7.950
Sky	3000x1911	6.778

D. *Experiment Analysis and Result*

Looking at the result of entropy table above, we can see that Lena and Flowers have a high entropy value while ITB and Sky have lower entropy value. If we look at the plain image of lena and flowers they both have a wide range of histogram curve which means they have more colors variation than ITB image or Sky image, make their encryption image have higher entropy than the other two.

The time complexity and memory complexity still can't be educed in this trial because somehow the addition and multiplication point process still take a long time to be done. The time taken to encrypt and decrypt an image is over five minutes for all images tested in this paper and make this approach not good enough to be a candidate for image cryptography. Possibility to optimize the encryption process can still be done with some advanced algorithm which hasn't been implemented in this paper like using curve transformation.

The advantages of this cryptography system is to reduce the number of key to memorize. The server doesn't need to save each client's key and can save more memory space. In this paper we still have to send the complete message not only the public key and the image, since we can't decrypt the image if we only send the image and the public key. This fact is based on the fact that for an elliptical curve equation y^2=x^3+ax+b mod p it would be very difficult to find y value given x value and vice versa, since there is many possibilities of y value resulting the same modulus of p.

ACKNOWLEDGMENT

With the completion of this paper the author would like to thanks to the Almighty God and also to Mr. Rinaldi Munir who has thought the cryptography subject which the author has a lot of new ideas and knowledge about cryptography field. Also the author's parents who have support for moral and materials as well as all friends who has given ideas and time to discuss about this topic.

REFERENCES

[1] Anoop MS, Elliptic Curve Cryptography An Implementation Guide

[2] Rinaldi, Bahan Kuliah IF4020 Kriptografi: Elliptical Curve Cryptography

[3] Ravi Kishore Kodali, ECC Implementation using Kolbitz's method

[4] Kefa Rabah, Elliptical Curve Elgamal Encryption and Signature Schemes

COPYRIGHT STATEMENT

I hereby declare that this paper is my own writing, no adaptation, or translation of others' paper, and not a plagiarism.

Bandung 19 May 2016

Ramandika Pranamulia