

Project: Supersingular Curves and the Weil Pairing in Elliptic Curve Cryptography

Instructor: Nigel Boston

Author: Sarah Knoop

1 Introduction

Even first semester calculus students are aware of how calculus, hence analysis, is used to solve problems in engineering. In recent decades the engineering world is gaining more exposure to algebra through the powerful problem solutions it provides. One area that algebra has made significant contributions to is cryptography and, more specifically, public key cryptography. In this paper we aim to illustrate such enhancements by discussing how supersingular elliptic curves and the Weil pairing have impacted elliptic curve cryptography. The paper is structured as follows: In the first few sections we give an overview of public key cryptography and elliptic curves in this context; Then we discuss supersingular curves and the Weil pairing and see how the pairing can be used to break and construct supersingular elliptic curve cryptosystems; Finally, to provide contrast and a little food for thought, we discuss other categories of elliptic curves, an alternative to the Weil pairing (the Tate pairing) and some exciting new index calculus methods for elliptic curves.

2 Basics of Public Key Cryptography

In contrast to symmetric encryption methods in which a secret key must be known to both parties in the secure message exchange, public key encryption features key pairs which do not have to be shared. The basic schema for two parties, say Alice (A) and Bob (B), to communicate securely is as follows:

1. A and B are provided with (public key, private key) pairs, say (U_a, R_a) and (U_b, R_b) , respectively. Each publishes his or her public key, making it accessible for anyone to see and use.
2. If A wants to send message M to B , all A needs to do is encrypt the message with B 's public key, $C = E_{U_b}(M)$, and send this to B . She is assured that only B , the possessor of the corresponding private key R_b to U_b will be able to decrypt the message.
3. B , to read the message, simply computes $D_{R_b}(C)$.

Public key cryptosystems give immediate digital signature schemes; just encrypt with the private key. These systems also tend to be more computation intensive than symmetric ones, many of which are fully executed in hardware alone. However, their algebraic foundations provide robust proofs of security that few symmetric schemes can match. Most such proofs are centered around an algebraic problem in which it is difficult to compute a solution.

It is important in public key cryptography to find encryption and decryption functions and corresponding key pairs such that for any key pair (U_i, R_i) and plaintext M , $D_{R_i}(E_{U_i}(M)) = M$ and all

E , D and (U_i, R_i) be quick to compute. Equally critical, we desire that decryption of the ciphertext be extremely difficult without the private key. Problems in finite groups provide the setting for such functions, key pairs and security proofs. We next consider one such problem, finding discrete logs.

2.1 The Discrete Log Problem over \mathbb{Z}_p^* and ElGamal Cryptosystem

Definition 1 (The Discrete Log Problem (DLP)). *Given a generator g of group \mathbb{Z}_p^* under multiplication, p prime. Given an element $a \in \mathbb{Z}_p^*$, find the smallest integer x , $0 \leq x \leq p - 2$, such that*

$$g^x \equiv a \pmod{p}.$$

Now that we have a problem over a finite group. So, we can build our cryptosystem.

Algorithm 1: The ElGamal Cryptosystem [12]

Given the public system parameters $G = \mathbb{Z}_p^*$, its generator g and p , the following three procedures constitute the cryptosystem. Here, we suppose that Alice, A wishes to send an encrypted message to Bob, B .

Key Generation:

STEP 1: Alice and Bob chooses (or are assigned by an authority) their distinct secret keys x_a and x_b randomly from $\{0, \dots, p - 2\}$.

STEP 2: Alice computes $h_a = g^{x_a}$ as her public key and Bob computes $h_b = g^{x_b}$ as his public key and both are published.

Encryption:

STEP 1: A converts her message M to an element of G (for long messages we can break the plaintext into blocks so that such conversion is possible).

STEP 2: A chooses k randomly from $\{0, \dots, p - 2\}$. She keeps this choice secret as well.

STEP 3: A computes $c_1 = g^k$ and $c_2 = Mh_b^k$. She then sends Bob the ciphertext pair (c_1, c_2) .

Decryption:

Bob recovers M by computing $c_2c_1^{-x_b} = M$.

Bob's computation of $c_2c_1^{-x_b}$ produces M since

$$\begin{aligned} c_2c_1^{-x_b} &= Mh_b^k g^{-k(x_b)} \\ &= Mg^{(x_b)k} g^{-k(x_b)} \\ &= Mg^{(x_b)k - k(x_b)} \\ &= Mg^0 \\ &= M \end{aligned}$$

Thus encryption and decryption act as expected for all key pairs and are easy to compute (use multiplication tables or repeated squares). The question is whether this system is difficult to crack. To do so, an eavesdropper needs to solve the discrete log problem on the presented system parameters. Success in this matter really depends on the size of the group. One can simply try all possible values for x , and if p is small such a search can execute in a short amount of time. If p is much bigger, brute force trial is impractical. The time to compute using this method grows exponentially with p . Other methods, like index calculus examined below, can compute discrete logs in subexponential time, given that $p - 1$ is a product of small primes. If $p - 1$ lack such structure, these solutions begin to become impractical for large p [10].

2.2 Cryptanalysis with Index Calculus

Index Calculus (summarized in [10]) methods involve forming relations among numbers in what is called a *factor basis*, described below. Once the number of relations found equals the number of prime factors of the elements in the factor basis we can solve the system of equations to recover the discrete logs of these primes. From these logs we can then, with a bit more computation, recover the discrete log of any chosen element.

In a little more detail, to choose a factor basis we form a list of the first r primes. How many we choose is critical. The factor basis, F consists of all integers whose prime factors are all less than or equal to the largest prime on our list. Then we start looking at the exponentiations of the generator, $g, g^2, g^3 \dots$, mapping these values to the integers if the field happens to be \mathbb{F}_p instead of \mathbb{Z}_p . If any value g^j is in F , we record it and the following relation. We can derive from g^j 's factorization into powers of their first r primes and the fact that \mathbb{Z}_p has order $p - 1$.

$$j \equiv \sum_{i=1}^r e_i(j) \log_g(p_i) \pmod{p-1}$$

where p_i is the i^{th} prime and $e_i(j)$ is its corresponding exponent in the factorization of g^j . We continue computing powers of the generator until we obtain r independent relations. We solve these equations (which are typically sparse) to get the discrete log of each of the r primes. Now, to find the discrete log of $a \in \mathbb{F}_p$ we compute the quantities a, ga, g^2a, \dots , and lift these as well to \mathbb{Z} if needed. We continue the computation until we find an element $g^j a$ that is in the factor basis. Again, we utilize the prime factorization of this value and the fact we are in \mathbb{Z}_p to obtain

$$j + \log_g(a) \equiv \sum_{i=1}^r f_i \log_g(p_i) \pmod{p-1}$$

where f_i is its corresponding exponent to the i^{th} prime. We already know the discrete log of each of the r primes, so we can just solve this equivalence for $\log_g(a)$. Here, we get a runtime that is subexponential in p provided we make a good choice for $r \approx e^{\sqrt{\log p}}$ [10].

How can we subvert this attack? We can use really large keys as previously discussed, but this has significant drawbacks. The system becomes ever slower with larger keys making it undesirable. Additionally, large keys require more computation space so such a cryptosystem cannot fit on small

devices that also need to utilize encryption (eg. smart cards). Since larger keys do not seem to offer much of a fix, we need to consider other changes. We note that we have, to this point, used \mathbb{Z}_p^* as the group to base our cryptosystem. We show in the next section that we can change this group to alleviate several of our concerns.

3 Elliptic Curve Cryptography (ECC)

Elliptic Curve Cryptography is based on abelian groups constructed from elliptic curves over finite fields [14].

Definition 2 (Elliptic Curve). *The following equation defines an elliptic curve*

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

where the coefficients lie in some field.

When we choose points on an elliptic curve whose coordinates lie in some finite field, usually \mathbb{F}_q where q is a prime power, we almost get a group. We fill in the necessary details below.

Definition 3 ($E(\mathbb{F}_q)$). *The abelian group $E(\mathbb{F}_q)$ is defined by*

$$E(\mathbb{F}_q) = \{(x, y) \in \overline{\mathbb{F}_q} | y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6\} \cup \{\infty\}$$

$\overline{\mathbb{F}_q}$ denotes the algebraic closure of \mathbb{F}_q , and ∞ , referred to as “the point at infinity”, is the identity. The group operation, $P + Q$ for $P = (x_1, y_1), Q = (x_2, y_2) \in E(\mathbb{F}_q)$ behaves as follows

$$\begin{aligned} P + Q &= \infty, \text{ if } x_1 = x_2 \text{ and } y_1 + y_2 + a_1x_2 + a_3 = 0 \\ P + Q &= (x_3, y_3), \text{ otherwise where} \\ &\quad x_3 = \lambda^2 + a_1\lambda - a_2 - x_1 - x_2 \\ &\quad y_3 = -(\lambda + a_1)x_3 - \nu - a_3 \end{aligned}$$

where λ and ν are rational functions of the curve coefficients and x_1, x_2, y_1, y_2 .

For details on the explicit formulas for λ and ν see [1]. For cryptographic purposes points on an elliptic curve will serve the same purpose as elements of \mathbb{Z}_p in the construction of a cryptosystem. We discuss such a system next and then turn to what advantages we have acquired with elliptic curves.

3.1 Discrete Logs and ElGamal for Elliptic Curves

The discrete log problem can be defined over $E(\mathbb{F}_q)$ and consequently we can build an ElGamal cryptosystem. First we define for $P \in E(\mathbb{F}_q)$

$$mP = \underbrace{P + P + \cdots + P}_m.$$

We say m is the order of P when $mP = \infty$ and use $\langle P \rangle$ to denote the cyclic subgroup of $E(\mathbb{F}_q)$ generated by P .

Definition 4 (The Discrete Log Problem for Elliptic Curves (ECDLP)). Given $\langle P \rangle$ of order m where $P \in E(\mathbb{F}_q)$ and a point $Q \in \langle P \rangle$ the find the smallest integer ℓ , $0 \leq \ell \leq m - 1$ such that $Q = \ell P$.

Algorithm 2: The ElGamal Cryptosystem for Elliptic Curves [1]

Given the public system parameters $\langle P \rangle$, its generator P and the order m , the following three procedures constitute the cryptosystem. Here, we suppose that Alice, A wishes to send an encrypted message to Bob, B .

Key Generation:

STEP 1: Alice and Bob choose (or are assigned) their distinct secret keys ℓ_a and ℓ_b randomly from $\{0, \dots, m - 1\}$

STEP 2: Alice computes $Q_a = \ell_a P$ as her public key and Bob computes $Q_b = \ell_b P$ his public key and both are published.

Encryption:

STEP 1: A converts her message M to an element of $\langle P \rangle$ (for long messages we can break the plaintext into blocks so that such conversion is possible).

STEP 2: A chooses k randomly from $\{0, \dots, m - 1\}$. She keeps this choice secret as well.

STEP 3: A computes $c_1 = kP$ and $c_2 = M + kQ_b$. She then sends Bob the ciphertext pair (c_1, c_2) .

Decryption:

Bob recovers M by computing $c_2 - \ell_b c_1 = M$.

3.2 Comparison With Cryptosystems Built on \mathbb{Z}_p^*

Good curves, those with many points, usually produce groups with very little structure, outside of knowing that the group was derived from an elliptic curve. This makes them quite nice for cryptography. Additionally, one can get by with shorter key lengths using elliptic curves while maintaining equivalent security guarantees to systems built atop \mathbb{Z}_n^* . Furthermore, index calculus methods have, until recently, seemed futile. Such a method on $E(\mathbb{F}_q)$ seems to involve the lifting of the problem to $E(\mathbb{Q})$ for computation. Methods prior to 2004 could not seem to find such a map nor provide nor provide guarantees of it being well defined, if it existed [10]. We'll examine these new index calculus based threats in Section 8.

This is quite the sales pitch for ECC, which is why it has, in recent years, become state of the art. Other cryptanalysis methods, however, have met with success by exploiting properties of certain curves, namely supersingular curves. The upshot here is that these methods in conjunction with particular supersingular curves have given way to new cryptosystems. We explore these developments in the next few sections.

4 Supersingular Curves

The definition of a supersingular curve depends on the number of points in $E(\mathbb{F}_q)$. For this we refer to a result by Hasse cited in [9].

Theorem 1. *The order of $E(\mathbb{F}_q)$, where q is a power of a prime, say p , is $q+1-t$ where $|t| \leq 2\sqrt{q}$.*

Definition 5 (supersingular). *$E(\mathbb{F}_q)$ is supersingular if p divides t . Clearly this is the case when $t = 0$, which is the case for many supersingular curves.*

Supersingularity imposes limitations on the different group structures $E(\mathbb{F}_q)$ can assume. It turns out supersingular curves have corresponding groups that are either cyclic of order q or isomorphic to either $\mathbb{Z}_{\sqrt{q+1}} \oplus \mathbb{Z}_{\sqrt{q+1}}$, $\mathbb{Z}_{\sqrt{q-1}} \oplus \mathbb{Z}_{\sqrt{q-1}}$ or $\mathbb{Z}_{(q+1)/2} \oplus \mathbb{Z}_2$ (Lemma 1,2, Table 1 of [9]).

The small number of possible group structures enables many algorithms designed for attacking curves to be modified. The result is a reduction in complexity to subexponential, and even polynomial time when they exploit this fact. This gives supersingular curves the stigma of being “weak” choices for ECC systems. We’ll see in Section 7, that smart choices for even supersingular curves can be turned into a different kind of ECC system because they are relatively easy to compute with.

5 The Weil Pairing

Before we present the attacks and cryptosystems, we need to discuss the mapping that makes these possible. The Weil pairing (outlined in [9]) is a mapping from pairs of points on an elliptic curve to $\overline{\mathbb{F}_q}$. More specifically for us, it establishes an isomorphism between a group $\langle P \rangle$ of order m and the m^{th} roots of unity in \mathbb{F}_{q^k} , a suitable extension field of \mathbb{F}_q . For cryptography this means that if we have an instance of ECDLP we can, with this pairing, map it to an instance of DLP in the extension field \mathbb{F}_{q^k} . If k is not too big, we can solve DLP in the extension field, thereby solving our original instance of ECDLP. We discuss this further in the next section, as we turn now to the details of the Weil pairing’s construction.

The Weil pairing operates on two points of $E(\overline{\mathbb{F}_q})$ with the same order m . The collection of all points $P \in E(\mathbb{F}_q)$ is denoted $E[m]$ and called the m -torsion points. Only under certain conditions is $E[m]$ a subgroup of $E(\mathbb{F}_q)$ (Lemma 3 of [9]), but, we can determine the smallest k such that $E[m] \subseteq E(\mathbb{F}_{q^k})$. This will be key for us later when we turn to actually computing such a k . So said, we may define the Weil Pairing.

Definition 6 (The Weil Pairing). *Let m be an integer relatively prime to q . The Weil pairing is a function:*

$$e_m : E[m] \times E[m] \rightarrow \overline{\mathbb{F}_q}$$

where q is a prime power of some prime p and $E[m]$ is as above.

To construct the pairing we begin with functions from the function field $\overline{\mathbb{F}_q}(E)$ which is, informally, the set of rational maps in x and y modulo the equation defining the elliptic curve whose coefficients lie in the algebraic closure of \mathbb{F}_q . We additionally need to know about *divisors*.

Definition 7 (Divisor). A divisor is a formal sum of points in $E(\overline{\mathbb{F}}_q)$

$$D := \sum_{P \in E(\overline{\mathbb{F}}_q)} n_P P$$

where $n_P \in \mathbb{Z}$ and $n_P = 0$ for all but finitely many $P \in E(\overline{\mathbb{F}}_q)$.

A divisor's *degree* is determined by $\sum n_P$. We will be most concerned with divisors of degree zero which form a group, D^0 . We can associate divisors with functions $f \in \overline{\mathbb{F}}_q(E)$ and, more importantly, such divisors will be of degree zero.

Definition 8 (Divisor of a function, $\text{div}(f)$). Let $f \in \overline{\mathbb{F}}_q(E)$. Let $P \in E(\overline{\mathbb{F}}_q)$ be of order $m > 0$. For each P we define $n'_P \in \mathbb{Z}$ to be m if P is a zero (root of the numerator) of f , or $-m < 0$ if P is a pole (root of the denominator) of f . Then

$$\text{div}(f) := \sum_P n'_P P.$$

A divisor is *principal* if $D = \text{div}(f)$ for some $f \in \overline{\mathbb{F}}_q(E)$. Let D_{prin} denote the set of principal divisors of $E(\overline{\mathbb{F}}_q)$ and note that $D_{\text{prin}} \subseteq D^0$. So, for each $D \in D^0$ there exists a unique point $P \in E(\overline{\mathbb{F}}_q)$ such that the divisor computed from $D - ((P) - (\infty))$ is principal. Here (P) and (∞) denote the divisors of P and ∞ which are each themselves, respectively.

The one final piece of information we need to construct the pairing is a way to connect all this information. We note that we can *evaluate* functions $f \in \overline{\mathbb{F}}_q(E)$ at a divisor D . If we let $D := \sum n_P P$ such that $\text{supp}(D) \cap \text{supp}(\text{div}(f)) = \emptyset$, then

$$f(D) = \sum_{P \in E(\overline{\mathbb{F}}_q)} f(P)^{n_P}.$$

Here, the support of a divisor $\text{supp}(D)$ is just the set $\{P | n_P \neq 0\}$ for $D := \sum n_P P$.

So, we construct the Weil pairing as follows:

Definition 9 (The Weil Pairing, construction). Let m be an integer relatively prime to q . Let $P, Q \in E[m]$. Let $D_P, D_Q \in D^0$ such that $D_P - ((P) - (\infty))$ and $D_Q - ((Q) - (\infty))$ are principal and $\text{supp}(D_P) \cap \text{supp}(D_Q) = \emptyset$. Let $f_P, f_Q \in \overline{\mathbb{F}}_q(E)$ such that $\text{div}(f_P) = mD_P$ and $\text{div}(f_Q) = mD_Q$. Then, the Weil pairing is

$$e_m(P, Q) := \frac{f_P(D_Q)}{f_Q(D_P)}.$$

The difficulty with the actual computing of the Weil pairing of two points is finding f_P and f_Q . Such functions, if represented by non-zero coefficients and corresponding monomials, could be exponential in size relative to their respective divisors. Miller (cited in [9]) noted that we do not need a full representation as above, as long as we can still evaluate them at points and divisors. What he suggests is that we use “straight line program” to represent the functions. These representations have polynomial size and can evaluate $f(P)$ in polynomial time, provided it is defined. For details in the is regard, refer to [9].

The Weil pairing has some rather nice properties which will be of great use to us.

1. Identity: $\forall P \in E[m], e_m(P, P) = 1$
2. Alteration: $\forall P, Q \in E[m], e_m(P, Q) = e_m(Q, P)^{-1}$
3. Bilinearity: $\forall P, Q, R \in E[m], e_m(P+Q, R) = e_m(P, R)e_m(Q, R)$ and $e_m(P, Q+R) = e_m(P, Q)e_m(P, R)$
4. Non-degeneracy: If $P \in E[m]$ and if $e_m(P, Q) = 1 \forall Q \in E[m]$, then $P = \infty$.

We explicitly note that if m is chosen poorly (such as not relatively prime to q), the Weil pairing can degenerate, badly.

Now to map an instance of ECDLP to DLP in an extension field, we need to demonstrate the aforementioned isomorphism between $\langle P \rangle$, the cyclic subgroup of \mathbb{F}_q of order m , and the m^{th} roots of unity in \mathbb{F}_{q^k} . Theorems 9 and 10 in [9] provide this for us and combine them here.

Theorem 2. *Let $P \in E[m]$, m relatively prime to q .*

(A) *There exists a $Q \in E[m]$ such that $e_m(P, Q)$ is a primitive m^{th} root of unity.*

(B) *Furthermore, Let $\theta : \langle P \rangle \rightarrow \{\mu_m\}$, where $\{\mu_m\} \subseteq \mathbb{F}_{q^k}$, be defined by $R \rightarrow e_m(R, Q)$. Then θ is a group isomorphism.*

Proof Sketch. To prove (A), we have by the bilinearity of the Weil pairing that

$$\begin{aligned} e_m(P, Q)^m &= e_m(P, mQ) \\ &= e_m(P, \infty) \\ &= 1. \end{aligned}$$

Thus, $e_m(P, Q)$ is a m^{th} root of unity. There are m cosets of $\langle P \rangle$ in $E[m]$. One can check, using the properties of the Weil pairing, that two points (say $P_1, P_2 \in E[m]$) are in the same coset of $\langle P \rangle$ if and only if their Weil pairings with P are equal ($e_m(P, P_1) = e_m(P, P_2)$). So, elements of the same coset will map to the same m^{th} root of unity. Then, by selecting one Q from each of the m cosets of $\langle P \rangle$, $e_m(P, Q)$ will then attain the value of each m^{th} root of unity, one being primitive.

The proof of (B) is then short. For $P_1, P_2 \in \langle P \rangle$,

$$\begin{aligned} \theta(P_1 + P_2) &\Leftrightarrow e_m(P_1 + P_2, Q) \\ &\Leftrightarrow e_m(P_1, Q)e_m(P_2, Q) \\ &\Leftrightarrow \theta(P_1)\theta(P_2) \end{aligned}$$

Since θ is onto (from (A)) and $|\langle P \rangle| = |\{\mu_m\}| = m$, we have an isomorphism. □

6 Cryptanalysis of Elliptic Curves with the Weil Pairing

We now turn to how the Weil Pairing can be utilized as a tool for cryptanalysis. Menezes, Okamoto and Vanstone [9] have developed the following reduction from ECDLP to DLP based on the Weil pairing.

Algorithm 3: The MOV Reduction

Input: A point on the elliptic curve $P \in E(\mathbb{F}_q)$ of order m , and $R \in \langle P \rangle$.

Output: An integer ℓ such that $R = \ell P$.

STEP 1: Determine the smallest integer k such that $E[m] \subseteq E(\mathbb{F}_{q^k})$

STEP 2: Find $Q \in E[m]$ such that $g = e_m(P, Q)$ has order m .

STEP 3: Compute $a = e_m(R, Q)$.

STEP 4: Compute $\ell = \log_g a$ in \mathbb{F}_{q^k} and return.

We can quickly check that the ℓ found in STEP 4 is in fact the ℓ in question. By bilinearity of the Weil pairing we have

$$\begin{aligned} a &= e_m(R, Q) \\ &= e_m(\ell P, Q) \\ &= e_m(P, Q)^\ell \\ &= g^\ell. \end{aligned}$$

The algorithm takes exponential time in $\log q$ in general. This is because we, to this point, have not presented methods of finding Q or k (which is exponentially large in general), other than the obvious, exhaustive search.

Yet, there is still hope. Recall that in Section 4 we noted that the groups induced by supersingular elliptic curves have only a few possible structures. Thus, all supersingular curves can be divided into six categories. For each category we can precompute k such that $E[m] \subseteq E(\mathbb{F}_{q^k})$. Table 1 of [9] summarizes this information and from here we can get for supersingular curves $k \leq 6$. This takes care finding k . To find Q we again take advantage of the limited group structures. Elliptic curve groups are, in general of the form $\mathbb{Z}_{n_1} \oplus \mathbb{Z}_{n_2}$. The extensions of each category of supersingular curve will be of the form $\mathbb{Z}_{cn_1} \oplus \mathbb{Z}_{cn_1}$ for appropriate c (details in [9]). This will help limit our choices for Q .

Algorithm 4: The MOV Reduction for Supersingular Curves

Input: A point on the elliptic curve $P \in E(\mathbb{F}_q)$ of order m , and $R \in \langle P \rangle$.

Output: An integer ℓ such that $R = \ell P$.

STEP 1: Determine the smallest integer k such that $E[m] \subseteq E(\mathbb{F}_{q^k})$ by using the table.

STEP 2: Pick a random $Q' \in E(\mathbb{F}_{q^k})$ and set $Q = (cn_1/m)Q'$.

STEP 3: Compute $g = e_m(P, Q)$ and $a = e_m(R, Q)$.

STEP 4: Compute $\ell' = \log_g a$ in \mathbb{F}_{q^k} .

STEP 5: Check if $R = \ell'P$. If so return $\ell = \ell'$. If not, the order of g must be less than m . Our choice was unfortunate. Return to STEP 2.

We give the following proof sketch of the improved runtime.

Theorem 3. *The above procedure runs in probabilistic polynomial time (in $\log q$) [9].*

Proof Sketch. We look up k in constant time. We can choose Q' in probabilistic polynomial time by choosing an x -coordinate and computing the corresponding y -coordinate which there are known methods for computing in polynomial time for points in $E(\mathbb{F}_q)$. Our choice of for the x -coordinate, however, may not yield a point in $E(\mathbb{F}_q)$. But, we can try again with probability of success at least $1/2 - 1/\sqrt{q}$ by the Hasse Theorem (Section 4). Since k is at most 6, we can use a similar procedure and get the same time complexity for choosing points in $E(\mathbb{F}_{q^k})$. From here we can get Q in polynomial time. Using the algorithm by Miller alluded to in Section 5 we can get g and a in probabilistic polynomial time. It turns out that we will need about $O(\log \log m)$ many guesses in order to get a Q that gives an $e_m(P, Q)$ of order m in \mathbb{F}_{q^k} . Finally, we can test if $R = \ell'P$ in polynomial time. \square

Menezes, Okamoto and Vanstone also showed that if q is prime or a power of a small prime, then with Algorithm 4 the ECDLP can be solved in probabilistic subexponential time (again, in $\log q$).

7 Identity Based Encryption (IBE)

Up until now, the story doesn't look too good for supersingular curves. We now provide a little redemption by showing how supersingular curves made the first fully functional identity based cryptosystem possible.

7.1 History

A. Shamir first published the concept of IBE in 1985 [11]. He envisioned a system in which users could communicate securely and verify signatures without having to agree on a shared key, look up a public key, maintain key tables or solicit a trusted third party for each exchange. Instead the user could use their name or email address or other personal information, provided it could not be repudiated later, as a public key. Secret keys would be generated from a "master" secret key by a service and provided, most likely embedded in hardware, when the user joins. This service is essential, because if the user could derive their secret key from their service identity, then so could anyone else.

In such a system we are presented with a variety of practical and computational security issues. Practically, the production and distribution of the secret key hardware must be expertly monitored and efforts must be made to prevent or recover from loss, duplication, or unauthorized access to such hardware. Computationally, IBE faces many of the same issues as standard public key cryptography. That is, when the secret key is known, computation is relatively easy and without the problem is intractable.

In [11], Shamir did not provided implementation specifics on how such a system would be constructed. Boneh and Franklin [3] published such a system based on supersingular elliptic curves and the Weil pairing and proved it secure against a powerful chosen ciphertext attack. We discuss this system next.

7.2 A Fully Functional IBE based on the Weil Pairing

Boneh and Franklin [3], [8] developed an IBE system in the flavor of ElGamal that rests squarely on the shoulders of a particular supersingular curve, a slightly modified version of the Weil pairing and the hardness of the Bilinear Diffie Hellman problem (BDH) (for a formal statement see [3]), a relative of ECDLP. We will first discuss the algebraic components of this system and the motivation behind such choices. We will then move on to discussing the algorithm and the security guarantees that it provides.

The curve Boneh and Franklin chose was $y^2 = x^3 + 1$ over \mathbb{F}_p , $p > 3$ prime such that $p \equiv 2 \pmod{3}$. The induced group, which we will refer to as $E(\mathbb{F}_p)$, has some key properties:

1. For any $y_0 \in \mathbb{F}_p$ there is a unique point (x_0, y_0) , namely, $x_0 = (y_0^2 + 1)^{1/3} \pmod{p-1} \in \mathbb{F}_p$.
2. Let $1 \neq \alpha \in \mathbb{F}_{p^2}$ be a solution of $x^3 - 1 \equiv 0 \pmod{p}$. Then the map $\psi(x, y) = (\alpha x, y)$ is an automorphism of the group of points on the curve. Note, for $Q \in E(\mathbb{F}_p)$ that $\psi(Q) \in E(\mathbb{F}_{p^2})$ but $\psi(Q) \notin E(\mathbb{F}_p)$.

The first property gives us a *computable* way to select points on this curve because we have unique cube-roots. This will allow us to assign points to users efficiently. The second property gives us way to overcome an obstacle we face with the Weil pairing e_m . For this cryptographic system m is chosen as a prime factor of $p + 1$ (so, relatively prime to p) such that m^2 does not divide $p + 1$. Because we are working over \mathbb{F}_p , for any pair of points $Q, R \in \mathbb{F}_p$ of order m , $e_m(P, Q) = 1$. In other words, the Weil pairing is degenerate on $E(\mathbb{F}_p)$! This, however, can be overcome. Recall that the Weil Pairing can be used to construct an isomorphism from a cyclic subgroup of order m to the m^{th} roots of unity in an extension field. Here the extension field is \mathbb{F}_{p^2} . This then enables us to reduce ECDLP to DLP, which was our motivation for cryptanalysis and is as well for this system. With this insight, we define a modified Weil pairing \hat{e}_m utilizing ψ as follows:

$$\hat{e}_m(P, Q) = e_m(P, \psi(Q)) : \langle P \rangle \times \langle P \rangle \rightarrow \{\mu_m\}$$

where $P \in E[m]$ is the generator of $\langle P \rangle \subseteq E(\mathbb{F}_p)$, $Q \in \langle P \rangle$ and $\{\mu_m\} \subseteq \mathbb{F}_{p^2}$. This map, our isomorphism, is bilinear and non-degenerate in the same sense as the original Weil pairing. Its is also computable using Miller's algorithm for the Weil pairing as briefly discussed in Section 5. We should particularly emphasize here that for the extension in which $E[m] \subseteq E(\mathbb{F}_{q^k})$, $k = 2$. Thus, supersingularity ensures that we will be able to perform the calculations in the extension efficiently. Too large an extension, which as noted in Section 6 is most often exponentially large, makes such calculations infeasible for a cryptosystem.

We now put all this algebra to work as we outline Boneh and Franklin's IBE. The full procedure makes use of hash functions to ensure the chosen ciphertext security. To better focus on the encryption, we do not discuss these here. We assume that they are appropriately defined for the domains and ranges required of them.

Algorithm 5: Boneh-Franklin IBE

We are given p as defined above, $E(\mathbb{F}_p)$, $\langle P \rangle$ and its generator P of order m , m as defined above, ℓ such that $\ell m = p + 1$, the map \hat{e}_m and appropriate cryptographic hash functions H_1, H_2, H_3, H_4 as public system parameters. Suppose, again, A wishes to send an encrypted message to Bob, B .

Key Generation:

SYSTEM SECRET MASTER KEY (s): The system selects $s \in \mathbb{Z}_m^*$ randomly and guards this secret carefully!

SYSTEM PUBLIC KEY (P_{pub}): The system computes then publishes $P_{\text{pub}} = sP$.

USER PUBLIC KEY(Q_{ID} for identity “ID”): First the system (or any user) applies H_1 to the ID to obtain an element $y_0 \in \mathbb{F}_p$. Then we compute $x_0 = (y_0^2 + 1)^{1/3} \pmod{p-1} \in \mathbb{F}_p$. Set $Q = (x_0, y_0)$. Then set $Q_{\text{ID}} = \ell Q$ so $Q_{\text{ID}} \in \langle P \rangle$ [see [3] for proof].

USER PRIVATE KEY(R_{ID} for identity “ID”): The system computes $R_{\text{ID}} = sQ_{\text{ID}}$ which is distributed in a secure manner to the user identified by “ID”.

Encryption:

STEP 1: A computes B ’s public key, Q_B , as described above from his identity B .

STEP 2: A computes $r = H_3(\sigma, M) \in \mathbb{Z}_m^*$ where σ is a random value, kept secret.

STEP 3: A computes and sends the ciphertext $C = (rP, \sigma \oplus H_2(\mu_{m_B}^r), M \oplus H_4(\sigma))$ where $\mu_{m_B}^r = \hat{e}_m(Q_B, P_{\text{pub}})$ and \oplus is just boolean exclusive-OR.

Decryption:

STEP 1: B receives ciphertext $C = (U, V, W)$.

STEP 2: B checks if $U \in \langle P \rangle$. If not reject the whole thing.

STEP 3: B computes $\sigma = V \oplus H_2(\hat{e}_m(R_B, U))$ where \oplus is boolean exclusive-OR and R_B is Bob’s private key.

STEP 4: B computes $M' = W \oplus H_4(\sigma)$ and $r' = H_3(\sigma, M)$.

STEP 5: B checks if $U = r'P$. If so, then $r' = r$ and $M' = M$ giving B the right message. Otherwise, B rejects the ciphertext.

It looks pretty complicated. If we don’t focus on all the hashing, which again provides a high security level and give us the “boost” we need to map identities to group elements, we can see the multiple instances and variations of ECDLP. An efficient solution for ECDLP would allow us to get r from rP and worse the system’s secret key s from $P_{\text{pub}} = sP$ which would then divulge all private keys. Clearly, this “single point of weakness”, namely s , is a system vulnerability. Extra precautions need to be taken to assure its secrecy, though a “central secret” is not unheard of in the world of cryptography. Certificate Authorities (like VeriSign™) used for authentication follow such a model [15]. So, it is critical that discrete logs be difficult to find in our problem setting. This means that ECDLP must be hard for $\langle P \rangle$ and DLP must be hard for the subgroup of m^{th} roots of unity in \mathbb{F}_{p^2} . Thus, \mathbb{F}_p should be chosen sufficiently large.

8 Other Constructions for Future Consideration

Boneh and Franklin’s IBE sparked new discussion in the community about the system’s algebraic constructive elements possible improvements. A. Joux [8] noted a trade-off between identity based digital signature schemes and encryption schemes. Ideally signatures are to be as short in length as possible. This implies long public keys for the same system which could be a significant limiting factor on some systems. Galbraith [4] then examined different choices for supersingular curves to base Boneh and Franklin’s IBE, particularly ones in which the extension is greater, say $k = 6$. He succeeded in showing that a different curve gave equivalent security with shorter key lengths.

Opening the flood gates further, one may wonder about the existence of other pairings and their effectiveness. What about non-supersingular curves? Wasn’t there something previously mentioned about a “new” index calculus attack on elliptic curves? Without further ado, we give a little food for thought.

8.1 The Tate Pairing and the Frey-Ruck Reduction

The Tate pairing is of similar construction to the Weil pairing. We use exactly the same group notation as was set for Section 5. The reader may want to quickly review this section.

Definition 10 (The Tate Pairing).

$$t(P, Q) = f_p(D_Q)^{\frac{p^{kn}-1}{m}}$$

where n is such that $q = p^n$.

Because the Tate pairing is virtually identical to the Weil pairing in construction it can be utilized the same way, to break and make cryptosystems, aided by a reduction from ECDLP to DLP given by Frey and Ruck (cited in [8], [7]). Both theoretically and experimentally the FR-reduction out-performs the MOV-reduction [8], [7].

8.2 Non-Supersingular Curves

Non-supersingular curves are precisely those elliptic curves whose group order does not conform to the restriction given in Definition 5. These curves, when up against the Weil or Tate pairings, usually find $E[m]$ in a much larger, often exponential, extension of \mathbb{F}_q . Beth and Schafer provide in [2] a family of non-supersingular curves which are robust against Weil pairing attacks. Attempts, nonetheless, have been made to devise pairing attacks for these curves. Harasawa, et al. [7] extended the MOV-reduction to non-supersingular curves and were met with extremely limited success. This does not preclude such a method from existence. There has also been some work regarding the construction of IBEs with such curves. According to Galbraith [4], non-supersingular curves with low values of k should exist, however, it is often *not* the case that the elliptic curve group order is divisible by a large prime, needed as the value of m for at least Boneh and Franklin’s IBE. There may be other methods lurking out there.

8.3 “New” Index Calculus

In Spring of this year, P. Gaudry [6] published a work that extended ideas of I. Semaev of using a summation of polynomials to solve ECDLP using index calculus methods. Index calculus, as previously noted, was thought futile for elliptic curves, making this result even more exciting. In brief summary, Gaudry with the problems similar to those index calculus methods face over \mathbb{Z} . He forms a factor basis of elliptic curve points by utilizing the Weil restriction of the curve (different from the Weil pairing) to obtain an abelian variety. “Factoring” such points (ie. $P = P_1 + P_2 + \dots + P_r$) is aided by Semaev’s polynomials. Forming relations ($R = aP + bQ$) among factor basis elements gets quite complicated and Gaudry appeals to Buchberger’s algorithm. Gaudry’s method produced runtimes quite comparable to the best bounds obtained for index calculus in \mathbb{Z} for elliptic curves over \mathbb{F}_{p^n} where p is prime and $n = 2, 3, 4$. Even more recently, C. Diem [14] has seemingly extended this idea for larger values of n for certain curves.

9 Concluding Remarks

Its remarkable to see how different elements of algebra seem inter-leave themselves among cryptographic problems. With the success of research related to the Weil and Tate pairings, supersingular curves and index calculus methods we can anticipate even more to come. The author would like to thank her proofreaders David Koop and Martin Hock for their efforts and Prof. Nigel Boston for such an opportunity. I enjoyed the class.

References

- [1] K. Araki, T. Satoh and S. Miura. Overview of Elliptic Curve Cryptography. *Public Key Cryptography*, PKC’98, LNCS 1431, 29–48.
- [2] T. Beth and F. Schaefer. Non-Supersingular Elliptic Curves for Public Key Cryptography. *Advances in Cryptography - EUROCRYPT’91*, LNCS 547, 316–327, 1991.
- [3] D. Boneh and M. Franklin. Identity-Based Encryption from the Weil Pairing. *CRYPTO 2001*, Springer LNCS 2139 (2001) 213–229.
- [4] S. Galbraith. Supersingular Curves in Cryptography. *ASIACRYPT 2001*, LNCS 2248 (2001), 495–513.
- [5] T. Garefalakis. The Generalized Weil Paring and the Discrete Logarithm Problem on Elliptic Curves. *LATIN 2002*, LNCS 2286 (2002), 118–130.
- [6] P. Gaudry. Index Calculus for Abelian Varieties and the Elliptic Curve Discrete Logarithm Problem. Preprint currently available at <http://www.lix.polytechnique.fr/Labo/Pierrick.Gaudry/papers.en.html>.
- [7] R. Harasawa, J. Shikata, J. Suzuki, H. Imai. Comparing the MOV and the FR Reductions in Elliptic Curve Cryptography. *EUROCRYPT’99* LNCS 1592 (1999), 190–205.
- [8] A. Joux. The Weil and Tate Pairings and building Blocks for Public Key Cryptosystems (Survey). *ANTS 2002*, LNCS 2369 (2002), 20–32.

- [9] A. Menezes, S. Vanstone, T. Okamoto. Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field. *IEEE transaction on Information Theory*, Vol 39 (1993), 1639-1646.
- [10] J. Silverman and J. Suzuki. Elliptic Curve Discrete Logarithms and the Index Calculus. *ASI-ACRYPT'98*, LNCS 1514 (1998), 110-125.
- [11] A. Shamir Identity-Based Cryptosystems and Signature Schemes. *CRYPTO'84*, LNCS 196 (1985), 47-53.
- [12] <http://en.wikipedia.org/wiki/ElGamal>.
- [13] http://en.wikipedia.org/wiki/Cyclic_group.
- [14] <http://www.isg.rhul.ac.uk/~sdg/ecc.html>.
- [15] http://www.webopedia.com/TERM/C/certification_authority.html