Contents

Chapter 1

Elliptic curve cryptography

Elliptic curve cryptography, in essence, entails using the group of points on an elliptic curve as the underlying number system for public key cryptography. There are two main reasons for using elliptic curves as a basis for public key cryptosystems. The first reason is that elliptic curve based cryptosystems appear to provide better security than traditional cryptosystems for a given key size. One can take advantage of this fact to increase security, or (more often) to increase performance by reducing the key size while keeping the same security. The second reason is that the additional structure on an elliptic curve can be exploited to construct cryptosystems with interesting features which are difficult or impossible to achieve in any other way. A notable example of this phenomenon is the development of identity based encryption and the accompanying emergence of pairing-based cryptographic protocols.

1.1 Motivation

Elliptic curves are useful in cryptography because the set of points on an elliptic curve form a group, and the discrete logarithm problem has been observed to be very hard on this group. In this section we review the basic facts about groups and discrete logarithms and explain the relationship between discrete logarithms and cryptography.

1.1.1 Groups in cryptography

Recall that a group $(G, \cdot)$ is a set G equipped with a binary operation satisfying the properties of associativity, existence of identity, and existence of inverses. Many of the most important cryptographic protocols are based upon groups, or can be described generically in terms of groups. For example, the Diffie-Hellman key exchange protocol [19], which was the first public key cryptography protocol ever published, can be described as follows:

Protocol 1.1.1. [Diffie-Hellman key exchange protocol] Two parties, named Alice and Bob, wish to establish a common secret key without making use of any private communication.

- Alice and Bob agree on a group G, and an element $g \in G$.
- Alice selects a secret value α, and sends g^α to Bob.
- Bob selects a secret value β, and sends g^β to Alice.
- Alice and Bob compute the shared secret $g^{\alpha\beta} = (g^\alpha)^\beta = (g^\beta)^\alpha$.

In Diffie and Hellman's original publication [19], the group G is specified to be the multiplicative group $\mathbb{Z}_p^*$ of nonzero integers modulo p, and the element g is specified to be a generator of G. However, it is clear from the above description that the protocol is not limited to this group, and that other groups can also be used.

1.1.2 Discrete logarithms

We wish to quantitatively measure the extent to which a group G is suitable for use in cryptographic protocols such as Diffie-Hellman. In order to do this, we recall the definition of discrete logarithms. Given any two group elements g and h, the *discrete logarithm* of h with respect to g, denoted $\mathrm{DLOG}_g(h)$, is the smallest non-negative integer x such that $g^x = h$ (if it exists). An adversary capable of computing discrete logarithms in G can easily break the Diffie-Hellman protocol. Therefore, in order for a group to be useful in Diffie-Hellman or in public key cryptography, the discrete logarithm problem in the group must be computationally difficult.

It is known that, in any group G with n elements, the computation of discrete logarithms can be performed probabilistically in expected time at most $O(\sqrt{n})$, using the Pollard rho algorithm [45]. This figure represents the maximum amount of security that one can hope for. Most groups, however, fall short of this theoretical maximum.

For example, consider the multiplicative group $\mathbb{Z}_p^*$ of nonzero integers modulo p, or more generally the multiplicative group $\mathbb{F}_q^*$ of nonzero elements in any finite field $\mathbb{F}_q$. Using the index calculus algorithm [34], discrete logarithms in this group can be computed probabilistically in $L_q(1/3, (128/9)^{1/3})$ expected time in the worst case, where q is the size of the field. Here $L_q(\alpha, c)$ denotes the standard expression

$$L_q(\alpha, c) = \exp((c + o(1))(\log q)^\alpha (\log\log q)^{1-\alpha})$$

interpolating between quantities polynomial in $\log q$ (when $\alpha = 0$) and exponential in $\log q$ (when $\alpha = 1$). Note that the theoretical optimum of $O(\sqrt{n}) = O(\sqrt{q})$ corresponds to $L_q(1, 1/2)$. Hence, in the multiplicative group of a finite field, the best known algorithms for computing discrete logarithms run in substantially faster than exponential time.

Elliptic curves over a finite field are of interest in cryptography because in most cases there is no known algorithm for computing discrete logarithms on the group of points of such an elliptic curve in faster than $O(\sqrt{n})$ time. In other words, elliptic curves are conjectured to attain the theoretical maximum possible level of security in the public key cryptography setting.

1.2 Definitions

This section contains the basic definitions for elliptic curves and related constructions such as the group law.

1.2.1 Finite fields

We briefly review the definition of a field, which plays a crucial role in the theory of elliptic curves. A *field* is a set equipped with two binary operations, $+$ (addition) and $\cdot$ (multiplication), which admit additive and multiplicative inverses, distinct additive and multiplicative identities, and satisfy the associative, commutative, and distributive laws. Examples of fields include $\mathbb{Q}$ (rational numbers), $\mathbb{R}$ (real numbers), $\mathbb{C}$ (complex numbers), and $\mathbb{Z}_p$ (integers modulo a prime p).

A *finite field* is a field with a finite number of elements. Every finite field has size equal to p^m for some prime p. For each pair (p, m), there is exactly one finite field of size $q = p^m$, up to isomorphism, and we denote this field $\mathbb{F}_{p^m}$ or $\mathbb{F}_q$. (In the literature, the field $\mathbb{F}_q$ is often called a *Galois field*, and denoted $\mathrm{GF}(q)$. In this chapter, however, we will use the $\mathbb{F}_q$ notation throughout.)

When $q = p$ is prime, the field $\mathbb{F}_p$ is equal to the field $\mathbb{Z}_p$ of integers modulo p. When $q = p^m$ is a prime power, the field $\mathbb{F}_{p^m}$ can be obtained by taking the set $\mathbb{F}_p[X]$ of all polynomials in X with coefficients in $\mathbb{F}_p$, modulo any single irreducible polynomial of degree m.

Example 1.2.1 (The finite field $\mathbb{F}_9$). The polynomial $X^2 + 1$ is irreducible in $\mathbb{F}_3[X]$ (does not factor into any product of smaller degree polynomials). The elements of $\mathbb{F}_9$ are given by

$$\mathbb{F}_9 = \{0, 1, 2, X, X+1, X+2, 2X, 2X+1, 2X+2\}.$$

Addition and multiplication in $\mathbb{F}_9$ are performed modulo 3 and modulo X^2+1, e.g.

$$\begin{aligned}(X+1) + (X+2) &= 2X + 3 = 2X \\ (X+1) \cdot (X+2) &= X^2 + X + 2X + 2 = X^2 + 3X + 2 \\ &= X^2 + 2 = (X^2+2) - (X^2+1) = 1\end{aligned}$$

The *characteristic* of a field F, denoted $\mathrm{char}(F)$, is the size of the smallest subfield in the field, or 0 if this subfield has infinite size. In the case of a finite field $\mathbb{F}_{p^m}$, the characteristic is always equal to p.

1.2.2 Elliptic curves

Roughly speaking, an elliptic curve is the set of points over a field satisfying a cubic equation in two variables x and y. By employing various substitutions, the general case can be reduced to one in which the y variable has degree 2 and the x variable has degree 3. In addition to the usual points of the form (x, y), there is an extra point, denoted ∞, which serves as the identity element in the group. The following is the technical definition of an elliptic curve. Note that Definition 1.2.2 is only for fields of characteristic not equal to 2; the characteristic 2 case is treated separately in Definition 1.2.7. Although it is possible to give a single definition that covers all cases, we have elected to use separate definitions for reasons of clarity.

Definition 1.2.2 (Elliptic curves in characteristic $\neq 2$)**.** Let F be a field whose characteristic is not equal to 2. An *elliptic curve* E defined over F, denoted E or E/F, is a set of the form

$$E = E(F) = \{(x, y) \in F^2 \mid y^2 = x^3 + a_2x^2 + a_4x + a_6\} \cup \{\infty\},$$

where a_2, a_4, a_6 are any three elements of F such that the discriminant $a_2^2a_4^2 - 4a_4^3 - 4a_2^3a_6 + 18a_2a_4a_6 - 27a_6^2$ of the polynomial $x^3 + a_2x^2 + a_4x + a_6$ is nonzero. The points of the form (x, y) are called *finite points* of E, and the point ∞ is called the *point at infinity*.

Essentially, an elliptic curve is the set of points (x, y) lying on a curve $f(x, y) = 0$, where $f(x, y) = y^2 - (x^3 + a_2x^2 + a_4x + a_6)$. This definition is analogous to the definition of the multiplicative group F^* as the set of points (x, y) satisfying $xy = 1$. The extra point ∞ is not a point in F^2; instead it arises from the mathematical point of view when considering E as a curve in projective space.

The cubic polynomial $x^3 + a_2x^2 + a_4x + a_6$ is called the *Weierstrass cubic* of E. The condition that the discriminant is nonzero is equivalent to requiring that the Weierstrass cubic have three distinct roots over (any algebraic closure of) F. This condition also ensures that the partial derivatives $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$ are never both zero on E. The nonvanishing of partial derivatives, in turn, implies that every finite point on E has a unique tangent line, a fact which is necessary in order to define the group law (Definition 1.2.4).

If the characteristic of F is not equal to either 2 or 3, then the substitution $x \leftarrow x - \frac{a_2}{3}$ eliminates the a_2 term from the Weierstrass cubic, leaving the simplified equation $y^2 = x^3 + ax + b$. In this case the discriminant of the Weierstrass cubic is equal to $-(4a^3 + 27b^2)$.

Example 1.2.3. Consider the elliptic curve $E : y^2 = x^3 + x + 6$ defined over the finite field $\mathbb{F}_{11}$ of 11 elements. The discriminant of the Weierstrass cubic is $-(4 \cdot 1^3 + 27 \cdot 6^2) \equiv 3 \bmod 11$, which is nonzero. There are thirteen points on the elliptic curve $E/\mathbb{F}_{11}$, as follows:

$$\begin{aligned} E(\mathbb{F}_{11}) = \{&\infty, (2, 4), (2, 7), (3, 5), (3, 6), (5, 2), (5, 9), \\ &(7, 2), (7, 9), (8, 3), (8, 8), (10, 2), (10, 9)\}. \end{aligned}$$

One can verify directly that each point lies on E. For example, $9^2 \equiv 10^3 + 10 + 6 \equiv 4 \bmod 11$, so $(10, 9)$ is on E.

1.2.3 Group law

We now provide a definition of the group law on an elliptic curve, valid when F has characteristic not equal to 2. If the characteristic of F is 2, then a different set of definitions is needed (Sect. 1.2.4).

Definition 1.2.4 (Group law—geometric definition)**.** Let F be a field whose characteristic is not equal to 2. Let

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

be an elliptic curve defined over F. For any two points P and Q in E, the point $P + Q$ is defined as follows.

- If $Q = \infty$, then $P + Q = P$.
- If $P = \infty$, then $P + Q = Q$.

In all other cases, let L be the unique line through the points P and Q. If $P = Q$, then let L be the unique tangent line to the curve $y^2 = x^3 + a_2x^2 + a_4x + a_6$ at P.

- If L does not intersect the curve $y^2 = x^3 + a_2x^2 + a_4x + a_6$ at any point other than P or Q, then define $P + Q = \infty$.
- Otherwise, the line L intersects the curve $y^2 = x^3 + a_2x^2 + a_4x + a_6$ in exactly one other point $R = (x', y')$.
- Define $P + Q = (x', -y')$.

Although Definition 1.2.4 is of a geometric nature, using it one can derive algebraic equations for $P + Q$ in terms of P and Q. In this way, we obtain a purely algebraic definition of the group law:

Definition 1.2.5 (Group law—algebraic definition)**.** Let F be a field whose characteristic is not equal to 2. Let

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

be an elliptic curve defined over F. For any two points P and Q in E, the point $P + Q$ is defined as follows.

- If $Q = \infty$, then $P + Q = P$.
- If $P = \infty$, then $P + Q = Q$.

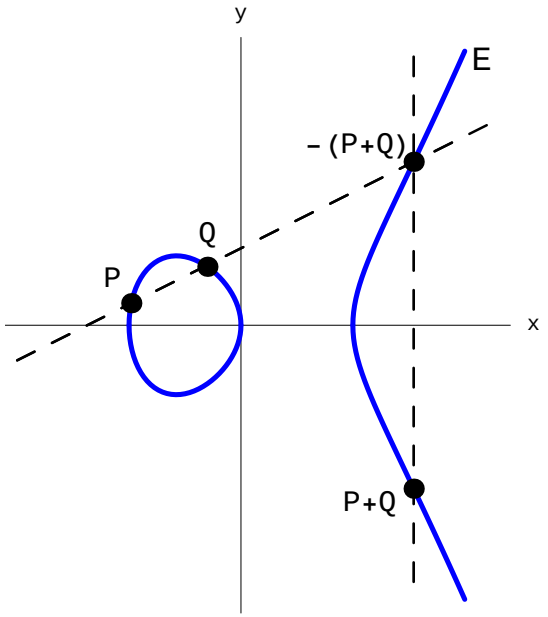

Figure 1.1: Illustration of the group law on an elliptic curve E

In all other cases, we can write $P = (x_1, y_1)$ and $Q = (x_2, y_2)$. If $x_1 = x_2$ and $y_1 = -y_2$, then define $P + Q = \infty$. Otherwise, set

$$
\begin{aligned}
m &= \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} & \text{if } P \neq Q, \\ \frac{3x_1^2 + 2a_2x_1 + a_4}{2y_1} & \text{if } P = Q. \end{cases} \\
x_3 &= m^2 - x_1 - x_2 \\
y_3 &= -(m(x_3 - x_1) + y_1)
\end{aligned}
$$

and define $P + Q$ to be the point (x_3, y_3).

In order to form a group, the addition operation must be associative and admit an additive identity and additive inverses. From the definition, it is easy to see that the addition operation is commutative, with identity element ∞ and inverse element $-P = (x, -y)$ for any point $P = (x, y)$. The associativity property is much harder to prove. One can show that the operation is associative by calculating the two quantities $(P+Q)+R$ and $P+(Q+R)$ using Definition 1.2.5 under a computer algebra system, but such a proof is tedious and we therefore omit the proof here.

Example 1.2.6 (Point addition). Let $E : y^2 = x^3 + x + 6$ be the curve given in

Example 1.2.3, defined over $\mathbb{F}_{11}$. We have:

$$\begin{aligned}(2,4)+(2,4) &= (5,9)\\ (2,4)+(5,2) &= (2,7)\\ ((2,4)+(2,4))+(5,2) &= (5,9)+(5,2)=\infty\\ (2,4)+((2,4)+(5,2)) &= (2,4)+(2,7)=\infty\end{aligned}$$

The last two computations illustrate the associativity property.

1.2.4 Elliptic curves in characteristic 2

For implementation purposes, it is often preferable to work over fields of characteristic 2 in order to take advantage of the binary nature of computer architectures. Hence, for completeness, we provide the applicable definitions and formulas in the characteristic 2 case.

Definition 1.2.7 (Elliptic curves in characteristic 2)**.** Let F be a field of characteristic 2. An *elliptic curve* E defined over F is a set of the form

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

where either

$$\begin{matrix} a_1 = 1 \\ a_3 = a_4 = 0 \\ a_6 \neq 0 \end{matrix} \quad \text{or} \quad \begin{matrix} a_1 = a_2 = 0 \\ a_3 \neq 0 \end{matrix}$$

For any two points P and Q on E, the point $P+Q$ is defined as follows:

- If $Q = \infty$, then $P+Q = P$.
- If $P = \infty$, then $P+Q = Q$.

In all other cases, we can write $P = (x_1, y_1)$ and $Q = (x_2, y_2)$. If $x_1 = x_2$ and $y_1 + y_2 + a_1x_1 + a_3 = 0$, then define $P+Q = \infty$. Otherwise, set

$$\begin{aligned} m &= \begin{cases} \frac{y_2-y_1}{x_2-x_1} & \text{if } P \neq Q, \\ \frac{3x_1^2+2a_2x_1+a_4-a_1y_1}{2y_1+a_1x_1+a_3} & \text{if } P = Q. \end{cases}\\ x_3 &= m^2 + a_1m - a_2 - x_1 - x_2\\ y_3 &= -(m(x_3-x_1) + y_1 + a_1x_3 + a_3)\end{aligned}$$

and define $P+Q$ to be the point (x_3, y_3).

1.3 Implementation issues

We now provide an overview of various topics related to implementations of elliptic curve cryptosystems. Because of space limitations, only the most essential material is presented here. More comprehensive and detailed treatments can be found in Hankerson et al. [28] or Cohen et al. [17].

1.3.1 Scalar multiplication

On an elliptic curve, the group operation is denoted additively. In such a group, the group exponentiation operation is also written using additive notation; that is, instead of using g^α to denote the α-fold product $g \times g \times \cdots \times g$ for $g \in G$, we use the notation αP to denote the α-fold sum $P + P + \cdots + P$ for $P \in E$. The process of multiplying a group element P by an integer α is known as *scalar multiplication*.

Virtually all cryptographic protocols based on elliptic curves, including the Diffie-Hellman protocol (Protocol 1.1.1) and the protocols in Sect. 1.4, rely on the ability to perform scalar multiplication efficiently. The standard algorithm for computing αP, known as double-and-add or square-and-multiply, is a recursive algorithm which accomplishes this task using $O(\log \alpha)$ group operations. Algorithm 1.3.1 contains an implementation of the double-and-add algorithm in pseudocode.

Algorithm 1.3.1 The double-and-add algorithm.

Given: $P \in E$, $\alpha \in \mathbb{N}$. **Output:** αP.

if $\alpha = 0$ **then**
  output ∞
else if α is even **then**
  $\beta \leftarrow \frac{\alpha}{2}$
  $Q \leftarrow \beta P$
  output $Q + Q$
else if α is odd **then**
  $\beta \leftarrow \alpha - 1$
  $Q \leftarrow \beta P$
  output $Q + P$
end if

Faster algorithms are available and are often appropriate depending on the situation. On an elliptic curve, computing additive inverses is almost free, and thus it is possible to speed up scalar multiplication using non-adjacent form representations [41]. Other approaches include the use of double-base number systems [20], and (in some cases) the use of special curves such as Edwards curves [8] or curves with additional endomorphisms [27].

Example 1.3.1 (Certicom ECC challenge). This example is taken from the Certicom ECCp-109 challenge [15]. Let

$$p = 564538252084441556247016902735257$$
$$a = 321094768129147601892514872825668$$
$$b = 430782315140218274262276694323197$$

and consider the elliptic curve $E : y^2 = x^3 + ax + b$ over $\mathbb{F}_p$. Let P be the point

$$(973390109870590665231561339089 35, 149670372846169285760682371978898)$$

on E, and let $k = 281183840311601949668207954530684$. The value of kP is

$$(44646769697405861057630861884284, 522968098895785888047540374779097).$$

1.3.2 Curve selection

Consider an elliptic curve E defined over a finite field $F = \mathbb{F}_q$. The number of points on E is finite, since, with the exception of ∞, the points on E have the form $(x, y) \in \mathbb{F}_q^2$. However, not all of these curves are equally suitable for cryptography. For example, in any group having cardinality n where n is composite, it is possible to compute discrete logarithms in $O(\sqrt{p})$ time where p is the largest prime divisor of n, using the Pohlig-Hellman algorithm [44]. Therefore, in order to be suitable for cryptographic purposes, the number of points on a curve should be equal to a prime, or at least admit a large prime divisor.

One way to find such a curve is to select curves at random and compute their cardinalities until an appropriate curve is found. A classical result in algebraic geometry, known as the Hasse-Weil bound, states that for any elliptic curve $E/\mathbb{F}_q$ the number of points $\#E$ on $E(\mathbb{F}_q)$ lies within the interval

$$q + 1 - 2\sqrt{q} \leq \#E \leq q + 1 + 2\sqrt{q}.$$

Moreover, Lenstra [35] has shown that for any subset consisting of a non-negligible proportion of numbers within this interval, a non-negligible proportion of elliptic curves $E/\mathbb{F}_q$ have cardinality within that subset. This result indicates that, in practice, a randomly chosen curve will with high probability have prime cardinality. In order to determine the cardinality of such a curve, it is necessary to employ a fast point counting algorithm. Examples of such algorithms include the Schoof-Elkies-Atkin algorithm [23, 36, 47] and the Satoh algorithm [46].

Use of pre-computed curves. An alternative approach is to use a pre-computed elliptic curve which has been verified ahead of time to possess good cryptographic properties. For example, the NIST FIPS 186-2 standard [43] contains 15 different pre-computed curves, including the curve P-192 given by

$$\begin{aligned} p &= 6277101735386680763835789423207666416083908700390324961279 \\ b &= 2455155546008943817740293915197451784769108058161191238065 \\ E &: y^2 = x^3 - 3x + b \text{ over } \mathbb{F}_p \end{aligned}$$

This curve has cardinality equal to

$$\#E = 6277101735386680763835789423176059013767194773182842284081$$

which is a prime.

1.3.3 Point representations

As we have seen, a point P on an elliptic curve is given by a pair of coordinates (x, y). Over a finite field $\mathbb{F}_q$, each coordinate requires $\lg(q)$ bits for transmission or storage. Hence, the naive representation of a point on an elliptic curve requires $2\lg(q)$ bits. In many situations, it is desirable for efficiency reasons to use smaller representations. One such optimization is to represent a point using $\lg(q) + 1$ bits by storing only the x-coordinate and determining y at runtime (e.g. via the formula $y = \sqrt{x^3 + a_2x^2 + a_4x + a_6}$ when the characteristic is not 2). Here the x-coordinate requires $\lg(q)$ bits and the extra bit is used to store the sign of the y-coordinate. This technique, known as *point compression*, is described in ANSI X9.62 [3] and in U.S. Patent 6,252,960 [48].

An alternative technique is to transmit only the x-coordinate of P with no additional information. In this case, the recipient must tolerate some ambiguity in the value of P, because there are two possible choices for the y-coordinate. Using the wrong value for the y-coordinate corresponds to using the point $-P$ instead of P. However, in the vast majority of ECC protocols, the central encryption or decryption operation involves a scalar multiplication of the form kP for some integer k. Note that, regardless of whether P or $-P$ is used in the computation of kP, the x-coordinate of the result is the same. In particular, this property holds for the hashed ElGamal and ECIES protocols described in Sect. 1.4.1, as well as for the BLS protocol (Sect. 1.5.2). Hence, for these protocols, one can choose to represent points using only their x-coordinates without affecting the validity of the protocols. This technique does not apply to ECDSA (Protocol 1.4.7), since the ECDSA protocol is already designed to transmit only the x-coordinate.

1.3.4 Generating points of prime order

In most elliptic curve-based protocols, it is necessary to generate a base point of order n where n is a large prime; that is, a point $P \neq \infty$ such that $nP = \infty$. When the cardinality of a curve E is prime, any non-identity point is suitable as a base point. Otherwise, we write the cardinality of E as a product of the form $\#E = hn$ where n is the largest prime factor. The integer h is called the *cofactor* of E. Since the cryptographic strength of E depends on n (cf. Sect. 1.1.2), it is best to maximize n, or in other words minimize h. In particular, we assume E is chosen so that $h \ll \sqrt{n}$. For such values of h, a base point P on E of order n can be obtained by computing $P = hQ$ where Q is any randomly selected point on E.

1.4 ECC protocols

In this section we provide some examples of elliptic curve cryptography (ECC) protocols that have been developed and proposed. Whenever possible, we give preference to protocols which have been approved in government or international standards documents.

1.4.1 Public key encryption

Protocol 1.4.1 (Textbook ElGamal encryption)**.** The textbook ElGamal protocol is one of the oldest and simplest public key encryption schemes. Here we give a straightforward adaptation of the classic ElGamal encryption scheme [22] to the setting of elliptic curves. We emphasize that this textbook protocol is for *illustration* purposes only, is *insecure* against active attackers, and *should not* be used except in very limited circumstances (see Remark 1.4.2).

Public parameters: An elliptic curve E defined over a finite field $\mathbb{F}_q$, and a base point $P \in E(\mathbb{F}_q)$ of large prime order n.

Key generation: Choose a random integer α in the interval $1 \leq \alpha < n$. The public key is αP. The private key is α.

Encryption: The message space is the set of all points $Q \in E(\mathbb{F}_q)$. To encrypt a message M, choose a random integer r between 0 and n, and compute

$$C_1 = rP$$
$$C_2 = r\alpha P + M$$

The ciphertext is (C_1, C_2).

Decryption: Given a ciphertext (C_1, C_2), compute

$$M' = C_2 - \alpha C_1$$

and output the plaintext M'.

Remark 1.4.2. The textbook ElGamal scheme is *malleable* [21], meaning that given a valid encryption for M it is possible to construct valid encryptions for related messages such as $2M$. In rare situations, such as when designing electronic voting schemes [18], this property is desirable, but in most cases malleability represents a security shortcoming and should be avoided.

Remark 1.4.3. In addition to the security shortcomings mentioned above, one drawback of the textbook ElGamal protocol is that it takes some work to transform an arbitrary binary string into an element of the message space, i.e. a point on the curve. In hashed ElGamal (Protocol 1.4.5) and ECIES (Protocol 1.4.6), this problem is addressed through the use of a hybrid public-key/symmetric-key scheme.

Example 1.4.4 (Textbook ElGamal with small parameters). Let $p = 2^{40} + 15 = 1099511627791$, $a = -3$, and $b = 786089953074$. Let E be the curve $y^2 = x^3 + ax + b$ defined over $\mathbb{F}_p$. Let P be the base point $(39282146988, 43532161490)$ on E. Then the point P has order 1099510659307, which is a prime. Using these parameters, a sample encryption and decryption operation is performed below.

Key generation: We choose $\alpha = 482363949216$ at random, and compute $\alpha P = (991136913417, 721626930099)$. The public key is αP and the private key is 482363949216.

Encryption: Suppose our message is $M = (556486217561, 262617177881)$. We choose $r = 843685127620$ at random, and compute

$$C_1 = rP = (332139500006, 485511205375)$$
$$C_2 = r\alpha P + M = (484509366473, 588381554550)$$

Decryption: One can check that $C_1 - \alpha C_2 = M$ for the above pair (C_1, C_2).

Protocol 1.4.5 (Hashed ElGamal encryption)**.** The hashed ElGamal scheme and its variants appear in [1, 14] and in the ANSI X9.63 standard [2]. This scheme is secure against passive (eavesdropping) attacks, but depending on the symmetric key encryption scheme that is used, may not be secure against active adversaries who are capable of obtaining decryptions of related messages. Compared to the ECIES protocol given in Protocol 1.4.6, the two protocols are identical except for the addition of a MAC in ECIES, which protects against active adversaries.

Public parameters: An elliptic curve E defined over a finite field $\mathbb{F}_q$, a base point $P \in E(\mathbb{F}_q)$ of large prime order n, a *key derivation function* H (based on a hash function), and a symmetric key encryption scheme $(\mathcal{E}, \mathcal{D})$.

Key generation: Choose a random integer α in the interval $1 \leq \alpha < n$. The public key is αP. The private key is α.

Encryption: The message space is the set of all binary strings. To encrypt a message m, choose a random integer r between 0 and n, and compute

$$Q = rP$$
$$k = H(r\alpha P)$$
$$c = \mathcal{E}_k(m)$$

The ciphertext is (Q, c).

Decryption: Given a ciphertext (Q, c), compute

$$k' = H(\alpha Q)$$
$$m' = \mathcal{D}_{k'}(c)$$

and output the plaintext m'.

Protocol 1.4.6 (Elliptic Curve Integrated Encryption Scheme (ECIES))**.** This protocol is the same as the hashed ElGamal scheme of Protocol 1.4.5 except for the addition of a message authentication code, or MAC, which affords some protection against active adversaries. It is part of the ANSI X9.63 standard [2].

Public parameters: An elliptic curve E defined over a finite field $\mathbb{F}_q$, a base point $P \in E(\mathbb{F}_q)$ of large prime order n, a *key derivation function* H which outputs a pair of keys, a *message authentication code* M, and a symmetric key encryption scheme $(\mathcal{E}, \mathcal{D})$.

Key generation: Choose a random integer α in the interval $1 \leq \alpha < n$. The public key is αP. The private key is α.

Encryption: The message space is the set of all binary strings. To encrypt a message m, choose a random integer r between 0 and n, and compute

$$\begin{aligned} Q &= rP \\ (k_1, k_2) &= H(r\alpha P) \\ c &= \mathcal{E}_{k_1}(m) \\ d &= M(k_2, c) \end{aligned}$$

The ciphertext is (Q, c, d).

Decryption: Given a ciphertext (Q, c, d), compute

$$\begin{aligned} (k_1', k_2') &= H(\alpha Q) \\ d' &= M(k_2', c) \end{aligned}$$

If $\alpha Q = \infty$ or $d \neq d'$, output NULL. Otherwise, compute

$$m' = \mathcal{D}_{k_1'}(c)$$

and output the plaintext m'.

1.4.2 Digital signatures

Protocol 1.4.7 (Elliptic Curve Digital Signature Algorithm (ECDSA))**.** The Elliptic Curve Digital Signature Algorithm is an adaptation of the Digital Signature Algorithm [9] to the elliptic curve setting. ECDSA is described in the ANSI X9.62 standard [3]. In the description below, the expression $x(Q)$ denotes the x-coordinate of a point $Q \in E$.

Public parameters: An elliptic curve E defined over $\mathbb{F}_p$, a base point $P \in E(\mathbb{F}_p)$ of large prime order n, and a hash function $H\colon \{0,1\}^* \to \mathbb{Z}_n$. In the ANSI X9.62 standard [3], the function H is specified to be SHA-1 [42].

Key generation: Choose a random integer α in the interval $1 \leq \alpha < n$. The public key is αP and the private key is α.

Signing: The message space is the set of all binary strings. To sign a message m, choose a random integer k in the interval $1 \leq k < n$. Compute

$$\begin{aligned} r &= x(kP), \\ s &= \frac{H(m) + \alpha r}{k} \bmod n. \end{aligned}$$

The signature of m is $\sigma = (r, s)$.

Verification: Check whether $0 < r < n$ and $0 < s < n$. If so, calculate

$$x\left((s^{-1} \bmod n)(H(m)P + r(\alpha P))\right) = x\left(\frac{H(m) + \alpha r}{s} \cdot P\right)$$

The signature is valid if and only if the above value equals r.

Example 1.4.8 (ECDSA signature generation). Let E be the curve P-192 given in Sect. 1.3.2. Let P be the point

$$\begin{aligned} P =&(6020462823756886567582134805875261119166989766368846848 18,\\ &1740503322936220314048575522802194103640234889273866506 41) \end{aligned}$$

on E. As indicated in Sect. 1.3.2, the point P has order

$$n = 6277101735386680763835789423176059013767194773182842284081$$

which is a prime. We use the hash function SHA-1 for H. Suppose that our private key is

$$\alpha = 9112467240057525352231330868224809147704361793152292787 9$$

and we wish to sign the ASCII message `Hello world!` (with no trailing newline). The SHA-1 hash of this message is

$$\begin{aligned} \text{SHA-1}(\texttt{Hello world!}) &= \text{d3486ae9136e7856bc42212385ea797094475802}_{16}\\ &= 1206212019512053528979580233526017047056064403458 \end{aligned}$$

To sign the message, we choose a random value

$$k = 5041532312768674859943633328080661292870652213606844754 61$$

and compute

$$\begin{aligned} r &= x(kP)\\ &= 2657489544731026965723991092274654411104210887805224396626\\ s &= \frac{H(m) + \alpha r}{k} \bmod n\\ &= 1131215894271817774617160471390853260507893393838210881939 \end{aligned}$$

The signature is (r, s). Note that even though E is defined over a field of 192 bits, the signature is 384 bits long because it consists of two elements mod n.

1.4.3 Public key validation

In most cases, achieving optimal security requires verifying that the points given in the public parameters or the public key actually lie within the elliptic curve in question. Failure to perform public key validation leads to a number of potential avenues for attack [4] which under a worst-case scenario can reveal the secret key. If we let E, $\mathbb{F}_q$, n, P, and αP denote the curve, field, order of the base point, base point, and public key respectively, then validation in this context means checking all of the following:

1. $q = p^m$ is a prime power.
2. The coefficients of E are in $\mathbb{F}_q$.
3. The discriminant of E is nonzero.
4. The integer n is prime and sufficiently large ([2] recommends $n > 2^{160}$).
5. The point P satisfies the defining equation for E, and the equations $P \neq \infty$ and $nP = \infty$.
6. The point αP satisfies the defining equation for E, and the equations $\alpha P \neq \infty$ and $n\alpha P = \infty$.

Items 1 through 5 need to be checked once, and item 6 once per public key.

Remark 1.4.9. The ANSI X9.62 [3] and X9.63 [2] standards also stipulate that the curve E should have large embedding degree (Definition 1.6.1), in order to avoid the MOV reduction (Sect. 1.6.2). This requirement is beneficial in most situations, but it cannot be met when employing pairing-based cryptography, since pairing-based cryptography requires small embedding degrees.

1.5 Pairing based cryptography

Initially, elliptic curves were proposed for cryptography because of their greater strength in discrete logarithm based protocols, which led to the development of shorter, more efficient cryptosystems at a given security level. However, in recent years, elliptic curves have found a major new application in cryptography thanks to the existence of bilinear pairings on certain families of elliptic curves. The use of bilinear pairings allows for the construction of entirely new categories of protocols, such as identity-based encryption (IBE) and short digital signatures. In this section we define the concept of bilinear pairings, state some of the key properties and limitations of pairings, and give an overview of what types of constructions are possible with pairings.

1.5.1 Definitions

We begin by presenting the basic definitions of bilinear pairings along with some motivating examples of pairing-based protocols. A priori, there is no relationship between bilinear pairings and elliptic curves, but in practice all commonly used pairings are constructed with elliptic curves (see Sect. 1.7).

Definition 1.5.1. A *bilinear pairing*, *cryptographic pairing*, or *pairing* is an efficiently computable group homomorphism

$$e\colon G_1 \times G_2 \to G_T$$

defined on prime order cyclic groups G_1, G_2, G_T, with the following two properties:

Bilinearity: For all $P_1, P_2, P \in G_1$ and $Q_1, Q_2, Q \in G_2$,

$$e(P_1 + P_2, Q) = e(P_1, Q) \cdot e(P_2, Q),$$
$$e(P, Q_1 + Q_2) = e(P, Q_1) \cdot e(P, Q_2).$$

Non-degeneracy: For all $P_0 \in G_1$ and $Q_0 \in G_2$,

$$e(P_0, Q) = 1 \text{ for all } Q \in G_2 \implies P_0 = \mathrm{id}_{G_1}$$
$$e(P, Q_0) = 1 \text{ for all } P \in G_1 \implies Q_0 = \mathrm{id}_{G_2}$$

Note that, as a consequence of the definition, the groups G_1 and G_2 have a common order $\#G_1 = \#G_2 = n$, and the image of the pairing in G_T has order n as well. In the literature, it is common to see the pair of groups (G_1, G_2) referred to as a *bilinear group pair*. All known usable examples of bilinear pairings are derived by taking G_1 and G_2 to be subgroups of an elliptic curve, and G_T to be a multiplicative subgroup of a finite field. Therefore, we will denote the group operation in G_1 and G_2 using additive notation, and G_T using multiplicative notation.

Sometimes a cryptographic protocol will require a pairing that satisfies some additional properties. The following classification from [26] is used to distinguish between different types of pairings.

Definition 1.5.2. A bilinear pairing $e\colon G_1 \times G_2 \to G_T$ is said to be a:

Type 1 pairing if either $G_1 = G_2$ or there exists an efficiently computable isomorphism $\phi\colon G_1 \to G_2$ with efficiently computable inverse $\phi^{-1}\colon G_2 \to G_1$. These two formulations are equivalent, since when $G_1 \neq G_2$ one can always represent an element $g \in G_2$ using $\phi^{-1}(g) \in G_1$.

Type 2 pairing if there exists an efficiently computable isomorphism $\psi\colon G_2 \to G_1$, but there does not exist any efficiently computable isomorphism from G_1 to G_2.

Type 3 pairing if there exist no efficiently computable isomorphisms from G_1 to G_2 or from G_2 to G_1.

Many pairing-based cryptographic protocols depend on the *bilinear Diffie-Hellman* (BDH) assumption, which states that the BDH problem defined below is intractable:

Definition 1.5.3. Let $e\colon G_1 \times G_2 \to G_T$ be a bilinear pairing. The *bilinear Diffie-Hellman* (BDH) problem is the following computational problem: given $P, \alpha P, \beta P \in G_1$ and $Q \in G_2$, compute $e(P, Q)^{\alpha\beta}$.

Note that in the special case where e is a Type 1 pairing with $G_1 = G_2$, the BDH problem is equivalent to the following problem: given $P, \alpha P, \beta P, \gamma P \in G_1$, compute $e(P, P)^{\alpha\beta\gamma}$. This special case is more symmetric and easier to remember.

1.5.2 Pairing-based protocols

Protocol 1.5.4 (Tripartite one-round key exchange)**.** The Diffie-Hellman protocol (Protocol 1.1.1) allows for two parties to establish a common shared secret using only public communications. A variant of this protocol, discovered by Joux [33], allows for three parties A, B, and C to establish a common shared secret in one round of public communication. To do this, the parties make use of a Type 1 pairing e with $G_1 = G_2$, and a base point $P \in G_1$. Each participant chooses respectively a secret integer α, β, and γ, and broadcasts respectively αP, βP, and γP. The quantity

$$e(P,P)^{\alpha\beta\gamma} = e(\alpha P, \beta P)^{\gamma} = e(\beta P, \gamma P)^{\alpha} = e(\gamma P, \alpha P)^{\beta}$$

can now be calculated by anyone who has knowledge of the broadcasted information together with at least one of the secret exponents α, β, γ. An eavesdropper without access to any secret exponent would have to solve the bilinear Diffie-Hellman problem in order to learn the common value.

Identity-based encryption. The most notable application of pairings to date is the seminal construction of an identity based encryption scheme by Boneh and Franklin [12]. An *identity based encryption* scheme, or IBE, is a public key cryptosystem with the property that any string constitutes a valid public key. Unlike traditional public key encryption, IBE requires private keys to be generated by a trusted third party instead of by individual users.

Protocol 1.5.5 (Boneh-Franklin IBE)**.** The Boneh-Franklin IBE scheme comes in two versions, a basic version which is secure against a passive adversary and a full version which is secure against chosen ciphertext attacks. For both versions, the security is contingent on the BDH assumption and the assumption that the hash function H is a random oracle. We describe here the basic version.

Public parameters: A bilinear pairing $e\colon G_1 \times G_2 \to G_T$ between groups of large prime order n, a hash function $H\colon \{0,1\}^* \to G_2$, a base point $P \in G_1$, and a point $\alpha P \in G_1$ where $\alpha \in_R \mathbb{Z}$ is a random integer chosen by the trusted third party. Although the point αP is made public, the integer α is **not** made public.

Key generation: Let $\sigma\{0,1\}^*$ be any binary string, such as an email address. Compute $Q = H(\sigma)$. The public key is σ and the private key is αQ. The owner of the public key (e.g., in this case, the owner of the email address) must obtain the corresponding private key αQ from the trusted third party, since only the trusted third party knows α.

Encryption: Given a public key σ and a message m, let $Q = H(\sigma) \in G_2$. Choose $r \in_R \mathbb{Z}$ at random and compute $c = m \oplus e(\alpha P, rQ)$ where $\oplus$ denotes bitwise exclusive or. The ciphertext is the pair (rP, c).

Note that encryption of messages can be performed even if the key generation step has not yet taken place.

Decryption: Given a ciphertext (c_1, c_2), compute

$$m' = c_2 \oplus e(c_1, \alpha Q)$$

and output m' as the plaintext.

For a valid encryption (c_1, c_2) of m, the decryption process yields

$$c_2 \oplus e(c_1, \alpha Q) = (m \oplus e(\alpha P, rQ)) \oplus e(rP, \alpha Q),$$

which is equal to m since $e(\alpha P, rQ) = e(rP, \alpha Q)$.

Short signatures. Using pairing based cryptography, it is possible to construct digital signature schemes having signature lengths equal to half the length of ECDSA signatures (Protocol 1.4.7), without loss of security. Whereas ECDSA signatures consist of two elements, a short signature scheme such as BLS (described below) can sign messages using only one element, provided that compressed point representations are used (Sect. 1.3.3).

Protocol 1.5.6 (Boneh-Lynn-Shacham (BLS))**.** The Boneh-Lynn-Shacham or BLS protocol [13] was the first short signature scheme to be developed. The security of BLS relies on the random oracle assumption for H and the co-Diffie-Hellman (co-DH) assumption for the bilinear group pair (G_1, G_2). The co-DH assumption states that given $P \in G_1$ and $Q, \alpha Q \in G_2$, it is infeasible to compute αP. When $G_1 = G_2$, the co-DH assumption is equivalent to the standard Diffie-Hellman assumption for G_1.

Public parameters: A bilinear pairing $e\colon G_1 \times G_2 \to G_T$ between groups of large prime order n, a hash function $H\colon \{0,1\}^* \to G_1$, and a base point $Q \in G_2$.

Key generation: Choose a random integer α in the interval $1 \leq \alpha < n$. The public key is αQ and the private key is α.

Signing: The message space is the set of all binary strings. To sign a message m, compute $H(m) \in G_1$ and $\sigma = \alpha H(m)$. The signature of m is σ.

Verification: To verify a signature σ of a message m, compute the two quantities $e(H(m), \alpha Q)$ and $e(\sigma, Q)$. The signature is valid if and only if these two values are equal.

For a legitimate signature σ of m, we have

$$e(H(m), \alpha Q) = e(H(m), Q)^{\alpha} = e(\alpha H(m), Q) = e(\sigma, Q),$$

so the signature does verify correctly.

1.6 Properties of pairings

In this section we list some of the main properties shared by all pairings arising from elliptic curves. Although the properties and limitations listed here are not necessarily direct consequences of the definition of pairing, all existing examples of pairings are constructed in essentially the same way and therefore share all of the attributes described herein.

1.6.1 Embedding degree

We begin with a few general facts about pairings. All known families of pairings are constructed from elliptic curves. Let E be an elliptic curve defined over $\mathbb{F}_q$. Suppose that the group order $\#E$ factors as $\#E = hn$ where n is a large prime and h is an integer (called the *cofactor*). Let G_1 be a subgroup of $E(\mathbb{F}_q)$ of order n. In most cases (namely, when $h \nmid n$), there is only one such subgroup, given by $G_1 = \{hP \mid P \in E(\mathbb{F}_q)\}$. Then, for an appropriate choice of integer k, there exists a pairing $e\colon G_1 \times G_2 \to G_T$, where $G_2 \subset E(\mathbb{F}_{q^k})$ and $G_T \subset \mathbb{F}_{q^k}^*$. When e is Type 1 , the group G_2 can be taken to be a subgroup not only of $E(\mathbb{F}_{q^k})$ but also of $E(\mathbb{F}_q)$.

Every bilinear pairing is a group homomorphism in each coordinate, and the multiplicative group $\mathbb{F}_{q^k}^*$ has order $q^k - 1$. Hence a necessary condition for the existence of a pairing $e\colon G_1 \times G_2 \to G_T$ is that n divides $q^k - 1$. One can show that this condition is also sufficient. These facts motivate the following definition.

Definition 1.6.1. For any elliptic curve $E/\mathbb{F}_q$ and any divisor n of $\#E(\mathbb{F}_q)$, the *embedding degree* of E with respect to n is the smallest integer k such that $n \mid q^k - 1$.

Example 1.6.2 (Type 1 pairing with $k = 2$). Let $p = 76933553304715506523$ and let E be the curve $y^2 = x^3 + x$ defined over $\mathbb{F}_p$. Then E is a *supersingular* curve (Sect. 1.6.3) with cardinality

$$\#E = p + 1 = 76933553304715506523 = 4 \cdot 19233388326178876631,$$

where $h = 4$ is the cofactor and $n = 19233388326178876631$ is prime. The embedding degree is 2, since $\frac{p^2-1}{n} = 307734213218862026088$ is an integer. Points in G_1 can be generated by choosing any random point in $E(\mathbb{F}_p)$ and multiplying it by the cofactor $h = 4$. One example of such a point is

$$P = (19249681072784673607, 27563138688248568100).$$

The modified Weil pairing (Sect. 1.8.1) forms a Type 1 pairing $e\colon G_1 \times G_1 \to G_T$ on G_1, with $G_2 = G_1$, where G_T denotes the unique subgroup of $\mathbb{F}_{p^2}^*$ of order n. Using the point P above, we have

$$e(P, P) = 58219392405889795452 + 671682975778577314\,i$$

where $i = \sqrt{-1}$ is a square root of -1 in $\mathbb{F}_{p^2}$.

Example 1.6.3 (Type 3 pairing with $k = 12$). Let $p = 1647649453$ and $n = 1647609109$. The elliptic curve $E : y^2 = x^3 + 11$ is a Barreto-Naehrig curve (Sect. 1.8.2) of embedding degree 12 and cofactor 1. Let $G_1 = E(\mathbb{F}_p)$ and let G_2 be any subgroup of $E(\mathbb{F}_{p^{12}})$ of order n. If we construct $\mathbb{F}_{p^{12}}$ as $\mathbb{F}_p[w]$ where $w^{12} + 2 = 0$, then the points

$$P = (1107451886, 1253137994) \in E(\mathbb{F}_p)$$
$$Q = (79305390\,w^4 + 268184452\,w^{10}, 311639750\,w^3 + 1463165539\,w^9) \in E(\mathbb{F}_{p^{12}})$$

generate appropriate groups G_1 and G_2. Here the point Q is obtained from a sextic twist [7]. Using the Tate pairing (Sect. 1.7.3), we obtain a Type 3 pairing $e\colon G_1 \times G_2 \to G_T$ where G_T is the unique subgroup of $\mathbb{F}^*_{p^{12}}$ of order n. The value of the Tate pairing at P and Q is

$$\begin{aligned} e(P,Q) = {} & 1285419312 + 881628570\,w + 506836791\,w^2 + 155425783\,w^3 + \\ & 1374794677\,w^4 + 1219941843\,w^5 + 285132062\,w^6 + 1621017742\,w^7 + \\ & 525459081\,w^8 + 1553114915\,w^9 + 1356557676\,w^{10} + 175456091\,w^{11} \end{aligned}$$

where $w^{12} + 2 = 0$ as above.

Example 1.6.4 (Curve with intractably large embedding degree). We must emphasize that, as a consequence of a result of Balasubramanian and Koblitz [5], the overwhelming majority of elliptic curves have extremely large embedding degrees, which render the computation of any bilinear pairings infeasible. In other words, *very few elliptic curves* admit a usable pairing.

For example, consider the Certicom ECCp-109 curve of Example 1.3.1. This curve has order $n = 564538252084441531840258143378149$, which is a prime. The embedding degree of this curve is equal to $n - 1 \approx 2^{109}$. Hence any bilinear pairing on this curve takes values in the field $\mathbb{F}_{p^{n-1}}$. However, the number p^{n-1} is so large that no computer technology now or in the foreseeable future is capable of implementing a field of this size.

1.6.2 MOV reduction

When the embedding degree is small, the existence of a bilinear pairing can be used to transfer discrete logarithms on the elliptic curve to the corresponding discrete logarithm problem in a finite field. In many cases, this reduction negates the increased security of elliptic curves compared to finite fields (Sect. 1.1.2). Of course, this concern only applies to the minority of elliptic curves which admit a bilinear pairing, and oftentimes the extra features provided by pairings outweigh the security concerns. Nonetheless, an understanding of this issue is essential whenever designing or implementing a scheme using pairings.

The reduction algorithm is known as the MOV reduction [38] or the Frey-Rück reduction [25] , and proceeds as follows. Given a bilinear pairing $e\colon G_1 \times G_2 \to G_T$, let P and αP be any pair of points in G_1. (The same reduction algorithm also works for G_2.) Choose any point $Q \in G_2$ and compute the

quantities $g = e(P, Q)$ and $h = e(\alpha P, Q)$. Then, by the bilinearity property, we have $h = g^\alpha$. Hence the discrete logarithm of h in G_T is equal to the discrete logarithm of αP in G_1. Since G_T is a multiplicative subgroup of a finite field, the index calculus algorithm [34] can be used to solve for discrete logarithms in G_T. Depending on the value of the embedding degree, the index calculus algorithm on G_T can be faster than the Pollard rho algorithm [45] on G_1.

Specifically, let $E/\mathbb{F}_q$ be an elliptic curve as in Sect. 1.6.1, with embedding degree k. An instance of the discrete logarithm problem on $G_1 = E(\mathbb{F}_q)$ can be solved *either* directly on G_1, or indirectly via index calculus on $G_T \subset \mathbb{F}^*_{q^k}$. Figure 1.6.2, based on [30], estimates the optimal choice of k for which the index calculus algorithm on $\mathbb{F}^*_{q^k}$ takes the same amount of time as the Pollard rho algorithm [45] on $E(\mathbb{F}_q)$. Although the comparison in [30] is based on integer factorization, the performance of the index calculus algorithm is comparable [34].

Size of $E(\mathbb{F}_q)$	Equivalent finite field size	Optimal embedding degree
110	512	4.5
160	1024	6.5
192	1536	8
256	3072	12

Figure 1.2: Estimates of the optimal embedding degree k for various curve sizes.

Not all applications require choosing an optimal embedding degree. For example, in identity based encryption, faster performance can be obtained by using a curve with a 512-bit q and embedding degree 2. However, bandwidth-sensitive applications such as short signatures require embedding degrees at least as large as the optimal value in order to attain the best possible security.

1.6.3 Overview of pairing families

In this section we give a broad overview of the available families of pairing based curves. Technical details are deferred to Sect.s 1.7 and 1.8.

Elliptic curves over finite fields come in two types: supersingular and ordinary. An elliptic curve $E/\mathbb{F}_{p^m}$ is defined to be *supersingular* if p divides $p^m + 1 - \#E$. All known constructions of Type 1 pairings use supersingular curves [32]. Menezes, Okamoto, and Vanstone [38] have shown that the maximum possible embedding degree of a supersingular elliptic curve is 6. More specifically, over fields of characteristic $p = 2$, $p = 3$, and $p > 3$, the maximum embedding degrees are 4, 6, and 3 respectively. Thus, the maximum achievable embedding degree at present for a Type 1 pairing is 6. Since many protocols, such as tripartite one-round key exchange (Protocol 1.5.4), require a Type 1 pairing, they must be designed and implemented with this limitation in mind.

An ordinary elliptic curve is any elliptic curve which is not supersingular. In

the case of ordinary elliptic curves, the Cocks-Pinch method [16, 24] is capable of producing curves having any desired embedding degree [24]. However, the curves obtained via this method do not have prime order. For prime order elliptic curves, the Barreto-Naehrig family of curves [7], having embedding degree 12, represents the largest embedding degrees available today, although for performance reasons the MNT family of curves [40], having maximum embedding degree 6, is sometimes preferred. Pairings on ordinary curves can be selected to be either Type 2 or Type 3 depending on the choice of which subgroup of the curve is used in the pairing [26].

1.7 Implementations of pairings

This section contains the technical definitions and concepts required in order to construct pairings. We also give proofs of some of the basic properties of pairings, along with concrete algorithms for implementing the standard pairings. An alternative approach, for readers who wish to skip the technical details, is to use a pre-existing implementation, such as Ben Lynn's `pbc` library [37], which is published under the GNU General Public License.

1.7.1 Divisors

All known examples of cryptographic pairings rely in an essential way on the notion of a *divisor* on an elliptic curve. In this section we give a brief self-contained treatment of the basic facts about divisors. We will then use this theory to give examples of cryptographic pairings and describe how they can be efficiently computed.

Recall that every nonzero integer (more generally, every rational number) admits a unique factorization into a product of prime numbers. For example,

$$\begin{aligned} 6 &= 2 \cdot 3 & 7/4 &= 7^1 \cdot 2^{-2} \\ 50 &= 2 \cdot 5^2 & 1 &= \emptyset \end{aligned}$$

or, in additive notation,

$$\begin{aligned} \log(6) &= \log(2) + \log(3) & \log(7/4) &= \log(7) + (-2)\log(2) \\ \log(50) &= \log(2) + 2\log(5) & \log(1) &= 0 \end{aligned}$$

Observe that prime factorizations satisfy the following properties:

1. The sum is finite,
2. The coefficient of each prime is an integer,
3. The sum is unique: no two sums are equal unless all the coefficients are equal.

These properties motivate the definition of divisor on an elliptic curve:

Definition 1.7.1. A *divisor* on an elliptic curve E is a formal sum $\sum_{P \in E} a_P(P)$ of points P on the curve such that:

1. The sum is finite,
2. The coefficient a_P of each point P is an integer,
3. The sum is unique: no two sums are equal unless all the coefficients are equal.

The *degree* of a divisor $D = \sum_{P \in E} a_P(P)$, denoted $\deg(D)$, is the integer given by the finite sum $\sum_{P \in E} a_P$.
The empty divisor is denoted $\emptyset$, and its degree by definition is 0.

Definition 1.7.2. Let $E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$ be an elliptic curve defined over a field F. A *rational function* on E is a function $f \colon E \to F$ of the form

$$f(x, y) = \frac{f_1(x, y)}{f_2(x, y)}$$

where $f_1(x, y)$ and $f_2(x, y)$ are polynomials in the two variables x and y.

Definition 1.7.3. Let $f(x, y) = \frac{f_1(x,y)}{f_2(x,y)}$ be a nonzero rational function on an elliptic curve E. For any point $P \in E$, the *order* of f at P, denoted $\mathrm{ord}_P(f)$, is defined as follows:

- If $f(P) \neq 0$ and $\frac{1}{f(P)} \neq 0$, then $\mathrm{ord}_P(f) = 0$.
- If $f(P) = 0$, then $\mathrm{ord}_P(f)$ equals the multiplicity of the root at P of the numerator $f_1(x, y)$.
- If $\frac{1}{f(P)} = 0$, then $\mathrm{ord}_P(f)$ equals the negative of the multiplicity of the root at P of the denominator $f_2(x, y)$.

Definition 1.7.4. Let f be a nonzero rational function on an elliptic curve E. The *principal divisor generated by* f, denoted $\mathrm{div}(f)$, is the divisor

$$\mathrm{div}(f) := \sum_{P \in \mathcal{E}} \mathrm{ord}_P(f) \cdot (P),$$

which represents the (finite) sum over all the points $P \in E$ at which either the numerator or the denominator of f is equal to zero.

A divisor D on E is called a *principal divisor* if $D = \mathrm{div}(f)$ for some rational function f on E.

Note that $\mathrm{div}(fg) = \mathrm{div}(f) + \mathrm{div}(g)$, and $\mathrm{div}(1) = \emptyset$. Hence div is a homomorphism from the multiplicative group of nonzero rational functions on E to the additive group of divisors on E. Accordingly, the image of div is a subgroup of the group of divisors.

Theorem 1.7.5. *For any rational function f on E, we have* $\deg(\mathrm{div}(f)) = 0$.

Proof. [49, II.3]. □

Example 1.7.6. Let E be the elliptic curve $y^2 = x^3 - x$. Let f be the rational function $f(x, y) = \frac{x}{y}$. We can calculate $\mathrm{div}(f)$ as follows. The numerator $f_1(x, y) = x$ is zero at the point $P = (0, 0)$, and $1/f_1 = 1/x$ is zero at $P = \infty$. Since the line $x = 0$ is tangent to the curve E at $(0, 0)$, we know that $\mathrm{ord}_{(0,0)}(f_1) = 2$. By Theorem 1.7.5, we must also have that $\mathrm{ord}_\infty(f_1) = -2$. Hence the principal divisor generated by x is

$$\mathrm{div}(x) = 2((0, 0)) - 2(\infty).$$

A similar calculation yields

$$\mathrm{div}(y) = ((0, 0)) + ((0, 1)) + ((0, -1)) - 3(\infty)$$

and hence

$$\mathrm{div}\,(f) = \mathrm{div}(x) - \mathrm{div}(y) = ((0, 0)) - ((0, 1)) - ((0, -1)) + (\infty).$$

Definition 1.7.7. Two divisors D_1 and D_2 are *linearly equivalent* (denoted by $D_1 \sim D_2$) if there exists a nonzero rational function f such that

$$D_1 - D_2 = \mathrm{div}(f).$$

The relation of linear equivalence between divisors is an equivalence relation. Note that, by Theorem 1.7.5, a necessary condition for two divisors to be equivalent is that they have the same degree.

Lemma 1.7.8. *For any two points $P, Q \in E$,*

$$(P) - (\infty) + (Q) - (\infty) \sim (P + Q) - (\infty),$$

where the addition sign on the right hand side denotes geometric addition.

Proof. If either $P = \infty$ or $Q = \infty$, then the two sides are equal, and hence necessarily equivalent. Suppose now that $P + Q = \infty$. Let $x - d = 0$ be the vertical line passing through P and Q. Then, by a calculation similar to that in Example 1.7.6, we find that

$$\mathrm{div}(x - d) = (P) + (Q) - 2(\infty)$$

so $(P) + (Q) - 2(\infty) \sim \emptyset = (\infty) - (\infty) = (P + Q) - (\infty)$, as desired.

The only remaining case is where P and Q are two points satisfying $P \neq \infty$, $Q \neq \infty$, and $P \neq -Q$. In this case, let $ax + by + c = 0$ be the equation of the line passing through the points P and Q, and let $x - d = 0$ be the equation of the vertical line passing through $P + Q$. These two lines intersect at a common point R lying on the elliptic curve.

We have

$$\begin{aligned}
\operatorname{div}(ax+by+c) &= (P)+(Q)+(R)-3(\infty),\\
\operatorname{div}(x-d) &= (R)+(P+Q)-2(\infty),\\
\operatorname{div}\left(\frac{ax+by+c}{x-d}\right) &= (P)+(Q)-(P+Q)-(\infty)\\
&= (P)-(\infty)+(Q)-(\infty)-[(P+Q)-(\infty)],
\end{aligned}$$

implying that $(P)-(\infty)+(Q)-(\infty)-[(P+Q)-(\infty)]$ is a principal divisor, as required. □

Remark 1.7.9. It is not possible for $(P)+(Q)$ to be equivalent to $(P+Q)$, since the first divisor has degree 2 and the second divisor has degree 1. Lemma 1.7.8 says that, after correcting for this discrepancy by adding ∞ terms, the divisors become equivalent.

Proposition 1.7.10. *Let $D=\sum a_P(P)$ be any degree zero divisor on E. Then*

$$D \sim \left(\sum a_P P\right) - (\infty),$$

where the interior sum denotes elliptic curve point addition.

Proof. Since D has degree zero, the equation

$$D = \sum_{P\in E} a_P[(P)-(\infty)]$$

holds. Now apply Lemma 1.7.8 repeatedly. □

The converse of Proposition 1.7.10 also holds, and its proof follows from a well known result known as the Riemann-Roch theorem.

Proposition 1.7.11. *Let $D_1=\sum a_P(P)$ and $D_2=\sum b_P(P)$ be two degree zero divisors on E. Then $D_1\sim D_2$ if and only if*

$$\sum_{P\in E} a_P P = \sum_{P\in E} b_P P.$$

Proof. [49, III.3.4]. □

1.7.2 Weil pairing

The Weil pairing was historically the first example of a cryptographic pairing to appear in the literature. In this section we define the Weil pairing and prove some of its basic properties.

Definition 1.7.12. Let E/F be an elliptic curve and let $n > 0$ be an integer. The set of *n-torsion points* of E, denoted $E[n]$, is given by

$$E[n] := \left\{ P \in E(\bar{F}) \mid nP = \infty \right\},$$

where $\bar{F}$ denotes the algebraic closure of F. The set $E[n]$ is always a subgroup of $E(\bar{F})$.

Remark 1.7.13. If the characteristic of F does not divide n, then the group $E[n]$ is isomorphic as a group to $\mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}$.

Definition 1.7.14. Let f be a rational function and let $D = \sum a_P(P)$ be a degree zero divisor on E. The value of f at D, denoted $f(D)$, is the element

$$f(D) := \prod f(P)^{a_P} \in F.$$

Definition 1.7.15. Let E be an elliptic curve over F. Fix an integer $n > 0$ such that $\mathrm{char}(F) \nmid n$ and $E[n] \subset E(F)$. For any two points $P, Q \in E[n]$, let A_P be any divisor linearly equivalent to $(P) - (\infty)$ (and similarly for A_Q). By Proposition 1.7.10, the divisor nA_P is linearly equivalent to $(nP) - (n\infty) = \emptyset$. Hence nA_P is a principal divisor. Let f_P be any rational function having divisor equal to nA_P (and similarly for f_Q).

The *Weil pairing* of P and Q is given by the formula

$$e(P, Q) = \frac{f_P(A_Q)}{f_Q(A_P)},$$

valid whenever the expression is defined (i.e., neither the numerator nor the denominator nor the overall fraction involves a division by zero).

Proposition 1.7.16. *The Weil pairing is well defined for any pair of points $P, Q \in E[n]$.*

Proof. The definition of Weil pairing involves a choice of divisors A_P, A_Q and a choice of rational functions f_P, f_Q. In order to prove the proposition, we need to show that for any two points P, Q there exists a choice such that $e(P, Q)$ is defined, and that any other set of choices for which $e(P, Q)$ is defined leads to the same value.

We will begin by proving the second part. To start with, the choice of f_P does not affect the value of $e(P, Q)$, since for any other function $\hat{f}_P$ sharing the same divisor, we have

$$\mathrm{div}(\hat{f}_P / f_P) = \emptyset,$$

which means $\hat{f}_P = c f_P$ for some nonzero constant $c \in F$. It follows then that $\hat{f}_P(A_Q) = f_P(A_Q)$, since A_Q has degree zero, and therefore the factors of c cancel out in the formula of Definition 1.7.14.

We now prove that the choice of A_P does not affect the value of $e(P, Q)$; the proof for A_Q is similar. If $\hat{A}_P$ is another divisor linearly equivalent to A_P, then

$\hat{A}_P = A_P + \mathrm{div}(g)$ for some rational function g. It follows that $\hat{f}_P := f_P \cdot g^n$ is a rational function whose divisor is equal to $n\hat{A}_P$. The value of $e(P,Q)$ under this choice of divisor is equal to

$$\begin{aligned}\hat{e}(P,Q) &= \frac{\hat{f}_P(A_Q)}{f_Q(\hat{A}_P)} = \frac{f_P(A_Q)g(A_Q)^n}{f_Q(A_P)f_Q(\mathrm{div}(g))} \\ &= \frac{f_P(A_Q)}{f_Q(A_P)}\,\frac{g(nA_Q)}{f_Q(\mathrm{div}(g))} = e(P,Q)\,\frac{g(\mathrm{div}(f_Q))}{f_Q(\mathrm{div}(g))}.\end{aligned}$$

The fraction $\frac{g(\mathrm{div}(f_Q))}{f_Q(\mathrm{div}(g))}$ is equal to one by the *Weil reciprocity formula*, which we will not prove here. A proof of Weil reciprocity can be found in [11, 39].

To complete the proof, we need to show that there exists a choice of divisors A_P and A_Q for which the calculation of $e(P,Q)$ does not involve division by zero. The naive choice of $A_P = (P) - (\infty)$, $A_Q = (Q) - (\infty)$ does not work whenever $Q \neq \infty$, because in this case $\mathrm{div}(f_Q) = n(Q) - n(\infty)$, so $1/f_Q$ equals zero at ∞, and consequently

$$f_Q(A_P) = \frac{f_Q(P)}{f_Q(\infty)} = 0.$$

To fix this problem, let R be any point in $E(\bar{F})$ not equal to any of the four points Q, ∞, $-P$, $Q - P$. Here $\bar{F}$ denotes the algebraic closure of F, over which E has infinitely many points, guaranteeing that such an R exists. Set $A_P = (P + R) - (R)$. Then A_P is linearly equivalent to $(P) - (\infty)$, and

$$f_Q(A_P) = \frac{f_Q(P+R)}{f_Q(R)} \in F^*,$$

since $\mathrm{div}(f_Q) = n(Q) - n(\infty)$, and we have chosen R in such a way that neither R nor $P + R$ coincides with either Q or ∞. Similarly, we find that

$$f_P(A_Q) = \frac{f_P(Q)}{f_P(\infty)} \in F^*,$$

because $\mathrm{div}(f_P) = n(P + R) - n(R)$, and neither Q nor ∞ coincides with R or $P + R$. □

Theorem 1.7.17. *The Weil pairing satisfies the following properties.*

- $e(P_1 + P_2, Q) = e(P_1, Q)\ e(P_2, Q)$ *and* $e(P, Q_1 + Q_2) = e(P, Q_1)\ e(P, Q_2)$ *(bilinearity)*
- $e(aP, Q) = e(P, aQ) = e(P, Q)^a$*, for all* $a \in \mathbb{Z}$
- $e(P, \infty) = e(\infty, Q) = 1$
- $e(P, Q)^n = 1$

- $e(P,Q) = e(Q,P)^{-1}$ *and* $e(P,P) = 1$ *(anti-symmetry)*
- *If* $P \neq \infty$ *and* F *is algebraically closed, there exists* $Q \in E$ *such that* $e(P,Q) \neq 1$ *(non-degeneracy)*

Proof. We begin with bilinearity. Suppose $P_1, P_2, Q \in E[n]$. Observe that

$$A_{P_1+P_2} \sim (P_1 + P_2) - (\infty) \sim (P_1) - (\infty) + (P_2) - (\infty) \sim A_{P_1} + A_{P_2}$$

by Lemma 1.7.8. Hence we may use $A_{P_1} + A_{P_2}$ as our choice of $A_{P_1+P_2}$. Moreover, if f_{P_1} and f_{P_2} are rational functions having divisor nA_{P_1} and nA_{P_2} respectively, then

$$\operatorname{div}(f_{P_1} f_{P_2}) = \operatorname{div}(f_{P_1}) + \operatorname{div}(f_{P_2}) = nA_{P_1} + nA_{P_2} = nA_{P_1+P_2}.$$

Accordingly, we may take $f_{P_1+P_2}$ to be equal to $f_{P_1} f_{P_2}$. Therefore,

$$\begin{aligned} e(P_1 + P_2, Q) &= \frac{f_{P_1+P_2}(A_Q)}{f_Q(A_{P_1+P_2})} = \frac{(f_{P_1} f_{P_2})(A_Q)}{f_Q(A_{P_1} + A_{P_2})} \\ &= \frac{f_{P_1}(A_Q) f_{P_2}(A_Q)}{f_Q(A_{P_1}) f_Q(A_{P_2})} = e(P_1, Q)\ e(P_2, Q), \end{aligned}$$

as desired. The proof that $e(P, Q_1 + Q_2) = e(P, Q_1)\ e(P, Q_2)$ is similar.

The property $e(aP, Q) = e(P, aQ) = e(P, Q)^a$ follows from bilinearity, and $e(P, \infty) = e(\infty, Q) = 1$ is a consequence of the definition of the Weil pairing. These two facts together imply that $e(P,Q)^n = e(nP, Q) = e(\infty, Q) = 1$.

Anti-symmetry follows from the definition of the Weil pairing, since

$$e(P,Q) = \frac{f_P(A_Q)}{f_Q(A_P)} = \left(\frac{f_Q(A_P)}{f_P(A_Q)}\right)^{-1} = e(Q,P)^{-1}.$$

We will not prove non-degeneracy, since it can be easily verified in practice via computation. A proof of non-degeneracy can be found in [11]. □

1.7.3 Tate pairing

The Tate pairing is a non-degenerate bilinear pairing which shares much in common with the Weil pairing. It is generally preferred over the Weil pairing in most implementations of cryptographic protocols, because it can be computed more efficiently.

Definition 1.7.18. Let E be an elliptic curve over a field F. Fix an integer $n > 0$ for which $\operatorname{char}(F) \nmid n$ and $E[n] \subset E(F)$. For any two points $P, Q \in E[n]$, the *Tate proto-pairing* of P and Q, denoted $\langle P, Q \rangle$, is given by the formula

$$\langle P, Q \rangle := f_P(A_Q) \in F^*/F^{*n},$$

valid whenever the expression $f_P(A_Q)$ is defined and nonzero.

Proposition 1.7.19. *The value of the Tate proto-pairing is well defined, independent of the choices of A_P, A_Q, and f_P.*

Proof. As in the case of the Weil pairing, the choice of f_P is irrelevant once A_P is fixed. We may thus take $A_P = (P) - (\infty)$ and $A_Q = (Q + R) - (R)$ where $R \neq P, \infty, -Q, P - Q$. For this choice of A_P and A_Q, the expression $f_P(A_Q)$ will be a nonzero element of F.

We now show that $\langle P, Q\rangle$ takes on the same value independent of the choice of A_P and A_Q. If a different value of A_Q is chosen, say $\hat{A}_Q = A_Q + \mathrm{div}(g)$, then, using Weil reciprocity, we find that

$$\begin{aligned}\widehat{\langle P, Q\rangle} &= f_P(\hat{A}_Q) = f_P(A_Q)\ f_P(\mathrm{div}(g)) = f_P(A_Q)\ g(\mathrm{div}(f_P)) \\ &= f_P(A_Q)\ g(nA_P) = f_P(A_Q)\ g(A_P)^n.\end{aligned}$$

The latter value is equal to $\langle P, Q\rangle = f_P(A_Q)$ in the quotient group F^*/F^{*n}. Likewise, if a different divisor $\hat{A}_P = A_P + \mathrm{div}(g)$ is used, then $n\hat{A}_P = \mathrm{div}(f_P \cdot g^n)$, so

$$\widehat{\langle P, Q\rangle} = \hat{f}_P(A_Q) = f_P(A_Q)g(A_Q)^n \equiv f_P(A_Q) \pmod{F^{*n}}.$$

□

Theorem 1.7.20. *The Tate proto-pairing satisfies the following properties.*

- $\langle P_1 + P_2, Q\rangle = \langle P_1, Q\rangle\ \langle P_2, Q\rangle$ *and* $\langle P, Q_1 + Q_2\rangle = \langle P, Q_1\rangle\ \langle P, Q_2\rangle$ *(bilinearity)*
- $\langle aP, Q\rangle = \langle P, aQ\rangle = \langle P, Q\rangle^a$ *for all* $a \in \mathbb{Z}$
- $\langle P, \infty\rangle = \langle \infty, Q\rangle = 1$
- $\langle P, Q\rangle^n = 1$
- *If $P \neq \infty$, and F is algebraically closed, there exists $Q \in E[n]$ such that $\langle P, Q\rangle \neq 1$ (non-degeneracy)*

Note that the Tate proto-pairing is *not* anti-symmetric.

Proof. As in the case of the Weil pairing, we may take $A_{Q_1+Q_2}$ to be $A_{Q_1} + A_{Q_2}$, and $f_{P_1+P_2}$ to be $f_{P_1} f_{P_2}$. In this case,

$$\begin{aligned}\langle P_1 + P_2, Q\rangle &= f_{P_1+P_2}(A_Q) = f_{P_1}(A_Q)\ f_{P_2}(A_Q) = \langle P_1, Q\rangle\ \langle P_2, Q\rangle, \\ \langle P, Q_1 + Q_2\rangle &= f_P(A_{Q_1} + A_{Q_2}) = f_P(A_{Q_1})\ f_P(A_{Q_2}) = \langle P, Q_1\rangle\ \langle P, Q_2\rangle.\end{aligned}$$

All of the other properties (except for non-degeneracy) follow from bilinearity and the definition of the pairing. We will not prove non-degeneracy (see [11] for a proof). □

The Tate pairing is obtained from the Tate proto-pairing by raising the value of the proto-pairing to an appropriate power. The Tate pairing is only defined for elliptic curves over finite fields.

Definition 1.7.21. Let $F = \mathbb{F}_{q^k}$ be a finite field. Let n be an integer dividing $q^k - 1$, and fix two points $P, Q \in E[n]$. The *Tate pairing* $e(P, Q)$ of P and Q is the value

$$e(P, Q) = \langle P, Q \rangle^{\frac{q^k-1}{n}} \in \mathbb{F}_{q^k}^*.$$

Theorem 1.7.22. *The Tate pairing satisfies all the properties listed in Theorem 1.7.20.*

Proof. Exponentiation by $\frac{q^k-1}{n}$ is an isomorphism from $\mathbb{F}_{q^k}^*/\mathbb{F}_{q^k}^{*\,n}$ to $\left(\mathbb{F}_{q^k}^*\right)^{\frac{q^k-1}{n}}$, so all of the properties in Theorem 1.7.20 hold for the Tate pairing. □

1.7.4 Miller's algorithm

The calculation of Weil and Tate pairings on the subgroup of n-torsion points $E[n]$ of an elliptic curve E can be performed in a number of field operations polynomial in $\log(n)$, thanks to the following algorithm of Miller [39], which we present here.

Fix a triple of n-torsion points $P, Q, R \in E[n]$. We assume for simplicity that n is large, since this is the most interesting case from an implementation standpoint. For each integer m between 1 and n, let f_m denote a rational function whose divisor has the form

$$\operatorname{div}(f_m) = m(P + R) - m(R) - (mP) + (\infty).$$

We will first demonstrate an algorithm for calculating $f_n(Q)$, and then show how we can use this algorithm to find $e(P, Q)$.

For any two points $P_1, P_2 \in E[n]$, let $g_{P_1,P_2}(x, y) = ax + by + c$ be the equation of the line passing through the two points P_1 and P_2. In the event that $P_1 = P_2$, we set $g_{P_1,P_2}(x, y)$ to be equal to the tangent line at P_1. If either P_1 or P_2 is equal to ∞, then g_{P_1,P_2} is the equation of the vertical line passing through the other point; and finally, if $P_1 = P_2 = \infty$, then we define $g_{P_1,P_2} = 1$. In all cases,

$$\operatorname{div}(g_{P_1,P_2}) = (P_1) + (P_2) + (-P_1 - P_2) - 3(\infty).$$

To calculate $f_m(Q)$ for $m = 1, 2, \ldots, n$, we proceed by induction on m. If $m = 1$, then the function

$$f_1(x, y) = \frac{g_{P+R,-P-R}(x, y)}{g_{P,R}(x, y)}$$

has divisor equal to $(P + R) - (R) - (P) + (\infty)$. We can evaluate this function at Q to obtain $f_1(Q)$.

For values of m greater than 1, we consider separately the cases of m even and m odd. If m is even, say $m = 2k$, then

$$f_m(Q) = f_k(Q)^2 \cdot \frac{g_{kP,kP}(Q)}{g_{mP,-mP}(Q)},$$

while if m is odd we have

$$f_m(Q) = f_{m-1}(Q) \cdot f_1(Q) \cdot \frac{g_{(m-1)P,P}(Q)}{g_{mP,-mP}(Q)}.$$

Note that every two steps in the induction process reduces the value of m by a factor of 2 or more. This feature is the reason why this method succeeds in calculating $f_n(Q)$ even for very large values of n.

The Tate pairing of two n-torsion points $P, Q \in E[n]$ can now be calculated as follows. Choose two random points $R, R' \in E[n]$. Set $A_P = (P+R)-(R)$ and $A_Q = (Q + R') - (R')$. Using the method above, find the values of $f_n(Q + R')$ and $f_n(R')$. Since $\mathrm{div}(f_n) = n(P + R) - n(R) - (nP) + (\infty) = nA_p = \mathrm{div}(f_P)$, we find that

$$\frac{f_n(Q + R')}{f_n(R')} = \frac{f_P(Q + R')}{f_P(R')} = f_P(A_Q).$$

It is now easy to calculate the Tate pairing $e(P, Q) = f_P(A_Q)^{\frac{q^k-1}{n}}$. To find the Weil pairing, simply repeat the procedure in order to find $f_Q(A_P)$, and divide it into $f_P(A_Q)$. As long as the integer n is sufficiently large, it is unlikely that the execution of this algorithm will yield a division by zero error. On the rare occasion when such an obstacle does arise, repeat the calculation using a different choice of random points R and R'. A description of Miller's algorithm in pseudocode can be found in Figure 1.3.

Note that the Tate pairing consists of only one divisor evaluation whereas the Weil pairing requires two. Since divisor evaluation is the most time consuming step in pairing computation, the Tate pairing is superior to the Weil pairing in terms of performance. In certain special cases, alternative pairings are available which are even faster [6, 29].

1.8 Pairing-friendly curves

As remarked in Sect. 1.6, low embedding degrees are necessary in order to construct pairings, and very few elliptic curves have low embedding degrees. In this section, we describe some families of elliptic curves having low embedding degree. Such curves are often called *pairing-friendly curves*.

1.8.1 Supersingular curves

Recall that a supersingular curve is an elliptic curve $E/\mathbb{F}_{p^m}$ such that p divides $p^m + 1 - \#E$. All supersingular elliptic curves have embedding degree $k \leq 6$ and hence are pairing-friendly. For any supersingular curve, the Weil or Tate pairing represents a cryptographic pairing on $E[n]$ where n is any prime divisor of $\#E$. Moreover, a type 1 pairing $\hat{e}$ can be obtained on E using the formula

$$\hat{e}(P, Q) = e(P, \psi(Q))$$

where $e\colon E[n] \times E[n] \to \mathbb{F}^*_{p^{mk}}$ is the usual Weil (or Tate) pairing, $P, Q \in E(\mathbb{F}_{p^m})$ and $\psi\colon E(\mathbb{F}_{p^m}) \to E(\mathbb{F}_{p^{mk}})$ is an algebraic map. Such a map ψ is called a

Algorithm 1.7.1 Computing $g(E, P_1, P_2, Q) = g_{P_1,P_2}(Q)$

Given: $E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$
Given: $P_1 = (x_1, y_1), P_2 = (x_2, y_2), Q = (x_Q, y_Q)$
  if $P_1 = \infty$ and $P_2 = \infty$ **then**
    output 1
  else if $P_1 = \infty$ **then**
    output $x_Q - x_2$
  else if $P_2 = \infty$ **then**
    output $x_Q - x_1$
  else if $P_1 = P_2$ **then**
    output $(3x_1^2 + 2a_2x_1 + a_4 - a_1y_1)(x_Q - x_1) - (2y_1 + a_1x_1 + a_3)(y_Q - y_1)$
  else
    output $(x_Q - x_1)(y_2 - y_1) + (y_Q - y_1)(x_1 - x_2)$
  end if

Algorithm 1.7.2 Computing $f(E, P_1, P_2, Q, m) = f_m(Q)$

Given: E, P_1, P_2, Q, m
  if $m = 1$ **then**
    output $\frac{g(E,P_1+P_2,-P_1-P_2,Q)}{g(E,P_1,P_2,Q)}$
  else if m is even **then**
    $k \leftarrow \frac{m}{2}$
    output $f(E, P_1, P_2, Q, k)^2 \frac{g(E,kP_1,kP_1,Q)}{g(E,mP_1,-mP_1,Q)}$
  else if m is odd **then**
    output $f(E, P_1, P_2, Q, m-1)f(E, P_1, P_2, Q, 1)\frac{g(E,(m-1)P_1,P_1,Q)}{g(E,mP_1,-mP_1,Q)}$
  end if

Algorithm 1.7.3 Computing the Weil pairing $e(P, Q)$ for $P, Q \in E[n]$

Given: E, P, Q, n
  $R_1 \leftarrow_R E[n]$
  $R_2 \leftarrow_R E[n]$
  return $\frac{f(E,P_1,R_1,Q+R_2,n)f(E,Q,R_2,R_1,n)}{f(E,P_1,R_1,R_2,n)f(E,Q,R_2,P_1+R_1,n)}$

Algorithm 1.7.4 Computing the Tate pairing $e(P, Q)$ for $P, Q \in E[n]$

Given: $E/\mathbb{F}_q, P, Q, n$
  $k \leftarrow$ embedding degree of E with respect to n
  $R_1 \leftarrow_R E[n]$
  $R_2 \leftarrow_R E[n]$
  return $\left(\frac{f(E,P_1,R_1,Q+R_2,n)}{f(E,P_1,R_1,R_2,n)}\right)^{\frac{q^k-1}{n}}$

Figure 1.3: Computing the Weil and Tate pairings

Field	Elliptic curve	Distortion map	Group order	emb. deg.
$\mathbb{F}_p$	$y^2 = x^3 + ax$ $p \equiv 3 \bmod 4$	$(x, y) \mapsto (-x, iy)$ $i^2 = -1$	$p + 1$	2
$\mathbb{F}_p$	$y^2 = x^3 + b$ $p \equiv 2 \bmod 3$	$(x, y) \mapsto (\zeta x, y)$ $\zeta^3 = 1$	$p + 1$	2
$\mathbb{F}_{p^2}$	$y^2 = x^3 + b$ $p \equiv 2 \bmod 3$ $b \notin \mathbb{F}_p$	$(x, y) \mapsto$ $\left(\frac{wx^p}{r^{(2p-1)/3}}, \frac{y^p}{r^{p-1}}\right)$ $r^2 = b, r \in \mathbb{F}_{p^2}$ $w^3 = r, w \in \mathbb{F}_{p^6}$	$p^2 - p + 1$	3
$\mathbb{F}_{2^m}$	$y^2 + y = x^3 + x$ or $y^2 + y = x^3 + x + 1$ $m \equiv 1 \bmod 2$	$(x, y) \mapsto$ $(x + s^2, y + sx + t)$ $s, t \in \mathbb{F}_{2^{4m}}, s^4 = s$ $t^2 + t = s^6 + s^2$	$2^m \pm 2^{\frac{m+1}{2}} + 1$	4
$\mathbb{F}_{3^m}$	$y^2 = x^3 + 2x \pm 1$ $m \equiv \pm 1 \bmod 12$	$(x, y) \mapsto (-x + r, uy)$ $u^2 = -1, u \in \mathbb{F}_{3^{2m}}$ $r^3 + 2r \pm 2 = 0$ $r \in \mathbb{F}_{3^{3m}}$	$3^m \pm 3^{\frac{m+1}{2}} + 1$	6
$\mathbb{F}_{3^m}$	$y^2 = x^3 + 2x \pm 1$ $m \equiv \pm 5 \bmod 12$	$(x, y) \mapsto (-x + r, uy)$ $u^2 = -1, u \in \mathbb{F}_{3^{2m}}$ $r^3 + 2r \pm 2 = 0$ $r \in \mathbb{F}_{3^{3m}}$	$3^m \mp 3^{\frac{m+1}{2}} + 1$	6

Figure 1.4: Supersingular curves, distortion maps, and embedding degrees

distortion map, and the corresponding pairing $\hat{e}$ above is known as the *modified* Weil (or Tate) pairing. All known families of Type 1 pairings arise from this construction, and Verheul [50] has shown that distortion maps do not exist on ordinary elliptic curves of embedding degree $k > 1$. Hence at present all known families of Type 1 pairings require the use of supersingular curves.

Figure 1.4 (an extended version of Fig. 1 in [32]) lists all the major families of supersingular elliptic curves together with their corresponding distortion maps and embedding degrees.

1.8.2 Ordinary curves

Certain applications such as short signatures require pairing-friendly elliptic curves of embedding degree larger than 6. In this section we describe two such constructions, the Barreto-Naehrig construction and the Cocks-Pinch method. Both techniques are capable of producing elliptic curves with embedding degree greater than 6. The Cocks-Pinch method produces elliptic curves of arbitrary embedding degree, but not of prime order. The Barreto-Naehrig construction, on the other hand, produces curves of embedding degree 12 and prime order.

Barreto-Naehrig curves. The Barreto-Naehrig family of elliptic curves [7] achieves embedding degree 12 while retaining the property of having prime order. This embedding degree is currently the largest available for prime order pairing-friendly elliptic curves.

Let $N(x)$ and $P(x)$ denote the polynomials

$$N(x) = 36x^4 + 36x^3 + 18x^2 + 6x + 1$$
$$P(x) = 36x^4 + 36x^3 + 24x^2 + 6x + 1$$

and choose a value of x for which both $n = N(x)$ and $p = P(x)$ are prime. (For example, the choice $x = 82$ yields the curve in Example 1.6.3.) Search for a value $b \in \mathbb{F}_p$ for which $b + 1$ is a quadratic residue (i.e. has a square root) in $\mathbb{F}_p$, and the point $Q = (1, \sqrt{b+1})$ on the elliptic curve $E : y^2 = x^3 + b$ satisfies $nQ = \infty$. The search procedure can be as simple as starting from $b = 1$ and incrementing b until a suitable value is found. For such a value b, the curve $E/\mathbb{F}_p$ given by the equation $y^2 = x^3 + b$ has n points and embedding degree 12, and the point $Q = (1, \sqrt{b+1})$ can be taken as a base point.

Cocks-Pinch method. The Cocks-Pinch method [16, 24] produces ordinary elliptic curves having arbitrary embedding degree. The disadvantage of this method is that it cannot produce curves of prime order.

Fix an embedding degree $k > 0$ and an integer $D < 0$. These integers need to be small; typically, one chooses $k < 50$ and $D < 10^7$. The method proceeds as follows.

1. Let n be a prime such that k divides $n - 1$ and D is a quadratic residue modulo n.
2. Let ζ be a primitive k-th root of unity in $\mathbb{F}_n^*$. Such a ζ exists because k divides $n - 1$.
3. Let $t = \zeta + 1 \pmod{n}$.
4. Let $y = \frac{t-2}{\sqrt{D}} \pmod{n}$.
5. Let $p = (t^2 - Dy^2)/4$.

If p is an integer and prime, then a specialized algorithm known as the *complex multiplication method* will produce an elliptic curve defined over $\mathbb{F}_p$ having embedding degree k with n points. The complex multiplication method requires a discriminant as part of its input, and in this case the value of the discriminant is the quantity D. Since the running time of the complex multiplication method is roughly cubic in D, it is important to keep the value of D small. The resulting elliptic curve will not have prime order, although for certain values of k there are various optimizations which produce curves of nearly prime order [24], for which the cofactor is relatively small.

A detailed discussion of the complex multiplication method is not possible within the scope of this work. Annex E of the ANSI X9.62 and X9.63

standards [2, 3] contains a complete implementation-level specification of the algorithm.

1.9 References and further reading

For elliptic curve cryptography and pairing-based cryptography, the most comprehensive sources of mathematical and background information are the two volumes of Blake, Seroussi, and Smart [10, 11] and the *Handbook of Elliptic and Hyperelliptic Curve Cryptography*, edited by Cohen and Frey [17]. Implementation topics are covered in [17] and in the *Guide to Elliptic Curve Cryptography* by Hankerson, Menezes, and Vanstone [28]. The latter work also contains a detailed treatment of elliptic curve-based cryptographic protocols.

Bibliography

[1] Michel Abdalla, Mihir Bellare, and Phillip Rogaway, *The oracle Diffie-Hellman assumptions and an analysis of DHIES*, Topics in cryptology—CT-RSA 2001 (San Francisco, CA), Lecture Notes in Comput. Sci., vol. 2020, Springer, Berlin, 2001, pp. 143–158.

[2] ANSI Standards Committee X9, *Public key cryptography for the financial services industry: Key agreement and key transport using elliptic curve cryptography*. ANSI X9.63-2001.

[3] ———, *Public key cryptography for the financial services industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*. ANSI X9.62-2005.

[4] Adrian Antipa, Daniel Brown, Alfred Menezes, René Struik, and Scott Vanstone, *Validation of elliptic curve public keys*, Public key cryptography—PKC 2003, Lecture Notes in Comput. Sci., vol. 2567, Springer, Berlin, 2002, pp. 211–223.

[5] 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** (1998), no. 2, 141–145.

[6] Paulo S. L. M. Barreto, Steven D. Galbraith, Colm Ó' hÉigeartaigh, and Michael Scott, *Efficient pairing computation on supersingular abelian varieties*, Des. Codes Cryptogr. **42** (2007), no. 3, 239–271.

[7] Paulo S. L. M. Barreto and Michael Naehrig, *Pairing-friendly elliptic curves of prime order*, Selected areas in cryptography, Lecture Notes in Comput. Sci., vol. 3897, Springer, Berlin, 2006, pp. 319–331.

[8] Daniel J. Bernstein and Tanja Lange, *Faster addition and doubling on elliptic curves*, Advances in cryptology—ASIACRYPT 2007 (Kuching, Malaysia), Lecture Notes in Comput. Sci., vol. 4833, Springer, Berlin, 2007, pp. 29–50.

[9] Ian F. Blake and Theodoulos Garefalakis, *On the security of the digital signature algorithm*, Des. Codes Cryptogr. **26** (2002), no. 1-3, 87–96. In honour of Ronald C. Mullin.

[10] Ian F. Blake, Gadiel Seroussi, and Nigel P. Smart, *Elliptic curves in cryptography*, London Mathematical Society Lecture Note Series, vol. 265, Cambridge University Press, Cambridge, 2000. Reprint of the 1999 original.

[11] ——— (ed.), *Advances in elliptic curve cryptography*, London Mathematical Society Lecture Note Series, vol. 317, Cambridge University Press, Cambridge, 2005.

[12] Dan Boneh and Matthew Franklin, *Identity-based encryption from the Weil pairing*, SIAM J. Comput. **32** (2003), no. 3, 586–615 (electronic).

[13] Dan Boneh, Ben Lynn, and Hovav Shacham, *Short signatures from the Weil pairing*, J. Cryptology **17** (2004), no. 4, 297–319.

[14] David Cash, Eike Kiltz, and Victor Shoup, *The twin Diffie-Hellman problem and applications*, Advances in cryptology—EUROCRYPT 2008 (Istanbul, Turkey), Lecture Notes in Comput. Sci., vol. 4965, Springer, Berlin, 2008, pp. 127–145.

[15] Certicom Corp., *Certicom ECC Challenge*, November 1997. `http://www.certicom.com/index.php/the-certicom-ecc-challenge`.

[16] Clifford C. Cocks and Richard G. E. Pinch, *Identity-based cryptosystems based on the Weil pairing*, 2001. Unpublished manuscript.

[17] Henri Cohen, Gerhard Frey, Roberto Avanzi, Christophe Doche, Tanja Lange, Kim Nguyen, and Frederik Vercauteren (eds.), *Handbook of elliptic and hyperelliptic curve cryptography*, Discrete Mathematics and its Applications (Boca Raton), Chapman & Hall/CRC, Boca Raton, FL, 2006.

[18] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers, *A secure and optimally efficient multi-authority election scheme*, Advances in cryptology—EUROCRYPT '97 (Konstanz), Lecture Notes in Comput. Sci., vol. 1233, Springer, Berlin, 1997, pp. 103–118.

[19] Whitfield Diffie and Martin E. Hellman, *New directions in cryptography*, IEEE Trans. Information Theory **IT-22** (1976), no. 6, 644–654.

[20] Vassil Dimitrov, Laurent Imbert, and Pradeep K. Mishra, *The double-base number system and its application to elliptic curve cryptography*, Math. Comp. **77** (2008), no. 262, 1075–1104.

[21] Danny Dolev, Cynthia Dwork, and Moni Naor, *Nonmalleable cryptography*, SIAM J. Comput. **30** (2000), no. 2, 391–437 (electronic).

[22] Taher ElGamal, *A public key cryptosystem and a signature scheme based on discrete logarithms*, Advances in cryptology (Santa Barbara, Calif., 1984), Lecture Notes in Comput. Sci., vol. 196, Springer, Berlin, 1985, pp. 10–18.

[23] Mireille Fouquet and François Morain, *Isogeny volcanoes and the SEA algorithm*, Algorithmic number theory (Sydney, 2002), Lecture Notes in Comput. Sci., vol. 2369, Springer, Berlin, 2002, pp. 276–291.

[24] David Freeman, Micahel Scott, and Edlyn Teske, *A taxonomy of pairing-friendly elliptic curves*, Cryptology ePrint Archive: Report 2006/372, 2006. `http://eprint.iacr.org/2006/372`.

[25] 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** (1999), no. 5, 1717–1719.

[26] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart, *Pairings for cryptographers*, Discrete Appl. Math. **156** (2008), no. 16, 3113–3121.

[27] Robert P. Gallant, Robert J. Lambert, and Scott A. Vanstone, *Faster point multiplication on elliptic curves with efficient endomorphisms*, Advances in cryptology—CRYPTO 2001 (Santa Barbara, CA), Lecture Notes in Comput. Sci., vol. 2139, Springer, Berlin, 2001, pp. 190–200.

[28] Darrel Hankerson, Alfred Menezes, and Scott Vanstone, *Guide to elliptic curve cryptography*, Springer Professional Computing, Springer-Verlag, New York, 2004.

[29] Florian Hess, Nigel P. Smart, and Frederik Vercauteren, *The eta pairing revisited*, IEEE Trans. Inform. Theory **52** (2006), no. 10, 4595–4602.

[30] Don B. Johnson and Alfred J. Menezes, *Elliptic curve DSA (ECSDA): an enhanced DSA*, SSYM'98: Proceedings of the 7th conference on USENIX Security Symposium (San Antonio, TX, 1998), USENIX Security Symposium, vol. 7, USENIX Association, Berkeley, CA, USA, 1998, pp. 13–13.

[31] Don B. Johnson, Alfred J. Menezes, and Scott A. Vanstone, *The elliptic curve digital signature algorithm (ECDSA)*, Intern. J. of Information Security **1** (2001), 36–63.

[32] Antoine Joux, *The Weil and Tate pairings as building blocks for public key cryptosystems*, Algorithmic number theory (Sydney, 2002), Lecture Notes in Comput. Sci., vol. 2369, Springer, Berlin, 2002, pp. 20–32.

[33] ———, *A one round protocol for tripartite Diffie-Hellman*, J. Cryptology **17** (2004), no. 4, 263–276.

[34] Antoine Joux, Reynald Lercier, Nigel Smart, and Frederik Vercauteren, *The number field sieve in the medium prime case*, Advances in cryptology—CRYPTO 2006, Lecture Notes in Comput. Sci., vol. 4117, Springer, Berlin, 2006, pp. 326–344.

[35] H. W. Lenstra Jr., *Factoring integers with elliptic curves*, Ann. of Math. (2) **126** (1987), no. 3, 649–673.

[36] Reynald Lercier and François Morain, *Counting the number of points on elliptic curves over finite fields: strategies and performances*, Advances in cryptology—EUROCRYPT '95 (Saint-Malo, 1995), Lecture Notes in Comput. Sci., vol. 921, Springer, Berlin, 1995, pp. 79–94.

[37] Ben Lynn, *The Pairing-Based Cryptography Library*. `http://crypto.stanford.edu/pbc/`.

[38] Alfred J. Menezes, Tatsuaki Okamoto, and Scott A. Vanstone, *Reducing elliptic curve logarithms to logarithms in a finite field*, IEEE Trans. Inform. Theory **39** (1993), no. 5, 1639–1646.

[39] Victor S. Miller, *The Weil pairing, and its efficient calculation*, J. Cryptology **17** (2004), no. 4, 235–261.

[40] Atsuko Miyaji, Masaki Nakabayashi, and Shunzo Takano, *New explicit conditions of elliptic curve traces for FR-reduction*, IEICE Transactions on Fundamentals **E84-A** (2001), no. 5, 1234-1243.

[41] F. Morain and J. Olivos, *Speeding up the computations on an elliptic curve using addition-subtraction chains*, RAIRO Inform. Théor. Appl. **24** (1990), no. 6, 531–543 (English, with French summary).

[42] National Institute of Standards and Technology, *Secure Hash Standard (SHS)*, Technical Report FIPS PUB 180–2, August 2002. `http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf`.

[43] ———, *Digital Signature Standard (DSS)*, Technical Report FIPS PUB 186–2, January 2000. `http://csrc.nist.gov/publications/fips/fips186-2/fips186-2-change1.pdf`.

[44] Stephen C. Pohlig and Martin E. Hellman, *An improved algorithm for computing logarithms over* GF(p) *and its cryptographic significance*, IEEE Trans. Information Theory **IT-24** (1978), no. 1, 106–110.

[45] J. M. Pollard, *Monte Carlo methods for index computation* (mod p), Math. Comp. **32** (1978), no. 143, 918–924.

[46] Takakazu Satoh, *The canonical lift of an ordinary elliptic curve over a finite field and its point counting*, J. Ramanujan Math. Soc. **15** (2000), no. 4, 247–270.

[47] René Schoof, *Elliptic curves over finite fields and the computation of square roots mod* p, Math. Comp. **44** (1985), no. 170, 483–494.

[48] Gadiel Seroussi, *Compression and decompression of elliptic curve data points*, June 26, 2001. U.S. Patent 6,252,960.

[49] Joseph H. Silverman, *The arithmetic of elliptic curves*, Graduate Texts in Mathematics, vol. 106, Springer-Verlag, New York, 1986.

[50] Eric R. Verheul, *Evidence that XTR is more secure than supersingular elliptic curve cryptosystems*, J. Cryptology **17** (2004), no. 4, 277–296.

Index