Elliptic Curve Cryptosystems

Mugino Saeki

School of Computer Science

McGill University, Montreal

February 1997

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfilment of the requirements of the degree of Master of Science in Computer Science.

Abstract

The application of elliptic curves to the field of cryptography has been relatively recent. It has opened up a wealth of possibilities in terms of security, encryption, and real-world applications. In particular, we are interested in public-key cryptosystems that use the elliptic curve discrete logarithm problem to establish security. The objective of this thesis is to assemble the most important facts and findings into a broad, unified overview of this field. To illustrate certain points, we also discuss a sample implementation of the elliptic curve analogue of the El Gamal cryptosystem.

Résumé

L'application des courbes elliptiques au domaine de la cryptographie est relativement récente. Elle a ouvert un éventail de possibilités en termes de sécurité, de chiffrement, et des applications pratiques. En particulier, nous nous intéressons aux systèmes à clé publique qui utilisent le problème du logarithme discret sur des courbes elliptiques pour établir la sécurité. L'objectif de cette thèse est de rassembler les résultats et les faits les plus importants en un aperçu large et unifié de ce domaine. Pour illustrer certains points, nous discutons aussi une mise-en-oeuvre de l'analogue du système El Gamal.

Acknowledgements

Many thanks go to my thesis supervisors David Avis (at McGill University) and Claude Crépeau (at Université de Montréal) for their patient guidance and generous advice.

Contents

Chapter 1

Introduction

Cryptography is the science of securely transmitting messages from a sender to a receiver. The objective is to encrypt the message in a way such that an eavesdropper would not be able to read it. A cryptosystem is a system of algorithms for encrypting and decrypting messages for this purpose. Computer cryptography, once the exclusive domain of the military, has only recently become accessible to the layperson with the advent of personal computers and the boom in public research over the last 20 years.

In contrast, elliptic curves are not new to the field of Number Theory — they have been studied and scrutinized for most of this past century. But the application of elliptic curves to the field of cryptography is a recent phenomenon, beginning barely 10 years ago. Some well-known cryptosystems work with multiplicative groups of fields, and as it turns out, elliptic curves over finite fields are a rich source of finite abelian groups. Faced with an infinite variety of elliptic curves to choose from, much research remains to be conducted on how different cryptosystems using different elliptic curves perform.

Future studies will not be motivated solely by the simple concept of applying elliptic curves to cryptographic schemes. As we will see in this thesis, the appeal of the elliptic curve cryptosystem is its strengths and its practical applications to the real world. Such systems involve elementary arithmetic operations that make it easy to implement (in either hardware or software). They can maintain reliable security with key lengths that are shorter (therefore more practical) than those in other public-key schemes. There are very few known attacks that can break the cryptosystems: each is effective only on a particular class of elliptic curves and even the best algorithms require exponential time. Therefore, these cryptosystems are generally more secure than others. Elliptic curves could easily be applied to other cryptosystems (or combinations of cryptosystems) and as stated above, there are countless elliptic curves to choose from.

It is fairly easy to learn the dry computational steps of an elliptic curve cryptosystem, but understanding the scheme's design or implementation requires a scholarly background in mathematics. The objective of this thesis is to assemble an overview of this field of study and its findings to date, while filtering out all but the basic concepts necessary for understanding this overview.

We begin with a cursory review (it is assumed that readers have at least an undergraduate background in Computer Science) of the mathematics used in the rest of the thesis. We also introduce some concepts from the field of cryptography. Chapter 3 defines elliptic curves, their arithmetic operations, the discrete logarithm problem on an elliptic curve, and some of its properties. Chapter 4 focuses on one particular elliptic curve cryptosystem — both in theory and in practice — then proceeds to break down and analyse the components of elliptic curve cryptosystems. We conclude by summarizing the latest findings and predicting the future course of study in this seemingly inexhaustible field.

Chapter 2

Essential Concepts

Before we begin any discussion on elliptic curves or public-key cryptosystems, we will first review some basics of number theory, linear algebra, cryptography, etc. that support the ideas of the chapters that follow.

2.1 Integers

The set of all **integers** will be denoted by Z. N stands for the set of all positive integers. For a finite set A, the number of elements of A is denoted by $\#A$.

An **equivalence relation** on a set A is a binary relation $\sim$ on A such that for any $x, y, z \in A$,

1. $x \sim x$ [**reflexivity**]
2. if $x \sim y$ then $y \sim x$ [**symmetry**]
3. if $x \sim y$ and $y \sim z$ then $x \sim z$ [**transitivity**]

Let $\sim$ be an equivalence relation on a set A. Then $P = \{[a] \mid a \in A\}$, where

$[a] = \{b \in A \mid a \sim b\}$ is a **partition** of A, that is

1. for each $S \in P$, $S \neq \emptyset$
2. if $S, T \in P$, then $S = T$ or $S \cap T = \emptyset$
3. $\bigcup_{S \in P} S = A$

An element $S \in P$ is called an **equivalence class** of the partition P.

We assume the reader's familiarity with some of the most basic properties of integers.

Theorem 2.1.1 (Euclid's Division Algorithm) For $a, b \in Z$, $b \neq 0$, there exist uniquely determined $q, r \in Z$ such that

$$a = bq + r, \quad (0 \leq r < |b|)$$

[15, page 43].

If $r = 0$, we say that b is a **divisor** of a, and denote it as $b|a$. Otherwise we write $b \not| a$. For $a_1, \ldots, a_k \in Z$, if $b|a_i$ $(i = 1, \ldots, k)$, then b is called a **common divisor** of $a_1, \ldots, a_k$. The largest common divisor of $a_1, \ldots, a_k$ always exists. It is denoted by $\gcd(a_1, \ldots, a_k)$. $a, b \in Z$ are called **relatively prime** (or **coprime**) if and only if $\gcd(a, b) = 1$.

Theorem 2.1.2 If $a, b \in Z$, not both zero, then $d = \gcd(a, b)$ is the smallest element in the set of all positive integers of the form $ax + by$ $(x, y \in Z)$.

Proof Let $C = \{c \in N \mid c = ax + by,\ x, y \in Z\}$. $C \neq \emptyset$, because if $a \neq 0$, $-a \in C$. Let

$$e = ax_0 + by_0$$

be the smallest element of C. We shall show that $d = e$. If $a = eq + r$, $0 \leq r < e$, then

$$r = a - eq = a(1 - qx_0) + b(-qy_0).$$

If $r \neq 0$, it would be in C and would contradict our choice of e. Thus, $e|a$. Similarly, $e|b$, so we have $e \leq d$. On the other hand, since $e = ax_0 + by_0$ and $d|a$, $d|b$, it follows that $d|e$. Hence, $d \leq e$. Therefore, $d = e$.

Corollary 2.1.3 There exist $x, y \in Z$ satisfying

$$ax + by = c$$

if and only if $d|c$, where $d = \gcd(a, b)$.

Proof If $a = ed$, $b = fd$, then clearly $d|c$. On the other hand, if $d|c$, let $kd = c$. Since there exist $x_0, y_0 \in Z$ such that

$$ax_0 + by_0 = d$$

then

$$a(kx_0) + b(ky_0) = kd = c$$

For $a, b, m \in Z$ we define

$$a \equiv b \ mod \ m \text{ if and only if } m|(a - b).$$

We can easily see that for a fixed m, this is an equivalence relation on Z. Consequently, Z is partitioned into equivalence classes: $Z_m = \{[a] \mid a \in Z\}$, where $[a] = \{b \in Z \mid a \equiv b \ mod \ m\}$. Each equivalence class $[a]$ is often represented by its element. For example, we can write $Z_m = \{0, 1, 2, \ldots, m - 1\}$.

Theorem 2.1.4 For $a, m \in Z$, there is a $x \in Z$ such that $ax \equiv 1 \; mod \; m$ if and only if $\gcd(a, m) = 1$.

Proof There is a $x \in Z$ such that $ax \equiv 1 \; mod \; m \; \Leftrightarrow$ there are $x, y \in Z$ such that $ax - my = 1$. Therefore, Corollary 2.1.3 completes the proof.

$p \in N$ is called a **prime number** if and only if $p > 1$ and $a \not| p$ for all $a \in Z$, $1 < a < p$. Let $p \in N$, $p > 1$. p is prime if and only if for any $a, b \in Z$,

$$p|ab \;\Rightarrow\; p|a \text{ or } p|b$$

(See [15, page 46] for the proof.)

Theorem 2.1.5 (Chinese Remainder Theorem) Suppose $m_1, \ldots, m_r \in N$ are relatively prime in pairs, i.e. $\gcd(m_i, m_j) = 1$ for $i \neq j$. Let $a_1, \ldots, a_r \in Z$. Then, the system of r congruences

$$x \equiv a_i \; (mod \; m_i) \;\; (1 \leq i \leq r)$$

has a unique solution modulo $M = m_1 \times \ldots \times m_r$ given by

$$x = \sum_{i=1}^{r} a_i M_i y_i \; mod \; M$$

where $M_i = M/m_i$ and $M_i y_i \equiv 1 \; mod \; m_i$.

Proof Note that M_i is the product of all m_j where $j \neq i$. So if $j \neq i$, then $M_i \equiv 0 \; mod \; m_j$. Note also that $\gcd(M_i, m_i) = 1$, so by Theorem 2.1.4, $M_i y_i \equiv 1 \; mod \; m_i$ has a solution y_i. Thus,

$$x = \sum_{i=1}^{r} a_i M_i y_i \equiv a_i M_i y_i \equiv a_i \; mod \; m_i$$

for all i, $1 \leq i \leq r$. Therefore, x is a solution to the system of congruences.

Euler's function $\phi : N \to N$ is defined as

$$\phi(m) = \#\{k \in N \mid 1 \leq k \leq m,\ \gcd(k,m) = 1\}$$

Theorem 2.1.6

$$\phi(m) = \#\{a \in Z_m \mid ab \equiv 1 \ mod\ m \text{ for some } b \in Z_m\}$$

Proof The proof follows from Theorem 2.1.4.

Example If p is a prime number, $\phi(p) = p - 1$ and for any $a \in Z_p$, $p \not| a$, there is $b \in Z_p$ such that $ab \equiv 1 \ mod\ p$.

Suppose p is an odd prime and $x \in Z$, $1 \leq x \leq p-1$. Then x is called a **quadratic residue** modulo p if $y^2 \equiv x \ mod\ p$ has a solution $y \in Z_p$. x is a **quadratic non-residue** if x is not a quadratic residue modulo p and $x \not\equiv 0 \ mod\ p$.

2.2 Groups

A **group** is a structure consisting of a set G and a binary operation $\star$ on G (i.e. for any $a, b \in G$, $a \star b \in G$ is defined) such that:

1. $a \star (b \star c) = (a \star b) \star c$ for $a, b, c \in G$ [**associativity**]
2. there is an element $e \in G$ such that

   $$e \star a = a \star e = a \text{ for every } a \in G.$$

   This unique element e is called the **neutral element** of G.
3. for each $a \in G$ there is an element $b \in G$ such that

   $$b \star a = a \star b = e.$$

b is uniquely determined and called the **inverse** of a.

We use the notation $\langle G, \star \rangle$ to represent a group with group operation $\star$. $\langle G, + \rangle$ and $\langle G, \cdot \rangle$ are called an **additive group** and a **multiplicative group**, respectively. In an additive group, the neutral element is represented by the symbol 0 and the inverse of a is denoted as $-a$. In a multiplicative group, the neutral element is represented by the symbol 1 and the inverse of a is denoted as a^{-1}.

$\langle G, \star \rangle$ is called an **abelian** or **commutative group** if $a \star b = b \star a$ for any a, b in G .

Let $\langle G, \star \rangle$ be a group and let H be a subset of G. The structure $\langle H, \odot \rangle$ is said to be a **subgroup** of $\langle G, \star \rangle$, if $\odot$ is the restriction of $\star$ to $H \times H$ and $\langle H, \odot \rangle$ is a group.

If G is a finite group, then the number of elements of G is called the **order of** G and it is denoted as $|G|$. Given a finite multiplicative group G, the **order of an element** $a \in G$ is the smallest positive integer m such that $a^m = 1$. Such an m exists for every element in a finite multiplicative group, as follows from the next theorem and its corollary.

Theorem 2.2.1 Let G be a finite multiplicative group of order n. If the order of an element $a \in G$ is m, then

$$a^k \equiv 1 \text{ if and only if } m|k$$

Proof If $k = mq$, then $a^k = (a^m)^q = 1$. For the converse, let $k = mq + r$, $0 \leq r < m$. Then $a^r = a^k \cdot (a^{-1})^{mq} = 1$. Therefore, it follows by the minimality of m that r must be 0.

Corollary 2.2.2 If G is a finite multiplicative group of order n, then

(1) for every element $a \in G$, $a^n = 1$.

(2) the order of any element of G divides $|G|$.

If $a \in G$ is of order m, then

$$H = \{a^k \mid k \in Z\}$$

is a subgroup of G of order m. If G has an element a of order $n = |G|$, then

$$G = \{a^k \mid k \in Z\}$$

and G is called **cyclic** and a is called a **generator** of G.

The set $Z_n = \{0, 1, 2, \ldots, n-1\}$ is a cyclic group of order n under addition modulo n, i.e. $a + b \equiv r \; mod \; n$, where $r < n$ (r is the remainder when $a + b$ is divided by n).

Theorem (Euler) For $a, m \in Z$ such that $(a, m) = 1$,

$$a^{\phi(m)} \equiv 1 \; mod \; m$$

Proof By Theorem 2.1.4

$$G_m = \{a \in Z_m \mid \gcd(a, m) = 1\}$$

forms a multiplicative group of order $\phi(m)$. So this is an immediate consequence of Corollary 2.2.2 (1).

Theorem (Fermat) Let p be a prime number and $a \in Z$.

(1) $a^{p-1} \equiv 1 \; mod \; p$, if $p \not| a$.

(2) $a^p \equiv a \; mod \; p$.

Proof (1) Since $\phi(p) = p - 1$, this is a special case of Euler's Theorem. (2) This is trivial if $a \equiv 0 \; mod \; p$. Otherwise, it follows from (1).

2.3 Rings

A **ring** is a set R together with two binary operations $+$ and $\cdot$ (called addition and multiplication, respectively) defined on R such that the following conditions are satisfied :

1. $\langle R, +\rangle$ is an abelian group
2. $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ for any $a, b, c \in R$ [**associativity of** $\cdot$]
3. $a \cdot (b + c) = a \cdot b + a \cdot c$ and $(a + b) \cdot c = a \cdot c + b \cdot c$ for any $a, b, c \in R$ [**distributivity of** $\cdot$ **over** $+$]

A ring in which the multiplication $\cdot$ is commutative is called a **commutative ring**. An element e in a ring R such that $e \cdot a = a \cdot e = a$ for each $a \in R$ is a **unity element** or **multiplicative identity**, and it is represented by 1. If R has a unity element, then it is said to be a **unitary ring** or a **ring with unity element**.

2.4 Mappings

Given that $\star$ and $\odot$ are binary operations on the sets A and B respectively, a mapping $f : A \to B$ preserves the operation of A if for all $a, b \in A$ we have

$$f(a \star b) = f(a) \odot f(b).$$

Suppose A and B are two groups (or two rings). We call $h : A \to B$ a **homomorphism** of A into B if h preserves the group operation (or ring operations $+$ and $\cdot$) of A. A homomorphism h is a **monomorphism** if h is **one-to-one** (i.e. if $a \neq b$ implies that $h(a) \neq h(b)$). h is said to be a map **onto** B if $\{h(a) \mid a \in A\} = B$. A monomorphism onto B is called an **isomorphism**. If there is an isomorphism of A onto B, then we say that A and B are **isomorphic** and we write $A \simeq B$.

2.5 Fields

A **field** F is a commutative ring with unity element $e \neq 0$ such that $F^* = \{a \in F \mid a \neq 0\}$ is a multiplicative group.

Theorem The ring Z_p is a field if and only if p is a prime number.

Proof Given $a, b \in Z$, we recall the fact that

$$p \text{ is a prime number } \Leftrightarrow p|ab \text{ implies } p|a \text{ or } p|b$$

If Z_p is a field, then by definition ${Z_p}^*$ forms a multiplicative group. If $p \not| a$, then $a \not\equiv 0 \; mod \; p$. This would imply that $a \in {Z_p}^*$ and that a^{-1} exists. So if $p|ab$, and $p \not| a$ then $p|(ab)a^{-1} = b$. Therefore, p is prime.

For the converse, suppose that p is prime. It is sufficient to show that ${Z_p}^*$ is a multiplicative group, i.e. we only need to show that every $x \in {Z_p}^*$ has its multiplicative inverse. For $a, b \in Z_p$ and $x \in {Z_p}^*$,

$$\text{if } xa \equiv xb \; mod \; p \text{ then } a \equiv b \; mod \; p \Rightarrow a - b \equiv 0 \; mod \; p$$

since $p|x(a-b) \Rightarrow p|x$ or $p|a-b$ and also $x \in {Z_p}^*$ implies that $p \not| x$. This shows that $xZ_p = \{xa \mid a \in Z_p\} = Z_p$, where $xa = 1$ for some $a \in Z_p$ since there must be a neutral element 1 in Z_p. Therefore, each $x \in {Z_p}^*$ has a multiplicative inverse.

Let F be a field. A subset K of F that is also a field under the operations of F (with restriction to K) is called a **subfield** of F. In this case, F is called an

extension field of K. If $K \neq F$ then K is a **proper subfield** of F. A field is called **prime** if it has no proper subfield.

For any field F, the intersection F_0 of all subfields of F has no proper subfield, and

$$F_0 \simeq Q \ (= \text{ the field of all rational numbers})$$

or

$$F_0 \simeq Z_p, \text{ where } p \text{ is a prime number}$$

A field F is said to have **characteristic 0** if $F_0 \simeq Q$, that is, if F contains Q as a subfield. A field F is said to have **characteristic** p if $F_0 \simeq Z_p$.

A **finite field** is a field that contains only finitely many elements. Every finite field has a prime number as its characteristic [17, page 16]. In a field F of prime characteristic p, for all $a \in F$,

$$pa = \overbrace{a + \cdots + a}^{p} = 0.$$

Let F be an extension field of a field K. $F = K(\alpha)$ if F is the smallest extension field (i.e. the intersection of all extension fields) of K which contains α. If F is a finite field of characteristic p, then the multiplicative group $F^* = F \setminus \{0\}$ is cyclic and $F = Z_p(\alpha)$, where α is a generator of the group F^* (see [17, pp. 46–47] for the proof). α is called a **primitive element** of F.

2.6 Vector Spaces

Let K be a field and let V be an additive abelian group. V is called a **vector space** over K if an operation $K \times V \to V$ is defined so that the following conditions are satisfied :

1. $a(u + v) = au + av$

2. $(a + b)u = au + bu$

3. $a(bu) = (a \cdot b)u$

4. $1u = u$

The elements of V are called **vectors** and the elements of K are called **scalars**.

Let V be a vector space over a field K and let $v_1, v_2, \ldots, v_m \in V$. Any vector in V of the form

$$c_1v_1 + c_2v_2 + \cdots + c_mv_m$$

where $c_i \in K$ $(i = 1, \ldots, m)$ is a **linear combination** of $v_1, v_2, \ldots, v_m$. The set of all such linear combinations is called the **linear span** of $v_1, v_2, \ldots, v_m$ and it is denoted by $\text{span}(v_1, v_2, \ldots, v_m)$. The vectors $v_1, v_2, \ldots, v_n$ are said to **span** or **generate** V if $V = \text{span}(v_1, v_2, \ldots, v_n)$.

Let V be a vector space over a field K. The vectors $v_1, v_2, \ldots, v_m \in V$ are said to be **linearly independent** over K if there are no scalars $c_1, c_2, \ldots, c_m \in K$ (not all 0) that satisfy

$$c_1v_1 + c_2v_2 + \cdots + c_mv_m = 0$$

A set $S = \{u_1, u_2, \ldots, u_n\}$ of vectors is a **basis** of V if and only if $u_1, u_2, \ldots, u_n$ are linearly independent and they span V. If S is a basis of V, then every element of V is uniquely represented as a linear combination of the elements of S. If a vector space V has a basis of a finite number of vectors, then any other basis of V will have the same number of elements. This number is called the **dimension** of V over K.

If F is an extension field of a field K, then F is a vector space over K. The dimension of F over K is called the **degree of the extension** of F over K.

2.7 Polynomial Rings

Let F be an arbitrary ring. A **polynomial of degree** n **over** F is an expression of the form

$$f(x) = \sum_{i=0}^{n} a_i x^i = a_0 + a_1 x + \cdots + a_n x^n$$

where n is a positive integer, the coefficients $a_i \in F$ $(0 \leq i \leq n)$, and x is a symbol not belonging to F, called an **indeterminate** over F. To evaluate a polynomial $f(a)$ for some $a \in F$, we replace every instance of the indeterminate x in $f(x)$ with a. Given two polynomials

$$f(x) = \sum_{i=0}^{n} a_i x^i \text{ and } g(x) = \sum_{i=0}^{n} b_i x^i$$

we define the **sum** of $f(x)$ and $g(x)$ as

$$f(x) + g(x) = \sum_{i=0}^{n} (a_i + b_i) x^i$$

Given two polynomials

$$f(x) = \sum_{i=0}^{n} a_i x^i \text{ and } g(x) = \sum_{j=0}^{m} b_j x^j$$

we define the **product** of $f(x)$ and $g(x)$ as

$$f(x)g(x) = \sum_{k=0}^{n+m} c_k x^k, \text{ where } c_k = \sum_{\substack{i+j=k \\ 0 \leq i \leq n, 0 \leq j \leq m}} a_i b_j$$

The ring formed by all polynomials over F with ordinary operations of addition and product is called the **polynomial ring** over F and denoted by $F[x]$.

In the following, we assume that F is a field.

Theorem (Division algorithm for $F[x]$) Let $f(x), g(x) \in F[x]$ be of positive degrees. Then there exist unique polynomials $q(x), r(x) \in F[x]$ such that

$$f(x) = g(x) \cdot q(x) + r(x)$$

where the degree of $r(x)$ is less than the degree of $g(x)$ [17, page 20].

If $r(x)$ is the zero polynomial (i.e. $r(x) = 0$), then $g(x)$ is said to be a **divisor** of $f(x)$. A non-constant polynomial $f(x)$ in $F[x]$ is **irreducible** in $F[x]$ if it has no divisor of lower degree than $f(x)$ in $F[x]$. An element $a \in F$ is a **root** or **zero** of the polynomial $f(x) \in F[x]$ if $f(a) = 0$.

Corollary An element $a \in F$ is a root of the polynomial $f(x) \in F[x]$ if and only if $x - a$ is a divisor of $f(x)$ in $F[x]$.

Proof In fact, let $f(a) = 0$. Since $f(x) = (x - a) \cdot q(x) + r(x)$, then the degree of $r(x)$ is less than 1, i.e. $r(x) = c \in F$. Hence, $c = f(a) = 0$. Conversely, if $f(x) = (x - a) \cdot q(x)$, then $f(a) = 0$.

Corollary A nonzero polynomial $f(x) \in F[x]$ of degree n can have at most n roots in F [17, page 27].

2.8 Finite Fields

A field of a finite number of elements is denoted F_q or $GF(q)$, where q is the number of elements.

Proposition Let F be a finite extension of degree n over a finite field K. If K has q elements, then F has q^n elements.

Proof In fact, let $\{\alpha_1, \ldots, \alpha_n\}$ be a basis for F as a vector space over K. Then every $\beta \in F$ is uniquely represented in the form

$$\beta = c_1\alpha_1 + \cdots + c_n\alpha_n$$

where $c_i \in K$ $(i = 1, \ldots, n)$. Since each c_i may be any of q elements of K, the total number of such a linear combination is q^n.

Corollary If F is a finite field of characteristic p then F has exactly p^n elements for some positive integer n [17, page 44].

Therefore, every finite field is an extension of finite degree of a field isomorphic to Z_p, where p is a characteristic of F.

Theorem A finite field $F = F_{p^n}$ is an extension field of Z_p of degree n and every element of F_{p^n} is a root of the polynomial $x^{p^n} - x$ over Z_p.

Proof The characteristic of F_{p^n} must be p. The set $F^* = F \setminus \{0\}$ forms a multiplicative group of order $p^n - 1$ under the field multiplication. For $\alpha \in F^*$, the order of α in this group divides the order of F^*, $p^n - 1$. Therefore, for every $\alpha \in F^*$, we have $\alpha^{p^n-1} = 1$, i.e. $\alpha^{p^n} = \alpha$. Since $x^{p^n} - x$ has at most p^n roots, F_{p^n} consists of all roots of $x^{p^n} - x$ over Z_p.

Example We can see that the field F_{2^r} contains F_2 (or Z_2). If we write the addition operation in F_{2^r} as the vector addition and write the product of k and v $(k, v \in F_{2^r})$ as the scalar product kv of $k \in F_2$ and $v \in F_{2^r}$, then F_{2^r} can be viewed as a vector space over F_2 with a dimension of r. Furthermore, let d denote the dimension of this vector space. A one-to-one correspondence can be drawn between the elements (vectors) of this d-dimensional vector space and the set of all d-tuples of elements in F_2. Therefore, there must be 2^d elements in this vector space. Since $d = r$, F_{2^r} is a vector space of dimension r.

Let F_{q^m} be an extension of F_q. Two elements $\alpha, \beta \in F_{q^m}$ are **conjugate** over F_q if α and β are roots of the same irreducible polynomial of degree m over F_q. $\alpha, \alpha^q, \alpha^{q^2}, \ldots, \alpha^{q^{m-1}}$ are called the **conjugates** of $\alpha \in F_{q^m}$ with respect to F_q [17, page 49].

Let F_{q^m} be an extension field of F_q. A basis of F_{q^m} (a vector space over F_q) of the form $\{\alpha, \alpha^q, \alpha^{q^2}, \ldots, \alpha^{q^{m-1}}\}$, consisting of a suitable $\alpha \in F_{q^m}$ and its conjugates with respect to F_q, is called a **normal basis** of F_{q^m} over F_q. For every extension field of finite degree of a finite field there is a normal basis. (See [17, page 56] for the proof.)

2.9 Projective Coordinates

Consider $L = K^{n+1} \backslash \{0\}$, where K is a field. For $A = (a_0, a_1, \ldots, a_n)$, $B = (b_0, b_1, \ldots, b_n) \in L$, define a relation $A \sim B$ to mean that A, B and the origin $O = (0, 0, \ldots, 0)$ are colinear, that is, there is a $\lambda \in K$ such that

$$\lambda a_i = b_i \quad (i = 0, 1, \ldots, n).$$

This relation $\sim$ is an equivalence relation, and defines a partition of L. The quotient set is a **projective space** denoted by $P^n(K)$.

In particular, the **projective plane** is the set of equivalence classes of triples (X, Y, Z) (not all components zero) where $(\lambda X, \lambda Y, \lambda Z) \sim (X, Y, Z)$ $(\lambda \in K)$. Each equivalence class (X, Y, Z) is called a **projective point** on the projective plane. If a projective point has $Z \neq 0$, then $(x, y, 1)$ is a representative of its equivalence class where we set $x = \frac{X}{Z}, y = \frac{Y}{Z}$. Therefore, the projective plane can be defined by all the points (x, y) of the ordinary (**affine**) plane (denoted in projective coordinates as $(x, y, 1)$) plus all the points for which $Z = 0$.

2.10 Cryptography

In this section, we discuss some well-known means by which **Alice** can send a private (i.e. encrypted) message to **Bob**. The information that Alice wants to share with Bob is called the **plaintext**. The encrypted plaintext that Alice actually sends to Bob is called the **ciphertext**. A **cryptosystem** consists of a finite set of possible plaintexts, a finite set of possible ciphertexts, a finite set of possible keys, an **encryption rule** for encrypting plaintext into ciphertext and a **decryption rule** for decrypting ciphertext back to plaintext. The general idea behind any cryptosystem is that Alice and Bob must *share* a secret **key**[1] which is used to encrypt a message, and without which the plaintext cannot be recovered.

Private-key Cryptosystems If there is a way for Alice and Bob to secretly share a key K *prior* to the transmission of plaintext, they can use encryption and decryption rules defined by their secret value of K. Cryptosystems of this form are called **private-key cryptosystems**. One approach to sharing keys is the **key agreement protocol** whereby Alice and Bob jointly establish the secret key by using values they have sent each other over a public channel.

In these systems, the decryption rule is identical to or easily derived from the encryption rule. Hence, exposure of the encryption rule to an eavesdropper will render the system insecure.

Public-key Cryptosystems The security of private-key systems depends on the secret exchange or establishment of keys between Alice and Bob. However, in **public-key cryptosystems** Bob keeps his key (and his decryption rule) to himself,

[1] The range of possible key values is called the **keyspace**.

whereas the corresponding encryption rule is publicly known. Therefore, Alice can send encrypted messages without any prior sharing of keys, and Bob will be the only person able to decrypt the messages sent to him.

2.10.1 The Discrete Logarithm Problem

For some group G, suppose $\alpha, \beta \in G$. Solving for an integer x such that $\alpha^x = \beta$ is called the **discrete logarithm problem (DLP)**. The DLP in Z_p is considered difficult (or **intractible**) if p has at least 150 digits and $p-1$ has at least one large prime factor (as close to p as possible). These criteria for p are safeguards against the known attacks on DLP. [33, page 162]

Numerous cryptosystems base their security on the difficulty of solving the DLP. One such public-key cryptosystem is the **El Gamal Cryptosystem** in ${Z_p}^*$ [33, page 163] which is presented in Figure 2.1. An attacker could decrypt Alice's message if Bob's secret key a_B could be computed from $\beta \equiv \alpha^{a_B}$ $(mod\ p)$ and α which are publicly known. This is the DLP.

The decryption rule can be explained as follows:

$$y_2({y_1}^{a_B})^{-1} \equiv x\beta^k(\alpha^{ka_B})^{-1} \equiv x\alpha^{a_B k}(\alpha^{-ka_B}) \equiv x\ mod\ p$$

The Diffie-Hellman Key Exchange [33, page 271] also involves the DLP. It is a key agreement protocol that is described in Figure 2.2. An eavesdropper, Oscar, could intercept $\alpha^{a_A}\ mod\ p$ and $\alpha^{a_B}\ mod\ p$; the security of this protocol is based on the (yet unproven/disproven) assumption that computing $K = \alpha^{a_A a_B}\ mod\ p$ from those intercepted values is *as hard* as obtaining x from $\alpha^x = \beta$ (i.e. the DLP). Oscar could also attempt to derive a_A or a_B from $\alpha^{a_A}\ mod\ p$ and $\alpha^{a_B}\ mod\ p$, respectively, then compute the key just as Alice or Bob would, but such computations would

Let p be a prime such that the DLP in Z_p is intractible, and let $\alpha \in {Z_p}^*$ be a primitive element. p and α are publicly known. Each user X chooses a secret key a_X (an integer, where $0 \leq a \leq p-2$) and publishes β where $\beta \equiv \alpha^{a_X}$ $(mod\ p)$.

For Alice to send her message $x \in {Z_p}^*$, she must choose a random number $k \in Z_{p-1}$ and send

$$(y_1, y_2) = (\alpha^k\ mod\ p,\ x\beta^k\ mod\ p)$$

To decrypt, the recipient Bob computes

$$y_2({y_1}^{a_B})^{-1}\ mod\ p$$

where a_B is his secret key.

Figure 2.1: The El Gamal Cryptosystem

be instances of the DLP. Therefore, this protocol is secure as long as the DLP is intractible.

There are several algorithms for solving the DLP, though none of them perform in polynomial time. **Shanks' algorithm** and the **Pohlig-Hellman algorithm** are among the strongest attacks, and they are presented in Figure 2.3 and Figure 2.4, respectively [33, pp. 165–170]. In both cases, we assume that p is prime and that α is a primitive element of Z_p. Given $\beta \in {Z_p}^*$, our goal is to find x $(0 \leq x \leq p-2)$ where $\alpha^x \equiv \beta$ $(mod\ p)$.

Let p be a (large) prime and assume that α is a primitive element of Z_p. p and α are publicly known.

1. Alice chooses a_A $(0 \leq a_A \leq p-2)$ at random.

2. Alice computes α^{a_A} *mod* p and sends it to Bob.

3. Bob chooses a_B $(0 \leq a_B \leq p-2)$ at random.

4. Bob computes α^{a_B} *mod* p and sends it to Alice.

5. Alice computes $K = (\alpha^{a_B})^{a_A}$ *mod* p

   whereas Bob computes $K = (\alpha^{a_A})^{a_B}$ *mod* p

In other words, both Alice and Bob compute the same key

$$K = \alpha^{a_A a_B} \; mod \; p$$

Figure 2.2: The Diffie-Hellman Key Exchange

2.10.2 Factoring

There are also a number of cryptosystems whose security is based on the difficulty of factoring large integers. One well-known example is the public-key system called the **RSA Cryptosystem** [28, 33]. It is presented in Figure 2.5. Note that Bob can compute $a = b^{-1} \; mod \; \phi(n)$ from b by using the **Extended Euclidean Algorithm** [33, page 119] presented in Figure 2.6.

For $x \in {Z_n}^*$, the decryption rule can be verified as follows: since $ab \equiv 1 \; (mod \; \phi(n))$, we can represent ab as $ab = k \cdot \phi(n) + 1$ for some integer $k \geq 1$. Then

$$\begin{aligned} y^a &\equiv (x^b)^a \; (mod \; n) \\ &\equiv x^{k \cdot \phi(n)+1} \; (mod \; n) \\ &\equiv (x^{\phi(n)})^k x \; (mod \; n) \\ &\equiv 1^k x \; (mod \; n) \\ &\equiv x \; (mod \; n) \end{aligned}$$

For RSA to be secure, it should be computationally infeasible to factor $n = pq$ even when using the best factoring algorithms, i.e. p and q should be sufficiently large. If p and q are known, it is easy to compute $\phi(n) = (p-1)(q-1)$ and derive a. At present, it is recommended that p and q should each be primes having around 100 digits [33, page 126]. However, it should be noted that there are also a number of attacks on RSA that do not involve the factoring of n at all. They generally exploit weaknesses in the setup of the cryptosystem, such as poor choices of a, or Bob's usage of the same n to communicate with other people. For further information, see [28, 33].

Set $m = \lceil \sqrt{p-1} \rceil$.

1. Compute α^{mj} *mod* p, where $0 \leq j \leq m-1$
2. Sort the m ordered pairs $(j, \alpha^{mj}$ *mod* $p)$ with respect to the second coordinates, producing a list L_1
3. Compute $\beta\alpha^{-i}$ *mod* p, where $0 \leq i \leq m-1$
4. Sort the m ordered pairs $(i, \beta\alpha^{-i}$ *mod* $p)$ with respect to the second coordinates, producing a list L_2
5. Find $(j, y) \in L_1$ and $(i, y) \in L_2$, i.e. pairs with identical second coordinates
6. Define $x = \log_\alpha \beta = mj + i$ *mod* $(p-1)$

Figure 2.3: Shanks' Algorithm for the DLP in Z_p

Suppose we factorize $p-1$:

$$p-1=\prod_{i=1}^{n} {q_i}^{c_i}$$

(the q_i's are distinct primes). For each q_i $(1 \le i \le n)$ we compute $a_0, \ldots, a_{c_i-1}$ where

$$\log_\alpha \beta \; mod \; {q_i}^{c_i} = \sum_{k=0}^{c_i-1} a_i {q_i}^k$$

using the pseudo-code below:

1. compute $\gamma_j = \alpha^{(p-1)j/q_i} \; mod \; p$ for $0 \le j \le q_i - 1$
2. set $k = 0$ and $\beta_k = \beta$
3. **while** $k \le c_i - 1$ **do**
   (a) compute $\delta = {\beta_k}^{(p-1)/{q_i}^{k+1}} \; mod \; p$
   (b) find j such that $\delta = \gamma_j$
   (c) $a_k = j$
   (d) $\beta_{k+1} = \beta_k \alpha^{-a_k {q_i}^k} \; mod \; p$
   (e) $k = k + 1$

Finally, we use the Chinese Remainder Theorem to solve the system of congruences $\log_\alpha \beta \; mod \; {q_i}^{c_i}$ $(1 \le i \le n)$. This gives us $\log_\alpha \beta$ modulo $\prod_{i=1}^{n} {q_i}^{c_i}$, i.e. $\log_\alpha \beta \; mod \; (p-1)$.

Figure 2.4: The Pohlig-Hellman Algorithm for the DLP in Z_p

Bob secretly chooses two primes, p and q, and publishes $n = pq$. Next, he randomly chooses b such that b and $\phi(n) = (p-1)(q-1)$ are relatively prime. Bob computes a such that $ab \equiv 1\ (mod\ \phi(n))$. a is his secret key, whereas b is revealed to the public.

Alice encrypts her plaintext message $x \in Z_n$ by computing

$$y = x^b\ mod\ n$$

and sends y to Bob.

Bob retrieves x by computing

$$y^a\ mod\ n$$

Figure 2.5: The RSA Cryptosystem

$n_0 = n$, $b_0 = b$, $t_0 = 0$, $t = 1$

$r = n_0$ div b_0

while $r > 0$ do

$\quad temp = t_0 - \lfloor \frac{n_0}{b_0} \rfloor \times t$

$\quad t_0 = t$, $t = temp$, $n_0 = b_0$, $b_0 = r$

$\quad r = n_0$ div b_0

If $b_0 \neq 1$ then b has no inverse modulo n, otherwise $b^{-1} = t\ mod\ n$.

Figure 2.6: The Extended Euclidean Algorithm for computing b^{-1} modulo n

Chapter 3

Elliptic Curves

Now we are ready to discuss elliptic curves and their various properties. The notation we present here will apply to the remainder of this thesis.

3.1 Introduction to Elliptic Curves

We begin with the definition of an elliptic curve.

Let K be a field. For example, K can be the finite (extension) field $\boldsymbol{F_{q^r}}$ of $\boldsymbol{F_q}$, the prime field $\boldsymbol{Z_p}$ where p is a (large) prime, the field $\boldsymbol{R}$ of real numbers, the field $\boldsymbol{Q}$ of rational numbers, or the field $\boldsymbol{C}$ of complex numbers.

Definition An elliptic curve over a field K is defined by the **Weierstrass equation**:

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \tag{3.1}$$

where $a_1, a_3, a_2, a_4, a_6 \in K$.

The elliptic curve E over K is denoted $E(K)$. The number of points on E (the cardinality) is denoted $\#E(K)$ or just $\#E$.

For fields of various characteristics, the Weierstrass equation can be transformed (and simplified) into different forms by a linear change of variables. We present the equations for fields of characteristic $\neq 2, 3$ and of characteristic 2. (The equation for a field of characteristic 3 was omitted since it is not central to the discussions in the remaining chapters.)

[**Characteristic** $\neq 2, 3$] Let K be a field of characteristic $\neq 2, 3$, and let $x^3 + ax + b$ (where $a, b \in K$) be a cubic polynomial with the condition that $4a^3 + 27b^2 \neq 0$ (this ensures that the polynomial has no multiple roots). An **elliptic curve** E **over** K is the set of points (x, y) with $x, y \in K$ that satisfy the equation

$$y^2 = x^3 + ax + b \tag{3.2}$$

and also an element denoted O and called the **point at infinity** (to be described in greater detail below).

[**Characteristic 2**] If K is a field of characteristic 2, then there are two types of elliptic curves:

An **elliptic curve of zero j-invariant**[1] is the set of points satisfying

$$y^2 + a_3 y = x^3 + a_4 x + a_6 \tag{3.3}$$

(where $a_3, a_4, a_6 \in F_q,\ a_3 \neq 0$) and O, the point at infinity. (It does not matter in this case whether the cubic on the right side of the equation has multiple roots or not.)

An **elliptic curve of nonzero j-invariant** is the set of points satisfying

$$y^2 + xy = x^3 + a_2 x^2 + a_6 \tag{3.4}$$

[1] The **j-invariant** of E over K is an element of K determined by a_1, a_2, a_3, a_4 and a_6. See [32, pp. 48–52] for further detail.

(where $a_2, a_6 \in F_q,\ a_6 \neq 0$) and O, the point at infinity.

The Point at Infinity The **line at infinity** is the collection of points on the projective plane for which $Z = 0$. The **point at infinity** is the point of intersection where the y-axis and the line at infinity meet. More precisely, the point at infinity is $(0, 1, 0)$ in the projective plane (the equivalence class with $X = Z = 0$).

An elliptic curve E over a finite field K can be made into an abelian group by defining an additive operation on its points. The operation is defined in the next section.

3.2 The Rules for Addition

Given two points $P, Q \in E(K)$ we define a third point $P + Q$ so that $E(K)$ forms an abelian group with this addition operation. If $P \neq Q$, then the line connecting P and Q intersects $E(K)$ in a uniquely determined point which we denote as PQ. If $P = Q$ then the tangent of $E(K)$ at P gives rise to the point PQ. It is tempting to take PQ as $P + Q$, but it would not define a group structure since there is no neutral element in this case. Therefore, we find a point of intersection where $E(K)$ meets the line connecting PQ and the point at infinity O, and call this point $P+Q$. By joining O to a point PQ on the affine part of $E(K)$, we mean that a vertical line is drawn through PQ. A vertical line intersects $E(K)$ at 3 points: (x, y), $(x, -y)$ and O. Hence, the point at infinity O serves as the additive identity element and $P + Q + PQ = O$ or $P + Q = -PQ$, the **inverse** of PQ. Figure 3.1 illustrates these concepts on the elliptic curve $y^2 = x^3 - x$, plotted in the xy-plane[2].

[2] The curve was drawn using Gnuplot v3.5 and Xfig v3.1

Figure 3.1: Adding points P and Q

For each of the three cases of elliptic curves described above, the algebraic formulas which represent $P+Q$ are easily derived from the following geometric procedures[3]:

The Addition Formula for 3.2 The inverse of $P=(x_1,y_1)\in E$ is $-P=(x_1,-y_1)$. If $Q\neq -P$, then $P+Q=(x_3,y_3)$ where

$$\begin{aligned} x_3 &= \lambda^2 - x_1 - x_2 \\ y_3 &= \lambda(x_1 - x_3) - y_1 \end{aligned}$$

where

If $P \neq Q$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$$

If $P = Q$

$$\lambda = \frac{3{x_1}^2 + a}{2y_1}$$

The Addition Formula for 3.3 The inverse of $P=(x_1,y_1)\in E$ is $-P=(x_1,y_1+a_3)$. If $Q\neq -P$, then $P+Q=(x_3,y_3)$ where

If $P \neq Q$

$$\begin{aligned} x_3 &= \left(\frac{y_1+y_2}{x_1+x_2}\right)^2 + x_1 + x_2 \\ y_3 &= \left(\frac{y_1+y_2}{x_1+x_2}\right)(x_1+x_3) + y_1 + a_3 \end{aligned}$$

[3]See [32, pp. 55–63] for further discussion of these addition formulas.

If $P = Q$

$$\begin{aligned} x_3 &= \left(\frac{{x_1}^4 + {a_4}^2}{{a_3}^2}\right) \\ y_3 &= \left(\frac{{x_1}^2 + a_4}{a_3}\right)(x_1 + x_3) + y_1 + a_3 \end{aligned}$$

The Addition Formula for 3.4 The inverse of $P = (x_1, y_1) \in E$ is $-P = (x_1, y_1 + x_1)$. If $Q \neq -P$, then $P + Q = (x_3, y_3)$ where

If $P \neq Q$

$$\begin{aligned} x_3 &= \left(\frac{y_1 + y_2}{x_1 + x_2}\right)^2 + \left(\frac{y_1 + y_2}{x_1 + x_2}\right) + x_1 + x_2 + a_2 \\ y_3 &= \left(\frac{y_1 + y_2}{x_1 + x_2}\right)(x_1 + x_3) + x_3 + y_1 \end{aligned}$$

If $P = Q$

$$\begin{aligned} x_3 &= \left(\frac{a_6}{{x_1}^2}\right) + {x_1}^2 \\ y_3 &= {x_1}^2 + \left(x_1 + \frac{y_1}{x_1}\right)x_3 + x_3 \end{aligned}$$

Theorem The addition operation defined above turns $E(K)$ into an abelian group that has O as the identity element [32, pp. 55–57]. (This is not too difficult to prove except for the step where we must show associativity.)

3.3 The Discrete Logarithm Problem

Exponentiation and Logarithm Since an elliptic curve E is made into an abelian group by an additive operation (as opposed to a multiplicative one), "the exponentiation of a point on E" actually refers to repeated addition. Therefore, the ith power of $\alpha \in E$ is ith multiple of α, i.e. $\beta = \alpha^i = i\alpha$. The **logarithm** of β to the base α would be i, the inverse of exponentiation.

The Discrete Logarithm Problem For some group G, suppose $\alpha, \beta \in G$. Recall that in the **discrete logarithm problem (DLP)** we solve for an integer x such that $\alpha^x = \beta$. Analogously, in the **elliptic curve discrete logarithm problem (EDLP)** we solve for an integer x such that $x\alpha = \beta$ given $\alpha, \beta \in E$. For the EDLP over $E(F_q)$ to be intractible, it is important to select an appropriate E and q such that $\#E(F_q)$ is divisible by a large prime (of more than 30 digits [22]) or such that q is itself a large prime [23]. The elliptic curve cryptosystems described in the next chapter are dependent on the presumed intractibility of the EDLP. It is believed that the EDLP is more intractible than the DLP since some of the strongest algorithms for solving the DLP cannot be adapted to the EDLP.

3.4 Computing $\#E(K)$

Elliptic curve cryptosystems generally involve the selection of a suitable elliptic curve E and a point P on E called the **base point**. To learn more about the structure of the group $E(K)$ (hence to make a wise selection), it is useful to know the exact value of $\#E(K)$. We will look at the case when K is F_q, a finite field of q elements. The following results are the best known methods to date for computing $\#E$.

Hasse's Theorem Let N be the number of points on an elliptic curve over F_q, a finite field with q elements. Then

$$|N - (q+1)| \leq 2\sqrt{q}$$

Stated in another way, Hasse's Theorem gives the estimate $\#E(F_q) = q+1-t$ where $|t| \leq 2\sqrt{q}$. [9, 12]

The Weil Conjecture In 1949, Weil made a series of conjectures in a general context regarding algebraic varieties (geometric objects) defined over finite fields. For the case of elliptic curves, Deligne proved the conjectures (now a theorem) in 1973, although the particular conjecture we present below was proved for elliptic curves in 1934 by Hasse [12, 32].

Let $t = q + 1 - \#E(F_q)$. Then

$$\#E(F_{q^k}) = q^k + 1 - \alpha^k - \beta^k$$

where $1 - tx + qx^2 = (1 - \alpha x)(1 - \beta x)$. In other words, it is possible to compute $\#E(F_{q^k})$ given $\#E(F_q)$. [10, 20]

Schoof's Algorithm In 1985, Schoof presented a deterministic algorithm that could compute $\#E(F_q)$ (its precise value; not a bound or an estimate) in $O(\log^9 q)$ bit operations (where F_q is a finite field of characteristic $\neq 2, 3$) [29]. This deterministic polynomial time algorithm is the fastest to date[4], and given few alternatives, it is the best choice for computing $\#E$. But in practice, it is awkward and costly to implement, particularly when q is large. The implementation of Schoof's algorithm is discussed at the end of Chapter 4.

These are the basic properties of elliptic curves that provide the seed for the concept of elliptic curve cryptosystems.

[4] Some improvements have been suggested very recently for Schoof's algorithm in [16].

Chapter 4

Elliptic Curve Cryptosystems

Finally, we are ready to discuss **elliptic curve cryptosystems**. Unlike earlier cryptosystems, an elliptic curve cryptosystem works with a finite abelian group formed by the points on an elliptic curve over a finite field.

4.1 History

In 1976, Diffie and Hellman [7] introduced a cryptographic protocol whose security over insecure communication channels was based on the presumed intractibility of the DLP. In other words, they had introduced the notion of a **trapdoor one-way function** or TOF. A TOF is easy to evaluate but computing the inverse without a secret "trapdoor" is an intractible problem. In 1985, Lenstra succeeded at using elliptic curves for integer factorization. This result suggested the possibility of applying elliptic curves to public-key cryptosystems.

Miller and Koblitz were the first to propose cryptosystems that employed elliptic curves. They did not invent new cryptographic algorithms but they were the first to implement existing public-key cryptosystems using elliptic curves. (Miller proposed an analogue of the Diffie-Hellman key exchange protocol[1] in 1985 [21]. Koblitz presented analogues of the El Gamal and Massey-Omura cryptosystems in 1987 [13].)

The first analogue of the RSA scheme and three new TOFs based on elliptic curves were introduced in 1991, by Koyama, Maurer, Okamoto and Vanstone [14]. (The analogue of RSA is computationally less efficient than RSA — operating at 1/6 the speed of RSA. Its security, as with the original RSA scheme, depends greatly on the difficulty of integer factorization. However, the analogue is more secure than the RSA scheme in terms of attacks that are not based on factoring. For example, the analogue is secure against the **Low Multiplier Attack** which can otherwise exploit RSA's weakness when the same plaintext is encrypted with several distinct moduli [14].)

Around the same time, Kaliski observed that elliptic curves could offer one-way functions that appear to require exponential time for inversion [11], while Menezes, Okamoto and Vanstone discovered the MOV reduction method for solving the EDLP in specific cases. Soon after, Miyaji found the conditions for an elliptic curve to be immune to the MOV attack [23] and proposed the real-world application of elliptic curves to the signature and identification schemes of smart cards [22]. In 1993, Demytko presented a new analogue of RSA based on elliptic curves over a ring Z_n that overcame the limitations of earlier versions [6], and Menezes and Vanstone proposed hardware implementations that would improve elliptic curve computations over finite fields [20]. Recently, the notion of con-

[1] The analogue of the Diffie-Hellman scheme appears to be around 20% faster than the Diffie-Hellman key exchange protocol.

structing elliptic curves for a cryptosystem (instead of randomly choosing one) has become a serious concern, as can be seen in [5].

4.2 Analogue of the El Gamal Cryptosystem

Since "elliptic curve cryptosystem" is a generic term for any cryptosystem that works in the domain of elliptic curves, we will illustrate the meaning of that term by focusing on one particular example: the analogue of the El Gamal cryptosystem.

Since the El Gamal protocol (see Figure 2.1) can be generalized to work in an arbitrary finite cyclic group, the analogue implemented on an elliptic curve (as proposed by Koblitz in 1987) over the field Z_p can be described as in Figure 4.1 [12, 13]. We discuss **imbedding** and the computation of the multiple $kP \in E(Z_p)$ below.

When we **imbed** plaintext on an elliptic curve E, we are representing the plaintext as points on E so that we may perform our computations in E. Note that imbedding is performed *prior* to encryption (this is not part of the encryption step, as demonstrated in the analogue of El Gamal).

Example Here is one probabilistic method of imbedding[2] a plaintext m on $E(Z_p)$, where p is a prime such that $p \equiv 3 \ (mod\ 4)$. Suppose that $E(Z_p)$ is given by equation 3.2 and the plaintexts m are integers such that $0 \leq m < p/1000 - 1$. Appending three digits to m will produce a value x such that $1000m \leq x < 1000(m + 1) < p$. We try appending different digits until we find an x such that $f(x) = x^3 + ax + b$

[2] This is a modified version of an example presented in [13].

We are given a prime field Z_p, an elliptic curve $E(Z_p)$, and a base point $P \in E$, all of which are fixed and publicly known. Each user X of this system chooses a random integer a_X which will be his/her own secret key, then computes and publishes the point $a_X P$.

Suppose Alice wishes to send a message m (an integer, let's say) to Bob. First, she imbeds the value m onto the elliptic curve E, i.e. she represents the plaintext m as a point $P_m \in E$. Now she must encrypt P_m. Let a_B denote Bob's secret key (so, $a_B P$ will be publicly known). Alice first chooses a random integer k and sends Bob a pair of points on E:

$$(C_1, C_2) = (kP, P_m + k(a_B P))$$

To decrypt the ciphertext, Bob computes

$$C_2 - a_B(C_1) = P_m + k(a_B P) - a_B(kP) = P_m$$

Figure 4.1: Analogue of the El Gamal Cryptosystem

is a square in Z_p and y (where $f(x) \equiv y^2 \ mod \ p$) satisfies $y \not\equiv -1 \ mod \ p$. Then, we define the imbedded point corresponding to m as

$$P_m = (x, f(x)^{\frac{(p+1)}{4}})$$

Let $z = f(x) = x^3 + ax + b \equiv y^2 \ mod \ p$. Then P_m is a point on $E(Z_p)$ (i.e. $z^{\frac{(p+1)}{4}} \equiv y \ mod \ p$) for the following reasons:

Since $p \equiv 3 \ (mod \ 4)$, we can write $p = 4k + 3$. Then

$$z^{\frac{(p+1)}{4}} \equiv y^{\frac{(p+1)}{2}} = y^{2k+2} \ mod \ p$$

If $y \equiv 0$ or $y \equiv 1 \ mod \ p$, then clearly $z^{\frac{(p+1)}{4}} \equiv y^{2k+2} \equiv y \ mod \ p$. Otherwise, let m be the order of $y \ mod \ p$ in the group $Z_p{}^*$. By Fermat's Theorem,

$$y^{p-1} = y^{4k+2} \equiv 1 \ mod \ p$$

hence $m|4k + 2 = 2(2k + 1)$. Since $y^2 \not\equiv 1 \ mod \ p$, it follows that $m|2k + 1$. Therefore, $y^{2k+1} \equiv 1 \ mod \ p$. Thus, by Fermat's Theorem again,

$$z^{\frac{(p+1)}{4}} \equiv y^{2k+2} \equiv y^{4k+3} \equiv y^p \equiv y \ mod \ p$$

We can easily retrieve a plaintext m from a point $P_m \in E(Z_p)$, by simply dropping the last three digits from the x-coordinate of P_m. $f(x)$ is a square for roughly $\frac{1}{2}$ of all x [12, page 163] since there is an equal number of quadratic residues and quadratic non-residues $mod \ p$. Therefore, the probability that $f(x)$ will not be a square is very small (around $\frac{1}{2^{1000}}$ since $1000m \leq x < 1000(m + 1)$).

$kP \in E(Z_p)$, where k is an integer, can be computed by adding the base point k times (a simple but tedious approach), or it could be found in $O(\log k \log^3 p)$ bit operations by using the **double-and-add algorithm**[3] which is described in Figure 4.2:

[3] analogous to the **square-and-multiply algorithm** for raising an element to the k-th power

Let $k_0, k_1, \ldots, k_{m-1}$ denote the binary digits of k, such that $k = k_0 2^0 + k_1 2^1 + k_2 2^2 + \cdots k_{m-1} 2^{m-1}$ (i.e. $k_i = 0$ or 1, and $k_{m-1} = 1$ is the most significant bit). Set $P_x = nil$ and $P_y = P$.

for $i = 0$ to $m - 1$

  if $k_i = 1$

    if $P_x = nil$ then $P_x = P_y$

    else $P_x = P_x + P_y$

  double P_y, i.e. set $P_y = P_y + P_y$

The resulting value of P_x is kP.

Figure 4.2: The Double-and-Add Algorithm

Security If an eavesdropper, Oscar, can solve the EDLP, then he could determine Bob's secret key a_B from the publicly known information P and $a_B P$ and consequently read Alice's message. Clearly, the security of the analogue system relies heavily on the intractibility of the EDLP, just as the original El Gamal cryptosystem relies on the intractibility of the DLP. In turn, the intractibility of the EDLP clearly depends on the choice of the elliptic curve E and the base point $P \in E$. Methods for selecting a suitable E and P are analysed at the end of this chapter.

Unlike some other cryptosystems (the analogue of the Massey-Omura system, for example), this scheme has the advantage that the value of $\#E(F_q)$ is not required in its computations. However, the latter cryptosystem has a message expansion factor[4] of 4, as opposed to the message expansion factor of 2 of the former

[4]This is the ratio of the number of field elements sent as the ciphertext to the number of field elements in the

cryptosystem.

A variant of the El Gamal analogue is the **Menezes-Vanstone Elliptic Curve Cryptosystem** [20, 33]. The difference between the Analogue of El Gamal presented above and this scheme is that Alice will "mask" her plaintext instead of "imbedding" it (this will be explained later in greater detail). Figure 4.3 describes the Menezes-Vanstone Cryptosystem.

The decryption rule can be explained as follows : since $y_0 = kP$, Bob can compute

$$a_B y_0 = a_B(kP) = k(a_B P) = (c_1, c_2)$$

and then

$$y_1 {c_1}^{-1} \equiv (c_1 x_1) {c_1}^{-1} \equiv x_1 \ mod\ p$$

$$y_2 {c_2}^{-1} \equiv (c_2 x_2) {c_2}^{-1} \equiv x_2 \ mod\ p$$

4.3 Sample Implementation

We have chosen to implement the Menezes-Vanstone Elliptic Curve Cryptosystem due to the conveniences that stem from "masking" vs. "imbedding" plaintext (explained in the next section). We use the elliptic curve E defined by

$$y^2 = x^3 + x + 13$$

over the prime field Z_{31} (i.e. $p = 31$). Therefore, E is over a field of characteristic $\neq 2, 3$ as in equation 3.2. We also fixed the base point to be $P = (9, 10)$. The underlying field of E is not large in cardinality, but we have used it for the sake of simplicity. As it turns out, $\#E(Z_{31}) = 34$ and P is an element of order 34

original plaintext.

Let E be an elliptic curve over the prime field Z_p ($p > 3$) such that E contains a cyclic subgroup H in which the EDLP is intractible. Z_p, $E(Z_p)$, and a base point $P \in E$ (preferably a generator of E), are fixed and publicly known. Each user X chooses a random integer a_X which will be his/her own secret key, then computes and publishes the point $a_X P$.

Suppose Alice wishes to send a message $M = (x_1, x_2) \in {Z_p}^* \times {Z_p}^*$ to Bob. Let a_B denote Bob's secret key. Alice chooses a random integer $k \in Z_{|H|}$ and sends

$$(y_0, y_1, y_2) = (kP,\ c_1 x_1 \ mod\ p,\ c_2 x_2 \ mod\ p)$$

where $(c_1, c_2) = k(a_B P)$.

To decrypt the ciphertext, Bob computes

$$(y_1 {c_1}^{-1} \ mod\ p,\ y_2 {c_2}^{-1} \ mod\ p) = (x_1, x_2)$$

where $a_B y_0 = (c_1, c_2)$.

Figure 4.3: The Menezes-Vanstone Elliptic Curve Cryptosystem

(these values were drawn from [33, page 201], though they are not required in the operation of this particular cryptosystem). All the points in E are listed in Table 4.1.

k	kP	k	kP	k	kP	k	kP	k	kP	k	kP
1	(9,10)	7	(6, 24)	13	(27, 10)	19	(5, 22)	25	(16, 23)	31	(23, 12)
2	(18, 29)	8	(24, 29)	14	(26, 21)	20	(26, 10)	26	(24, 2)	32	(18, 2)
3	(23, 19)	9	(16, 8)	15	(5, 9)	21	(27, 21)	27	(6, 7)	33	(9, 21)
4	(4, 22)	10	(20, 2)	16	(19, 3)	22	(28, 18)	28	(17, 13)	34	O
5	(25, 16)	11	(22, 22)	17	(10, 0)	23	(22, 9)	29	(25, 15)		
6	(17, 18)	12	(28, 13)	18	(19, 28)	24	(20, 29)	30	(4, 9)		

Table 4.1: The Points in $E(Z_{31})$

Since we are masking plaintext instead of imbedding it, the plaintext space is ${Z_{34}}^* \times {Z_{34}}^*$. Each plaintext (x_1, x_2) represents two alphabetic characters in this case, and "a" corresponds to 1, "b" to 2, "c" to 3, …, "z" to 26 (0 is avoided since it is not allowed in the plaintext). Inverses modulo p were computed using the Extended Euclidean Algorithm that was described in Figure 2.6. Multiples kP of a point $P \in E$ were computed using the double-and-add algorithm.

A sample output of the program GAMAL.C[5] is shown in Figure 4.4. Note that we have printed out each important step in the encryption and decryption process. The lines of input are marked with % .

[5]The source code for this implementation is provided on the World Wide Web at ftp://ftp-cgrl.cs.mcgill.ca/pub/crypto/saeki/gamal.c. It was written in C and tested using Turbo C++ ©1990, 1992, version 3.0.

```
Bob: Enter your secret key

% 12

Bob's public key = (28,13)

Alice: Please enter your message

% crypto

Alice: Chose k=7

Alice: Now sending ciphertext((6,24), 26, 23)

Alice: Chose k=29

Alice: Now sending ciphertext((25,15), 11, 30)

Alice: Chose k=1

Alice: Now sending ciphertext((9,10), 2, 9)

Decryption starting

Bob: Reading Alice's message

crypto
```

Figure 4.4: Sample Output of GAMAL.C

The encryption and decryption steps are straightforward and easy to implement. Our program could be used with any elliptic curve defined by equation 3.2, and it could also be adapted to other types of elliptic curves. The program's performance could also be improved by applying the various techniques described in the next section.

However, this alone is not enough to ensure the security of the cryptosystem. To preclude any attacks, the program should be preceded by an algorithm for selecting an elliptic curve with secure properties, i.e. a curve where $\#E$ has a large prime factor or is itself a large prime. Therefore, we are compelled to compute the value of $\#E$, as discussed (more thoroughly) in section 4.4.3.†

† It should be noted that the El Gamal algorithm is unpatented but Public Key Partners (PKP) dubiously considers it to be covered under the Diffie-Hellman patent[6] which will expire on April 29, 1997, making it the first public-key cryptography algorithm (for encryption and digital signatures) unencumbered by patents in the United States.[28, page 479]

[6] Hellman, M.E., Diffie, W., Merkle, R.C., "Cryptographic Apparatus and Method," U.S. Patent #4,200,770, 29 Apr 1980.

4.4 Analysis of Techniques

Let us now analyse some of the better known techniques that can enhance the implementation and security of an elliptic curve cryptosystem. We shall draw examples from the sample implementation above.

4.4.1 Software/Hardware Optimization Techniques

There are various ways of simplifying the computations involved in an elliptic curve cryptosystem. These tricks and shortcuts can speed up the computations or reduce storage requirements for intermediate results. Unfortunately, one improvement comes at the expense of the other, so one must weigh the importance of speed versus space before implementing these techniques.

Imbedding vs. Masking Plaintext There are basically two ways of representing plaintext in an elliptic curve cryptosystem. Imbedding (or "embedding") plaintext on an elliptic curve E is one way. The other way is to use an elliptic curve to "mask" the plaintext.

Imbedding We face three key issues when choosing to imbed our plaintext. The first is that users will want a simple system of imbedding such that the relationship between the plaintext and its corresponding point on the elliptic curve is clear. It should be easy for any authorized user to convert back and forth between the plaintext (integers) and the coordinates of the points on E. Secondly, when we make these conversions from plaintext to points on E, we need a fast, systematic way of generating these imbedded points on E. And finally, there aren't any deterministic polynomial time algorithms for imbedding a *large* number of points on an arbitrary elliptic curve E over F_q.[12, page 163]

Masking To **mask** an ordered pair of elements (m_1, m_2) with an elliptic curve means to alter the pair by multiplying m_1 and m_2 with the x and y coordinate, respectively, of some point on the curve. In the case of the Menezes-Vanstone Elliptic Curve Cryptosystem, we are masking the pair of plaintexts $M = (x_1, x_2)$ with the point $(c_1, c_2) = k(a_B P)$. Although $a_B P$ is publicly known, the masking point is protected from eavesdroppers by the secret value k, which thereby protects the plaintext as well. Consequently, plaintexts and ciphertexts are not required to be imbedded as points on an elliptic curve: they can be any ordered pair of (nonzero) field elements. In the sample implemention, the plaintext space is ${Z_{31}}^* \times {Z_{31}}^*$, allowing $900 = 30 \times 30$ plaintexts. If we had used an imbedding algorithm, we would be restricted to just $\#E(Z_{31}) = 34$ plaintexts. Masking instead of imbedding kept the cryptosystem simple, and also saved us some valuable computing time. Masking does not appear to be any more or less secure than imbedding since both methods rely on the EDLP for security.[20, 33]

Affine vs. Projective Coordinates Projective coordinates (or *homogeneous coordinates*) have the distinct advantage of being able to explicitly represent the point at infinity as $(0, 1, 0)$. They also make it possible for us to avoid field inversions (divisions) in our calculations (an example will follow). This is particulary useful since — at present — field inversions are considerably more expensive to compute than field multiplications [20, 30]. Special techniques are being developed for calculating inverses or "reciprocals" more efficiently (this is the subject we will present next), but for now, it would be advisable to avoid inversions as much as possible, making good use of the properties of projective coordinates [6, 20].

Example Suppose we have an elliptic curve E over a finite field K of characteristic $\neq 2, 3$. Therefore, this is an elliptic curve defined by equation 3.2. We shall consider addition and subtraction in the field K to be negligible computations since they take significantly less time than multiplication and division. For the sake of simplicity, multiplying a field element with a small constant (such as 2, 3, 4 or 8 in this example) will also be considered negligible [22].

Recall the rules of addition for (3.1). Given $P = (x_1, y_1)$, $Q = (x_2, y_2)$ where $P, Q \in E(K)$ and $P, Q \neq O$, the addition formula for computing $P + Q = (x_3, y_3)$ involves two field multiplications and one inversion when $P \neq \pm Q$, and three multiplications and one inversion when $P = Q$. To rewrite the addition formula in the projective plane, let $P = (X_1, Y_1, Z_1)$, $Q = (X_2, Y_2, Z_2)$ and $P + Q = (X_3, Y_3, Z_3)$. Then we will have:

If $P \neq \pm Q$

$$X_3 = v_7 v_{12}$$

$$Y_3 = v_6(v_{10} v_3 - v_{12}) - v_{11} v_1$$

$$Z_3 = v_{11} v_5$$

where the following values are computed *and saved* in this rough order:

Step 1 $v_1 = Y_1 Z_2,\ v_2 = Y_2 Z_1,\ v_3 = X_1 Z_2,\ v_4 = X_2 Z_1,\ v_5 = Z_1 Z_2$

Step 2 $v_6 = v_2 - v_1,\ v_7 = v_4 - v_3,\ v_8 = v_4 + v_3$

Step 3 $v_9 = {v_6}^2,\ v_{10} = {v_7}^2,\ v_{11} = {v_7}^3 = v_7 \cdot v_{10},\ v_{12} = v_9 \cdot v_5 - v_{10} \cdot v_8$

If $P = Q$

$$X_3 = 2v_{11}v_4$$

$$Y_3 = v_6(4v_7 - v_{11}) - 8v_2v_8$$

$$Z_3 = 8v_9$$

where the following values are computed *and saved* in this rough order:

Step 1 $v_1 = {X_1}^2,\ v_2 = {Y_1}^2,\ v_3 = {Z_1}^2,\ v_4 = Y_1Z_1,\ v_5 = X_1Y_1$

Step 2 $v_6 = av_3 + 3v_1,\ v_7 = v_4v_5,\ v_8 = {v_4}^2$

Step 3 $v_9 = v_4v_8,\ v_{10} = {v_6}^2$

Step 4 $v_{11} = v_{10} - 8v_7$

If we follow the above steps, the formula for $P \neq \pm Q$ will consist of 15 multiplications and **no** inversions, whereas the formula for $P = Q$ will require 12 multiplications and **no** inversions.

The resulting projective coordinate (X_3, Y_3, Z_3) can be converted back to affine coordinates by dividing each coordinate by Z_3 (or by multiplying the inverse of Z_3 to each coordinate). In effect, we have managed to avoid all but one inversion that is required at the end of all our computations on the projective plane.

Note that our count of multiplications in a formula depends on how the formula is written and which intermediate results we choose to store in memory. For instance, if we did not save the value of λ during our calculations in affine coordinates, we would have to perform three times as many inversions in a single addition operation. Clever substitutions and frugal storage of intermediate results have a substantial effect on computing speed. However, the need to store so much data is also its weakness: this technique offers its speed at the expense of storage space.

Faster Inversions For a long time, many have placed emphasis on the heavy computational costs of field inversions and have gone out of their way to avoid inversions by any means possible. But as we saw in the example above, bypassing an inversion leads to a dramatic increase in the number of multiplications. Clearly, there comes a point when the cost of all the extra multiplications surpasses the cost of computing a reciprocal. Recent improvements in the area of fast field divisions have highlighted this issue and have been slowly restoring the appeal of reciprocals. Schroeppel, Orman, O'Malley and Spatscheck[30] have proposed a "relatively fast algorithm for field inversion" that takes approximately three times as long as a multiplication. This is considerably faster than the performance of previous algorithms.

The new algorithm is aptly named **The Almost Inverse Algorithm**. Given an element α from the field F_q, it first computes β and k such that $\alpha\beta \equiv u^k \ mod\ q$ using a combination of known algorithms. Then it uses a smart strategy of bit operations to divide u^k out of β, thus finding the reciprocal of α. The proposed algorithm was written for the field $F_{2^{155}}$ (specifically, a polynomial extension field) and it would be interesting to see if and how it applies to other fields.

Montgomery's Method The x coordinate of a point on an elliptic curve is surprisingly malleable and informative. Two ideas have sprung from the interesting properties of the x coordinate:

1. rewriting part of the addition formula using only the x coordinates of points, and
2. reconstructing the value of the y coordinate using only x and a single bit from y.

The former is referred to as **Montgomery's Method**. The latter concept will be discussed next.

An idea by Montgomery was adapted to the addition formula of elliptic curves in [20]. Given an elliptic curve E, $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ where $P, Q \in E$ and $P \neq -Q$, and supposing that $P + Q = (x_3, y_3)$, then Montgomery's Method is to express x_3 using only x_1, x_2 and x_4 where $P - Q = (x_4, y_4)$. Note that $P - Q$ is the addition of P and $-Q$. Unfortunately, this technique does not apply to every elliptic curve, since it depends on the equation of the curve E and the definition of $-Q$ with respect to $Q \in E$. According to [20], it works well with "supersingular" curves over F_{2^m} (see equation 3.3) of the form $y^2 + y = x^3 + a_4 x + a_6$, resulting in the expression

$$x_3 = x_4 + \frac{1}{(x_1 + x_2)^2}$$

when $P \neq Q$. Not only is x_3 expressed using only the x coordinates of points, but it can also be calculated using only one inversion.

Reconstructing the y coordinate Recall that the Menezes-Vanstone Elliptic Curve Cryptosystem masked its plaintext and had a message expansion factor of 2. Since it is possible to recover the y coordinate of a point on an elliptic curve with just the x coordinate and a single bit from y (explained in [20]), we can reduce the message expansion factor of the Menezes-Vanstone scheme down to $\frac{3}{2}$. More specifically, we only need to publish and send the x coordinate of the public key $a_X P$ (using the notation from before). Therefore, if we use kP_x to denote the x coordinate of kP, then $y_0 = kP_x$ will suffice, where (y_0, y_1, y_2) is the ciphertext that Alice sends to Bob.

If Montgomery's Method applies, then it could be combined with this recovery technique to limit most (or all) calculations to the x coordinate alone. Focusing on the x coordinate of points will help reduce the complexity of computations and also save storage space. Demytko's new analogue of RSA [6] performs encryption and decryption on the x coordinate only, using projective coordinates and a new scheme to his advantage. Other schemes can benefit from the same approach [22].

Hardware Implementations Menezes and Vanstone [20] have noted that arithmetic in the finite field F_{2^r} is especially suitable for hardware implementation. An arithmetic processor efficiently designed to compute in F_{2^r} could readily apply to implementations of elliptic curve cryptosystems over the same field. Hence, it is worth examining some of the properties of the field F_{2^r}.

Looking at F_{2^r} as a vector space of dimension r over F_2 (recall the example from Chapter 2), the elements of F_{2^r} can be represented as binary vectors (or strings) of length r, given a suitable basis of this vector space. This makes it easy to store data in hardware (ideally in shift registers of length r). Addition in F_{2^r} can be performed in one clock cycle by bitwise XOR-ing the operands.

If we use a normal basis[7], then by definition it would have the form

$$\{\beta, \beta^2, \beta^{2^2}, \ldots, \beta^{2^{(r-1)}}\}$$

[7] Constructing a special class of normal basis called an optimal normal basis [26] could further minimize hardware complexity.

for some appropriate $\beta \in F_{2^r}$. Then any $\alpha \in F_{2^r}$ can be expressed as

$$\alpha = \sum_{i=0}^{r-1} a_i \beta^{2^i}$$

where $a_i \in F_2$. Conveniently,

$$\alpha^2 = \sum_{i=0}^{r-1} a_i \beta^{2^{i+1}} = \sum_{i=1}^{r} a_{i-1} \beta^{2^i}$$

Therefore, squaring an element in F_{2^r} is merely a matter of rotating its vector representation, which can be done in one clock cycle.

4.4.2 Summary of Attacks

Just like any other encryption system, elliptic curve cryptosystems are by no means immune to attack. However, the effective attack algorithms — all of which attempt to invert the EDLP in subexponential time — are few in number, and those that perform at practical, usable speeds are fewer still. From a cryptanalytic view, elliptic curve cryptosystems are generally very secure.

The MOV Reduction The most effective and important attack to date is the **MOV reduction** (also called the **MOV attack**), introduced by Menezes, Okamoto and Vanstone in 1991 [19]. Essentially, it is a method for reducing the elliptic curve logarithm problem in $E(F_q)$ to the discrete logarithm problem in F_{q^k} for some integer k — it exploits an isomorphism between the elliptic curve and finite field when $\gcd(\#E(F_q), q) = 1$. It is the first subexponential algorithm for solving the EDLP when k is small. Consequently, its effectiveness is limited to a special class of elliptic curves called **supersingular curves** (such as those defined by equation 3.3) since it has been shown that $k \leq 6$ for these curves. For most other curves (called **nonsupersingular curves**), k is too large for the MOV reduction to apply. (Both classes of curves will be examined in greater detail in the next section.)

Miyaji [23] observed that the reduction applies well to elliptic curves defined over F_{2^r}. But it was also proposed that elliptic curves defined over F_p (where p is a large prime) are immune to the attack. Furthermore, Miyaji proposed a construction for such an elliptic curve that would make the reduction of the EDLP to the DLP impossible. Therefore, not all elliptic curve cryptosystems are susceptible to the MOV attack.

Other Attacks Before the MOV reduction was proposed in 1991, the best attacks were Shanks' "baby-step giant-step" method, which works in exponential time (in $\log \#E$), and a modified version of the Pohlig-Hellman attack, whose running time is proportional to the square root of the largest prime factor of $\#E$ [21]. They are algorithms for solving the DLP in the prime field Z_p that can be extended to the EDLP. A combination of both will also serve as a good "general-purpose" algorithm for the EDLP [20]. Another known attack on the EDLP is the Pollard ρ-method [22].

It is possible, however, to thwart the Pohlig-Hellman attack. To avoid an easy solution to the EDLP, we want an elliptic curve E over F_q that contains a cyclic subgroup H in which the EDLP is intractible, i.e. we want the order of the subgroup (or $\#E$) to be divisible by at least one large prime factor (of more than 30 digits [22]). This technique applies to any finite abelian group.

Various other attacks have proven to be ineffective against elliptic curve cryptosystems. Most notably, there are no known adaptations of the Index Calculus attack (which is a powerful algorithm for solving the DLP) to the EDLP. The analogue of the Diffie-Hellman key exchange protocol is apparently immune to the attack methods of Western, Miller, and also Adleman's subexponential-time attacks [21]. Demytko's analogue of RSA is safe from homomorphism attacks [6]. The schemes proposed in [14] are believed to be immune to homomorphism attacks, isomorphism attacks and low multiplier attacks.

4.4.3 Choosing an Elliptic Curve

After reviewing the attacks we have mentioned, it should be apparent that the choice of the elliptic curve E and its underlying field K has enormous impact on the speed, efficiency, key length (i.e. practicality) and security of any elliptic curve cryptosystem. Although E, K and a base point $P \in E$ are all fixed and publicly known *prior* to the encryption process, the task of selecting them for a given scheme is the most important step. We will explore some of the choices here.

The Field K

Let us review the influence that K has on the group structure of $E(K)$ and on any cryptosystem over $E(K)$.

In the first place, an elliptic curve E over a finite field forms an abelian group, which makes it useable in cryptosystems. We have seen that certain fields such as F_{2^r} are amenable to hardware implementations and fast field operations. In fact, computations such as doubling a point (i.e. computing $P + P$, $P \in E$) using field arithmetic in F_{2^r} can be "free" (of negligible cost) if the field elements are represented by a normal basis. For example, the formula for doubling a point $P = (x_1, y_1)$ in an elliptic curve defined by $y^2 + y = x^3$ can be simplified to

$$\begin{aligned} x_3 &= {x_1}^4 \\ y_3 &= {y_1}^4 + 1 \end{aligned}$$

(because $a_3 = 1$, $a_4 = a_6 = 0$, and F_{2^r} has characteristic 2). Since the addition of field elements and squaring a field element each take only one clock cycle, they are considered to be "free" computations. Therefore, $(x_3, y_3) = P + P$ can be computed in 5 clock cycles in this case, which is a negligible amount of time. [20]

Elliptic curves over F_{2^r} are vulnerable to the MOV reduction which can solve the EDLP in subexponential time, whereas curves over F_p (p is a large prime) are safe against such attacks. Clearly, elliptic curves on the prime field F_p [23] and curves on the finite field F_{q^n} [20, 30] have well-established properties that make them attractive for practical implementations.

In addition, recall that it is advantageous to know the value $\#E(K)$. For example, E with an appropriate value $\#E$ would be immune from the Pohlig-Hellman attack. It can be computed using Schoof's deterministic polynomial time algorithm which was proposed for elliptic curves over a finite field F_q with characteristic $\neq 2, 3$. The speed of Schoof's algorithm depends on the size and characteristic of K. For example, when r is small, $\#E(F_{2^r})$ can be computed slightly faster than $\#E(F_p)$ for a prime p whose size is comparable to 2^r, but as r increases, the former takes much more time to compute than the latter [16]. Future improvements in this area may change this result.

Types of Elliptic Curves

To choose the "right" elliptic curve, we first need to know what kind of curve we want and what types we can use. There are infinite varieties of elliptic curves to choose from but a select few have been of interest to the study of elliptic curve cryptosystems. In the previous section, we looked at the fields K that have demonstrated qualities amenable to fast computation and security. We shall present two classes of elliptic curves that have been used in various encryption schemes.

Supersingular Curves Menezes and Vanstone [20] have examined the advantages of **supersingular** elliptic curves in cryptosystems, specifically those over the field F_{2^r}. An elliptic curve over a finite field of q elements is said to be supersingular if $t^2 = 0, q, 2q, 3q$ or $4q$ where t is defined in Hasse's theorem as $t = q + 1 - \#E(F_q)$, $|t| \leq 2\sqrt{q}$. An elliptic curve over a field of characteristic 2 or 3 is supersingular if and only if it has a zero j-invariant. For example, an elliptic curve defined by equation 3.3 is a supersingular curve.

As stated before, the arithmetic operations for supersingular curves over F_{2^r} can be implemented in hardware and the elements of F_{2^r} can be efficiently represented by a normal basis. Also, given a supersingular curve over F_{2^r}, if we choose $a_3 = 1$ (see equation 3.3) then inversions can be eliminated when doubling points (adding a point to itself) [20].

Unfortunately, certain supersingular curves are vulnerable to the MOV attack (namely, the curves over F_{2^r}). For supersingular curves, it has been shown that $k \leq 6$ [19]. A supersingular curve could be protected from this attack if a finite field F_q of sufficiently large size is chosen, so that the DLP in F_{q^k} would be intractible even when using the best known algorithms for this problem.

Nonsupersingular Curves A **nonsupersingular curve** or an "ordinary" elliptic curve has a nonzero j-invariant. Equation 3.4 describes such a curve. The computation techniques that apply to supersingular curves — projective coordinates, optimal normal basis representation, hardware implementation, etc. — can easily be extended to the case of nonsupersingular curves. The advantage that a nonsupersingular curve has over a supersingular curve is that it can provide the same level of security as the supersingular curve, but with a much smaller underlying field

[20]. This shortens the key length, making it attractive for use in smart cards.

Much emphasis has been placed on supersingular curves, but they are vulnerable to the MOV attack, and as it turns out, they make up only a small minority of the domain of elliptic curves [5]. Nonsupersingular curves are a practical alternative.

Nonsupersingular curves appear to be immune to the MOV attack (for example, those with a cyclic subgroup of size 2^{160}). Therefore, the best known attack on these curves is Shanks' exponential algorithm. The order of the subgroup should be divisible by at least one large prime factor to guard it from a Pohlig-Hellman attack.

Selection Methods

There are several approaches to making the "right" choices. To date, curves have often been selected randomly, though this method is losing some of its appeal due to the lack of control exercised over the value of $\#E(K)$ in the selection process. This technique is being replaced by the relatively recent idea of *constructing* the desired elliptic curve with specific attributes in mind (i.e. attributes that preclude known attacks). Yet another alternative would be to create a cryptographic scheme whose security is not dependent on the EDLP (like the elliptic curve based analogues of RSA), thereby making the appropriate selection of elliptic curves a non-issue.

Notice that elliptic curve cryptosystems actually work in the cyclic subgroup of a curve E generated by the base point P, rather than the entire group E. Therefore, it is also important to select an appropriate P.

Randomly Choosing Elliptic Curves Randomly picking an elliptic curve E over the field K and a base point $P \in E$ is essentially a process of trial and error. K has been chosen and fixed in advance. Koblitz's random selection method [12, page 166] for curves over F_q (for large q) is described in Figure 4.5 (suppose we are dealing with F_q of characteristic $\neq 2, 3$).

---

1. Randomly select three elements from F_q; call them x, y, a
2. Set the value for b by computing $b = y^2 - (x^3 + ax)$ since equation 3.2 is $y^2 = x^3 + ax + b$
3. Check that the cubic on the right side of 3.2 does not have multiple roots, i.e. check that $4a^3 + 27b^2 \neq 0$
4. **if** the previous condition is not met, return to step 1.
5. **else** set $P = (x, y)$ and let $y^2 = x^3 + ax + b$ be our elliptic curve

---

Figure 4.5: Koblitz's Random Selection Method

Other random selection methods are similar, except for the condition in step 3. which could be any desired condition(s) to be met by the elliptic curve E.

The problem with this approach is that we waste time by repeating steps 1.–3. until we finally obtain an acceptable result. Note that the probability that a random $x \in F_q$ is in fact the x coordinate of a point in E is approximately $\frac{1}{2}$ (by Hasse's Theorem). This method offers us very little *direct* control over the structure of the elliptic curve and the base point — their properties are more or less left up to chance — and therefore it denies us control over the security of the cryptosystem.

Constructing an Elliptic Curve A more complex approach is to **construct** the elliptic curve we want. Ideally, it would be desirable for our design strategy to exercise total control over the group structure of the the elliptic curve we choose. In other words, we would first like to specify the properties we want in an elliptic curve, then set out to construct one that meets all our conditions.

However, in practice, the best known strategy is to place more demanding conditions in step 3. or elsewhere in the random selection method. The more demanding the conditions become, the less unpredictable the resulting selections will be.

Example For security, we want the cyclic subgroup generated by the base point P to be a group in which the EDLP is intractible. To satisfy this condition, we could verify in step 3. that the order of $P = (x, y)$ is divisible by a large prime (as close to $\#E$ as possible).

To date, Miyaji has suggested some constructions for elliptic curves over F_p (where p is a large prime) in [22, 23]. Chao, Tanada and Tsujii [5] very recently modified Atkin and Morain's algorithm [1, 25] for building curves with complex multiplication that satisfy specifications on $\#E$.

Unfortunately, the control that we want over our choice of elliptic curves comes at the expense of speed. (For example, the construction algorithm in [5] takes exponential time.) Not surprisingly, the computation of $\#E$ is required in all constructions interested in the security of the elliptic curve, and therefore, Schoof's cumbersome algorithm (the best to date for computing $\#E$) often accounts for the compromise of speed.

We implemented (a slightly modified version of) Koblitz's construction algorithm [5], which is described in Figure 4.6. As indicated, Schoof's algorithm was involved, and the size of the resulting program (nearly 700 lines of code) made the algorithm's complexity plainly obvious.

1. Randomly choose a (large) prime q
2. Use Koblitz's random selection method to find an elliptic curve $E(F_q)$ of the type defined by equation 3.2
3. Use Schoof's algorithm [29] to compute $\#E(F_q)$
4. Verify that $\#E(F_q)$ is a (large) prime.
5. **if** the previous condition is not met, return to step 2.

Figure 4.6: Koblitz's Construction Algorithm

If we perform Koblitz's algorithm, then any point in E other than O would be a generator of E (since any group of prime order is cyclic), and the EDLP over E would be intractible. Once the desired elliptic curve is found, it can be used in the cryptosystems described earlier in this chapter.

Schoof's algorithm essentially consists of four steps, as described in Figure 4.7.

Step 2. is the most computationally taxing step, as can be seen in the processes described in the Appendix. It involves numerous evaluations of complicated poly-

1. Let $l_1 = 3, l_2 = 5, l_3 = 7, \ldots, l_k$ be the k consecutive primes starting at 3, where k is the largest integer such that
$$\prod_{i=1}^{k} l_i \leq 4\sqrt{q}$$
and set $L = l_k$. (*Note:* Schoof's paper [29] asks for $\prod_{i=1}^{k} l_i > 4\sqrt{q}$ to be satisfied, which appears to be a mistake.)

2. Compute τ_i $(mod\ l_i)$ for all i $(1 \leq i \leq k)$ via the steps described in the Appendix.

3. Use the Chinese Remainder Theorem to compute
$$t = \sum_{i=1}^{k} \tau_i M_i y_i \ mod\ M$$
where $M = \prod_{i=1}^{k} l_i$, $M_i = \frac{M}{l_i}$ and $M_i y_i \equiv 1\ mod\ l_i$. Find a t that satisfies $|t| \leq 2\sqrt{q}$ (Hasse's Theorem), i.e. if $t > 2\sqrt{q}$ set $t = t - M$

4. Compute $\#E(F_q) = q + 1 - t$

Figure 4.7: Schoof's Algorithm

nomials such as $\Psi_n(x, y)$ and $f_n(x)$, and a maze of tests that eventually yield the final result.

Various other functions clutter the program. For example, the square-and-multiply algorithm [33, page 127] and the Extended Euclidean Algorithm were borrowed from the program described in section 4.3. Prime generation is performed via trial division [8, pp. 37–40] and primality testing is performed by the Miller-Rabin primality test [33, page 137] (applied five times to reduce the probability that a composite number will pass the test [28, page 260]). Euler's Criterion

[33, page 131] is used to determine whether a number is a quadratic residue or not, and the square root modulo p (where p is an odd prime) is computed by an algorithm presented in [12, pp. 47–48]. For brevity, we will not examine these algorithms in further detail.

Unfortunately, there is no definitive answer yet that determines the probability that $\#E$ will be prime for a random E. Certainly, the extra criterion on $\#E$'s properties forces the program to test and discard many elliptic curves. But there is no way of predicting how the program will perform, as can be seen in Table 4.2. Note that **# Tries** refers to the number of curves that were rejected by the program before the first "acceptable" curve was found, and **Time** indicates the number of seconds this process took[8]. **# Tries** also reflects how frequently the program fails to produce desirable output at step 4. of Koblitz's algorithm.

Another difficulty with the implementation is that there is no easy way of testing the validity of the program's output for large q. For small q, verification is a simple, straightforward matter of generating all the points on $E(F_q)$, but this method becomes less and less practical as q becomes large.

It should also be noted that much of the program depends on the randomness of the random numbers it generates. Since the best a computer can do is generate a pseudo-random sequence of numbers, there is a threat to the security of a cryptosystem if the number generation turns out to be predictable (which it is, in the case of the **rand()** function in Turbo C++ ©1990, 1992, version 3.0, with which this program was tested).

[8]These results were obtained on a Dell Pentium XPS P90.

Elliptic Curves Over a Ring Z_n Finally, we would like to take this opportunity to mention a concept that doesn't quite fit in anywhere else in the thesis: cryptographic schemes based on elliptic curves over a **ring** Z_n where n is a product of two large primes. Most elliptic curve cryptosystems are designed around the EDLP, relying on the intractibility of the problem for its security. However, a public-key cryptographic scheme that uses curves over a ring Z_n rely on the difficulty of factoring n — a familiar, "traditional" approach to security in cryptography, used in RSA, for example. This frees us from the grand task of selecting a curve from a vast number of choices and the restrictions that other cryptosystems place on us whenever we choose the "right" (or "wrong") elliptic curve for the scheme.

Koyama, Maurer, Okamoto and Vanstone were the first to propose TOFs based on elliptic curves over the ring Z_n [14]. A couple of years later, Demytko modified these early concepts so that the selection of elliptic curves could be more flexible: "the scheme [...] can be used on elliptic curves with arbitrary parameters." [6]

q	# Tries	$E(F_q)$	$\#E(F_q)$	Time (sec)
11	2667	$y^2 = x^3 + 8x + 1$	17	0.164835
13	11	$y^2 = x^3 + 2x + 9$	17	0.000000
17	60	$y^2 = x^3 + 9x + 5$	11	0.054945
19	2	$y^2 = x^3 + 5x + 12$	19	0.054945
23	18	$y^2 = x^3 + 2x + 6$	29	0.000000
29	31	$y^2 = x^3 + 22x + 16$	37	0.054945
31	71	$y^2 = x^3 + 5x + 3$	41	0.054945
37	5	$y^2 = x^3 + 8x + 14$	47	0.000000
41	1153	$y^2 = x^3 + 8x + 4$	43	0.274725
43	2	$y^2 = x^3 + 27x + 22$	29	0.000000
47	43	$y^2 = x^3 + 38x + 6$	37	0.054945
53	113	$y^2 = x^3 + 5x + 12$	43	0.054945
59	17	$y^2 = x^3 + 4x + 49$	53	0.000000
61	34	$y^2 = x^3 + 31x + 49$	61	0.054945
67	12	$y^2 = x^3 + 2x + 56$	37	0.000000
71	9	$y^2 = x^3 + 57x + 14$	47	0.054945
73	71	$y^2 = x^3 + 33x + 34$	79	0.000000
79	3	$y^2 = x^3 + 75x + 6$	61	0.000000
83	8	$y^2 = x^3 + 3x + 78$	67	0.000000
89	149	$y^2 = x^3 + 54x + 52$	103	0.054945
97	97	$y^2 = x^3 + 32x + 33$	97	0.054945

Table 4.2: Program Performance

Chapter 5

Conclusion

So far, practical applications of elliptic curve cryptosystems have primarily involved hardware implementations in arithmetic processors. In conjunction with Cryptech Systems Inc. (Canada), Newbridge Microsystems Inc. manufactured a single chip device that computes arithmetic in the field $F_{2^{593}}$ for implementing various cryptosystems. A custom gate array device was constructed for field arithmetic in $F_{2^{155}}$, specifically designed for efficient elliptic curve point additions [20].

In light of these results, the idea of implementing digital signature/identification schemes in the form of smart cards has quickly gained momentum. Since the convenience of smart cards depends on their portable size, the arithmetic processors they employ should be restricted to an area of approximately $20\ \mathrm{mm}^2$. Current technology can't produce chips that meet this criterion.[20] However, elliptic curve cryptosystems can provide security with short key lengths, requiring less data for storage on a smart card and less computation.[22] According to Menezes and Vanstone, a chip designed to perform arithmetic in F_{2^m} where $m \approx 200$ could occupy just 15% of that allotted area. In maintaining a secure channel of communication, the hardware described above could be shared by all users, regardless of what elliptic curve they choose, as long as everyone uses curves over the same field K. [20]

Next Computer Inc. recently patented the Fast Elliptic Encryption (FEE) algorithm[1] which uses elliptic curves and pragmatically features private keys that are allowed to be strings. This makes a key easy to remember and use like an ordinary password [28, page 481]. However, this is a dubious advantage since keys that are easy to remember have a limited keyspace.

The infinitude of elliptic curves — with familiar cryptographic properties, but conveniently without properties that commonly facilitate cryptanalysis — suggests the need to continue these studies with different elliptic curves and different cryptosystems. Previously neglected elliptic curves might be applied to the cryptosystems studied so far, since we have seen that the choice of curves can seriously affect the security and efficiency of an elliptic curve cryptosystem. The search for suitable elliptic curves will be ongoing. Or, we could examine other existing cryptosystems to which elliptic curves have yet to be applied, since the advantages of elliptic curves vary from cryptosystem to cryptosystem. Some have recently proposed public-key cryptosystems using hyperelliptic curves [27]. The manner in which elliptic curves are chosen could also be changed by welcome improvements in Schoof's indispensable algorithm for calculating the cardinality of an elliptic curve.[16]

These ideas for improving the computational speed, efficiency and security of

[1] R.E. Crandell, "Method and Apparatus for Public-Key Exchange in a Cryptographic System," U.S. Patent #5,159,632, 27 Oct 1992.

elliptic curve cryptosystems are useful for improving practical implementations. However, the exact nature of the relationship between the EDLP and the DLP remains unclear. It is a critical open problem whose solution would determine the security (or lack thereof) of elliptic curve cryptosystems, especially since the MOV reduction seems to apply only to specific types of curves. Are there any more practical methods for solving the EDLP expediently? Are there any more TOFs that cannot be inverted in (sub)exponential time?

Furthermore, new results in the area of quantum computing may eventually make crytosystems based on the EDLP obsolete. Quantum computers are machines based on principles of quantum mechanics (for more information, see [3]). Shor [31] presented an algorithm that would theoretically allow a quantum computer to solve the DLP in polynomial time, and recently, Boneh and Lipton [2] showed that a quantum computer would be able to solve the EDLP in polynomial time as well.

Appendix A

Schoof's Algorithm

This section describes step 2. of Schoof's Algorithm (see Figure 4.7).

First, we define the polynomials $\Psi_n(x,y) \in F_q[x,y]$ and $f_n(x) \in F_q[x]$ for $n \in Z_{\geq -1}$.

$$\Psi_{-1}(x,y) = -1, \;\; \Psi_0(x,y) = 0, \;\; \Psi_1(x,y) = 1, \;\; \Psi_2(x,y) = 2y,$$

$$\Psi_3(x,y) = 3x^4 + 6ax^2 + 12bx - a^2,$$

$$\Psi_4(x,y) = 4y(x^6 + 5ax^4 + 20bx^3 - 5a^2x^2 - 4abx - 8b^2 - a^3),$$

$$\Psi_{2m}(x,y) = \Psi_m(\Psi_{m+2}\Psi_{m-1}^2 - \Psi_{m-2}\Psi_{m+1}^2)/2y \quad (m \in Z_{\geq 1}),$$

$$\Psi_{2m+1}(x,y) = \Psi_{m+2}\Psi_m^3 - \Psi_{m+1}^3\Psi_{m-1} \quad (m \in Z_{\geq 1})$$

If we replace all y^2-terms in Ψ_n with $x^3 + ax + b$ (see equation 3.2), we call the resulting polynomial $\Psi_{n'}(x,y)$. So we define

$$f_n(x) = \begin{cases} \Psi_{n'}(x,y)/y & \text{if } n \text{ is even and } n > 0 \\ \Psi_{n'}(x,y) & \text{otherwise} \end{cases}$$

For simplicity, we will use l and τ to denote l_i and τ_i, respectively. For a given l, perform the following:

1. Compute
$$\begin{cases} \gcd((x^{q^2} - x)f_k^2(x)(x^3 + ax + b) + f_{k-1}(x)f_{k+1}(x),\ f_l(x)) & \text{if } k \text{ is even} \\ \gcd((x^{q^2} - x)f_k^2(x) + f_{k-1}(x)f_{k+1}(x)(x^3 + ax + b),\ f_l(x)) & \text{if } k \text{ is odd} \end{cases}$$
where $k \equiv q \ (mod\ l)$ and $1 \leq k < l$

2. **if** the value computed in step 1. is $\neq 1$ then goto step 3.

   else goto step 8.

3. **if** q is not a quadratic residue modulo l then set $\tau \equiv 0 \ (mod\ l)$ **[END]**

   else goto step 4.

4. Compute
$$\begin{cases} \gcd((x^q - x)f_w^2(x)(x^3+ax+b) + f_{w-1}(x)f_{w+1}(x),\ f_l(x)) & \text{if } w \text{ is even} \\ \gcd((x^q - x)f_w^2(x) + f_{w-1}(x)f_{w+1}(x)(x^3+ax+b),\ f_l(x)) & \text{if } w \text{ is odd} \end{cases}$$
   where $w^2 \equiv q \ (mod\ l)$

5. **if** the value computed in step 4. is $= 1$ then set $\tau \equiv 0 \ (mod\ l)$ **[END]**

   else goto step 6.

6. Compute
$$\begin{cases} \gcd(4(x^3+ax+b)^{(q-1)/2} f_w^3(x) - f_{w+2}^2(x)f_{w-1}(x) & \\ \qquad\qquad + f_{w-2}^2(x)f_{w+1}(x),\ f_l(x)) & \text{if } w \text{ is even} \\ \gcd(4(x^3+ax+b)^{(q+3)/2} f_w^3(x) - f_{w+2}^2(x)f_{w-1}(x) & \\ \qquad\qquad + f_{w-2}^2(x)f_{w+1}(x),\ f_l(x)) & \text{if } w \text{ is odd} \end{cases}$$

7. **if** the value computed in step 6. is $= 1$ then set $\tau \equiv -2w \ (mod\ l)$ **[END]**

   else set $\tau \equiv 2w \ (mod\ l)$ **[END]**

8. Find a τ $(0 < \tau < l)$ that satisfies the following two conditions:
$$\begin{aligned} &((\Psi_{k-1}\Psi_{k+1} - \Psi_k(x^{q^2} + x^q + x))\beta^2 + \Psi_k^2\alpha^2)\Psi_\tau^{2q} \\ &\qquad\qquad + \Psi_{\tau-1}^q\Psi_{\tau+1}^q\beta^2{\Psi_k}^2 \quad \equiv \quad 0\ (mod\ f_l(x)) \end{aligned}$$
$$\begin{aligned} &4y^q\Psi_\tau^{3q}(\alpha((2x^{q^2}+x)\Psi_k^2 - \Psi_{k-1}\Psi_{k+1}) - y^{q^2}\beta\Psi_k^2) \\ &\qquad\qquad - \beta\Psi_k^2(\Psi_{\tau+2}\Psi_{\tau-1}^2 - \Psi_{\tau-2}\Psi_{\tau+1}^2)^q \quad \equiv \quad 0\ (mod\ f_l(x)) \end{aligned}$$
   where $\alpha = \Psi_{k+2}\Psi_{k-1}^2 - \Psi_{k-1}\Psi_{k+1}^2 - 4y^{q^2+1}\Psi_k^3$

   and $\beta = ((x - x^{q^2})\Psi_k^2 - \Psi_{k-1}\Psi_{k+1})4y\Psi_k$ **[END]**

Bibliography

[1] A. Atkin and F. Morain. Elliptic curves and primality proving. *Mathematics of Computation*, Vol. 61, No. 203, pp. 29–68, July 1993.

[2] D. Boneh and R. Lipton. Quantum Cryptoanalysis of Hidden Linear Functions. *Advances in Cryptology - CRYPTO '95*, pp. 424–437, 1995.

[3] G. Brassard. A quantum jump in computer science. *Computer Science Today, Lecture Notes in Computer Science*, Vol. 1000, pp. 1–14, 1995.

[4] J. W. S. Cassels. *Lectures on Elliptic Curves*. Cambridge University Press, 1991.

[5] J. Chao, K. Tanada, and S. Tsujii. Design of Elliptic Curves with Controllable Lower Boundary of Extension Degree for Reduction Attacks. *Advances in Cryptology - CRYPTO '94*, Vol. 839, pp. 50–55, 1994.

[6] N. Demytko. A New Elliptic Curve Based Analogue of RSA. *Advances in Cryptology - EUROCRYPT '93*, pp. 40–49, 1994.

[7] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, Vol. 22, No. 6, pp. 644–654, 1976.

[8] P. Giblin. *Primes and Programming: An Introduction to Number Theory with Computing*. Cambridge University Press, 1993.

[9] H. Hasse. Beweis des Analogons der Riemannschen Vermutung für die Artinschen u. F. K. Schmidtschen Kongruenzzetafunktionen in gewissen elliptischen Fällen, Vorläufige Mitteilung. *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen I*, **42**: 253–262, 1933.

[10] H. Hasse. Zur Theorie der abstrakten elliptischen Funktionenkörper. *Journal für reine u. angewandte Math, I, II, III*, **175**: 55–62, 69–88, 193–208, 1936.

[11] B. S. Kaliski Jr. One-Way Permutations on Elliptic Curves. *Journal of Cryptology*, pp. 187–199, 1991.

[12] N. Koblitz. *A Course in Number Theory and Cryptography*, Springer-Verlag New York Inc., 1987.

[13] N. Koblitz. Elliptic Curve Cryptosystems. *Mathematics of Computation*, Vol. 48, No. 177, pp. 203–209, January 1987.

[14] K. Koyama, U. M. Maurer, T. Okamoto, and S. A. Vanstone. New Public-Key Schemes Based on Elliptic Curves over the Ring Z_n. *Advances in Cryptology - CRYPTO '91*, pp. 252–265, 1991.

[15] J. Landin. *An Introduction to Algebraic Structures*. Dover Publications, Inc., 1989.

[16] R. Lercier and F. Morain. Counting the number of points on elliptic curves over finite fields: strategies and performances. *Advances in Cryptology - EUROCRYPT '95*, pp. 79–94, 1995.

[17] R. Lidl and H. Niederreiter. *Introduction to finite fields and their applications*. Cambridge University Press, 1986.

[18] S. Lipschutz. *Linear Algebra*. 2nd ed., McGraw-Hill, Inc., 1991.

[19] A. Menezes, T. Okamoto, and S. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *Proceedings of the 23rd Annual ACM Symposium on the Theory of Computing*, pp. 80–89, 1991.

[20] A. Menezes and S. Vanstone. Elliptic Curve Cryptosystems and Their Implementation. *Journal of Cryptology*, pp. 209–224, 1993.

[21] V. S. Miller. Use of Elliptic Curves in Cryptography. *Advances in Cryptology - CRYPTO '85*, pp. 417–426, 1986.

[22] A. Miyaji. Elliptic Curves over F_p Suitable for Cryptosystems. *Advances in Cryptology - AUSCRYPT '92*, pp. 479–491, 1993.

[23] A. Miyaji. On Ordinary Elliptic Curve Cryptosystems. *Advances in Cryptology - ASIACRYPT '91*, pp. 460–469, 1991.

[24] P. Montgomery. Speeding the Pollard and elliptic curve methods of factorization. *Mathematics of Computation*, Vol. 48, No. 177, pp. 243–264, January 1987.

[25] F. Morain. Building cyclic elliptic curves modulo large primes. *Advances in Cryptology - EUROCRYPT '91, Lecture Notes in Computer Science*, **547**: 328–336, 1991.

[26] R. Mullin, I. Onyszchuk, S. Vanstone, and R. Wilson. Optimal normal bases in $GF(p^n)$. *Discrete Applied Mathematics*, **22**: 149–161, 1988/89.

[27] T. Okamoto and K. Sakurai. Efficient Algorithms for the Construction of Hyperelliptic Cryptosystems. *Advances in Cryptology - CRYPTO '91 Proceedings*, pp. 267–278, 1992.

[28] B. Schneier. *Applied Cryptography*. 2nd ed., John Wiley & Sons, Inc., 1996.

[29] R. Schoof. Elliptic Curves Over Finite Fields and the Computation of Square Roots mod *p*. *Mathematics of Computation*, Vol. 44, No. 170, pp. 483–494, April 1985.

[30] R. Schroeppel, H. Orman, S. O'Malley, and O. Spatscheck. Fast Key Exchange with Elliptic Curve Systems. *Advances in Cryptology - CRYPTO '95*, pp. 43–56, 1995.

[31] P. W. Shor. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. *Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science*, pp. 124–134, 1994.

[32] J. H. Silverman. *The Arithmetic of Elliptic Curves*. Springer-Verlag New York Inc., 1986.

[33] D. R. Stinson. *Cryptography: theory and practice*. CRC Press, Inc., 1995.