The Application of the Mordell-Weil Group to Cryptographic Systems

by

André Weimerskirch

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Computer Science

by

April 2001

APPROVED:

Prof. Christof Paar, Thesis Advisor ECE and CS Department	Prof. William J. Martin, Thesis Reader Mathematical Sciences Department
Prof. Gabor Sarkozy, Thesis Reader CS Department	Prof. Micha Hofri, Department Head CS Department

Abstract

This thesis examines the Weil-Mordell group for application in cryptography. This approach has recently been proposed by Gerhard Frey. The use of the Mordell-Weil group for discrete logarithm schemes is a variant of elliptic curve cryptosystems. We extended the original idea by Frey with the goal of a performance improvement. The arithmetic complexity using the Mordell-Weil group will be compared to ordinary elliptic curve cryptosystems. The main goals of this thesis are (1) to investigate the algorithmic complexity of Weil-Mordell cryptosystems relative to elliptic curve cryptosystems; (2) the appropriate selection of the group parameters for a successful adaptation to different platforms; (3) a C++ library which makes it possible to easily use this algebra for cryptographic systems based on groups; and (4) to obtain software performance measures for the new cryptosystem. Point multiplication, the crucial operation for elliptic curve cryptosystems, is more than 20% less complex in the Mordell-Weil group than in an ordinary elliptic curve while preserving the same level of security. We show how to further improve the system such that it is particularly suited to 32-bit and 16-bit hardware platforms. The speed-up of the Mordell-Weil group approach comes at the cost of a slightly larger bit-size that is needed to represent a curve point and a more costly curve generation.

Preface

This thesis describes research that I conducted while completing my Master's degree at Worcester Polytechnic Institute. I hope that this work is of use and will be carried on.

First of all I would like to thank Prof. Paar for all the help, advice, and support that he gave me. Since I came to WPI last year he supported me all the time through expertise, friendship, and financial aid. I want to thank him for giving me the possibility to do research in an area I love in an atmosphere of friendship and camaraderie. He has an essential impact for my life career.

I took my first cryptography course two years ago with Prof. Buchmann at the Darmstadt University of Technology, Germany. I was fascinated by this topic since the first class and want to thank Prof. Buchmann for introducing me into this area with such an excellence. I also want to thank Prof. Frey from Essen University, Germany, for making this cutting edge research possible for me, and I am grateful to him and his students, especially Guido Blady, for supporting me and answering so many of my questions.

I would like to give special thanks to all my colleagues and friends. Jorge Guajardo who spent so much time explaining me the secrets of cryptography and who was always being there and provided me with support. Also many thanks to Thomas Wollinger for being such a good friend, and to Dan Bailey, Adam Elbirt and Adam Woodbury for providing a special working atmosphere in our Crypto Lab.

I want to thank my thesis readers Prof. Martin and Prof. Sarkozy from Worcester Polytechnic Institute for taking the time to review this work and giving me many valuable suggestions and comments, and having plenty of fruitful conversations. Furthermore I am grateful to Prof. Selkow for helping me with and explaining me all the paperwork that I had to manage. Also I want to thank Dr. Stärk and the

German Academic Exchange Service (DAAD) for the scholarship they gave me and making my studies at WPI possible.

Finally I want to give special thanks to my family for always supporting and encouraging me and making my life so much easier.

Thank you all!

Andre

Contents

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Motivation

Since elliptic curves were introduced by Koblitz [9] and Miller [18] the use of elliptic curves in public-key cryptography has become more and more important in practice. Public-key systems are based on one-way functions which are by definition relatively easy to compute, but computationally infeasible to reverse. The most popular systems use one-way functions based on integer factorization and the discrete logarithm problem (DLP). The integer factorization problem is based on the difficulty to factor a large integer number into its prime factors. An example for a public-key system based on this method is the popular RSA system. The discrete logarithm problem will be described later in this thesis. Systems based on the DLP require an abelian group. The points on an elliptic curve generate such an abelian group, which seems to be of advantage for crytographic purposes. The same level of security can be obtained by using elliptic curves with a smaller key length compared to RSA and DLP systems over finite fields, which results in smaller memory and processor requirements. For instance, a key length of 173 bits for an

elliptic curve cryptosystem is believed to be security equivalent to an RSA system with 1024 bits [2]. This is advantageous for applications with restricted resources like smart cards and embedded systems such as personal digital assistants. Frey and Naumann [21] suggested recently a new group derived from elliptic curves, the Mordell-Weil group. While point multiplication methods in elliptic curves consist of doubling and addition steps, the properties of the Mordell-Weil group allow to use the Frobenius map to decrease the number of doublings. Hence, if this group is used for public-key systems based on the DLP there is a potential for an increased performance compared to elliptic curve cryptosystems. The Mordell-Weil group will be the main topic of this thesis.

1.2 Thesis Outline

This thesis is organized as follows. Chapter 2 gives an overview of the existing research. Chapter 3 introduces the theory behind this thesis, a new group proposed by Frey and Naumann [21] which is derived from elliptic curves and can improve the benefits of elliptic curves. Chapter 4 describes our extension of their work. Chapter 5 shows the arithmetic of general elliptic curves and proposes efficient arithmetic for the Mordell-Weil group and discusses implementation issues. Chapter 6 describes the algorithmic complexity of different point multiplication methods for ordinary elliptic curves and the Mordell-Weil group and compares them. In Chapter 7 an example is given.

Chapter 2

Previous Work

2.1 Cryptography Based on Finite Groups

This section gives a quick survey of public-key systems based on the discrete logarithm problem (DLP). A more detailed description can be found in [17] and [22]. We assume that G is a finite abelian group with $\#G$ elements which is cyclic, i.e., generated by an element g such that $G = \langle g \rangle$. To derive a cyclic group G from an arbitrary one O we select an element $g \in O$ of large order, where the order of an element g is the smallest positive number e such that $g^e = 1$. We use the subgroup which is generated by g, $G = \langle g \rangle$, as the cyclic group for the cryptographic system. The number of elements in the group G is called the order of G and it is equal to e. An important example is points on an elliptic curve. For our purpose it is important that the group operation can be efficiently implemented, while computing discrete logarithms in this group is computationally impossible with current technology and algorithms. The DLP is as follows: Given g and $b \in G$ find the smallest positive integer x such that $g^x = b$. The element x exists since G is cyclic and g is a generator. Someone who can solve the DLP can also break the system, therefore this

problem must be computationally infeasible. As an example of how to use the DLP for a public-key system, we will give a short description of the Diffie-Hellman key exchange protocol. There are also digital signature methods based on the DLP, e.g., the Digital Signature Algorithm (DSA).

Assume Alice and Bob want to share a secret integer, but their only way to communicate is over an insecure connection. This integer can be used as a secret key for a conventional cryptosystem in subsequent communication sessions. Alice and Bob (and any intruder) know the group G and an element $g \in G$ of large known order. Now Alice and Bob do the following:

Algorithm 1 Diffie-Hellman key exchange

1: Alice generates a random integer $x_A \in \{1, \ldots, \#G - 1\}$ and sends Bob the element g^{x_A}.
2: Bob generates a random integer $x_B \in \{1, \ldots, \#G - 1\}$ and sends Alice the element g^{x_B}.
3: Alice computes $(g^{x_B})^{x_A} = g^{x_A x_B}$.
4: Likewise, Bob computes $(g^{x_A})^{x_B} = g^{x_A x_B}$.

It can easily be seen that a third person, who can eavesdrop on the channel, knows G, g, g^{x_A}, g^{x_B} and who can solve the DLP, can also recover $g^{x_A x_B}$. Again, solving the DLP means computing x_A or x_B from G, g, g^{x_A} and g^{x_B}. It is believed for most groups in use in cryptography recovering the Diffie-Hellman key and solving the DLP are equivalent. However, if there is a fast way to solve the DLP in G, then the cryptosystems based on it are not secure. Today, discrete logarithms in finite fields can be found in sub-exponential time, using the index-calculus method as described in [17].

2.2 Elliptic Curve Cryptosystems

In the previous section we discussed how to use algebraic groups for cryptosystems. Now we look at the special case of an elliptic curve as "our group of choice" for the DLP. The DLP for elliptic curves is usually denoted as ECDLP. An introduction to elliptic curves in cryptography can be found in [2], a comprehensive introduction to the theoretical background in [25].

An elliptic curve is the set of zeros of a cubic polynomial in two variables over some field, for our purposes some finite field $\mathbb{F}_q$. A finite field $\mathbb{F}_q$, where $q = p^n$ with p prime and $n \in \mathbb{N}$, has q elements. If $n > 1$ we call $\mathbb{F}_q$ an extension field or Galois field. The elements of $\mathbb{F}_p$, where p prime, can be denoted by $\{0, \ldots, p-1\}$, the elements of the field $\mathbb{F}_{p^n}$ can be denoted by degree $n-1$ polynomials with coefficients from $\mathbb{F}_p$. Arithmetic can be done by usual polynomial addition and multiplication with subsequent reduction modulo an irreducible polynomial. A comprehensive introduction to finite fields can be found in [15]. More details are also provided in Section 5.1.

2.2.1 General Elliptic Curves

We can write an elliptic curve as

$$E : Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6$$

with $a_i \in \mathbb{F}_p$ and call this the affine version of the Weierstrass equation. If $\mathbb{F}_p$ has characteristic $char(\mathbb{F}_p) \neq 2, 3$ this can be simplified to

$$E : Y^2 = X^3 + aX + b \tag{2.1}$$

where $a, b \in \mathbb{F}_p$ for which $4a^3 + 27b^2 \neq 0 \bmod p$. Equation (2.1) is said to be in short Weierstrass form. If the coefficients a and b are elements of $\mathbb{F}_p$ we denote the elliptic curve by $E|\mathbb{F}_p$. The points on the elliptic curve are points of the closure $\bigcup_{n\in\mathbb{N}} \mathbb{F}_{p^n}$ which solve the equation. If we define the points over $\mathbb{F}_q$ with $q = p^n$ and $n \in \mathbb{N}$ we say that all solutions $(x, y) \in {\mathbb{F}_q}^2$ are $\mathbb{F}_q$-rational. All $\mathbb{F}_q$-rational points including an additional point $\mathcal{O}$ at infinity form an abelian group which is denoted by $E(\mathbb{F}_q)$. Note that groups groups are often written with multiplication as group operation. However, for elliptic curves it is more customary to write them additively. To add two points $\mathcal{P} = (x_P, y_P) \in E(\mathbb{F}_p)$ and $\mathcal{Q} = (x_Q, y_Q) \in E(\mathbb{F}_p)$ we do the following:
if $x_P \neq x_Q$ we set

$$\lambda = \frac{y_Q - y_P}{x_Q - x_P} \tag{2.2}$$

and if $x_P = x_Q, y_P \neq 0$ we set

$$\lambda = \frac{3{x_P}^2 + a}{2y_P} \tag{2.3}$$

if $\mathcal{R} = (x_R, y_R) = \mathcal{P} + \mathcal{Q} \neq 0$, then x_R and y_R are given by the formula

$$x_R = \lambda^2 - x_P - x_Q \tag{2.4}$$

$$y_R = (x_P - x_R)\lambda - y_P \tag{2.5}$$

Furthermore we use the group law for inversion

$$-\mathcal{P} = (x_P, -y_P)$$

Multiplication of a point by a scalar is done as repeated addition

$$k \cdot \mathcal{P} = \underbrace{\mathcal{P} + \ldots + \mathcal{P}}_{\text{k times}}$$

This operation is called scalar multiplication or point multiplication and is the additive equivalent of exponentiation appearing in the previous section.

In this group operations can be efficiently calculated but the DLP is believed to be hard to solve. Compared to other public-key systems, systems based on elliptic curves need smaller key sizes for the same security level. Furthermore, there is no sub exponential algorithm known to solve the DLP in elliptic curve groups. The best known method, Pollard's rho method [17], needs $O(\sqrt{K})$ steps, where K is the number of elements (points) of the elliptic curve group. The Mordell-Weil group is a subset of general elliptic curve groups. Therefore, all encryption and signature algorithms that are available for elliptic curves are also applicable to the Mordell-Weil group, e.g., the Diffie-Hellman key exchange for elliptic curves and the Elliptic Curve Digital Signature Algorithm (ECDSA) [8].

2.2.2 Special Elliptic Curves

After general elliptic curves were introduced into cryptography there were several proposals to use special types of curves. Supersingular curves have the advantage that the group order, i.e., the number of points, can be easily calculated. However, there is a polynomial time reduction of the ECDLP on $E(\mathbb{F}_q)$ to the DLP in $\mathbb{F}_{q^l}$ for some small integer l. This is the MOV attack [16]. Therefore this type of curves is not recommended for cryptographic purposes.

Anomalous curves were introduced as they are particularly able to resist the MOV attack. For anomalous curves the number of points equals the number of

elements in the underlying field, i.e., $\#E(\mathbb{F}_q) = q$. However, an attack in linear time was also found [26]. Thus, they are no longer used for cryptographic purposes.

A recent attack on elliptic curves uses the Weil descent [4]. The attack weakens systems that define curves over composite fields with characteristic 2 or 3, i.e., fields $\mathbb{F}_{p^k}$, $k = n \cdot m$, $p = 2, 3$. It is unclear at the time of writing how the Weil descent may weaken other elliptic curve cryptosystems.

Koblitz curves were introduced in [11]. They use the group of $\mathbb{F}_{p^d}$-rational points, $E(\mathbb{F}_{p^d})$, of a curve defined over $\mathbb{F}_p$, for p relatively small. This is of particular interest if $p = 2$. Since the multiplication in Koblitz curves can be done using the theory of complex multiplication (CM), they have better performance than general elliptic curves. Furthermore, there is no attack known with complexity substantially better than the general Pollard's rho.

2.2.3 Standardization Efforts

Elliptic curve cryptosystems have been recently introduced in various standards bodies. The National Institute of Standards and Technology (NIST) published recommended ellipitic curves for Federal government use in May 1999 [20]. NIST also included elliptic curves in the Digital Signature Standard (DSS) [19]. They are also being incorporated in several other standards, such as ISO and SSL. Another example is the IEEE P1363 standard [7] which includes elliptic curve cryptosystems. We used this document as foundation for our software implementation.

Chapter 3

Background: The Mordell-Weil Group for Cryptographic Purposes

To increase the performance and enhance the security of elliptic curves Frey and Naumann [21] proposed the use of the Mordell-Weil group for cryptographic purposes. In this chapter we will give a brief overview of their idea.

Assume a finite field $\mathbb{F}_p$ with p prime and $p \equiv 4, 7 \bmod 9$ and an elliptic curve $E|\mathbb{F}_p$. Using the first isomorphism theorem for groups [3] it can be shown that we can find almost always an isomorphism

$$E(\mathbb{F}_p) \times \mathcal{A}(\mathbb{F}_p) \cong \mathcal{W}(\mathbb{F}_p) \tag{3.1}$$

We call $\mathcal{W}$ the Weil restriction and $\mathcal{A}$ the Mordell-Weil group. $\mathcal{W}$ and $\mathcal{A}$ are unique for each E. Furthermore it can be shown that

$$\mathcal{W}(\mathbb{F}_p) \cong E(\mathbb{F}_{p^3}) \tag{3.2}$$

This identifies uniquely a group

$$\mathcal{A}(\mathbb{F}_p) := \{\mathcal{P} \in E(\mathbb{F}_{p^3}) \mid \text{Trace}(\mathcal{P}) = \mathcal{O}\} \tag{3.3}$$

The Trace of $\mathcal{P}$ can be expressed as

$$\text{Trace}(\mathcal{P}) = (\pi^2 + \pi + 1)(\mathcal{P})$$

where π denotes the Frobenius map acting on a point $\mathcal{P} = (x, y)$. The Frobenius map is an endomorphism in $E(\mathbb{F}_q)$ for curves $E|\mathbb{F}_p$ with $q = p^n$

$$\pi : E(\mathbb{F}_q) \to E(\mathbb{F}_q)$$

defined by

$$\pi : (x, y) \mapsto (x^p, y^p)$$

Example 1 *Let* $E : Y^2 = X^3 + 5X + 6$ *be defined over* $\mathbb{F}_7$. *Then* $\#E(\mathbb{F}_7) = 4$ *and* $\#E(\mathbb{F}_{7^3}) = 364$. *Because of (3.1) and (3.2) we obtain* $\#\mathcal{A}(\mathbb{F}_7) = \frac{364}{4} = 91 = 7 \cdot 13$. *Therefore* $\mathcal{A}(\mathbb{F}_7)$ *has two subgroups with group order* 7 *and* 13, *respective. A generating element of* $\mathcal{A}(\mathbb{F}_7)$ *is given by* $\mathcal{Q} = (x^2 + 2x + 1, 3x^2 + 6x + 3)$, *where the field polynomial is given by* $f(x) = x^3 + 5x^2 + x + 3$.

If we apply the Frobenius map to a point $\mathcal{Q}$ we use the notation $\pi(\mathcal{Q})$, or π if we are inside of the endomorphism ring $End(\langle\mathcal{P}\rangle)$. If we apply the Frobenius map i-times to the same point $\mathcal{Q}$ we write this as $\pi^i(\mathcal{Q})$ and π^i, respective. Note that the trace maps a point $\mathcal{P}$ to a point $\mathcal{Q} = \text{Trace}(\mathcal{P}) = \pi^2(\mathcal{P}) + \pi(\mathcal{P}) + \mathcal{P}$. We now restrict our attention to the group $\mathcal{A}(\mathbb{F}_p)$. Most statements that were made earlier about general elliptic curves are also true for the Mordell-Weil group since it is a subgroup

of an elliptic curve. The group $\mathcal{A}(\mathbb{F}_p)$ should satisfy the following conditions:

1. There must be an efficient group operation in $\mathcal{A}(\mathbb{F}_p)$.

2. One should be able to compute $\#\mathcal{A}(\mathbb{F}_p)$ and with high probability there exists a subgroup with large prime order. Otherwise we can use the Pohlig-Hellman method [17] to significantly decrease the complexity to break the DLP.

3. The key length of the Mordell-Weil group should be similar in length to the key size of an elliptic curve group that offers the same level of security.

4. There has to be a base point of large order which can be found. This point is used as a generator for the cyclic subgroup to be used.

As will be shown later the number of elements of $E(\mathbb{F}_{p^2})$ or $E(\mathbb{F}_{p'})$ with $p' \approx p^2$ and $\mathcal{A}(\mathbb{F}_p)$ are similar. Using today's knowledge about attacking the elliptic curve DLP this means that these groups are security equivalent. Therefore we will compare the Mordell-Weil group $\mathcal{A}(\mathbb{F}_p)$ to the elliptic curve group $E(\mathbb{F}_{p^2})$. In the following we will discuss these items and then generalize the idea of Frey and Naumann.

3.1 The Order of the Mordell-Weil Group $\mathcal{A}(\mathbb{F}_p)$

We require the order of the Mordell-Weil group to be large enough such that an attack will fail. Pollard's rho algorithm needs about $\sqrt{\#\mathcal{A}}$ steps to solve the DLP in a group of order $\#A$ [17]. The Pohlig-Hellman method [17] reduces this to the square root of the largest prime factor of $\#\mathcal{A}$. Therefore $\#\mathcal{A}$ must contain a large prime factor, i.e., there has to be a subgroup of large prime order. We calculate the order by using the Hasse-Weil theorem, which is stated as follows:

Theorem 1 (Hasse-Weil) *The number of points on an elliptic curve $E(\mathbb{F}_q)$ satisfies*

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

□

In the following we will we say that $\#E(\mathbb{F}_q) \approx q$. Let t be the trace defined by

$$\#E(\mathbb{F}_q) = 1 + q - t. \tag{3.4}$$

Let α be the complex number which satisfies

$$\alpha^2 - t\alpha + p = 0 \tag{3.5}$$

Let $\bar{\alpha}$ be the complex conjugate of α. We call α the Weil number. We know that $E(\mathbb{F}_p) \times \mathcal{A}(\mathbb{F}_p) \cong \mathcal{W}(\mathbb{F}_p)$ and $\mathcal{W}(\mathbb{F}_p) \cong E(\mathbb{F}_{p^3})$. Using the zeta function [2] we obtain

$$\#\mathcal{A}(\mathbb{F}_p) = \frac{\#\mathcal{W}(\mathbb{F}_p)}{\#E(\mathbb{F}_p)} = \frac{(1-\alpha^3)(1-\bar{\alpha}^3)}{(1-\alpha)(1-\bar{\alpha})} \approx \frac{p^3}{p} = p^2 \tag{3.6}$$

3.2 Calculating in the Mordell-Weil Group $\mathcal{A}(\mathbb{F}_p)$

In this section we show how to calculate in $\mathcal{A}(\mathbb{F}_p)$ and compare this to calculating in $E(\mathbb{F}_{p^2})$. An elliptic curve addition over a field $\mathbb{F}_{p^n}$ needs about $\log^2(p^n) = n^2 \log^2(p)$ elementary operations since we need an inversion, multiplications, and additions over $\mathbb{F}_{p^n}$ as described in formulas (2.2) – (2.5). The complexity of these operations are described in [12]. Hence, if we have two points $\mathcal{P}, \mathcal{Q} \in \mathcal{A}(\mathbb{F}_p) \subset E(\mathbb{F}_{p^3})$ we can perform an elliptic curve addition with $9 \log^2(p)$ operations. To add two points on an elliptic curve $E(\mathbb{F}_{p^2})$, which has similar security, we need only $4 \log^2(p)$ operations.

The point multiplication $k \cdot \mathcal{P}$ is the core operation in elliptic curve cryptosystems. To calculate the multiple of a point $k \cdot \mathcal{P} \in E(\mathbb{F}_{p^2})$, $\mathcal{P} \in E(\mathbb{F}_{p^2})$, k an integer, we use the double-and-add method [17]. On average this method requires $\log(k)$ point doubles and $\frac{1}{2}\log(k)$ point additions. Algorithm 2 summarizes this method.

Algorithm 2 Binary double-and-add method

INPUT: A non-negative integer k and an elliptic point $\mathcal{P} \in E(\mathbb{F}_p)$.
OUTPUT: The elliptic point $k \cdot \mathcal{P} \in E(\mathbb{F}_p)$.

1: Let $(k_{l-1}k_{l-2}\ldots k_1k_0)_2$ be the binary representation of k, where the most significant bit k_{l-1} is 1.
2: $\mathcal{R} \leftarrow \mathcal{O}$
3: $\mathcal{S} \leftarrow \mathcal{P}$
4: **for** $i = 0$ to $l - 1$ **do**
5: **if** $k_i = 1$ **then**
6: $\mathcal{R} \leftarrow \mathcal{R} + \mathcal{S}$
7: **end if**
8: $\mathcal{S} \leftarrow 2\,\mathcal{S}$
9: **end for**
10: **Return** $\mathcal{R}$

If we choose $\mathcal{P} \in E(\mathbb{F}_{p^2})$, then $k \approx p^2$ and we need $\log(p^2) + \frac{1}{2}\log(p^2) = 3\log(p)$ elliptic curve group operations over $\mathbb{F}_{p^2}$. Using the same method for a point $\mathcal{P} \in \mathcal{A}(\mathbb{F}_p)$ and $k \approx p^2$ is obviously much slower as the group operation requires arithmetic in $\mathbb{F}_{p^3}$. To calculate in the Mordell-Weil group a slightly different approach may be used. First, we describe the background. Let $q = ord(\mathcal{P})$ be prime. Then define

$$C := \min\left\{\frac{q}{t+p}, \frac{p-1}{\gcd(p-1, t+1)}\right\} \tag{3.7}$$

with t being the trace of $E(\mathbb{F}_p)$ as defined in (3.4). Note that $q \approx p^2$, $t + p \approx p$, and in order for C to be large we require $\gcd(p-1, t+1)$ to be small. Now we use a map

$$\phi : \mathbb{Z}^2 \to End(\langle \mathcal{P} \rangle)$$

via

$$\phi : (\lambda_0, \lambda_1) \mapsto \lambda_0 \; id + \lambda_1 \pi$$

where π denotes the Frobenius map, and id the identity map. It can be shown that there are at least C^2 endomorphisms of this form in $End(\langle \mathcal{P} \rangle)$. Let $X := \{(\lambda_0, \lambda_1) \mid 0 \leq \lambda_i < C\} \subset \mathbb{Z}^2$. We know from (3.3) that

$$\pi^2 + \pi + 1 = 0 \text{ in } End(\langle \mathcal{P} \rangle) \tag{3.8}$$

Furthermore we know the characteristic polynomial of π [2] is

$$\pi^2 - t\pi + p = 0 \text{ in } End(\langle \mathcal{P} \rangle) \tag{3.9}$$

where t is the trace of the curve as defined in (3.4). Then we show:

Proposition 1 (Naumann and Frey [21]) *The restriction of ϕ to X is one-to-one.*

Proof. *Let be $(\lambda_0, \lambda_1), (\lambda_0', \lambda_1') \in X$ with*

$$\lambda_0 + \lambda_1 \pi = \lambda_0' + \lambda_1' \pi \; in \; End(\langle \mathcal{P} \rangle)$$

Then for $\alpha_i := \lambda_i - \lambda_i'$

$$\alpha_0 + \alpha_1 \pi = 0 \; with \; |\alpha_i| < C \tag{3.10}$$

By subtracting (3.9) from (3.8) we obtain

$$(t+1)\pi = p - 1$$

Multiplying 3.10 with $(t+1)$ *and using* $(t+1)\pi = p-1$ *we get*

$$(t+1)\alpha_0 + (p-1)\alpha_1 = 0 \text{ in } End(\langle\mathcal{P}\rangle)$$

It is known that the endomorphism ring $End((\langle\mathcal{P}\rangle)$ *is isomorphic to* $\mathbb{F}_q$ *as ring via* $f(k) = k \cdot id(\mathcal{P}), k = 0, \ldots q-1$. *Since* $|\alpha_i| < \frac{q}{t+p}$ *we see that the former equation is also true in* $\mathbb{Z}$

$$(t+1)\alpha_0 + (p-1)\alpha_1 = 0 \text{ in } \mathbb{Z}$$

This is true if $\alpha_i = 0$ *or if* α_0 *divisible by* $\frac{p-1}{\gcd(p-1,\ t+1)}$ *and* α_1 *divisible by* $\frac{t+1}{\gcd(p-1,\ t+1)}$. *Since we chose* $\alpha_i < C$ *it follows that*

$$\alpha_0 = \alpha_1 = 0$$

□

As described in [21] we introduce a new point multiplication method. We define $(\lambda_0, \lambda_1) \cdot \mathcal{Q} := \phi(\lambda_0, \lambda_1)(\mathcal{Q}) = \lambda_0\mathcal{Q} + \lambda_1\pi(\mathcal{Q})$. Note that the endomorphism ring $End(\langle\mathcal{P}\rangle)$ is isomorphic to $\mathbb{F}_q$ where the later one is considered as a ring. Therefore $\pi(\mathcal{Q}) = x \cdot \mathcal{Q}$ and hence $\lambda_0\mathcal{Q} + \lambda_1\pi(\mathcal{Q}) = (\lambda_0 + x\ \lambda_1)\mathcal{Q}$. If $C \approx p$, i.e., $\gcd(p-1,\ t+1)$ small, we can replace the operation $k \cdot \mathcal{P}$ by by the new point multiplication method without losing security. Algorithm 3 shows how this can be done via a modified double-and-add method.

Note that $S = (2^i\mathcal{P}, \pi(2^i\mathcal{P}), -\pi^2(2^i\mathcal{P}))$ at the end of iteration i and $2^i\mathcal{P} + \pi(2^i\mathcal{P}) = -\pi^2(2^i\mathcal{P})$. Since the application of the Frobenius map has negligible cost as described later, the calculation of $S = (S_1, S_2, S_3)$ is done by

1. $S_1 \leftarrow S_1 + S_1$
2. $S_2 \leftarrow \pi(S_1)$

Algorithm 3 Basic double-and-add method for the Mordell-Weil group

INPUT: A tuple (λ_0, λ_1) of non-negative integers and an elliptic point $\mathcal{P} \in \mathcal{A}(\mathbb{F}_p)$.
OUTPUT: The elliptic point $(\lambda_0, \lambda_1) \cdot \mathcal{P} = \lambda_0 \cdot \mathcal{P} + \lambda_1 \cdot \pi(\mathcal{P}) \in \mathcal{A}(\mathbb{F}_p)$

.

1: Let $(\lambda_{i,l-1}\lambda_{i,l-2} \ldots \lambda_{i,1}\lambda_{i,0})_2$ be the binary representation of λ_i, where the most significant bit $\lambda_{i,l-1}$ equals 1 for $i = 0$ and $i = 1$.
2: $\mathcal{R} = \mathcal{O}$
3: $S \leftarrow (S_1, S_2, S_3) = (\mathcal{P}, \pi\mathcal{P}, -\pi^2\mathcal{P})$
4: **for** $i = 0$ to $l - 1$ **do**
5: **if** $\lambda_{0,i} = 1$ and $\lambda_{1,i} = 1$ **then**
6: $\mathcal{R} \leftarrow \mathcal{R} + S_3$
7: **else if** $\lambda_{0,i} = 1$ and $\lambda_{1,i} = 0$ **then**
8: $\mathcal{R} \leftarrow \mathcal{R} + S_1$
9: **else if** $\lambda_{0,i} = 0$ and $\lambda_{1,i} = 1$ **then**
10: $\mathcal{R} \leftarrow \mathcal{R} + S_2$
11: **end if**
12: $S \leftarrow S + S = (S_1 + S_1, S_2 + S_2, S_3 + S_3)$
13: **end for**
14: **Return** $\mathcal{R}$

3. $S_3 \leftarrow -\pi(S_2)$

Assuming that $\lambda_i \approx p$, on average there are $\frac{3}{4}$ point additions and 1 point doubling in each iteration, altogether $\frac{3}{4}\log(p) + \log(p) = \frac{7}{4}\log(p)$ elliptic curve group operations over $\mathbb{F}_{p^3}$. Chapter 5 and 6 present more sophisticated point multiplication methods and analyze their complexity.

The new point multiplication method shares an important characteristic with scalar multiplication. In particular,

$$(\lambda_0, \lambda_1) \cdot (\lambda_0', \lambda_1') \cdot \mathcal{P} = (\lambda_0', \lambda_1') \cdot (\lambda_0, \lambda_1) \cdot \mathcal{P}$$

Note that this is required by the Diffie-Hellman key exchange protocol. The secret key for this protocol is a pair (λ_0, λ_1) instead of a scalar k. For a signature scheme like ECDSA we have to be able to compute (λ_0', λ_1') for given (λ_0, λ_1) such that $(\lambda_0', \lambda_1')(\lambda_0, \lambda_1)\mathcal{P} = \mathcal{P}$ though. This cannot be done efficiently yet.

3.3 Key Length and Security

After looking at algebraic properties of the Mordell-Weil group we now consider its significance for cryptosystems. We define the key length as the bit length of a private key. The private key is an integer in the order of the number of elements of the group. As a measure of security we use the average number of steps for a successful attack, i.e., the complexity of the attack. As described before the best known attack needs about $\sqrt{\#G}$ steps to break the system, where $\#G$ is the number of elements of the group G. Let us first consider the group $E(\mathbb{F}_p)$. The elements of $E(\mathbb{F}_p)$ are the points on E and therefore the best attack will take on average $\sqrt{\#E(\mathbb{F}_p)}$ steps. According to the Hasse-Weil Theorem $\#E(\mathbb{F}_p) \approx p$, thus the attack will take about $\sqrt{p}$ steps. The private key is an integer in the order of p and it is specified by $\lceil \log(p) \rceil$ bits. To represent a public key, i.e., a point on $E(\mathbb{F}_p)$ we need the X coordinate and the sign of the Y coordinate, therefore we need $\lceil \log(p) \rceil + 1$ bits.

Now we compare the elliptic curve $E(\mathbb{F}_{p^2})$ to the Mordell-Weil group $\mathcal{A}(\mathbb{F}_p)$. Both have approximately p^2 elements, i.e., a successful attack will take on average p steps in both cases. The private key is in the order of magnitude of p^2 and has a length of $\lceil 2 \log(p) \rceil$ bits. Note that in the second case the private key is a 2-tuple (λ_0, λ_1) with $0 \leq \lambda_i < C$, where $C \approx p$ as defined in (3.7).

We compare now the bit length of the public keys which are group elements. In the first case an element of $E(\mathbb{F}_{p^2})$ can be represented as a value $X \in \mathbb{F}_{p^2}$ together with an additional bit for the sign of Y. Therefore we get a length of $2 \lceil \log(p) \rceil + 1$ bits. In the second case we can represent an element of $\mathcal{A}(\mathbb{F}_p)$ as a value $X \in \mathbb{F}_{p^3}$ and an additional bit for the sign, thus we need $3 \lceil \log(p) \rceil + 1$ bits. However, we can shorten this according to Frey and Naumann [21]. Let X be an element of order 9 which generates the extension $\mathbb{F}_{p^3} \supset \mathbb{F}_p$. This is equivalent to the requirement

$p \equiv 4, 7 \bmod 9$, or interpreted in a different way, 9 is a divisor of the order of $\mathbb{F}_{p^3}$, $p^3 - 1$, but not a divisor of $p - 1$. Let $\mu := X^3 \in \mathbb{F}_p$. We can denote the curve function as

$$A : 3{x_0}^4 + (6a - 18\mu x_1 x_2){x_0}^2 + (12b + 12(\mu {x_1}^3 + \mu^2 {x_2}^3))x_0 - (a + 3\mu x_1 x_2)^2 = 0$$

The values a and b are the coefficients of the curve in short Weierstrass form. Then A is an irreducible polynomial in $\mathbb{F}_p(x_1, x_2)[x_0]$ with $x_i \in \mathbb{F}_p$, and we can encode a point $\mathcal{P} \in \mathcal{A}(\mathbb{F}_p)$ by a vector

$$(x_1, x_2, v)$$

where $1 \leq v \leq 4$ determines the number of the zero of A in x_0 using a fixed ordering of the zeroes. Then we have a key length of $2\lceil \log(p) \rceil + 2$, which, when compared to the first case, has the same security with only one more bit. However, the transformation

$$\mathcal{P} = (x, y) \in \mathcal{A}(\mathbb{F}_p) \leftrightarrow (x_1, x_2, v)$$

requires the factorization of a degree 4 polynomial over $\mathbb{F}_p$, which is expensive. Therefore we use the traditional point representation.

Note that since the Mordell-Weil group does not contain the elliptic curve $E(\mathbb{F}_p)$ which we used to construct it, i.e., $E(\mathbb{F}_p) \not\subset \mathcal{A}(\mathbb{F}_p)$, it seems that there are less possibilities for an attack and therefore $\mathcal{A}(\mathbb{F}_p)$ might be more secure than an elliptic curve $E(\mathbb{F}_{p^2})$.

3.4 Computing a Base Point

Assume we have a Mordell-Weil group $\mathcal{A}(\mathbb{F}_p)$. For our purposes we need a cyclic group, i.e., a group which is generated by one element. This element g, which is

called primitive element or generator, generates the group $G = \langle g \rangle$. Note that the order of this element has to be a large prime. To find a point $\mathcal{P} = (x_P, y_P)$ on an elliptic curve $E(\mathbb{F}_p)$ we just choose a value $x_P \in \mathbb{F}_p$ at random and use it in the curve equation to calculate $f(x_P)$. According to the Hasse-Weil Theorem [25] $f(x_P)$ is a quadratic residue with probability of about 50%, i.e., there exists a $y_P \in \mathbb{F}_p$ with ${y_P}^2 = f(x_P)$. Actually, if $f(x_P) \neq 0$ we find no or two solutions. If we find two solutions we have to take one of them and check the order. If this search fails, we just select another random x_P and try it again until we find a point $\mathcal{P}$ of large order.

To obtain a point in the Mordell-Weil group we do the following. First we have to find a point $\mathcal{P} \in E(\mathbb{F}_{p^3}) - E(\mathbb{F}_p), \mathcal{P} \neq \mathcal{O}$ as described in the previous paragraph. Then we calculate the point

$$\mathcal{Q} := \mathcal{P} - \pi(\mathcal{P})$$

where π is the Frobenius map. The point $\mathcal{Q}$ is obviously an element in $E(\mathbb{F}_{p^3})$. Furthermore we get $\text{Trace}(\mathcal{Q}) = \text{Trace}(\mathcal{P} - \pi(\mathcal{P})) = (\pi^2 + \pi + 1)(\mathcal{P} - \pi(\mathcal{P})) = \pi^2(\mathcal{P}) + \pi(\mathcal{P}) + \mathcal{P} - \pi^3(\mathcal{P}) - \pi^2(\mathcal{P}) - \pi(\mathcal{P}) = \mathcal{P} - \pi^3(\mathcal{P}) = \mathcal{O}$, therefore $\mathcal{Q} \in \mathcal{A}(\mathbb{F}_p)$ as desired. Since we chose $P \notin E(\mathbb{F}_p)$ we assure that $\mathcal{Q} \neq \mathcal{O}$. At the end we have to check whether our point has a large order by using (3.6). We repeat this process until we achieve a point of sufficiently large order. Every point in the Mordell-Weil group can be chosen by using this method as we show in the following proposition which extends the work of Frey and Naumann.

Proposition 2 *Let $\mathcal{Q} \in \mathcal{A}(\mathbb{F}_p)$, $\mathcal{Q} \neq \mathcal{O}$, with $\pi^2(\mathcal{Q}) + \pi(\mathcal{Q}) + \mathcal{Q} = \mathcal{O}$. Let $\pi(\mathcal{Q}) = x \cdot \mathcal{Q}$ for some $x \in \mathbb{N}$ and $(1 - x)\ y \equiv 1 \bmod q$ where $q = ord(\mathcal{Q})$, q prime. Furthermore let $\mathcal{R} \in E(\mathbb{F}_p)$ with $r = ord(\mathcal{R})$. Then $\mathcal{P} = y \cdot \mathcal{Q} + i \cdot \mathcal{R}$, $i \in \mathbb{N}$, satisfies $\mathcal{Q} = \mathcal{P} - \pi(\mathcal{P})$.*

Proof.

$$\begin{aligned}\mathcal{P} - \pi(\mathcal{P}) &= y \cdot \mathcal{Q} + i \cdot \mathcal{R} - \pi(y \cdot \mathcal{Q} + i \cdot \mathcal{R}) = y \cdot \mathcal{Q} - y \cdot \pi(\mathcal{Q}) + i \cdot \mathcal{R} - i \cdot \pi(\mathcal{R}) \\ &= y(1-x)\mathcal{Q} + i \cdot \mathcal{R} - i \cdot \pi(\mathcal{R}) = \mathcal{Q}\end{aligned}$$

□

Note that there are at least r points $\mathcal{P}_i = y\ \mathcal{Q} + i\ \mathcal{R}$, $0 \leq i < r$ that are mapped on $\mathcal{Q}$. By choosing an appropriate point $\mathcal{R}$ with large order $r \approx p$ we can conclude that finding points $\mathcal{Q} \in \mathcal{A}(\mathbb{F}_p)$ using the above described method is almost random.

Chapter 4

The Mordell-Weil Group over Higher Extension Degrees

Frey and Naumann considered an extension of degree 3 for the Mordell-Weil group, i.e., $\mathcal{A}(\mathbb{F}_p) \subset E(\mathbb{F}_{p^3})$. We now extend this to higher extension degrees which bears computational advantages as will be shown later. First we define the Mordell-Weil group for extension degree d:

$$\mathcal{A}(\mathbb{F}_{p^d}) := \{\mathcal{P} \in E(\mathbb{F}_{p^d}) \mid \text{Trace}(\mathcal{P}) = \mathcal{O}\}$$

The Trace of $\mathcal{P}$ for degree d is defined as

$$\text{Trace}(\mathcal{P}) = \sum_{i=0}^{d-1} \pi^i(\mathcal{P})$$

where $\pi(\mathcal{P})$ denotes the Frobenius map acting on a point $\mathcal{P}$. For convenience we define that $\pi^0 = id$. We require the degree d of the extension to be a prime, otherwise an attack based on the Weil descent [4] may be applicable. Note that $E(\mathbb{F}_p) \not\subset \mathcal{A}(\mathbb{F}_{p^d})$ which might be a security advantage.

Almost always there exists an isomorphism $E(\mathbb{F}_p) \times \mathcal{A}(\mathbb{F}_{p^d}) \cong \mathcal{W}(\mathbb{F}_p)$ with $\mathcal{W}(\mathbb{F}_p) \cong E(\mathbb{F}_{p^d})$ because of the first isomorphism theorem [21]. The order of the Mordell-Weil group for arbitrary extension degree is calculated according to (3.6):

$$\#\mathcal{A}(\mathbb{F}_{p^d}) = \frac{\#\mathcal{W}(\mathbb{F}_p)}{\#E(\mathbb{F}_p)} = \frac{(1-\alpha^d)(1-\bar{\alpha}^d)}{(1-\alpha)(1-\bar{\alpha})}$$

where α is the Weil number. Since $\mathcal{W}(\mathbb{F}_p) \cong E(\mathbb{F}_{p^d})$ we obtain

$$\#\mathcal{A}(\mathbb{F}_{p^d}) \approx \frac{p^d}{p} = p^{d-1} \text{ for any } d$$

A base point can also be found in the same manner as before, i.e., by choosing a point $\mathcal{P} \in E(\mathbb{F}_{p^d}) - E(\mathbb{F}_p), \mathcal{P} \neq \mathcal{O}$, and computing $\mathcal{Q} := \mathcal{P} - \pi(\mathcal{P})$. Proposition 2 extends easily to the general case.

To accelerate point multiplication in the Mordell-Weil group with $d = 3$ we replaced the usual point multiplication by a multiplication with a tuple (λ_0, λ_1) defined via $(\lambda_0, \lambda_1) \cdot \mathcal{P} := \lambda_0 \mathcal{P} + \lambda_1 \pi(\mathcal{P})$. We use the same idea for higher extension degrees. Let

$$\phi : \mathbb{Z}^{d-1} \rightarrow End(\langle \mathcal{P} \rangle)$$

via

$$\phi : (\lambda_0, \ldots, \lambda_{d-2}) \mapsto \sum_{i=0}^{d-2} \lambda_i \pi^i \tag{4.1}$$

We define the new scalar multiplication as $(\lambda_0, \ldots, \lambda_{d-2}) \cdot \mathcal{Q} := \phi(\lambda_0, \ldots, \lambda_{d-2})(\mathcal{Q}) = \sum_{i=0}^{d-2} \lambda_i \pi^i(\mathcal{Q})$. First one has to check that we do not lose security, i.e., that we can find large C such that ϕ is injective for $0 \leq \lambda_i < C$. If we can find $C \approx p$ then we would get an effective key space of size p^{d-1}. We show this for $d = 5$.

Definition 1 *Let $\mathcal{A}(\mathbb{F}_{p^5})$ be the Mordell-Weil group defined over $\mathbb{F}_{p^5}$ and $\mathcal{P} \in \mathcal{A}(\mathbb{F}_{p^5})$ with $q = ord(\mathcal{P})$. Let $f = p^2 - p(1+t+t^2)+1$ and $g = p(1+2t)-1-t-t^2-t^3$*

where t is the trace of $E(\mathbb{F}_p)$. Then C is p-limited if the following is true:

(i) $t > 0$

(ii) $f > 0$ *and* $g > 0$

(iii) $g,\ f,\ gp + tf$ *and* $gtp + pf + ft^2$ *are pairwise coprime*

(iv) $C \leq \frac{q}{g+gp+gtp+f+ft+fp+ft^2}$

(v) $C \leq \frac{gtp+pf+ft^2}{g+f+gp+tf}$

(vi) $C \leq \frac{f}{1+p}$

(vii) $C \leq \frac{g}{1+t}$

(viii) $C \leq p$

□

Note that $(viii)$ follows immediately from (vi) and (i).

Proposition 3 *Let C be p-limited as defined in Definition 1. Let*

$$X := \{(\lambda_0, \lambda_1, \lambda_2, \lambda_3) \mid 0 \leq \lambda_i < C\} \subset \mathbb{Z}^4.$$

Then the restriction of ϕ to X is injective.

Proof. *Let (i)–(viii) denote the conditions for p-limited C. We know that*

$$\pi^4 + \pi^3 + \pi^2 + \pi + 1 = 0 \text{ in } End(\langle \mathcal{P} \rangle)$$

and

$$\pi^2 - t\pi + p = 0 \text{ in } End(\langle \mathcal{P} \rangle) \tag{4.2}$$

Multiplying (4.2) by $\pi^2 + \pi(1+t) + (1-p+t+t^2)$ and subtracting it from the first

equation we obtain

$$-\pi(p(2t+1)-t^3-t^2-t-1)+p^2-p(t^2+t+1)+1=0$$

Let $g=p(2t+1)-t^3-t^2-t-1$ *and* $f=p^2-p(t^2+t+1)+1$. *Then we write*

$$g\pi=f$$

Let $(\lambda_0,\lambda_1,\lambda_2,\lambda_3),(\lambda_0',\lambda_1',\lambda_2',\lambda_3')\in X$ *with*

$$\lambda_0+\lambda_1\pi+\lambda_2\pi^2+\lambda_3\pi^3=\lambda_0'+\lambda_1'\pi+\lambda_2'\pi^2+\lambda_3'\pi^3$$

Then for $\alpha_i:=\lambda_i-\lambda_i'$

$$\alpha_0+\alpha_1\pi+\alpha_2\pi^2+\alpha_3\pi^3=0 \text{ with } |\alpha_i|<C$$

We shorten this using (4.2) to

$$\alpha_0+\alpha_2p+\alpha_3tp+\pi(\alpha_1+\alpha_2t+\alpha_3p+\alpha_3t^2)=0 \text{ in } End(\langle\mathcal{P}\rangle)$$

and substitute π

$$g(\alpha_0+\alpha_2p+\alpha_3tp)+f(\alpha_1+\alpha_2t+\alpha_3p+\alpha_3t^2)=0 \text{ in } End(\langle\mathcal{P}\rangle)$$

Because of (iv) the previous equation is also true in $\mathbb{Z}$.

$$g(\alpha_0+\alpha_2p+\alpha_3tp)+f(\alpha_1+\alpha_2t+\alpha_3p+\alpha_3t^2)=0 \text{ in } \mathbb{Z}$$

We obtain

$$\alpha_0 g + \alpha_1 f + \alpha_2(gp + tf) + \alpha_3(gtp + pf + ft^2) = 0$$

Since we assume (iii) there are no common divisors. We have to show that besides the trivial solution all solution involve at least one α_i *that is not smaller than* C*. Choosing* $|\alpha_3| > 0$ *yields that one of* $|\alpha_0|$*,* $|\alpha_1|$ *or* $|\alpha_2|$ *has to be larger or equal to* C *because of (v). Therefore we assume* $\alpha_3 = 0$*. The equation*

$$\alpha_0 g + \alpha_1 f + \alpha_2(gp + af) = 0$$

can be written as

$$g(\alpha_0 + \alpha_2 p) + f(\alpha_1 + t\alpha_2) = 0$$

All possible solutions satisfy

$$\alpha_0 + \alpha_2 p = \pm fd$$

and

$$\alpha_1 + t\alpha_2 = \mp gd$$

where d *is an arbitrary integer. If* $d \neq 0$ *then* $|\alpha_0|$*,* $|\alpha_1|$ *or* $|\alpha_2|$ *has to be larger or equal to* C *because of (vi) and (vii). If* $d = 0$ *then* $|\alpha_0| = |\alpha_2 p|$*. Because of (viii) that is not possible. Therefore* $\alpha_i = 0$ *is the only solution.* □

The proposition shows that an appropriate curve parameter choice yields $C \approx p$ and hence a key space of p^4. We have to select the curve carefully. The trace t must be positive and small, and g, f, $gp + tf$ and $gtp + pf + ft^2$ must be pairwise coprime. However, other choices might not compromise the security of the system. In practice one might choose parameters such that g, f, $gp + tf$ and $gtp + pf + ft^2$ have a pairwise small greatest common divisor. A curve with negative trace that is

close to 0 might also not weaken the system. The curve $E(\mathbb{F}_{p^d}) \supset \mathcal{A}(\mathbb{F}_{p^d})$ has to be resistant to any known attack, and there must be a cyclic subgroup of large prime order in $\mathcal{A}(\mathbb{F}_{p^d})$. Using the LiDIA package [14] we found reasonable curves quickly. An example is given in Chapter 7. Now we can speed up the multiplication using Algorithm 4 to calculate $(\lambda_0, \ldots, \lambda_{d-2}) \cdot \mathcal{P}$.

Algorithm 4 Point multiplication for higher extension degrees

INPUT: A $d-1$ tuple $(\lambda_0, \ldots, \lambda_{d-2})$ of non-negative integers and an elliptic curve point $\mathcal{P} \in \mathcal{A}(\mathbb{F}_{p^d})$.
OUTPUT: The elliptic point $(\lambda_0, \ldots, \lambda_{d-2}) \cdot \mathcal{P} \in \mathcal{A}(\mathbb{F}_{p^d})$.

1: Let $(\lambda_{i,l-1}\lambda_{i,l-2}\ldots\lambda_{i,1}\lambda_{i,0})_2$ be the binary representation of λ_i where the most significant bit $\lambda_{i,l-1}$ is 1 for any i.
2: $\mathcal{R} \leftarrow \mathcal{O}$
3: $\mathcal{S} \leftarrow \mathcal{P}$
4: **for** $i = 0$ to $l-1$ **do**
5: $\quad\mathcal{T} \leftarrow \mathcal{S}$
6: $\quad$**for** $j = 0$ to $d-2$ **do**
7: $\quad\quad$**if** $\lambda_{j,i} = 1$ **then**
8: $\quad\quad\quad\mathcal{R} \leftarrow \mathcal{R} + \mathcal{T}$
9: $\quad\quad$**end if**
10: $\quad\quad$**if** $j \neq d-2$ **then**
11: $\quad\quad\quad\mathcal{T} \leftarrow \pi(\mathcal{T})$
12: $\quad\quad$**end if**
13: $\quad$**end for**
14: $\quad\mathcal{S} \leftarrow 2\mathcal{S}$
15: **end for**
16: **Return** $\mathcal{R}$

The advantage of higher extension degrees are clear. We save point doublings and decrease the overhead when doing point arithmetic in $E(\mathbb{F}_{p^d})$. For example, let $p \approx 2^{80}$ and $p' \approx 2^{40}$. A point multiplication in $\mathcal{A}(\mathbb{F}_{p^3})$ requires 80 point doublings in $E(\mathbb{F}_{p^3})$ with $p^3 \approx 240$. One point multiplication in $\mathcal{A}(\mathbb{F}_{p'^5})$ requires only 40 point doublings in $E(\mathbb{F}_{p'^5})$ with $p'^5 \approx 200$. Furthermore it seems that a point multiplication in $\mathcal{A}(\mathbb{F}_{p'^5})$ can be done faster than in $E(\mathbb{F}_{p'^4})$. A thorough analysis of the complexity is carried out in Chapter 6.

We would like to stress the implications of using a Mordell-Weil group with higher extension degree. First, since the group order is approximately p^{d-1}, the bit length of the base field is by a factor $d-1$ shorter than an ECC with similar group order. For instance, for $d = 5$, a prime field with a bit length of about 40 bits would provide a cyclic group for a DL system with an approximate order of 2^{160}. Second, the private key is not a single integer any more but a $d-1$ tuple of integers, where every integer has approximately $\log(p)$ bits. For example, for $d = 5$ and a desired group order of about 2^{160}, the private key would consist of four integers $\lambda_0, \cdots, \lambda_3$, where each integers has a bit length of about 40 bits.

Chapter 5

Arithmetic

When looking at elliptic curves there are several important issues which affect the speed of an implementation:

- choice of the underlying field
- choice elliptic curve parameters
- method of point representation
- choice of algorithms for point operations

All of the above items have to be considered carefully since they are crucial for performance and security. In the following we discuss these points and explain their importance for the Mordell-Weil group.

5.1 The Underlying Field

Elliptic curves are defined over an underlying field. For cryptographic purposes this field is finite. Usually there are two types of field to choose from: binary or

prime fields. For the Mordell-Weil group we use an extension field with large prime characteristic.

As mentioned before we define the Mordell-Weil group $\mathcal{A}(\mathbb{F}_{p^d})$ to be a subset of $E(\mathbb{F}_{p^d})$ where p and d are primes. To add two points in the Mordell-Weil group we do arithmetic in the underlying field $\mathbb{F}_{p^d}$. Therefore fast field arithmetic is crucial for efficient elliptic curve implementations. One possibility to speed-up elliptic curve cryptosystems is to choose the underlying field with advantageous properties, e.g., one can choose the field of integers modulo a Mersenne prime, since modular reduction is particularly efficient in that case [13]. The following lemma is the basis for arithmetic in a finite field. All facts in this section are from [15].

Lemma 1 *Let $f(X) \in \mathbb{Z}_p[X]$ be an irreducible polynomial of degree m. Then $\mathbb{Z}_p[X]/(f(X))$ is a finite field of order p^m. Addition and multiplication of polynomials is performed modulo $f(X)$.*

5.1.1 Element Representation

A commonly used representation for the elements of a finite field is a polynomial basis representation. Each element of the finite field $\mathbb{F}_{p^d}$ is represented by a polynomial $P(X)$ of degree at most $d-1$. The coefficients are elements of $\mathbb{F}_p$. To optimize computation in the underlying field we choose the field polynomial $f(X)$ to be a binomial of the form $X^d - \mu$ with $\mu \in \mathbb{F}_p$ as proposed in [1]. We choose X to be an element of order d^2 that generates the extension $\mathbb{F}_{p^d} \supset \mathbb{F}_p$. Therefore $\mu^d = 1$. This is equivalent to choosing

$$p \equiv id + 1 \bmod d^2 \text{ for } i = 1, \ldots, d-1 \tag{5.1}$$

For example, for $d = 3$ we choose $p \equiv 4, 7 \bmod 9$, and for $d = 5$ we use $p \equiv 6, 11, 16, 21 \bmod 25$. This choice ensures that the Frobenius map can be computed efficiently as described later. To find such a μ we use the following lemma:

Lemma 2 *Let G be a group. If the order of $a \in G$ is t, then the order of a^k is $t/\gcd(t, k)$.*

Suppose p and d are given with $d|p-1$, $d^2 \not| p-1$ and $d^2|p^d-1$. First we search for a primitive element $a \in \mathbb{F}_p$ of order $p-1$. This is easy since there are $\Phi(p-1)$ elements of order $p-1$ where Φ is the Euler function. Since d^2 divides p^d-1, we can calculate $\mu = a^k$ with $k = (p-1)/d$. The binomial $X^d - \mu$ is irreducible due to the following theorem:

Theorem 2 *Let $t \geq 2$ be an integer and $\mu \in \mathbb{F}_p^*$. Then the binomial $X^t - \mu$ is irreducible in $\mathbb{F}_p[X]$ if and only if the following two conditions are satisfied: (i) each prime factor of t divides the order e of μ in $\mathbb{F}_p^*$, but not $(p-1)/e$; (ii) $p \equiv 1 \bmod 4$ if $t \equiv 0 \bmod 4$.*

Since $t = e$, $d^2 \not| p-1$ and (5.1) the binomial $X^d - \mu$ is irreducible.

5.1.2 Field Arithmetic in $\mathbb{F}_{p^d}$

In a field there is addition, subtraction and multiplication defined for all elements, and every element except the zero element has a multiplicative inverse. Since the field elements are represented by polynomials the field operations can be implemented as polynomial operations with modular reduction afterwards. To add two elements $A(X) \in \mathbb{F}_{p^d}$ and $B(X) \in \mathbb{F}_{p^d}$ the following is done

$$A(X) +_F B(X) = (A(X) +_P B(X)) \mod f(X)$$

where $+_F$ is the field addition, $+_P$ the usual polynomial addition and $f(x)$ the field polynomial. Field multiplication is done similarly:

$$A(X) \cdot_F B(X) = (A(X) \cdot_P B(X)) \mod f(X)$$

Inversion can be done using the extended Euclidean algorithm or Fermat's little theorem.

5.2 Elliptic Curve Parameters

The security of an elliptic curve cryptosystem depends on the parameters of the elliptic curve. Therefore we have to choose the elliptic curve in such a way that the cryptosystem is not vulnerable against any known attacks. For the Mordell-Weil group we have to ensure further properties.

The Mordell-Weil group is defined over an elliptic curve $E(\mathbb{F}_p)$. The curve should be non-supersingular and non-anomalous. Furthermore it must be resistant to the MOV attack [16]. The same has to be true for $E(\mathbb{F}_{p^d})$. Methods to check a curve for these properties can be found in [7]. The Mordell-Weil group is subject to the further requirement that the function ϕ which is used for the point multiplication be injective. For extension degree 5 we require the greatest common divisors of f, g, $gp+tf$ and $gtp+pf+ft^2$ to be pairwise small as described in Proposition 3. For degree 3 the $\gcd(p-1, t-1)$ has to be small.

For cryptographic purposes we are calculating in a cyclic subgroup of large order of the Mordell-Weil group. The Mordell-Weil group has to be chosen in such a way that there exists such a subgroup, i.e., that $\#\mathcal{A}(\mathbb{F}_{p^d})$ has a large prime factor.

5.3 Elliptic Curve Point Representation

The points on an elliptic curve can be represented in affine and projective coordinates. When using affine coordinates we write a finite point on an elliptic curve $E(\mathbb{F}_q)$ as $\mathcal{P} = (x, y)$ with $x, y \in \mathbb{F}_q$. We can represent the same point in projective coordinates. There are multiple types of projective coordinates. In standard projective coordinates the projective point $(X, Y, Z), Z \neq 0$, corresponds to the affine point $(X/Z, Y/Z)$. We will instead use Jacobian projective coordinates where the projective point $(X, Y, Z), Z \neq 0$, corresponds to the affine point $(X/Z^2, Y/Z^3)$. The projective coordinates of a point are not unique because $(X, Y, Z) = (\lambda^2 X, \lambda^3 Y, \lambda Z)$ for every nonzero $\lambda \in \mathbb{F}_q$, i.e., each "point" in projective 2-space represents a line in affine 3-space. The point at infinity in the affine space is represented by the Z-plane, i.e., the projective coordinates of the point at infinity $\mathcal{O}$ are $(\lambda^2, \lambda^3, 0)$, where $\lambda \neq 0$. The conversion from affine to projective coordinates can easily be done by $(x, y) \mapsto (X, Y, Z)$ where

$$X = x,\ Y = y,\ Z = 1 \text{ for finite points}$$

Using projective point coordinates for point addition and doubling saves divisions at the cost of more multiplications in $\mathbb{F}_q$. This may speed-up the system when inversions in $\mathbb{F}_q$ are expensive. For an introduction to algebraic geometry, see [3].

5.4 Basic Functions

5.4.1 The Frobenius Map

The Frobenius map π acts on a point $\mathcal{P} = (x, y) \in E(\mathbb{F}_{p^3})$ in such a way that $\pi(\mathcal{P}) = (x^p, y^p)$. First we consider the case $d = 3$. In polynomial basis representation

x can be represented as $a_2X^2 + a_1X + a_0$, with $a_i \in \mathbb{F}_p$. Let $f(X)$ be the field polynomial with $f(X) = X^3 - \mu$ as described in Subsection 5.1.1. Then $x^p = (a_2X^2 + a_1X + a_0)^p \equiv a_2X^{2p} + a_1X^p + a_0 \bmod f(X)$ [15]. We get

$$x^p = \begin{cases} a_2X^8 + a_1X^4 + a_0 \equiv a_2\mu^2X^2 + a_1\mu X + a_0 \bmod f(X) & \text{if } p \equiv 4 \bmod 9 \\ a_2X^5 + a_1X^7 + a_0 \equiv a_2\mu X^2 + a_1\mu^2 X + a_0 \bmod f(X) & \text{if } p \equiv 7 \bmod 9 \end{cases}$$

Note that $X^3 = \mu$ and $X^9 = \mu^3 = 1$, i.e., X is a 9-th root of unity and 9 divides the group order $p^3 - 1$. The Frobenius map for a y coordinate $y = b_2X^2 + b_1X + b_0$ is exactly the same. Therefore we can obtain the "Frobenius of a point $\mathcal{P}$" by two field multiplications for each coordinate of $\mathcal{P}$.

If we choose extension degree 5 for the Mordell-Weil group, the Frobenius map can be computed in the same manner. Since we chose $\mu = X^5$ and $\mu^5 = X^{25} = 1$, we achieve

$$x^p = \begin{cases} a_4\mu^4X^4 + a_3\mu^3X^3 + a_2\mu^2X^2 + a_1\mu X + a_0 & \text{if } p \equiv 6 \bmod 25 \\ a_4\mu^3X^4 + a_3\mu X^3 + a_2\mu^4X^2 + a_1\mu^2X + a_0 & \text{if } p \equiv 11 \bmod 25 \\ a_4\mu^2X^4 + a_3\mu^4X^3 + a_2\mu X^2 + a_1\mu^3X + a_0 & \text{if } p \equiv 16 \bmod 25 \\ a_4\mu X^4 + a_3\mu^2X^3 + a_2\mu^3X^2 + a_1\mu^4X + a_0 & \text{if } p \equiv 21 \bmod 25 \end{cases}$$

As before X is a 25-th root of unity and 25 divides the group order $p^5 - 1$.

5.4.2 Point Addition

Point addition and doubling are crucial for most point multiplication methods. Algorithm 5 performs a point addition of two distinct points that are given in projective coordinates. It is the projective version of the affine point addition as described in Section 2.2.1.

Algorithm 5 Projective point addition [7]

INPUT: An elliptic curve point $\mathcal{P}_0 = (X_0, Y_0, Z_0) \in E(\mathbb{F}_p)$ and $\mathcal{P}_1 = (X_1, Y_1, Z_1) \in E(\mathbb{F}_p)$ with $\mathcal{P}_0 \neq \mathcal{P}_1, \mathcal{P}_0, \mathcal{P}_1 \neq \mathcal{O}$. The curve $E(\mathbb{F}_p)$ is defined by $y^2 = x^3 + ax + b \bmod p$.

OUTPUT: The elliptic curve point $\mathcal{P}_0 + \mathcal{P}_1 = (X_2, Y_2, Z_2) \in E(\mathbb{F}_p)$

1: $U_0 \leftarrow X_0 Z_1^2$
2: $S_0 \leftarrow Y_0 Z_1^3$
3: $U_1 \leftarrow X_1 Z_0^2$
4: $S_1 \leftarrow Y_1 Z_0^3$
5: $W \leftarrow U_0 - U_1$
6: $R \leftarrow S_0 - S_1$
7: $T \leftarrow U_0 + U_1$
8: $M \leftarrow S_0 + S_1$
9: $Z_2 \leftarrow Z_0 Z_1 W$
10: $X_2 \leftarrow R^2 - TW^2$
11: $V \leftarrow TW^2 - 2X_2$
12: $2Y_2 \leftarrow VR - MW^3$
13: **Return** (X_2, Y_2, Z_2)

This algorithm requires 16 field multiplications in the underlying field and 7 temporary variables [7]. To subtract two elliptic points we use the simple point inversion:

$$\mathcal{P}_2 = \mathcal{P}_0 - \mathcal{P}_1 = \mathcal{P}_0 + (-\mathcal{P}_1)$$

where

$$-\mathcal{P} = -(X, Y, Z) = (X, -Y, Z)$$

5.4.3 Point Doubling

Given $\mathcal{P}$, the point doubling function computes the elliptic curve point $2\mathcal{P}$. This is equivalent to adding a point to itself. Algorithm 6 requires 10 field multiplications in the underlying field and 5 temporary variables [7]. The algorithm is the projective version of the affine point doubling as described in Section 2.2.1.

Algorithm 6 Projective point doubling [7]

INPUT: An elliptic curve point $\mathcal{P} = (X_1, Y_1, Z_1) \in E(\mathbb{F}_p), \mathcal{P} \neq \mathcal{O}$. The curve $E(\mathbb{F}_p)$ is defined by $y^2 = x^3 + ax + b \bmod p$.
OUTPUT: The elliptic curve point $2\mathcal{P} = (X_2, Y_2, Z_2) \in E(\mathbb{F}_p)$.

1: $M \leftarrow 3X_1^2 + aZ_1^4$
2: $Z_2 \leftarrow 2Y_1Z_1$
3: $S \leftarrow 4X_1Y_1^2$
4: $X_2 \leftarrow M^2 - 2S$
5: $T \leftarrow 8Y_1^4$
6: $Y_2 \leftarrow M(S - X_2) - T$
7: **Return** (X_2, Y_2, Z_2)

5.5 Point Multiplication

5.5.1 Basic Point Multiplication Methods

There are two basic multiplication methods: right-to-left and left-to-right multiplication. Algorithm 7 and 8 describe these methods for base 2 representation. The algorithms can easily be generalized for arbitrary base m where m is a power of 2. Note that step 8 in Algorithm 7 need not be executed in the last iteration.

Algorithm 7 Right-to-left binary multiplication [17]

INPUT: An integer k and an elliptic point $\mathcal{P} \in E(\mathbb{F}_p)$.
OUTPUT: The elliptic curve point $k \cdot \mathcal{P} \in E(\mathbb{F}_p)$.

1: Let $(k_{l-1}k_{l-2} \ldots k_1k_0)_2$ be the binary representation of k where the most significant bit k_{l-1} equals 1.
2: $\mathcal{R} \leftarrow \mathcal{O}$
3: $\mathcal{S} \leftarrow \mathcal{P}$
4: **for** $i = 0$ to $l - 1$ **do**
5: **if** $k_i = 1$ **then**
6: $\mathcal{R} \leftarrow \mathcal{R} + \mathcal{S}$
7: **end if**
8: $\mathcal{S} \leftarrow 2\mathcal{S}$
9: **end for**
10: **Return** $\mathcal{R}$

On average both algorithms need l doublings and $1/2$ l additions where $l = \log(p)$. However, in Algorithm 8, Step 6 is always done for $\mathcal{P}$. This can give a

Algorithm 8 Left-to-right binary multiplication [17]

INPUT: An integer k and an elliptic point $\mathcal{P} \in E(\mathbb{F}_p)$.
OUTPUT: The elliptic curve point $k \cdot \mathcal{P} \in E(\mathbb{F}_p)$.

1: Let $(k_{l-1}k_{l-2}\ldots k_1k_0)_2$ be the binary representation of k where the most significant bit k_{l-1} equals 1.
2: $\mathcal{R} \leftarrow \mathcal{O}$
3: **for** $i = l-1$ down to 0 **do**
4: $\quad \mathcal{R} \leftarrow 2\mathcal{R}$
5: $\quad$ **if** $k_i = 1$ **then**
6: $\quad\quad \mathcal{R} \leftarrow \mathcal{R} + \mathcal{P}$
7: $\quad$ **end if**
8: **end for**
9: **Return** $\mathcal{R}$

speed-up if $\mathcal{P}$ is known.

5.5.2 Addition-Subtraction Method

The addition-subtraction method is recommended for point multiplication in the IEEE P1363 standard [7]. The idea of the addition-subtraction method is to write the scalar k in such a form that the number of non-zero entries is small. One popular approach is to write k as $k = \sum k_i 2^i$ with $k_i \in \{-1, 0, 1\}$, such that no two consecutive coefficients are non-zero. This is called the nonadjacent form (NAF) of k and is unique for every positive integer k. Moreover, the NAF of k has the smallest number of nonzero coefficients of any signed binary expansion of k [5]. For example, the NAF of $k = 156 = (10011100)_2$ is $(10100\text{–}100)_{NAF}$. On average only 1/3 of the coefficients are nonzero. Thus, Algorithm 9 needs on average l doublings and $l/3$ additions, where $l = \log(p)$, to perform a point multiplication.

5.5.3 Addition-Subtraction Method for the Mordell-Weil Group

The new point multiplication for the Mordell-Weil group was described in Section 3.2. We attempt to speed up this version by using the addition-subtraction

Algorithm 9 Addition-Subtraction method [7]

INPUT: An integer k and an elliptic curve point $\mathcal{P} \in E(\mathbb{F}_p)$.
OUTPUT: The elliptic curve point $k \cdot \mathcal{P} \in E(\mathbb{F}_p)$.

if $k = 0$ **then**
  Return $\mathcal{O}$
end if
if $k < 0$ **then**
  $\mathcal{S} \leftarrow -\mathcal{P}$ and $n \leftarrow -k$
else
  $\mathcal{S} \leftarrow \mathcal{P}$ and $n \leftarrow k$
end if
Let $(h_{l-1}h_{l-2}\dots h_1h_0)_2$ be the binary representation of $3n$, where the most significant bit h_{l-1} is 1.
Let $(n_{l-1}n_{l-2}\dots n_1n_0)_2$ be the binary representation of n.
$\mathcal{R} \leftarrow \mathcal{S}$
for $i = l - 2$ down to 1 **do**
  $\mathcal{R} \leftarrow 2\mathcal{R}$
  if $h_i = 1$ and $n_i = 0$ **then**
    $\mathcal{R} \leftarrow \mathcal{R} + \mathcal{S}$
  end if
  if $h_i = 0$ and $n_i = 1$ **then**
    $\mathcal{R} \leftarrow \mathcal{R} - \mathcal{S}$
  end if
end for
Return $\mathcal{R}$

method as described in Section 5.5.2. Algorithm 10 describes how to compute $(\lambda_0, \ldots, \lambda_{d-2}) \cdot \mathcal{P}$, where this product is defined as in (4.1). The point addition-subtraction step is applied for each λ_i (steps 8–13). However, the point doubling step is only done once for all $\lambda_i's$ (step 18). The other point doublings are obtained through the Frobenius map (step 15), which is computationally far more efficient.

Algorithm 10 Addition-Subtraction method for the Mordell-Weil group

INPUT: An elliptic curve point $\mathcal{P} = (X_p, Y_p, Z_p) \in \mathcal{A}(\mathbb{F}_{p^d})$ and a tuple $(\lambda_0, \ldots, \lambda_{d-2})$ of non-negative integers.
OUTPUT: The elliptic curve point $(\lambda_0, \ldots, \lambda_{d-2}) \cdot \mathcal{P} \in \mathcal{A}(\mathbb{F}_{p^d})$.

1: Let $(h_{i,l-1}h_{i,l-2}\ldots h_{i,1}h_{i,0})_2$ be the binary representation of $3\lambda_i$.
2: Let $(\lambda_{i,l-1}\lambda_{i,l-2}\ldots \lambda_{i,1}\lambda_{i,0})_2$ be the binary representation of λ_i.
3: $\mathcal{S} \leftarrow \mathcal{P}$
4: $\mathcal{R} \leftarrow \mathcal{O}$
5: **for** $i = 1$ to $l - 1$ **do**
6: $\quad \mathcal{T} \leftarrow \mathcal{S}$
7: $\quad$ **for** $j = 0$ to $d - 2$ **do**
8: $\quad\quad$ **if** $h_{j,i} = 1$ and $\lambda_{j,i} = 0$ **then**
9: $\quad\quad\quad \mathcal{R} \leftarrow \mathcal{R} + \mathcal{T}$
10: $\quad\quad$ **end if**
11: $\quad\quad$ **if** $h_{j,i} = 0$ and $\lambda_{j,i} = 1$ **then**
12: $\quad\quad\quad \mathcal{R} \leftarrow \mathcal{R} - \mathcal{T}$
13: $\quad\quad$ **end if**
14: $\quad\quad$ **if** $j \neq d - 1$ **then**
15: $\quad\quad\quad \mathcal{T} \leftarrow \pi(\mathcal{T})$
16: $\quad\quad$ **end if**
17: $\quad$ **end for**
18: $\quad \mathcal{S} = 2\mathcal{S}$
19: **end for**
20: **Return** $\mathcal{R}$

On average the algorithm needs l point doublings and $\frac{1}{3}(d-1)l$ additions where $l = \log(p)$. For $d = 3$ this is only a small improvement compared to Algorithm 3. For higher extension degrees we need less point doublings and continue to do arithmetic over the smaller field $\mathbb{F}_p$. Let us compare the cases $d = 3$ and $d = 5$. For $d = 3$ let $p \approx 2^{80}$ and for $d = 5$ let $p' \approx 2^{40}$. A point multiplication in $\mathcal{A}(\mathbb{F}_{p^3})$ requires

80 point doublings in $E(\mathbb{F}_{p^3})$ with $p^3 \approx 2^{240}$. One point multiplication in $\mathcal{A}(\mathbb{F}_{p'^5})$ requires only 40 point doublings in $E(\mathbb{F}_{p'^5})$ with $p'^5 \approx 2^{200}$. The number of point additions stays the same.

5.5.4 Window Methods

An overview of the available window methods for elliptic curve cryptosystems is given in [2]. To compute $k \cdot \mathcal{P}$ window methods process more than one bit of the scalar k per iteration, thus saving additions. However, they require precomputations. Algorithm 11 presents one of the fastest point multiplication methods for general elliptic curves, the sliding window method. Instead of a fixed window size the window is variable.

Using sliding windows has an effect equivalent to using fixed windows one bit larger, but without increasing the precomputation cost [2]. Therefore on average we need $2^{w-1}-1$ additions and 1 doubling for the precomputation, l doublings and $\frac{l}{w+1}$ additions where $l = \log(p)$. The sliding window method cannot easily be applied to the Mordell-Weil group. However, we can use a combination of a window method and simultaneous multiple exponentiation [17]. The core idea of Algorithm 12 is to step through all λ_i's simultaneously and hence reduce the number of point additions. Note that the precomputations require only $2^{w-1}-1$ additions using the Frobenius map.

Using this method, a scalar multiplication can be done on average using $2^{d-2}-1$ point additions for the precomputation, and $l - l/2^{d-1}$ additions and l doublings where $l = \log(p)$. For degree $d = 5$ this is slightly faster than the addition-subtraction method for the Mordell-Weil group, but requires more memory storage.

Algorithm 11 Sliding window method [2]

INPUT: A window width w, an integer k and an elliptic curve point $\mathcal{P} \in E(\mathbb{F}_p)$.
OUTPUT: The elliptic curve point $k \cdot \mathcal{P} \in E(\mathbb{F}_p)$.

Let $(k_{l-1}k_{l-2}\dots k_1k_0)_2$ be the binary representation of k where the most significant bit k_{l-1} is 1.
$\mathcal{P}_1 \leftarrow \mathcal{P}$
$\mathcal{P}_2 \leftarrow 2\mathcal{P}$
for $i = 1$ to $2^{w-1} - 1$ **do**
  $\mathcal{P}_{2i+1} \leftarrow \mathcal{P}_{2i-1} + \mathcal{P}_2$
end for
$j \leftarrow l - 1$
$\mathcal{Q} \leftarrow \mathcal{O}$
while $j \geq 0$ **do**
  if $k_j = 0$ **then**
    $\mathcal{Q} \leftarrow 2\mathcal{Q}$
    $j \leftarrow j - 1$
  else
    Let t be the least integer such that $j - t + 1 \leq w$ and $k_t = 1$
    $h_j \leftarrow (k_j k_{j-1} \dots k_t)_2$
    $\mathcal{Q} \leftarrow 2^{j-t+1} \cdot \mathcal{Q} + \mathcal{P}_{h_j}$
    $j \leftarrow t - 1$
  end if
end while
Return $\mathcal{Q}$

Algorithm 12 Window method for the Mordell-Weil group

INPUT: An elliptic curve point $\mathcal{P} = (X_p, Y_p, Z_p) \in \mathcal{A}(\mathbb{F}_{p^d})$ and a tuple $(\lambda_0, \dots, \lambda_{d-2})$ of non-negative integers. The window size is $w = d - 1$.
OUTPUT: The elliptic curve point $(\lambda_0, \dots, \lambda_{d-2}) \cdot \mathcal{P} \in \mathcal{A}(\mathbb{F}_{p^d})$.

Let $(\lambda_{i,l-1}\lambda_{i,l-2} \dots \lambda_{i,1}\lambda_{i,0})_2$ be the binary representation of λ_i.
Precompute $\mathcal{P}_j \leftarrow \sum j_i \pi^i \mathcal{P}$ for $0 < j < 2^w$, where $(j_{w-1}, j_{w-2}, \dots, j_0)_2$ is the binary representation of j.
$\mathcal{Q} \leftarrow \mathcal{O}$
for $j = l - 1$ down to 0 **do**
  $\mathcal{Q} \leftarrow 2\mathcal{Q}$
  $i \leftarrow \lambda_{0,j} + 2\ \lambda_{1,j} + \dots + 2^{w-1}\ \lambda_{w-1,j}$
  if $i \neq 0$ **then**
    $\mathcal{Q} \leftarrow \mathcal{Q} + \mathcal{P}_i$
  end if
end for
Return $\mathcal{Q}$

5.5.5 Precomputation of Points

In the previous algorithms we saved operations while calculating in the Mordell-Weil group by saving doubling steps. For example, let us compare $E(\mathbb{F}_{q^4})$ to $\mathcal{A}(\mathbb{F}_{q^5})$; Note that both groups have roughly the same cardinality. We need $\log(q^4) = 4\log(q)$ doublings in the first case, but only $\log(q)$ doublings in the second case.

The doublings are applied to a base point $\mathcal{P}$ that in some applications usually does not change. For example, when using the Diffie-Hellman key exchange protocol one can use a fixed base point $\mathcal{P}$ and calculate $k \cdot \mathcal{P}$ where k is the secret key to obtain the public key. To accelerate the operation, certain multiples of $\mathcal{P}$ can be precomputed. This is done as described in Algorithm 13 [6]. We define $[a_{w-1}, \ldots, a_0]\mathcal{P} := \sum_{i=0}^{w-1} a_i 2^{id} \cdot \mathcal{P}$ where $d = \lceil l/w \rceil$, l the bit-length of p and $a_i \in \mathbb{Z}_2$.

Algorithm 13 Point multiplication using precomputed points [6]

INPUT: An elliptic curve point $\mathcal{P} \in E(\mathbb{F}_p)$ and an integer k.
OUTPUT: The elliptic curve point $k \cdot \mathcal{P} \in E(\mathbb{F}_p)$.

1: Precompute $[a_{w-1}, \ldots, a_0]\mathcal{P}\ \forall (a_{w-1}, \ldots, a_0) \in \mathbb{Z}_2^w$.
2: Write $k = K^{w-1} \ldots K^0$, where each K^j is a bit string of length d. If necessary, pad k on the left with 0's. Let K_i^j denote the i-th bit of K^j.
3: $\mathcal{R} \leftarrow \mathcal{O}$.
4: **for** $i = d - 1$ down to 0 **do**
5: $\quad \mathcal{R} \leftarrow 2\mathcal{R}$.
6: $\quad \mathcal{R} \leftarrow \mathcal{R} + [K_i^{w-1}, \ldots, K_i^0]\mathcal{P}$.
7: **end for**
8: **Return** $\mathcal{R}$

This method requires $d = \lceil l/w \rceil$ point doublings and additions. To apply precomputed points to the Mordell-Weil group we use Algorithm 12. The precomputation step is done only once though. The window can have a size that is a multiple of $d - 1$. If $w = c\ (d - 1)$ is the window size a window covers the operand bits $(\lambda_{i,0}, \ldots, \lambda_{i,d-2}; \lambda_{i-1,0}, \ldots, \lambda_{i-1,d-2}; \ldots; \lambda_{i-c+1,0}, \ldots, \lambda_{i-c+1,d-2})$. The algorithmic complexity is similar to the complexity of Algorithm 13, but point arithmetic

is done over a larger underlying field. However, protocols do not only use fixed point scalar multiplication. For example, the second step in the Diffie-Hellman key exchange protocol requires the multiplication of the private key with a public key that is not fixed.

Chapter 6

Algorithmic Complexity

In this section we compare the algorithmic complexity of the different methods for point multiplication in general elliptic curve cryptosystems (ECC) and the Mordell-Weil group. We compare the curves in such a way that the level of security, i.e., the average number of steps required to solve the DLP are similar.

6.1 Basic Functions

The Frobenius map is crucial for the point multiplication in the Mordell-Weil group. As described in Section 5.4.1 the Frobenius of a point $\mathcal{P}$ can be calculated with few multiplications in the underlying prime field. Let $\mathcal{P} = (X, Y, Z)$ with $X, Y, Z \in \mathbb{F}_{p^d}$. As shown in Subsection 5.4.1 the Frobenius of $\mathcal{P}$ can be calculated cheaply with $d-1$ multiplications over $\mathbb{F}_p$ for each coordinate, i.e., with $3(d-1)$ multiplications for one point. Table 6.1 summarizes the results. Note that the field multiplications are done over the subfield $\mathbb{F}_p$ and not $\mathbb{F}_{p^d}$. One field multiplication over $\mathbb{F}_{p^3}$ needs roughly 9 multiplications over the subfield $\mathbb{F}_p$ while one field multiplication over $\mathbb{F}_{p'^5}$ needs around 25 multiplications over $\mathbb{F}_{p'}$. This shows that the cost to apply the Frobenius map can be neglected.

	$E(\mathbb{F}_{p^3})$	$E(\mathbb{F}_{p'^5})$
# mult.	6 over $\mathbb{F}_p$	12 over $\mathbb{F}_{p'}$

Table 6.1: Field multiplications to compute the Frobenius map.

	Point addition (Alg. 5)	Point doubling (Alg. 6)
# mult.	16	10

Table 6.2: Complexity of elementary point operations over $\mathbb{F}_{p^d}$.

The basic point operations for all point multiplication methods are point addition and point doubling. By considering the algorithms in the Sections 5.4.2 and 5.4.3 we count the number of multiplications in the underlying field $\mathbb{F}_{p^d}$ for one point addition and one point doubling, respective. To simplify the following calculations we assume that squaring and general multiplication are similar in complexity over $\mathbb{F}_{p^d}$. We need 16 field multiplications for one point addition and 10 field multiplications for one point doubling. Table 6.2 summarizes the results.

6.2 Point Multiplication in $E(\mathbb{F}_q)$

This section describes the complexity of the point multiplication methods as described in Section 5.5 for general elliptic curves $E(\mathbb{F}_{p^d})$. To keep it simple we express the complexity of the algorithms over p. Therefore we assume $q \approx p^d$ where q and p are prime. Table 6.3 is organized as follows: The columns describe the point multiplication methods. These are the addition-subtraction method and the sliding window method for general elliptic curves. The rows describe the number of operations needed. The first two rows describe the number of point additions and point doublings for one scalar multiplication. The next row counts the number of field multiplications over $\mathbb{F}_{p^d}$ for the point multiplication. Therefore the number of point

	Add.-Subt. (Alg. 9)	Sliding Window (Alg. 11)
# point add.	$d/3\ \log(p)$	$2^{w-1} - 1 + \frac{d}{w+1}\log(p)$
# point doubl.	$d\ \log(p)$	$d\log(p) + 1$
# field mult.	$15\frac{1}{3}d\ \log(p)$	$16 \cdot 2^{w-1} - 6 + (\frac{16}{w+1}d + 10d)\log(p)$
# elem. oper.	$15\frac{1}{3}\log^3(p)d^3$	$(16 \cdot 2^{w-1} - 6 + (\frac{16}{w+1}d + 10d)\log(p))d^2\log^2(p)$

Table 6.3: Complexity of point multiplication over $E(\mathbb{F}_q)$ where $q \approx p^d$.

additions is multiplied by 16, the number of point doublings by 10 and the result is accumulated. The last row approximates the number of elementary operations, i.e., the overall complexity. Therefore we assume that one field multiplication over $\mathbb{F}_{p^d}$ has square complexity, i.e., $(d\log(p))^2$.

The sliding window method is better than the addition-subtraction method. It could be improved furthermore by precomputing the points in affine coordinates and then using a mixed point addition method to add a projective and affine point. To keep the algorithmic complexity calculation simply we do not use mixed coordinates since the difference is small.

6.3 Point Multiplication in $\mathcal{A}(\mathbb{F}_{p^d})$

The results in this section are for arbitrary extension degrees of the Mordell-Weil group. Table 6.4 describes the complexity for one point multiplication in $\mathcal{A}(\mathbb{F}_{p^d})$. The remainder of this chapter analyzes the complexity for general elliptic curves and Mordell-Weil group with extension degree 2 and 3, and 4 and 5 respectively.

From now on we fix p, p' and q such that $p \approx \sqrt{q}$ and $p' \approx \sqrt[4]{q}$, with p, p' and q prime. To keep it simple we always express the complexity of the algorithms over p. As an example we chose $q \approx 2^{163}$, $p \approx 2^{82}$ and $p' \approx 2^{41}$. Specifically, we will compare the complexity of one point multiplication in $E(\mathbb{F}_q)$, $\mathcal{A}(\mathbb{F}_{p^3})$ and $\mathcal{A}(\mathbb{F}_{p'^5})$. Note that

	Addition-Subtraction (Alg. 10)	Window Method (Alg. 12)
# point add.	$1/3(d-1)\log(p)$	$\log(p)(1+1/2^{d-1})+2^{d-2}-1$
# point doubl.	$\log(p)$	$\log(p)$
# field mult.	$\frac{16}{3}d\log(p)+\frac{14}{3}\log(p)$	$26\log(p)+\frac{16\log(p)}{2^{d-1}}$ $+16\ 2^{d-2}-16$
# elem. oper.	$(\frac{16}{3}d\log(p)+\frac{14}{3}\log(p))d^2\log^2(p)$	$(26\log(p)+\frac{16\log(p)}{2^{d-1}}$ $+16\ 2^{d-2}-16)d^2\log^2(p)$

Table 6.4: Complexity of point multiplication over $\mathcal{A}(\mathbb{F}_{p^d})$.

	Addition-Subtraction (Alg. 9)	Sliding Window (Alg. 11)
# point add.	$2/3\ \log(p)$	$7+2/5log(p)$
# point doubl.	$2\ \log(p)$	$1+2\log(p)$
# field mult.	$30\frac{2}{3}\ \log(p)$	$122+26\frac{2}{5}\log(p)$
# elem. oper.	$122\frac{2}{3}\ \log^3(p)$	$(488+105\frac{3}{5}\log(p))\log^2(p)$

Table 6.5: Complexity of point multiplication over $E(\mathbb{F}_q)$ where $q \approx p^2$.

all these curves provide the same level of security which is often considered to be security equivalent to an RSA system with 1024-bit key lengths.

6.4 Point Multiplication in $E(\mathbb{F}_q)$, $\mathcal{A}(\mathbb{F}_{p^3})$ and $\mathcal{A}(\mathbb{F}_{p'^5})$

We first compare the algorithmic complexity of $E(\mathbb{F}_q)$ and $\mathcal{A}(\mathbb{F}_{p^3})$. Table 6.5 describes it for $E(\mathbb{F}_q)$. For our implementations we used the sliding window method with $w = 4$.

Table 6.6 shows the complexity for the Mordell-Weil group with extension degree 3. There are two basic methods for the Mordell-Weil group $\mathcal{A}(\mathbb{F}_{p^3})$. The first one is only applicable to extension degree 3 while the second one is always available.

Table 6.7 describes the complexity for $\mathcal{A}(\mathbb{F}_{p'^5})$ which is also security equivalent to the previous groups. One can see that the window method is slightly faster for our example of $p \approx 2^{82}$.

	Basic (Alg. 3)	Basic (Alg. 4)	Addition-Subtraction (Alg. 10)
# point add.	$3/4\log(p)$	$\log(p)$	$2/3\log(p)$
# point doubl.	$\log(p)$	$\log(p)$	$\log(p)$
# field mult.	$22\log(p)$	$26\log(p)$	$20\frac{2}{3}\log(p)$
# elem. oper.	$198\log^3(p)$	$234\log^3(p)$	$186\log^3(p)$

Table 6.6: Complexity of point multiplication over $\mathcal{A}(\mathbb{F}_{p^3})$.

	Addition-Subtraction (Alg. 10)	Window Method (Alg. 12)
# point add.	$4/3\log(p')$ $\approx 2/3\log(p)$	$\log(p')(1-1/16)+7$ $\approx 1/2\log(p)(1-1/16)+7$
# point doubl.	$\log(p') \approx 1/2\log(p)$	$\log(p') \approx 1/2\log(p)$
# field mult.	$31\frac{1}{3}\log(p')$ $\approx 15\frac{2}{3}\log(p)$	$25\log(p')+112$ $\approx 12\frac{1}{2}\log(p)+112$
# elem. oper.	$783\frac{1}{3}\log^3(p')$ $\approx 98\log^3(p)$	$(625\log(p')+2800)\log^2(p')$ $\approx (78\frac{1}{8}\log(p)+700)\log^2(p)$

Table 6.7: Complexity of point multiplication over $\mathcal{A}(\mathbb{F}_{p'^5})$ where $p'^5 \approx \sqrt{p}$.

6.5 Comparison between Elliptic Curves and the Mordell-Weil Group

The previous sections gave a detailed analysis of the complexity of one point multiplication in a general elliptic curve and the Mordell-Weil group. All groups we looked at have a similar private key length. By assuming $q \approx p^2 \approx p'^4$, where q, p and p' prime, the groups $E(\mathbb{F}_q)$, $\mathcal{A}(\mathbb{F}_{p^3})$ and $\mathcal{A}(\mathbb{F}_{p'^5})$ all have an effective private key length of $\log(q) \approx 2\log(p)$ binary bits. We saw that the sliding window method is the fastest method for elliptic curves, while for the Mordell-Weil group the addition-subtraction and window method are both efficient. Therefore the cost for one point multiplication is $(105\frac{3}{5}\log(p)+488)\log^2(p)$ elementary operations compared to $98\log^3(p)$ and $(78\frac{1}{8}\log(p)+700)\log^2(p)$, respective. Assuming $q \approx 2^{163}$, i.e., $p \approx 2^{82}$ the computation of a public key using the window method in the Mordell-

Weil group is computationally about 22% more efficient than in the elliptic curve case. In cases where the sliding window method is not applicable for ECC, e.g., due to memory constraints, and the addition-subtraction method is used for ECC point multiplication, the Mordell-Weil group is more than 20% more efficient.

A logical extension of the work presented here is to choose an extension degree of 7 for the Mordell-Weil group. In this case, we can choose p'' such that $p''^6 \approx 2^{163}$, i.e., $p'' \approx 2^{27}$. Note that for degree $d = 7$ the addition-subtraction method is more efficient than the window method. Using the complexity formula in Table 6.4 for the addition-subtraction method with these values of p'' and d, we obtain that one point multiplication needs around $77 \log^3(p)$ elementary operations. This would be an improvement of 31% compared to general elliptic curves and 37% if precomputation is not applicable. Furthermore a base field of this kind would be a very promising match for 32-bit hardware platforms, so that actual performance improvements might be significantly higher. However, one would have to prove that higher extension degrees yield appropriate groups which do not result in a loss of security, similar to the result shown in Proposition 3.

Chapter 7

Implementation Results

We implemented the Mordell-Weil group using the C++ library LiDIA [14]. The timing measurements confirm the results of the complexity analysis. Since LiDIA is not optimized for a special hardware or a particular choice of the underlying field the absolute time values are relatively slow though. We chose primes p, p' and q in such a way, that we get a private key size of around 163 bits, i.e., we chose $q \approx 2^{163}$, $p \approx \sqrt{q} \approx 2^{82}$ and $p' \approx \sqrt[4]{q} \approx 2^{41}$. This is often considered to be roughly security equivalent to an RSA system with 1024-bit key length.

7.1 Implementation Parameters

First we selected an elliptic curve and appropriate curve and field parameters. We chose an example used in [21] and [24]:

$$E : Y^2 = X^3 - 90 \cdot 27^{-1} X + 56 \cdot 27^{-1}$$

We choose $q \approx 2^{163}$

$$q = 10000000000000000000006018000000000000000009054201$$

and $p \approx \sqrt{q} \approx 2^{82}$

$$p = 10000000000000000000003009$$

Using LiDIA we obtain the number of points on the elliptic curve as

$$\#E(\mathbb{F}_p) = 999999999998639223372928$$

and the trace

$$t = 1 + p - \#E(\mathbb{F}_p) = 1360776630082$$

Therefore we obtain the Weil number as described in (3.5)

$$\alpha = 680388315041 + 2183697751608\sqrt{-2}$$

and by (3.6) the group order of the Mordell-Weil group

$$\#\mathcal{A}(\mathbb{F}_{p^3}) = \frac{(1-\alpha^3)(1-\bar{\alpha}^3)}{(1-\alpha)(1-\bar{\alpha})}$$

$$= 163 \cdot 6134969325154209065420919745503867550506867050 59$$

By (3.7) we obtain

$$C = 61349693251533742331306$$

and get the number of different endomorphisms in $End(\langle \mathcal{P} \rangle)$ for a point $\mathcal{P} \in \mathcal{A}(\mathbb{F}_{p^3})$ of maximal prime order as $C^2 \approx 2^{151}$.

As an example for the Mordell-Weil group of extension degree 5 we used the

same curve E and chose $\sqrt{p'} \approx \sqrt[4]{q} \approx 2^{41}$

$$p' = 3162280535273$$

According to LiDIA the number of points on the elliptic curve is

$$\#E(\mathbb{F}_{p'}) = 3162280535436$$

and the trace equals

$$t = 1 + p' - \#E(\mathbb{F}_{p'}) = -162$$

$|t|$ is small and the pairwise greatest common divisors of f, g, $gp' + tf$ and $gtp' + p'f + ft^2$ is 2. We obtain the Weil number by using (3.5)

$$\alpha = -81 + 1257434\sqrt{-2}$$

and the number of points of the Mordell-Weil group

$$\#\mathcal{A}(\mathbb{F}_{p'^5}) = \frac{(1-\alpha^5)(1-\bar{\alpha}^5)}{(1-\alpha)(1-\bar{\alpha})}$$

$$= 79811 \cdot 1252964674925836672418853211529856042119867851$$

We found these parameters quickly. A good curve should have a subgroup of large order though since the size of C depends thereon significantly, as stated in Propositions 1 and 3. Therefore much effort should be spent to find good curves. Since this is a problem of elliptic curve generation and not intrinsically related to the Mordell-Weil group, this issue is not further discussed here.

	time
$\mathbb{F}_q$	0.095
$\mathbb{F}_{p^3}$	0.131
$\mathbb{F}_{p'^5}$	0.159

Table 7.1: Time for a field multiplication in msec.

7.2 Environment

The implementation was done in C++ using the LiDIA library [14]. We implemented the basic point operations based on the P1363 standard. The time measurements were done on a Sun Ultra 5 Workstation with a 333 MHz CPU and 128 MB memory.

7.3 Basic Functions

All elliptic curve point multiplications finally depend on the field multiplication and the Frobenius map. Table 7.1 shows the result of the time measurements for one field multiplication. The left column denotes the field, the right one the time for one multiplication in this field in milliseconds. The field multiplication was provided by the LiDIA library. Note that a multiplication in a field with high extension degree is slower than one with small extension degree but same effective operand length. For example, a multiplication in $\mathbb{F}_{p'^4}$ is more than three times slower than in $\mathbb{F}_q$, even though both fields have a roughly similar order.

Table 7.2 displays the time for one application of the Frobenius map. It should be around half the time that is needed for one field multiplication. The LiDIA library does not allow efficient handling of the involved polynomial coefficients.

Table 7.3 shows the time for one point addition and doubling, respectively. Again we note that the time for one point operation is affected by the underlying field, i.e.,

	time
$\mathcal{A}(\mathbb{F}_{p^3})$	0.273
$\mathcal{A}(\mathbb{F}_{p'^5})$	0.464

Table 7.2: Time for the Frobenius map in msec.

	Point addition (Alg. 5)	Point doubling (Alg. 6)
$E(\mathbb{F}_q)$	1.68	1.36
$E(\mathbb{F}_{p^3})$	2.06	1.55
$E(\mathbb{F}_{p'^5})$	2.70	2.14

Table 7.3: Time for point addition and doubling in msec.

that higher extension degrees are slower in LiDIA. We want to stress that this is a property of LiDIA and not necessarily a general characteristic of extension fields.

7.4 Point Multiplication in $E(\mathbb{F}_q)$, $\mathcal{A}(\mathbb{F}_{p^3})$ and $\mathcal{A}(\mathbb{F}_{p'^5})$

The results in Table 7.4 validate our computations about the complexity in the previous chapter. One sees that the running time of a point multiplication in the general elliptic curve is better than in the Mordell-Weil group $\mathcal{A}(\mathbb{F}_{p^3})$. However, one point multiplication in the Mordell-Weil group of degree 5, $\mathcal{A}(\mathbb{F}_{q^5})$, has a better runtime than in the general elliptic curve group.

	Basic (Alg. 3/Alg. 4)	Addition-Subtraction (Alg. 9/10)	Sliding Window/ Window Method (Alg. 11 / Alg. 12)
$E(\mathbb{F}_q)$	-	363.3	313.8
$\mathcal{A}(\mathbb{F}_{p^3})$	351.7	321.3	-
$\mathcal{A}(\mathbb{F}_{q^5})$	372.6	299.4	263.5

Table 7.4: Time for one point multiplication in msec.

Chapter 8

Summary and Suggestions for Future Work

In this thesis, a new algebraic group for cryptographic purposes was investigated and implemented in software. The original idea by Frey and Naumann was restricted to extension degree 3 which does not bear computational advantages over conventional ECC. We extended Mordell-Weil cryptosystems to higher extension degrees which yields a point multiplication that is more efficient than in general elliptic curve groups. For the case of $d = 5$, we were able to prove that the Mordell-Weil group has a sufficient cardinality under certain conditions. We provided new point multiplication algorithms for the Mordell-Weil group and a detailed complexity analysis.

The main disadvantage of the Mordell-Weil group is that the length of the public keys are larger, that point precomputation is not supported efficiently, and that curve generation is even more constrained than for general elliptic curves. There are several open research problems related to the Mordell-Weil group:

1. As described the usage of larger extension degrees could speed-up scalar multiplication considerably. This would be particularly of interest for implemen-

tations on embedded systems where each operand would fit into one register. For 16-bit platforms an extension degree of 11 should be investigated. It has to be proven that this does not result in a lost of security similar as it was done in Proposition 3.

2. Proposition 3 might be generalized. It would be extremely valuable to find a proof for a generalization of this proposition for arbitrary extension degree. For practical purposes it might be enough to repeat the proof for degree 7 and 11.

3. We noted that popular signature schemes are not applicable using the Mordell-Weil group. Hence it is important to find an efficient signature scheme.

4. Further research could also be done about curve generation. Generation of elliptic curve is difficult and slow. The generation of Mordell-Weil groups is even more constrained. Therefore a set of appropriate curves should be recommended.

5. Our implementation using the LiDIA library is very slow. A more efficient implementation of the field arithmetic should be done, e.g., using Optimal Extension Fields [1].

Bibliography

[1] D. Bailey and C. Paar. Optimal Extension Fields for Fast Arithmetic in Public Key Algorithms. *Advances in Cryptology – Crypto '98 (LNCS 1462)*, 472–485, 1998.

[2] I. Blake, G. Seroussi and N. Smart. *Elliptic Curves in Cryptography*. Cambridge University Press, Cambridge, 1999.

[3] W. Fulton. *Algebraic Curves – An Introduction to Algebraic Geometry*. W.A. Benjamin, Inc., New York, 1969.

[4] P. Gaudry, F. Hess, and N. P. Smart. Constructive and Destructive Facets of Weil Descent on Elliptic Curves. Technical Report CSTR-00-016, Department of Computer Science, University of Bristol, October 2000.

[5] D. M. Gordon. A Survey of Fast Exponentiation Methods. *Journal of Algorithms* 27, 129–146, 1988.

[6] D. Hankerson, J. L. Hernandez and A. Menezes. Software Implementation of Elliptic Curve Cryptography Over Binary Fields. Cryptographic Hardware and Embedded Systems, CHES 2000, August 17–18, 2000, Worcester MA, USA.

[7] IEEE P1363, *Standard Specifications for Public-Key Cryptography*, 2000.

[8] D. Johnson and A. Menezes. The elliptic curve digital signature algorithm (ECDSA). Technical report CORR 99-34, Dept. of C&O, University of Waterloo, 1999.

[9] N. Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation* 48, 203–209, 1987.

[10] N. Koblitz. *A Course in Number Theory and Cryptography, Second Edition*. Springer-Verlag, New York, 1994.

[11] N. Koblitz. CM-curves with good cryptographic properties. *Advances in Cryptology – Crypto '91 (LNCS 576)*, 279–287, 1992.

[12] N. Koblitz. *Introduction to Elliptic Curves and Modular Forms*. Springer-Verlag, New York, 1984.

[13] D.E. Knuth. *Seminumerical Algorithms*. Addision-Wesley, 1981.

[14] LiDIA – A C++ Library For Computational Number Theory. J. Buchmann, Technische Universität Darmstadt, available from http://www.informatik.tu-darmstadt.de/TI/LiDIA/Welcome.html.

[15] R. Lidl and H. Niederreiter. *Introduction to Finite Fields and their Applications, Revised Edition*. Cambridge Unversity Press, Cambridge, United Kingdom, 1994.

[16] A.J. Menezes, T. Okamoto and S.A. Vanstone. Reducing Ellipitic Curve logarithms to a Finite Field. *IEEE Trans. Info. Theory*, 39, 1639–1646, 1993.

[17] A.J. Menezes, P.C. van Oorschot and S.A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[18] V.S. Miller. Use of elliptic curves in cryptography. *Advances in Cryptography – Crypto '85 (LNCS 218)*, 417–426, 1986.

[19] National Institute of Standards and Technology. *Digital Signature Standard*, FIPS Publication 186-2, February 2000.

[20] National Institute of Standards and Technology. *Recommended Elliptic Curves for Government Use*, May 1999, available from http://csrc.nist.gov/encryption.

[21] N. Naumann. Weil-Restriktionen abelscher Varietäten. Diplomarbeit (MS Thesis), Department of Mathematics, Universität GHS Essen, Germany, August 1999.

[22] B. Schneier. *Applied Cryptography*. John Wiley & Sons, 1996.

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

[24] J.H. Silverman. *Advanced Topics in the Arithmetic of Elliptic Curves*. Springer-Verlag, New York, 1994.

[25] J.H. Silverman and J. Tate. *Rational Points on Elliptic Curves*. Springer-Verlag, New York, 1992.

[26] N. Smart. The Discrete Logarithm Problem on Elliptic Curves of Trace one. *Journal of Cryptology*, 12(3),193–196, October 1999.