EFFICIENT APPROACH OF FINDING CURVE ORDER

SUBMITTED BY

ZEESHAN MEHBOOB

EXAMINATION ROLL NUMBER: M4SWE19006
REGISTRATION NUMBER: 140964 of 2017-2018

A THESIS SUBMITTED TO
THE FACULTY OF ENGINEERING & TECHNOLOGY OF JADAVPUR UNIVERSITY IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF ENGINEERING
IN
SOFTWARE ENGINEERING

UNDER THE SUPERVISION OF

MR. UTPAL KUMAR RAY
ASSISTANT PROFESSOR

DEPARTMENT OF INFORMATION TECHNOLOGY
JADAVPUR UNIVERSITY

2019

Department of Information Technology
Faculty of Engineering & Technology

Jadavpur University

Certificate of Submission

*I hereby recommend that the thesis, entitled **"Efficient Approach of finding Curve Order"**, prepared by **Zeeshan Mehboob** (Registration Number:140964 of 2017-2018) under my supervision, be accepted in partial fulfilment of the requirements for the degree of **Master of Engineering** in **Software Engineering** from the **Department of Information Technology** under **Jadavpur University**.*

__________________________________-
Mr. Utpal Kumar Ray,
Assistant Professor,
Department of Information Technology,
Jadavpur University

Countersigned by:

__________________________________-
Head of the Department,
Information Technology,
Jadavpur University

__________________________________-
Dean,
Faculty of Engineering and Technology,
Jadavpur University.

JADAVPUR UNIVERSITY

DEPARTMENT OF INFORMATION TECHNOLOGY
FACULTY OF ENGINEERING AND TECHNOLOGY

CERTIFICATE OF APPROVAL

The thesis at instance is hereby approved as a creditable study of an Engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned does not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve this thesis for the purpose for which it is submitted.

——————————————-	——————————————-
Signature of External Examiner	Signature of the Supervisor
Prof. Dr. Saswat Chakrabarti	Mr. Utpal Kumar Ray
G. S. Sanyal School of Tele-Comm,	Assistant Professor
IIT Kharagpur, PIN - 721302,	Department of Information Technology
West Bengal	Jadavpur University
E-Mail : saswat@ece.iitkgp.ac.in	

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that this thesis contains literature survey and original research work by me, as a part of my Master of Engineering in Software Engineering studies.

All information in this document have been obtained and presented in accordance with academic rules and ethical conduct.
I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name: ZEESHAN MEHBOOB
Roll Number: M4SWE19006
Thesis Title: EFFICIENT APPROACH OF FINDING CURVE ORDER

__________________________________-

Signature (with date)

ACKNOWLEDGEMENTS

I would like to express my heartfelt gratitude to Prof. Utpal Kumar Ray, Assistant Professor, Faculty, Department of Information Technology, Jadavpur University, for the help, cooperation and guidance that he has bestowed during the course of this thesis work. I am highly indebted to him for providing a mentally stimulating environment and for all those hours of enjoyable discussions which helped me to complete my thesis work.

My sincere obligation goes to Dr. Bhaskar Sardar, Head of The Department, Department of Information Technology, Jadavpur University for his constant support.

I would also like to thank the staffs of the Department of Information Technology, Jadavpur University, and my friends for their help.

(ZEESHAN MEHBOOB)
M.E. in Software Engineering
JADAVPUR UNIVERSITY

ABSTRACT

Elliptic Curve Cryptography (ECC) has existed since the mid-1980s, but it is still looked on as the newcomer in the world of SSL, and has only begun to gain adoption in the past few years. ECC is a fundamentally different mathematical approach to encryption than the venerable RSA algorithm. An elliptic curve is an algebraic function ($y^2 = x^3 + ax + b$) which looks like a symmetrical curve parallel to the x axis when plotted. As with other forms of public key cryptography, ECC is based on a one-way property in which it is easy to perform a calculation but infeasible to reverse or invert the results of the calculation to find the original numbers. This elliptic curve has cardinality (curve order) by which we can calculate the point order of a point on the curve, this point order later uses in the cryptographic function.

This thesis work focuses upon the detailed description of finding out the curve order of the elliptic curve using a much more efficient way using Schoof's algorithm.

The implementation of Schoof's algorithms for finding the curve order of elliptic curve in 'C' is done by incorporating a special library called FLINT & MPFR Library which provides some features that gives faster processing for big integers. This library also includes the GMP library which supports the operation on big integers.

Keywords: Cryptography, Elliptic Curves, Schoof's Algorithm, Cardinality, Hasse's Theorem, Division Polynomials.

CONTENTS

LIST OF FIGURES

1 INTRODUCTION

1.1 Motivation

Nowadays data security is the most important of our life. Securely transferring the data over the internet is necessity. Because many people are eavesdropping or literally hack the network to check your personal messages, confidential file, bank details etc. For all of these cryptography came into play. Cryptography gives our data integrity, confidentiality, non-repudiation & authentication. Cryptography mainly is of two types: Symmetric Key Cryptography & Asymmetric Key Cryptography. Symmetric key cryptography used only single key for encryption & decryption, their algorithm for encryption & decryption is just the reversed operation based on shifting, substitution, and permutation. On the other hand asymmetric key cryptography is a complex mathematical process by which the encryption and decryption is done, there are two keys private key & public key for each person. Public key is used to encrypt and Private Key is used to decrypt. Asymmetric key cryptography is more secure than Symmetric key cryptography but the computational time factor comes into play, being uses of mathematical operation asymmetric use more time than symmetric.

There is where our Elliptic curve cryptography (type of asymmetric key cryptography) comes into play. In 1985, ***Victor Miller*** and ***Neal Kobiltz*** proposed this cryptosystem based on elliptic curves, whose security relies on ECDLP (Elliptic curve discrete logarithmic problem). Now, Elliptic curve cryptography (ECC) can be applied to data-encryption and decryption, digital signatures, and key exchange procedures. ECC provides more security with less computational time and small key length. According to ***NIST*** (National Institute of Standards and Technology) Elliptic curve uses less key bit over RSA (***Rivest-Shamir-Adelman***) for providing same level of security.

Now using Elliptic curve as an encryption system we have to find the curve order. There are two approaches to do that Naïve Approach, slow one merely a brute force type algorithm, and Advance Approach (using Schoof's Algorithm) which is an efficient approach to calculate the point on elliptic curve. Schoof's Algorithm is a deterministic polynomial time algorithm helps to find out the curve order to judge the difficulty of solving the discrete logarithm problem in the group of points on an elliptic curve.

1.2 Focus

The aim of this project is to gain knowledge of the implementation of Schoof's Algorithm using the GMP, FLINT & MPFR library. Schoof's Algorithm itself is a complex algorithm in terms of implementation. Schoof's approach to computing the cardinality $\# \mathrm{E}(\mathbb{F}_q)$ makes use of Hasse's theorem on elliptic curves along with the Chinese remainder theorem and division polynomials.

The mathematics involves divisor theory on elliptic curves and consists of number of theoretical and algebraic aspects. Many articles have appeared, often treating (part of) the subject from a different point of view, making different assumptions and using a different notation. We try to give an overview of currently known theory.

In order to deal with Schoof’s algorithm, we describe the Elliptic curve arithmetic. Then we briefly describe the application on Elliptic curve, comparison between RSA and Elliptic curve, Naïve approach to find the curve order. After that we discuss the Schoof’s Algorithm and its utility to reduce the time to finding the curve order.

1.3 Organization

The outline of the thesis is as follows:

In Chapter 2, we state introductory part of the Elliptic curve. Then we will discover the arithmetic approaches associated with Elliptic curve. After that we will discuss about the application of Elliptic curve in cryptography. Then we will discuss over RSA and Elliptic curve’s comparison. Why finding curve order is necessary?

In Chapter 3, we first state what we mean by finding curve order. Then we give a brief idea how to get the curve order and its different approaches. Then we describe we describe the Schoof’s Algorithm and give some analysis.

In Chapter 4, we give some result and conclusion based on our thesis. How effect the Schoof’s Algorithm is?

2

ELLIPTIC CURVE

2.1 Introduction

An Elliptic curve is a curve given by equation of the form

$$\boxed{y^2 = x^3 + ax + b}$$

There is also a requirement that the discriminant $\Delta = 4A^3 + 27B^2$ is nonzero. Equivalently, the polynomial $x^3 + ax + b$ has distinct roots. This ensures that the curve is non-singular, i.e., its graph has no cusps or self-intersections. For reasons to be explained later, an extra point, $\mathcal{O}$, has been tossed in that is "at infinity", so $\mathrm{E}(a, b)$ is the set

$$\mathrm{E} = \{(x, \quad y) : y^2 = x^3 + ax + b\} \cup \{\mathcal{O}\}$$

It is a curve that's also naturally a group and group law is constructed geometrically. Elliptic curves have nothing to do with ellipses and it appears in many diverse areas of mathematics, ranging from number theory to complex analysis, and from cryptography to mathematical physics. Elliptic curves can have points with coordinates in any field, such as$\mathbb{F}_p, \mathbb{Q}, \mathbb{R}$ or $\mathbb{C}$. Elliptic curves with points in $\mathbb{F}_p$ are finite groups.

Elliptic curves have purely algebraic properties which are quite remarkable too. Most importantly, one can easily define an operation on the points of an elliptic curve that turns the whole curve into an abelian group. One can use geometry to make the points of an elliptic curve turn into a group. We can see and example below where we add two point using geometry only.

If $y^2 = P(x)$, where P is any polynomial of degree three in x with no repeated roots, the solution set is a non-singular plane curve of genus one, an elliptic curve. If P has degree four and is square-free 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 to act as the identity.

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, of Fermat's Last Theorem. They also find applications in elliptic curve cryptography (ECC) and integer factorization.

Let K denote a field and $\overline{K}$ its algebraic closure. Throughout the report, we write K^* for $K/\{0\}$.

Consider the homogeneous function over the projective plane $\mathbb{P}^2(\overline{K})$ given by

$$F(X, Y, Z) = Y^2Z + a_1XYZ + a_3YZ^2 - X^3 - a_2X^2Z - a_4XZ^2 - a_6Z^3,$$

where $a_1, a_2, a_3, a_4, a_6 \in \overline{K}$. An elliptic curve E is defined to be the set of solutions to $F(X, Y, Z) = 0$ in the projective plane $\mathbb{P}^2(\overline{K})$, where points are equivalent to their multiples, i.e. $(X : Y : Z) \sim (\lambda X : \lambda Y : \lambda Z)\, for\, \lambda \in K$. There is exactly one point in E with $Z = 0$, namely $(0 : 1 : 0)$, which we will call the point at infinity and denote by $\mathcal{O}$. By convention, we require the curve to be non-singular, i.e. for all points $P \in E$ the three partial derivatives $\frac{\partial F}{\partial X}$, $\frac{\partial F}{\partial Y}$ and $\frac{\partial F}{\partial Z}$ are not all zero at P. For simplicity, we will use affine coordinates $x = X/Z$ and $y = Y/Z$, and denote an elliptic curve by the (affine) ***Weierstrass equation***

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6.$$

Then the elliptic curve is the set of points $(x,y) \in \mathbb{A}^2(\overline{K}) = \overline{K} \times \overline{K}$ that satisfy the above equation, together with the extra point at infinity $\mathcal{O}$. E is said to be defined over K, denoted E/K, if $a_1, a_2, a_3, a_4, a_6 \in K$. Then K is called the definition field. We denote by $E(K)$ the set of K- rational points, i.e. the points with both coordinates in K, together with $\mathcal{O}$. Observe that by definition, we can write $E = E(\overline{K})$.

Two elliptic curves $E_1/K, E_2/K$ are isomorphic over K, denoted $E_1/K \cong E_2/K$, if there exist $u, r, s, t \in K,\ u \neq 0$, such that the admissible change of variables

$$(x,y) \rightarrow (u^2x + r, \qquad u^3y + u^2sx + t)$$

transforms the equation of E_1 into the equation of E_2. We will generally be working with elliptic curves over a finite field $K = \mathbb{F}_q$ consisting of q elements, where $q = p^m$ is a prime power. In this case, the algebraic closure of K is given by $\overline{K} = \bigcup_{i \geq 1} \mathbb{F}_{q^i}$. If the characteristic p of K is greater than 3, then any curve defined over K is isomorphic with a curve of particularly simple form, namely:

$$E : y^2 = x^3 + ax + b, \qquad a, b \in K$$

where $4a^3 + 27b^2 \neq 0$. Apart from the Weierstrass elliptic curve, there are various other forms of elliptic curves which are used for cryptographic purposes. Montgomery curve is a form of elliptic curve, different from the usual Weierstrass form, introduced by Peter L. Montgomery in 1987. A Montgomery curve equation over a finite field F is denoted by

$$By^2 = x^3 + Ax^2 + x$$

where $A, B \in F$ and with $B(A^2 - 4) \neq 0$. Another form of elliptic curve is that of twisted Edwards's curves. The equation for twisted Edwards's curves over a finite field $\mathbb{F}$ is given as

$$ax^2 + y^2 = 1 + dx^2 + y^2$$

where a and d are distinct non-zero elements of $\mathbb{F}$. However, in our paper, we will mostly be stressing on Weierstrass elliptic curves for further discussions.

An elliptic curve over real numbers may be defined as the set of points (x, y) which satisfy an elliptic curve equation of the form:

$$\boxed{y^2 = x^3 + ax + b, \qquad where\ x, y, a\ \&\ b\ are\ real\ numbers}$$

Each choice of the numbers $a\ \&\ b$ yields a different elliptic curve. For example, $a = -4\ \&\ b = 0.67$ gives the elliptic curve with equation $y^2 = x^3 - 4x + 0.7$; the graph of this curve is shown below:

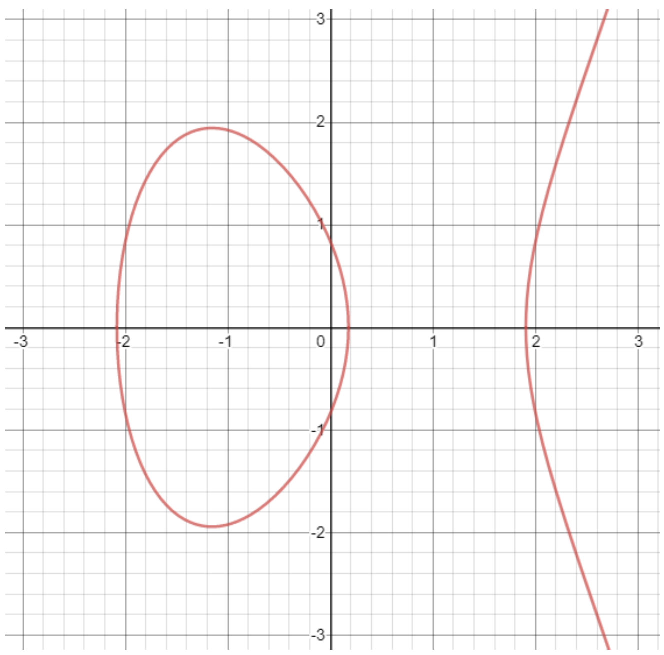

Fig 2.1.1: Elliptic curve where a = -4 and b = 0.7

If $x^3 + ax + b$ contains no repeated factors, or equivalently if $4a^3 + 27b^2$ is not 0, then the elliptic curve $y^2 = x^3 + ax + b$ can be used to form a group. An elliptic curve group over real numbers consists of the points on the corresponding elliptic curve, together with a special point $\mathcal{O}$ called the point at infinity.

The finite field $\mathbb{F}_p$ uses the numbers from 0 to p– 1, and computations end by taking the remainder on division by p. For example, in $\mathbb{F}_{23}$ the field is composed of integers from 0 to 22, and any operation within this field will result in an integer also between 0 and 22.

An elliptic curve with the underlying field of $\mathbb{F}_p$ can be formed by choosing the variables a and b within the field of $\mathbb{F}_p$. The elliptic curve includes all points (x, y) which satisfy the elliptic curve equation modulo p (where x and y are numbers in $\mathbb{F}_p$). For Elliptic Curves over $\mathbb{F}_p$, we use a form of the Weierstrass Equation:

$$y^2 \equiv (x^3 + ax + b) \, mod \, p$$

If $x^3 + ax + b$ contains no repeating factors (or, equivalently, if $\{(4a^3 + 27b^2) \, mod \, p\} \neq 0$), then the elliptic curve can be used to form a group. There are finitely many points on such an elliptic curve. It should be noted the seemingly random spread of points for the elliptic curve over $\mathbb{F}_p$.

<u>EXAMPLE OF AN ELLIPTIC CURVE GROUP OVER</u> $\mathbb{F}_p$

As a very small example, an elliptic curve over the field $\mathbb{F}_{61}$ has been considered. With $a = -1$ and $b = 0$, the elliptic curve equation is:

$$y^2 = x^3 - x$$

.

The point $(9,7)$ satisfies this equation since:

$$y^2 \equiv (x^3 - x) \, mod \, p$$

$=> \; 49 \, mod \, 23 = (729 - 9) \, mod \, 23$

$=> \; 49 \, mod \, 23 = 720 \, mod \, 23$

$=> \; 0 = 0$

The 71 points which satisfy this equation are: (0, 0), (1, 0), (4, 11), (4, 50), (6, 24), (6, 37), (8, 4), (8, 57), (9, 7), (9, 54), (10, 21), (10, 40), (11, 10), (11, 51), (13, 7), (13, 54), (14, 31), (14, 32), (15, 26), (15, 35), (17, 4), (17, 57), (18, 18), (18, 43), (22, 16), (22, 45), (23, 26), (23, 35), (24, 21), (24, 40), (25, 17), (25, 44), (27, 21), (27, 40), (28, 5), (28, 56), (33, 6), (33, 55), (34, 13), (34, 48), (36, 4), (36, 57), (37, 13), (37, 48), (38, 19), (38, 42), (39, 7), (39, 54), (43,15), (43, 46), (44, 17), (44, 44), (46, 19), (46, 42), (47, 25), (47, 36), (48, 16), (48, 45), (50, 12), (50, 49), (51,13), (51, 48), (52, 16), (52, 45), (53, 17), (53, 44), (55, 20), (55, 41), (57, 1), (57, 60), (60, 0). These points may be graphed as below:

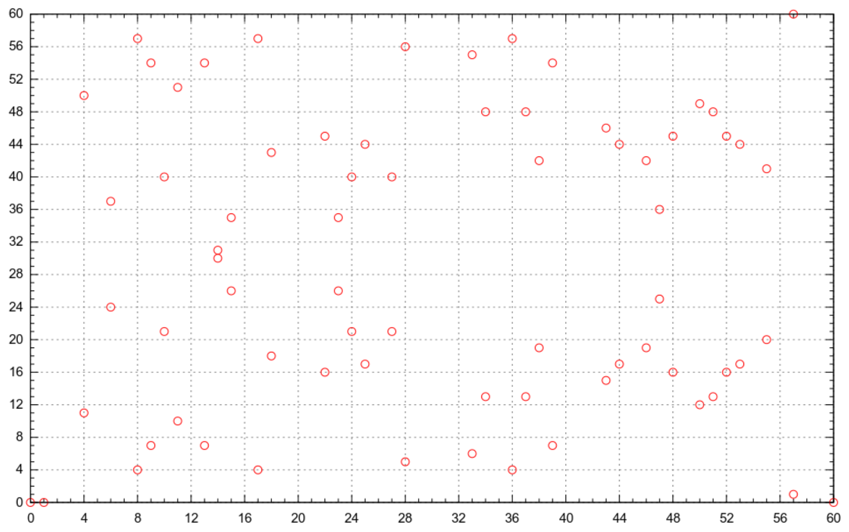
Fig 2.1.2: Affine points of elliptic curve $y^2 = x^3 - x$ over finite field $\mathbb{F}_{61}$

There are two points for every x value. Even though the graph seems random, there is still symmetry about $y = 31.5$. In elliptic curves over real numbers, there exists a negative point for each point which is reflected on the x-axis. But, in the field of $\mathbb{F}_{61}$, the negative components in the y-values are taken modulo 61, resulting in a positive number as a difference from 61. Here, $-P = (x_P, -y_P \; mod \; 61)$. It should be noted that these rules as well as the addition rules are exactly the same as those for elliptic curve groups over real numbers, with the exception that computations are performed using modulo p.

2.2 Elliptic Curve Arithmetic

2.2.1 Geometric Approach

- $\mathcal{O}$ serves as the additive identity:

  $\mathcal{O} = -\mathcal{O}$; for any point P on the elliptic curve, $P + \mathcal{O} = P$, where $P \neq \mathcal{O}$.

- The negative of point P:

  If $P = (x, y)$, then the negative $-P$ is the point with the same x-coordinate but negative of the y-coordinate of P, i.e.,

  $$-P = -(x, y) = (x, -y)$$

  It should be noted that for each point P on an elliptic curve, the point $-P$ is also on the Curve.

- Adding two distinct points:

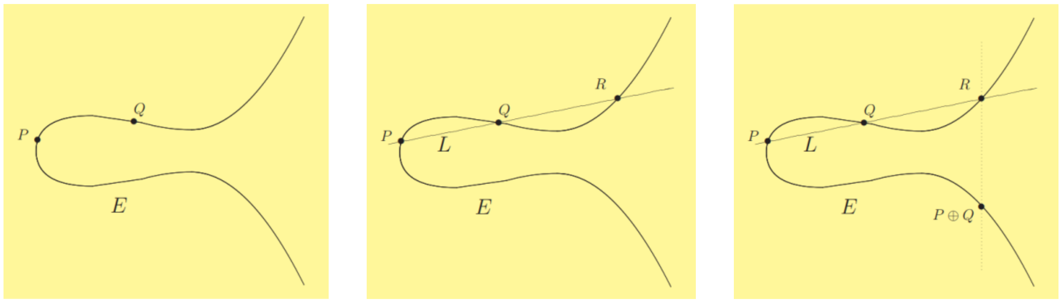

Fig 2.2.1: Adding two point using geometry in Elliptic curve

Here $P\ and\ Q$ are two points on Elliptic curve E. After drawing a line L through $P\ and\ Q$. That line L instersect curve E in a third point R. If we draw a vertical line through R, it hits E in another point. We define that point as the sum of $P\ and\ Q\ on\ E$ to be the reflected point. We denote it by $P\ \oplus\ Q$ or just $P + Q$.

- Adding two vertically opposite points:

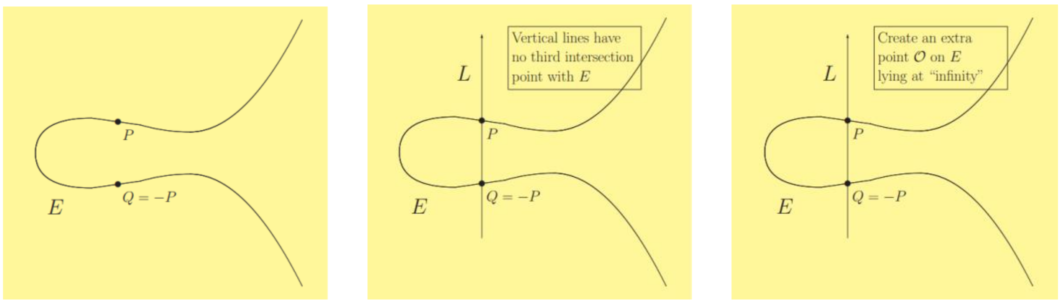
P
Q = −P
E
L
Vertical lines have no third intersection point with E
Create an extra point 𝒪 on E lying at "infinity"

Fig 2.2.2: Adding two vertically opposite point in Elliptic curve

Here $P\ and\ Q$ are two points on Elliptic curve E. Where $Q = -P$, means Q is point on elliptic curve exactly vertically opposite to the point P. As we see there is no intersection point on elliptic curve after connecting the two points. The line goes to the infinity point, $\mathcal{O}$. We denote it by $P\ \oplus\ Q = \mathcal{O}$.

- Doubling a point:

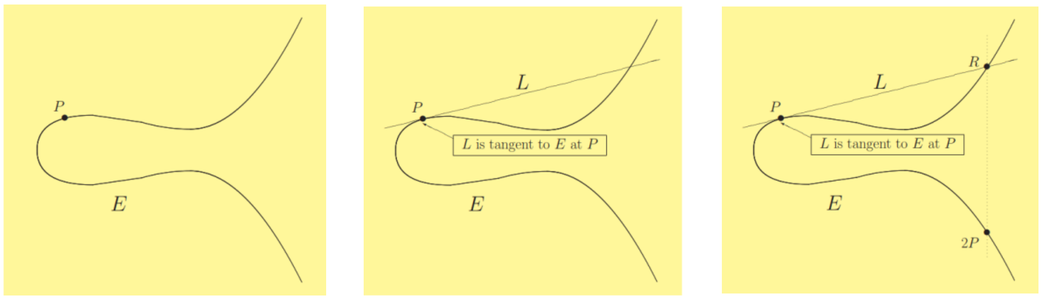

Fig 2.2.3: Doubling a point in Elliptic curve

Here we have only a single point P. In which we have to add itself or to double it up. If we join the point we know we get a tangent type line L, which intersect the elliptic curve at a specific point R. After drawing a vertical line from R, we will get the actual point which we called as $2P$.

- Doubling a point P where $y_P = 0$:

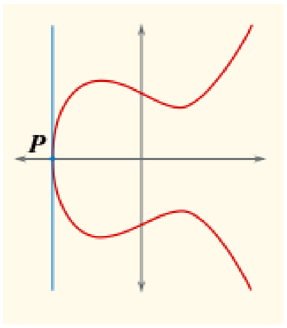

Fig 2.2.4: Doubling a point at $y_P = 0$

As we see in Fig 2.2.4, even after the drawing a vertically line, it never intersect the elliptic curve. This line goes to $\mathcal{O}$ $(infinity)$. So if $y_P = 0$, doubling a point gives us $\mathcal{O}$. It means $2P = \mathcal{O}$. But $3P = 2P + P = P$ and $4P = 2(2P) = \mathcal{O}$.

2.2.2 Algebraic Approach

Although the previous geometric descriptions of elliptic curves provide an excellent method of illustrating elliptic curve arithmetic, it is not a practical way to implement arithmetic computations. Algebraic formulae are constructed to efficiently compute the geometric arithmetic.

Let, $P = (x_P, y_P), Q = (x_Q, y_Q), P + Q = R = (x_R, y_R)$ be the points on an Elliptic Curve and λ=Slope of the line connecting P and Q.

- Adding two points:

i. When $P \neq Q$:

a. If $P = -Q$, then R is point at infinity or $R = \mathcal{O}$
b. If $P \neq -Q$, then $x_R = l^2 - x_P - x_Q$ and $y_R = l^2(x_P - x_R) - y_P$

Where $l = (y_P - y_R)/\left(x_P - x_Q\right)$.

ii. When $P = Q$:

a. If $y_P = 0$, then R is point at infinity or $R = \mathcal{O}$
b. If $y_P \neq 0$, then $x_R = l^2 - 2x_P$ and $y_R = l^2(x_P - x_R) - y_P$

Where $l = (3{x_P}^2 + a)/2y_P$.

- Adding operations:

To define the elliptic curve group, an operation, called addition, denoted by +, has to be defined for the set $E(a, b)$, where a and b satisfy the equation $y^2 = x^3 + ax + b$. Any two points on an elliptic curve when added together will result in a third point on the same curve. Sum of two points, say P and Q, on an elliptic curve is defined as the reflected point, denoted by $P + Q$.

It is assumed that there is an extra imaginary point $\mathcal{O}$ on the elliptic curve, also known as the identity element or the point at infinity. This point has no specific (x, y) coordinates, but one might imagine that its location is infinitely high above the curve where all vertical lines converge. $\mathcal{O}$ can be considered as a point on every vertical line.

PROPERTIES OF THE ADDITION OPERATION

1. Closure: If $(P, Q) \in E$, then $(P + Q) \in E$.
2. Associative: $P + (Q + R) = (P + Q) + R,\ for\ all\ P, Q, R \in E.$
3. Identity Element: $P + O = O + P = P\ for\ all\ P \in E.$
4. Inverse Element: $P + (-P) = (-P) + P = O\ for\ all\ P \in E.$
5. Commutative: $P + Q = Q + P\ for\ all\ P, Q \in E.$

The addition operation makes the points on E into a commutative or an abelian group.

2.3 Application on Cryptography

Elliptic curves are applicable for encryption, digital signatures, pseudo-random generators and other tasks. They are also used in several integer factorization algorithms that have applications in cryptography, such as Lenstra elliptic curve factorization.

An elliptic curve E over the finite field (or Galois Field) GF is defined by the following equation, known as the Weierstrass equation for elliptic curves in nonhomogeneous form :

$$y^2 + a_1xy + a_3y = x^3 - a_2x^2 - a_4x - a_6,$$

where $a_1, a_2, a_3, a_4, a_6 \in GF$ and $\Delta \neq 0$, being Δ the discriminant of E calculated in the following way :

$$\Delta = -d_2^2d_8 - 8d_4^3 - 27d_6^2 + 9d_2d_4d_6,$$

being $d_2 = a_1^2 + 4a_2, d_4 = 2a_4 + a_1a_3, d_6 = a_3^2 + 4a_6$, and finally $d_8 = a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2$.

Condition $\Delta \neq 0$ assures that the curve is non-singular, and thus there are no curve points with two or more different tangent lines.

The homogeneous form of the Weierstrass equation is

$$Y^2Z + a_1XYZ + a_3YZ^2 = X^3 + a_2X^2Z + a_4XZ^2 + a_6Z^3,$$

and this implies the existence of a special point which can only be interpreted in the projective plane: the point at infinity $\mathcal{O}$. This point is paramount in the usage of elliptic curves in cryptography, as it is the identity element that, together with the rest of the points of the elliptic curve and the addition operator (which allows to add two points of the elliptic curve, P and Q, in order to generate another point, $R = P + Q$), characterizes the elliptic curve with the mathematical structure of an abelian group.

When the same point is added several times to itself in the abelian group defined by an elliptic curve, the addition operator is transformed into the scalar multiplication, which in practice allows to multiply an elliptic curve point P by a positive integer n in order to produce another elliptic curve point, $S = n \cdot P$.

The number of points of an elliptic curve (concept also known as the cardinal or the order of the curve) is represented as $\#E$. In contrast, the order of a point P that belongs to an elliptic curve E is the smaller integer n that produces the result $n \cdot P = \mathcal{O}$.

From a cryptographic point of view, not every elliptic curve is useful. Cryptographers are interested in elliptic curves that form cyclic abelian groups, and also in elliptic curves with cyclic subgroups, so that the cofactor is a small number (e.g. 2, 4, etc.). As a consequence of Lagrange's theorem (which states that for any finite group M, the order of every subgroup N of M divides the order of M), the order of the generator (i.e. the elliptic curve point that generates all the points of the cyclic subgroup) always divides the order of the elliptic curve (which not necessarily is a prime number). Two types of finite fields $GF(q)$, with $q = pm$ elements, are used in ECC: prime finite fields $GF(p)$ (where p is an odd prime and $m = 1$) and binary finite fields $GF(2^m)$ (where $p = 2$ and m can be any integer greater than 1). When working with finite fields, using the proper change of variables it is possible to simplify the Weierstrass equation, obtaining new equations less general (they are adapted to specific finite fields) but easier to manage. If the characteristic of the finite field is 2, then $GF(q) = GF(2^m)$. If $a_1 \neq 0$, the equation $y^2 + a_1xy + a_3y = x^3 - a_2x^2 - a_4x - a_6$ can be reduced to the form

$$y^2 + xy = x^3 + ax^2 + b$$

where the discriminant is $\Delta = b$.

If $a_1 = 0$, that equation is transformed into

$$y^2 + cy = x^3 + ax + b$$

where the discriminant is $\Delta = c^4$.

Moreover, if the characteristic of the finite field is 3, then two cases appear. If $a_1^2 \neq -a_2$, that equation is reduced to

$$y^2 = x^3 + ax^2 + b,$$

where the discriminant is $\Delta = a^3b$.

In contrast, if $a_1^2 = -a_2$, that equation is reduced to

$$y^2 = x^3 + ax + b,$$

where the discriminant is $\Delta = -a^3$.

Finally, if the characteristic of $GF(q)$ is neither 2 nor 3, using the proper change of variables that equation can be transformed into

$$y^2 = x^3 + ax + b,$$

where the discriminant is $\Delta = -16(4a^3 + 27b^2)$.

The set of parameters to be used in any ECC implementation depends on the underlying finite field. When the field is $GF(p)$, the set of parameters that define the curve is (p, a, b, G, n, h), whereas if the finite field is $GF(2^m)$, the set of parameters is $(m, f(x), a, b, G, n, h)$. The meaning of each element in both sets is the following:

- p is the prime number that characterizes the finite field $GF(p)$.
- m is the integer number specifying the finite field $GF(2^m)$.
- $f(x)$ is the irreducible polynomial of grade m defining $GF(2^m)$.
- a and b are the elements of the finite field $GF(q)$ taking part in all equations.
- $G = (G_x, G_y)$ is the point of the curve that will be used as a generator of the points representing public keys.
- n is the prime number whose value represents the order of the point G (i.e. $n \cdot G = \mathcal{O}$).
- h is the cofactor of the curve, computed as $h = \#E/n$, where n is the order of the generator G.

2.3.1 Elliptic Curve Diffie Hellman

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. This shared secret may be directly used as a key, or to derive another key. The key, or the derived key, 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.

The main objective of key exchange protocols is to put in contact two or more entities communicating through an open and insecure channel, sharing a secret key that will provide data confidentiality and integrity to any information exchanged using that channel.

The following example will illustrate how a key establishment is made. 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 represented by a point Q (where $Q = dG$, that is, the result of adding G to itself 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 know the other party's public key prior to execution of the protocol.

Alice computes point $(x_k, y_k) = d_A Q_B$. Bob computes point $(x_k, y_k) = d_B Q_A$. The shared secret is x_k (the x coordinate of the point). Most standardized protocols based on ECDH derive 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.

2.3.2 Elliptic Curve Digital Signature

Elliptic Curve Digital Signature Algorithm (ECDSA) offers a variant of the Digital Signature Algorithm (DSA) which uses elliptic curve cryptography. As with elliptic-curve cryptography in general, the bit size of the public key believed to be needed for ECDSA is about twice the size of the security level, in bits. For example, at a security level of 80 bits (meaning an attacker requires a maximum of about 2^{80} operations to find the private key) the size of an ECDSA public key would be 160 bits, whereas the size of a DSA public key is at least 1024 bits. On the other hand, the signature size is the same for both DSA and ECDSA: approximately $4t$ bits, where t is the security level measured in bits, that is, about 320 bits for a security level of 80 bits.

SIGNATURE GENERATION ALGORITHM

Suppose Alice wants to send a signed message to Bob. Initially, they must agree on the curve parameters (E, G, n). In addition to the field and equation of the curve, we need G, a base point of prime order on the curve; n is the multiplicative order of the point G.

The order n of the base point G must be prime. Indeed, we assume that every nonzero element of the ring $\mathbb{Z}/n\mathbb{Z}$ are invertible, so that $\mathbb{Z}/n\mathbb{Z}$ must be a field. It implies that n must be prime.

Alice creates a key pair, consisting of a private key integer d_A, randomly selected in the interval $[1, n-1]$; and a public key curve point $Q_A = d_A \times G$. We use $\times$ to denote elliptic curve point multiplication by a scalar.

For Alice to sign a message m, she follows these steps:

1. Calculate $e = HASH(m)$. (Here HASH is a cryptographic hash function, such as SHA-2, with the output converted to an integer.)
2. Let z be the L_n, leftmost bits of e, where L_n is the bit length of the group order n. (Note that z can be greater than n but not longer)
3. Select a cryptographically secure random integer k from $[1, n-1]$.

4. Calculate the curve point $(x_1,\ y_1) = k \times\ G$.
5. Calculate $r = x_1\ mod\ \ n$. If $r = 0$, go back to step 3.
6. Calculate $s = k^{-1}\ (z + rd_A)\ mod\ n$. If $s = 0$, go back to step 3.
7. The signature is the pair (r, s). (And $(r, -s\ mod\ n)$ is also a valid signature.)

As the standard notes, it is not only required for k to be secret, but it is also crucial to select different k for different signatures, otherwise the equation in step 6 can be solved for d_A, the private key: Given two signatures (r, s) and (r, s'), employing the same unknown k for different known messages m and m', an attacker can calculate z and z', and since $s - s' = k^{-1}(z - z')$ (all operations in this paragraph are done modulo n) the attacker can find $k = \frac{z-z\prime}{s-s\prime}$. Since $s = k^{-1}(z + rd_A)$, the attacker can now calculate the private key $d_A = \frac{sk-z}{r}$.

SIGNATURE VERIFICATION ALGORITHM

For Bob to authenticate Alice's signature, he must have a copy of her public-key curve point Q_A. Bob can verify Q_A is a valid curve point as follows:

1. Check that Q_A is not equal to the identity element $\mathcal{O}$, and its coordinates are otherwise valid
2. Check that Q_A lies on the curve
3. Check that $n\ \times Q_A =\ \mathcal{O}$

After that, Bob follows these steps:

1. Verify that r and s are integers in $[1, n - 1]$. If not, the signature is invalid.
2. Calculate $e = HASH(m)$, where HASH is the same function used in the signature generation.
3. Let z be the L_n leftmost bits of e.
4. Calculate $u_1 = zs^{-1}\ mod\ n$ and $u_2 = rs^{-1}\ mod\ n$.

5. Calculate the curve point $(x_1, y_1) = u_1 \times G + u_2 \times Q_A$. If $(x_1, y_1) = \mathcal{O}$ then the signature is invalid.
6. The signature is valid if $r \equiv x_1 \ mod\ n$, invalid otherwise.

Note that an efficient implementation would compute inverse $s^{-1}\ mod\ n$ only once. Also, using Shamir's trick, a sum of two scalar multiplications $u_1 \times G + u_2 \times Q_A$ can be calculated faster than two scalar multiplications done independently.

2.3.3 Elliptic Curve Integrated encryption scheme

The most extended encryption and decryption scheme based on ECC is the Elliptic Curve Integrated Encryption Scheme (ECIES). This scheme is a variant of the ***ElGamal*** scheme proposed by ***Abdalla***, ***Bellare***, and ***Rogaway*** in “DHAES: An encryption scheme based on the Diffie-Hellman problem”, submission to IEEE P1363a, 1998.

As its name properly indicates, ECIES is an integrated encryption scheme which uses the following functions:

• Key Agreement (KA): Function used for the generation of a shared secret by two parties.

• Key Derivation Function (KDF): Mechanism that produces a set of keys from keying material and some optional parameters.

• Encryption (ENC): Symmetric encryption algorithm.

• Message Authentication Code (MAC): Data used in order to authenticate messages.

• Hash (HASH): Digest function, used within the KDF and the MAC functions.

ENCRYPTION PROCESS

In order to describe the steps that must be taken in order to encrypt a clear message, we will follow the tradition and will assume that Alice wants to send a message to Bob. In that scenario, Alice’s ephemeral private and public keys will be represented as u and U, respectively. Similarly, we will refer to Bob‘s private and public keys as v and V, respectively. In ECC, private keys are elements of the finite field, either $GF(p)$ or $GF(2^m)$, whilst public keys are points belonging to the elliptic curve and calculated as the product of the private key and the generator G of the elliptic curve. The steps (shown in Fig. 2.3.1) that Alice must complete are the following:

1) Alice must create an ephemeral key pair consisting in the finite field element u and the elliptic curve point $U = u \cdot G$. That key pair should be generated pseudo randomly exclusively for the current process.

2) After the ephemeral keys u and U are generated, Alice will use the Key Agreement function, KA, in order to create a shared secret value, which is the result of the scalar multiplication $u \cdot V$, considering as input values Alice's ephemeral private key u and Bob's public key V.

3) Then, Alice must take the shared secret value $u \cdot V$ and optionally other parameters (e.g. the binary representation of the ephemeral public key U) as input data for the Key Derivation Function, KDF. The output of this function is the concatenation of the symmetric encryption key, k_{ENC}, and the MAC key, k_{MAC}.

4) With the element k_{ENC} and the clear message, m, Alice will use the symmetric encryption algorithm, ENC, in order to produce the encrypted message, c.

5) Taking the encrypted message c, k_{MAC} and optionally other parameters, such as a text string previously agreed by both parties, Alice must use the selected MAC function in order to produce a tag.

6) Finally, Alice will take the temporary public key U, the tag, and the encrypted message c, and will send the cryptogram $(U||tag||c)$ consisting of those three concatenated elements to Bob.

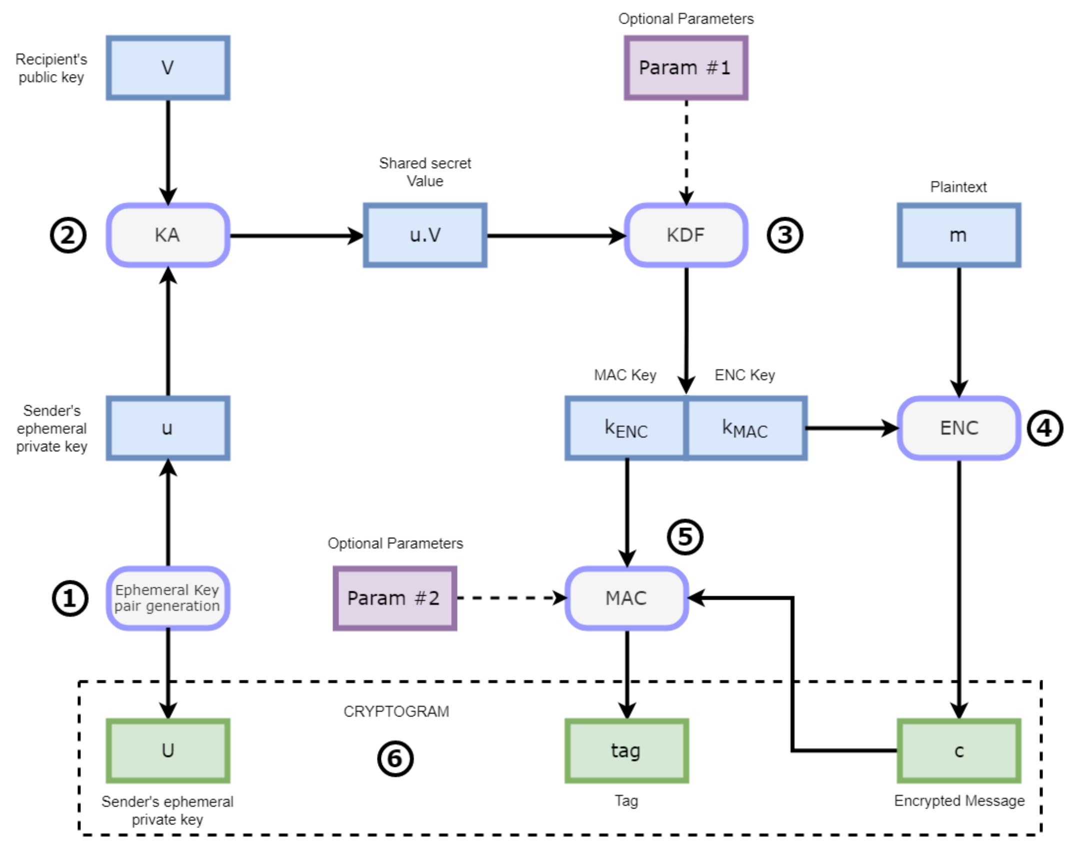

Fig 2.3.1: ECIES encryption functional diagram

DECRYPTION PROCESS

Regarding the decryption process, the steps that Bob must perform (shown in Fig. 2.3.2) are the following:

1) After receiving the cryptogram $(U||tag||c)$ from Alice, Bob must retrieve the ephemeral public key U, the tag, and the encrypted message c, so he can deal with those elements separately.

2) Using the retrieved ephemeral public key, U, and his own private key, v, Bob will multiply both elements in order to produce the shared secret value $v \cdot U$, as the result of this computation is the same that the product $u \cdot V$, which is the core of the Diffie-Hellman procedure.

3) Taking as input the shared secret value $v \cdot U$ and the same optional parameters that Alice used, Bob must produce the same encryption and MAC keys by means of the KDF procedure.

4) With the MAC key k_{MAC}, the encrypted message c, and the same optional parameters used by Alice, Bob will first compute the element tag*, and then he will compare its value with the tag that he received as part of the cryptogram. If the values are different, Bob must reject the cryptogram due to a failure in MAC verification procedure.

5) If the tag value generated by Bob is the correct one, then he will continue the process by deciphering the encrypted message c using the symmetric ENC algorithm and k_{ENC}. At the end of the decryption process, Bob will be able to access the plaintext that Alice intended to send him.

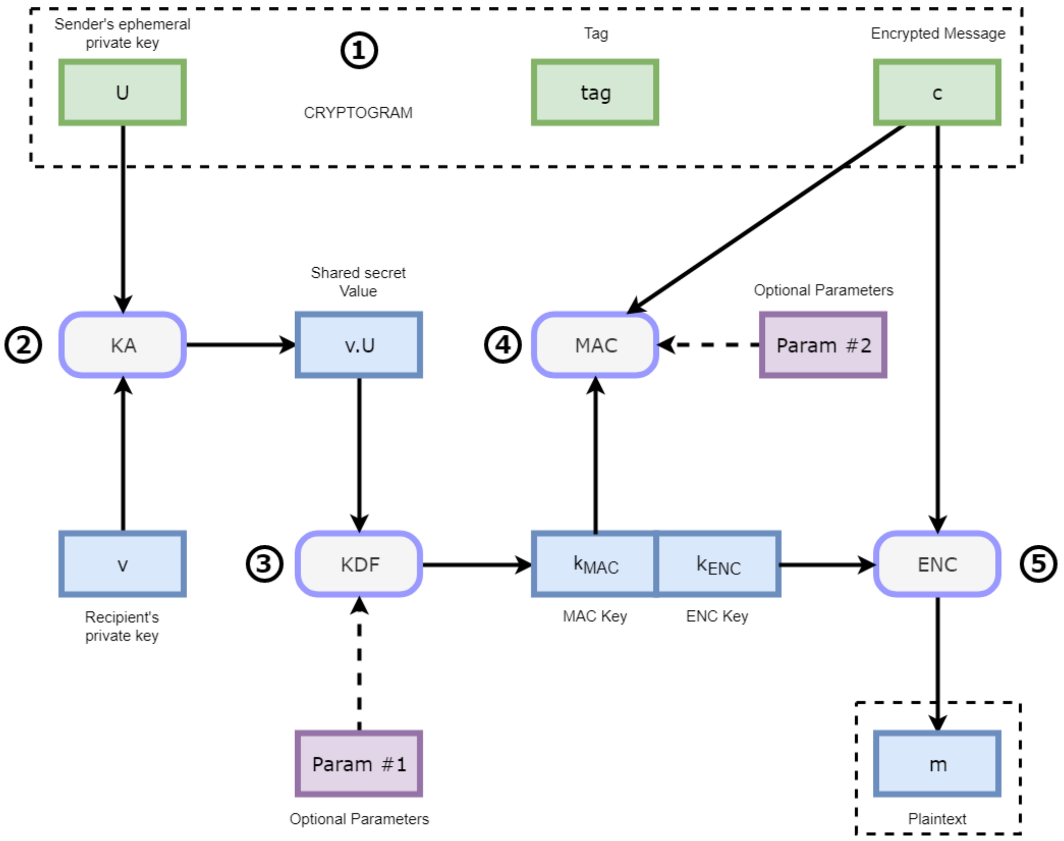

Fig 2.3.2: ECIES decryption functional diagram

2.4 Comparison over RSA

RSA – Rivest-Shamir-Adleman

- **Well established**. RSA was first described in the seventies, and it is well understood and used for secure data transmission. It lasted spectacularly as an encryption scheme for decades in which public key is used to encrypt the information while the private key is used to decrypt the information.

- **RSA is based on the difficulty of factoring large integers**. Factoring large integers in order to identify prime numbers is processor-intensive, and hence has been very effective as a defense until now. But it comes with overhead: slow key generation, maximum consumption of computer resources, for instance.

- **Scalability is not optimal**. This is a significant flaw at a time when we know that the proliferation of internet-linked devices – the Internet of Things – will put demands on a system that's already expected to be obsolete by 2030. RSA is vulnerable against quantum computers and brute force attack, hence a new algorithm is required, which can offer a better performance for a specified security level.

- **Very fast, very simple encryption**. RSA encryption is based on simple principles, and in the right environment can run faster than ECC. RSA might not be scalable, but in certain situations, for instance, for internal organizations, it may be faster. In RSA, the reliability and security devolves on the level of difficulty of integer factorization.

ECC – Elliptic Curve Cryptography

- **Need Special Adjustment**. Depending on your audience and your ability to maintain your systems for legacy equipment, you might need to implement ECC encryption in an environment that is not prepared for it. There is a way to set up called hybrid SSL that allows implementation of ECC cryptography on RSA trusted root keys, for that, you will have to discuss this with your CA and hosting provider.

- **Relies on detecting the separate logarithm of a random elliptic curve**. The ECC algorithm works on Elliptic Curve Discrete Logarithm Problem (ECDLP) that is hard to crack for hackers. There is no known solution to the mathematical problem posed by the equation that produces the elliptical curve in a graph, and so the only solution is to try random numbers. However, each bit size provides more options than RSA, making it hard that the brute force approach is unlikely to succeed.

- **Shorter keys in ECC encryption are as strong as long keys for RSA**. This results in much less network overhead, allowing faster performance and a better customer or user experience. It also means that in the long term, there is more room for growth, because each additional bit gives more options than each additional bit in RSA. That also means a slower growth in bit size over time, which makes it more scalable, potentially, for the Internet of Things.

- **Smaller certificate size**. Again, the amount of information necessary to exchange for validation is significantly less than RAS, lowering network overhead and increasing performance, which provides an improved user or customer experience. It also improves scalability by providing an environment in which increased traffic can be handled by the server because of the lower overhead, without changing the infrastructure.

- **Low on CPU consumption and memory usage**. For both client and server, this is an improved experience, streamlining the connection and simplifying the process. ECC consumes less computing power and battery resource. RSA certificate can hold 450 requests per second with 150 millisecond average response time where ECC requires only 75 milliseconds for responding to the same amount of requests per second. ECC has great response time when it communicates for server to desktop.

- **Hybrid SSL for ECC to work**. For some organizations, it is necessary that a website works successfully with older equipment, and in that case, each organization must consider a technique of hybrid certificates that allows an ECC algorithm to support even on RSA trusted root certificate.

Minimum size (bits) of Public keys			Key Size Ratio
Security (bits)	RSA	ECC	ECC to RSA
80	1024	160-223	1:6
112	2048	224-255	1:9
128	3072	256-383	1:12
192	7680	384-511	1:20
256	15360	512+	1:30

Keys in ECC are significantly shorter than in other cryptosystems such as RSA. A shorter key implies easier data management, lower hardware requirements (in terms of buffers, memory, data storage, etc.), less bandwidth when transmitting the keys over a network, and longer battery life in devices where it is important, such as mobile phones.

2.5 Finding Curve Order

The number of points on an elliptic curve is defined as the cardinality of the set of pairs of points that lie in the finite field $\mathbb{F}_p$ over which the elliptic curve is defined, and which satisfy the equation of the elliptic curve, plus an additional point to represent the point at infinity. The order of the elliptic curve plays an important role in the selection of the curve for cryptographic uses.

Many of the cryptographic standards select a secure curve based on the curve statistics provided by the curve order. One of the most common requirements is that the order of the curve defined under a finite field $\mathbb{F}_p$ not be a product of small primes. It is usually preferred that the group order be a multiple of large primes or that the group order at least contain a large prime of the order of at least 2^{160}.

In Elliptic curve cryptography, the curve order is very important because using the curve we solve most of the problem in Diffie Hellman, Digital Signature and Encryption & Decryption scheme. It is also used to find out the point order of a point. Curve order is nothing but just number of points on the Elliptic curve which satisfy the elliptic curve equation.

There are several approaches to the problem. Beginning with the naive approach, we trace the developments up to Schoof's definitive work on the subject. There are three type of approach to finding the curve order:

1) Naïve Approach
2) Advance Approach
3) Schoof's Algorithm

Several algorithms make use of the fact that groups of the form $E(\mathbb{F}_p)$ are subject to an important theorem due to Hasse, that bounds the number of points to be considered. The Hasse's theorem states that if E is an elliptic curve over the finite field $\mathbb{F}_p$, then the cardinality of $E(\mathbb{F}_p)$ satisfies

$$\left||E(\mathbb{F}_p)| - (q+1)\right| \leq 2\sqrt{q}$$

3
VARIOUS APPROACHES FOR FINDING CURVE ORDER

3.1 Naïve Approach

The naive approach to counting points, which is the least sophisticated, involves running through all the elements of the field $\mathbb{F}_q$ and testing which ones satisfy the Weierstrass form of the elliptic curve

$$y^2 = x^3 + Ax + B$$

Example

Let E be the curve $y^2 = x^3 + x + 1$ over $\mathbb{F}_5$. To count points on E, we make a list of the possible values of x, then of the quadratic residues of $x \, mod \, 5$ (for lookup purpose only), then of $x^3 + x + 1 \, mod \, 5$, then of y of $x^3 + x + 1 \, mod \, 5$. This yields the points on E.

x	x^2	x^3+x+1	y	**Points**
0	0	1	1,4	$(0,1),(0,4)$
1	1	3	_	_
2	4	1	1,4	$(2,1),(2,4)$
3	4	1	1,4	$(3,1),(3,4)$
4	1	4	2,3	$(4,2),(4,3)$

E.g. the last row is computed as follows: If you insert $x = 4$ in the equation $y^2 = x^3 + x + 1$ you get 4 as result (3rd column). This result can be achieved if $y = 2,3$ (Quadratic residues can be looked up in the 2nd column). So the points for the last row are $(4,2),(4,3)$.

Therefore, $E(\mathbb{F}_5)$ has cardinality of 9: the 8 points listed before and the point at infinity.

This algorithm requires running time $O(q)$, because all the values of $x \in \mathbb{F}_q$ must be considered. This is the simplest form of the point counting algorithm, but with a complexity of $O(q)$, it is not efficient for large values of q. For a field $\mathbb{F}_q$ having an extremely large characteristic q, the iteration of the loop is formidable. If we were to consider counting points on a cryptographic curve, the calculations would be in the magnitude of 2^{160} and above, which would be impossible to perform within reasonable time. It is obvious that it is not feasible to count points on curves defined over large finite fields using this algorithm.

3.2 Advance Approach

Talking about the advance approach it means it takes less time from the former naive approach to counting points. There is an algorithm which we called as Baby-step giant-step which counts the point in $O(\sqrt[4]{q})$ time.

We pick an element $P = (x,y) \in E(\mathbb{F}_q)$ by selecting random values of x until $x^3 + Ax + B$ is a square in $\mathbb{F}_q$ and then computing the square root of this value in order to get y. Hasse's theorem tells us that $|E(\mathbb{F}_q)|$ lies in the interval $\left(q + 1 - 2\sqrt{q}, q + 1 + 2\sqrt{q}\right)$. Thus, by Lagrange's theorem, finding a unique M lying in this interval and satisfying $MP = \mathcal{O}$, results in finding the cardinality of $E(\mathbb{F}_q)$. The algorithm fails if there exist two integers

M and M' in the interval such that $MP = M'P = \mathcal{O}$. In such a case it usually suffices to repeat the algorithm with another randomly chosen point in $E(\mathbb{F}_q)$.

Trying all values of M in order to find the one that satisfies $MP = \mathcal{O}$ takes around $4\sqrt{q}$ steps. However, by applying the baby-step giant-step algorithm to $E(\mathbb{F}_q)$, we are able to speed this up to around $4\sqrt[4]{q}$ steps. The algorithm is as follows.

```
1. Choose $m$ integer, $m > \sqrt[4]{q}$
2. FOR{ $j = 0\ to\ m$} DO
3.        $P_j \longleftarrow jP$
4. END FOR
5. $L \longleftarrow 1$
6. $Q \longleftarrow (q+1)P$
7. REPEAT compute the points $Q + k(2mP)$
8. UNTIL $\exists j$: $Q + k(2mP) = \pm P_j$          \\the $x$-coordinates are compared
9. $M \leftarrow q + 1 + 2mk \mp j$                   \\note $MP = \mathcal{O}$
10. Factor $M$. Let $p_1, \dots, p_r$ be the distinct prime factors of $M$.
11. WHILE $i \leq r$ DO
12.       IF $\frac{M}{p_i}P = \mathcal{O}$
13.              THEN $M \leftarrow \frac{M}{p_i}$
14.              ELSE $i \leftarrow i + 1$
15.       ENDIF
16. ENDWHILE
17. $L \leftarrow lcm(L, M)$                          \\note $M$ is the order of the point $P$
18. WHILE $L$ divides more than one integer $N$ in $\left(q + 1 - 2\sqrt{q}, q + 1 + 2\sqrt{q}\right)$
19.       DO choose a new point $P$ and go to 1.
20. ENDWHILE
21. RETURN $N$                                       \\it is the cardinality of $E(\mathbb{F}_q)$
```

One drawback of this method is that there is a need for too much memory when the group becomes large. In order to address this, it might be more efficient to store only the x coordinates of the points jP (along with the corresponding integer j). However, this leads to an extra scalar multiplication in order to choose between $-j$ and $+j$.

3.3 Schoof’s Algorithm

Schoof's algorithm is an efficient algorithm to count points on elliptic curves over finite fields. The algorithm has applications in elliptic curve cryptography where it is important to know the number of points to judge the difficulty of solving the discrete logarithm problem in the group of points on an elliptic curve.

The algorithm was published by ***René Schoof*** in 1985 and it was a theoretical breakthrough, as it was the first deterministic polynomial time algorithm for counting points on elliptic curves. Before Schoof's algorithm, approaches to counting points on elliptic curves such as the naive and baby-step giant-step algorithms were, for the most part, tedious and had an exponential running time.

Schoof’s algorithm uses division polynomials and Hasses’s theorem, along with the Chinese remainder theorem. Using this algorithm it reduces the time taken for calculating the curve order.

When generating curves for elliptic curve cipher systems, the order of the group of points is important. The main method for generating these curves depends on the point counting problem. We require, at least, for the group to be divisible by a large prime factor. By large we mean at least 160 bits in length.

Being able to randomly choose an elliptic curve over large finite fields is important in elliptic curve cryptosystems. The point counting problem, the problem of determining the number of points on this curve, is important. In some elliptic curve cipher systems, the order does not need to be known, but its security depends upon the order having large prime factors. The following algorithm is one technique which has become the basis of most current efficient schemes for counting points on an elliptic curve.

4

SCHOOF'S ALGORITHM

4.1 INTRODUCTION

A theoretical breakthrough for the problem of computing the cardinality of groups of the type $E(\mathbb{F}_q)$ was achieved by ***René Schoof***, who, in 1985, published the first deterministic polynomial time algorithm. Central to Schoof's algorithm are the use of ***division polynomials*** and ***Hasse's theorem***, along with the ***Chinese remainder theorem***.

Schoof's insight exploits the fact that, by Hasse's theorem, there is a finite range of possible values for $|E(\mathbb{F}_q)|$. It suffices to compute $|E(\mathbb{F}_q)|$ modulo an integer $N > 4\sqrt{q}$. This is achieved by computing $|E(\mathbb{F}_q)|$ modulo primes $\ell_1, \ldots, \ell_s$ whose product exceeds $4\sqrt{q}$, and then applying the Chinese remainder theorem. The key to the algorithm is using the division polynomial ψ_l to efficiently compute $|E(\mathbb{F}_q)|$ modulo ℓ.

The running time of Schoof's Algorithm is polynomial in $n = \log q$, with an asymptotic complexity of $O(n^2 M(n^3)/\log n) = O(n^{5+o(1)})$, where $M(n)$ denotes the complexity of integer multiplication. Its space complexity is $O(n^3)$.

4.2 Algorithm

Let E be an elliptic curve defined over the finite field $\mathbb{F}_q$, where $q = p^n$ for p a prime and n an integer ≥ 1. Over a field of characteristic $\neq 2,3$ an elliptic curve can be given by a (short) Weierstrass equation

$$y^2 = x^3 + Ax + B$$

with $A, B \in \mathbb{F}_q$. The set of points defined over $\mathbb{F}_q$ consists of the solutions $(a,b) \in \mathbb{F}_q^2$ satisfying the curve equation and a point at infinity $\mathcal{O}$. Using the group law on elliptic curves restricted to this set one can see that this set $E(\mathbb{F}_q)$ forms an abelian group, with $\mathcal{O}$ acting as the zero element. In order to count points on an elliptic curve, we compute the cardinality of $E(\mathbb{F}_q)$. Schoof's approach to computing the cardinality $\#E(\mathbb{F}_q)$ makes use of Hasse's theorem on elliptic curves along with the Chinese remainder theorem and division polynomials.

HASSE'S THEOREM

Hasse's theorem states that if $E(\mathbb{F}_q)$ is an elliptic curve over the finite field $\mathbb{F}_q$, then $\#E(\mathbb{F}_q)$ satisfies

$$|q + 1 - \#E(\mathbb{F}_q)| \leq 2\sqrt{q}$$

This powerful result, given by Hasse in 1934, simplifies our problem by narrowing down $\#E(\mathbb{F}_q)$ to a finite (albeit large) set of possibilities. Defining t to be $q + 1 - \#E(\mathbb{F}_q)$, and making use of this result, we now have that computing the value of t modulo N where $N > 4\sqrt{q}$, is sufficient for determining t, and thus $\#E(\mathbb{F}_q)$. While there is no efficient way to compute $t \ (mod \ N)$ directly for general N, it is possible to compute $t(mod \ l)$ for l a small prime, rather efficiently. We choose $S = \{l_1, l_2, \ldots, l_r\}$ to be a set of distinct primes such that $\prod l_i = N > 4\sqrt{q}$. Given $t(mod \ l_i)$ for all $l_i \in S$, the Chinese remainder theorem allows us to compute $t(mod \ N)$.

In order to compute $t(mod \ l)$ for a prime $l \neq p$, we make use of the theory of the Frobenius endomorphism $\emptyset$ and division polynomials. Note that considering primes $l \neq p$ is no loss since we can always pick a bigger prime to take its place to ensure the product is big enough. In any case Schoof's algorithm is most frequently used in addressing the case $q = p$ since there are more efficient, so called p adic algorithms for small-characteristic fields.

DIVISIONAL POLYNOMIALS

Division polynomials provide a way to calculate multiples of points on elliptic curves and to study the fields generated by torsion points. They play a central role in the study of counting points on elliptic curves in Schoof's algorithm.

The set of division polynomials is a sequence of polynomials in $\mathbb{Z}[x, y, A, B]$ with x, y, A, B free variables that is recursively defined by:

$$\psi_0 = 0$$
$$\psi_1 = 1$$
$$\psi_2 = 2y$$
$$\psi_3 = 3x^3 + 6Ax^2 + 12Bx - A^2$$
$$\psi_4 = 4y(x^6 + 5Ax^4 + 20Bx^3 - 5A^2x^2 - 4ABx - 8B^2 - A^3)$$
$$\vdots$$
$$\psi_{2m+1} = \psi_{m+2}\psi_m^3 - \psi_{m-1}\psi_{m+1}^3 \;\; for\, m \;\geq 2$$
$$\psi_{2m} = \left(\frac{\psi_m}{2y}\right).(\psi_{m+2}\psi_{m-1}^2 - \psi_{m-2}\psi_{m+1}^2) \;\; for\, m \;\geq 3$$

This polynomial ψ_n is called the n^{th} division polynomial. Which we will use in Schoof's algorithm.

Divisional polynomial properties are:

- In practice, one sets $y^2 = x^2 + Ax + B$, and then $\psi_{2m+1} \in \mathbb{Z}[x, A, B]$ and $\psi_{2m} \in 2y\mathbb{Z}[x, A, B]$.
- The division polynomials form a generic elliptic divisibility sequence over the ring $\mathbb{Q}[x, y, A, B]/(y^2 - x^3 - Ax - B)$.
- If an elliptic curve E is given in the Weierstrass form $y^2 = x^3 + Ax + B$ over some field K, i.e. $A, B \in K$, one can use these values of A, B and consider the division polynomials in the coordinate ring of E. The roots of ψ_{2n+1} are the x-coordinates of the points of $E[2n + 1]\backslash\mathcal{O}$, where $E[2n + 1]$ is the $(2n + 1)^{th}$ torsion subgroup of E. Similarly, the roots of $\psi_{2n}/2y$ are the x-coordinates of the points of $E[2n]\backslash E[2]$.

- Given a point $P = (x_P, y_P)$ on the elliptic curve $E: y^2 = x^3 + Ax + B$ over some field K, we can express the coordinates of the nth multiple of P in terms of division polynomials:

$$nP = \left(\frac{\phi_n(x)}{\psi_n^2(x)}, \frac{\omega_n(x,y)}{\psi_n^3(x,y)}\right) = \left(x - \frac{\psi_{n-1}\psi_{n+1}}{\psi_n^2(x)}, \frac{\psi_{2n}(x,y)}{2\psi_n^4(x)}\right)$$

where ϕ_n and ω_n are defined by:

$$\phi_n = x\psi_n^2 - \psi_{n+1}\psi_{n-1}$$

$$\omega_n = \frac{\psi_{n+2}\psi_{n-1}^2 - \psi_{n-2}\psi_{n+1}^2}{4y}$$

Using the relation between ψ_{2m} and ψ_{2m+1}, along with the equation of the curve, the functions ψ_n^2, $\frac{\psi_{2n}}{y}$, ψ_{2n+1} , $\phi_n$are all in $K[x]$.

Let $p > 3$ be prime and let be $E: y^2 = x^3 + Ax + B$ an elliptic curve over the finite field $\mathbb{F}_p$, i.e., $A, B \in \mathbb{F}_p$. The ℓ-torsion group of E over $\overline{\mathbb{F}}_q$ is isomorphic to $\mathbb{Z}/\ell \times \mathbb{Z}/\ell$ if $\ell \neq p$, and to $\mathbb{Z}/\ell$ or $\{0\}$ if $\ell = p$. Hence the degree of ψ_l is equal to either $\frac{1}{2}(l^2 - 1)$, $\frac{1}{2}(l - 1)$, or 0.

René Schoof observed that working modulo the ℓth division polynomial allows one to work with all ℓ-torsion points simultaneously. This is heavily used in Schoof's algorithm for counting points on elliptic curves.

Main Algorithm

Input:

1. An elliptic curve $E = y^2 - x^3 - Ax - B$.
2. An integer q for a finite field $\mathbb{F}_q$ with $q = p^b, b \geq 1$.

Output:

The number of points of E over $\mathbb{F}_q$

Choose a set of odd primes S not containing p such that $N = \prod_{l \in S} l > 4\sqrt{q}$.

Put $t_2 = 0$ if $gcd(x^q - x, x^3 + Ax + B) \neq 1$, else $t_2 = 1$.

Compute the division polynomial ψ_l.

All computations in the loop below are performed in the ring

$$\mathbb{F}_q[x,y]/(y^2 - x^3 - Ax - B, \psi_l)$$

.

```
For $l \in S$ do:
  Let $\bar{q}$ be the unique integer such that $q \equiv \bar{q} \bmod l$ and $|\bar{q}| < l/2$.
  Compute $(x^q, y^q)$, $(x^{q^2}, y^{q^2})$ and $(x_{\bar{q}}, y_{\bar{q}})$.
  If $x^{q^2} \neq x_{\bar{q}}$ then
    Compute $(X, Y)$.
    for $1 \leq \bar{t} \leq (l-1)/2$ do:
      if $X = x_{\bar{t}}^q$ then
        if $Y = y_{\bar{t}}^q$ then
          $t_l = \bar{t}$;
        else
          $t_l = -\bar{t}$.
  else if $q$ is a square modulo $l$ then
    compute $w$ with $q \equiv w^2 \bmod l$
    compute $w(x^q, y^q)$
    if $w(x^q, y^q) = (x^{q^2}, y^{q^2})$ then
      $t_l = 2w$
    else if $w(x^q, y^q) = (x^{q^2}, -y^{q^2})$ then
      $t_l = -2w$
    else
      $t_l = 0$
  else
    $t_l = 0$
Use the Chinese Remainder Theorem to compute $t$ modulo $N$
  from the equations $t \equiv t_l \bmod l$, where $l \in S$.
Output $q + 1 - t$
```

4.3 Analysis

For our example curve $E: y2 = x3 + 46x + 74$ over $\mathbb{F}_{97}$, Schoof's algorithm produces the following results. First, since $\lceil 4\sqrt{q} \rceil = 40$, we need a product of small primes at least this large so the algorithm selects the primes $\{2, 5, 7\}$ with $\prod l_i = 70$.

Also $9 < \sqrt{97} < 10$, so Hasse's theorem gives $79 \leq \# E(\mathbb{F}_q) \leq 117$.

The next step is to compute $t (mod\ 2)$. For this step we find
$x^p\ (mod\ x^3 + ax + b) = 47 + 60x + 30x^2$ and
$\gcd(x^p - x, x^3 + ax + b) = 40 + x$

Since the gcd is not equal to 1, $E[2]$ is not empty so $t \equiv 0\ (mod\ 2)$.

Next we test $\phi_P{}^2 P = \pm kP\ for\ k \equiv p(mod\ 5)$ and we find

$$\begin{aligned} f[5] &= 23 + 67x + 11x^2 + 38x^3 + 77x^4 + 43x^5 + 93x^6 + 26x^7 + 47x^8 \\ &\quad + 87x^9 + 39x^{10} + 5x^{12} p_{16} \\ &= 7 + 91x + 40x^2 + 24x^3 + 81x^4 + 69x^5 + 43x^6 + 45x^7 \\ &\quad + 39x^8 + 14x^9 + 30x^{10} + 79x^{11} \end{aligned}$$

$gcd(p_{16}, f_l) = 1$. Hence, there is no point in $E[5]$ satisfying $\phi_P{}^2 P = \pm kP$ so we proceed to case two.

Next we test $\phi_P{}^2 P = \tau\phi_P$ until we find for $\tau = 2$ that

$p_{19x} \equiv 0\ (mod\ f_l, p)$ and
$gcd(p_{19x}, f_l) = 23 + 67x + 11x^2 + 38x^3 + 77x^4 + 43x^5 + 93x^6 + 26x^7 + 47x^8 + 87x^9 + 39x^{10} + 5x^{12}$.

Since the gcd is not equal to 1, we know that $\tau = \pm 2$ so we compute

$$\begin{aligned} p_{19y} &= 39 + 52x + 48x^2 2 + 33x^3 + 91x^4 + 3x^5 + 23x^6 + 59x^7 + 16x^8 \\ &\quad + 37x^9 + 33x^{10} + 74x^{11} \end{aligned}$$

$gcd(p_{19y}, f_l) = 1.$

Since this gcd is 1, there is no point in $E[5]$ satisfying ${\phi_P}^2 P = \tau\phi_P$, so we must have $t \equiv 2 \ (mod\ 5) \equiv 3 \ (mod\ 5)$. Similarly, for $l = 7$ we find at $\tau = 3$ that $gcd(p_{19x}, f_7) \neq 1$ and $gcd(p_{19x}, f_7) = 1$ so that $t \equiv 2 \ (mod\ 7) \equiv 3 \ (mod\ 7)$. Thus we have the following set of simultaneous congruence.

$$t \equiv 0 \ (mod\ 2), t \equiv 3 \ (mod\ 5), t \equiv 4 \ (mod\ 7)$$

Using the Chinese Remainder theorem we find that the smallest positive integer satisfying this set of congruences is $t = 18$. Since $p + 1 - t = 80$ and 80 is within Hasse's bounds we can conclude $\#E(\mathbb{F}_p) = 80$.

4.4 Result and Conclusion

Schoof's method, as here implemented, has primarily educational value. This is so for several reasons. The practicality of the algorithm is greatly limited by the quadratic growth of the degree of the division polynomials. For example, for an elliptic curve over $\mathbb{F}_p$ where p have 200 digits, we must perform modular polynomial arithmetic using the 55th division polynomial, which produces intermediate products of degree greater than 9×10^6. Our implementation is believed to be correct for all cases based on cross testing against an algorithm employing the Baby-Step, Giant-Step method to determine random EC point orders. Performance was enhanced by replacing $PolynomialMod[P, Q, p]$ with $PolynomialRemainder[P, Q, x, Modulus \rightarrow p]$ and by significant code factoring to reduce redundant computations. On the other hand, the major advantage of our implementation is as an exploration tool. All of the algorithms we implemented are well-documented, and rely only on low-level functions with GMP, MPFR & FLINT, making the operation of the algorithms transparent and open to experimentation.

5 CONCLUSION AND FURTHER STUDY

This thesis has barely touched the vast and rich mathematical theory of elliptic curves. And even in the small stream of cryptography, we have merely skimmed the surface of the subject. In the vaster realm of mathematics, the theory of elliptic curves appears and reappears in contexts too numerous to list, ranging from Hasse’s Theorem to division polynomial to Schoof’s algorithm and beyond. The annotated bibliography includes a few references to assist you in learning more about the number theory and cryptographic applications of elliptic curves.

We can further reduce the complexity using ***Schoof-Elkies-Atkin*** algorithm. The improvements due to ***Elkies*** and ***Atkin***, called the SEA algorithm, reduce to nearly linear growth the degree of certain divisors of the division polynomials which we can use in their place, making the method applicable to elliptic curves of with cryptographic utility.

BIBLIOGRAPHY

[1] W. Diffie and M.E. Hellman, “New directions in cryptography”, IEEE Transactions in Information Theory, vol. 22, pp. 644654, 1976.

[2] R. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems”, Communications of the ACM, vol. 26, pp. 96-99, 1983.

[3] T. ElGamal. “A public key cryptosystem and a signature scheme based on discrete logarithms”, IEEE Transactions on Information Theory, vol. 31, pp. 469—472, 1985.

[4] V.S. Miller, “Use of elliptic curves in cryptography”, Lecture Notes in Computer Science, vol. 218, pp. 417-426, 1986.

[5] N. Koblitz, “Elliptic curve cryptosystems”, Mathematics of Computation, vol. 48, pp. 203-209, 1987.

[6] Bundesamt für Sicherheit in der Informationstechnik (BSI), Elliptic Curve Cryptography, TR 03111, 2009. http://www.bsi.de/literat/tr/tr03111/BSI-TR-03111.pdf

[7] D. Hankerson, A. J. Menezes, and S. Vanstone, Guide to Elliptic Curve Cryptography. New York: Springer-Verlag, 2003.

[8] National Institute of Standards and Technology (NIST), Recommendation for key management – Part 1: General, SP 800-57, 2007.

[9] V. Gayoso Martínez, L. Hernández Encinas, and C. Sánchez Ávila, “A Comparison of the Standardized Versions of ECIES”, Proceedings of the Sixth International Conference on Information Assurance and Security – IAS 2010, Atlanta, 2010.

[10] J. Silverman, The Arithmetic of Elliptic Curves. New York: Springer-Verlag, 1986.

[11] American National Standards Institute (ANSI), Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography, X9.63, 2001.

[12] Institute of Electrical and Electronics Engineers (IEEE), Standard Specifications for Public Key Cryptography, Std. 1363, 2000.

[13] National Institute of Standards and Technology (NIST), Recommendation for Pair-wise Key Establishment Schemes Using Discrete Logarithm Cryptography, SP 800-56A, 2005.

[14] Standards for Efficient Cryptography Group (SECG), Elliptic Curve Cryptography, SEC 1, version 2, 2009. http://www.secg. org/download/aid-780/sec1-v2.pdf

[15] National Institute of Standards and Technology (NIST), Digital Signature Standard (DSS), FIPS 186-2, 2000.

[16] American National Standards Institute (ANSI), Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA), X9.62, 1998.

[17] M. Abdalla, M. Bellare, and P. Rogaway, “DHAES: An encryption scheme based on the Diffie-Hellman problem”, submission to IEEE P1363a, 1998. http://grouper.ieee.org/groups/1363/P1363a/contributions/ dhaes.pdf

[18] M. Abdalla, M. Bellare, and P. Rogaway, DHIES: An encryption scheme based on the Diffie-Hellman problem, unpublished, 2001. http://www.cs.ucdavis.edu/~rogaway/papers/dhies.pdf

[19] Institute of Electrical and Electronics Engineers (IEEE), Standard Specifications for Public Key Cryptography - Amendment 1: Additional Techniques, Std. 1363a, 2004.

[20] International Organization for Standardization/International Electrotechnical Commission (ISO/IEC), Information Technology – Security Techniques – Encryption Algorithms – Part 2: Asymmetric Ciphers, 18033-2, 2006.

[21] National Institute of Standards and Technology (NIST), Secure Hash Standard, FIPS 180-2, 2002.

[22] H. Dobbertin, A. Bosselaers, and B. Preneel, “RIPEMD-160: A Strengthened Version of RIPEMD”, Lecture Notes in Computer Science, vol. 1039, pp. 71-82, 1996.

[23] International Organization for Standardization/International Electrotechnical Commission (ISO/IEC), Information Technology -- Security Techniques -- Hash-functions -- Part 3: Dedicated Hashfunctions, 10118-3, 2004.

[24] American National Standards Institute (ANSI), Triple Data Encryption: Modes of Operation, X9.52, 1998.

[25] National Institute of Standards and Technology (NIST), Advanced Encryption Standard, FIPS 197, 2001.

[26] M. Matsui, Specification of MISTY1 - A 64-bit Block Cipher, submission to NESSIE, 2000. https://www.cosic.esat.kuleuven.be/nessie/workshop/submi ssions/misty1.zip

[27] C. Adams, The CAST-128 Encryption Algorithm, RFC 2144, 1997. http://www.ietf.org/rfc/rfc2144.txt

[28] K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita, “Camellia: A 128-Bit Block Cipher Suitable for Multiple Platforms - Design and Analysis”, Lecture Notes in Computer Science, vol. 2012, pp. 39-56, 2001.