Attacking the Elliptic Curve Discrete Logarithm Problem

by

Matthew Musson

Thesis submitted in partial fulfillment of the requirements of the degree

of Master of Science (Mathematics and Statistics)

Acadia University

Spring Convocation, 2006

This thesis by Matthew Musson was defended successfully in an oral examination on March 28^{th}, 2006.

The examination committee for this thesis was:

Dr. Sajid Hussain, Chair

Dr. Rob Gallant, External Reader

Dr. Franklin Mendivil, Internal Reader

Dr. Jeff Hooper, Supervisor

Dr. Paul Cabilio, Head

This thesis is accepted in its present form by the Division of Research and Graduate Studies as satisfying thesis requirements for the degree Master of Science(Mathematics and Statistics).

Table of Contents

List of Tables

List of Algorithms

Abstract

The purpose of this thesis is an in-depth examination of the Elliptic Curve Discrete Logarithm Problem(ECDLP) including up-to-date techniques in attacking cryptosystems dependent on the ECDLP. The thesis is presented as a how-to guide and included are programs written in `Pari/GP` for various attacks. We then use the knowledge of these attacks in an attempt to generate cryptographically strong elliptic curves.

Abbreviations and Symbols

Acknowledgments

I would like to thank my supervisor Dr. Jeff Hooper for providing useful comments throughout this entire process, and for his hard work in helping me understand necessary background material for this document. I would also like to thank the Department of Mathematics and Statistics at Acadia University for supporting me over these two years and allowing me to do my own thing, all the readers of this thesis who have provided me with their useful feedback, and all the others who have supported me throughout this entire process including family and friends.

1 Introduction

Introduced to cryptography in 1985, elliptic curves are quickly being adapted for cryptographic purposes. Elliptic curve cryptography is quickly becoming a leader in the industry, and is challenging other cryptosystems such as RSA and DSA to become the industrial standard; this is due to an increase in speed during implementation, the use of less memory, and smaller key sizes. Another advantage of such a cryptosystem lies in the difficulty of solving the Elliptic Curve Discrete Log Problem (ECDLP). If an elliptic curve is chosen with some care, the ECDLP is believed to be infeasible, even with today's computational power. On the other hand, this obstacle has not deterred those in their attempts to crack elliptic curve cryptosystems. A multitude of attacks have been developed, tested, and analyzed when attacking the ECDLP. For the most part the ECDLP has withstood all attempts; however, in some special cases the problem is actually quite easy. It is simply these cases that must be avoided when building such a cryptosystem.

Using elliptic curves presents a great advantage in a few areas. For instance, compared to RSA cryptosystems, elliptic curve based systems require less memory; for example, a key size of 4096 bits for RSA gives the same level of security as 313 bits in an elliptic curve system [5]. Also, using a PalmPilot, generating a 512-bit RSA

key takes around 3.4 minutes, while generating an equivalent 163-bit ECC-DSA key takes 0.597 seconds [87, 159]. Immediately we begin to see the advantages of using elliptic curves, especially on a small hand-held devices with little computing power. It is clear that this now gives us the advantage of setting up schemes that require smaller chip sizes, use less memory, require less resources to run, require less power consumption, etc; and can be placed in small electronic devices, such as smart cards and cell phones.

Many elliptic curve cryptosystems take advantage of what is known as the ECDLP. Analogous to the Discrete Logarithm Problem (DLP) over a finite field $\mathbb{F}_p^\times$, the ECDLP is the following problem: given two points P and Q on an elliptic curve E defined over a field $\mathbb{F}_q$, where q is prime or a prime power, if $P = [m]Q$ for some $m \in \mathbb{Z}$, determine m. Schemes and protocols such as the Deffie-Hellman key exchange, Massey-Omura encryption, El-Gamal public key encryption and El-Gamal digital signatures and even the Elliptic Curve Digital Signature Algorithm(ECDSA), all use the fact that attempting to solve the ECDLP is a difficult, if not intractable, problem. In fact it is believed that the ECDLP is as difficult if not more so than solving the DLP over $\mathbb{F}_p$ [5].

As mentioned although the ECDLP is thought to be an intractable problem, it has not stopped people attempting to attack such a cryptosystem. Various attacks have been devised, tested and analyzed by many leading mathematicians over the years, in attempts to find weaknesses in elliptic curve cryptosystems. Some have been partially successful, while others have not. The ECDLP has been shown to be easily solved for the following situations:

1. If $\#E(\mathbb{F}_p) = p + 1$ (the supersingular case) then the ECDLP can be reduced to the DLP on the multiplicative group of the finite field with p^k elements. This is practical if k is not too large.

2. If $\#E(\mathbb{F}_p) = p$ (the anomalous case) then the ECDLP can be reduced to simple addition in $\mathbb{F}_p$, essentially by lifting the curve modulo p^2.

3. If $\#E(\mathbb{F}_p)$ is divisible by only small primes, then one can use the Pohlig-Hellman method which solves the problem in time $O(\sqrt{p'})$, where p' is the largest prime divisor of $E(\mathbb{F}_p)$.

In each of these three cases the underlying curve can easily be modified so as to thwart each attack. For example if we had a curve for which our point P had large prime order, that is $[m]P = \mathcal{O}$, for a large prime m, then the Pohlig-Hellman method becomes impractical.

The purpose of this thesis is to provide a detailed examination of the leading attacks against the ECDLP, and to use the knowledge of these attacks in an attempt to generate cryptographically strong elliptic curves.

2 Attacking the Discrete Logarithm Problem

Before diving immediately into the realm of elliptic curves, we first present a brief treatment of the Discrete Logarithm Problem and the various attacks available, so that:

1. the reader becomes familiar with the setup and attacks on cryptosystems.

2. the reader is familiar with these attacks for they will reappear, albeit briefly, when discussing Pairing attacks on elliptic curves. These attacks attempt to take the ECDLP and transform it to the DLP in an isomorphic group in an abelian variety of higher dimension.

3. the reader is able to draw analogies between the attacks in the DLP setting and the ECDLP setting.

4. the reader gains a better understanding of how certain attacks have failed to translate to the elliptic curve setting. For instance, as we shall see, the most powerful attack on the DLP, the Index Calculus fails to translate over to the setting of elliptic curves.

1 The Discrete Logarithm Problem

The security of many cryptosystems depends on the intractability of the discrete logarithm problem. For instance one of the more famous public key cryptosystems, El-Gamal encryption, relies heavily on the intractability of this problem. The following is referred to as the DLP or even sometimes as the Generalized DLP.

Definition 2.1 *Let G be a finite cyclic group of order n. Let α be a generator of G, and $\beta \in G$. Determine the unique integer x, $0 \leq x \leq n-1$ such that $\alpha^x = \beta$.*

In the specific setting we take $G = \mathbb{Z}_p$ and attempt to solve the congruence $\alpha^x \equiv \beta \mod p$. It is this setting that is commonly referred to as the DLP, but either setting will suffice for our purposes.

Attacks on the DLP can be divided into three main categories [62]:

1. algorithms that work in arbitrary groups, such as the exhaustive search and the Baby-Step Giant-Step algorithm,

2. algorithms that work in arbitrary groups with special conditions present in the group, like Pollard's λ-Method, and

3. algorithms that work only in specific groups, such as the Index Calculus.

1.1 Exhaustive Search

As its name suggests, this attack involves simply computing powers of α until the value of β is found. This attack is completely inefficient when dealing with concrete cryptographic situations.

1.2 Baby-Step, Giant-Step Algorithm

This attack uses a combination of computational power and memory storage to solve the DLP. Let G be a cyclic group with generator α. Suppose that α has order n and set $m = \lceil \sqrt{n} \rceil$. Observe that if $\beta = \alpha^x$, then using the euclidean algorithm we can write x as follows: $x = im + j$, where $0 \leq i, j < m$. Thus we have that $\beta = \alpha^x = \alpha^{im+j} = \alpha^{im}\alpha^j$, which implies that $\beta(\alpha^{-m})^i = \alpha^j$. To compute the discrete logarithm, we begin by computing and storing the values (j, α^j) for $0 \leq j \leq m$. We then compute $\beta(\alpha^{-m})$ and raise that to the exponent i for $0 \leq i \leq m-1$ and check these values against the stored values of α^j to find a match. When a match is found we have solved the DLP and we have $x = im + j$ as required.

The drawbacks of this algorithm lie in the computation and formulation of the table of pairs (j, α^j). At each stage we are required to compute a power of α and look in the table to see if it returns a match. If this is successful then the DLP has been solved. Unfortunately, one has to store around $O(\sqrt{n})$ group elements, perform around $O(\sqrt{n})$ multiplications to find the correct power of α, and in turn perform $O(\sqrt{n})$ table look-ups [62]. As a consequence this algorithm has an expected running time of $O(\sqrt{n})$, which makes it impractical for cryptographic purposes.

1.3 Pollard's ρ-Method

This algorithm has a similar running time to the Baby-Step Giant-Step method above yet requires less memory, an immediate advantage. Let G be a cyclic group of order n, where n is prime. G is then partitioned into three subsets of roughly equal size,

call these sets S_1, S_2 and S_3. We then define a sequence of group elements, $\{x_i\}$, as follows: $x_0 = 1$ and

$$x_{i+1} = f(x_i) \stackrel{\text{def}}{=} \begin{cases} x_i\beta & \text{if } x_i \in S_1, \\ x_i^2 & \text{if } x_i \in S_2, \\ x_i\alpha & \text{if } x_i \in S_3, \end{cases}$$

for $i \geq 0$. This in turn defines two sequences of integers $\{a_i\}$ and $\{b_i\}$ satisfying $x_i = \alpha^{a_i}\beta^{b_i}$ for $i \geq 0$. The sequences $\{a_i\}$ and $\{b_i\}$ are defined as follows: set $a_0 = 0 = b_0$ and for $i \geq 0$,

$$a_{i+1} = \begin{cases} a_i & \text{if } x_i \in S_1, \\ 2a_i \mod n & \text{if } x_i \in S_2, \\ a_i + 1 \mod n & \text{if } x_i \in S_3, \end{cases}$$

and

$$b_{i+1} = \begin{cases} b_i + 1 \mod n & \text{if } x_i \in S_1, \\ 2b_i \mod n & \text{if } x_i \in S_2, \\ b_i & \text{if } x_i \in S_3. \end{cases}$$

We then begin with a pair (x_1, x_2) and iteratively compute pairs (x_i, x_{2i}) until we find a pair of group elements such that $x_i = x_{2i}$ for some i[1]. When such a pair is found we then have the following relation: $\alpha^{a_i}\beta^{b_i} = \alpha^{a_{2i}}\beta^{b_{2i}}$. Thus $\beta^{b_i - b_{2i}} = \alpha^{a_i - a_{2i}}$. Taking the logarithm here to the base α, we obtain the equation

$$(b_i - b_{2i}) \log_\alpha \beta = a_i - a_{2i} \mod n.$$

[1]This technique is commonly known as using Floyd's Cycle Finding Algorithm, details of which can be found in [38].

Provided that $b_i \not\equiv b_{2i} \mod n$ we can invert the quantity $b_i - b_{2i}$ (recall that n here was prime so that $\gcd(b_i - b_{2i}, n) = 1$), and thus obtain a solution for $\log_\alpha \beta$ namely, $(b_i - b_{2i})^{-1}(a_i - a_{2i}) \mod n$.

Note that this algorithm is a randomized algorithm and has the potential, albeit very small, to terminate without finding a solution. In the event that the algorithm fails, we can run through the process a second time starting with new values for a_0 and b_0 in $[1, n-1]$ [62]. Since this algorithm requires less storage then the Baby-Step Giant-Step method it is preferable over the latter method; yet since this algorithm has a similar expected running time, $O(\sqrt{n})$, this attack is inefficient for practical purposes.

1.4 Pollard's λ-Method

In [66] and [67] Pollard describes a method for computing the discrete log in a group where we know a little more information than usual. This method is more commonly referred to as Pollard's Kangaroo Method, since the method was first described as having a wild kangaroo W run through random values of $\alpha^j, j \in [1, p-2]$ and a tame kangaroo, T whose job it is to run through and set a trap for the wild kangaroo and catch him. When the two kangaroos meet, the discrete logarithm is solved. This setting will be made more precise below. Let $G(= \mathbb{Z}_p)$ be a cyclic group of prime order, α a generator for G and $\beta \in G$ such that $\alpha^x = \beta$. Suppose further that we know that $x \in [a, b] \subset [0, p-1]$ where the value $l = b - a$ is small[2].

[2]Suppose that $b - a \approx 2^{100}$ then $\sqrt{b-a} \approx 2^{50}$, which is a fairly manageable quantity. This would be a fair definition of small.

Remaining true to the original setting, our tame kangaroo T will attempt to catch the wild kangaroo W. To do this we must have a way to keep track of where they are. We do this as follows. Let $J = \lfloor \log_2(l) \rfloor$, since $l = b - a$ is a small quantity then so is J. Let $S = \{2^0, 2^1, \ldots, 2^{J-1}\} = \{s(0), s(1), \ldots, s(J-1)\}$. Each jump made by a kangaroo will use a distance that is randomly selected from the set S. We need a way to keep track of the distance traveled by each kangaroo. Let's begin with our tame kangaroo T. Let T begin his journey at a point that we know, set $t_0 = \alpha^b \mod p$. We can then track its path

$$t(i+1) \equiv t(i)\alpha^{s(t(i) \mod J)} \mod p \text{ for } i = 1, 2, 3, \ldots \tag{2.1}$$

Of course T cannot jump forever. Let T jump n steps and then stop. Discussion of n will follow when we see what the wild kangaroo will do. After n jumps we record the distance traveled by T as

$$d(n) = \sum_{i=0}^{n} s(t(i) \mod J)$$

Using this expression for the distance traveled by T we can express (2.1) as

$$t(n) \equiv \alpha^{b+d(n-1)} \mod p$$

Now we have to deal with the distance that W will travel. We can use the same idea, except that W will start from an unknown point, namely $w_0 = \alpha^x$; this unknown starting point is why this kangaroo is deemed the wild kangaroo. Similar to above the path traveled by W is

$$w(j+1) \equiv w(j)\alpha^{s(w(j) \mod J)} \mod p \text{ for } j = 1, 2, 3, \ldots \tag{2.2}$$

and its distance traveled is recorded as

$$D(j) = \sum_{k=0}^{j} s(w(k) \mod J).$$

Hence we can express (2.2) as

$$w(i) \equiv \alpha^{x+D(i-1)} \mod p.$$

Due to the birthday problem[3], after approximately $\sqrt{l}$ jumps a collision should occur. When this happens we have some indices i, j such that $t(i) = w(j)$ and from this point onward $t(s) = w(r)$ for all $s \geq i$, $r \geq j$. When a collision has occurred we obtain the following relation:

$$\alpha^{x+D(m-1)} \equiv \alpha^{b+d(n-1)} \mod p$$

Hence we've solved for the unknown quantity x, and the DLP, since

$$x = b + d(n-1) - D(m-1).$$

Note that we are still unsure of how many jumps T should make. If we take $n = \sqrt{l}$ then the birthday problem tells us that the probability of a collision tends to 1 quickly if the number of steps exceeds $\sqrt{l}$, hence setting n to this quantity increases the likelihood of a collision [56].

This of course makes the algorithm probabilistic. If the algorithm fails to yield a collision after n steps, it can be re-initialized with a new starting value for the wild kangaroo [56]. This algorithm has a much better running time since we are only searching for a solution to the discrete log in the interval $[a, b]$. Hence the running

[3]See [56] for more details.

time for the algorithm depends on the lengths of this interval, more precisely the algorithm is expected to have a running time of $O(\sqrt{l})$ [67].

A speed up of this algorithm can also be obtained. Suppose that the algorithm is set up on a parallel computing system with P processors. Then an application of the algorithm has expected running time of $O(\frac{\sqrt{l}}{P})$ [67], making this parallelized version fairly efficient. Of course this all presupposes that knowledge of the interval to which x belongs is known and is small.

1.5 The Pohlig-Hellman Method

The Pohlig-Hellman algorithm is an effective attack on the discrete logarithm in $\mathbb{F}_p$, for a prime p, provided that a factorization of $p-1$ can easily be found, and that it uses relatively small primes[4]. The setup is as follows.

Suppose that p is a prime number and α a generator for the cyclic group $\mathbb{F}_p^\times$. Assume that $\beta \in \mathbb{F}_p^\times$ is such that $\beta = \alpha^x$. We want to solve $x = \log_\alpha \beta$. Assume further that

$$p - 1 = \prod_{i=1}^{k} q_i^{r_i}$$

where the q_i's are prime numbers in the factorization of $p-1$. The main idea here is that we will solve a system of congruences modulo the q_i's, and then we will reassemble, using the Chinese Remainder Theorem(CRT), a solution for the original problem.

For the moment let's fix choices q, r and we will work mod q^r. Recall that we are

[4]Here small refers to the amount of digits in the prime number.

searching for a solution to $x = \log_\alpha \beta$. We can perform the following trick. Write x as

$$x \equiv x_0 + x_1 q + x_2 q^2 + x_3 q^3 + \ldots + x_{r-1} q^{r-1} \mod q^r \text{ for } 0 \leq x_i \leq q-1. \quad (2.3)$$

The important thing to notice here is that we can successively compute the x_i's. We can then take this equation for x, multiply by the constant $\frac{p-1}{q}$ and obtain

$$x(\frac{p-1}{q}) \equiv x_0(\frac{p-1}{q}) + x_1(p-1) + x_2 q(p-1) + x_3 q^2(p-1) + \ldots + x_{r-1} q^{r-2}(p-1)$$

or simply

$$x(\frac{p-1}{q}) \equiv x_0(\frac{p-1}{q}) + (p-1)m, \text{ for some } m \in \mathbb{Z}.$$

(Where the above congruences hold true modulo q^r). Now that we have obtained this, we can see what happens when we work with our generator α and our solution β all modulo p.

Raising β to the exponent of $\frac{p-1}{q}$, we get:

$$\beta^{\frac{p-1}{q}} \equiv (\alpha^x)^{\frac{p-1}{q}} \equiv \alpha^{x_0(\frac{p-1}{q})+(p-1)m} \equiv \alpha^{x_0(\frac{p-1}{q})} \equiv (\alpha^{\frac{p-1}{q}})^{x_0} \mod p$$

It is now possible to run through a list of stored values in an attempt to find a match for x_0. The stored values are obtained by setting $c \equiv g^{\frac{p-1}{q}} \mod p$ and computing $c^j \mod p$ where $0 \leq j \leq q-1$. The CRT implies that one and only one value will be congruent to $\beta^{\frac{p-1}{q}}$. When a match is found we are able to solve for x_0, namely $j = x_0$. Once we have solved for x_0 we can perform a similar trick to solve for x_1, this time we multiply (2.3) by the quantity $\frac{p-1}{q^2}$ to obtain

$$x(\frac{p-1}{q^2}) \equiv x_0(\frac{p-1}{q^2}) + x_1(\frac{p-1}{q}) + (p-1)m_1, \text{ for some } m_1 \in \mathbb{Z}. \quad (2.4)$$

Unfortunately we cannot immediately raise β to this exponent; we first need to shift it somehow to compensate for the extra power of q that we have on the left hand side of (2.4). To do this, we set $\beta_1 = \beta \cdot \alpha^{-x_0}$. Thus we have that

$$\begin{aligned}\beta_1^{\frac{p-1}{q^2}} &\equiv \left(\alpha^x \cdot \alpha^{-x_0}\right)^{\frac{p-1}{q^2}} \\ &\equiv \left(\alpha^{q(x_1+x_2q+\ldots)}\right)^{\frac{p-1}{q^2}} \equiv \alpha^{\frac{p-1}{q}(x_1+x_2q+x_3q^2+\ldots)} \\ &\equiv \alpha^{(\frac{p-1}{q})x_1+(p-1)m_1} \equiv \alpha^{(\frac{p-1}{q})x_1} \mod p\end{aligned}$$

and again we can look in our table of precomputed values and determine x_1. This is then repeated until all the x_i's are known and for every prime q_i appearing in the factorization of $p-1$. A final application of the CRT applied with each q_i gives us the final value of x allowing us to solve the DLP in this case.

There are two issues that we should be aware of for this attack. The first is that we are assuming that $p-1$ can be factored efficiently and that it contains only small primes in its factorization. The second issue lies in the precomputation of the c^j values. Since the $p-1$ contains only small primes, the values of the q_i's are small and hence so are the values of the c^j's. Since these values are small, they can be efficiently computed, and require relatively small storage for each list. The expected running time for this algorithm is $O(\sum_{i=1}^{k} r_i(\log_2(p-1) + \sqrt{q_i}))$ [63], or equivalently $O(\sqrt{q_i'})$ where q_i' is the largest prime factor of $p-1$. To avoid this attack in its entirety, one can chose the value of p carefully so that $p-1$ has a large prime factor.

1.6 The Index Calculus

The Index Calculus is the most powerful attack against the DLP [62]. Unfortunately it does not always apply, but when it does it results in a subexponential running

time. Until now all running times have been exponential in the order of the input, making them impractical.

If we assume that a solution to the DLP exists then the problem can be reduced to solving the following:

$$k = \log_\alpha \beta$$

Here's how the index calculus works. First we chose a *Factor Basis* $\mathcal{F}_r = \{2, 3, 5, 7, 11, \ldots, p_r\}$ made up of primes for some r which will be chosen later. We then compute the semi-group generated by $\mathcal{F}_r$, ie. $\langle \mathcal{F}_r \rangle$[5].

The next step consists of computing powers of α and lifting each of these values from $\mathbb{F}_p$ to $\mathbb{Z}$.

$$\alpha^j \equiv a_j \mod p \quad 1 \leq a_j < p$$

Each a_j is then checked against $\langle \mathcal{F}_r \rangle$. If $a_j \in \langle \mathcal{F}_r \rangle$, we record the value

$$a_j = \prod_{i=1}^{r} p_i^{e_i(j)} \tag{2.5}$$

Notice that since $a_j = \alpha^j \in \mathbb{F}_p^\times$, and $\mathbb{F}_p^\times$ has order $p - 1$, each relation in (2.5) gives a linear equation

$$j \equiv \sum_{i=1}^{r} e_i(j) \log_\alpha(p_i) \mod p - 1 \tag{2.6}$$

We compute powers of α until we obtain r independent linear relations of the form (2.6). We then have r equations with r unknowns, $\log_\alpha(p_1), \ldots, \log_\alpha(p_r)$. This is where choosing the proper value for r comes into play. It should be chosen so as to include the complete factorization of a number a. This is a drawback to this method of trying to solve the DLP using the index calculus.

[5]A semi-group is a set with a binary operation such that multiplication is associative.

The final step is to compute and lift the quantities $\beta\alpha^i, 1 \leq i \leq r$ to $\mathbb{Z}$ as before:

$$\text{ie. we compute } \alpha^i\beta \equiv b_i \mod p \quad 1 \leq b_i < p$$

We do this until we find a single value of i for which $b_i \in \langle \mathcal{F}_r \rangle$, say

$$b_i = \prod_{j=1}^{r} p_j^{f_j}.$$

Since $b_i = \alpha^i\beta \in \mathbb{F}_p^{\times}$ we have that

$$i + \log_\alpha \beta \equiv \sum_{j=1}^{r} f_i \log_\alpha(p_i) \mod p - 1$$

We then have the value of $\log_\alpha \beta$. Thus we have solved the DLP.

The main drawback of this algorithm lies in the choice of the factor base. If the factor base is not chosen properly then there is a possibility that we could fail to obtain enough linear relations to be able to solve the problem. If we chose too many primes to use in our factor base then we obtain too many relations, in which case it may also not be possible to solve the problem [62]. This attack also assumes that we are able to factor group elements into products of primes, a process that cannot be performed in all groups.

The Index Calculus can be used in both $\mathbb{Z}_p$ and in $\mathbb{F}_{2^m}$. In the latter field, the factor base is chosen to consist of all irreducible polynomials of degree at most some prescribed bound $n \leq m - 1$ [62]. Again, just as in the case of choosing the number of primes in a factor basis in the setting of $\mathbb{Z}_p$, choosing a proper bound n on the degree of the irreducible polynomials is a drawback of this attack. With properly chosen factor bases in either situation the expected running time for this algorithm

is

$$L_q[\frac{1}{2}, c] = O(exp((c + o(1))(\log_2 q)^{\frac{1}{2}}(\log_2 \log_2 q)^{\frac{1}{2}}))$$

where $q = p$ or 2^m, and $c > 0$ is a constant [62]. This is a subexponential running time. This is the first expected subexponential running algorithm that we have encountered.

1.7 Conclusions

The following table summarizes our results thus far.

Attack	Expected Running Time
Exhaustive Search	$O(n)$
BSGS	$O(\sqrt{n})$
ρ-method	$O(\sqrt{n})$
λ-method	$O(\sqrt{l})$
Pohlig-Hellman	$O(\sum_{i=1}^{k} r_i(\log_2(p-1) + \sqrt{q_i})) = O(\sqrt{q_i'})$
Index Calculus	$L_q[\frac{1}{2}, c]$

Table 2.1: Expected Running Times of the Attacks on the DLP

The Index Calculus offers the only subexponential running time in the most general setting where no extra information about the structure of the underlying group is known. It is currently the most powerful attack against the DLP when it can be used.

As mentioned at the beginning of this chapter, it is important to be aware of these attacks since they will be of use when we discuss Pairing attacks on elliptic curves

in IV.3.2. To put this in context, several of these elliptic curve attacks, such as the MOV and the Frey-Rück attacks, attempt to reduce an instance of the ECDLP to an instance of the DLP in an isomorphic group. Depending on the structure of the group, one of the methods discussed in this chapter can be used to solve the DLP, hence solving the given instance of the ECDLP.

3 Elliptic Curves and Other Essentials

1 What is an Elliptic Curve?

There are several ways to introduce the subject of elliptic curves. One approach is from the realm of complex analysis. One could build a lengthy theory on Weierstrass $\wp$-functions, and realize that an elliptic curve is nothing more than an torus in the complex plane[6]. A second method would be to develop the theory of curves and varieties. Once this has been done, we then appeal to the Riemann Roch theorem to not only prove the existence of elliptic curves, but the Riemann Roch theorem can also be used to prove the group law, which we will see below[7]. Since we want to begin using elliptic curves for cryptosystems we will assume one of the above settings and begin by introducing topics and definitions which we will need throughout the rest of this document[8].

[6] See [2] and [78] for these details.

[7] These details can be found in [2] and [77].

[8] Other excellent sources of material in this introductory chapter are [5], [38], [81] and [87].

1.1 Definitions

An elliptic curve E defined over a field K is the locus of points satisfying an equation of the form

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

where $a_i \in K$ for all i. To this equation we can associate several quantities which each have their own role in the study of elliptic curves.

The *discriminant* of an elliptic curve E, denoted as Δ, is the quantity

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

where

$$\begin{aligned} d_2 &= a_1^2 + 4a_2 \\ d_4 &= 2a_4 + a_1a_3 \\ d_6 &= a_3^2 + 4a_6 \\ d_8 &= a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2 \end{aligned}$$

We also define a quantity known as the j-invariant of a curve. This quantity will come in useful later when we discuss the method of Complex Multiplication and isomorphisms of elliptic curves in Chapter V. The *j-invariant* is the quantity $(d_2^2 - 24d_4)^3/\Delta$. This quantity completely classifies the isomorphism classes of E, since two elliptic curves are isomorphic iff their j-invariants are equal [5].

The equation in (3.1) is known as the general Weierstrass equation. If the characteristic of K is not equal to 2 or 3 we can perform a linear transformation to obtain a new form of (3.1). One can find the explicit details of this transformation in either

[38] or [77]. The net result of this transformation is that we reduce to an equation of the form

$$y^2 = x^3 + Ax + B. \tag{3.2}$$

This equation is the most common form that we will see throughout this document. It is not however the only one. Many implementations of elliptic curve cryptosystems take place over $\mathbb{F}_2$ or an extension field $\mathbb{F}_{2^n}$ for some $n \in \mathbb{Z}$. In the case that the characteristic of the field is 2, then an alternate linear change of variables can be made to obtain an equation of the form

$$y^2 + xy = x^3 + a_2x^2 + a_6. \tag{3.3}$$

1.2 The Group Law

Let E be an elliptic curve over a field K, with $\text{char}(K) \neq 2, 3$. In this section we define, first geometrically and then algebraically, the group law on an elliptic curve.

One approach to defining the group law on an elliptic curve is to do so in projective space, and then to reduce to the affine case by taking the point at infinity to be the point $[0 : 1 : 0]$[9]. Instead we take this setup for granted and begin to define the group law in the affine setting, denoting the point at infinity as $\mathcal{O}$. The group law can then be defined geometrically. Suppose P_1 and P_2 are two points on an elliptic curve and we wish to determine $P_1 \oplus P_2$. We connect the points P_1 and P_2 with a line l. This line l will intersect the curve at a third point, which we will denote P_3'. We then connect P_3' with the point at infinity, which will simply be a vertical line

[9]This treatment can be found in [81]

in this case, with the line l'. The line l' will then intersect the curve E in a third point as well. It is this point that we will denote as $P_1 \oplus P_2$, the sum of P_1 and P_2 on E. Later, we will drop the $\oplus$ notation in favour of $+$ where there should be no confusion. From this type of geometric construction we can immediately see how to define things algebraically.

Theorem 3.1 (The Group Law) *Let E be an elliptic curve over K with char$(K) \neq 2, 3$, with defining equation $E : y^2 = x^3 + Ax + B$. Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ be points on E such that $P_1, P_2 \neq \mathcal{O}$. We define $P_1 + P_2 = P_3 = (x_3, y_3)$ as follows.*

1. *If $x_1 \neq x_2$ then $x_3 = m^2 - x_1 - x_2, y_3 = m(x_1 - x_3) - y_1$ where $m = \frac{y_2 - y_1}{x_2 - x_1}$.*
2. *If $x_1 = x_2$ but $y_1 \neq y_2$, then $P_1 + P_2 = \mathcal{O}$*
3. *If $P_1 = P_2$ and $y_1 \neq 0$, then $x_3 = m^2 - 2x_1, y_3 = m(x_1 - x_3) - y_1$ where $m = \frac{3x_1^2 + A}{2y_1}$.*
4. *if $P_1 = P_2$ and $y_1 = 0$ then $P_1 + P_2 = \mathcal{O}$*

Notice that we did not take into account that P_1 or P_2 could in fact be the point at infinity here. Doing so results in several special cases which we omit but can be found in a variety of sources including [5], [38], [77] and [87]. We can also define the group law algebraically over fields of characteristic 2 and 3; however we do not do it here. Excellent sources for these definitions are [5], [38] and [77, Appendix A].

Regardless of the field of definition, and including all special cases for points on E, we now have a group which satisfies the following properties,

Theorem 3.2 *The addition law defined above on an elliptic curve E gives E the structure of an Abelian Group. We will denote the identity element as $\mathcal{O}$, and the inverse of point P as $-P$.*

Proof: The proof of this theorem can be approached two different ways. We can either give a geometric proof, since the group law was defined as such at the outset, or we can prove these statements algebraically. In either case the group axioms are all easily checked, the only difficult axiom is associativity. Normally the way that one proves associativity is with an argument in projective space using Bézout's theorem. The statement about Bézout's theorem is as follows and a proof can be found in [37].

Theorem 3.3 (Bézout's Theorem) *Let C_1 and C_2 be two projective curves defined over $\mathbb{C}$ of degrees m and n respectively which share common component. Then the sum of the intersection numbers*[10]*, counting multiplicities, at the point of intersection P is mn.*

With Bézout's theorem in hand we observe that if we constructed $(P_1 + P_2) + P_3$ and $P_1 + (P_2 + P_3)$ geometrically, this gives us the conditions to satisfy the statement of the theorem, thus we have two projective conics intersecting in eight points, hence the ninth point must also coincide and we have that $(P_1 + P_2) + P_3 = P_1 + (P_2 + P_3)$, and thus the group law is associative. □

As mentioned in the outset of this chapter, one could prove the group law using the Riemann Roch theorem. We invite and encourage the interested reader to prove the above theorem using the algebraic relations for the group law.

[10] see [37] for the definition of intersection number

1.3 Elliptic Curves Over Finite Fields

The above section is a general construction; the group law applies to elliptic curves over all fields. Since we are concerned with cryptographic applications, we will primarily deal with elliptic curves over finite fields. We still make the same convention that $\mathcal{O}$ is the point at infinity and is the identity element for our groups. When the theory of elliptic curves over $\mathbb{Q}$ is fully developed, we can easily translate all constructions to a finite field $\mathbb{F}_q$, for some prime q or $q = p^n$ for some $n \in \mathbb{Z}$. Theorems and constructions valid for elliptic curves over $\mathbb{Q}$ remain true over $\mathbb{F}_q$ by virtue of the following theorem.

Theorem 3.4 (Reduction Modulo p) *Let E be a non-singular elliptic curve over $\mathbb{Q}$ with $\Delta \neq 0$. Let $\Phi \subseteq E(\mathbb{Q})$ be the subgroup of all points of finite order. Then for any prime p with $p \nmid 2\Delta$, the map*

$$\begin{aligned} \Phi &\rightarrow E(\mathbb{F}_p) \\ P &\mapsto \tilde{P} = \begin{cases} (\tilde{x}, \tilde{y}) \text{ if } P = (x, y) \\ \tilde{\mathcal{O}} \text{ if } P = \mathcal{O} \end{cases} \end{aligned}$$

is an isomorphism of Φ and a subgroup of $E(\mathbb{F}_p)$.

Of course if we go back and examine the group law we have to be careful with the denominators to make sure we don't divide by zero, that is, where $x_2 - x_1$, say is a multiple p. There are a few more cases to consider, but we can develop corresponding formulas to perform arithmetic on the curve $E(\mathbb{F}_p)$. There are many sources that discuss finite field arithmetic on elliptic curves. Two excellent sources

are [5] and [38], both of which give detailed accounts and present algorithms about fast arithmetic in finite fields, and especially in the case where $\text{char}(K) = 2$.

There are a few additional results that should also be mentioned here. The first will be used substantially throughout this document, and will lead us into the next section.

Theorem 3.5 (The Hasse-Weil Theorem) *Let C be a non-singular curve of genus g defined over a finite field $\mathbb{F}_q$. Then the number of points on C is $q + 1 + \epsilon$, where $|\epsilon| \leq 2g\sqrt{q}$.*

An elliptic curve E is a curve of genus $g = 1$, which reduces the above relation to the following:

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

This theorem is fairly significant, since we will be able to at least estimate the number of points on $E(\mathbb{F}_q)$. In the next section, we will see that we can actually calculate the number of points explicitly. The estimation will come into play again in Chapter V when we look at generating cryptographically strong elliptic curves. Algorithms for generating curves generally estimate the number of points on E so as to avoid some of the more well known attacks on elliptic curves(which we will encounter in Chapter IV).

The following results will appear again when we talk about Schoof's algorithm in the following section and Pairing attacks IV.3.2, and in particular when we discuss the MOV attack.

Definition 3.1 *Let $\mathbb{F}_q$ be a finite field with algebraic closure $\overline{\mathbb{F}_q}$. The map*

$$\begin{aligned} \varphi_q : \overline{\mathbb{F}_q} &\rightarrow \overline{\mathbb{F}_q} \\ x &\mapsto x^q \end{aligned}$$

is called the Frobenius endomorphism. φ_q acts on points on $E(\overline{\mathbb{F}_q})$ as $\varphi_q(x, y) = (x^q, y^q)$ and $\varphi_q(\mathcal{O}) = \mathcal{O}$.

Along with this definition comes the following theorem, which will be used several times throughout this thesis, whose proof can be found in [87].

Theorem 3.6 *Let E be an elliptic curve defined over $\mathbb{F}_q$. Let $a = q + 1 - \#E(\mathbb{F}_q) = q + 1 - deg(\varphi_q - 1)$ then*

$$\varphi_q^2 - a\varphi_q + q = 0$$

as endomorphisms of E. Furthermore, a is the unique integer such that this equation is satisfied and $a \equiv Trace((\varphi_q)_m) \mod m$ for all m with $\gcd(m, q) = 1$.

Finally, we discuss an important family of mappings between elliptic curves. These mappings are defined over arbitrary fields K and not simply $\mathbb{F}_q$. In particular they will help us extend the GHS attack which will be looked at in the next chapter.

Definition 3.2 *Let E_1 and E_2 be two elliptic curves defined over K. An isogeny from E_1 to E_2 is a morphism $\phi : E_1 \rightarrow E_2$ such that $\phi(\mathcal{O}) = \mathcal{O}$. In particular $\#E_1 = \#E_2$.*

2 Schoof's Algorithm

The purpose of this section is to introduce and give a description of Schoof's algorithm for counting the number of points on an elliptic curve, given in Weierstrass form, over the finite field $\mathbb{F}_p$ for large primes p. Determining the order of the group will be an essential tool when attempting to generate cryptographically strong elliptic curves.

Schoof presented his original findings in his 1985 paper [69]. In this paper he was able to show that the algorithm runs in polynomial time and takes at most $O\left(\log^9 p\right)$ bit operations to complete.

Let φ_p be the p^{th} power Frobenius endomorphism as defined in Definition 3.1. It can be shown that φ_p maps points on E to points on E, and that it respects the group law. Thus φ_p is a group endomorphism of E over $\mathbb{F}_p$. Let the trace of the Frobenius endomorphism be t. Then the following equation is satisfied:

$$\varphi_p^2 - [t]\varphi_p + [p] = \mathcal{O}. \tag{3.4}$$

Thus, for any point P on E we have the following:

$$(x^{p^2}, y^{p^2}) - [t](x^p, y^p) + [p](x, y) = \mathcal{O}$$

Notice here that the subtraction and addition operations are curve operations. Thanks to Hasse, we have a bound on the value of the trace of φ_p, $|t| \leq 2\sqrt{p}$. At the heart of Schoof's algorithm lies a calculation for the value of t. We will come back to this shortly.

For non-negative integers m, we define the set of *m-torsion points* of E, denoted $E[m]$, as follows;

$$E[m] = \{P \in E(\overline{\mathbb{F}}_p) : [m]P = \mathcal{O}\}.$$

From the algebraic expressions of the group law given above, we can see that the coordinates of the sum $P_1 + P_2$ of two points on the curve are rational functions of P_1 and P_2 [5, 39]. Thus the multiplication by m map $(x, y) \mapsto [m](x, y)$ can be expressed in terms of rational functions in x and y. We then have the following lemma.

Lemma 3.1 *Let E be an elliptic curve defined over the field $\mathbb{F}_p$, and let $m \in \mathbb{Z}, m \geq 2$. There exists polynomials $\psi_m \in E(\overline{\mathbb{F}}_p)[x, y]$ such that, for $P = (x, y) \in E(\overline{\mathbb{F}}_p)$ with $[m]P \neq \mathcal{O}$ we have*

$$[m]P = \left(x - \frac{\psi_{m-1}(x,y)\psi_{m+1}(x,y)}{\psi_m^2(x,y)} \ , \ \frac{\psi_{m+2}(x,y)\psi_{m-1}^2(x,y) - \psi_{m-2}(x,y)\psi_{m+1}^2(x,y)}{4y\psi_m^3(x,y)} \right).$$

The polynomial $\psi_m(x, y)$ is called the *mth division polynomial* of the curve E. These polynomials play a central role in Schoof's algorithm. Below are explicit recursive formulas for ψ_m. The *mth* division polynomial is defined as follows:

$$\begin{aligned}
\psi_0(x, y) = 0, \quad \psi_1(x, y) = 1, \quad \psi_2(x, y) = 2y \\
\psi_3(x, y) = 3x^4 + 6ax^2 + 12bx - a^2, \\
\psi_4(x, y) = 4y(x^6 + 5ax^4 + 20bx^3 - 5a^2x^2 - 4abx - 8b^2 - a^3), \\
\psi_{2m}(x, y) = \psi_m(\psi_{m+2}\psi_{m-1}^2 - \psi_{m-2}\psi_{m+1}^2)/2y, \quad (m > 2) \\
\psi_{2m+1}(x, y) = \psi_{m+2}\psi_m^3 - \psi_{m+1}^3\psi_{m-1} \qquad (m \geq 2)
\end{aligned}$$

Notice that from these we may define another set of polynomials. This time they will be polynomials in one variable, but they depend on the definitions of the division polynomials. We define $f_m \in \mathbb{F}_p[x]$ as follows. Eliminate all y^2-terms in ψ_m using the equation defining the curve E. The resulting polynomial, $\psi'(x, y)$, is in either $E(\overline{\mathbb{F}}_p)[x]$ or $yE(\overline{\mathbb{F}}_p)[x]$. Define $f_m(x) = \psi'(x, y)$ if m is odd, and $f_m(x) = \psi'(x, y)/y$ if

m is even. Defining these polynomials will help with certain calculations during the algorithm. Before we get into the algorithm we present one more theorem that will help us during its description.

Theorem 3.7 *: Let P be a point in $E(\overline{\mathbb{F}}_p)\backslash\{\mathcal{O}\}$, and let $m \geq 1$. Then $P \in E[m]$ iff $\psi_m(P) = 0$.*

Notice that this theorem can also be stated in terms of the polynomials f_m we defined earlier; see [5, 41].

For a prime $s \neq p$ we have a map which relates the Frobenius endomorphism to endomorphisms on $E[s]$ under the Galois action of $Gal(\overline{\mathbb{F}}_{|}\backslash\mathbb{F}_p)$.

$$\mathrm{End}_{\mathbb{F}_p}E \rightarrow \mathrm{End}_{Gal(\overline{\mathbb{F}}_p\backslash\mathbb{F}_p)}E[s] \tag{3.5}$$

In essence what this is saying is that we can view the Frobenius φ_p, as a map invariant under the Galois action that takes torsion elements to torsion elements[11]. This property of φ_p allows us to do the following: if we let ϕ_s denote the image of φ_p on the right side in (3.5), then by (3.4) we have the following relation holding on $E[s]$:

$$\phi_s^2 - t\phi_s + p = 0 \tag{3.6}$$

However, if we suppose that the equation

$$(\phi_s^2 - t'\phi_s + p)P = \mathcal{O} \tag{3.7}$$

holds for each $P \in E[s]$, and some $t' \in \mathbb{Z}$, then using (3.7) it can be shown that $(t' - t)\phi_s P = \mathcal{O}$ for all $P \in E[s]$. Since ϕ_s is invertible we get that $t \equiv t' \bmod s$.

[11]For greater details as to why this is so see [5, 46].

Therefore we may compute the trace of the Frobenius mod s by checking to see which relations hold in equation (3.7) over $E[s]$ [69]. This allows us to compute the order for each prime $s \neq p$ and then reassemble the order of $E(\mathbb{F}_p)$ using the CRT.

We are now ready to present the steps in the algorithm. Let E be an elliptic curve over $\mathbb{F}_p$, where $\text{char}(\mathbb{F}_p) \neq 2, 3$. The case where the characteristic of the field is 2 or 3 is handled in a similar manner, but the definitions above must be slightly altered to take care of division by zero. These cases are explained in [5]. For the time being we shall not concern ourselves with these cases.

To compute $\#E(\mathbb{F}_p)$, we compute the trace of the Frobenius endomorphism. Since we have a bound on t, we can compute the trace by computing $t \mod l$ for sufficiently small prime numbers l. If we compute $t \mod l$ for $l = 3, 5, 7, \ldots, L$ such that

$$\prod_{\substack{l \neq L \\ l \neq 2,p}} l > 4\sqrt{p} \tag{3.8}$$

we can determine t using the CRT. We can compute $t \mod l$ by checking which relations hold in (3.7). That is for a given element $\tau \in \mathbb{Z}/l\mathbb{Z}$ we check to see if

$$\phi_l^2 + p = \tau\phi_l \tag{3.9}$$

holds true on $E[l]$. To perform these computations we will use the division polynomials. If $P = (x, y) \neq \mathcal{O} \in E[l]$, then by Theorem 3.7, we have that the equation (3.9) holds iff

$$(x^{p^2}, y^{p^2}) + \left(x - \frac{\psi_{p-1}\psi_{p+1}}{\psi_p^2}, \frac{\psi_{p+2}\psi_{p-1}^2 - \psi_{p-2}\psi_{p+1}^2}{4y\psi_p^3} \right)$$

$$= \begin{cases} \mathcal{O} & \text{if } \tau \equiv 0 \bmod l, \\ \left(x^p - \left(\frac{\psi_{\tau-1}\psi_{\tau+1}}{\psi_\tau^2} \right)^p, \left(\frac{\psi_{\tau+2}\psi_{\tau-1}^2 - \psi_{\tau-2}\psi\tau+1^2}{4y\psi_\tau^3} \right)^p \right), & \text{otherwise.} \end{cases}$$

Then by Lemma 3.1, the point $P \in E[l]$ iff $\psi_l = 0$ or equivalently if $f_l = 0$. Using the equation for E and the formulas for the group law we can reduce this to checking which relations hold over polynomials of the form $H_1(x) = 0$ and $H_2(x) = 0$, for some polynomials in $\mathbb{F}_p[x]$. We can do this since $-P$ has the same x coordinate as P. Then for every $\tau \in \mathbb{Z}/l\mathbb{Z}$ we check $H_1(x), H_2(x) \equiv 0 \bmod f_l$, until we encounter a value of τ for which (3.9) holds. When this happens we have found a value for which $t \equiv \tau \bmod l$. Now we present the steps of Schoof's algorithm.

Here are the Steps needed to compute $\#E(\mathbb{F}_p)$.

1. First we compute L, for which equation (3.8) holds. At this time we also compile a list of the polynomials f_m for $m = 1, 2, \ldots, L$.

2. In this step we compute $t \bmod l$ for every prime $l \leq L$. First we test if there is a nonzero point $P \in E[l]$ for which $\phi_l^2 P = \pm kP$ holds. Here $k \equiv p \bmod l$, and $1 \leq k < l$. So we have to test if the following holds true;

$$x^{p^2} = x - \frac{f_{k-1}(x)f_{k+1}(x)}{f_k^2(x)(x^3+ax+b)} \text{ ,when k is even}$$

$$x^{p^2} = x - \frac{f_{k-1}(x)f_{k+1}(x)(x^3+ax+b)}{f_k^2}, \text{ when k is odd}$$

Note that the denominators do not vanish on $E[l]$, so that $\phi_l^2 = \pm kP$ iff

$$(x^{p^2} - x)f_k^2(x)(x^3 + ax + b) + f_{k-1}(x)f_{k+1}(x) = 0 \quad \text{for even } k$$

$$(x^{p^2} - x)f_k^2(x) + f_{k-1}(x)f_{k+1}(x)(x^3 + ax + b) = 0 \quad \text{for odd } k$$

and we can test if a point P exists in $E[l]$ by computing the gcd of the above polynomials with the polynomials f_l. If the gcd $\neq 1$ then we have the existence of such a point. Otherwise if the gcd is 1, we have that $\tau \neq 0$ in (3.9). We now arrive at two cases:

Case 1: This is the case in which there exists some nonzero point $P \in E[l]$ with $\phi_l^2 P = -pP$. If this is the case, then by (5), we have that $t\phi_l P = \mathcal{O}$. Since we have that $\phi_l P \neq \mathcal{O}$, then $t \equiv 0 \bmod l$. If it is the case that $\phi_l^2 P = pP$, then again by (3.7) we have that $(2p - t\phi_l)P = \mathcal{O}$ and that $\phi_l P = \frac{2p}{t}P$. From this we find that $t^2 \equiv 4p \bmod l$. Now, let $w \in \mathbb{Z}$, with $0 < w < l$, denote a square root of $p \bmod l$. We can find such a w by simple trial and error. Since $(\phi_l - \frac{1}{2}t^2) = 0$, the eigenvalues of ϕ_l acting on $E[l]$ are w or $-w$, say. Now we simply test to see if either $\phi_l P = \pm wP$ holds. In the first case we have that $t \equiv 2w \bmod l$, otherwise we get $t \equiv -2w \bmod l$. So we compute the gcd of the following polynomials to check if the gcd is 1.

$$\gcd(4(x^3 + ax + b)^{\frac{p-1}{2}} f_w^3(x) - f_{w+2}(x)f_{w-1}^2(x) + f_{w-2}(x)f_{w+1}^2(x), f_l(x))$$

$$\gcd(4(x^3 + ax + b)^{\frac{p+3}{2}} f_w^3(x) - f_{w+2}(x)f_{w-1}^2(x) + f_{w-2}(x)f_{w+1}^2(x), f_l(x))$$

The first case corresponds to w being odd, while the second case is for even w. When the gcd is 1, then we have $t \equiv -2w \bmod l$ for odd w, and $t \equiv 2w \bmod l$ for even w. In every case we have a solution for t, thus its value can be recovered

for step 3, which is explained below.

Case 2: In the case where $\phi_l^2 P \neq \pm pP$ for any $P \in E[l]$, then we test which of the relations 3.9 for $\tau \in \mathbb{Z}/l\mathbb{Z}^\times$. Then for $P = (x, y)$ and $k \equiv p \bmod l$ we have that

$$\phi_l^2 P + pP = \left(-x^{p^2} - x + \frac{\psi_{k-1}\psi_{k+1}}{\psi_k^2} + \lambda^2, -y^{p^2} - \lambda \left(-2x^{p^2} - x + \frac{\psi_{k-1}\psi_{k+1}}{\psi_k^2} + \lambda^2 \right) \right)$$

where

$$\lambda = \frac{\psi_{k+2}\psi_{k-1}^2 - \psi_{k-2}\psi_{k+1}^2 - 4y^{p^2+1}\psi_k^3}{4\psi_k y((x-x^p)\psi_k^2 - \psi_{k-1}\psi_{k+1})}.$$

Notice that the denominator of λ does not vanish over $E[l]$ since ψ_k has no zeros on $E[l]$. So we now let $\tau \in \mathbb{Z}$ with $0 < \tau < l$; as a result we now get the following.

$$\tau\phi_l P = \left(x^p - \left(\frac{\psi_{\tau+1}\psi_{\tau-1}}{\psi_\tau^2} \right)^p, \left(\frac{\psi_{\tau+2}\psi_{\tau-1}^2 - \psi_{\tau-2}\psi_{\tau+1}^2}{4y\psi_\tau^3} \right)^p \right)$$

In a completely similar way to what was presented in Case 1, we can now test which of the relations hold for equation (3.9) for $\tau = 1, 2, \cdots, l-1$. These computations involve evaluating polynomials modulo $f_l(x)$, and testing to see if they are zero $\bmod f_l(x)$. This then completes the description of step 2 of the algorithm.

3. The third and final step of the algorithm consists of recovering t using the CRT and the values of $t \bmod l$. Compute $\#E(\mathbb{F}_p) = p + 1 - t$.

It is clear that step 2 is the most time consuming step in algorithm. In his original analysis, Schoof was able to show that this is indeed the truth, and that step 2 takes at

most $O(\log^9 p)$ steps to complete, so that it dominates the calculation of $\#E(\mathbb{F}_p)$. It is also clear that a lot of the difficulty will lie in computing the gcd of the polynomials ψ_m and f_m.

Since 1985, Schoof's algorithm has been improved upon by Atkin and Elkies, effectively reducing the expected running time. Step 2 of the algorithm has been refined to include variations of the step by computing within the kernel of an isogeny of degree l, and computing a factor of f_l of degree $(l-1)/2$. Atkin gave a *sort and match* method used for "bad primes" l [13]. These improvements have since been incorporated in the algorithm, and it is now known as the SEA algorithm, in honour of all three. But many others have continued to refine step 2 of the original, and even the SEA algorithm, in order to develop faster methods of computing $\#E(\mathbb{F}_p)$. Nevertheless, we now have a way of determining $\#E(\mathbb{F}_p)$ when required.

Example: We present a small example of how to calculate $\#E(\mathbb{F}_q)$. This example is due to Washington in [87].

Let E be the elliptic curve over $\mathbb{F}_{19}$ defined by the equation $y^2 = x^3 + 2x + 1$. Let $\#E(\mathbb{F}_{19}) = 19 + 1 - a$, our task is to determine a.

Since $4\sqrt{19} < 30$ we see that we simply need to compute $a \mod l$ for $l = 2, 3, 5$.

For $l = 2$: Notice that we did not include $l = 2$ in the above algorithm. This case can be handle with a simple argument. If $x^3 + 2x + 1$ has a root in $e \in \mathbb{F}_{19}$, then $(e, 0) \in E[2]$, and so $\#E(\mathbb{F}_{19})$ has even order and thus $a \equiv 0 \mod 2$. To determine if x^3+2x+1 has a root in $\mathbb{F}_{19}$ we could just simply try all possible values in $\mathbb{F}_{19}$, instead we make use of some theory of polynomials over finite fields. The roots of $x^{19} - x$ are exactly the elements in $\mathbb{F}_{19}$ [16]. Thus we compute $\gcd(x^{19} - x, x^3 + 2x + 1)$, if this

is one there are no roots for $x^3 + 2x + 1$ and thus $a \equiv 1 \mod 2$. Indeed this is the case, hence $a \equiv 1 \mod 2$.

For $l = 3$: From Schoof's algorithm we see that to determine $a \mod 3$ we have to determine whether

$$(x^{361}, y^{361}) + (x, y) = \pm(x^{19}, y^{19})$$

for $(x, y) \in E[3]$.

We compute the x-coordinate of $(x^{361}, y^{361}) + (x, y)$ and substitute in the equation for E to obtain

$$(x^3 + 2x + 1)\left(\frac{(x^3 + 2x + 1)^{180} - 1}{x^{361} - x}\right)^2 - x^{361} - x$$

We now need to reduce this modulo ψ_3, the third division polynomial defined in the previous section. When we do this we realize that we need to compute the multiplicative inverse of $x^{361} - x \mod \psi_3$. However, $\gcd(x^{361} - x, \psi_3) = x - 8$ so the multiplicative inverse does not exist. On the other hand we notice that this means $x = 8$ is a root of ψ_3 and so the point $(8, 4) \in E(\mathbb{F}_{19})$ and has order three. Thus we have that $\#E(\mathbb{F}_{19}) \equiv 0 \mod 3$, and thus $a \equiv 2 \mod 3$.

For $l = 5$: Here we need to determine

$$(x^{361}, y^{361}) + (x, -y) = \pm 2(x^{19}, y^{19})$$

where $[19](x, y) = [-1](x, y) = (x, -y) \in E[5]$, hence the middle term in the above equation. Again as above, computing the x-coordinate we get

$$\left(\frac{y^{361} + y}{x^{361} - x}\right) - x^{361} - x \equiv \left(\frac{3x^{38} + 2}{2y^{19}}\right) - 2x^{19} \mod \psi_5.$$

Substituting x^3+2x+1 for y^2 and reducing ψ_5 we can determine that $a \equiv \pm 2 \mod 5$. Now we need to determine the y-coordinate so we may determine the sign on a. The y-coordinate of $(x^{361}, y^{361}) + (x, -y)$ can be shown to be

$$y_1 = y(9x^{11} + 13x^{10} + 15x^9 + 15x^7 + 18x^6 + 17x^5 + 8x^4 + 12x^3 + 8x + 6) \mod \psi_5$$

while the y-coordinate of $2(x^{19}, y^{19})$ is

$$y_2 = y(13x^{10} + 15x^9 + 16x^8 + 13x^7 + 8x^6 + 6x^5 + 17x^4 + 18x^3 + 8x + 18) \mod \psi_5.$$

A computation of $(y_1 + y_2^{19}/y) \mod \psi_5$ yields zero, which tells us that

$$(x^{361}, y^{361}) + (x, -y) = -2(x^{19}, y^{19})$$

and so $a \equiv -2 \mod 5$.

Thus we are left with solving the system of congruence

$$\begin{aligned} a &\equiv 1 \mod 2 \\ a &\equiv 2 \mod 3 \\ a &\equiv 3 \mod 5 \end{aligned}$$

which in turn yields the solution $a \equiv 23 \mod 30$. From the Hasse-Weil Theorem we know that $|a| < 2\sqrt{19} < 9$, thus $a = -7$ and hence $\#E(\mathbb{F}_{19}) = 19 + 1 + 7 = 27$ as required.

3 Divisors and Pairings

3.1 Divisors

In this section we present a treatment of divisors of curves. This section contains the necessary background for Pairing attacks on the ECDLP which we examine in

IV.3.2. The theorems provided in this section come from the standard sources on the subject [77] and [87].

Definition 3.3 *The divisor group of a curve E, written $Div(E)$, is the free abelian group generated by the points of E. A divisor is then a formal sum*

$$D = \sum_{P \in E} n_P(P)$$

with $n_P \in \mathbb{Z}$ and $n_P = 0$ for all but a finitely many $P \in E$. The degree of a divisor D is defined as

$$deg(D) = \sum_{P \in E} n_P.$$

We also define the subgroup of divisors of degree zero, $Div^0(E)$, as

$$Div^0(E) = \{D \in Div(E) \mid deg\ D = 0\}.$$

Definition 3.4 *[77] Let E be a non-singular(except possibly at the point at infinity) elliptic curve and P a non-singular point on E. The ring $\overline{K}[E]_P$ is a discrete valuation ring, with valuation*

$$\begin{aligned} ord_P : \overline{K}[E]_P &\rightarrow \mathbb{N} \cup \{\infty\} \\ ord_P(f) &= \max\{d \in \mathbb{N} \mid f \in M_P^d\} \end{aligned}$$

where M_P is the maximal ideal of $\overline{K}[E]_P$. We can extend this definition to the function field $\overline{K}(E)$ using $ord_P(f/g) = ord_P(f) - ord_P(g)$. This of course extends our original range of our ord_P operator to $\mathbb{Z} \cup \{\infty\}$.

This now leads to the following definition.

Definition 3.5 *The order of a function f at the point P is $ord_P(f)$. If $ord_P(f) > 0$ then f has a zero at P, while if $ord_P(f) < 0$ then f has a pole at P.*

We can now associate to f a divisor.

Definition 3.6 *The divisor of a function, denoted div(f), is given by the following*

$$div(f) = \sum_{P \in E} ord_P(f)(P)$$

The above construction is a result of the following homomorphism

$$\begin{aligned} \text{div} : \overline{K}(E)^{\times} &\rightarrow \text{Div}(E) \\ f &\mapsto \sum_{P \in E} ord_P(f)(P) \end{aligned}$$

This then gives rise to an important class of divisors D in the divisor group $\text{Div}(E)$, those that can be associated with a function f.

Definition 3.7 *A divisor $D \in Div(E)$ is said to be principal if $D = div(f)$ for some function $f \in \overline{K}(E)^{\times}$. Two divisors D_1, D_2 are said to be linearly equivalent, $D_1 \sim D_2$ if $D_1 - D_2$ is principal.*

From Definitions 3.3 and 3.7 we obtain an important class of divisors that will be seen extensively when we discuss the Weil Descent attack of Gaudry, Hess and Smart and the Abelian Variety attack of Gaudry in IV.3.3 and IV.3.6 respectively.

Definition 3.8 *The divisor class group or the Picard group of E, denoted Pic(E), is the quotient of Div(E) by the subgroup of principal divisors. The degree zero part of the class group, $Pic^0(E)$, is the quotient of $Div^0(E)$ by the subgroup of principal divisors.*

The following theorem summarizes all of the above information, its proof can be found in [87].

Theorem 3.8 *Let E be an elliptic curve defined over a field K. Let $D = \sum_P n_P(P)$ be a divisor of degree zero on E. Then there is a function f such that $D = div(f)$ iff $\sum_P [n_P](P) = \mathcal{O}$ on E.*

With the above material in hand we are now going to construct both the Weil pairing and the Tate-Lichtenbaum pairing. These pairings will then be used in IV.3.2.1 and IV.3.2.2 to describe attacks on certain types of elliptic curve cryptosystems.

3.2 The Weil Pairing

Let $n \in \mathbb{Z}$ be such that $\text{char}(K) \nmid n$, and let E be an elliptic curve such that $E[n] = \{P \in E(K) \mid [n]P = \mathcal{O}\} \subseteq E(K)$. We want to construct a function, which will be our pairing, which maps an element of $E[n] \times E[n]$ to an n^{th} root of unity, μ_n, in $\overline{K}$. So let $T \in E[n]$. By Theorem 3.8, there is a function f such that $\text{div}(f) = n(T) - n(\mathcal{O})$. Furthermore, if we chose a point $T' \in E[n^2]$ such that $[n]T' = T$, we can use the above theorem a second time to construct a function g such that $\text{div}(g) = \sum_{R \in E[n]}[(T' + R) - (R)]$. Of course since we want this to be a divisor we have to check to make sure that the sum is indeed $\mathcal{O}$. But since there are n^2 points $R \in E[n]$, the point R in $\sum(T' + R)$ will cancel with $\sum(R)$, thus making the sum $[n^2]T' = [n]T = \mathcal{O}$. Notice that g is independent of the choice of T' [87], since any two choices for T' will differ by an element in $E[n]$.

Now let $f \circ n$ denote the function that multiplies a point by n then applies the

function f. The point $P = T' + R$ with $R \in E[n]$ is a point such that $[n]P = T$. Then

$$\mathrm{div}(f \circ n) = [n]\{\sum_R (T' + R)\} - [n]\{\sum_R (R)\} = \mathrm{div}(g^n)$$

Thus up to multiplication by a suitable constant we can assume that $f \circ n = g^n$.

Lastly if $S \in E[n]$ and $P \in E(\overline{K})$, we have that

$$g(P+S)^n = f([n](P+S)) = f([n]P) = g(P)^n$$

As a result we have that $g(P+S)/g(P) \in \mu_n$, and is independent of the choice of the point P. We then define the map we have just constructed by

$$e_n(S,T) = \frac{g(P+S)}{g(P)}.$$

The following theorem lists the important properties of this pairing. The proof can be found in [77] or [87].

Theorem 3.9 (Properties of The Weil Pairing) *Let E be an elliptic curve defined over a field K and let n be a positive integer. Assume that $\mathrm{char}(K) \nmid n$. The Weil Pairing*

$$e_n : E[n] \times E[n] \to \mu_n$$

satisfies the following properties:

1. *e_n is linear in each variable.*
2. *e_n is non-degenerate in each variable*
3. *for all $T \in E[n]$, $e_n(T,T) = 1$*
4. *for all $S, T \in E[n]$, $e_n(S,T) = e_n(T,S)^{-1}$*
5. *the pairing is Galois invariant*

The first four properties, especially the first two, will be exploited extensively when development of the MOV attack takes place in IV.3.2.1. This deeper understanding of the Weil pairing also gives us means to compute the pairing in an attempt to solve the ECDLP.

3.3 The Tate-Lichtenbaum Pairing

In a similar manner to above, we will construct the Tate-Lichtenbaum Pairing that will be used in IV.3.2.2 by the Frey-Rück attack.

In the construction of the Tate-Lichtenbaum pairing we will need a very powerful and remarkable result; we state it here without proof, which can be found in either [6] or [77, Exer. 2.11].

Theorem 3.10 (Weil Reciprocity) *Let f and g be non-zero constant functions defined on a curve C over K, with $div(f)$ and $div(g)$ having disjoint support*[12]*. Then $f((g)) = g((f))$.*

As we will see below, this theorem will give us a means of actually computing the pairing.

Let E be an elliptic curve defined over a field K_0. Let n be a positive integer with $\gcd(\mathrm{char}(K_0), n) = 1$. Define $K = K_0(\mu_n)$ to be the extension field of K_0 generated by the set of n^{th} roots of unity, $\mu_n = \{u \in \overline{K}_0^{\times} \mid u^n = 1\}$. Take $E(K)[n] = \{P \in E(K) \mid [n]P = \mathcal{O}\}$, and $nE(K) = \{[n]P \mid P \in E(K)\}$. Notice that $nE(K)$ is a subgroup of $E(K)$, and hence we can look at the quotient group $E(K)/nE(K)$. We

[12]The support of a divisor $D = \sum n_P(P)$, is the set of points where $n_P \neq 0$. Thus the disjoint support of two divisors implies that the sets of points for which $n_P \neq 0$ are disjoint.

are now going to define a pairing on $E(K)[n] \times E(K)/nE(K)$, however we need a place to map to. If we define the following set, $(K^\times)^n = \{u^n \mid u \in K^\times\}$, we can form the quotient $K^\times/(K^\times)^n$, which is a group of exponent n and is isomorphic to μ_n [6].

Now let $P \in E(K)[n]$ and $Q \in E(K)/nE(K)$. Notice here that technically we should be writing Q as a coset in the second group, instead we will simply think of Q are representative of an equivalence class. Now since $[n]P = \mathcal{O}$, we can find a function f such that $div(f) = n(P) - n(\mathcal{O})$. Take D to be a degree zero divisor equivalent to $(Q) - (\mathcal{O})$, and such that D is defined over K with disjoint support from $div(f)$. To do this we can simply choose a random $S \in E(K)$ and define $D = (Q + S) - (S)$. Since both $div(f)$ and D are defined over K, the value $f(D) \in K$. Since $div(f)$ and D were constructed to have disjoint support, $f(D) \neq 0$, thus $f(D) \in K^\times$. We now define the Tate-Lichtenbaum pairing in the following theorem.

Theorem 3.11 (The Tate-Lichtenbaum Pairing) *Let E be an elliptic curve defined over a field K_0. Let n be a positive integer with $\gcd(char(K_0), n) = 1$. Set $K = K_0(\mu_n)$, $D = \sum_{P \in E} n_P(P)$. The map*

$$\begin{aligned} \langle \cdot, \cdot \rangle : E(K)[n] \times E(K)/nE(K) &\rightarrow K^\times/(K^\times)^n \\ \langle P, Q \rangle &\mapsto f(D) = \textstyle\prod_P f(P)^{n_P} \end{aligned}$$

is called the Tate-Lichtenbaum pairing and satisfies the following properties:

1. *$\langle \cdot, \cdot \rangle$ is linear in each variable*
2. *$\langle \cdot, \cdot \rangle$ is non-degenerate*
3. *$\langle \cdot, \cdot \rangle$ is Galois invariant*

In IV.3.2.2 we will see a version of the pairing which is defined over finite fields of the form $\mathbb{F}_q$ which is where our cryptographic schemes will take place. In IV.3.2 we will

also discuss how to calculate each pairing, give a comparison of the two, and explain the advantages and disadvantages of each.

4 Gröbner Bases

We provide a very brief introduction to Gröbner bases for those who may not be familiar with them. A very good introduction to the subject can be found in [11], with a more advanced treatment in [18].

Gröbner Bases arise in the exploration of polynomial rings defined over a field and their ideals. If we have a polynomial ring defined over a field k in one variable, we can give a complete description of any ideal $I \subset k[x]$. Since k is a field the ring $k[x]$ is a principal ideal domain, and so $I = \langle g \rangle$ for some polynomial $g \in k[x]$. $k[x]$ is also a unique factorization domain(UFD), and so we have access to various algorithms, such as the euclidean algorithm for polynomials, and the extended euclidean algorithm. What happens now if we have $k[x_1, x_2]$? Or $k[x_1, \ldots, x_n]$ for that matter? Do we still have a euclidean algorithm in this setting? The answer is yes, but we need to find a way to order things to be able to systematically deal with our polynomials. As we shall see there are many different ways to produce orderings to do this. We begin with some definitions.

Definition 3.9 *A monomial ordering on $k[x_1, \ldots, x_n]$ is any relation on the set of monomials $x^\alpha, \alpha \in \mathbb{Z}^n_{\geq 0}$ such that*

1. *we have a total ordering on $\mathbb{Z}^n_{\geq 0}$*
2. *if $\alpha > \beta$, then for all $\gamma \in \mathbb{Z}^n_{\geq 0}$, $\alpha + \gamma > \beta + \gamma$*
3. *we have a well-ordering on $\mathbb{Z}^n_{\geq 0}$*

As mentioned we can define several monomial orderings. Here are a few common examples.

Definition 3.10 *Let $\alpha, \beta \in \mathbb{Z}^n_{\geq 0}$.*

1. *Lexicographic Order - We say $\alpha >_{lex} \beta$ if $\alpha - \beta \in \mathbb{Z}^n_{\geq 0}$.*

   We then write $x^\alpha >_{lex} x^\beta$.

2. *Graded Lex Order - We say $\alpha >_{grlex} \beta$ if*

$$|\alpha| = \sum_{i=1}^{n} \alpha_i > |\beta| = \sum_{i=1}^{n} \beta_i \text{ or } |\alpha| = |\beta| \text{ and } \alpha >_{lex} \beta$$

3. *Graded Reverse Lex Order - We say $\alpha >_{grevlex} \beta$ if*

$$|\alpha| = \sum_{i=1}^{n} \alpha_i > |\beta| = \sum_{i=1}^{n} \beta_i \text{ or } |\alpha| = |\beta|$$

   and in $\alpha - \beta \in \mathbb{Z}^n$, the right most entry is negative.

Such an ordering now gives us a way of ordering our polynomials in $k[x_1, \ldots, x_n]$. For the most part our first instinct is to use the lexicographic ordering, but this ordering may not always result in the nicest Gröbner basis for a given ideal $I \in k[x_1, \ldots, x_n]$. We make one last definition before reaching our main subject of this section.

Definition 3.11 *Let $f = \sum_\alpha a_\alpha x^\alpha$ be a non zero polynomial in $k[x_1, \ldots, x_n]$ and let $>$ be a monomial order.*

1. *The multidegree of f is $md(f) = \max\{\alpha \in \mathbb{Z}^n_{\geq 0} \mid a_\alpha \neq 0\}$.*
2. *The leading coefficient of f is $LC(f) = a_{md(f)} \in k$.*
3. *The leading monomial of f is $LM(f) = x^{md(f)}$.*

4. The leading term of f is $LT(f)$ =$LC(f)\times LM(f)$.

We can now define a Gröbner basis.

Definition 3.12 *For any fixed monomial ordering, a finite subset $G = \{g_1, \ldots, g_p\} \subset I$ of an ideal I is said to be a Gröbner basis if*

$$\langle LT(g_1), \ldots, LT(g_p)\rangle = \langle LT(I)\rangle$$

With Gröbner bases we can answer many interesting questions about the polynomial ring $k[x_1, \ldots, x_n]$, including the question about determining the points that are in the variety $\mathbf{V}(f_1, \ldots, f_s)$, that is determine the set of solutions to the polynomial equations

$$f_1(x_1, \ldots, x_n) = \cdots = f_s(x_1, \ldots, x_n) = 0$$

We will see the use of Gröbner bases again when we discuss an Index Calculus attack on Abelian varieties developed by Gaudry in IV.3.6.

5 Resultants

In the discussion of Semaev's attack in IV.3.5, the theory of Resultants will play an important role. Classically resultants are closely tied to the ideas of Elimination theory. In fact the resultant of two polynomials always lies in the first elimination ideal of $\langle f, g\rangle$ [11]. However, the full results of elimination theory are beyond the scope of this thesis. To summarize, given a system of polynomial equations, we can determine a Gröbner basis for the ideal generated by these polynomials and can successively eliminate variables from our set of equations for the Gröbner basis. We

are then essentially solving the system of equations by back-substitution with respect to the fixed ordering. These ideas can be best demonstrated by an example. This example is due to Cox, Little and O'Shea in [11].

Suppose that we want to solve the system of equations

$$\begin{aligned} x^2 + y + z &= 1 \\ x + y^2 + z &= 1 \\ x + y + z^2 &= 1 \end{aligned}$$

If we let $I = \langle x^2 + y + z - 1, x + y^2 + z - 1, x + y + z^2 - 1 \rangle$ and we compute a Gröbner basis with respect to the lex ordering we get the polynomials

$$\begin{aligned} g_1 &= x + y + z^2 - 1 \\ g_2 &= y^2 - y - z^2 + z \\ g_3 &= 2yz^2 + z^4 - z^2 \\ g_4 &= z^6 - 4z^4 + 4z^3 - z^2 \end{aligned}$$

But now the polynomial g_4 is an equation in one variable and we can solve for z. In turn we can then solve for g_2, obtaining solutions for y, and finally for x using g_1. So we eliminated the variables in reverse compared to our ordering.

And now the definition of a resultant.

Definition 3.13 *Given polynomials $f, g \in k[x]$ of positive degree, write them in the form*

$$\begin{aligned} f &= a_0 x^l + \ldots + a_l \\ g &= b_0 x^m + \ldots + b_m. \end{aligned}$$

We then form the Sylvester matrix of f and g with respect to x. The Sylvester matrix is an $(l+m)\times(l+m)$ matrix, with l and m not necessarily equal, of the form

$$Syl(f,g,x) = \begin{pmatrix} a_0 & & & & b_0 & & & \\ a_1 & a_0 & & & b_1 & b_0 & & \\ a_2 & a_1 & \ddots & & b_2 & b_1 & \ddots & \\ & a_2 & \ddots & a_0 & & b_2 & \ddots & b_0 \\ \vdots & & \ddots & a_1 & \vdots & & \ddots & b_1 \\ & \vdots & \ddots & a_2 & & \vdots & \ddots & b_2 \\ a_l & & \vdots & & b_m & & \vdots & \\ & a_l & & \vdots & & b_m & & \vdots \\ & & \ddots & & & & \ddots & \\ & & & a_l & & & & b_m \end{pmatrix}$$

where the entries above and below the coefficients are all zero. The resultant of f and g with respect to x, $Res(f,g,x)$, is the determinant of this matrix.

The ideas and results of Gröbner bases, Elimination theory, and Resultants will be used in the descriptions of later attacks. In particular we will see them again in the GHS Attacks and the techniques of Weil Descent in IV.3.3, The Summation Polynomial attack by Semaev in IV.3.5, and Gaudry's Index Calculus attack on Abelian Varieties in IV.3.6.

6 Algebraic Geometry, Algebraic Groups and Abelian Varieties

Algebraic Geometry is the study of the relationship between geometric objects, such as curves, and polynomial rings which will define these geometric structures. In this section we discuss some key concepts that will help us understand, not only the

relationship between the algebraic and geometric structure, but future attacks that will be proposed on elliptic curve cryptosystems.

The results in this section come from a variety of sources on the subject. For a good exploration of the relationship between the geometry and the algebra we refer the reader to [26] and [37]. The results concerning the algebra come from a variety of source including [2], [39][13], [76], and (related specifically to elliptic curves) [77]. Any results needed from commutative algebra are from [1] and [18].

6.1 Varieties and Dimension

Let K be a field with algebraic closure $\overline{K}$. Recall that the set of points in n-dimensional projective space can be defined as

$$\mathbb{P}^n(\overline{K}) = \{(X_0 : \ldots : X_n) \mid X_i \in \overline{K}, \exists X_i \neq 0 \text{ for some } i\}/\sim$$

where $\sim$ is an equivalence relation given by setting $(X_0 : \ldots : X_n) \sim (Y_0 : \ldots : Y_n)$ iff there exists $\lambda \neq 0 \in \overline{K}$ such that $(X_0 : \ldots : X_n) = \lambda(Y_0 : \ldots : Y_n)$.

For any extension field L such that $K \subset L \subset \overline{K}$, the set of L-rational points

$$\mathbb{P}^n(L) = \{(X_0 : \ldots : X_n) \mid \exists \lambda \neq 0 \in \overline{K}\ \forall i : \lambda X_i \in L\}.$$

This is the set of points fixed by the absolute galois group of L, $\mathrm{Gal}(\overline{K}/L)$.

Suppose now that we have a homogeneous polynomial $f(X_0, \ldots, X_n)$ $\in K[X_0, \ldots, X_n]$, and an ideal $I \subseteq K[X_0, \ldots, X_n]$ with I generated by homogeneous

[13]Hartshorne approaches the ideas of Algebraic Geometry from the point of view of Sheaves and Schemes. We will not introduce any of that here.

polynomials and $I \neq \langle X_0 \ldots, X_n \rangle$. Then the following sets are well defined

$$\begin{aligned} D_f(L) &= \{P \in \mathbb{P}^n(L) \mid f(P) \neq 0\} \\ V_I &= \{P \in \mathbb{P}^n(\overline{K}) \mid f(P) = 0, \forall f \in I\}. \end{aligned}$$

The sets $D_f(L)$ and V_I are the open and closed sets respectively, in the Zariski topology, attached to K in projective space.

Analogously we can define everything over affine space of K: n-dimensional affine space is given by

$$\mathbb{A}^n(\overline{K}) = \{(x_1, \ldots, x_n) \mid x_i \in \overline{K}\}$$

with the set of L-rational points being

$$\mathbb{A}^n(L) = \{(x_1, \ldots, x_n) \mid x_i \in L\}.$$

Similarly one may define $D_f(L)$ and $V_I(L)$ to be the open and closed sets respectively, with respect to the Zariski topology on $\mathbb{A}^n$. However we have a little more structure in $\mathbb{A}^n$. A set $S \subset \mathbb{A}^n$ is closed if there is an ideal $I \in K[x_1, \ldots, x_n]$ with $S = V_I$. Unfortunately the ideal I is not uniquely determined by S. However, we can make a selection for such an ideal. Take the maximal ideal containing such an ideal I, this will be the radical ideal and is defined as

$$\sqrt{I} = \{f \in K[x_1, \ldots, x_n] \mid f^n \in I, \text{ for some } n \in \mathbb{N}\}.$$

Definition 3.14 *Let V be a closed set (in either affine or projective space). V is said to be irreducible if it cannot be written as $V = V_1 \cup V_2$ for two proper closed subsets of V. If V is closed and irreducible then V is said to be a variety (in either*

affine or projective space). A variety V is said to be absolutely irreducible if V is also irreducible in $\overline{K}$, this makes V irreducible over all extensions L of K.

One way to determine if $V \subseteq \mathbb{A}^n$ (or $\mathbb{P}^n$) is a variety is the following.

Lemma 3.2 *[2] A subset $V \subseteq \mathbb{A}^n$ is a variety (affine or projective) iff $V = V_I$ with I a prime ideal in $K[x]$.*

This lemma is a good demonstration of the relationship between the geometry and the algebra; a set of polynomial equations which define a geometric object is irreducible iff the corresponding ideal associated to this set is prime in its corresponding polynomial ring.

One final note before concluding this section is the idea about the dimension of a variety. These ideas will be revisited when we talk about the GHS attack in IV.1.3.

Definition 3.15 *$\mathbb{A}^n$ and $\mathbb{P}^n$ are Noetherian topological spaces, and hence any descending sequence of closed subsets $S_1 \supseteq S_2 \supseteq \ldots$ becomes stationary. This in turn makes the respective polynomial rings noetherian as well. Suppose now that V is a variety. The dimension, $dim(V)$, is defined to be the supremum on the lengths of all chains of distinct irreducible closed subspaces of V[14]. If $dim(V)$ is 1, then V is called a curve.*

[14]A chain is a sequence of containments $V = V_0 \supseteq V_1 \supseteq V_2 \supseteq \ldots \supseteq V_w$. The length of a chain is then the number of subspaces in that sequence.

6.2 Function Fields, Morphisms and Rational Maps

Definition 3.16 *Let V be an affine variety over K, and I the corresponding prime ideal. Then*

$$K[V] = K[x_1, \ldots, x_n]/I$$

is called the coordinate ring of V. We can also form the function field of V denoted $K(V)$; this is simply the quotient field of the coordinate ring $K[V]$.

Definition 3.17 *A morphism φ from $\mathbb{A}^n$ to $\mathbb{A}^m$ is given by an m-tuple of polynomials in $K[x]$, that is*

$$\begin{array}{rcl} \varphi : \mathbb{A}^n & \to & \mathbb{A}^m \\ P & \mapsto & (f_1(P), \ldots, f_M(P)) \end{array}$$

A morphism of varieties from $V \subset \mathbb{A}^n$ to $W \subset \mathbb{A}^m$ is given by the restriction to V from $\mathbb{A}^n$ to $\mathbb{A}^m$ with image in W. We will denote the set of morphisms from one variety to another as $Mor_K(V, W)$, and we will drop K from this when the context of the underlying field is understood.

Suppose now that $\varphi \in \mathrm{Mor}_K(V, W)$, and $f \in K[W]$. The composition of functions induces a morphism on the coordinate rings:

$$\varphi^* : K[W] \to K[V];\ \varphi^*(f) = f \circ \varphi$$

Note that φ^* is injective iff φ is surjective, and φ is injective iff φ^* is surjective.

Definition 3.18 *A rational map is simply the quotient of two functions of $K[V]$ defined on an open subset of V such that the denominator is not zero at the point P.*

Normally when we talk about morphisms and rational maps we will distinguish between the two. When we talk about rational maps, we will abbreviate this by simply calling them maps.

6.3 Abelian Varieties

6.3.1 The Definition

Definition 3.19 *An algebraic group $\mathcal{G}$ over a field K is an absolutely irreducible variety defined over K together with a group structure given by the morphisms*

1. *the addition morphism*
$$m : \mathcal{G} \times \mathcal{G} \to \mathcal{G}$$
2. *the inverse morphism*
$$i : \mathcal{G} \to \mathcal{G}$$
3. *the neutral element, which is a K-rational point*
$$0 \in \mathcal{G}(K)$$

satisfying the following group laws

$$m \circ (Id_{\mathcal{G}} \times m) = m \circ (m \times Id_{\mathcal{G}}) \text{ (associativity)},$$
$$m_{|\{0\}\times\mathcal{G}} = p_2, \text{ and}$$
$$m \circ (i \times Id_{\mathcal{G}}) \circ \delta_{\mathcal{G}} = c_0.$$

where p_2 is the projection of $\mathcal{G} \times \mathcal{G}$ onto the second argument, $\delta_{\mathcal{G}}$ the diagonal map from $\mathcal{G}$ to $\mathcal{G} \times \mathcal{G}$, and c_0 is the map which sends $\mathcal{G}$ to 0.

Definition 3.20 *Projective algebraic groups are Abelian Varieties and are Commutative.*

6.3.2 Homomorphisms, Isomorphisms and Isogenies

Since abelian varieties are a class of varieties we can consider morphisms between them. Let $\mathcal{A}$ and $\mathcal{B}$ be abelian varieties. A remarkable result is that any $\varphi \in \mathrm{Mor}_K(\mathcal{A}, \mathcal{B})$ is actually a homomorphism of groups. That is, if $\oplus$ and $\oplus'$ represent the addition laws on $\mathcal{A}$ and $\mathcal{B}$ respectively, then for all points $P, Q \in \mathcal{A}$, $\varphi(P \oplus Q) = \varphi(P) \oplus' \varphi(Q)$ iff the neutral element of $\mathcal{A}$ is mapped to the neutral element of $\mathcal{B}$. Better still, we can define a translation map,

$$\begin{aligned} t_P : \mathcal{A} &\rightarrow \mathcal{A} \\ Q &\mapsto P \oplus Q \end{aligned}$$

for which we get the following result

Theorem 3.12 *Every morphism from $\mathcal{A}$ to $\mathcal{B}$ is a homomorphism up to the translation map $t_{-(\varphi(0_{\mathcal{A}}))}$.*

The following theorem, from [2], introduces some fundamental concepts and some important notation.

Theorem 3.13 *Let $\varphi \in Hom_K(\mathcal{A}, \mathcal{B})$.*

1. *The image, $Im(\varphi)$, of φ is a subvariety of $\mathcal{B}$, which becomes an abelian variety by restricting the addition law of $\mathcal{B}$ to $Im(\varphi)$.*
2. *The kernel, $ker(\varphi)$, of φ is closed in $\mathcal{A}$. It contains a maximal absolutely irreducible subvariety, $ker(\varphi)^0$ containing $0_{\mathcal{A}}$. $ker(\varphi)^0$ is an abelian subvariety of $\mathcal{A}$ and is called the connected component of unity of $ker(\varphi)$.*
3. *The dimension of $\mathcal{A}$ is*

$$dim(Im(\varphi)) + dim(ker(\varphi)^0) = dim(\mathcal{A})$$

Definition 3.21 *1. The map φ is said to be an isogeny if $Im(\varphi) = \mathcal{B}$ and $ker(\varphi)$ is finite, while*

2. A morphism, φ is said to be an isomorphism if there exists a morphism $\psi \in Hom_K(\mathcal{B}, \mathcal{A})$, such that $\varphi \circ \psi = id_{\mathcal{A}}$ and $\psi \circ \varphi = id_{\mathcal{B}}$.

Granting Theorem 3.13, one can easily show that the following holds true.

Lemma 3.3 *$\varphi \in Hom_K(\mathcal{A}, \mathcal{B})$ is an isogeny iff $dim(\mathcal{A}) = dim(\mathcal{B})$ and $dim(ker(\varphi)^0) = 0$.*

Before we link these results to elliptic curves, we present one last definition.

Definition 3.22 *An abelian variety is said to be simple if it does not contain a proper abelian subvariety.*

Notice here that if we look at everything at the level of elliptic curves (abelian varieties of dimension one), we get the following result, which may seem more familiar.

Theorem 3.14 *[77] Let E_1 and E_2 be elliptic curves defined over K. If $\varphi : E_1 \to E_2$ is a non-constant isogeny defined over K, then φ induces an injection $\varphi^* : \overline{K}(E_2) \to \overline{K}(E_1)$ which fixes K. Also $\overline{K}(E_1)$ is a finite extension field of $\varphi^*(\overline{K}(E_2))$, and the degree of φ is the degree of the extension $[\overline{K}(E_1) : \varphi^*(\overline{K}(E_2))]$. It follows that if φ is a map of degree one then φ is an isomorphism.*

4 Attacking the Elliptic Curve Discrete Logarithm Problem

1 Introduction

We now want to focus on how to solve the Elliptic Curve Discrete Logarithm Problem(ECDLP). Recall that the ECDLP can be stated as follows:

Definition 4.1 *Let $E(\mathbb{F}_q)$ be an elliptic curve defined over $\mathbb{F}_q$, where $q = p$ or $q = p^m$ for some $m \in \mathbb{N}$. Suppose that $P \in E(\mathbb{F}_q)$, and that $Q \in E(\mathbb{F}_q)$ is such that $Q \in \langle P \rangle$, ie. the subgroup generated by P. Determine the unique integer n, such that $Q = [n]P$.*

Attacks on the ECDLP can be split into two main categories: attacks that work in the general setting regardless of properties of a given elliptic curve, and attacks that use specific properties of the elliptic curve to develop a different approach. Consequently, we have subdivided this chapter into two parts. The first part deals with the general attacks, while the second part will deal with more advanced attacks, including Pairing attacks, the method of Weil Descent, a discussion of how the Index Calculus fails to translate to the situation of elliptic curves, and the attack that was developed from this idea. Before continuing onto the next section, the reader should be reminded of

the attacks on the DLP. This next section contains the analogies between the two problems, and hence one reason for introducing the attacks on the DLP.

Also within the body of this text the reader will find `Pari/GP` code, which one could use to calculate results for a specific instance of the ECDLP. For those unfamiliar with `Pari/GP`, an appendix at the end of this thesis explains some of the functions that were used in the programs.

2 General Attacks

2.1 Exhaustive Search

Of course one way to attack the ECDLP is to perform an exhaustive search when the points P and Q are given. Since, in practice, P is chosen to have significantly large order, this then makes the exhaustive search infeasible.

Algorithm 4.1 Exhaustive Search

```
1: print("Please enter a Prime"); p=input();
2: print("Please enter coefficients for an elliptic curve");
3: a=input();b=input();c=input();d=input();e=input();
4: Ep=ellinit([Mod(a,p),Mod(b,p),Mod(c,p),Mod(d,p),Mod(e,p)]);
5: print("Please enter points P and Q for which you wish to solve the
   ECDLP");
6: P=input();Q=input();
7: print("Please enter the order of the generator"); n=input();
8: for(i=1,n,R=ellpow(Ep,P,i);
9: if(R==Q, print("The answer is:" i)); break(1))
```

2.2 Baby-Step, Giant-Step Algorithm

Similar in nature to the setting of the DLP, this attack uses a combination of computational power and storage in an attempt to solve the ECDLP. Let $E(\mathbb{F}_q)$ be an

elliptic curve with generator P. Suppose that P has order n, and let $Q \in \langle P \rangle$. Suppose that we want to solve $Q = [k]P$. Set $m = \lceil \sqrt{n} \rceil$ and compute $[m]P$. We now make a list of $[i]P$ for $0 \leq i < m$, and store this list. We can now compute $Q - [j]([m]P)$, for $0 \leq j \leq m-1$ until we have found a match from the list that we have stored. Once we have a match we then have the following:

$$\begin{aligned} [i]P &= Q - [j]([m]P) \qquad \text{hence,} \\ Q &= [i]P + [j]([m]P). \end{aligned}$$

Therefore we have solved the ECDLP since $k \equiv i + jm \mod n$.

Again this attack takes at most $\sqrt{n}$ operations and stores $\sqrt{n}$ values in a list to check for a match. Thus the expected running time of this algorithm is $O(\sqrt{n})$ [87]. Notice here that we can make this slightly more efficient. When we compute the points $[i]P$, we only need to store half of these values. In other words, we only have to compute $[i]P$ for $0 \leq i \leq \frac{m}{2}$, and then we can check if $Q - [j]([m]P) = \pm[i]P$[87].

Algorithm 4.2 Baby-Step, Giant-Step

```
 1: print("Please enter a Prime"); p=input();
 2: print("Please enter coefficients for an elliptic curve");
 3: a=input();b=input();c=input();d=input();e=input();
 4: Ep=ellinit([Mod(a,p),Mod(b,p),Mod(c,p),Mod(d,p),Mod(e,p)]);
 5: print("Please enter point P and Q for which you wish to solve the
    ECDLP");
 6: P=input();Q=input();
 7: print("Please enter the order of the generator"); n=input();
 8: m=ceil(sqrt(n));
 9: R=ellpow(Ep,P,-m);
10: for(i=0,m,W=ellpow(Ep,P,i);
11: for(j=0,m-1,Z=elladd(Ep,Q,ellpow(Ep,R,j));
12: if(Z==W, print("The answer is:" Mod(i+j*m,p));break(2))))
```

Note that there are some issues here with storing this list. This must be done

properly so that we avoid too many table look-ups. The algorithm as presented here would require roughly n table look-ups to find a match and would no longer have an expected running time of $O(\sqrt{n})$. To avoid this, the stored list must be sorted and more sophisticated searching techniques must be used.

Example: Let E be the elliptic curve $y^2 = x^3 + 130x + 565$ defined over $\mathbb{F}_{719}$. Suppose that $P = (107, 443)$ and that $Q = (608, 427)$, and we want to determine the unique integer λ such that $Q = [\lambda]P$. Note here that it can be shown that P has order 699. Using the Baby-Step, Giant-Step method we first compute $m = \lceil \sqrt{699} \rceil = 27$, and calculate $[m]P = [27](107, 443) = (635, 361)$. We now create and store a list. To avoid this we could sort the stored list, by x-coordinate say, so that when a new point is generated we know where in the list to look to find a match. for all values of $[i]P$ for $0 \leq i < m$.

Now we can calculate $Q - [j]([m]P)$ for $0 \leq j \leq m - 1$ until we find a match in the table to the right. So we compute

$$\begin{aligned} Q - [0]P &= Q \\ Q - [1]P &= (24, 637) \\ Q - [2]P &= (551, 578) \\ Q - [3]P &= (642, 619) \\ &\vdots \\ Q - [15]P &= (596, 564) \\ Q - [16]P &= (597, 529) \\ Q - [17]P &= (406, 409) \\ Q - [18]P &= (106, 576) \end{aligned}$$

i	$[i]P$	i	$[i]P$
0	$\mathcal{O}$	1	$(107, 443)$
2	$(303, 175)$	3	$(460, 25)$
4	$(233, 580)$	5	$(715, 585)$
6	$(631, 182)$	7	$(106, 576)$
8	$(220, 206)$	9	$(325, 326)$
10	$(575, 481)$	11	$(98, 415)$
12	$(213, 106)$	13	$(434, 522)$
14	$(51, 162)$	15	$(425, 144)$
16	$(468, 681)$	17	$(234, 497)$
18	$(392, 319)$	19	$(44, 294)$
20	$(314, 300)$	21	$(670, 460)$
22	$(142, 478)$	23	$(471, 631)$
24	$(404, 91)$	25	$(598, 565)$
26	$(256, 690)$		

Table 4.1: Data for Baby-Step, Giant-Step Attack

At which point we can stop since we realize that we have a match. Hence we find that $\lambda \equiv i + jm \equiv 7 + 18 \times 27 \equiv 493 \mod 699$ as required.

2.3 Pollard's ρ-Method

Let $E(\mathbb{F}_q)$ be an elliptic curve and $P \in E(\mathbb{F}_q)$. Suppose that P has order n, where n is prime, and let $Q \in \langle P \rangle$. Suppose that we want to solve $Q = [k]P$. In this attack we will attempt to find distinct pairs of integers (a, b) and (a', b') modulo n such that $[a]P + [b]Q = [a']P + [b']Q$. Rearranging this we can obtain a solution for k, namely $k \equiv (a - a')(b' - b)^{-1} \mod n$. (Note that since n was assumed here to be prime the difference of b and b' can be inverted).

One method for finding these pairs of integers is to simply select $a, b \in [0, n-1]$ uniformly at random, compute the point $[a]P + [b]Q$, and then store the triple $(a, b, [a]P + [b]Q)$. We continue to generate pairs (a, b) uniformly at random and check these against all previously stored triples until we find a pair (a', b') with $[a']P + [b']Q$ where $(a, b) \neq (a', b')$. When this happens we have solved the ECDLP and as mentioned above, we can rearrange $[a]P + [b]Q = [a']P + [b']Q$ as $[a - a']P = [b' - b]Q = [b' - b]([k]P)$, and thus $k \equiv (a - a')(b' - b)^{-1} \mod n$.

Again as in the setting of the DLP, the birthday problem governs the expected running time of this algorithm. This first method gives an expected running time of $O(\sqrt{\frac{\pi n}{2}})$ [38], but unfortunately requires approximately $O(\sqrt{\frac{\pi n}{2}})$ amount of storage for the triples that we have computed.

A second approach that has roughly the same running time, but uses less storage is also known. Instead of storing a list of triples, we define a function $f : \langle P \rangle \to \langle P \rangle$ so

that for any $X \in \langle P \rangle$ and $a, b \in [0, n-1]$ with $X = [a]P + [b]Q$, we can easily compute $f(X) = X'$ and $a', b' \in [0, n-1]$ with $X' = [a']P + [b']Q$. One way to define such a function is to partition $\langle P \rangle$ into L sets of roughly equal size, say $\{S_1, S_2, \ldots, S_L\}$. We define a second function H so that $H(X) = j$ if $X \in S_j$. Then $a_j, b_j \in [0, n-1]$ are chosen uniformly at random for $1 \leq j \leq L$. Now our function $f : \langle P \rangle \to \langle P \rangle$ is defined by

$$f(X) = X + [a_j]P + [b_j]Q, \text{ where } j = H(X).$$

So, if $X = [a]P + [b]Q$, then $f(X) = X' = [a']P + [b']Q$ where $a' = a + a_j \mod n$ and $b' = b + b_j \mod n$ [38]. This then determines a sequence of points in $\langle P \rangle$. Since $\langle P \rangle$ is finite we will eventually obtain a collision, thus obtaining our pairs of integers (a, b) and (a', b'), and so enabling us to solve the ECDLP.

As mentioned, this approach has a similar running time to the first, but requires less storage, since we are no longer required to store ordered triples in order to find a collision.

Note that to ensure that a match has been made, we could modify this program and simply use n in place of m. Although we expect to obtain a match within m steps thanks in part to the birthday problem, there is nothing guaranteeing a match in `ceil(sqrt(n))` steps[15].

[15] Again a storage issue arises in this algorithm. See the note following Algorithm 4.2.

Algorithm 4.3 Pollard's Rho

```
 1: print("Please enter a Prime"); p=input();
 2: print("Please enter coefficients for an elliptic curve");
 3: a=input();b=input();c=input();d=input();e=input();
 4: Ep=ellinit([Mod(a,p),Mod(b,p),Mod(c,p),Mod(d,p),Mod(e,p)]);
 5: print("Please enter point P and Q for which you wish to solve the
    ECDLP");
 6: P=input();Q=input();
 7: print("Please enter the order of the generator"); n=input();
 8: m=ceil(sqrt(n));
 9: va=vector(m,X,random(n));
10: vb=vector(m,Y,random(n));
11: R=vector(m,i,elladd(Ep,ellpow(Ep,P,va[i]),ellpow(Ep,Q,vb[i])));
12: for(j=1,m-1,
13: for(k=j+1,m,
14: if(R[j]==R[k],print(Mod((va[j]-va[k])*((vb[k]-vb[j])^(-1)),n));
15: break(2))))
```

2.3.1 Speeding up Pollard's ρ-Method

There are two ways that one can improve on the expected running time of Pollard's ρ-Method. The first method simply involves a parallelized attack. Suppose that we have M processors available at our disposal. Recall that above we have created a sequence of points in $\langle P \rangle$. Denote this sequence as $\{X_i\}_{i \geq 0}$, where $X_i \in \langle P \rangle$. Feed each processor a random starting point X_0, and let them all use the same function f, similar to f defined above, to compute further members of the sequence $\{X_i\}_{i \geq 0}$. If two different processors collide, then the two sequences will be identical afterwards, which is clear from the way we have defined our function f. The trick is to determine an efficient method of finding a collision. One method is to establish a *distinguishing property*[16] of a point. When this distinguished point is hit in the sequence, the

[16] An example of a *distinguishing property* could be that the leading t bits of a point are all zero say[38].

processor can send the information back to the central server where it can be stored. When the server receives the distinguished point a second time, it can compute the discrete logarithm and terminate the M processors. Denote the proportion of points in $\langle P \rangle$ that have this distinguishing property by θ. The expected number of steps per processor before a collision occurs is $\frac{1}{M}\sqrt{\frac{\pi n}{2}}$, and a distinguished point is expected after $\frac{1}{\theta}$ steps. Thus the total expected running time before a collision of distinguished points is expected is $\frac{1}{M}\sqrt{\frac{\pi n}{2}} + \frac{1}{\theta}$ [38].

The second method to speed up Pollard's ρ-Method is by using automorphisms. Let $\psi : \langle P \rangle \to \langle P \rangle$ be an automorphism of groups, where $P \in E(\mathbb{F}_q)$ has order n. Suppose that ψ has order t, in other words, t is the smallest positive integer such that $\psi^t(R) = R$ for all $R \in \langle P \rangle$. In this way we can define an equivalence relation on $\langle P \rangle$ by the following. $R_1 \sim R_2$ iff $R_1 = \psi^j(R_2)$ for some $j \in [0, t-1]$. We now define the equivalence class $[R]$ containing $R \in \langle P \rangle$, simply as powers of $\psi(R)$, thats is,

$$[R] = \{R, \psi(R), \psi^2(R), \ldots, \psi^{l-1}(R)\},$$

where l is the smallest positive divisor of t such that $\psi^l(R) = R$.

We are attempting here to define our function f on the equivalence classes of $\langle P \rangle$ rather than just points, to speed up our calculations. To do this we have to choose representatives for each class $[R]$, denote this by $\overline{R}$. We can then define a new function on our representatives by setting $g(R) = \overline{f(R)}$. So suppose we know an integer $\lambda \in [0, n-1]$, such that $\psi(P) = \lambda P$. Since ψ is an automorphism, $\psi(R) = \lambda R$ for all $R \in \langle P \rangle$. Thus if we know integers a and b such that $X = [a]P + [b]Q$, then we can efficiently determine integers a' and b' such that $\overline{X} = [a']P + [b']Q$. This is since

if we have that $\overline{X} = \psi^j(X)$, then $a' = \lambda^j a \mod n$ and $b' = \lambda^j b \mod n$ [38]. We can now use g and the equivalence classes in the parallelized version of the algorithm above and obtain a speed up. If each equivalence class has size approximately t, then we've reduced the search space by a factor of approximately $\frac{n}{t}$, thus making the expected running time of this algorithm $O(\frac{1}{M}\sqrt{\frac{\pi n}{2t}} + \frac{1}{\theta})$ [38].

Example: Here we present an example of the speed-up for Pollard's Rho. Let $\psi(P) = -P$, that is the automorphism which sends P to its negative, $-P$. Clearly ψ has order two and thus the expected running time for Pollard's Rho becomes $O(\frac{\sqrt{\pi n}}{2})$.

As an example E be the elliptic curve $y^2 = x^3 + 130x + 565$ defined over $\mathbb{F}_{719}$. This time suppose that $P = (312, 90)$ and that $Q = (475, 662)$. We want to determine the unique integer λ such that $Q = [\lambda]P$, note here that it can be shown that P has prime order 233.

a	b	$[a]P + [b]Q$
179	123	$(47, 297)$
207	134	$(168, 508)$
152	50	$(210, 129)$
118	199	$(119, 665)$
51	55	$(649, 199)$
70	104	$(47, 422)$
207	99	$(305, 430)$
57	76	$(140, 298)$
53	205	$(414, 453)$
210	16	$(293, 81)$
137	85	$(133, 221)$
135	172	$(501, 547)$
180	171	$(22, 542)$
113	192	$(671, 569)$
160	77	$(280, 500)$
207	28	$(463, 17)$
231	89	$(260, 296)$
62	151	$(284, 505)$
181	99	$(316, 540)$
110	130	$(588, 453)$

Table 4.2: Data for Pollard's Rho Attack

To solve the ECDLP using the speed-up of Pollard's-ρ, we choose uniformly at random $a, b \in [0, 232]$, calculate $R = [a]P + [b]Q$ and store the triple (a, b, R) until such time we encounter a second triple (a', b', R') such that $R = R'$ or $R = -R'$. From the birthday problem we expect we should only need to calculate $\frac{\sqrt{\pi n}}{2} = \frac{\sqrt{232\pi}}{2} = 14$ such triples before a match is obtained, instead of roughly 20 such triples using the regular Pollard's Rho.

We notice that after only calculating six of such triples that we have matched our

x-coordinate and can solve the ECDLP.

Hence we have that $[179](312, 90)+[123](475, 662) = -([70](312, 90)+[104](475, 662))$ $\iff$ $[179 + 70](312, 90) = -([104 + 123])(475, 662)$ $\iff$ $(-[104 + 123])^{-1}[179 + 70](312, 90) = (475, 662)$, which in turn allows us to solve for λ. Hence we have that $\lambda \equiv (-[104 + 123])^{-1}[179 + 70] \mod 233$ and thus $\lambda \equiv 158 \mod 233$ which gives us the solution to the ECDLP in this case, as required.

2.4 Pollard's λ-Method

As in II.1.4 one can describe this algorithm in terms of a tame kangaroo attempting to catch a wild kangaroo. If the solution to the ECDLP is known to lie in a certain interval, say $[a, b] \subset [0, n]$ where n is the order of the subgroup generated by P on a curve $E(\mathbb{F}_q)$ for which the instance of the ECDLP is attempting to be solved, then the setup is entirely similar to II.1.4. Instead we focus our attention on the parallelized version of the algorithm and provide immediate speedups of the original.

In the tradition of Pollard's original setup and as in [86], we describe the parallelized version involving kangaroos. Instead of simply having one tame and one wild kangaroo, we now have two herds of kangaroos, a wild herd and a tame herd. Suppose that we want to employ M processors in attempting to solve the ECDLP. We launch $\frac{M}{2}$ tame kangaroos from known starting points, and $\frac{M}{2}$ wild kangaroos from unknown starting points. Whenever a kangaroo lands on a distinguished point, we store this value in a list that is common to all processors. With each distinguishing point we have to record also which type of kangaroo landed on this point, wild or tame, along with its distance traveled from its original staring point. Now if kangaroos of the

same type land on the same distinguishing point then it is clear that they will follow the same path from that point on. If this is the case then we can alter the path of one of the kangaroos by multiplying by a small random distance. If the kangaroos are of a different type then we can trace back the jumps and subtract the distance to obtain our solution for the ECDLP. It can be shown, as in [86], that this method of parallelization results in a linear speedup in the number of processors, namely M in this case. Thus the expected number of steps for this parallelized version to solve an instance of the ECDLP is then $O(\frac{\sqrt{\frac{\pi n}{2}}}{M})$ where M is the number of processors being employed.

2.5 The Pohlig-Hellman Method

The setup for this attack is similar to the setup in the case of the DLP. Suppose that we are given an elliptic curve $E(\mathbb{F}_q)$, a point $P \in E(\mathbb{E}_q)$ of order n and $Q \in \langle P \rangle$. We again want to solve for the unique integer k such that $Q = [k]P$. Suppose further that we know the factorization of n, say $n = \prod_{i=1}^{r} l_i^{e_i}$, where each l_i is prime. Similar to the situation in the case of the DLP, we will now attempt to solve for k by reducing the problem to solve for values of $k_i \equiv k \mod l_i^{e_i}$ for $0 \leq i \leq r$, which gives us a

system of congruences modulo each prime l_i, namely

$$\begin{aligned} k &\equiv k_1 \mod l_1^{e_1} \\ k &\equiv k_2 \mod l_2^{e_2} \\ k &\equiv k_3 \mod l_3^{e_3} \\ &\vdots \\ k &\equiv k_r \mod l_r^{e_r}. \end{aligned} \tag{4.1}$$

The Chinese Remainder Theorem guarantees the existence of a unique solution, namely k.

Let's take a closer look at how this works. For the moment fix a prime say $l_1^{e_1}$. We compute k_1 as follows. We write the base-l_1 representation of k_1,

$$k_1 \equiv a_0 + a_1 l_1 + a_2 l_1^2 + \ldots + a_{e_1-1} l_1^{e_1-1} \mod l_1^{e_1}, \text{ where each } a_i \in [0, l_1 - 1]. \tag{4.2}$$

We begin by computing a list of small values for each prime divisor l_i of n. Set $T_i = \{[j]([\frac{n}{l_i}]P) : 0 \leq j \leq l_i - 1\}$. We will look for a match with these points and values that we will determine below. When we find a match we have solved for a given coefficient in the base-l_1 expansion of k. We can now compute the following,

$$\begin{aligned} [\frac{n}{l_1}]Q &= [\frac{n}{l_1}]([a_0 + a_1 l_1 + \ldots + a_{e_1-1} l_1^{e_1-1}]P) \\ &= [a_0][\frac{n}{l_1}]P + ([a_1 + a_2 l_1 + \ldots])[n]P = [a_0][\frac{n}{l_1}]P. \end{aligned}$$

Thus we can now look in our list T_1, find the matching point in the list and read off the coefficient a_0.

To solve for the next coefficient, a_1, we have to change our starting point which can be easily done. Since we have already solved for a_0 we can use it and set $Q_1 =$

$Q - [a_0]P$ then perform the above calculation using Q_1 instead, and shifting by the proper quantity to isolate for a_1. If we multiply (4.2) by $\frac{n}{l_1^2}$, after a_0 has been removed this will then give us

$$[\frac{n}{l_1^2}]Q_1 = ([a_1 + a_2 l_1 + \ldots])[\frac{n}{l_1}]P = [a_1]([\frac{n}{l_1}]P),$$

and again we look in our list T_1 for a matching solution. This then gives us a result for a_1. We continue in this way until we have solved for each coefficient in the base-l_1 expansion of k_1. We then continue and solve for each k_i in the same manner. When this is done we solve the system as in (4.1) and recover the original value of k in our original problem $Q = [k]P$, thus solving the ECDLP.

The expected running time of this algorithm is $O(\sqrt{l'})$ [87], where l' is the largest prime divisor of n. In practice this attack becomes infeasible when n has a large prime divisor. If this is the case, it then becomes difficult to make and store the list T to find matches, let alone attempting to solve for k' in its base-l' expansion.

Example: Let E be the elliptic curve $y^2 = x^3 + 130x + 565$ defined over $\mathbb{F}_{719}$. Suppose that $P = (107, 443)$ and that $Q = (608, 427)$. Now we want to determine the unique integer λ such that $Q = [\lambda]P$, note here that it can be shown that P has order 699, which factors as $699 = 3 \times 233$.

Using the Pohlig-Hellman attack, we need to compute $\lambda \mod 3$ and $\lambda \mod 233$, we will them obtain a unique solution by using the CRT.

$\lambda \mod 3$: We start by computing our list

$$T = \{j([\frac{699}{3}]P) \mid 0 \leq j \leq 2\} = \{\mathcal{O}, (24, 82), (24, 637)\}.$$

and we compute $[\frac{699}{3}]Q = [\frac{699}{3}](608, 427) = (24, 82)$. We now appeal to T to determine

a match and we find that $\lambda \equiv 1 \mod 3$.

$\lambda \mod 233$: We have a much larger list to compute this time.

$$T = \{j([\frac{699}{233}]P) \mid 0 \leq j \leq 222\} = \{j([3]P) \mid 0 \leq j \leq 222\}.$$

We do not list these result here, but include them at the end of this thesis in an appendix for the interested reader. Calculating $[\frac{699}{233}]Q = [3]Q = (306, 52)$, we find that this matches with entry 27 in our list T. This then yields $\lambda \equiv 27 \mod 233$.

Using the CRT on the system of congruences

$$\begin{aligned} \lambda &\equiv 1 \mod 3 \\ \lambda &\equiv 27 \mod 233 \end{aligned}$$

we find that we obtain the unique solution $\lambda \equiv 493 \mod 699$ as required.

2.6 Conclusions

The following table summarizes the expected running time of our general attacks.

Attack	Expected Running Time	Speed Up
Exhaustive Search	$O(n)$	
Baby-Step, Giant-Step	$O(\sqrt{n})$	
Pollard's ρ	$O(\sqrt{\frac{\pi n}{2}})$	$O(\frac{1}{M}\sqrt{\frac{\pi n}{2}} + \frac{1}{\theta})$
		$O(\frac{1}{M}\sqrt{\frac{\pi n}{2t}} + \frac{1}{\theta})$
Pollard's λ	$O(\sqrt{\frac{\pi n}{2}})$	$O(\frac{\sqrt{\frac{\pi n}{2}}}{M})$
Pohlig-Hellman	$O(\sqrt{l'})$	

Table 4.3: Expected Running Times of the General Attacks on the ECDLP

All of the general attacks on the ECDLP are expected to run in fully exponential time. The best attacks out of the above, are Pollard's ρ-Method with its speedups, and the Pohlig-Hellman algorithm when the factorization of n is known to be composed of small primes. The best known algorithm for a general purpose attack is known to be a combination of Pollard's ρ and the Pohlig-Hellman attack which runs in $O(\sqrt{p})$, where p is the largest prime divisor of n [38].

Recall that at the outset of this document we mentioned three cases in which the ECDLP can be easily solved, namely,

1. If $\#E(\mathbb{F}_p) = p+1$ (the supersingular case) then the ECDLP can be reduced to the DLP on the multiplicative group of the finite field with p^k elements.

2. If $\#E(\mathbb{F}_p) = p$ (the anomalous case) then the ECDLP can be reduced to simple addition in $\mathbb{F}_p$, essentially by lifting the curve modulo p^2.

3. If $\#E(\mathbb{F}_p)$ is divisible by only small primes, then one can use the Pohlig-Hellman method which solves the problem in time $O(\sqrt{p'})$, where p' is the largest prime divisor of $E(\mathbb{F}_p)$.

We have just seen the first attack that is possible in this setting. The Pohlig-Hellman Method is an efficient attack as long as $\#E(\mathbb{F}_p)$ is divisible by small primes. In the next section, we discuss the attacks involving Supersingular curves and Anomalous curves, and how to avoid these attacks.

3 Specialized Attacks

3.1 Anomalous Curves

We now begin to use the properties of elliptic curves to help in our quest to solve instances of the ECDLP. Recall that the Hasse-Weil theorem gives us an approximation for the number of rational points on an elliptic curve, that is $|\#E(\mathbb{F}_q)| \leq q+1+2\sqrt{q}$. Using Schoof's algorithm or one of its variants we can compute explicitly the number of rational points on a curve. It turns out that this result is a simple classification of anomalous curves.

Definition 4.2 *An Elliptic Curve is said to be anomalous if* $\#E(\mathbb{F}_q) = q$.

Attacks on these curves were developed independently by Satoh and Araki in [68], Semaev in [72] and Smart in [82]. Each version of the attack uses different ideas to compute the discrete logarithm and hence are all worth exploring. However, we will explore only the attacks presented by Smart and Semaev, the main reason being that these attacks have a similar running time and are quicker than the attack described by Satoh and Araki in [68]. Although all algorithms yield polynomial running times, the algorithm in [68] is $O((\log p)^3)$ compared to algorithms that run in $O(\log p)$ presented in [72] and [82]. The attack by Smart will be discussed in detail while the attack by Semaev will be explained but some of the details will be omitted. We will also make use of the following lemma which is taken from [77]:

Lemma 4.1 (Hensel's lemma) *Let R be a ring which is complete with respect to some ideal $I \subset R$, and let $F(x) \in R[x]$ be a polynomial. Suppose that $a \in R$ satisfies,*

for some $n \geq 1$,

$$F(a) \in I^n \text{ and } F'(a) \in R^\times.$$

Then for any $\alpha \in R$ *with* $\alpha \equiv F'(a) \mod I$, *the sequence*

$$w_0 = a,\ w_{m+1} = w_m - \frac{F(w_m)}{\alpha}$$

converges to an element $b \in R$ *such that*

$$F(b) = 0 \text{ and } b \equiv a \mod I^n.$$

If further R *is an integral domain then* b *is uniquely determined.*

This lemma will enable us to perform a lift of an elliptic curve, a process which we describe below.

Let E be an elliptic curve defined over a prime field $\mathbb{F}_p$, with $\#E(\mathbb{F}_p) = p$. Suppose that P and Q are points on E such that $P = [m]Q$ for some integer m, and we wish to find a solution for m. What we would like to do is apply a map that is a homomorphism from $E(\mathbb{F}_p)$ into a group where solving the ECDLP would be easier, say $\mathbb{F}_p^+$. Unfortunately we cannot immediately apply such a map over $\mathbb{F}_p$, however we can apply such a map for curves defined over $\mathbb{Q}_p$[17].

First we compute a lift of the original curve E to a curve $\mathcal{E}$ defined over $\mathbb{Q}_p$, that takes points P and Q to points $\mathcal{P}$ and $\mathcal{Q}$ respectively, with the condition that upon reduction modulo p the result returns E. The above lift can be done using Hensel's lemma. If $\mathcal{P} = (x, y)$ then upon reduction modulo p, $P = (x, y')$ where P and $\mathcal{P}$

[17]These are commonly referred to as the p-adic numbers. An excellent introduction to the p-adic numbers can be found in [2].

share the same x-coordinate and y' is computed via Hensel's lemma. When we do this we have the following:

$$Q - [m]P = R \in E_1(\mathbb{Q}_p)$$

where $E_1(\mathbb{Q}_p)$ is the kernel of the map $\phi : E(\mathbb{Q}_p) \to E(\mathbb{F}_p)$ [82]. Also note that

$$E_0(\mathbb{Q}_p)/E_1(\mathbb{Q}_p) \cong E(\mathbb{F}_p) \text{ and } E_1(\mathbb{Q}_p)/E_2(\mathbb{Q}_p) \cong E(\mathbb{F}_p^+)$$

where $E_n(\mathbb{Q}_p) = \{P \in E(\mathbb{Q}_p) \mid \nu(x(P)) \leq -2n\} \cup \{\mathcal{O}\}$ and $\nu(x(P))$ denotes the p-adic valuation on the x-coordinate of the point P.

Since the groups $E(\mathbb{F}_p)$ and $\mathbb{F}_p^+$ have the same order by assumption we obtain the following equation

$$[p]Q - [m]([p]P) = [p]R \in E_2(\mathbb{Q}_p).$$

We then apply the p-adic elliptic logarithm ψ_p, to each term in the equation above, which are all points in $E_1(\mathbb{Q}_p)$ and hence well defined. We thus obtain

$$\psi_p([p]Q) = m\psi_p([p]P) = \psi_p([p]R) \equiv 0 \mod p^2.$$

However since $\psi_p \equiv \frac{-x}{y} \mod p^2$ when $(x, y) \in E_1(\mathbb{Q}_p)$ [82], we have that

$$m \equiv \frac{\psi_p([p](Q))}{\psi_p([p]P)} \mod p.$$

Thus we have found m and have solved the ECDLP in this case.

The governing step in this algorithm is to simply compute $[p]P$ and $[p]Q$ which takes $O(\ln p)$ steps [82]. Note that this is somewhat of a randomized algorithm. There is a possibility that when computing an arbitrary lift of the points P and Q

at the outset of the algorithm, that we have actually computed a canonical lift[18] of the original points and that the method above will not work. The likelihood of this happening is $\frac{1}{p}$ [82], which in practice is insignificant. In the case that this does happen we can simply compute another lift of the points P and Q and the original curve E.

The approach presented in [72] by Semaev is similar in nature and also produces an algorithm that runs in $O(\ln p)$. The difference here is that Semaev uses the theories behind divisors to produce a map from the group of points on the elliptic curve in question to a multiplicative group of an extension of $\mathbb{F}_{q^k}$ where k should be small. There are two cases to consider when doing this.

1. Suppose that the subgroup generated by P produces a subgroup of order m with $\gcd(m, p) = 1$. Then it is the case that $\langle P \rangle$ is isomorphic to a subgroup in the extension $\mathbb{F}_{q^k}$ with $q^k \equiv 1 \mod m$. Elements in the isomorphism of $\langle P \rangle \to \mathbb{F}_q^\times$ can be easily determined and take no more then $O(\ln m)$ steps [72]. When k is small we then have a effective algorithm for computing the ECDLP.

2. If on the other hand $\gcd(m, p) \neq 1$, this approach will not work. We can however do the following. Set $m = p^s m_1$ with $s > 0$, and $\gcd(m_1, p) = 1$. In this case we can then do the above reduction with m replaced by m_1 and determine the extension $\mathbb{F}_{q^k}$ with minimal k such that $q^k \equiv 1 \mod m_1$.

[18] A canonical lift from E to $\mathcal{E}$ is one that $\mathcal{E}$ reduced mod p yields E and that the respective endomorphism rings are isomorphic. A canonical lift produces no information about the ECDLP here.

Since an elliptic curve E is isomorphic to the quotient of the group of divisors of degree zero by the subgroup of principal divisors, we can write a point Q as $D_Q = \sum n_T T$, say for example $D_Q = (Q) - (\mathcal{O})$. Further if Q is an element of the subgroup generated by P then pD_Q is a principal divisor that we can denote as $div(f_Q) = pD_Q$ for some function f_Q on E.

Semaev further goes on to prove the following lemmas which will be stated here, the proofs can be found in [72].

Lemma 4.2 *Let f be a function on E such that $div(f) = pD$ for some principal divisor D. Let $f' = df/dx$ be the derivative with respect to x. Then $div(f') = div(f) - div(y)$.*

Lemma 4.3 *The map*

$$\phi \quad : \quad \langle P \rangle \rightarrow \mathbb{F}_q$$

$$\phi(Q) \quad = \quad \begin{cases} div(f'_Q/f_Q)(R) & \text{if } Q \neq \mathcal{O}, \\ 0 & \text{if } Q = \mathcal{O}. \end{cases}$$

is a well defined isomorphic embedding of $\langle P \rangle$ into $\mathbb{F}_q$.

The third lemma is simply a statement of the expected time needed to evaluate the function f'_Q/f_Q at a point R in $\mathbb{F}_q$, which takes no more then $O(\ln p)$ operations.

With all three of these lemmas we can see that to determine an integer n such that $Q = [n]P$ in $E(\mathbb{F}_q)$ we simply need to calculate $\phi(Q), \psi(P) \in \mathbb{F}_q$, then $n = \phi(Q)(\psi(P))^{-1}$ [72], with ϕ defined as above and ψ defined as in lemma 3 of [72].

Example: This example will use a slightly different technique than the results in this section. The difference here being the logarithm map that we are using. The methods of this example are closely related to the methods of Smart in [82], however, we ease the method of determining m above by using another method to compute our lift, and a different logarithm map to compute m. This example, along with the new logarithm map can be found in [87].

Let E be an elliptic curve over $\mathbb{F}_{853}$ defined by the equation $y^2 = x^3+108x+4$. Let $P = (0, 2)$ and $Q = (563, 755)$. Note here that $[853]P = \mathcal{O}$, hence P is a generator for $E(\mathbb{F}_{853})$. To perform a lift of E we note that can do the following. If we consider the equations

$$\begin{aligned} y_1^2 &= x_1^3 + \mathcal{A}x_1 + \mathcal{B} \\ y_2^2 &= x_2^3 + \mathcal{A}x_2 + \mathcal{B} \end{aligned}$$

where (x_1, y_1) and (x_2, y_2) are the lifted points, we can easily obtain solutions for $\mathcal{A}$ and $\mathcal{B}$, namely

$$\mathcal{A} = \frac{y_2^2 - y_1^2 - (x_2^3 - x_1^3)}{x_2 - x_1} \quad \text{and} \quad \mathcal{B} = y_1^2 - x_1^3 - Ax_1.$$

We now lift E to $\mathcal{E}$ and obtain the equation $y^2 = x^3 + 714069x + 4$ for $\mathcal{E}$. The points P and Q are lifted to $\mathcal{P} = (0, 2)$ and $\mathcal{Q} = (563, 755)$. If we check, we see that modulo p, $\mathcal{P} \mapsto P$, $\mathcal{Q} \mapsto Q$ and that $\mathcal{E} \mapsto E$, hence the required condition of our lift are satisfied.

Now instead of calculating $[p]P$ and $[p]Q$ as above, instead we separate this into two calculations: calculate $\mathcal{P}_1 = [p-1]\mathcal{P} \equiv (x', y') \mod p^2$ and $\mathcal{Q}_1 = [p-1]\mathcal{Q} \equiv (x'', y'') \mod p^2$. We do this to obtain integer coordinates for $\mathcal{P}$ and $\mathcal{Q}$. Since p

should not appear in the denominators of any of the coordinates, these can all be inverted modulo p^2. We then calculate

$$m = p(\frac{y' - y_1}{x' - x_1}) \quad \text{and} \quad n = p(\frac{y'' - y_2}{x'' - x_2})$$

and check to see if the valuation[19] of m and n are greater than zero. If the valuation is greater than zero, we can compute the desired quantity for the ECDLP and hence $\lambda \equiv \frac{m}{n} \mod p$ (which is essentially the technique that Smart describes).

Hence in our example we have that

$$\begin{aligned} \mathcal{P}_1 &= [852]P \equiv (525448, 365082) \mod 853^2 \\ \mathcal{Q}_1 &= [852]P \equiv (543924, 505074) \mod 853^2 \end{aligned}$$

and that

$$m = 853(\frac{365082 - 2}{525448 - 0}) = \frac{45635}{77} \quad \text{and} \quad n = 853(\frac{505074 - 755}{543924 - 563}) = \frac{504319}{637}.$$

At which point we may now recover a solution for the ECDLP, hence $\lambda \equiv m/n \equiv 234 \mod 853$, as required.

3.2 Pairing Attacks

3.2.1 The MOV Attack

The MOV attack, named after its developers Menezes, Okamoto and Vanstone, introduced in [61], attempts to reduce the ECDLP on an elliptic curve $E(\mathbb{F}_q)$ to the DLP in a suitable extension $\mathbb{F}_{q^k}$ of $\mathbb{F}_q$. The map that is constructed goes from the subgroup

[19] If a/b is a rational number the p-adic valuation is defined to be $v_p(a/b) = r$, where $(a/b) = p^r(a_1/b_1)$ with $p \nmid a_1, b_1$.

generated by P on E to the group of n^{th} roots of unity in $\mathbb{F}_{q^k}$, where n is the order of the point P. The isomorphism is given by the Weil pairing. The net result of this map is that we can now solve the DLP in subexponential time using Index Calculus methods discussed earlier in II.1.6. Unfortunately we have to be careful about the size of the extension field that we wish to solve the DLP in. Fortunately, the work done in [61] provided a maximum value of $k = 6$, this in turn means that the attack will be effective for certain classes of curves. We discuss these ideas further below.

Let $E(\mathbb{F}_q)$ be an elliptic curve with group structure $\mathbb{Z}_{n_1} \oplus \mathbb{Z}_{n_2}$ with $n_2|n_1$. We will also assume that $\gcd(\#E(\mathbb{F}_q), q) = 1$.

Suppose that k is the smallest positive integer such that $E[n] \subseteq E(\mathbb{F}_{q^k})$.

Lemma 4.4 *Let $E(\mathbb{F}_q)$ be an elliptic curve such that $E[n] \subseteq E(\mathbb{F}_{q^k})$, with $\gcd(n, q) = 1$, and let $P \in E[n]$ have order n. Then for all $P_1, P_2 \in E[n]$, P_1 and P_2 are in the same coset of $\langle P \rangle$ within $E[n]$ iff $e_n(P, P_1) = e_n(P, P_2)$.*

Theorem 4.1 *There exists $Q \in E[n]$, such that $e_n(P, Q)$ is a primitive n^{th} root of unity.*

Proof:[61] Let $Q \in E[n]$. From the Weil pairing we have that $e_n(P, Q)^n = e_n(P, [n]Q)$ $= e_n(P, \mathcal{O}) = 1$. Thus $e_n(P, Q) \in \mu_n$, the subgroup of the n^{th} roots of unity in $\mathbb{F}_{q^k}$. Now there are n cosets of the subgroup generated by P, and by the above lemma, as Q varies among the representatives among these cosets, $e_n(P, Q)$ varies among the elements of μ_n.$\square$

Thus if we let $Q \in E[n]$ such that $e_n(P, Q)$ is a primitive n^{th} root of unity we get the following map and theorem.

Theorem 4.2 *The map*

$$\begin{aligned} f : \langle P \rangle &\rightarrow \mu_n \\ R &\mapsto e_n(R, Q) \end{aligned}$$

is a group isomorphism.

Proof: The proof is quite easy and was omitted from [61]. We prove it here for completeness. Clearly f is a homomorphism due to the properties of the Weil pairing. Suppose that $e_n(R,Q) = e_n(R',Q)$, then $e_n(R,Q)e_n(R',Q)^{-1} = 1 \Rightarrow e_n(R,Q)e_n(-R',Q) = 1 \Rightarrow e_n(R-R',Q) = 1 \Rightarrow R - R' = \mathcal{O} \Rightarrow R = R'$, thus f is injective. Now since both $\langle P \rangle$ and μ_n are finite of order n, this implies that f is surjective and hence bijective. Therefore $\langle P \rangle \cong \mu_n$ as required.□

Using all of the above we can now describe the reduction process which will enable us to solve the instance of the ECDLP.

Let $P \in E(\mathbb{F}_q)$ be of order n, and $R \in \langle P \rangle$. The first thing to do is to determine the smallest integer k such that $E[n] \subseteq E(\mathbb{F}_{q^k})$. Next we determine an element $Q \in E[n]$ such that $\alpha = e_n(P,Q)$ has order n. We compute $\beta = e_n(R,Q)$. Now we can determine a solution to the ECDLP by solving an instance of the DLP in $\mathbb{F}_{q^k}$. That is, we are searching for l such that $Q = [l]P$ on $E(\mathbb{F}_q)$, but by using the Weil pairing and the map from Theorem 4.2 we have successfully turned our ECDLP into an instance of the DLP, namely $\beta = \alpha^l$, where there exists subexponential algorithms to solve this problem.

The above setting can now be used to attack supersingular elliptic curves.

Definition 4.3 *Let $E(\mathbb{F}_q)$ be an elliptic curve with $q = p^m$. E is said to be supersingular if $p \mid \#E(\mathbb{F}_q) = q + 1 - t$. Equivalently if E is defined over $\mathbb{F}_p$ with p prime, then E is supersingular iff $\#E(\mathbb{F}_p) = p + 1$.*

In the above reduction we can see that we will encounter two problems. We need to determine the proper value of k such that $E[n] \subseteq E(\mathbb{F}_{q^k})$ and we need to determine the point $Q \in E[n]$ such that $\alpha = e_n(P, Q)$ has order n. As is shown in [61] we can explicitly determine a maximum value of k based on classes of supersingular elliptic curves. It turns out that if $E(\mathbb{F}_q)$ is supersingular, then there are six possibilities that we can obtain for t[20] and that k must be ≤ 6. Determining Q however is a little more complicated. If one were to choose a random point $Q \in E[n]$, then $\alpha = e_n(P, Q)$ may or may not have order n. In practice one could attempt to factor n then we could use this factorization to help find the order of α. This would avoid having to solve several instances of the DLP and obtaining partial information about the correct value of l we are attempting to solve for [61].

This algorithm then solves an instance of the ECDLP in probabilistic subexponential time. The fact that the algorithm is subexponential is clear, since we have transfered the ECDLP to the DLP we simply apply the fastest known algorithm to the DLP as discussed in II.1.6. The reason that the algorithm is probabilistic is due to an algorithm by Miller which calculates the Weil pairing, used in the isomorphism constructed in Theorem 4.2, in probabilistic polynomial time. Thus the overall complexity of the algorithm is then $L[\frac{1}{2}, q^k]$ if q is prime, and $L[\frac{1}{3}, q^k]$[21], if q is a prime

[20] [61] gives the six possible values for t along with complete group structures for each value of t.

[21] Recall the definition of $L[\alpha, \beta]$ given in II.1.6

power [61].

3.2.2 The Frey-Rück Attack

The Frey-Rück attack is quite similar in nature to the MOV attack, but uses the Tate-Lichtenbaum pairing instead of the Weil pairing.

Just like the MOV attack, the Frey-Rück attack attempts to reduce the ECDLP to the DLP in a suitable extension field over which the elliptic curve in question is defined, where the DLP can be solved with subexponential algorithms.

We recall the following setup from III.3.3. Suppose that K is a perfect field[22], and that E is an elliptic curve defined over K. Recall that the set of n-torsion points is denoted $E(K)[n]$. We further define the following set, let $nE(K) = \{[n]P \mid P \in E(K)\}$. Then the Tate-Lichtenbaum pairing is

$$\langle \cdot, \cdot \rangle : E(K)[n] \times E(K)/nE(K) \to K^\times/(K^\times)^n$$

which is a bilinear, non-degenerate pairing. Note here that the group $K^\times/(K^\times)^n$ is isomorphic to the roots of unity μ_n, thus an instance of the ECDLP on $E(K)$ is mapped to an instance of the DLP in μ_n.

Clearly this pairing can then be constructed over finite fields and yields significant cryptographic applications. Sometimes referred to as the modified Tate-Lichtenbaum

[22]A perfect field K, is one for which every algebraic extension of K is separable

pairing [87], we can define τ_n to be the following bilinear map:

$$\begin{aligned} \tau_n(\cdot,\cdot) : E(\mathbb{F}_q)[n] \times E(\mathbb{F}_q)/nE(\mathbb{F}_q) &\rightarrow \mu_n \\ \tau_n(P,Q) &= \langle P,Q\rangle^{q-1/n} \end{aligned}$$

Although the setting is exactly the same, the second setup is more desirable since it will yield a definite answer instead of a coset in $\mathbb{F}_q^\times$ modulo n^{th} powers. Again since we are mapping into the group of n^{th} roots of unity, we are mapping into a suitable extension field $\mathbb{F}_q^k$ such that $\mu_n \subseteq \mathbb{F}_q^k$. Again, as in the case of the Weil pairing, if we were to apply this to the situation of supersingular curves, we have that $k \leq 6$ and this result in a subexponential algorithm solving the ECDLP.

We present the following algorithm which summarizes both the MOV and the Frey-Rück attacks.

Algorithm 4.4 MOV/Frey-Rück Attack

Input: $P, Q \in E(\mathbb{F}_q)$, of prime order r, such that $Q = [\lambda]P$, for unknown λ
Output: The Discrete log λ of Q to the base P

1: Construct the field $\mathbb{F}_{q^k}$ such that $r \mid (q^k - 1)$
2: Choose a point $S \in E(\mathbb{F}_{q^k})$ uniformly at random with $e(P,S) \neq 1$
3: $\zeta_1 \leftarrow e(P,S)$
4: $\zeta_2 \leftarrow e(Q,S)$
5: Determine λ such that $\zeta_1^\lambda = \zeta_2$ in $\mathbb{F}_{q^k}^\times$using index calculus methods
6: Return λ

3.2.3 Calculating and Comparing the Pairings

So far we have simply described the methods by which these pairings are set up to attack the ECDLP and the net result of each mapping. In this section we give ways of calculating each pairing and discuss their relationship to one another.

Essentially, both pairings reduce to determining a function f such that $\text{div}(f) = n(P+R)-n(R)$ for points $P \in E[n]$ and $R \in E$, we would then evaluate $f(Q_1)/f(Q_2)$ for points Q_1, Q_2.

An algorithm, due to Miller, was produced to do this efficiently. Below is a description of the algorithm that appears in [6].

Algorithm 4.5 Miller's Algorithm

Input: $P, Q \in E(K)$ where P has order n.
Output: $\langle P, Q \rangle$

1: Choose a point $S \in E(K)$
2: $Q' \leftarrow Q + S$
3: $T \leftarrow P$
4: $m \leftarrow \lfloor \log_2(n) \rfloor - 1, f \leftarrow 1$
5: While $m \geq 0$ do:
6: Calculate lines l and v for doubling T.
7: $T \leftarrow [2]T$
8: $f \leftarrow f^2 \frac{l(Q')v(S)}{v(Q')l(S)}$
9: if the n^{th} bit of n is one, then:
10: Calculate line l and v for addition of T and P
11: $T \leftarrow T + P$
12: $f \leftarrow f \frac{l(Q')v(S)}{v(Q')l(S)}$
13: $m \leftarrow m - 1$
14: Return f

The functions l and v arise from the fact that we can express the group law in terms of divisors[23]. If one were to take two points on an elliptic curve E, one would normally proceed in the geometric sense by drawing a line connecting the two points. This line can then be interpreted as a function defined on E. Similarly when one connects the third point of intersection of the curve to the point of infinity, we can define the vertical line v in terms of a function as well. Thus adding points $P_1 + P_2 = P_3$, with P_3' as the intermediate point of $P_1 + P_2$, yields divisors of the form

[23]This is the Riemann Roch Theorem at work here, see [2] and [6].

$div(l) = (P_1) + (P_2) + (P_3') - 3(\mathcal{O})$ and $div(v) = (P_3') + (P_3) - 2(\mathcal{O})$.

There are two other concerns: the choice of S and constructing the function f.

One way to choose S is to simply choose a random point in $E(K)$. We can also set $S = [i]P$ with the condition that i is not a segment in the binary representation of n. Lastly we could also set $S = Q$ if $P \in E(\mathbb{F}_q)$ and $Q \notin E(\mathbb{F}_q)$.

The functions f_i can be chosen such that the following properties hold:

1. $f_1 = 1$

2. Let l and v be the straight lines used in the computation of $[i]P + [j]P$. Then

   $f_{i+j} = f_i f_j \frac{l}{v}$

We are then ultimately concerned with a value for f_n where $div(f_i) = i(P) - ([i]P) - (i-1)(\mathcal{O})$ [6].

This algorithm runs in polynomial time and has been improved upon in [7], [19], [20] and [46]; further reducing the running time of computing of the Weil and Tate-Lichtenbaum pairings.

One might now wonder which pairing to choose when trying to solve an instance of the ECDLP defined over a supersingular curve. There are a couple of subtle differences in each pairing, which is a result of the space for which each pairing is defined. Observe that in the Weil pairing we need $E[n] \subseteq E(\mathbb{F}_q)$ which in turn implies that $\mu_n \subseteq \mathbb{F}_q^\times$. For the Tate-Lichtenbaum pairing we require $\mu_n \subseteq \mathbb{F}_q^\times$, but only need one point of order n to be in $E(\mathbb{F}_q)$, and not the entire group $E[n]$ [87]. Thus the Tate-Lichtenbaum pairing can be used in circumstances where the Weil pairing does not apply. The Tate-Lichtenbaum pairing is also faster to compute,

especially with the enhancements listed in the resources above. A summary of some of these improvements are listed here and can be found, along with a few others, in [4]:

1. Exploiting properties of the definition of the underlying field
2. Changing the base in the Algorithm
3. Replacing divisors by points
4. Choosing points of Low Hamming Weight
5. Speeding up the final exponentiation.

Note that there also exists elliptic curves which are not supersingular that are vulnerable to a pairing attack. As an example, observe that for every prime power $q = p^\alpha$ with $p > 2$, there exists elliptic curves E over $\mathbb{F}_q$ with $q - 1$ points and the reduction algorithm requires no extension of the base field [6].

With a little more versatility and more efficient implementation techniques, the Tate-Lichtenbaum pairing is better suited for attempts at solving the ECDLP when applicable.

Example: Let E be the the elliptic curve $y^2 + y = x^3 - x^2 - 10x + 7$ defined over $\mathbb{F}_q$ with 1609667 elements. Using Schoof's algorithm, or the SEA algorithm, it can be shown that $\#E(\mathbb{F}_q) = 2 \times 804833$. This makes the trace of the Frobenius equal to 2 and hence for $p = 804833$ the p^{th} roots of unity are contained in $\mathbb{F}_q$. This means that we must apply the Tate-Lichtenbaum pairing instead of the Weil pairing since all points of order p are not contained in $E(\mathbb{F}_q)$ [24]. Let $P = (797482, 1369997) \in E(\mathbb{F}_q)$,

which can be shown to have order $p = 804833$. Let $Q = (822050, 1036146)$. We now wish to solve the ECDLP, $[\lambda]P = Q$.

As described in the previous section, we begin by calculating $[p]P$ and see which numbers do not occur in this calculation. Very quickly we notice that $[5]P$ and $[6]P$ do not occur and so we will use these numbers for our divisor. To calculate $\langle P, P\rangle$ we use the divisors $D_P = (P) - (\mathcal{O})$ and $D_{P'} = ([6]P) - ([5]P)$. Similarly we calculate $\langle Q, P\rangle$, and use the divisor $D_Q = (Q) - (\mathcal{O})$ and $D_{P'} = ([2]P) - (P)$. When we do this we obtain the following

$$\begin{aligned} \tau_n(P, P) = 822530^{(q-1)/p} &= 1293131 \\ \tau_n(Q, P) = 824365^{(q-1)/p} &= 508028. \end{aligned}$$

Hence we have mapped our ECDLP to the DLP, and so we can solve the following equation

$$1293131^{\lambda} \equiv 508028 \mod q$$

using the Index Calculus discussed in II.1.6. Hence we find that $\lambda = 89865$ as required.

3.3 Weil Descent and the GHS Attack

3.3.1 The Attack

The GHS attack uses a techniques known as Weil Descent in an attempt to solve the ECDLP on a given elliptic curve defined over $\mathbb{F}_{2^m}$[24]. The techniques of Weil Descent

[24]Note that this attack has been extended to elliptic curves defined over fields of odd characteristic by Diem in [14].

were first introduced to the cryptographic community in a talk by Frey in [22] at ECC '98. Following this, Galbraith and Smart devised a preliminary construction in [32], and finally Gaudry, Hess and Smart were able to give not only a complete construction of how to attack an elliptic curve cryptosystem using Weil Descent, but also attempted to construct hyperelliptic curve cryptosystems using this technique; a problem that Frey was considering in his original introduction of this subject.

The GHS attack is akin to pairing attacks in the sense that it attempts to reduce the ECDLP to an easier problem. Instead of reducing the ECDLP to the DLP, we will now attempt to reduce it to the HCDLP, that is the Hyperelliptic Curve Discrete Logarithm Problem. The process of reducing the ECDLP to the HCDLP is not a trivial matter; however it can be shown that the reduction process does not govern the expected time cost in the algorithm. The governing step in the algorithm is solving the HCDLP on a hyperelliptic curve. There are many algorithm that can do this: Index Calculus methods of Hafner-McCurley, or the methods of Enge and Gaudry for instance. When we discuss methods of solving the HCDLP we will use the latter method.

As a final note before we begin the description of this attack, we consider when this attack is thought of as being successful. Since the fastest known algorithm to solve the ECDLP are the methods of Pollard, the GHS attack is thought to be successful if the expected running time of the algorithm solves an instance of the ECDLP in less time then Pollard's methods. We quantify this a little more. Gaudry was able to produce an algorithm to solve an instance of the HCDLP with expected running $O(g^3 q^2 \log^2 q + g^2 g! \log^2 q)$, which becomes impractical for values of $g \geq 10$, [43].

This algorithm was later improved by Gaudry and Enge to yield a subexponential algorithm with expected running time of $L_{q^g}[\sqrt{2}] = L_{q^{2g+1}}[1]$ [43]. Thus one can see immediately that if the resulting hyperelliptic curve obtained in our reduction has genus which is too large, then the GHS will have longer expected running time then Pollard's methods, and hence the attack is considered to have failed.

We first sketch the GHS Attack before giving a detailed description. Given an elliptic curve E defined over a field $\mathbb{F}_{q^n}$ we construct an abelian variety, known as the Weil Restriction of Scalars of E over $\mathbb{F}_{q^n}$. Next, we find a curve C defined over $\mathbb{F}_q$ lying on our constructed abelian variety, such that C has an $\mathbb{F}_q$-rational point P_0 at the point of infinity of the abelian variety. The points P and Q of the ECDLP in $E(\mathbb{F}_{q^n})$ correspond to divisors D_1 and D_2 in $\mathrm{Pic}^0(C)(\mathbb{F}_q)$, which is where we will solve the HCDLP using the methods of Enge and Gaudry.

We will need the following result to get a better handle on the Weil Restriction of Scalars.

Theorem 4.3 *Let $\mathcal{A}$ be an abelian variety over a field k and let $\mathcal{B}$ be an abelian subvariety of $\mathcal{A}$. Then there is an abelian subvariety $\mathcal{C}$ of $\mathcal{A}$ such that $\mathcal{A}$ is isogenous to $\mathcal{B} \times \mathcal{C}$.*

With this result we can construct the Weil Restriction as follows. We first take a basis for $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$, and expand the coordinate functions on the curve $E(\mathbb{F}_{q^n})$ in terms of the new basis. By substituting into the equation for E, expanding, and equating the coefficients of the new basis, we obtain a system of n equations in $2n$ variables. This then defines our variety $W_{\mathbb{F}_{q^n}/\mathbb{F}_q}(E)$, the Weil Restriction of Scalars

of E. The group operation on $W_{\mathbb{F}_{q^n}/\mathbb{F}_q}(E)$ is induced by the group operation on the curve E.

The above theorem now comes into play. There are two cases to consider when we define $W_{\mathbb{F}_{q^n}/\mathbb{F}_q}(E)$. The following theorem can be found in [32].

Theorem 4.4 *If E is defined over $\mathbb{F}_q$ then $W_{\mathbb{F}_{q^n}/\mathbb{F}_q}(E) \cong E(\mathbb{F}_q) \times V$, where V is an abelian variety of codimension 1. If n is coprime to $\#E(\mathbb{F}_q)$ then $V = \{P \in W_{\mathbb{F}_{q^n}/\mathbb{F}_q}(E) \mid Tr_{\mathbb{F}_{q^n}/\mathbb{F}_q}(P) = \mathcal{O}\}$, where the trace is computed using the mapping from $W_{\mathbb{F}_{q^n}/\mathbb{F}_q}(E)$ to $E(\mathbb{F}_q)$.*

Thus if E is not defined over $\mathbb{F}_q$ we set $\mathcal{A} = W_{\mathbb{F}_{q^n}/\mathbb{F}_q}(E)$, otherwise set $\mathcal{A} = V$ from Theorem 4.4.

So we can define the abelian variety. What does this give us? Well this gives us a starting point for finding our hyperelliptic curve which we hope will lead us to a quicker solution for the ECDLP than Pollard's methods. What we can do with this variety is use techniques mentioned in the previous chapter in an attempt to obtain an equation of a hyperelliptic curve of relatively small genus.

If we can intersect our abelian variety $\mathcal{A}$ with $\dim(\mathcal{A}) - 1$ hyperplanes in standard position, then by elimination theory this will give us a variety of dimension one. This in turn will hopefully define an equation for a hyperelliptic curve of genus g where we would like to solve the HCDLP.

We now elaborate on this sketch. Suppose that we wish to solve the ECDLP $P_1 = [\lambda]P_2$ on $E(\mathbb{F}_q)$. We construct our Weil Restriction of Scalars, as above, and then find a curve C defined on $\mathcal{A}$. The by the universal mapping property of Jacobians,

see [32], there is a map $\psi : Jac(C) \to \mathcal{A}$. The points used in the ECDLP correspond to points on $\mathcal{A}$, which can be pulled back using ψ to divisors D_1 and D_2 in $\mathrm{Pic}^0(C)$ where we solve the problem using Index Calculus methods. One outstanding matter is description of ψ. The following is proved in [32].

Theorem 4.5 *The map $\psi : Pic^0(C) \to \mathcal{A}$ is given by*

$$\psi(\sum_{i=1}^{d}(Q_i) - d(P_0)) = \sum_{i=1}^{d}\phi(Q_i)$$

where ϕ is a map from C to $\mathcal{A}$ and each Q_i are points on $C(\overline{\mathbb{F}_q})$.

Inverting ψ requires finding a divisor which maps ψ to a given point, P, on $\mathcal{A}$. To do this we must find p non-singular points on C, where p is to be determined. We label these $\{P_1, \ldots, P_p\}$. Now using ϕ, we maps these points to our variety $\mathcal{A}$ and obtain the following

$$Q_i = \phi(P_i),\ i = 1, \ldots, p.$$

If we think of the coordinates of each of these points as variables, we see that we have obtained p equations in $2p$ unknowns.

Using the group law on $\mathcal{A}$ we can determine the equations for the coordinates of the sum $\sum_{i=1}^{p} Q_i$ and then equate this to the given element P on $\mathcal{A}$. Since $\mathcal{A}$ has dimension n we now have another n equations, and thus have a total of $p + n$ equations in $2p$ unknowns. Notice that when $p > n$ we expect that the variety to have dimension at least $p - n$, hence choosing p large enough will produce a curve, surface, etc., for which we can find the points P_i that lie on C.

We now construct divisors $D_i = (Q_i) - (P_0)$ in $\mathrm{Pic}^0(C)$ and thus $\psi(\sum_{i=1}^{p} D_i) = P$ as required, with different points, P_0, on the variety giving rise to different divisors

D_i. Suppose now that we have determined divisors D_1' and D_2' such that $\psi(D_1') = P_1$ and $\psi(D_2') = P_2$. We now compute $D_i = [\#\text{Pic}^0(C)(\mathbb{F}_q)/\#E(\mathbb{F}_q)]D_i'$ and attempt to solve the discrete logarithm problem in $\text{Pic}^0(C)$ which yields a solution to the original ECDLP on $E(\mathbb{F}_q)$ [32].

3.3.2 Extending this Attack Using Isogenies

We can extend the GHS using isogenies. Recall that an isogeny is a rational map $\phi : E_1 \to E_2$ between elliptic curves E_1 and E_2, and that $\#E_1 = \#E_2$. The idea behind this is that if we have an elliptic curve, defined over $\mathbb{F}_q$, and we attempt to solve the ECDLP using the GHS attack, then it may happen that when we obtain our equation for our hyperelliptic curve, we have obtained one with too large a genus for the Index Calculus algorithm to be effective in solving an instance of the HCDLP. We could return to our original curve E and attempt to find an isogeny ϕ from E to E_1 say such that the GHS will be an effective attack for an instance of the ECDLP over E_1. We would then use ϕ to solve the ECDLP on E. In [31] the authors not only give an explicit construction of how to determine ϕ, but, building on the work of Galbraith in [27], the authors are able to show that the worst case average running time for constructing this isogeny is $O(q^{\frac{n}{4}+\epsilon})$ [31]. The implications of determining isogenies, and effectively increasing the number of curves that can now be attacked by the GHS is discussed below.

3.3.3 Implications and Results

This attack is fairly significant. Since this attack is defined for curves defined over the field $\mathbb{F}_{2^m}, m \in \mathbb{Z}$, the attack could be applied to many situations of cryptographic interest since in particular, being defined over $\mathbb{F}_{2^m}$ yields quick arithmetic operations and efficient cryptosystems for a given curve E. In particular, there are industrial standards that allow elliptic curves to be defined over $\mathbb{F}_{2^{155}}$ and $\mathbb{F}_{2^{185}}$. Analysis done by Menezes et. al. in [43] and Menezes and Qu in [57] suggests that there is little chance that an elliptic curve is susceptible to such an attack. In particular they were able to demonstrate that roughly 1 in 2^{122} could be attacked using the GHS. However, work done by Galbraith et. al. in [31] shows that by extending the GHS attack using isogenies allows a greater number of curves to be attacked by this method. They were able to improve the original chance that a curve could be attacked from 1 in 2^{122} to 1 in 2^{52} [31], a significant improvement. This suggests that this field is weak for cryptosystems. A similar analysis has been done on the field $\mathbb{F}_{2^{185}}$, which can also be shown to be weak for cryptosystems. This then raises the question: What fields are weak when it comes to the GHS or the extended GHS?

Further analysis of what can be called the Generalized GHS, or the extended GHS, has been done by Menezes and Teske in [59] and gives a thorough answer to the above question. In [59] the authors are able to characterize fields as follows:

1. The fields $\mathbb{F}_{2^{6l}}$, $\mathbb{F}_{2^{7l}}$ and $\mathbb{F}_{2^{8l}}$ are weak.
2. The fields $\mathbb{F}_{2^N}$ where $N \nmid 3, 5, 6, 7$ or 8 are not weak under the generalized GHS

In particular this analysis, as well as the original construction, shows that if E

is a curve defined over a field $\mathbb{F}_{2^p}$ where $p \in [160, 600]$ is a prime, then the GHS attack and the generalized GHS attack do not apply. Thus to avoid this attack in its entirety, one can simply choose a curve defined over $\mathbb{F}_{2^p}$ where p is a prime of the recommended form.

3.3.4 Further Work

When this author first encountered the GHS attack, I wondered why the Descent technique used hyperplanes in standard position. This brief section explores this in a little more depth.

At first glance we could potentially carve up our variety $\mathcal{A}$ with something other than hyperplanes in standard position. Perhaps we could use $\dim(\mathcal{A})-2$ hyperplanes, and a quadratic hypersurface of some sort. Suppose that we did this, does this result in a variety of dimension one? In other words does the resulting equation define a curve for which we could solve the ECDLP in its Jacobian? Well, if we intersect our variety with $n-2$ hyperplanes in standard position and a quadratic hypersurface, then by the dimension theorem, the resulting variety should have dimension at least 1. To ensure that the variety has dimension exactly one, we must choose a hypersurface in such a manner that it does not vanish on all of $\mathcal{A}$. If we do this then we can intersect $\mathcal{A}$ with any type of surface that we wish. Now the question becomes whether or not the result is a hyperelliptic curve with low enough genus that subexponential algorithms can be employed to solve the HCDLP.

Recently, Diem has announced that solving an instance of the DLP in class groups of plane curves of genus 3 is asymptotically faster than solving the HCDLP in the

genus 3 case [15]. Diem's results suggests that these ideas can be exploited even more. Perhaps it is not essential that the Descent step in the GHS attack result in a hyperelliptic curve, but instead just any plane curve of low genus. Thus if any method of intersecting the variety $\mathcal{A}$ to obtain a plane curve of low genus could be employed, the result would then lead to a quicker solution for an instance of the ECLDP.

The interesting thing here would be to determine if the method of intersecting the variety with something other than hyperplanes in standard position would result in an equation for a plane curve with too high of degree, and thus too high of a genus. I believe that further work is needed in this area.

Example: We divide this example into two sections. The first will demonstrate the method of Descent that we apply to our abelian variety $\mathcal{A}$, while the second section will give a concrete example of the transfer of the ECDLP to the HCDLP omitting the lengthier calculations involved in applying the Descent method to this example. The first example is due to Smart and Galbraith in [32], while the second is due to Menezes, Jacobson and Stein in [43].

Example One: Let $k = \mathbb{F}_{2^{n_1}}$ and set $m = nn_1$ for some n. Let K be an extension field of k such that K has an Optimal Normal Basis[25] over k. Choosing such a basis means that $n+1$ should be prime and that 2^{n_1} should be primitive in $\mathbb{F}_{n+1}$. Thus the n^{th} roots of $(x^{n+1}-1)/(x-1)$ form such a basis of K over k.

For simplicity choose $n = 4$ and let $\{\theta, \theta^2, \theta^4, \theta^8\}$ be such a basis. Let E be the elliptic curve $y^2 + xy = x^3 + b$ where $b \neq 0$. By writing out b, x and y in terms of

[25]For details about Optimal Normal Bases consult [5, 22].

elements of the basis and substituting in this equation, expanding and equating powers of θ, we obtain four equations in eight unknowns, namely $\{x_0, \ldots, x_3, y_0 \ldots, y_3\}$ where $x_i, y_i \in k$ and are the unknown coefficients of x and y when they are expressed in term of the basis. This defines our abelian variety $\mathcal{A}$, which is a four dimensional variety in eight dimensional affine space. When we intersect this variety with our hyperplanes in standard position, $x_0 = x_1 = x_2 = x_3$, we obtain the following variety

$$\begin{aligned} y_3^2 + y_0 x_0 + x_0^3 + b_0 &= 0 \\ y_0^2 + y_1 x_0 + x_0^3 + b_1 &= 0 \\ y_1^2 + y_2 x_0 + x_0^3 + b_2 &= 0 \\ y_2^2 + y_3 x_0 + x_0^3 + b_3 &= 0 \end{aligned}$$

We can then eliminate the variables y_3 and y_1 by taking the resultant of the first and fourth equations, and the second and third equations respectively and obtain a new variety with defining equations

$$\begin{aligned} y_2^4 + x_0^6 + b_3^2 + y_0 x_0^3 x_0^5 + b_0 x_0^2 &= 0 \\ y_0^4 + x_0^6 + b_1^2 + y_2 x_0^3 x_0^5 + b_2 x_0^2 &= 0. \end{aligned}$$

Lastly we eliminate y_2 from these equations and set $x_0 = x$ and $y_0 = y$ and obtain the following equation for the affine curve

$$C: \quad y^{16} + x^{15} y + (x^{24} + x^{20} + x^{18} + x^{17} + b_0 x^{14} + b_3^2 x^{12} + b_2^4 x^8 + b_1^8)$$

Thus an instance of the ECDLP on the original curve E will now be mapped to C where it can be solved using the methods of Enge and Gaudry mentioned in the previous section.

Example Two: Let E be the elliptic curve $y^2+xy = x^3+ax+b$ defined over $\mathbb{F}_{2^{124}}$, where $a = z^{105}$,

$$b = \begin{array}{l} z^{108}+z^{106}+z^{102}+z^{101}+z^{99}+z^{93}+z^{87}+z^{85}+z^{75}+z^{70}+z^{68}+z^{67}+ \\ z^{66}+z^{62}+z^{59}+z^{58}+z^{56}+z^{55}+z^{54}+z^{53}+z^{50}+z^{49}+z^{48}+z^{46}+z^{45} \\ +z^{44}+z^{42}+z^{41}+z^{40}+z^{33}+z^{32}+z^{29}+z^{27}+z^{24}+z^{23}+z^{22}+z^{20} \\ +z^{18}+z^{16}+z^{15}+z^{14}+z^{9}+z^{8}+z^{7}+z^{3}+z^{2}+z \end{array}$$

and the irreducible polynomial over $\mathbb{F}_2$ is $z^{124}+z^{19}+1$.

It can be shown, that $\#E(\mathbb{F}_{2^{124}}) = 2r$ where

$$r = 10633823966279326985483775888689817121.$$

Let P and Q be points on E for which we wish to solve the ECDLP. In [43] these points were generated verifiably at random to obtain the following

$$\begin{aligned} P &= (191662899311163509148924354609692288 9, \\ &\quad 39549266381157102372793271078772986 63) \\ Q &= (141524161371548670426547540065416908 09, \\ &\quad 157332415929030717233515654264947118 69). \end{aligned}$$

Now the challenge is to determine the appropriate $\lambda \in [0, r-1]$ such that $[\lambda]P = Q$. To do this we apply the Weil Descent method and map the ECDLP into the Jacobian of a hyperelliptic curve, and attempt to solve the HCDLP. Using this technique E is mapped to the following curve defined over $\mathbb{F}_{2^4}$, with the chosen irreducible polynomial as w^4+w+1.

$$\begin{array}{c} v^2+(w^3u^{31}+w^9u^{30}+wu^{28}+w^{11}u^{24}+w^{12}u^{16}+w^{12})v \\ = (w^6u^{63}+w^{14}u^{60}+w^6u^{56}+w^6u^{48}+1) \end{array}$$

Now we need to calculate divisors in $Jac(C(\mathbb{F}_{2^4}))$. To do this we choose a point R of order r and add it to P and Q. When we apply the descent procedure the point $P+R$, $Q+R$ and R are mapped to divisors D_1, D_2 and D_3 respectively. In this example the following point R was used:

$$\begin{aligned} R &= (119493869221292418542879192570498114 85, \\ &\quad 138197028178387310271941932901208011 07). \end{aligned}$$

While Divisors D_1, D_2 and D_3 are calculated to be

$$\begin{aligned} D_1 &= div(u^{31}+w^6u^{30}+w^4u^{29}+w^5u^{28}+w^{10}u^{27}+w^3u^{26}+w^{14}u^{25}+w^4u^{24} \\ &\quad +w^{14}u^{23}+u^{22}+w^5u^{21}+w^9u^{20}+w^{14}u^{19}+w^4u^{18}+w^{14}u^{17}+w^{12}u^{16} \\ &\quad +w^6u^{15}+w^{14}u^{14}+w^7u^{13}+w^7u^{12}+w^2u^{11}+w^7u^{10}+w^{13}u^9+w^7u^8 \\ &\quad +u^7+w^9u^6+w^{14}u^5+w^3u^4+w^2u^3+w^{10}u^2+w^9u+1, u^{30}+w^8u^{29} \\ &\quad +wu^{28}+w^8u^{27}+w^{14}u^{26}+w^5u^{24}+w^{10}u^{23}+w^4u^{22}+w^8u^{21} \\ &\quad +w^9u^{19}+w^2u^{18}+w^3u^{16}+w^5u^{15}+w^{13}u^{14}+w^{11}u^{13}+w^7u^{12} \\ &\quad +u^{11}+w^8u^{10}+u^9+w^2u^8+w^6u^7+u^6+wu^5+w^9u^4+w^{13}u^3+w^2u+w^7), \end{aligned}$$

$$\begin{aligned} D_2 &= div(u^{31}+w^{12}u^{30}+w^3u^{29}+w^8u^{28}+w^{12}u^{27}+w^{14}u^{26}+w^{13}u^{25} \\ &\quad +w^9u^{24}+w^7u^{23}+w^{12}u^{22}+u^{20}+w^3u^{18}+w^{12}u^{17}+u^{16}+w^{12}u^{15} \\ &\quad +w^3u^{14}+w^9u^{13}+w^6u^{12}+w^9u^{11}+w^7u^{10}+w^2u^9+w^8u^8+w^{11}u^7+w^9u^6 \\ &\quad +w^{12}u^5+w^{10}u^4+w^{11}u^3+w^{11}u^2+w^{11}u+1, w^{14}u^{29}+w^6u^{28}+u^{27} \\ &\quad +w^{11}u^{26}+w^{11}u^{25}+w^4u^{24}+w^{14}u^{22}+w^5u^{21}+w^3u^{20}+w^{14}u^{19} \\ &\quad +w^5u^{18}+w^2u^{17}+w^8u^{15}+u^{14}+w^4u^{13}+w^7u^{12}+w^{10}u^{11}+w^6u^{10} \\ &\quad +w^4u^9+w^2u^8+w^{14}u^7+wu^6+w^{11}u^4+w^{11}u^3+w^2u^2+w^9u+w^6), \end{aligned}$$

$$\begin{aligned} D_3 &= div(u^{31}+w^{14}u^{30}+w^5u^{28}+u^{27}+w^8u^{26}+w^{11}u^{25}+w^{13}u^{24}+w^2u^{23} \\ &\quad +w^5u^{22}+w^9u^{21}+w^7u^{20}+w^{12}u^{19}+w^4u^{18}+w^9u^{17}+w^{13}u^{16}+w^4u^{15} \\ &\quad +w^{13}u^{14}+u^{12}+wu^{11}+w^3u^{10}+w^6u^9+w^8u^8+w^7u^7+w^{14}u^6 \\ &\quad +u^5+w^5u^4+w^9u^2+w^7u+w^9, w^7u^{30}+w^3u^{29}+w^4u^{28}+wu^{27}+w^6u^{26} \\ &\quad +w^7u^{25}+wu^{23}+w^6u^{22}+w^7u^{21}+w^9u^{19}+w^9u^{18}+w^2u^{16}+w^5u^{15} \\ &\quad +w^2u^{13}+w^5u^{12}+u^{11}+w^6u^{10}+u^9+w^2u^8+w^5u^7+w^7u^6+w^2u^5+w^9u^4 \\ &\quad +w^2u^3+w^7u^2+w^3u+w^{13}). \end{aligned}$$

Hence we have reduced our task to solving $(D_2 - D_3) = [\lambda](D_1 - D_3)$ in the Jacobian of C. Applying the Enge -Gaudry method mentioned about we find that the solution for the HCDLP, and hence the ECDLP is

$$\lambda = 289697194482016303350776099807354482$$

as required.

3.4 The Xedni Calculus

Born from a thought that the Index Calculus would never be an effective attack against Elliptic Curve Cryptosystems; Joseph Silverman, in 1999, presented a new attack dubbed the Xedni Calculus, since it stood the Index Calculus on its head. There was great anticipation surrounding the algorithm. After its introduction, it was shown, by Koblitz, that if this attack is successful, it could be modified to attack not only elliptic curve cryptosystems, but also the Digital Signature Standard and RSA cryptosystems [44]. Thus essentially all public-key cryptosystems would be threatened.

The original idea was to reproduce an index calculus type of attack on the ECDLP. The initial setup is as follows. Suppose that we wanted to find a k such that $Q = [k]P$ on an elliptic curve E over $\mathbb{F}_p$ for some prime p. We can than lift E, P and Q to an elliptic curve $\mathcal{E}$ over $\mathbb{Z}$ with points $\mathcal{P}$, $\mathcal{Q}$. If we can find k' such that $\mathcal{Q}= [k']\mathcal{P}$ then we have solved the equation over $\mathbb{F}_p$, ie. $Q = k'P$. The problem is that in most cases, the points $\mathcal{P}$ and $\mathcal{Q}$ are independent and as such, no k' exists [87, 156]. Thus the Index Calculus fails to translate to the elliptic curve situation. Silverman, however,

devised a way around this. One of the difficulties lies in the lifting of the points from $E(\mathbb{F}_p)$ to $E(\mathbb{Q})$. Instead of lifting the points to a curve $E(\mathbb{Q})$ we would instead choose a curve that goes through the points that had been lifted. We would then look for relationships among these lifted points. As a result, we obtain a system of linear equations which we can readily solve; we then convert the curve to Weierstrass form.

The hope is that there would be one or more relations among the set of lifted points. These relations could then be reduced mod p to obtain relations between P and Q, and thus solving the ECDLP. Unfortunately, previous work by Néron and Masser [79] suggests that the set of lifted points will usually be independent. Silverman then describes further restrictions, what he calls Reverse Mestre Conditions [79], to hopefully result in a curve $\mathcal{E}$ which has smaller rank than expected.

3.4.1 Background

As mentioned by Miller, and elaborated on by Silverman and Suzuki in [80], the Index Calculus, although a subexponential algorithm to solve the DLP, failed miserably at attacking ECDLP because of two main drawbacks:

1. Rank/Height Obstruction

2. Lifting Obstruction

Both obstructions are intertwined. The idea is to take a point of $E(\mathbb{F}_p)$ and lift it to a curve $E(\mathbb{Q})$. The problems are that, in the first case, the probability of lifting a point of $E(\mathbb{F}_p)$ to a point $E(\mathbb{Q})$ whose height is bounded by something reasonable is small. And secondly, there is the problem of actually lifting the point. One possibility is to

take a point $(x, y) \in E(\mathbb{Z}/p^k\mathbb{Z})$ but there are too many possible choices of lifts of this point [80].

As it will be seen, the lifting problem that is inherent in the Index Calculus approach, does not appear in the Xedni Calculus attack. In its place will be the problem of trying to force lifted points to be dependent so that the ECDLP can be solved[26].

In general theory, the Reduction Modulo p Theorem gives us a way from passing from an elliptic curve defined over $\mathbb{Q}$ to that of a curve defined over the finite field $\mathbb{F}_p$ for some prime p. The map is a well-defined homomorphism from $E(\mathbb{Q}) \rightarrow E(\mathbb{F}_p)$. The trouble is working our way back from $E(\mathbb{F}_p)$ to $E(\mathbb{Q})$. As mentioned above, there are many possibilities for the point P to be lifted to. Instead we are now presented with the task of lifting a set of given points, simply choosing a representative for them, then forcing a curve to pass through these points. The lifting of these points is the easier of the two problems. Forcing a curve through the lifted points is quite technical, and requires a more intimate knowledge of linear algebra as well as exact sequences. This information can be found in [79, Appendix B].

Step 6 of the algorithm below will require a lifting of points in $\mathbb{P}^2$ modulo p and modulo M to a point in $\mathbb{P}^2(\mathbb{Q})$. Given a point $R_i = [\alpha_i, \beta_i, \gamma_i], 1 \leq i \leq I$, with integer coordinates, we want to find a corresponding point $R = [\alpha, \beta, \gamma]$ such that

$$R \equiv R_i \mod m_i \text{ in } \mathbb{P}^2 \quad \forall\ 1 \leq i \leq I.$$

[26] this will appear in Step 7

We will assume that the m_i's are pairwise relatively prime and we will take $m = \prod_{j=1}^{I} m_j$. The first step is to use the Chinese Remainder Theorem to find integers a, b, c that satisfy

$$a \equiv \alpha_i \mod m_i, \quad b \equiv \beta_i \mod m_i, \quad c \equiv \gamma_i \mod m_i, \quad \forall\ 1 \leq i \leq I.$$

This gives us a point in $\mathbb{P}^2(\mathbb{Q})$ with the desired property. Following this we consider the lattice generated by the columns of the matrix

$$\begin{pmatrix} a & m & 0 & 0 \\ b & 0 & m & 0 \\ c & 0 & 0 & m \end{pmatrix}.$$

We then find a vector $[\alpha, \beta, \gamma]$ in this lattice. Note that this vector should satisfy

$$[\alpha, \beta, \gamma] \equiv d[a, b, c] \mod m$$

for some integer d, so that $[\alpha, \beta, \gamma]$ and $[a, b, c]$ represent the same point in $\mathbb{P}^2(\mathbb{Q})$. If $\gcd(d, m) = 1$ then they reduce to the same point and we are done. Otherwise, we find a basis for the kernel of the matrix and us it to adjust $[a, b, c]$ [79].

A second, quite technical aspect, is Silverman's idea of employing Reverse Mestre Conditions, to obtain a greater probability of gaining dependence among the r lifted points. Mestre devised a way to obtain elliptic curves of higher than expected rank. Conversely, Silverman would like to apply Mestre's formula in reverse, so to speak, to obtain a curve of smaller than expected rank, hence obtaining a dependency relation among the r points. Mestre used congruence conditions modulo l, for small primes l, to force the quantity

$$\#E(\mathbb{F}_l) = l + 1 - a_l(E)$$

to be large. Here $a_l(E)$ are the Fourier coefficients of E over $\mathbb{Q}$. This idea is based on the Birch and Swinnerton-Dyer conjecture[27]. If $E(\mathbb{Q})$ has high rank, then the point on the curve should be sparse modulo l to force $E(\mathbb{F}_l)$ to be large. Silverman is now concerned with making $a_l(E)$ to be as large as possible for small primes l thus making the quantity $\#E(\mathbb{F}_l)$ smaller and increasing the likelihood that the curve has smaller than expected rank (hopefully $\leq r-1$).

3.4.2 The Algorithm

Step 1

Fix an integer $4 \leq r \leq 9$ and an integer M which is a product of small primes. We shall assume that the characteristic of the field $p \nmid M$. Here r is the number of points to be lifted, where M is the product of primes for which the reverse Mestre conditions will be imposed.

Step 2

For any set of r triples, $P_i = [x_i, y_i, z_i], 1 \leq i \leq r$, define an r-by-10 matrix $\mathbf{B} = \mathbf{B}(P_1, \ldots, P_r)$ of cubic polynomials as

$$B = \begin{pmatrix} x_1^3 & x_1^2y_1 & x_1y_1^2 & y_1^3 & x_1y_1z_1 & y_1^2z_1 & x_1z_1^2 & y_1z_1^2 & z_1^3 \\ x_2^3 & x_2^2y_2 & x_2y_2^2 & y_2^3 & x_2y_2z_2 & y_2^2z_2 & x_2z_2^2 & y_2z_2^2 & z_2^3 \\ \vdots & & & \ddots & & \ddots & & & \vdots \\ x_r^3 & x_r^2y_r & x_ry_r^2 & y_r^3 & x_ry_rz_r & y_r^2z_r & x_rz_r^2 & y_rz_r^2 & z_r^3 \end{pmatrix}$$

[27] for more information about this consult [44]

We choose r points

$$P_{M,i} = [x_{M,i}, y_{M,i}, z_{M,i}], \qquad 1 \leq i \leq r,$$

having integer coefficients and that satisfy:

1. the first 3 points are the triangle of reference and the 4^{th} point, the unit point: $[1,0,0], [0,1,0], [0,0,1]$ and $[1,1,1]$.

2. for every prime $l|M$, the matrix $\mathbf{B}(P_{M,1}, \ldots, P_{M,r})$ has maximal rank modulo l.

We further choose coefficients $u_{M,1}, \ldots, u_{M,10}$ so that the points $P_{M,1}, \ldots, P_{M,r}$ satisfying the congruence:

$$\begin{aligned} u_{M,1}x^3 + u_{M,2}x^2y + u_{M,3}xy^2 + u_{M,4}y^3 + u_{M,5}x^2z \\ + u_{M,6}xyz + u_{M,7}y^2z + u_{M,8}xz^2 + u_{M,9}yz^2 + u_{M,10}z^3 \equiv 0 \bmod M. \end{aligned} \tag{4.3}$$

Observe that from condition 1 above we have that $u_{M,1} = u_{M,4} = u_{M,10} = 0$ and all the other coefficients will sum to zero modulo M. The idea is to make these choices so that equation (4.3) has the smallest number of solutions modulo M subject to the above conditions. Also for any particular prime l, imposing these conditions to force equation (4.3) to have a small number of solutions modulo l is quite simple. We can then use the Chinese Remainder Theorem to find the values of the $u_{M,i}$'s modulo M.

Now the choices of the points $P_{M,i}$'s must be made with a certain level of care since they may impose certain constraints on the linear relations satisfied by the lifted points.

Step 3

We now choose r random pairs of integers (s_i, t_i) satisfying $1 \leq s_i, t_i \leq \#E(\mathbb{F}_p)$ and for each $i \in [1, r]$, compute the points $P_{p,i} = (x_{p,i}, y_{p,i})$ defined by

$$P_{p,i} = [s_i]S - [t_i]T \in E(\mathbb{F}_p).$$

Notice here that we can assume that $P_{p,i} \neq 0$ and $P_{p,i} \neq \pm P_{p,j}$ for all $i \neq j$ otherwise the ECDLP is solved.

Step 4

Working in the projective space $\mathbb{P}^2$, we can make a change of variables of the form

$$\begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = \begin{pmatrix} \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} \\ \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} \\ \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{4.4}$$

Under this transformation we have that the first four points correspond to the triangle of reference and the unit point, that is $P_{p,1} = [1, 0, 0], P_{p,2} = [0, 1, 0], P_{p,3} = [0, 0, 1]$ and $P_{p,4} = [1, 1, 1]$. The equation of our new curve E over $\mathbb{F}_p$ then has the following form:

$$\begin{aligned} u_{p,1}x^3 + u_{p,2}x^2y + u_{p,3}xy^2 + u_{p,4}y^3 + u_{p,5}x^2z \\ + u_{p,6}xyz + u_{p,7}y^2z + u_{p,8}xz^2 + u_{p,9}yz^2 + u_{p,10}z^3 = 0 \end{aligned}$$

Recall that in Step 2 we actually have that $u_{p,1} = u_{p,4} = u_{p,10} = 0$ and all other coefficients sum to zero modulo p. Notice that the matrix in (4.4) is easily computed by solving a system of 8 homogeneous equations in 9 variables over $\mathbb{F}_p$. If the system

is incompatible then three of the four points $P_{p,1}, P_{p,2}, P_{p,3}, P_{p,4}$ will be collinear, and will sum to $\mathcal{O}$ on $E(\mathbb{F}_p)$, in which case we have solved the ECDLP.

Step 5

A quick step. Simply use the Chinese Remainder Theorem to find integers $u'_1, \ldots, u'_{10}$ satisfying

$$u'_i \equiv u_{p,i} \bmod p \quad \text{and} \quad u'_i \equiv u_{M,i} \bmod M \quad \forall\ 1 \leq i \leq 10$$

Step 6

We lift the chosen points to $\mathbb{P}^2(\mathbb{Q})$. Choose the points

$$P_i = [x_i, y_i, z_i], \quad 1 \leq i \leq r$$

with integer coordinates satisfying

$$P_i \equiv P_{p,i} \bmod p \quad \text{and} \quad P_i \equiv P_{M,i} \bmod M \quad \forall\ 1 \leq i \leq r. \tag{4.5}$$

Now, we can take $P_1 = [1, 0, 0], P_2 = [0, 1, 0], P_3 = [0, 0, 1]$ and $P_4 = [1, 1, 1]$. Also the congruences in (4.5) all take place in $\mathbb{P}^2$, and can be solved using the Chinese Remainder Theorem, then by an extended gcd-type algorithm to find all solutions with small coordinates [79].

Step 7

Let $\mathbf{B} = \mathbf{B}(P_1, \ldots, P_r)$ be the matrix of cubic monomials defined earlier, and consider the system of equations

$$\mathbf{Bu} = \mathbf{0}. \tag{4.6}$$

We now find a small integer solution $\mathbf{u} = [u_1, \ldots, u_{10}]$, to which (4.6) has the property that

$$\mathbf{u} \equiv [u'_1, u'_2, \ldots, u'_{10}] \mod M, \tag{4.7}$$

where $u'_1, u'_2, \ldots, u'_{10}$ are the coefficients that were computed in Step 5 of the algorithm.

Let $C_{\mathbf{u}}$ denote the associated cubic curve

$$C_{\mathbf{u}} : u_1x^3 + u_2x^2y + u_3xy^2 + u_4y^3 + u_5x^2z + u_6xyz + u_7y^2z + u_8xz^2 + u_9yz^2 + u_{10}z^3 = 0.$$

Now, by construction we have three facts

- formula (4.5) in Step 6 ensure that the points $P_1, \ldots, P_r$ are lifts of the original points $P_{p,1}, \ldots, P_{p,r}$.
- formula (4.7) in Step 7 ensure that the curve $C_{\mathbf{u}}$ is a lift of the original curve $E(\mathbb{F}_p)$
- formula (4.6) in Step 7 ensures us that $C_{\mathbf{u}}$ contains the points $P_1, \ldots, P_r$[79].

Thus the lifting problem no longer appears in the Xedni Calculus. Unfortunately, there's a new problem, namely trying to force the lifted points to be dependent. There is also a question of existence of a solution here. Notice that the existence of a solution for (4.6), satisfying (4.7), is guaranteed by condition 2 in Step 2 and a small algebraic lemma, see [79] for these details.

Step 8

We can now make a change of coordinates to put $C_{\mathbf{u}}$ in standard Weierstrass form using $P_1 = [1, 0, 0]$ as the point at infinity. The resulting equation is given as

$$E_{\mathbf{u}} : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6,$$

where $a_1 \ldots, a_6 \in \mathbb{Z}$. Let $Q_1, \ldots, Q_r$ denote the images of $P_1, \ldots, P_r$ under this change of coordinates, and $\Delta(\mathbf{u})$ be the discriminant of $E_{\mathbf{u}}$.

Note that there is a possibility that the coefficients of $C_{\mathbf{u}}$ are very large, so that there is a possibility of running through the rest of the algorithm without finding an explicit equation for $E_{\mathbf{u}}$.

Steps 9 & 10

Now these two sections appear as optional in Silverman's original algorithm, and I will omit them here, but not without a summary. They can be found in their entirety in [79].

Step 9 involves deriving a new curve $E_{\mathbf{v}}$ and computing the discriminant of this curve. If $|\Delta(\mathbf{v})|$ is smaller than $|\Delta(\mathbf{u})|$ then replace $\mathbf{u}$ by $\mathbf{v}$ and repeat. This would lead to an elliptic curve with a locally minimized discriminant.

Step 10 involves computing a sum for which we obtain a value that may allow us to return to Step 2 or 3, as we see fit. The idea of computing the sum is to give us the idea of the rank of the curve, and is based on the formulas due to Mestre. Silverman points out that if the sum is too small then we would discard the curve and return

to a previous step and search for another curve. This is what Silverman refers to as Reverse Mestre Conditions.

Step 11

We now have to check if the new points $Q_2, \ldots, Q_r \in E_{\mathbf{u}}(\mathbb{Q})$ are independent. If they are independent, we return to Steps 2 or 3. Otherwise we compute the relationship of dependence

$$n_2 Q_2 + n_3 Q_3 + \cdots + n_r Q_r = \mathcal{O}$$

and we can now set

$$n_1 = -n_2 - n_3 - \cdots - n_r$$

and we can continue on to the next step.

Before we continue to the next step, we make the observation that there are two ways to perform this step. The Descent Method chooses a set of primes, say $\mathcal{P}$, and looks at the map

$$E_{\mathbf{u}}(\mathbb{Q})/2E_{\mathbf{u}}(\mathbb{Q}) \to \prod_{p \in \mathcal{P}} E_{\mathbf{u}}(\mathbb{F}_p)/2E_{\mathbf{u}}(\mathbb{F}_p) \cong \mathbb{F}_2^K$$

and uses quadratic reciprocity and linear algebra over $\mathbb{F}_2$. The second method is the Height Method which computes the determinant of the canonical height regulator $\langle Q_i, Q_j \rangle_{2 \le i,j \le r}$, and uses LLL to find a linear relation on the columns. Silverman points out that this method may be the faster of the two, but may fail to terminate if the points are dependent [79, 10].

Step 12

We compute two values

$$s = \sum_{i=1}^{r} n_i s_i \quad \text{and} \quad t = \sum_{i=1}^{r} n_i t_i$$

Recall that the s_i's and the t_i's were chosen in Step 3. Now if the $\gcd(s, \#E(\mathbb{F}_p)) > 1$, we return to Steps 2 or 3. Otherwise we may compute an inverse of s modulo $\#E(\mathbb{F}_p)$. Then $ss' \equiv 1 \bmod \#E(\mathbb{F}_p)$ and we have that

$$\log_T(S) \equiv s't \bmod \#E(\mathbb{F}_p)$$

and the ECDLP is now solved. Of course we should still check to see if $S = [m]T$ in $E(\mathbb{F}_p)$ with $m \equiv s't \bmod \#E(\mathbb{F}_p)$. Since there is a possibility that the system in (4.6) had less than maximal rank modulo p, we could have arrived at an incorrect value for m.

3.4.3 Analysis and Conclusion

In experiments conducted at the University of Waterloo, shortly after the Xedni Calculus attack was announced, it was determined that the attack was impractical for large primes p used in elliptic curve cryptography. Properties of the canonical logarithmic height show that the coefficients in a dependency relation among the lifted points are bounded by an absolute constant [44], which implies a running time of $O(p)$. More can be said about this constant.

Theorem 4.6 *Under certain assumptions, there exists an absolute constant C_0, such that the probability of success of the Xedni algorithm in finding a discrete logarithm*

on an elliptic curve over $\mathbb{F}_p$ is less than C_0/p.

The problem was that the constant C_0 was fairly large. So that if both C_0 and p were large their ratio would be close to one and the algorithm could be worth implementing.

Lemma 4.5 *[44] Assume that* $\log|D| \geq C_1 max_{i=1,\ldots,r}\ \hat{h}(Q_i)$ *for the lifted curves in the Xedni algorithm, where D is the discriminant of the lifted curve, Q_i the lifted points, $\hat{h}$ is the canonical height logarithm, and C_1 is a positive constant. Then under Lang's conjecture, if the lifted points are dependent, they satisfy a nontrivial relation with coefficients bounded above by an absolute constant C_2.*

Using the lemma and the conjecture by Lang, the group working on determining the running time of this algorithm were able to show that the above theorem does hold. This shows that any relation among the lifted points Q_i can be reduced modulo p to get a relation among the original points $P_{p,i}$ that were constructed at random in Step 3, and that it is unlikely that the random points on $E(\mathbb{F}_p)$ will satisfy any linear relations with coefficients less than a certain constant bound [44]. Thus subject to various largely proved conjectures, the Xedni algorithm must be repeated at least $O(p)$ times in order to solve the ECDLP.

The group was also able to show that the probability of finding a lifting of points with dependency decreases as the discriminant of the curve increases. Thus taking p in the proper range for practical cryptographic purposes severely decreases the probability of finding such a lifting. It also turned out that when applying the

so called reverse Mestre conditions, the discriminant of the given curve increased drastically, doing more harm than good.

3.4.4 Further Results

A publication by Heon et. al. [9], produced an alternative algorithm to compute dependencies among the lifted points. They showed that if one could then lift the set of points to an elliptic curve over $\mathbb{Q}$ such that the curve had rank one, then the attack would be very efficient, and the ECDLP could readily be solved. Although the algorithm proved to be efficient, the road block of finding a lifting to a curve of rank one stood in their way. To determine the possibility of lifting to a curve of rank one, the authors used data from Brumer, about the rank distribution [47], with $|\Delta|$ prime. The results were a step in the right direction. If one were to take an arbitrary elliptic curve over $\mathbb{Q}$, one would expect it to have rank one. Unfortunately, since we were considering elliptic curves with large rational points, the rank of the lifted curve was generally higher than expected. On the other hand, if a curve has a non-trivial point of order two, its rank is bounded by the number of 'bad' primes [47]. The thought was then to use a theorem to possibly bound the rank of a curve.

Theorem 4.7 *Let*

$$E : y^2 = x(x^2 + ax + b), \quad a, b \in \mathbb{Z}$$

be an equation of an elliptic curve. Let $w(x)$ denote the number of distinct primes dividing x. Then

$$rank(E/\mathbb{Q}) \leq w(b) + w(a^2 - 4b) - 1$$

If a and b could be chosen such that b and $a^2 - 4b$ are prime, then $\text{rank}(E/\mathbb{Q}) \leq 1$. Thus choosing these quantities to have distinct prime factors, or as few as possible, can reduce the rank of the lifted elliptic curve. Unfortunately this idea seemed to terminate without any real conclusions[28].

It was also interesting to find that within articles [9] and [47], the authors were able to show that the lifting problem, lifting points on $E(\mathbb{F}_p)$ to $E(\mathbb{Q})$, for p an odd prime and $p = 2^n$ for some n, implies the ECDLP. Not only does the lifting problem imply the ECDLP, but it was also shown to imply the DLP and the Integer Factorization Problem [47, 10].

3.5 Semaev's Summation Polynomials

In this section we examine an idea present by Semaev in [71]. Although the approach is incomplete, the author was able to determine that this approach yields a solution to the ECDLP in polynomial time, and subexponential time for larger inputs.

We will assume that we wish to solve the following instance of the ECDLP. Let E be an elliptic curve defined over $\mathbb{F}_p$ of p elements by the equation

$$Y^2 = X^3 + AX + B. \tag{4.8}$$

Let $P, Q \in E(\mathbb{F}_p)$ such that $[n]P = Q$ for some $n \in \mathbb{Z}$.

The idea behind the Summation Polynomials is to find bounded solutions to explicit modular multivariate polynomials[29] [71]. So let E be an elliptic curve defined

[28] the authors conclude the article with a discussion of an ongoing experiment about a family of elliptic curves that contain at least two lifted points. Unfortunately they cannot determine which curves are of rank one [47].

[29] For a treatment of modualr functions, see [78].

over a finite field K of characteristic $\neq 2, 3$. For $n \geq 2 \in \mathbb{N}$, we define the polynomial $f_n = f_n(X_1, X_2, \ldots, X_n)$ in n variables which will be related to the group operations on E. f_n will be defined by the following properties: let $(x_1, x_2, \ldots, x_n) \in \overline{K}$, then $f_n(x_1, x_2, \ldots, x_n) = 0$ iff there exists $y_1, y_2, \ldots, y_n \in \overline{K}$ such that (x_i, y_i) are on E and $(x_1, y_1) + (x_2, y_2) + \cdots + (x_n, y_n) = \mathcal{O}$ on $E(\overline{K})$. The polynomials f_n are then what are called the Summation Polynomials.

The following theorem defines and lists further properties that the Summation Polynomials have. The proof is omitted here, but can be found in [71].

Theorem 4.8 *The polynomial f_n may be be defined by $f_2(X_1, X_2) = X_1 - X_2$, and*

$$f_3 = (X_1 - X_2)^2 X_3 - 2((X_1 + X_2)(X_1 X_2 + A) + 2B)X_3 + ((X_1 X_2 - A)^2 - 4B(X_1 + X_2)),$$

where A and B are coefficients of (4.8), and

$$f_n(X_1, X_2, \ldots, X_n) = Res_X(f_{n-k}(X_1, \ldots, X_{n-k-1}, X), f_{k+2}(X_{n-k}, \ldots, X_n, X)),$$

for any $n \geq 4$ and $n - 3 \geq k \geq 1$.

The polynomial f_n is symmetric and of degree 2^{n-2} in each variable X_i for any $n \geq 3$.

The polynomial f_n is absolutely irreducible and

$$f_n(X_1, \ldots, X_n) = f_{n-1}^2(X_1, \ldots, X_{n-1}) X_n^{2^{n-2}} + \ldots,$$

for any $n \geq 3$.

Now, let us look at what happens when we use these polynomials in our attempt to solve the ECDLP from above. Fix $n \geq 2$. Let $R = (x, y) = [w]P + [v]Q$, for some

random $w, v \in \mathbb{Z}$. Now consider the equation

$$f_{n+1}(x_1, \ldots, x_n, x) \equiv 0 \mod p \tag{4.9}$$

in variables $x_1, \ldots, x_n$. Then with high probability [71], (4.9) has a solution, say $x_1^0, \ldots, x_m^0$, where x_i^0 are integers bounded by $p^{\frac{1}{n}+\delta}$ for some $\delta > 0$ or x_i^0 are rational numbers where the numerator and the denominator are bounded by $p^{\frac{1}{2n}+\delta}$. This would then imply that we have found a relation

$$(x_1^0, y_1^0) + \cdots + (x_n^0, y_n^0) = [w]P + [v]Q \tag{4.10}$$

for some $y_1^0, \ldots, y_n^0 \in \mathbb{F}_p$ or $\mathbb{F}_{p^2}$.

We could then combine the relations from (4.10) with the relation from a second summation polynomial, say f_m, with $m \geq n$, which yields $(x_1, y_1) + \ldots (x_m + y_m) = \mathcal{O}$, according to Theorem 4.8, and

$$f_m(x_1, \ldots, x_m) \equiv 0 \mod p. \tag{4.11}$$

Thus finding a bounded solution to both (4.11) and (4.10) yields a solution to the ECDLP over on $E(\mathbb{F}_p)$. The overall complexity of this would be

$$t_{p,n} p^{\frac{1}{n}+\delta} + p^{\frac{2}{n}+2\delta},$$

where $t_{p,n}$ is the time complexity for finding a bounded solution to both (4.11) and (4.10) [71].

There are a few unanswered questions that we must address.

1. One should avoid the trivial solutions to both (4.11) and (4.10) like $x_1, x_1, , x_2, x_2 \ldots, x_k, x_k$, which is always a solution to (4.11), where $m = 2k$ [71].

2. There is concern about a solution y_i^0 being in $\mathbb{F}_{p^2}$. Suppose that this is the case. Then sum of all such points in (4.10) is a point of order two in E, and so being in $\mathbb{F}_{p^2}$ is not important.

3. How do we find a solution to (4.9)? As of now, no such algorithm exists. It would be interesting to see if one could describe an algorithm that would solve such a system of equations. Also, since one needs about $p^{\frac{1}{n}+\delta}$ nontrivial solutions in order to solve the ECDLP [71], one would expect that this could lead to difficulty in solving the ECDLP in a satisfactory amount of time.

4. Lastly the value for $t_{p,n}$ is unknown. Since the question above is not fully answered, then neither is the value of its expected running time. However, there exists modular multivariate polynomials for which a bounded solution may be found in polynomial time or even in subexponential time [71]. This then gives hope that an algorithm may be produced to yield solutions to (4.11) and (4.10) which would yield a very good time complexity for an overall algorithm. The authors of [60] and [71] both speculate that this may produce an algorithm whose expected running time is faster than Pollard's methods.

3.6 An Index Calculus for Abelian Varieties

This section explores a recent development in methods of attacking the ECDLP. This attack is the first of its kind: it is the first known algorithm that solves the ECDLP in subexponential time. All other subexponential running algorithms have transfered the ECDLP to another problem, either the DLP or HCDLP, and have not directly

solved the ECDLP in subexponential time.

This attack will use a few ideas that we have already seen. In particular it relies on Gröbner basis and Resultant calculations. When we transfer the idea from the general case to the specific case of an elliptic curve, an abelian variety of dimension one, we will use Semaev's Summation Polynomials to ease some calculations[30].

The main result of this section is the following:

Theorem 4.9 *[36] Let n be a fixed integer and let q be a prime or prime power which grows to infinity. There exists an algorithm that can solve a discrete logarithm problem on any elliptic curve over a finite field with q^n elements in time $O(q^{2-2/n})$ up to constant logarithmic factors in q and where the constant depends on n.*

Unfortunately the hidden constant in the big-O notation depends on n and grows very rapidly as n increases, so in particular this attack is applicable only for elliptic curves defined over small extension fields. Hence in practice we can avoid this attack by choosing to use elliptic curves over $\mathbb{F}_{2^p}$ for $p \in [160, 600]$ where p is prime and $\mathbb{F}_p$ for large primes p.

We first give a description of this process in the general sense on any abelian variety, then give the explicit description relating to elliptic curve, followed by a complexity analysis.

Let $\mathcal{A}$ be an abelian variety with P and Q on $\mathcal{A}$ and Q multiple of P. We start by computing linear combinations $R = [a]P + [b]Q$ for random integers a and b bounded by the order of the subgroup generated by P. We will attempt to decompose R on

[30]Recall that although the polynomials were defined, there was no full algorithm to attack the ECDLP.

a factor base using a Gröbner basis calculation. If we get a solution, then we store it as a relation. It is possible to get more then one solution for a single R, in this case we simply get more relations. After having collected more relations than the cardinality of the factor base, we use some linear algebra on the relations in the hope that we generate a non-trivial linear combination of P and Q. When we obtain this nontrivial linear combination we can solve the ECDLP.

There are now three things we must describe a little more in detail. First, we will need to know how to perform operations on $\mathcal{A}$; this requires an explicit description of $\mathcal{A}$. Second, we need to know how to define a factor base, and determine its order. Lastly, we need to know how we decompose a point on the factor base. We now describe these ideas in more detail.

3.6.1 A Representation

Let $\mathcal{A}$ be an abelian variety of dimension n defined over $\mathbb{F}_q$. We will work with an explicit embedding in the affine plane of dimension $n+m$. $P \in \mathcal{A}$ can be represented by $n+m$ coordinates, $P = (x_1, \ldots, x_n, y_1, \ldots, y_m)$, where $x_i, y_i \in \mathbb{F}_q$. We can do this for almost all points in $\mathcal{A}$, we can also assume that for each choice of $x_1, \ldots, x_n \in \overline{\mathbb{F}_q}$ there exists only finitely many $y_1, \ldots, y_m \in \overline{\mathbb{F}_q}$ such that these $n+m$-tuples yield a point in $\mathcal{A}$ [36].

The coordinates (x_i, y_i) of a point on $\mathcal{A}$ will satisfy some equations which form a triangular set: the first equation being a polynomial in y_1 and the x_i's, the second in y_1, y_2 and the x_i's, and so on until the last equation is a polynomial in all coordinates. This system has m equations and locally defines $\mathcal{A}$ [36].

3.6.2 The Factor Base

We define the factor basis as follows:

$$\mathcal{F} = \{P \in \mathcal{A} \cap H_1 \cap H_2 \cap \ldots \cap H_n \mid P \in \mathbb{F}_q\}$$

where H_i is the hyperplane of the equation $x_i = 0$.

Then $\mathcal{F} = \{(x_1, 0, 0, \ldots, 0, y_1, \ldots, y_m) \mid x_1, y_i \in \mathbb{F}_q\}$ is an algebraic variety of dimension one, and is a non-empty union of curves [36]. The number of curves and their genus are bounded, independently of q, by the degrees of the y_i's in the triangular set of equations for $\mathcal{A}$. We also know how many elements are in $\mathcal{F}$. From Weil's bound, $\#\mathcal{F} = q + O(\sqrt{q})$ [36].

3.6.3 Decomposing a Point

To decompose a point over $\mathcal{F}$, we need to answer the following questions: *Let P be a point on $\mathcal{A}$. Are there points $P_1, P_2, \ldots, P_n \in \mathcal{F}$ such that*

$$P = \sum_{i=1}^{n} P_i,$$

and how do we compute all the solutions?

Let $\mathcal{G}_n$ be the n^{th} symmetric group, and define the map $f : \mathcal{F}^n/\mathcal{G}_n \to \mathcal{A}$ by $f : (P_1, P_2, \ldots, P_n) \mapsto \sum_{i=1}^{n} P_i$. Hence the n^{th} symmetric group is acting on points in our factor base.

Since the group law on $\mathcal{A}$ is defined by rational functions in terms of the coordinates there exists $n + m$ explicit rational functions such that

$$\sum_{i=1}^{n} P_i = (\varphi_1(P_1, P_2, \ldots, P_n), \ldots, \varphi_{n+m}(P_1, P_2, \ldots, P_n))$$

The net result of this map is a system of more equations than unknowns, and will generally have a finite number of solutions over $\overline{\mathbb{F}_q}$.

For a given point P, finding all these solutions can be done via a Gröbner basis calculation, followed by a factorization of a univariate polynomial, whose degree is bounded by the degree of the ideal defined by all the equations in the system [36].

3.6.4 Overall Complexity

Notice that when we decompose a point P, we are simply computing the number of pre-images $f^{-1}(P)$. The expected number of pre-images is

$$\sum_{P \in \mathcal{A}} \frac{\#f^{-1}(P)}{\#\mathcal{A}} = \frac{1}{\#\mathcal{A}} \#(\mathcal{F}^n/\mathcal{G}_n).$$

By using the estimate that $\#\mathcal{A} \approx q^n$, we get that the expected number of relations is approximately $\frac{1}{n!}$ [36].

We can now look at the overall complexity of this algorithm, at least if we assume that the parameters of $\mathcal{A}$ remain fixed and q tends to infinity. Notice that the point decomposition process can be done in polynomial time in $\log(q)$. This is due to the fact that we need to check that the ideal obtained from this process is of dimension zero and Gröbner basis computations can be done in the size of $\mathbb{F}_q$ [36]. We also need around $O(q)$ operations to collect the $\frac{1}{n!}$ relations which were constant since the parameters of $\mathcal{A}$ were constant, and solving the relations for a solution takes $O(q^2)$ operations using Lanczo's methods for sparse linear algebra systems [36].

Thus the complexity can be deduced as being $O(q^{2-2/n})$ [36]. Notice that for $n = 3$ we have that a DLP can be solved in $O(q^{4/3})$ compared to Pollard's Rho which

yields $O(q^{3/2})$.

3.6.5 Transfer to Elliptic Curves

The above procedure may not be completely obvious in the general setting so we transfer the case to the specific setting of elliptic curves. The methods discussed in previous sections, III.4, III.5, IV.3.5, and IV.1.3 will all be used in this attack.

Let E be an elliptic curve over $\mathbb{F}_{q^n}$ given by the equation $y^2 = x^3 + ax + b$. We choose an explicit basis representation for the elements in $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$; in other words we select a monic irreducible polynomial over $\mathbb{F}_q$ so that $\mathbb{F}_{q^n} = \mathbb{F}_q[t]/(f(t))$. We then form the Weil Restriction $\mathcal{A}$ of E as the set of $2n$-tuples of elements $(x_0, \ldots, x_{n-1}, y_0, \ldots, y_{n-1})$ in $\mathbb{F}_q$ such that $x = x_0 + x_1 t + \ldots + x_{n-1}t^{n-1}$ and $y = y_0 + y_1 t + \ldots + y_{n-1}t^{n-1}$ are the coordinates of a point in E. Notice that since the group law is inherited from E, $\mathcal{A}$ is indeed an abelian variety. The factor base will contain points that are on E whose x-coordinate lie in $\mathbb{F}_q$, that is $\mathcal{F} = \{P = (x, y) \in E \mid x \in \mathbb{F}_q\}$.

To decompose over the factor base we must write down a very large system of equations and solve it using a Gröbner basis calculation. The decomposition step is made easier with Semaev's Summation Polynomials. Recall their definition from Section 3.5. Let R be a point of E, which we want to write as a sum of $P_1, \ldots, P_n$, whose x-coordinate is in $\mathbb{F}_q$. We denote this as $x_P = x_0 + x_1 t + \ldots + x_{n-1}^{t-1}$, which in turn requires us to solve

$$f_{n+1}(x_{P_1}, x_{P_2}, \ldots, x_{P_n}, x_R) = 0$$

where x_R is known. We now rewrite this equation allowing t to enter the game, and we reduce modulo $f(t)$ to obtain an equation of the form

$$\sum_{i=1}^{n-1} \varphi_i(x_{0,P_1}, \ldots, x_{0,P_n})t^i = 0$$

which gives us n equations in n indeterminates [36]. We then apply Buchburger's algorithm to find solutions to this system.

If we find a solution defined over $\mathbb{F}_q$ then we simply look for rational roots of the corresponding polynomial to find the x-coordinate for P_i.

Below is an example of how this process works. This example can also be found in [36].

Example: Let $p = 1019$. The polynomial $f(t) = t^2+1$ is irreducible over $\mathbb{F}_p$, thus $\mathbb{F}_{p^2} \cong \mathbb{F}_p[t]/(t^2+1)$; that is the quotient of the polynomial field $\mathbb{F}_p[t]$ and the ideal generated by the irreducible polynomial $f(t)$. Define E over $\mathbb{F}_{p^2}$ by $y^2 = x^3 + ax + b$ with $a = 214 + 364t$ and $b = 123 + 983t$. Note here that it can be shown that E has prime order 1039037. Let $P = (401+517t, 885+15t)$ and $Q = (935+210t, 740+617t)$. We now want to solve the ECDLP $[\lambda]P = Q$.

To define the factor base, we let $\mathcal{F}$ be the set of points whose x-coordinate lie in $\mathbb{F}_p$ (note that this follows with our definition of the factor base from above). This factor base can be shown to have 1011 elements. Now we test random linear combination of P and Q to see if they can be written in terms of elements of $\mathcal{F}$. This is where we will make use of Semaev's summation polynomials to ease calculations. Suppose that we computed $R = [459328]P + [313814]Q = (415 + 211t, 183 + 288t)$. If $R = P_1 + P_2$ for points $P_1, P_2 \in \mathcal{F}$ then by Semaev's Summation polynomials we

have that $f_3(x_1, x_2, x_R) = 0$ (this is simply checking that the sum of these points is the point at infinity). If we define $m = x_1 + x_2$ and $n = x_1 x_2$ we obtain the following equation:

$$(m^2 - 4n)x_R^2 - 2(mn + am + 2b)x_R + a^2 + n^2 - 2an - 4bm = 0.$$

Hence we have an equation that relates quantities in $\mathbb{F}_{p^2}$ and has two unknowns which lie in $\mathbb{F}_p$. We now use our knowledge of the structure of $\mathbb{F}_{p^2}$ to relate this into an equation over $\mathbb{F}_p$, that is we substitute for a and b and reduce modulo $f(t)$. When we do this we obtain the following equation,

$$(881m^2+597mn+31m+843n+669)t+(329m^2+189mn+971m+n^2+294n+740) = 0.$$

For this equation to hold true, the coefficients of t must be zero, hence we obtain two equations in two unknowns over $\mathbb{F}_p$. We can solve this system using a Gröbner basis calculation which yields the solution $(m, n) = (845, 1003)$. From this pair we can solve for x_1 and x_2 by solving the relation$(x - x_1)(x - x_2) = x^2 - mx + n$. From this we find that $x_1 = 92$ and $x_2 = 753$. Once we have these, we can determine the corresponding y values for our points P_1 and P_2 in our factor base. Thus we find that

$$P_1(92, 779 + 754t) \quad \text{and} \quad P_2 = (753, 628 + 629t).$$

We now have to repeat this process until we obtain more relations than elements in our factor base; that is until we have 1012 relations collected. After producing this many relation we solve a linear algebra system to get a non-trivial combination of points P and Q that is zero, which allows us to solve the ECDLP. After solving such a system we determine that $\lambda = 76982$, hence $[76982]P = Q$ as required.

3.6.6 Conclusion

The algorithm presented here has limitations that can trivially be avoided when constructing cryptosystems for everyday use. Since this algorithm has a running time of $O(q^{2-2/n})$ for a small fixed n, we can simply rule out the possibility of using elliptic curves defined over small extension fields. Notice that in the case of elliptic curves since we used Semaev's Summation Polynomials, which were of degree 2^{n-2}, the hidden constant in the big-O notation depends very badly on n, hence as n grows this attack becomes impractical.

3.7 Conclusions

Below is a table summarizing the results obtained for the specialized attacks. The Anomalous Curve case runs in polynomial time while the rest of the attacks run in subexponential time, with the exceptions being the Xedni Calculus and the Summation polynomial attack. Each attack however can be addressed and eluded quite easily when constructing cryptographically strong elliptic curve cryptosystems, the subject of our next chapter. Almost immediately we can determine a pattern for good and bad curves: $\#E(\mathbb{F}_p)$ should not be close to p, and if elliptic curves are to be defined over an extension field, either $\mathbb{F}_{2^m}$ or $\mathbb{F}_{p^m}$ then m must be sufficiently large, and preferably prime.

Attack	Expected Running Time	Can be Applied When
Anomalous Curves	$O(\ln p)$	$\#E(\mathbb{F}_p) = p$
MOV/Frey-Rück	$L_q[\frac{1}{2}, c]$	$\#E(\mathbb{F}_{p^m}) = p^m + 1 - t$ where $p \mid t$ and $n \mid p^{mk} - 1$, $1 \leq k \leq 6$
Weil Descent	$O(q^{\frac{n}{4}+\epsilon})$	$\mathbb{F}_{q^m}$, m not prime
Xedni Calculus	$O(p)$	always, but not feasible
Semaev's Summation Polynomials	$O(t_{p,n}p^{\frac{1}{n}+\delta} + p^{\frac{2}{n}+2\delta})$	algorithm incomplete
Index Calculus for Abelian Varieties	$O(q^{2-2/n})$	when E is define over $\mathbb{F}_{q^n}$ with small n

Table 4.4: Expected Running Times of the Specialized Attacks on the ECDLP

5 Generating Cryptographically Strong Elliptic Curves

1 Introduction

With several attacks on elliptic curve cryptosystems having been discussed and analyzed we now turn our attention to constructing cryptographically strong elliptic curves. Notice we did not use the term cryptographically secure. This is mainly due to the fact that new attacks continue to be developed to attack these systems by exploiting various properties of elliptic curves. To this point we can simply hope to generate strong curves that resist all known attacks; there is nothing that guarantees that they will resist future attacks.

There are currently two main methods in generating elliptic curves for use in cryptography. The first is simply generating curves at random then running them through a series of tests to see if they satisfy certain properties. The second is a method that is called Complex Multiplication(CM). This second method builds a specific elliptic curve with certain properties already built in.

In addition we remind the reader what we have done so far. The attacks that we have just analyzed now defined certain security constraints that we must adhere to

in order to ensure a cryptographically strong elliptic curve. From the general attacks we see that $\#E(\mathbb{F}_q)$ must be divisible by a large prime l, with $l > 2^{160}$. This provides maximum resistance against both the Pohlig-Hellman attack and Pollard's methods. Also from the specialized attacks, we know that $\#E(\mathbb{F}_q) \neq q, q+1$ which avoids the Anomalous curve attack and the MOV, and in general to avoid the Frey-Rück attack we should make sure that l does not divide $q^k - 1$ for $1 \leq k \leq 20$. This condition ensures that the DLP in $\mathbb{F}_{q^k}^{\times}$ is intractable. To avoid the GHS attack and Gaudry's Abelian Variety attack, we could simply choose a curve over $\mathbb{F}_{2^p}$ for a prime $p \in [160, 600]$.

The Xedni Calculus, and Semaev's Summation Polynomials do not currently apply. The Xedni Calculus was shown to have an expected running time of $O(p)$ for the prime p in question, which for cryptographic purposes makes this an exponential running time. Semaev's attack is, as of today, incomplete. Although the algorithm has a promising expected running time, there is no full algorithm to solve the ECDLP here. Hence we do not concern ourselves with these attacks.

With our security parameters fully understood, we now turn to methods of generating cryptographically strong elliptic curves.

2 Generating Curves at Random

The main idea here is that we are going to somehow generate an elliptic curve at random that will hopefully satisfy certain properties. The properties that we are interested in are based on our security parameters defined above. Thus generating

an elliptic curve for cryptographic purposes involves two stages:

1. generation process
2. verification process

We first have to generate the curve with some method, we then need to check that the curve satisfies various properties as to ensure that our security parameters above are met.

There are several ways to go about this process depending on the underlying field chosen. We present two algorithms from [38], which will generate a curve over a prime field and a binary field. In both algorithms || denotes concatenation.

Algorithm 5.1 Generating Random Elliptic Curves over $\mathbb{F}_p$

Input: A prime $p > 3$ and an l-bit hash function H
Output: A seed S, and $a, b \in \mathbb{F}_p$ defining $E : y^2 = x^3 + ax + b$.

1: Set $t \leftarrow \lceil \log_2 p \rceil, s \leftarrow \lfloor (t-1)/l \rfloor, v \leftarrow t - sl$.
2: Select an arbitrary bit string S of length $g \geq l$.
3: Compute $h = H(S)$, and let r_0 be the bit string of length $v$obtained by taking the v rightmost bits of h.
4: Let R_0 be the bit string obtained by setting the leftmost bit of r_0 to 0.
5: Let z be the integer whose binary representation is S.
6: For i from 1 to s do:

   1. Let s_i be the g-bit binary representation of the integer $(z + i) \mod 2^g$
   2. Compute $R_i = H(S_i)$

7: Let $R = R_0 || R_1 || \ldots || R_s$
8: Let r be the the integer whose binary representation is R.
9: If $r = 0$ or $4r + 27 \equiv 0 \mod p$ then go to step 2.
10: Select arbitrary $a, b \in \mathbb{F}_p$ not bot zero, such that $rb^2 \equiv a^3 \mod p$.
11: Return (S, a, b)

The condition in step 9 of this algorithm ensures that we do not generate a singular elliptic curve.

Generating elliptic curves over $\mathbb{F}_{2^m}$ are equally if not more important, since operation in $\mathbb{F}_{2^m}$ can be performed very efficiently. In fact the original algorithm presented in [38] is an algorithm for an arbitrary, but sufficiently, large integer for cryptographic purposes. Instead we modify the algorithm to eliminate any possibility of applying the GHS or Gaudry's attack to such a curve, by taking m to be a prime greater than 160.

Algorithm 5.2 Generating Random Elliptic Curves over $\mathbb{F}_{2^p}$

Input: A prime number $p > 160$, and an l-bit hash function.
Output: A seed S, and $a, b \in \mathbb{F}_{2^p}$ defining $E : y^2 + xy = x^3 + ax^2 + b$.

1: Set $s \leftarrow \lfloor (m-1)/l \rfloor, v \leftarrow m - sl$.
2: Select an arbitrary bit string S of length $g \geq l$.
3: Compute $h = H(S)$, and let b_0 be the bit string of length $v$obtained by taking the v rightmost bits of h.
4: Let z be the integer whose binary representation is S.
5: For i from 1 to s do:

   1. Let s_i be the g-bit binary representation of the integer $(z + i) \mod 2^g$

   2. Compute $b_i = H(s_i)$

6: Let $b = b_0 || b_1 || \ldots || b_s$
7: If $b = 0$ then go to step 2.
8: Select arbitrary $a \in \mathbb{F}_{2^p}$.
9: Return (S, a, b)

As a part of this process, at least from the point of view of a person who receives an elliptic curve that was supposedly generated at random, we should have a verification process to test that the curve was indeed generated at random. This step is essential to avoid some attacks, which are not necessarily mathematical on the ECDLP, but could be deployed in practical situations. It could be possible to act as an oracle and feed someone an elliptic curve for which the adversary knows a solution to the ECDLP.

Without checking to see if the curve was generated at random anyone employing that curve for cryptographic purposes would be using an insecure curve, and the adversary could recover any information that he or she wishes. Algorithms to check, in both cases, that the curves have been indeed generated at random can be found in [38].

Note that we also require a random number generator for each of these algorithms. For Algorithm 5.1 we need to generate a large prime number suitable for cryptographic purposes, larger than 2^{160}, and in step 10 we need to select uniformly at random $a, b \in \mathbb{F}_p$. The same is also true for Algorithm 5.2, we need to generate a prime $p > 160$ and $a \in \mathbb{F}_{2^p}$.

Suppose now that we have generated a curve $E(\mathbb{F}_p)$ or $E(\mathbb{F}_{2^p})$. We need to check that this randomly generated curve now fits in with our security parameters. Suppose that we are dealing with a curve defined over $\mathbb{F}_p$, the case where E is defined over $\mathbb{F}_{2^p}$ is entirely similar, except that different algorithms will be employed to deal with the characteristic 2 situation. The first thing to do would be to calculate $\#E(\mathbb{F}_p)$[31]. If $\#E(\mathbb{F}_p)$ is equal to $p, p+1$ or divisible by small primes, then we reject the curve. Otherwise we continue along with our check and ensure that $\#E(\mathbb{F}_p) \nmid p^k - 1$ for $1 \leq k \leq 20$, as to avoid the MOV and the Frey-Rück attacks. Once all these checks are performed and assuming that the curve $E(\mathbb{F}_p)$ has passed all checks, including verified as being generated at random, then the elliptic curve has satisfied our security parameters and is cryptographically strong.

[31] Or we can simply obtain an estimate using the Hasse-Weil Theorem and only fully count the points on E once it has passed all other checks.

Notice also here that these checks could be easily paralleled on several processors to yield a quicker generation time. Since each check is independent of the others the parallelization process is trivial. As mentioned, $\#E(\mathbb{F}_p)$ could first be estimated until all other checks have been passed. Once this happens we could then subject $E(\mathbb{F}_p)$ to a full point counting algorithm. This version of the parallelized check system would then be governed by the time it would take to run the fastest point counting algorithms[32].

3 The Method of Complex Multiplication(CM)

The method of Complex Multiplication requires a few results about elliptic curves over $\mathbb{C}$, and some results about class field theory. We introduce only necessary topics to understand this method of generating elliptic curves. A greater exposition of elliptic curves over the complex numbers can be found in [2], [38], [78] and [87].

In Chapter III we talked about isogenies from elliptic curves E_1 to E_2. We could have also talked about the set of isogenies on an elliptic curve E to itself. These maps are commonly known as *endomorphisms*; the set of endomorphisms on E along with the zero map form a ring which we will denote $\mathrm{End}(E)$. There are three possibilities when it comes to the structure of $\mathrm{End}(E)$:

1. $\mathrm{End}(E) = \mathbb{Z}$: although this does not occur over finite fields,
2. $\mathrm{End}(E)$ is an order in an imaginary quadratic field; which we explain below,
3. $\mathrm{End}(E)$ is the maximal order in a quaternion algebra; a case we do not concern ourselves with.

[32]This author would be interested to see if these point counting algorithms could be parallelized to speed up this process. I currently do not know of any point counting algorithms that run in parallel for elliptic curves.

A proof that $\mathrm{End}(E)$ has these possible structures can be found in [77]. However it can be easily seen that $\mathrm{End}(E)$ contains $\mathbb{Z}$. Since the multiplication by n map is an isogeny of E to itself it is therefore an endomorphism of degree n^2 [5]. Thus we trivially have $\mathbb{Z} \subseteq \mathrm{End}(E)$. We now take a closer look at the structure of $\mathrm{End}(E)$ in the second case.

Definition 5.1 *Let $E(\mathbb{C})$ be an elliptic curve. E is said to have Complex Multiplication if $End(E)$ is strictly larger then $\mathbb{Z}$. If $End(E)$ is larger then $\mathbb{Z}$, then it is an order in an imaginary quadratic field.*

We explain this last sentence a little further. Suppose that $d > 0$ is a square free integer. Let

$$K = \mathbb{Q}(\sqrt{-d}) = \{a + b\sqrt{-d} \mid a, b \in \mathbb{Q}\}.$$

Then K is an imaginary quadratic field. We define the largest subring of K that is also a finitely generated abelian group as

$$\mathfrak{O}_K = \begin{cases} \mathbb{Z}[\frac{1+\sqrt{-d}}{2}], \text{ if } d \equiv 3 \mod 4 \\ \mathbb{Z}[\sqrt{-d}], \text{ if } d \equiv 1, 2 \mod 4 \end{cases}$$

An *order* in an imaginary quadratic field is a ring R such that $\mathbb{Z} \subset R \subset \mathfrak{O}_K$. Notice that R is finitely generated as an abelian group and has the form of $R = \mathbb{Z} + \mathbb{Z}f\delta$ where $f > 0$ and δ is one of the forms above [87]. f is called the conductor of R and is the index of R in $\mathfrak{O}_K$. The *discriminant* of R is

$$D = \begin{cases} -f^2 d, \text{ if } d \equiv 3 \mod 4 \\ -4f^2 d, \text{ if } d \equiv 1, 2 \mod 4. \end{cases}$$

The last concept that we need is an essential tool used in the method of Complex Multiplication.

The minimal polynomial of $j(E)$, is the Hilbert class polynomial[33]

$$H_d(X) = \prod_{r=1}^{h_d}(X - j(A_r)),$$

where $j(A_r)$ is the j-invariant of the elliptic curves corresponding to the representatives A_r in the class group $\mathfrak{O}_K$, and h_d is the order of the ideal class group in $\mathfrak{O}_K$.

To generate curves using the CM method we first select an order N suitable for cryptographic purposes, we then construct an elliptic curve with that order. This method is very efficient provided that the finite field order q and the elliptic curve order $N = q + 1 - t$ are chosen so that the field $\mathbb{Q}(\sqrt{t^2 - 4q})$ has small class number [38].

Given a Hilbert class polynomial, we can reduce it modulo primes, l, which correspond to the product of principal primes in $\mathfrak{O}_K$ [2] which can be factored. We then obtain a j_l-invariant corresponding to this which results in an elliptic curve E_l defined over $\mathbb{F}_p$. It is this curve E_l, or one of its quadratic twists[34] we will use for our curve. Recall that for any given element j an elliptic curve with j-invariant, $j \neq 0, 12^3$ is isomorphic to

$$E_j : y^2 = x^3 - \frac{27j}{4(j - 12^3)}x + \frac{27j}{4(j - 12^3)}.$$

Hence once we know that value of j_l we can construct E_l quite easily.

[33]See [2] for more details concerning Hilbert Class polynomials

[34]A quadratic twist $\tilde{E}$ of an elliptic curve E is a curve isomorphic to E over a field extension which depends on the equation for E. See [2, 71] for more details of these forms.

The following algorithm to generate an elliptic curve that has Complex Multiplication is given below. The algorithm is from [2]. Here $\tilde{E_j}$ denotes a quadratic twist of E_j.

Algorithm 5.3 Generating Elliptic Curves via CM

INPUT: A squarefree integer $d \neq 1, 3$ parameters ϵ and δ, Hilbert class polynomial $H_d(X)$ the desired size of p and properties Pr.

OUTPUT: A prime p of the desired size, and elliptic curve $E/\mathbb{F}_p$ whose group order satisfies property Pr.

1: repeat
2: repeat chose p prime of desired size
3: until $\epsilon p = x^2 + dy^2$ with $x, y \in \mathbb{Z}$
4: $n_1 \leftarrow p + 1 - 2x/\delta$ and $n_2 \leftarrow p + 1 + 2x/\delta$
5: until n_1 or n_2 satisfies property Pr
6: compute a root j of $H_d(X)$
7: compute $E_j/\mathbb{F}_p$ from 3 and its twist $\tilde{E}_j/\mathbb{F}_p$
8: while true do
9: take $P \in E_j/\mathbb{F}_p$ uniformly at random and compute $Q \leftarrow [n_1]P$
10: if $Q = \mathcal{O}$ and $[n_2]P \neq \mathcal{O}$ then return p, E_j
11: else if $Q \neq \mathcal{O}$ then return $p, \tilde{E}_j$

The properties Pr in the above algorithm are the properties that we need to achieve our security parameters defined at the beginning of this chapter. The largest time-consuming step in this algorithm is to compute a root of the Hilbert class polynomial, but this only needs to be done once. A quick modification of this algorithm can be made at the outset: we can set $p = \frac{(x^2+dy^2)}{\epsilon}$. This ensures that p will split in $\mathfrak{O}_K$ [2].

4 Random Curves versus The CM Method

Both methods each have their advantages. The CM method generates a curve with a given order, remarkably fast. In fact a CM curve over a 160-bit field can be generated in about one minute [38], which is much faster than generating a curve at random and running it through the required security parameter tests. However there are some who believe that it would be best to use a curve that has been generated at random. The thought is that there could be possible attacks that exploit the fact that a CM curve has a small class number [5]. As of yet no such attack has been developed but there are those who feel that a small class number could be exploited to be used for a future attack.

As a result we recommend that a random curve be used for cryptographic purposes. Not only does it remove the doubt about a possible small number class attack, but the security parameter check could easily be paralleled. In the end all checks can be easily performed. The overall running time to then ensure that a curve would be cryptographically strong would clearly be equal to the time taken to compute $\#E(\mathbb{F}_q)$, which is terribly important since determining $\#E(\mathbb{F}_q)$ explicitly makes it possible to avoid three different attacks.

6 Conclusions and Future Work

In this document we analyzed the various techniques that are known to attack elliptic curve cryptosystems whose security is based around the ECDLP.

With the analysis of these attacks completed we have looked at the generation of what we called cryptographically strong elliptic curves. We have refrained from using the term secure, simply because we do not know if there will be an attack developed in the future that will make these curves unsuitable for use. As a result we now required binary elliptic curves to be generated over $\mathbb{F}_{2^p}$ for primes $p > 160$, instead of $\mathbb{F}_{2^m}$ for $m > 160$. This, as discussed above, eliminates the possibility of applying the GHS attack, and will also avoid the possibility of applying the Index Calculus attack by Gaudry. Of course, to avoid all other possible specialized attacks the security checks outlined in the previous chapter must be implemented.

Of all these attacks, several of them can be applied in special case scenarios based on certain properties of either the underlying curve or the underlying field. All of these attacks can easily be avoided when building an elliptic curve cryptosystem as we have shown above. As a result only the general purpose attacks will *always* apply, and so if one were to attack an elliptic curve cryptosystem at random the best attack would be to use Pollard's ρ or λ method; both of which have expected exponential running

times and hence are infeasible given today's technology. This suggests that elliptic curve cryptosystems are superior to currently deployed public key cryptosystems since not only do they offer a greater level of security when the underlying parameters are chosen correctly, but they offer a greater advantage due to factors mentioned in the outset of this document, including shorter key sizes, faster generation of systems, smaller space requirements and efficient implementation techniques.

Future Work

I believe more work should be done with the techniques of Weil Descent and the GHS attack. In this document we were able to show that there is no reason why we could not choose something else to intersect the abelian variety that results in this procedure with something other then hyperplanes in standard position. However, my belief is that the resulting equation could have degree that is too large, resulting in a curve, possibly hyperelliptic, with too large a genus to apply the index calculus algorithm in solving the HCDLP.

With Diem's new result in hand, it would be interesting to see if one could classify curves so that one would know what to expect as a result of the applying Weil Descent - a plane curve, a hyperelliptic curve, or something else? It would also be interesting to see if one could classify what types of curves result in a low genus curve after Weil Descent is applied. Results on these subjects could lead to larger classifications of weak curves resulting in a modification process in what curves are used in cryptography.

Combining the above ideas could lead to a larger classification of curves that could

be vulnerable to the GHS attack. If it is indeed possible to intersect the abelian variety with something other than hyperplanes in standard position, we could then apply the results of Diem in attempting to solve the ECDLP.

A second issue that should be examined in greater detail is the idea about a parallelized point counting algorithm. In this thesis we saw that point counting techniques were quite important when it came to generating cryptographically strong elliptic curves. Both methods involve computing $\#E(\mathbb{F}_q)$, thus a method to increase the speed at which the group order can be determined very desirable. If we examine Schoof's algorithm again, we can see that it could in fact be trivially parallelized. Recall that we created a list of primes up to a certain bound, at which point we calculated the number of points in $E[l]$ for various primes l less than the prescribed bound. Using the CRT the total number of points of $\#E(\mathbb{F}_q)$ is then calculated. A trivial parallelization of the algorithm would be to send each prime to a single processor so that a single processor compute $E[l]$ for a give prime l, then send the result to a central processor that can then use the CRT to reassemble $\#E(\mathbb{F}_q)$ when all processors have finished. The expected time on this would then be the time taken to compute the largest subgroup $E[l']$ for some l' in our list of prime less than the prescribed bound. Combining this result with the parallelized version of generating curves at random could result in an overall speedup which could be comparable to generating curves using the CM method. More work is needed in this area.

Bibliography

[1] M. F. Atiyah and I. G. Macdonald. *Introduction to commutative algebra*. Addison-Wesley Publishing Co., Reading, Mass.-London-Don Mills, Ont., 1969.

[2] Roberto M. Avanzi, Henri Cohen, Christophe Doche, Gerhard Frey, Tanja Lange, Kim Nguyen, and Frederik Vercauteren. *Handbook of Elliptic and Hyperelliptic Curve Cryptography*. CRC Press Series on Discrete Mathematics and its Applications. CRC Press, Boca Raton, FL, 2006. Scientific Editors Henri Cohen and Gerhard Frey.

[3] R. Balasubramanian and Neal Koblitz. The improbability that an elliptic curve has subexponential discrete log problem under the Menezes-Okamoto-Vanstone algorithm. *J. Cryptology*, 11(2):141–145, 1998.

[4] Paulo S. L. M. Barreto, Steven Galbraith, Colm O hEigeartaigh, and Michael Scott. Efficient pairing computation on supersingular abelian varieties. Cryptology ePrint Archive, Report 2004/375, 2004. `http://eprint.iacr.org/`.

[5] I. F. Blake, G. Seroussi, and N. P. Smart. *Elliptic curves in cryptography*, volume 265 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, Cambridge, 2000. Reprint of the 1999 original.

[6] I. F. Blake, G. Seroussi, and N. P. Smart. *Advances in Elliptic Curve Cryptography*, volume 317 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, Cambridge, 2005.

[7] Ian Blake, Kumar Murty, and Guangwu Xu. Refinements of miller's algorithm for computing weil/tate pairing. Cryptology ePrint Archive, Report 2004/065, 2004. http://eprint.iacr.org/.

[8] Qi Cheng and Ming-Deh Huang. On partial lifting and the elliptic curve discrete logarithm problem. Online at: http://www.cs.ou.edu/ qcheng/pub.html. The Proceeding of the 15th Annual International Symposium on Algorithms and Computation, 342–351, LNCS 3341.

[9] Jung Hee Cheon, Dong Hoon Lee, and Sang Geun Hahn. Elliptic curve discrete logarithms and wieferich primes.

[10] Jean-Marc Couveignes. Algebraic groups and discrete logarithm. In *Public-key cryptography and computational number theory (Warsaw, 2000)*, pages 17–27. de Gruyter, Berlin, 2001.

[11] David Cox, John Little, and Donal O'Shea. *Ideals, varieties, and algorithms*. Undergraduate Texts in Mathematics. Springer-Verlag, New York, 1992. An introduction to computational algebraic geometry and commutative algebra.

[12] David Cox, John Little, and Donal O'Shea. *Using algebraic geometry*, volume 185 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1998.

[13] L. Dewaghe. Remarks on the Schoof-Elkies-Atkin algorithm. *Math. Comp.*, 67(223):1247–1252, 1998.

[14] Claus Diem. The GHS attack in odd characteristic. *J. Ramanujan Math. Soc.*, 18(1):1–32, 2003.

[15] Claus Diem. Index calculus in class groups of plane curves of small degree. online at `http://www.exp-math.uni-essen.de/~diem/english.html`, April 2005. Preprint.

[16] David S. Dummit and Richard M. Foote. *Abstract algebra*. Prentice Hall Inc., Englewood Cliffs, NJ, 1991.

[17] I. Duursma, P. Gaudry, and F. Morain. Speeding up the discrete log computation on curves with automorphisms. In *Advances in cryptology—ASIACRYPT'99 (Singapore)*, volume 1716 of *Lecture Notes in Comput. Sci.*, pages 103–121. Springer, Berlin, 1999.

[18] David Eisenbud. *Commutative algebra*, volume 150 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1995. With a view toward algebraic geometry.

[19] K. Eisentraeger, K. Lauter, and P.L. Montgomery. Improved weil and tate pairings for elliptic and hyperelliptic curves. Online at: http://research.microsoft.com/ klauter/.

[20] Kirsten Eisenträger, Kristin Lauter, and Peter L. Montgomery. Fast elliptic curve arithmetic and improved Weil pairing evaluation. In *Topics in cryptology—CT-RSA 2003*, volume 2612 of *Lecture Notes in Comput. Sci.*, pages 343–354. Springer, Berlin, 2003.

[21] Andreas Enge and Pierrick Gaudry. A general framework for subexponential discrete logarithm algorithms. *Acta Arith.*, 102(1):83–103, 2002.

[22] G. Frey. How to disguise an elliptic curve, 1998. online at: http://www.cacr.math.uwaterloo.ca/conferences/1998/ecc98/slides.html.

[23] Gerhard Frey. Applications of arithmetical geometry to cryptographic constructions. In *Finite fields and applications (Augsburg, 1999)*, pages 128–161. Springer, Berlin, 2001.

[24] Gerhard Frey, Michael Müller, and Hans-Georg Rück. The Tate pairing and the discrete logarithm applied to elliptic curve cryptosystems. *IEEE Trans. Inform. Theory*, 45(5):1717–1719, 1999.

[25] Gerhard Frey and Hans-Georg Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Math. Comp.*, 62(206):865–874, 1994.

[26] William Fulton. *Algebraic curves. An introduction to algebraic geometry.* W. A. Benjamin, Inc., New York-Amsterdam, 1969. Notes written with the collaboration of Richard Weiss, Mathematics Lecture Notes Series.

[27] Steven D. Galbraith. Constructing isogenies between elliptic curves over finite fields. *LMS J. Comput. Math.*, 2:118–138 (electronic), 1999.

[28] Steven D. Galbraith. Limitations of constructive Weil descent. In *Public-key cryptography and computational number theory (Warsaw, 2000)*, pages 59–70. de Gruyter, Berlin, 2001.

[29] Steven D. Galbraith. Supersingular curves in cryptography. In *Advances in cryptology—ASIACRYPT 2001 (Gold Coast)*, volume 2248 of *Lecture Notes in Comput. Sci.*, pages 495–513. Springer, Berlin, 2001.

[30] Steven D. Galbraith, Keith Harrison, and David Soldera. Implementing the Tate pairing. In *Algorithmic number theory (Sydney, 2002)*, volume 2369 of *Lecture Notes in Comput. Sci.*, pages 324–337. Springer, Berlin, 2002.

[31] Steven D. Galbraith, Florian Hess, and Nigel P. Smart. Extending the GHS Weil descent attack. In *Advances in cryptology—EUROCRYPT 2002 (Amsterdam)*, volume 2332 of *Lecture Notes in Comput. Sci.*, pages 29–44. Springer, Berlin, 2002.

[32] Steven D. Galbraith and Nigel P. Smart. A cryptographic application of Weil descent. In *Cryptography and coding (Cirencester, 1999)*, volume 1746 of *Lecture Notes in Comput. Sci.*, pages 191–200. Springer, Berlin, 1999.

[33] Paul Garrett. *Making, Breaking Codes: An Introduction to Cryptology*. Prentice Hall, Upper Saddle River, NJ, 2001.

[34] P. Gaudry. Some remarks on the elliptic curve discrete logarithm. Online at: http://www.lix.polytechnique.fr/Labo/Pierrick.Gaudry/papers.en.html, November 2003.

[35] P. Gaudry, F. Hess, and N. P. Smart. Constructive and destructive facets of Weil descent on elliptic curves. *J. Cryptology*, 15(1):19–46, 2002.

[36] Pierrick Gaudry. Index calculus for abelian varieties and the elliptic curve discrete logarithm problem. Preprint, October 26 2004.

[37] C. G. Gibson. *Elementary geometry of algebraic curves: an undergraduate introduction*. Cambridge University Press, Cambridge, 1998.

[38] Darrel Hankerson, Alfred Menezes, and Scott Vanstone. *Guide to elliptic curve cryptography*. Springer Professional Computing. Springer-Verlag, New York, 2004.

[39] Robin Hartshorne. *Algebraic geometry*. Springer-Verlag, New York, 1977. Graduate Texts in Mathematics, No. 52.

[40] F. Hess. Generalising the GHS attack on the elliptic curve discrete logarithm problem. *LMS J. Comput. Math.*, 7:167–192 (electronic), 2004.

[41] F. Hess. A note on the Tate pairing of curves over finite fields. *Arch. Math. (Basel)*, 82(1):28–32, 2004.

[42] Thomas W. Hungerford. *Algebra*, volume 73 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1980. Reprint of the 1974 original.

[43] Michael Jacobson, Alfred Menezes, and Andreas Stein. Solving elliptic curve discrete logarithm problems using Weil descent. *J. Ramanujan Math. Soc.*, 16(3):231–260, 2001.

[44] Michael J. Jacobson, Neal Koblitz, Joseph H. Silverman, Andreas Stein, and Edlyn Teske. Analysis of the xedni calculus attack. *Des. Codes Cryptogr.*, 20(1):41–64, 2000.

[45] Antoine Joux and Reynald Lercier. "Chinese & match", an alternative to Atkin's "match and sort" method used in the SEA algorithm. *Math. Comp.*, 70(234):827–836, 2001.

[46] Bo Gyeong Kang and Je Hong Park. Powered tate pairing computation. Cryptology ePrint Archive, Report 2005/260, 2005. `http://eprint.iacr.org/`.

[47] Hwan Joon Kim, Jung Hee Cheon, and Sang Geun Hahn. On remarks of lifting problems for elliptic curves. *Adv. Stud. Contemp. Math. (Pusan)*, 2:21–36, 2000.

[48] Neal Koblitz. *Algebraic aspects of cryptography*, volume 3 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, 1998. With an appendix by Alfred J. Menezes, Yi-Hong Wu and Robert J. Zuccherato.

[49] Neal Koblitz, Alfred Menezes, and Scott Vanstone. The state of elliptic curve cryptography. *Des. Codes Cryptogr.*, 19(2-3):173–193, 2000. Towards a quarter-century of public key cryptography.

[50] Serge Lang. *Introduction to algebraic geometry*. Interscience Publishers, Inc., New York-London, 1958.

[51] Serge Lang. *Abelian varieties*. Interscience Tracts in Pure and Applied Mathematics. No. 7. Interscience Publishers, Inc., New York, 1959.

[52] R. Lercier and F. Morain. Algorithms for computing isogenies between elliptic curves. In *Computational perspectives on number theory (Chicago, IL, 1995)*, volume 7 of *AMS/IP Stud. Adv. Math.*, pages 77–96. Amer. Math. Soc., Providence, RI, 1998.

[53] R. Lercier and F. Morain. Computing isogenies between elliptic curves over $\mathbf{F}_{p^n}$ using Couveignes's algorithm. *Math. Comp.*, 69(229):351–370, 2000.

[54] Reynald Lercier. Finding good random elliptic curves for cryptosystems defined over $\mathbf{F}_{2^n}$. In *Advances in cryptology—EUROCRYPT '97 (Konstanz)*, volume 1233 of *Lecture Notes in Comput. Sci.*, pages 379–392. Springer, Berlin, 1997.

[55] Reynald Lercier and François Morain. Counting the number of points on elliptic curves over finite fields: strategies and performances. In *Advances in cryptology—EUROCRYPT '95 (Saint-Malo, 1995)*, volume 921 of *Lecture Notes in Comput. Sci.*, pages 79–94. Springer, Berlin, 1995.

[56] Wenbo Mao. *Modern Cryptography.* Prentice Hall, Upper Saddle River, NJ, 4th edition, 2004.

[57] Alfred Menezes and Minghua Qu. Analysis of the Weil descent attack of Gaudry, Hess and Smart. In *Topics in cryptology—CT-RSA 2001 (San Francisco, CA)*, volume 2020 of *Lecture Notes in Comput. Sci.*, pages 308–318. Springer, Berlin, 2001.

[58] Alfred Menezes, Edlyn Teske, and Annegret Weng. Weak fields for ECC. In *Topics in cryptology—CT-RSA 2004*, volume 2964 of *Lecture Notes in Comput. Sci.*, pages 366–386. Springer, Berlin, 2004.

[59] Alfred J. Menezes and Teske Edlyn. Cryptographic implications of Hess' generalized GHS attack. Online at: http://www.cacr.math.uwaterloo.ca/ ajmeneze/research.html, December 2004.

[60] Alfred J. Menezes and Steven Galbraith. Algebraic curves and cryptography. Online at: http://www.cacr.math.uwaterloo.ca/ ajmeneze/research.html, December 2005.

[61] Alfred J. Menezes, Tatsuaki Okamoto, and Scott A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Trans. Inform. Theory*, 39(5):1639–1646, 1993.

[62] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of applied cryptography.* CRC Press Series on Discrete Mathematics and its Applications. CRC Press, Boca Raton, FL, 1997. With a foreword by Ronald L. Rivest.

[63] Richard A. Mollin. *An introduction to cryptography.* CRC Press Series on Discrete Mathematics and its Applications. Chapman & Hall/CRC, Boca Raton, FL, 2001.

[64] Volker Müller. On generation of cryptographically strong elliptic curves. Online at: http://lecturer.ukdw.ac.id/vmueller/publications.php. Preprint.

[65] Elizabeth Oswald. Introduction to elliptic curve cryptograhy. Online at: http://www.iaik.tu-graz.ac.at/aboutus/people/oswald.

[66] J. M. Pollard. Monte Carlo methods for index computation (mod p). *Math. Comp.*, 32(143):918–924, 1978.

[67] J. M. Pollard. Kangaroos, Monopoly and discrete logarithms. *J. Cryptology*, 13(4):437–447, 2000.

[68] Takakazu Satoh and Kiyomichi Araki. Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. *Comment. Math. Univ. St. Paul.*, 47(1):81–92, 1998.

[69] René Schoof. Elliptic curves over finite fields and the computation of square roots mod p. *Math. Comp.*, 44(170):483–494, 1985.

[70] René Schoof. Counting points on elliptic curves over finite fields. *J. Théor. Nombres Bordeaux*, 7(1):219–254, 1995. Les Dix-huitièmes Journées Arithmétiques (Bordeaux, 1993).

[71] I. Semaev. Summation polynomials and the discrete logarithm problem on elliptic curves. Preprint, February 5 2004.

[72] I. A. Semaev. Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p. *Math. Comp.*, 67(221):353–356, 1998.

[73] I.A. Semaev. Elliptic curve points over multiquadratic extensions and the discrete log problem. Preprint, January 2003.

[74] I.A. Semaev. A reduction of the space for the parallelized pollard lambda search on elliptic curves over prime finite fields and on anomalous binary elliptic curves. Preprint, August 2003.

[75] Igor A. Semaev. An algorithm for evaluation of discrete logarithms in some nonprime finite fields. *Math. Comp.*, 67(224):1679–1689, 1998.

[76] I. R. Shafarevich. *Basic algebraic geometry*. Springer-Verlag, New York, 1974. Translated from the Russian by K. A. Hirsch, Die Grundlehren der mathematischen Wissenschaften, Band 213.

[77] Joseph H. Silverman. *The arithmetic of elliptic curves*, volume 106 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1986.

[78] Joseph H. Silverman. *Advanced topics in the arithmetic of elliptic curves*, volume 151 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1994.

[79] Joseph H. Silverman. The xedni calculus and the elliptic curve discrete logarithm problem. *Des. Codes Cryptogr.*, 20(1):5–40, 2000.

[80] Joseph H. Silverman and Joe Suzuki. Elliptic curve discrete logarithms and the index calculus. In *Advances in cryptology—ASIACRYPT'98 (Beijing)*, volume 1514 of *Lecture Notes in Comput. Sci.*, pages 110–125. Springer, Berlin, 1998.

[81] Joseph H. Silverman and John Tate. *Rational points on elliptic curves*. Undergraduate Texts in Mathematics. Springer-Verlag, New York, 1992.

[82] N. P. Smart. The discrete logarithm problem on elliptic curves of trace one. *J. Cryptology*, 12(3):193–196, 1999.

[83] Nigel P. Smart. How secure are elliptic curves over composite extension fields? In *Advances in cryptology—EUROCRYPT 2001 (Innsbruck)*, volume 2045 of *Lecture Notes in Comput. Sci.*, pages 30–39. Springer, Berlin, 2001.

[84] B. Sury. Elliptic curves over finite fields. In *Elliptic curves, modular forms and cryptography (Allahabad, 2000)*, pages 33–47. Hindustan Book Agency, New Delhi, 2003.

[85] The PARI Group, Bordeaux. *PARI/GP, version* `2.1.10`, 2004. available from `http://pari.math.u-bordeaux.fr/`.

[86] Paul C. van Oorschot and Michael J. Wiener. Parallel collision search with cryptanalytic applications. *J. Cryptology*, 12(1):1–28, 1999.

[87] Lawrence C. Washington. *Elliptic curves*. Discrete Mathematics and its Applications (Boca Raton). Chapman & Hall/CRC, Boca Raton, FL, 2003. Number theory and cryptography.

[88] Michael J. Wiener. The full cost of cryptanalytic attacks. *J. Cryptology*, 17(2):105–124, 2004.

[89] Michael J. Wiener and Robert J. Zuccherato. Faster attacks on elliptic curve cryptosystems. In *Selected areas in cryptography (Kingston, ON, 1998)*, volume 1556 of *Lecture Notes in Comput. Sci.*, pages 190–200. Springer, Berlin, 1999.

Appendix A

In this appendix we include description of the syntax used in our algorithms that were programed in `Pari/GP`. The standard source for these commands is [85].

1. `ellpow`(`Ep`, P, i) - computes multiples of the point P defined on the curve E_p

2. `elladd`(`Ep`, P, R) - adds points P and R on a given elliptic curve E_p

3. `ellinit`($*$) - initializes an elliptic curve defined over a given field. For example, `Ep=ellinit([Mod(a,p),Mod(b,p),Mod(c,p),Mod(d,p),Mod(e,p)])` initializes Ep over $\mathbb{F}_p$ for a give prime p. The coefficients places are defined to be $a_1, \ldots, a_6$ as in the definition of an elliptic curve in equation (3.1).

4. `Mod, ceil, sqrt` are all standard function in the Pari library and are the calls for modular arithmetic, the ceiling function and the square root function respectively.

5. `vector(n,expression)` - produces a row vector of length n with the desired expression. So in the algorithm for Pollard's Rho, `va` and `vb` are row vector of length m where each entry is a random number in the range $[0, n-1]$.

Appendix B

The following table is the set T which we originally omitted from the example of the Pohlig-Hellman attack.

j	j([3]P)	j	j([3]P)	j	j([3]P)	j	j([3]P)	j	j([3]P)	j	j([3]P)
0	$\mathcal{O}$	1	$(460, 25)$	2	$(631, 182)$	3	$(325, 326)$	4	$(213, 106)$	5	$(425, 144)$
6	$(392, 319)$	7	$(670, 460)$	8	$(404, 91)$	9	$(635, 361)$	10	$(242, 221)$	11	$(422, 363)$
12	$(663, 494)$	13	$(617, 604)$	14	$(284, 505)$	15	$(541, 392)$	16	$(168, 508)$	17	$(591, 204)$
18	$(80, 368)$	19	$(290, 673)$	20	$(421, 410)$	21	$(567, 681)$	22	$(548, 262)$	23	$(704, 331)$
24	$(436, 453)$	25	$(161, 275)$	26	$(133, 221)$	27	$(306, 52)$	28	$(475, 57)$	29	$(41, 54)$
30	$(288, 586)$	31	$(647, 146)$	32	$(212, 643)$	33	$(210, 129)$	34	$(374, 14)$	35	$(636, 521)$
36	$(344, 498)$	37	$(195, 475)$	38	$(147, 678)$	39	$(120, 281)$	40	$(482, 313)$	41	$(192, 542)$
42	$(514, 106)$	43	$(646, 415)$	44	$(602, 605)$	45	$(463, 702)$	46	$(711, 613)$	47	$(140, 421)$
48	$(224, 366)$	49	$(56, 159)$	50	$(676, 68)$	51	$(315, 147)$	52	$(294, 594)$	53	$(393, 486)$
54	$(119, 54)$	55	$(581, 310)$	56	$(14, 152)$	57	$(366, 290)$	58	$(400, 173)$	58	$(501, 172)$
59	$(273, 663)$	60	$(540, 544)$	61	$(624, 385)$	62	$(627, 608)$	63	$(305, 430)$	64	$(198, 79)$
65	$(515, 387)$	66	$(1, 290)$	67	$(87, 369)$	68	$(340, 672)$	69	$(322, 633)$	70	$(570, 543)$
71	$(62, 372)$	72	$(558, 270)$	73	$(606, 329)$	74	$(178, 247)$	75	$(453, 612)$	76	$(323, 196)$
77	$(671, 569)$	78	$(22, 542)$	79	$(701, 399)$	80	$(47, 422)$	81	$(698, 545)$	82	$(559, 665)$
83	$(447, 611)$	84	$(474, 53)$	85	$(508, 598)$	86	$(121, 292)$	87	$(699, 615)$	88	$(513, 145)$
89	$(293, 638)$	90	$(630, 190)$	91	$(260, 296)$	92	$(241, 293)$	93	$(561, 312)$	94	$(649, 520)$
95	$(0, 495)$	96	$(626, 551)$	97	$(588, 453)$	98	$(48, 364)$	99	$(221, 712)$	100	$(185, 436)$
101	$(494, 167)$	102	$(387, 153)$	103	$(316, 540)$	104	$(669, 338)$	105	$(312, 90)$	106	$(599, 517)$
107	$(280, 219)$	108	$(352, 290)$	109	$(638, 718)$	110	$(414, 453)$	111	$(603, 524)$	112	$(505, 542)$
113	$(665, 336)$	114	$(677, 463)$	115	$(63, 276)$	116	$(63, 443)$	117	$(677, 256)$	118	$(665, 383)$
119	$(505, 177)$	120	$(603, 195)$	121	$(414, 266)$	122	$(638, 1)$	123	$(352, 429)$	124	$(280, 500)$
125	$(599, 202)$	126	$(312, 629)$	127	$(669, 381)$	128	$(316, 179)$	129	$(387, 566)$	130	$(494, 552)$
131	$(185, 283)$	132	$(221, 7)$	134	$(48, 355)$	135	$(588, 266)$	136	$(626, 168)$	137	$(0, 224)$
138	$(649, 199)$	139	$(561, 407)$	140	$(241, 426)$	141	$(260, 423)$	142	$(630, 529)$	143	$(293, 81)$
144	$(513, 574)$	145	$(699, 104)$	146	$(121, 427)$	147	$(508, 121)$	148	$(474, 666)$	149	$(447, 108)$
150	$(559, 54)$	151	$(698, 174)$	152	$(47, 297)$	153	$(701, 320)$	154	$(22, 177)$	155	$(671, 150)$
156	$(323, 523)$	157	$(453, 107)$	158	$(178, 472)$	159	$(606, 390)$	160	$(558, 449)$	161	$(62, 347)$
162	$(570, 176)$	163	$(322, 86)$	164	$(340, 47)$	165	$(87, 350)$	166	$(1, 429)$	167	$(515, 332)$
168	$(198, 640)$	169	$(305, 289)$	170	$(627, 111)$	171	$(624, 334)$	172	$(540, 175)$	173	$(273, 56)$
174	$(501, 547)$	175	$(400, 546)$	176	$(366, 429)$	177	$(14, 567)$	178	$(581, 409)$	179	$(119, 665)$
180	$(393, 233)$	181	$(294, 125)$	182	$(315, 572)$	183	$(676, 651)$	184	$(56, 560)$	185	$(224, 353)$
186	$(140, 298)$	187	$(711, 106)$	188	$(463, 17)$	189	$(602, 114)$	190	$(646, 304)$	191	$(514, 613)$
192	$(192, 177)$	193	$(482, 406)$	194	$(120, 438)$	195	$(147, 41)$	196	$(195, 244)$	197	$(344, 221)$
198	$(636, 198)$	199	$(374, 705)$	200	$(210, 590)$	201	$(212, 76)$	202	$(647, 573)$	203	$(288, 133)$
204	$(41, 665)$	205	$(475, 662)$	206	$(306, 667)$	207	$(133, 498)$	208	$(161, 444)$	209	$(436, 266)$
210	$(704, 388)$	211	$(548, 457)$	212	$(567, 38)$	213	$(421, 309)$	214	$(290, 46)$	215	$(80, 351)$
216	$(591, 515)$	217	$(168, 211)$	218	$(541, 327)$	219	$(284, 214)$	220	$(617, 115)$	221	$(663, 225)$
222	$(422, 356)$										

Table 6.1: Omitted Set T for the Pohlig-Hellman Attack