Implementation of Elliptical Curve Cryptography on MATLAB.

Enrollment Number : 101016

Name of Student : Vishal Bansal

Name of Supervisor : Mr. Akhil Ranjan

May 2014

Project report submitted in partial fulfillment of the degree of

Bachelor of Technology

In

Electronics & Communication Engineering

Certificate

This is to certify that project report entitled "Implementation of Elliptical Curve Cryptography on MATLAB.", submitted by Vishal Bansal in partial fulfillment for the award of degree of Bachelor of Technology in Electronics and Communication Engineering to Jaypee University of Information Technology, Waknaghat, Solan has been carried out under my supervision.

This work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.

Date: 26th May 2014 **Supervisor's Name:-Mr Akhil Ranjan**

Designation:- Assistant Professor(Grade-II)

Acknowledgement

First of all, we would like to thank our project guide Mr. Akhil Ranjan whose assistance has been invaluable to us. Also we would like to thank the faculty of Jaypee University of Information Technology as they did not hesitate to help us whenever we were in need. Also we would like to thank our parents for their blessings.

Date: 26thMay 2014

Name of the student

Vishal Bansal

TABLE OF CONTENTS

Certificate…………………………………………………………………………………i

Acknowledgement……………………………………………………………………..ii

List of Figures…………………………………………………………………………iii

List of Tables…………………………………………………………………………..iv

1. Objective...1

2. Methodology………………………………………………………………………..1

3. Introduction…………………………………………………………………………1

3.1. Cryptography……………………………………………………………...2

3.2. Network Security………………………………………………………….3

3.2.1. Network Security Concepts……………………………………......3

3.3. Elliptic Curve…………………………………………………………...…4

3.4. Elliptic Curve Cryptography……………………………………………....6

3.5. Finite Field …………………………………………………………..…7

3.6. Types Of Cryptography…………………………………………...………8

3.6.1. Symmetric Key Cryptography……………………………………,8

3.6.2. Public Key Cryptography……………………………………….....8

3.7. Diffie Hellman Key Exchange Algorithm ………………………………..9

3.8. RSA…………………………………………………………………….....10

3.8.1. Decryption in RSA………………………………………………...10

3.8.2. Encryption in RSA………………………………………………...10

3.9. El-Gamal Cryptosystem…………………………………………………..11

3.10. Mathematics in Elliptic Curve Cryptography……………………...………12

3.11. Elliptic Curve Diffie-Hellman Algorithm……………………….………15

3.11.1. Key Establishment Protocol………………………..…………15

3.12. Encryption and Decryption Algorithm Proposed...18

4. Result………………………………………………………………...………..19

4.1 Result for elliptic curve for finite Field F_{239}..19

4.2 Result for Another curve for finite Field F_{503}..21

4.3 Result for encryption and decryption...22
5 References…………………………………………………………………..……29

List of Figures

1. Elliptic Curve…..4

2. Finite field……………………………….…………………………………..…...6

3. Diffie-Hellman Figure……………………………….……………………….......8

4. RSA Block Diagram……………………………………….……………….......9

5. El-Gamal Block Diagram…………………………………………………….……10

6. Point Addition Graph……………………………………………………………11

7. Point Doubling Graph……………………………………………………….....12

8. Formulae………………………………………………………………………13

9. ECDH Block Diagram…………………………………………………………...15

10. Elliptic curve…………………………………………………………………....18

11. Finite field……………………………………………………………………18

12. Elliptic curve..19

13. Finite Field...19

List of Tables

1. Key Size Comparison……………………………………………………………5

1. OBJECTIVE:

Elliptic curve public key cryptosystems have evolved in the recent past as an important alternative to established systems like RSA. They offer more security than a RSA key at a much lesser key size and thus they are a better option. This project shows an implementation of ECC and the subsequent encryption and decryption of a text message.

2. METHODOLOGY:

The code that is implemented in MATLAB consists of plotting the elliptic curve, plotting the finite field elliptic curve and generation of private and public key using Diffie-Hellman Key Exchange Algorithm and the encryption and decryption of the message

.

3. INTRODUCTION

In the modern technological era, when most of the real world problems and solutions are done using programming and computing, there is a need of securing the information. From the ATM cards to electronic commerce, from electronic mails to online transaction, every process must be secured. The method that allows us to maintain the privacy of our data is cryptography. The present project is about FPGA based implementation of Elliptic Curve Coprocessor using synthesizable VHDL code. Digital communication networks, like the Internet, are incorporated today in many applications that require secure connections. Channels in those networks are often public, i.e. eavesdropping or altering of the transmitted data cannot be prevented. Cryptosystems ensure the detection of altered data and prevent the extraction of information from eavesdropped data by unauthorized parties. In order to make security requiring applications economically feasible, it is necessary to implement efficient cryptosystems. Modern cryptosystems fall in one of two categories: symmetric and asymmetric or public key cryptosystems. In symmetric cryptosystems a single key is used for both, the encryption and the decryption process. This implies that the key must be known by both communicating parties and thus must already have been transmitted through some secure channel. All communication must be planned beforehand. Spontaneous secure communication, which is necessary in many applications (for example online shopping) is not possible. In public key cryptosystems, as first proposed in, each participant has two keys, one of which is the public key and one the private key. While the private key is known only by its owner, his public key is published in a dictionary and thus accessible by everyone. When person A wants to send sensitive data to person B, she encrypts the data with **B**'s public key, which can be found in the dictionary. The data can only be decrypted with B's private key, which is only known by person B. Today's cipher chips for symmetric systems deliver a data throughput in the Gb/s range, whereas current implementations of asymmetric systems are an order of magnitude slower. The strengths of both systems are usually combined in that a key for the symmetric system is encrypted with the asymmetric cryptosystem. After the key has been exchanged the actual communication is symmetrically encrypted. In order to increase the security, the symmetric key is usually exchanged frequently during a communication session. This implies a need for an efficient implementation of the applied public key cryptosystem. Taher ElGamal proposed in a public key cryptosystem, which is based on the *Diffie-Hellman Key Exchange* and thus relies on the difficulty to compute discrete logarithms. Neil Koblitz and Victor Miller independently developed elliptic curve based analogs of the ElGamal system. Nowadays elliptic curve based cryptosystems have emerged as an important alternative to the well known RSA system and are applied in many applications. The most expensive operation applied in elliptic curve based cryptosystems is the "multiplication" of a large natural number with a point on an elliptic curve. The coprocessor presented in this paper is an elliptic curve multiplier that speeds up the multiplication operation significantly in comparison to a purely software based implementation. We first give a brief introduction to the mathematics involved in elliptic curve cryptography. Now to explain a few terms given in the article above: -

3.1 CRYPTOGRAPHY

Cryptography is derived from a Greek word '*kruptō*' meaning hidden secret and '*graphein*' meaning writing. Cryptography is analogous to encryption of a message. In this process the given data or plaintext is encrypted to form an encrypted message or ciphertext, which is accessible only to the one who has the cipher or the key to decrypt the message. Modern cryptography makes use of extensive mathematical tools and computer science.

3.2 NETWORK SECURITY

Network security consists of the provisions and policies adopted by a network administrator to prevent and monitor unauthorized access, misuse, modification, or denial of a computer network and network-accessible resources. Network security involves the authorization of access to data in a network, which is controlled by the network administrator. Users choose or are assigned an ID and password or other authenticating information that allows them access to information and programs within their authority. Network security covers a variety of computer networks, both public and private, that are used in everyday jobs conducting transactions and communications among businesses, government agencies and individuals. Networks can be private, such as within a company, and others which might be open to public access. Network security is involved in organizations, enterprises, and other types of institutions. It does as its title explains: It secures the network, as well as protecting and overseeing operations being done. The most common and simple way of protecting a network resource is by assigning it a unique name and a corresponding password.

3.2.1 NETWORK SECURITY CONCEPTS

Network security starts with authenticating, commonly with a username and a password. Since this requires just one detail authenticating the user name —i.e. the password— this is sometimes termed one-factor authentication. With two-factor authentication, something the user 'has' is also used (e.g. a security token or 'dongle', an ATM card, or a mobile phone); and with three-factor authentication, something the user 'is' is also used (e.g. a fingerprint or retinal scan).

Once authenticated, a firewall enforces access policies such as what services are allowed to be accessed by the network users. Though effective to prevent unauthorized access, this component may fail to check potentially harmful content such as computer worms or Trojans being transmitted over the network. Anti-virus software or an intrusion prevention system (IPS) help detect and inhibit the action of such malware. An anomaly-based intrusion detection system may also monitor the network and traffic for network may be logged for audit purposes and for later high-level analysis.

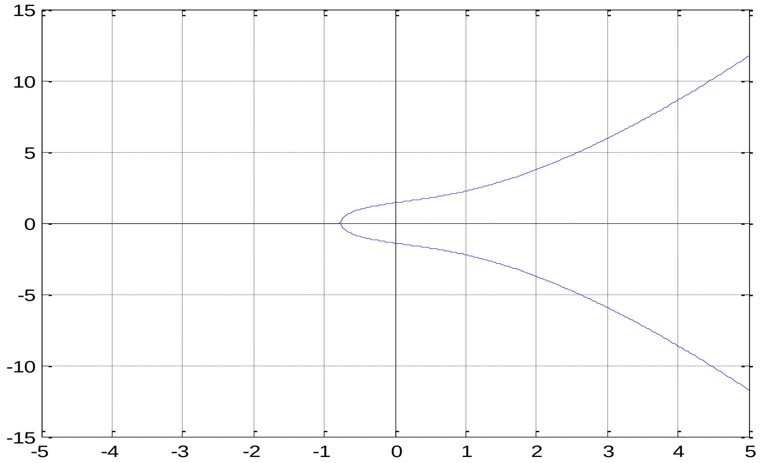
3.3 ELLIPTIC CURVE

In mathematics, an elliptic curve (EC) is a smooth, projective algebraic curve of genus one, on which there is a specified point O. An elliptic curve is in fact an abelian variety – that is, it has a multiplication defined algebraically, with respect to which it is a (necessarily commutative) group – and O serves as the identity element. Often the curve itself, without O specified, is called an elliptic curve.

Any elliptic curve can be written as a plane algebraic curve defined by an equation of the form:

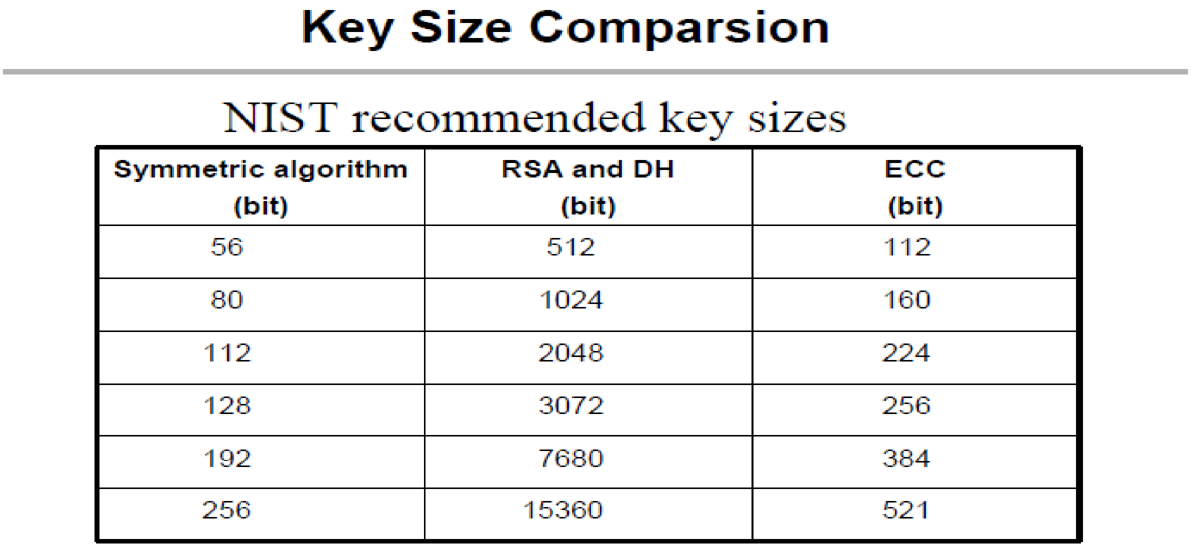
$$\mathbf{y^2=x^3+2x+2}$$

which is non-singular; that is, its graph has no cusps or self-intersections. (When the characteristic of the coefficient field is equal to 2 or 3, the above equation is not quite general enough to comprise all non-singular cubic curves; see below for a more precise definition.) The point O is actually the "point at infinity" in the projective plane. If y2 = P(x), where P is any polynomial of degree three in x with no repeated roots, then we obtain a nonsingular plane curve of genus one, which is thus also an elliptic curve. If P has degree four and is squarefree this equation again describes a plane curve of genus one; however, it has no natural choice of identity element. More generally, any algebraic curve of genus one, for example from the intersection of two quadric surfaces embedded in three-dimensional projective space, is called an elliptic curve, provided that it has at least one rational point. Using the theory of elliptic functions, it can be shown that elliptic curves defined over the complex numbers correspond to embeddings of the torus into the complex projective plane. The torus is also an abelian group, and in fact this correspondence is also a group isomorphism. Elliptic curves are especially important in number theory, and constitute a major area of current research; for example, they were used in the proof, by Andrew Wiles (assisted by Richard Taylor), of Fermat's Last Theorem. They also find applications in elliptic curve cryptography (ECC) and integer factorization.



Elliptic curve obtained in MATLAB of eq. $y^2=x^3+2x+2$

3.4 ELLIPIC CURVE CRYPTOGRAPHY



Key Size Comparsion

NIST recommended key sizes

Symmetric algorithm (bit)	RSA and DH (bit)	ECC (bit)
56	512	112
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	521

Public-key cryptography is based on the intractability of certain mathematical problems. Early public-key systems are secure assuming that it is difficult to factor a large integer composed of two or more large prime factors. For elliptic-curve-based protocols, it is assumed that finding the discrete logarithm of a random elliptic curve element with respect to a publicly known base point is infeasible –this is the "elliptic curve discrete logarithm problem" or ECDLP. The entire security of ECC depends on the ability to compute a point multiplication and the inability to compute the multiplicand given the original and product points. The size of the elliptic curve determines the difficulty of the problem.

The primary benefit promised by ECC is a smaller key size, reducing storage and transmission requirements, i.e. that an elliptic curve group could provide the same level of security afforded by an RSA-based system with a large modulus and correspondingly larger key – e.g., a 256-bit ECC public key should provide comparable security to a 3072-bit RSA public key.

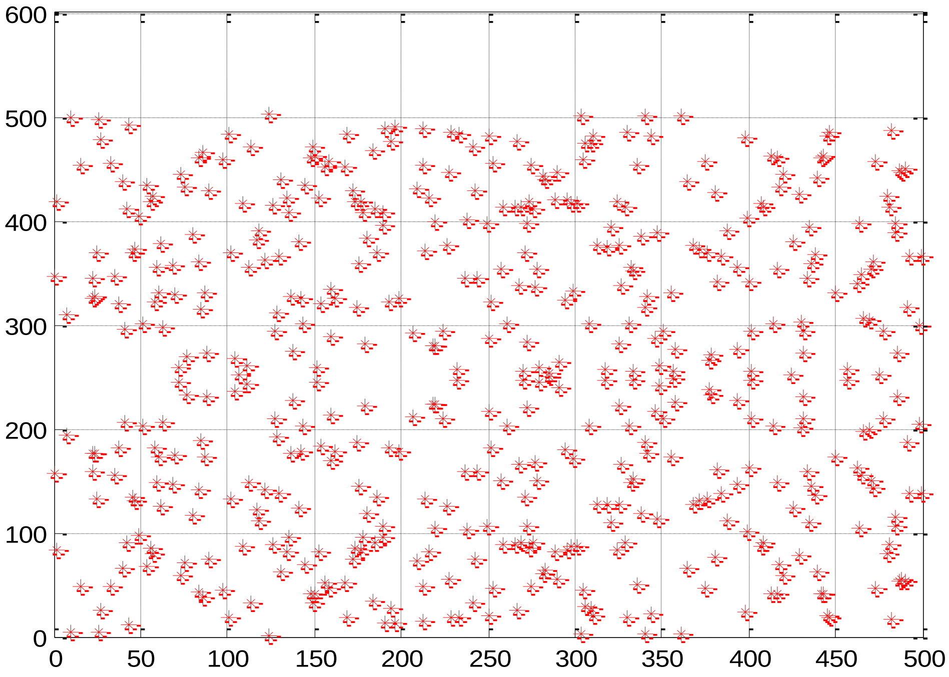
3.5 FINITE FIELD

In abstract algebra, a finite field or Galois field (so named in honor of Évariste Galois) is a field that contains a finite number of elements. Finite fields are important in number theory, algebraic geometry, Galois theory, cryptography, coding theory and quantum error correction. The finite fields are classified by size; there is exactly one finite field up to isomorphism of size p^k for each prime p and positive integer k. Each finite field of size q is the splitting field of the polynomial $x^q - x$, and thus the fixed field of the Frobenius endomorphism which takes x to x^q. Similarly, the multiplicative group of the field is a cyclic group.

The finite fields are classified as follows

- The order, or number of elements, of a finite field is of the form p^n, where p is a prime number called the characteristic of the field, and n is a positive integer.
- For every prime number p and positive integer n, there exists a finite field with p^n elements.
- Every element is Finite field follows the Additive inverse identity and the Multiplicative inverse identity

In our project we have utilized a finite field to compute the elements of our elliptic curve.



The Finite field 503 of Elliptic curve $y^2=x^3+2x+2$

3.6 TYPES OF CRYPTOGRAPHY

There are two types of cryptography:

3.6.1 SYMMETRIC KEY CRYPTOGRAPHY

- Symmetric-key cryptography refers to encryption methods in which both the sender and receiver share the same key (or, less commonly, in which their keys are different, but related in an easily computable way). This was the only kind of encryption publicly known until June 1976.
- Symmetric key ciphers are implemented as block ciphers . A block cipher enciphers input in blocks of plaintext as opposed to individual characters, the input form used by a stream cipher.
- Symmetric-key cryptosystems use the same key for encryption and decryption of a message, though a message or group of messages may have a different key than others.

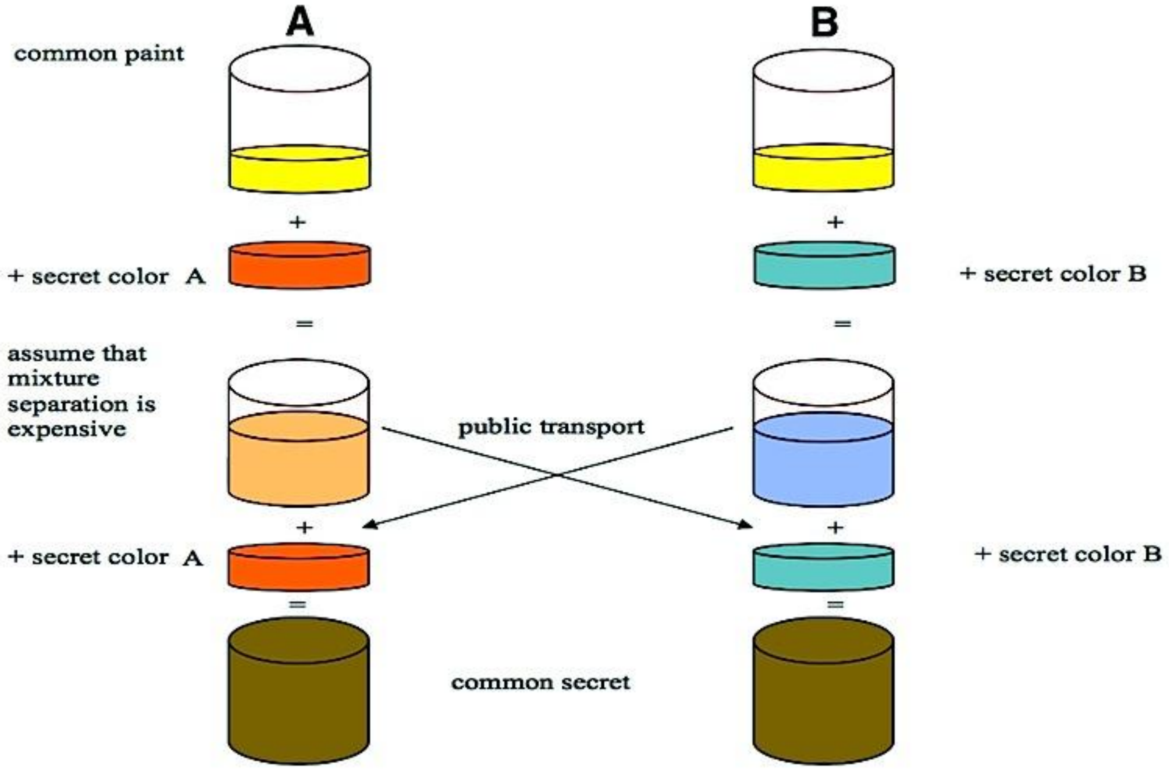
3.6.2 PUBLIC KEY CRYPTOGRAPHY

- A significant disadvantage of symmetric ciphers is the key management necessary to use them securely. Each distinct pair of communicating parties must, ideally, share a different key, and perhaps each ciphertext exchanged as well.
- The number of keys required increases as the square of the number of network members, which very quickly requires complex key management schemes to keep them all straight and secret.
- The difficulty of securely establishing a secret key between two communicating parties, when a secure channel does not already exist between them, also presents a chicken-and-egg problem which is a considerable practical obstacle for cryptography users in the real world.
- In public-key cryptosystems, the public key may be freely distributed, while its paired private key must remain secret. In a public-key encryption system, the *public key* is used for encryption, while the *private* or *secret key* is used for decryption.
- While Diffie and Hellman could not find such a system, they showed that public-key cryptography was indeed possible by presenting the Diffie–Hellman key exchange protocol, a solution that is now widely used in secure

communications to allow two parties to secretly agree on a shared encryption key

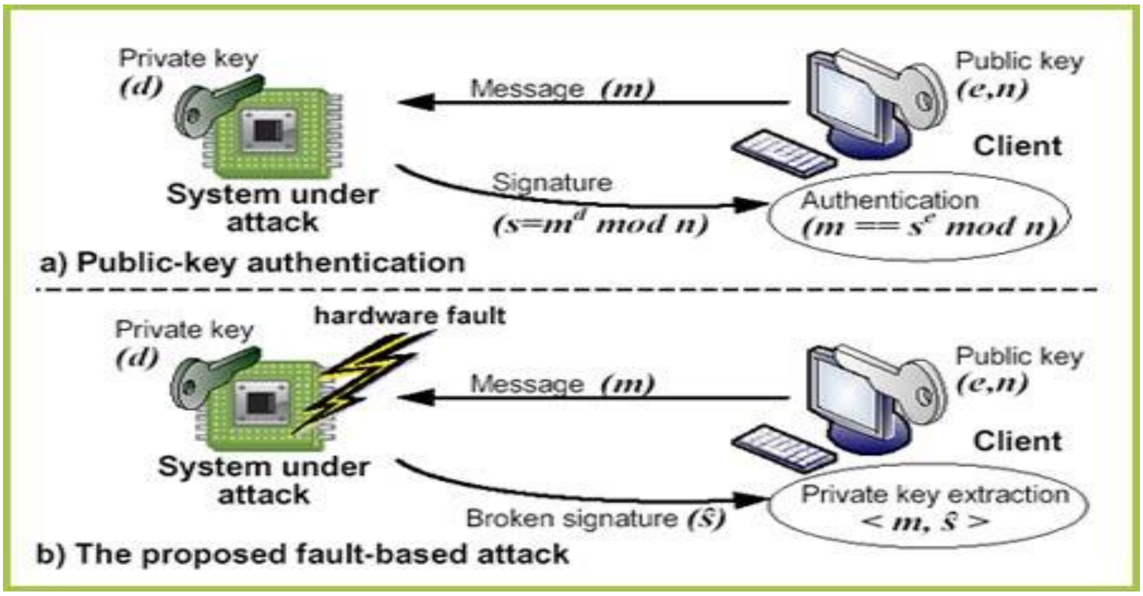
3.7 DIFFIE HELLMAN KEY EXCHANGE ALGORITHM

Diffie–Hellman key exchange (D–H)[nb 1] is a specific method of exchanging cryptographic keys. It is one of the earliest practical examples of key exchange implemented within the field of cryptography. The Diffie–Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher. The Diffie Hellman Key Exchange algorithm can also be explained using the diagram on the next page using the concept of colors.



3.8 RSA

RSA is a cryptosystem, which is known as one of the first practicable public-key cryptosystems and is yet widely used for secure data transmission. In such a cryptosystem, the encryption key is public and differs from the decryption key which is kept secret. In RSA, this asymmetry is based on the practical difficulty of factoring the product of two large prime numbers, the factoring problem. RSA stands for Ron Rivest, Adi Shamir and Leonard Adleman, who first publicly described the algorithm in 1977.



3.8.1 ENCRYPTION in RSA

Alice transmits her public key (n, e) to Bob and keeps the private key secret. Bob then wishes to send message M to Alice.

He first turns M into an integer m, such that $0 \leq m < n$ by using an agreed-upon reversible protocol known as a padding scheme. He then computes the ciphertext c corresponding to

$$c \equiv m^e \pmod{n}.$$

This can be done quickly using the method of exponentiation by squaring. Bob then transmits c to Alice.

Note that at least nine values of m will yield a ciphertext c equal to m,[6] but this is very unlikely to occur in practice.

3.8.2 DECRYPTION in RSA

Alice can recover *m* from *c* by using her private key exponent *d* via computing

$$m \equiv c^d \pmod{n}.$$

Given *m*, she can recover the original message *M* by reversing the padding scheme.

3.9 EL-GAMAL CRYPTOSYSTEM

Users of the general system are assumed to know that G is cyclic, and that a certain specified element g ∈ G is a generator. It is also assumed that they express plaintext and ciphertext messages as elements of G in some standard way. In the general El-Gamal cryptosystem, each user, such as Bob, chooses a private key b∈ N, and computes his public key b ∈ G by the rule $b = g^b$

When Alice wishes to send Bob a message m ∈ G she chooses a token t ∈ N, and applies the encryption function

$Eb(m, t) = (g^t,m \ast b^t)$.

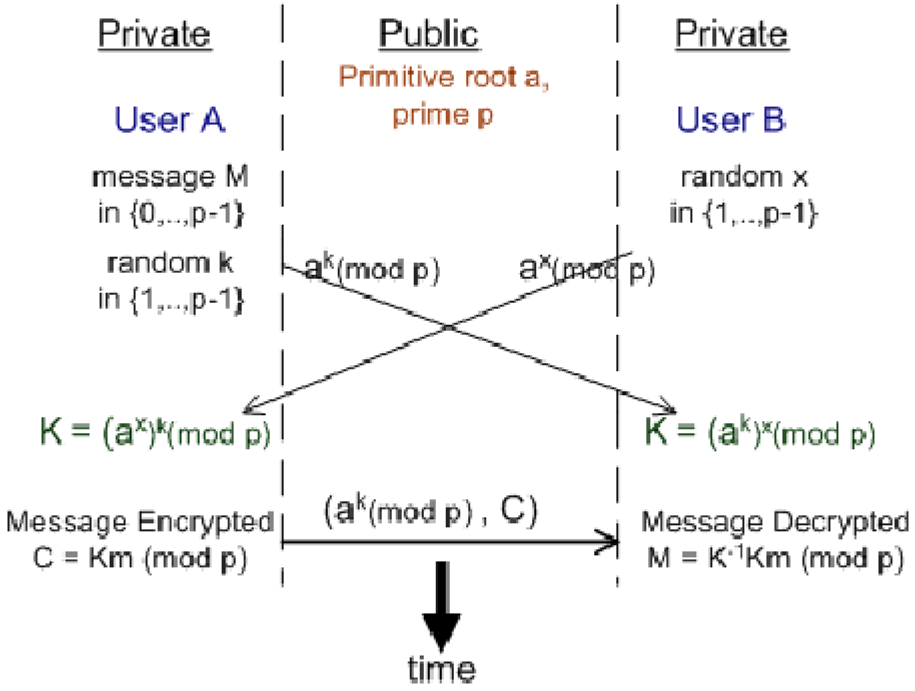
Bob receives ciphertext in two parts: the first part is the ‘leader’ $l = g^t$, and the other part is the encrypted message $c = m \ast b^t$. Bob’s decryption function is

$Db\ (l, c) = c \ast (l^{-1})^{\ b}$.

Using essentially the same algebra as in Lemma 14.7, it is easy to check that

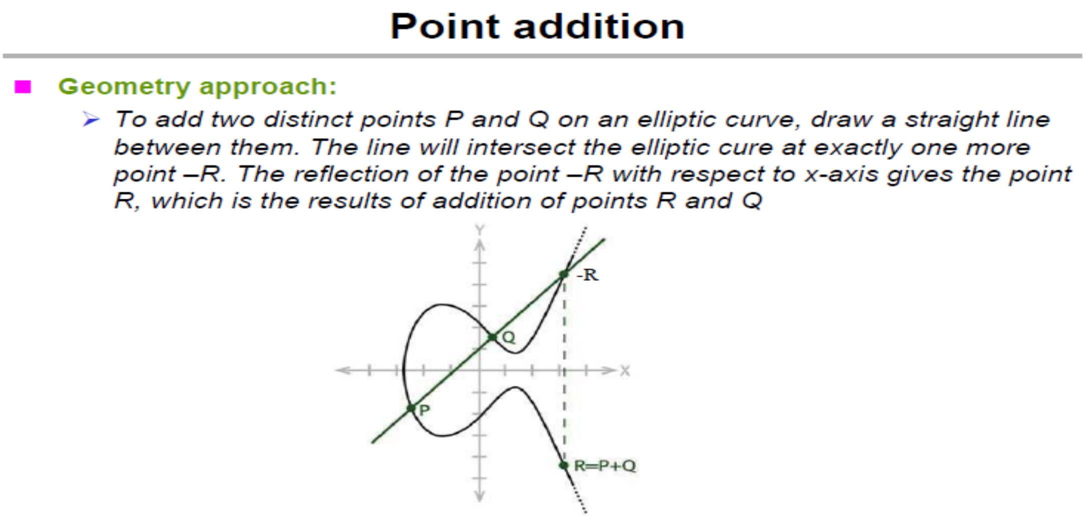
Db (Eb(m, t)) = m for all m ∈ G, and all t ∈ N:

$$\begin{aligned} Db\ (Eb(m, t)) = Db(g^t,m \ast b^t) &= (m\ast b^t) \ast ((g^t)-1)^b \\ &= m\ast b^t \ast ((g^b)^{-1})^t \\ &= m\ast b^t \ast (b-1)^{\ t} \\ &= m. \end{aligned}$$



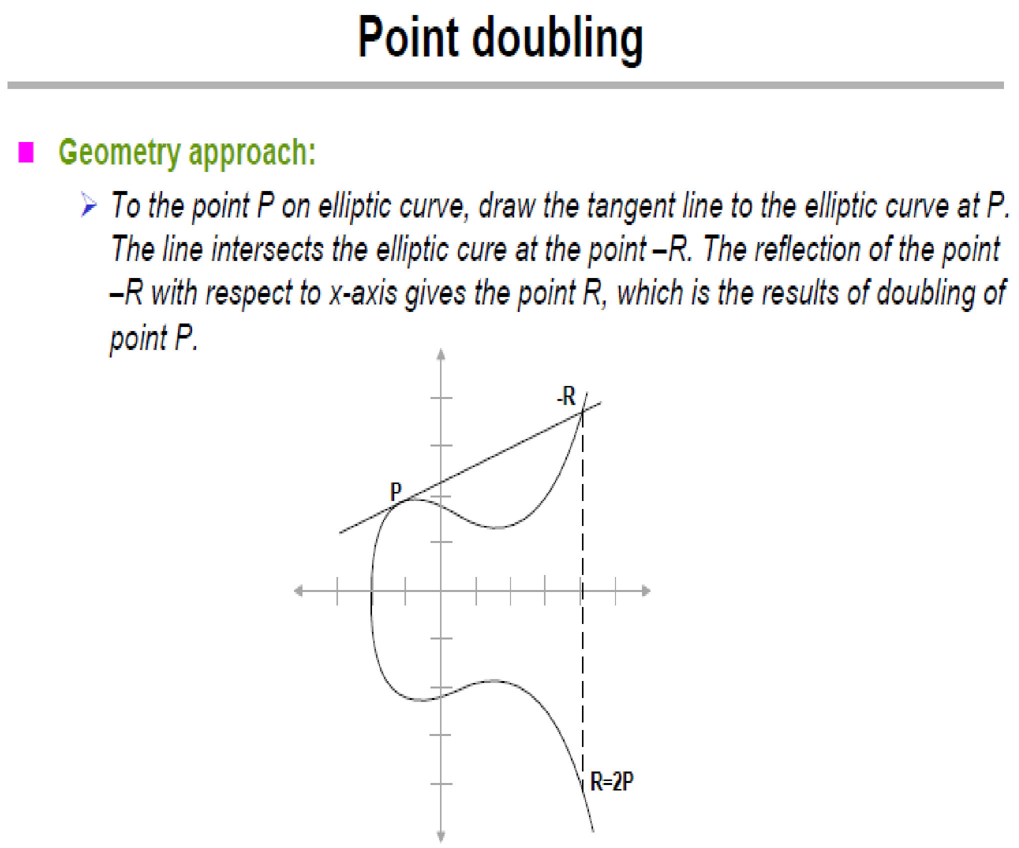
3.10 MATHEMATICS in ELLIPTIC CURVE CRYPTOGRAPHY

We basically utilize point addition, point doubling and point multiplication in Elliptic curve cryptosystems to calculate public and private key. This is because the operation of exponentiation is equivalent to the process of point multiplication.



Point addition

- Geometry approach:
  - *To add two distinct points P and Q on an elliptic curve, draw a straight line between them. The line will intersect the elliptic cure at exactly one more point –R. The reflection of the point –R with respect to x-axis gives the point R, which is the results of addition of points R and Q*



Point doubling

- Geometry approach:
  - *To the point P on elliptic curve, draw the tangent line to the elliptic curve at P. The line intersects the elliptic cure at the point –R. The reflection of the point –R with respect to x-axis gives the point R, which is the results of doubling of point P.*

Utilizing both point addition and point doubling we can do the operation of point multiplication.

For example:-

If private key k = 17

Therefore to calculate Q = k*P

Given the value of P

Q = 2*(2*(2*(2*P)))) + P

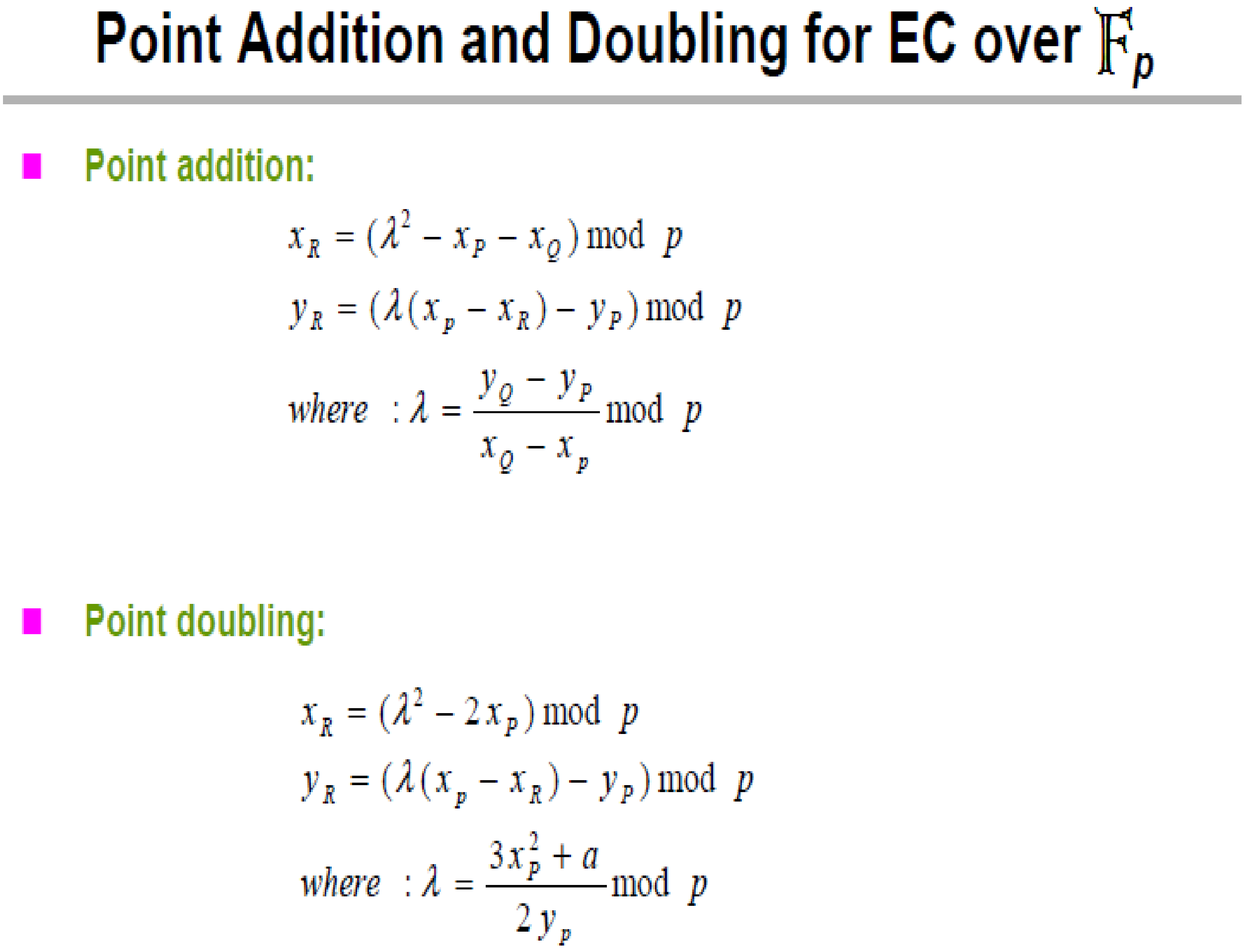
Hence we first doubled the point P four times and then we added the point to the result to obtain the final result.

The approach utilized is called the double and add approach.

This required only 5 operations rather than 16 operations if we had only added the points.

This is how we do point multiplication in Elliptic curve cryptosystems

The formulae used for the operations addition and doubling are given below:-



Point Addition and Doubling for EC over $\mathbb{F}_p$

- **Point addition:**

$$x_R = (\lambda^2 - x_P - x_Q) \bmod p$$
$$y_R = (\lambda(x_p - x_R) - y_P) \bmod p$$
$$where\ : \lambda = \frac{y_Q - y_P}{x_Q - x_p} \bmod p$$

- **Point doubling:**

$$x_R = (\lambda^2 - 2x_P) \bmod p$$
$$y_R = (\lambda(x_p - x_R) - y_P) \bmod p$$
$$where\ : \lambda = \frac{3x_P^2 + a}{2y_p} \bmod p$$

3.11 ELLIPTIC CURVE DIFFIE-HELLMAN ALGORITHM

Elliptic curve Diffie–Hellman (ECDH) is an anonymous key agreement protocol that allows two parties, each having an elliptic curve public–private key pair, to establish a shared secret over an insecure channel.[1][2][3] This shared secret may be directly used as a key, or better yet, to derive another key which can then be used to encrypt subsequent communications using a symmetric key cipher. It is a variant of the Diffie–Hellman protocol using elliptic curve cryptography.

3.11.1 Key establishment protocol

Suppose Alice wants to establish a shared key with Bob, but the only channel available for them may be eavesdropped by a third party. Initially, the domain parameters (that is, (p,a,b,G,n,h) in the prime case or (m,f(x),a,b,G,n,h) in the binary case) must be agreed upon. Also, each party must have a key pair suitable for elliptic curve cryptography, consisting of a private key d (a randomly selected integer in the interval [1, n-1]) and a public key Q (where Q = d G, that is, the result of adding G together d times). Let Alice's key pair be (d_A, Q_A) and Bob's key pair be (d_B, Q_B). Each party must have the other party's public key (an exchange must occur).

Alice computes (x_k, y_k) = d_A Q_B. Bob computes (x_k, y_k) = d_B Q_A. The shared secret is x_k (the x coordinate of the point). Most standardized protocols based on ECDH derived a symmetric key from x_k using some hash-based key derivation function.

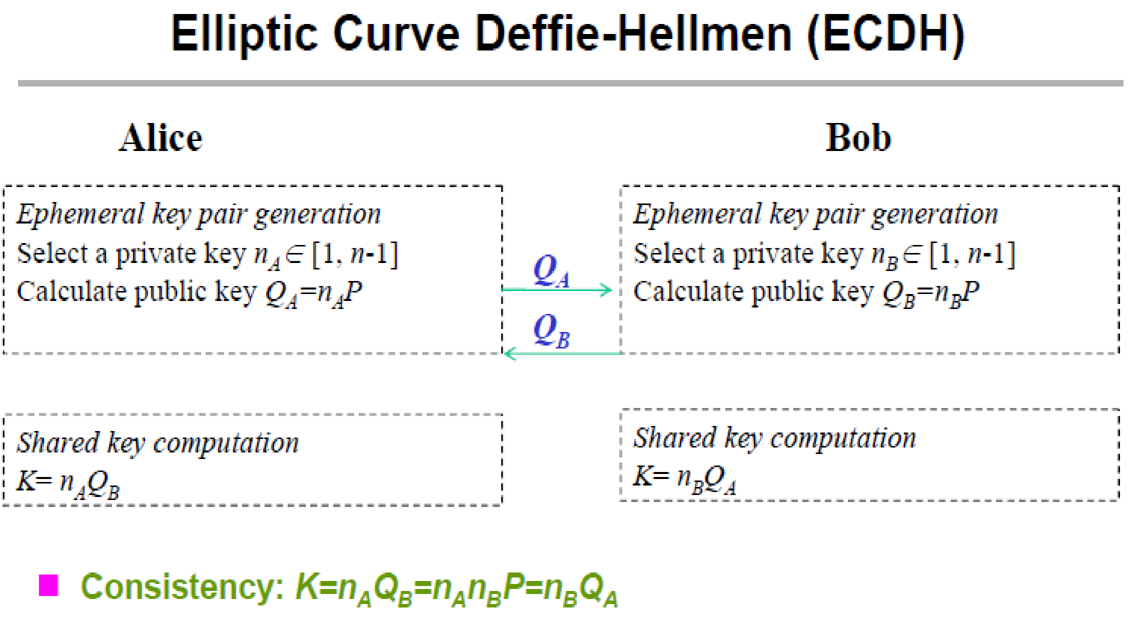
The shared secret calculated by both parties is equal, because d_A Q_B = d_A d_B G = d_B d_A G = d_B Q_A.

The only information about her private key that Alice initially exposes is her public key. So, no party other than Alice can determine Alice's private key, unless that party can solve the elliptic curve discrete logarithm problem. Bob's private key is similarly secure. No party other than Alice or Bob can compute the shared secret, unless that party can solve the elliptic curve Diffie–Hellman problem.

The public keys are either static (and trusted, say via a certificate) or ephemeral. Ephemeral keys are temporary and not necessarily authenticated, so if authentication is desired, authenticity assurances must be obtained by other means. Authentication is necessary to avoid man-in-the-middle attacks. If one of Alice or Bob's public key is static then man-in-the-middle attacks are thwarted. Static public keys provide neither forward secrecy nor key-compromise impersonation resilience, among other advanced security properties. Holders of static private keys should validate the other public key, and should apply a secure key derivation function to the raw Diffie–Hellman shared secret to avoid leaking information about the static private key. For schemes with other security properties, see ECMQV and FHMQV.

While the shared secret may be used directly as a key, it is often desirable to hash the secret to remove weak bits due to the Diffie–Hellman exchange.

In our project we have implemented the ECDH algorithm.



Elliptic Curve Deffie-Hellmen (ECDH)

Alice

Ephemeral key pair generation
Select a private key $n_A \in [1, n\text{-}1]$
Calculate public key $Q_A = n_A P$

Bob

Ephemeral key pair generation
Select a private key $n_B \in [1, n\text{-}1]$
Calculate public key $Q_B = n_B P$

Shared key computation
$K = n_A Q_B$

Shared key computation
$K = n_B Q_A$

- Consistency: $K = n_A Q_B = n_A n_B P = n_B Q_A$

An Example

The generator p is selected by both Alice and Bob

P =

480 80

Now Alice selects her private key Na

```
Na =
   147
```

After selecting her private key she performs point multiplication to get Qa

```
Qa =
   310   482
```

Similarly Bob selects his own private key Nb

```
Nb =
   320
```

Then he performs Point multiplication to get Qb

```
Qb =
   129   192
```

Now Alice receives Qb from Bob and multiplies it with her private key to obtain the shared private key

```
K1 =
   276    91
```

Similarly Bob receives Qa from Alice and multiplies it with his private key to obtain the shared private key

```
K2 =
   276    91
```

It’s observed that K1=K2, thus successfully completing the algorithm.

3.12 Encryption and Decryption Algorithm Proposed

We propose this algorithm as it has multiple layers of security

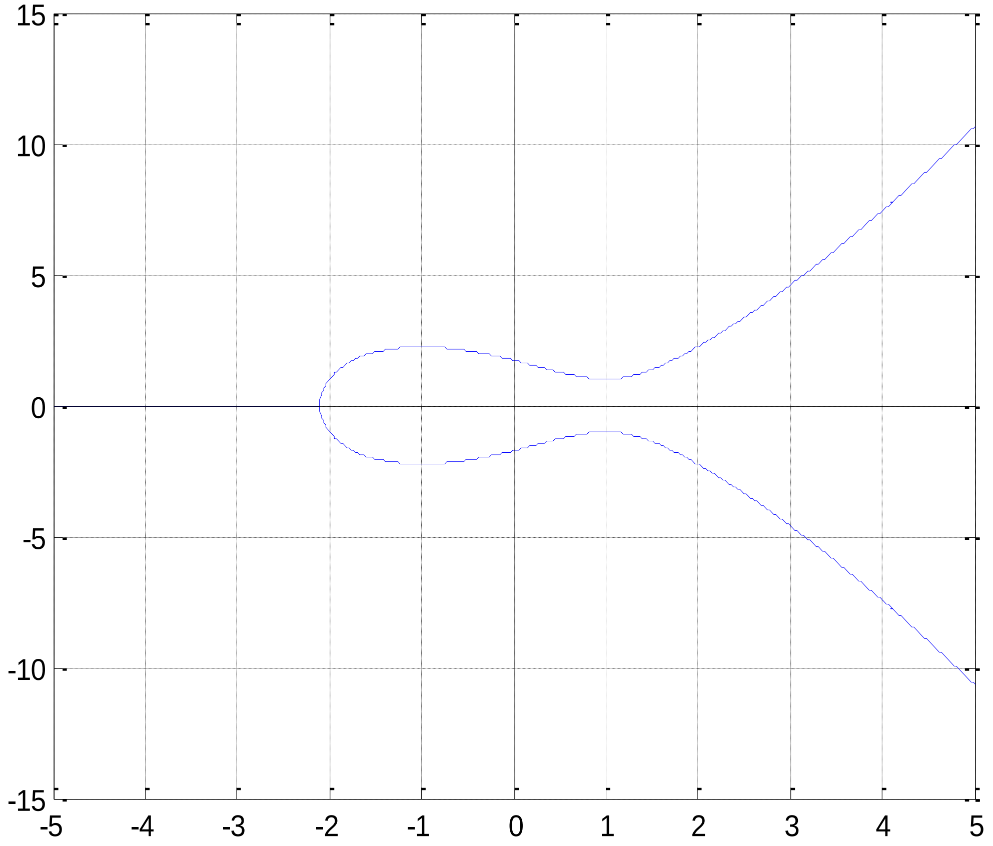
1. First the each no. is assigned a point on the elliptic curve.
2. After that we form a matrix of base 3 numbers each denoting a particular point on the curve.
3. Each element is circular shifted to add security.
4. Now, the person who's supposed to receive the message chooses a random no. 'a' and publishes the point 'aP' where 'P' is publically known.
5. Now the sender selects a no. 'l' and computes

   $P_1 = l*P$

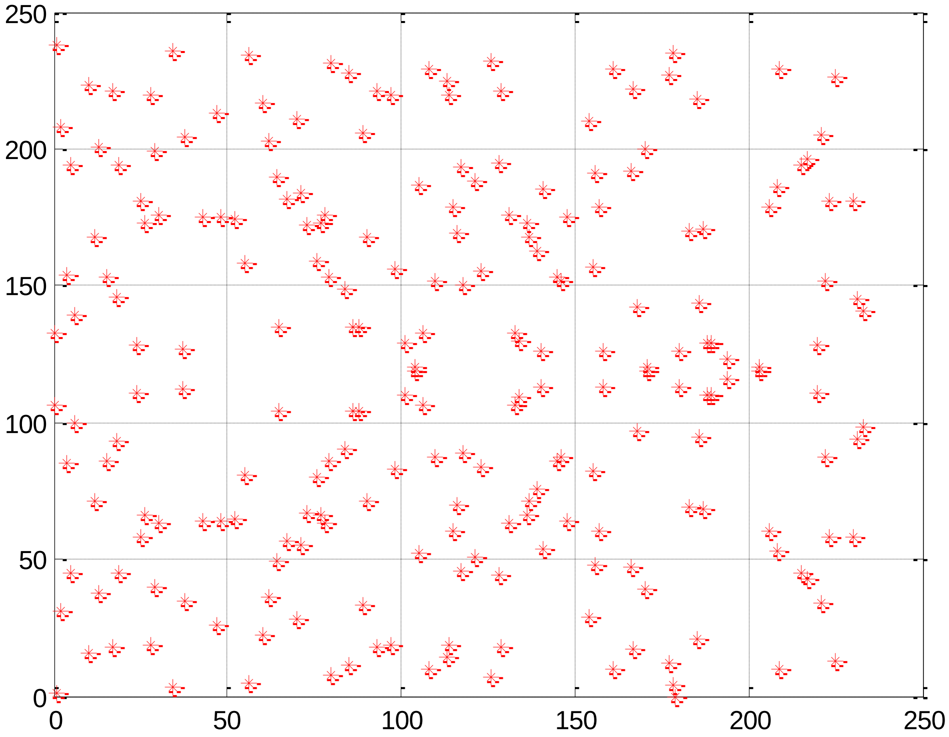
   $P_2 = P_i + l*(a*P)$
6. Now the numbers are encoded into binary as 0->00,1->01 and 2->10
7. The receiver first decodes the message into base 3 numbers by 00->0,01->1 and 10->2
8. Then he circular shifts these numbers.
9. The numbers should now be converted into decimal and stored in 'k' variable.
10. Now (k+1)*P is obtained which is equal to P_1.
11. Similarly P_2 is obtained.
12. The receiver now multiplies P_1 into 'a' and then subtracts it from P2 to obtain the message.

Chapter 4

4.1 RESULTS FOR ELLIPTIC CURVE FOR FINITE FIELD F_{239}



Elliptic curve obtained in MATLAB of eq. $y^2=x^3-3x+3$



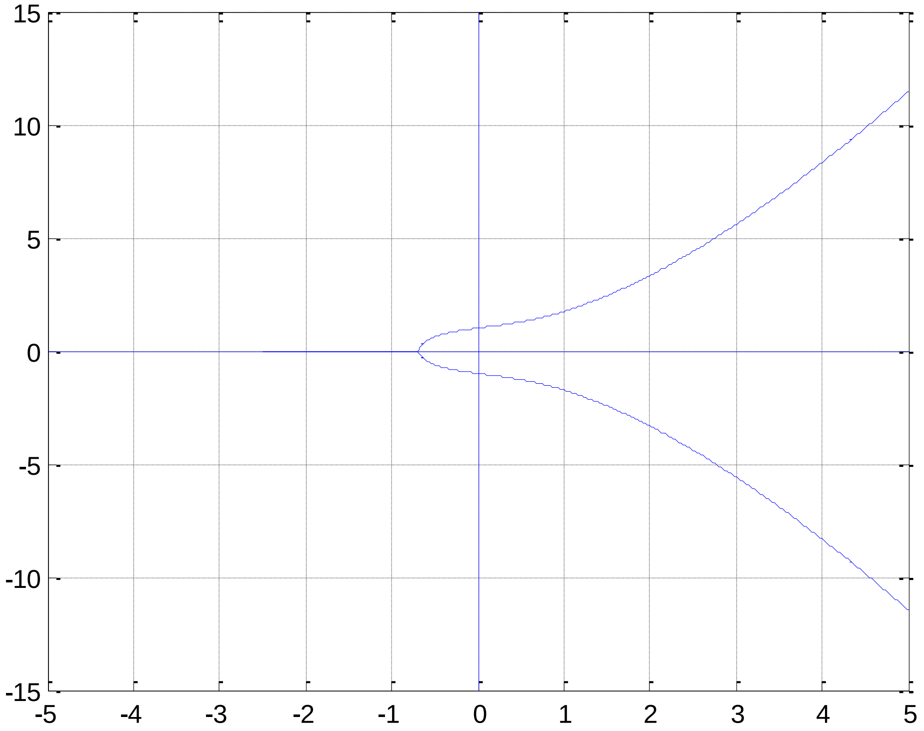
Finite Field F_{239} Graph Of Elliptic Curve

```
p =

   215   194

Qa =

   126   232

na =

   165

Qb =

   133   133

nb =

    59
```

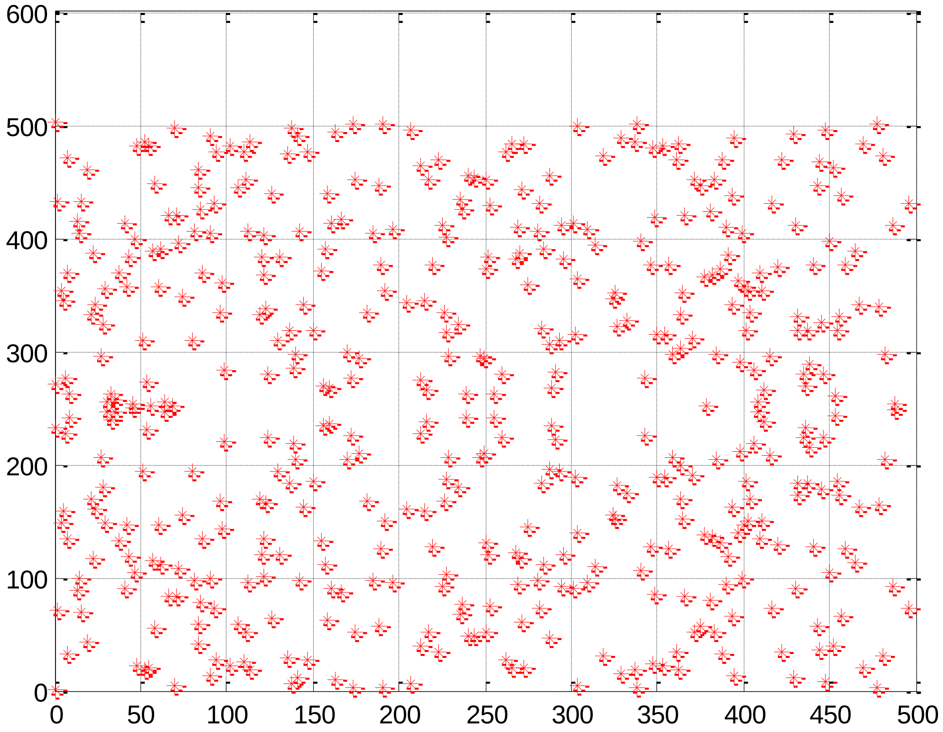
The shared Private key is

```
K1 =

    62    36

K2 =

    62    36
```

4.2 RESULTS FOR ANOTHER CURVE FOR FINITE FIELD F_{503}



Equation for Elliptic curve $y^2 = x^3 + x + 1$



Finite Field F_{503} of elliptic curve given above

```
p =

   456   173

Qa =

   293   310

na =

   147

Qb =

   236    68

nb =

   320

The shared Private key is

K1 =

   353    22

K2 =

   353    22
```

4.3 Results for Encryption and decryption

The matrix generator element

```
p =

     5     7
```

The matrix generated

```
x =

     5     7
    28     4
    18     1
    22    12
     6    20
    13     5
     2    14
    21    11
    23     3
    10     7
    14    22
    16    23
     7    27
     1     4
     0     4
     0    25
     1    25
     7     2
    16     6
    14     7
    10    22
    23    26
    21    18
     2    15
    13    24
```

6	9
22	17
18	28
28	25
5	22

The matrix after encoding in base 3

M =

0	0	0	0
0	0	0	1
0	0	0	2
0	0	1	0
0	0	1	1
0	0	1	2
0	0	2	0
0	0	2	1
0	0	2	2
0	1	0	0
0	1	0	1
0	1	0	2
0	1	1	0
0	1	1	1
0	1	1	2
0	1	2	0
0	1	2	1
0	1	2	2
0	2	0	0
0	2	0	1
0	2	0	2
0	2	1	0
0	2	1	1
0	2	1	2

```
     0     2     2     0
     0     2     2     1
     0     2     2     2
     1     0     0     0
     1     0     0     1
     1     0     0     2
```

The matrix after left circular shift

```
M =
     0     0     0     0
     1     0     0     0
     2     0     0     0
     0     0     0     1
     1     0     0     1
     2     0     0     1
     0     0     0     2
     1     0     0     2
     2     0     0     2
     0     0     1     0
     1     0     1     0
     2     0     1     0
     0     0     1     1
     1     0     1     1
     2     0     1     1
     0     0     1     2
     1     0     1     2
     2     0     1     2
     0     0     2     0
     1     0     2     0
     2     0     2     0
     0     0     2     1
     1     0     2     1
     2     0     2     1

     0     0     2     2

     1     0     2     2

     2     0     2     2

     0     1     0     0

     1     1     0     0

     2     1     0     0

Message to be sent:-'this is fun'

inp =

this is fun

Encrypted Message

ans =

  Columns 1 through 27

     0     0     1     1     0     0     2     1     0     0     1     1     0     0     1     0     0     0     1
1     1     0     1     0     0     0     1

  Columns 28 through 54

     1     2     0     2     0     0     0     1     1     1     1     0     0     0     0     1     1     1     0
1     0     0     0     1     1     2     0

  Columns 55 through 81

     2     0     0     0     1     1     1     1     0     0     0     0     1     1     1     0     0     2     0
0     1     1     1     0     2     1     0

  Columns 82 through 88

     0     1     1     0     0     1     2

Encryped message after binary Conversion

ans =

  Columns 1 through 27

     0     0     0     0     0     1     0     1     0     0     0     0     1     0     0     1     0     0     0
0     0     1     0     1     0     0     0

  Columns 28 through 54

     0     0     1     0     0     0     0     0     0     0     1     0     1     0     1     0     0     0     1
0     0     0     0     0     0     0     1
```

```
  Columns 55 through 81

     0     1     1     0     0     0     1     0     0     0     0     0     0     0     0     1     0     1     0
1     0     1     0     0     0     0     0

  Columns 82 through 108

     0     0     0     0     1     0     1     0     1     0     0     0     1     0     0     0     0     0     0
0     1     0     1     1     0     0     0

  Columns 109 through 135

     1     0     0     0     0     0     0     0     0     1     0     1     0     1     0     1     0     0     0
0     0     0     0     0     0     1     0

  Columns 136 through 162

     1     0     1     0     0     0     0     1     0     0     0     0     0     0     1     0     1     0     1
0     0     1     0     0     1     0     0

  Columns 163 through 176

     0     0     0     1     0     1     0     0     0     0     0     1     1     0

Received message

ans =

  Columns 1 through 27

     0     0     0     0     0     1     0     1     0     0     0     0     1     0     0     1     0     0     0
0     0     1     0     1     0     0     0

  Columns 28 through 54

     0     0     1     0     0     0     0     0     0     0     1     0     1     0     1     0     0     0     1
0     0     0     0     0     0     0     1

  Columns 55 through 81

     0     1     1     0     0     0     1     0     0     0     0     0     0     0     0     1     0     1     0
1     0     1     0     0     0     0     0

  Columns 82 through 108

     0     0     0     0     1     0     1     0     1     0     0     0     1     0     0     0     0     0     0
0     1     0     1     1     0     0     0

  Columns 109 through 135

     1     0     0     0     0     0     0     0     0     1     0     1     0     1     0     1     0     0     0
0     0     0     0     0     0     1     0
```

```
  Columns 136 through 162

    1    0    1    0    0    0    0    1    0    0    0    0    0    0    1    0    1    0    1
0    0    1    0    0    1    0    0

  Columns 163 through 176

    0    0    0    1    0    1    0    0    0    0    0    1    1    0

Received message after Conversion

ans =

  Columns 1 through 27

    0    0    1    1    0    0    2    1    0    0    1    1    0    0    1    0    0    0    1
1    1    0    1    0    0    0    1

  Columns 28 through 54

    1    2    0    2    0    0    0    1    1    1    1    0    0    0    0    1    1    1    0
1    0    0    0    1    1    2    0

  Columns 55 through 81

    2    0    0    0    1    1    1    1    0    0    0    0    1    1    1    0    0    2    0
0    1    1    1    0    2    1    0

  Columns 82 through 88

    0    1    1    0    0    1    2

The sent message decrypted

frecms =

this is fun
```

REFERENCES

1. Michael Jung. FPGA Based Implementation Of An Elliptic Curve Coprocessor Utilizing Synthesizable VHDL code, IEEE conference,(2009)
2. Elaine Brow .Elliptic Curve Cryptography , IEEE conference, (December 2010)
3. Martin Leslie. Elliptic Curve Cryptography, Advanced Combinatorics,(June 5, 2006)
4. Fuwen Liu. A Tutorial on Elliptic Curve Cryptography(ECC), Brandenburg Technical University of Cottbus, Computer Networking Group
5. I. Blake, G. Seroussi, N. Smart. Elliptic Curves in Cryptography. Cambridge University Press (1999).
6. I. Blake, G. Seroussi, N. Smart. Advances in Elliptic Curve Cryptography. Cambridge University Press (2005).
7. D.M. Gordon. A survey of fast exponentiation algorithms. J. Algorithms 27 (1998) 129-146.
8. D.E. Knuth. The Art of Computer Programming II - Semi-numerical Algorithms. Addison-Wesley (third edition, 1997).
9. J.H. Silverman. The Arithmetic of Elliptic Curves. Springer-Verlag (1986).
10. L.C. Washington. Elliptic Curves: Number Theory and Cryptography. Chapman and Hall / CRC Press (2003)