J. Cryptology (1993) 6: 209–224

Journal of
CRYPTOLOGY

Elliptic Curve Cryptosystems and Their Implementation

Alfred J. Menezes and Scott A. Vanstone
Department of Combinatorics and Optimization, University of Waterloo,
Waterloo, Ontario, Canada N2L 3G1

Communicated by Andrew M. Odlyzko

Received 24 July 1991 and revised 13 August 1992

Abstract. Elliptic curves have been extensively studied for many years. Recent interest has revolved around their applicability to factoring integers, primality testing, and to cryptography. In this paper we explore the feasibility of implementing in hardware an arithmetic processor for doing elliptic curve computations over finite fields. Of special interest, for practical reasons, are the curves over fields of characteristic 2. The elliptic curve analogue of the ElGamal cryptosystem is also analyzed.

Key words. Elliptic curve cryptosystems, Public-key cryptography, Implementation.

1. Introduction

In 1976 Diffie and Hellman in their seminal paper [9] on public-key cryptography described a protocol whereby two parties can share a common piece of secret information over an insecure communications channel. The security of this protocol is based on the presumed intractability of the problem of computing logarithms in the multiplicative group of a large finite field. Later, in 1985, ElGamal [10] described methods for exploiting the intractability of this same problem in order to construct a public-key encryption scheme and a signature scheme. All three protocols mentioned can be generalized to work in an arbitrary finite cyclic group.

The K-rational points on an elliptic curve E defined over a field K form an abelian group. The addition operation of this abelian group involves a few arithmetic operations in the underlying field K, and is easy to implement, both in hardware and software. Hence the group E can be used to implement the Diffie–Hellman key-passing scheme, and the ElGamal public-key cryptosystem and signature schemes. This use of elliptic curves in designing cryptosystems was first suggested by Koblitz [13] and Miller [21].

Elliptic curve cryptosystems have the potential to provide security equivalent to that of the existing public-key schemes, but with shorter key lengths. Having short key lengths is a factor that can be crucial in some applications, for example, the design of smart-card systems. The arithmetic processor on a smart card is restricted in size to an area of roughly 20 mm^2. An RSA chip designed to do modular multiplication of 155 decimal digit numbers has about 50,000 transistors, while a chip designed to perform arithmetic in the field $F_{2^{593}}$ has about 100,000 transistors. With current technology, these devices are too large to be placed on a smart card. By comparison, a chip designed to do arithmetic in F_{2^m}, where $m \approx 200$, would have less than 15,000 transistors, and would occupy about 15% of the 20 mm^2 area assigned for the processor. Another advantage to be gained by using elliptic curves is that each user may select a different curve E, even though all users use the same underlying field K. Consequently, all users require the same hardware for performing the field arithmetic.

Recent advances in the computation of elliptic curve logarithms [18] necessitate that the elliptic curve and the underlying field be judiciously chosen. In this report we consider various issues that arise in the secure and efficient hardward implementation of the elliptic curve analogue of the ElGamal public-key cryptosystem.

We begin with a brief review of elliptic curves. For an elementary introduction to elliptic curves the reader is referred to Chapter 6 of the book by Koblitz [14], while for a more thorough treatment of the subject we refer the reader to [30]. Section 4 mentions how arithmetic in F_{2^m} can be efficiently implemented. This discussion helps in understanding why we choose (supersingular) elliptic curves over fields of characteristic 2, and this is done in Section 5. The elliptic curve analogue of the ElGamal cryptosystem is studied in Section 6. In Sections 7 and 8 we present two alternate schemes for adding points on an elliptic curve also suitable for the implementation of the ElGamal cryptosystem. In Section 9 we predict the performance of the cryptosystem. Section 10 extends the discussion of Sections 5–7 to nonsupersingular elliptic curves over F_{2^m}. Finally, in Section 11, we explain how elliptic curves can be used to implement some digital signature schemes.

We use the following notation. F_q denotes the finite field on q elements. By $\mathbb{Z}_n$ we denote the cyclic group of order n. The cardinality of a set S is denoted by $\# S$. Tr and Te are the functions $Tr: F_{2^m} \to F_2$, $Te: F_{2^m} \to F_4$, defined by $Tr(\alpha) = \alpha + \alpha^2 + \alpha^{2^2} + \cdots + \alpha^{2^{m-1}}$, $Te(\alpha) = \alpha + \alpha^{2^2} + \alpha^{2^4} + \cdots + \alpha^{2^{m-2}}$ (Te is only defined when m is even).

2. Review of Elliptic Curves

Assume first that F_q has characteristic greater than 3. An elliptic curve over F_q (in affine coordinates), denoted by $E(F_q)$, or simply by E, is the set of all solutions $(x, y) \in F_q \times F_q$ to the equation

$$y^2 = x^3 + ax + b, \tag{1}$$

where $a, b \in F_q$, and $4a^3 + 27b^2 \neq 0$, together with a special point $\mathcal{O}$, called the point at infinity.

It is well known that $E(F_q)$ is an (additively written) abelian group of rank 1 or 2, with the point $\mathcal{O}$ serving as its identity element. We have $E(F_q) \cong \mathbb{Z}_{n_1} \oplus \mathbb{Z}_{n_2}$, where n_2 divides n_1, and $n_2 | q - 1$. The rules for the group addition are summarized below.

Addition Formula for (1)

If $P = (x_1, y_1) \in E$, then $-P = (x_1, -y_1)$. If $Q = (x_2, y_2) \in E$, $Q \neq -P$, then $P + Q = (x_3, y_3)$, where

$$x_3 = \lambda^2 - x_1 - x_2,$$

$$y_3 = \lambda(x_1 - x_3) - y_1,$$

and

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \text{if} \quad P \neq Q, \\[2ex] \dfrac{3x_1^2 + a}{2y_1} & \text{if} \quad P = Q. \end{cases}$$

If F_q is a field of characteristic 2, then there are two types of elliptic curves over F_q. An elliptic curve of zero j-invariant is the set of solutions to the equation

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

where $a_3, a_4, a_6 \in F_q$, $a_3 \neq 0$, together with the point at infinity $\mathcal{O}$.

An elliptic curve of nonzero j-invariant is the set of solutions to the equation

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

where $a_2, a_6 \in F_q$, $a_6 \neq 0$, together with the point at infinity $\mathcal{O}$.

The addition formulae for the two types of curves over F_{2^m} is given below.

Addition Formula for (2)

Let $P = (x_1, y_1) \in E$; then $-P = (x, y_1 + a_3)$. If $Q = (x_2, y_2) \in E$ and $Q \neq -P$, then $P + Q = (x_3, y_3)$, where

$$x_3 = \begin{cases} \left(\dfrac{y_1 + y_2}{x_1 + x_2}\right)^2 + x_1 + x_2, & P \neq Q, \\[2ex] \dfrac{x_1^4 + a_4^2}{a_3^2}, & P = Q, \end{cases}$$

and

$$y_3 = \begin{cases} \left(\dfrac{y_1 + y_2}{x_1 + x_2}\right)(x_1 + x_3) + y_1 + a_3, & P \neq Q, \\[2ex] \left(\dfrac{x_1^2 + a_4}{a_3}\right)(x_1 + x_3) + y_1 + a_3, & P = Q. \end{cases}$$

Addition Formula for (3)

Let $P = (x_1, y_1) \in E$; then $-P = (x_1, y_1 + x_1)$. If $Q = (x_2, y_2) \in E$ and $Q \neq -P$, then $P + Q = (x_3, y_3)$, where

$$x_3 = \begin{cases} \left(\dfrac{y_1 + y_2}{x_1 + x_2}\right)^2 + \dfrac{y_1 + y_2}{x_1 + x_2} + x_1 + x_2 + a_2, & P \neq Q, \\[2ex] \dfrac{a_6}{x_1^2} + x_1^2, & P = Q, \end{cases}$$

and

$$y_3 = \begin{cases} \left(\dfrac{y_1 + y_2}{x_1 + x_2}\right)(x_1 + x_3) + x_3 + y_1, & P \neq Q, \\ x_1^2 + \left(x_1 + \dfrac{y_1}{x_1}\right)x_3 + x_3, & P = Q. \end{cases}$$

The well-known theorem of Hasse states that $\# E(F_q) = q + 1 - t$, where $|t| \leq 2\sqrt{q}$. The curve $E(F_q)$ is said to be supersingular if $t^2 = 0, q, 2q, 3q$, or $4q$. If the characteristic of F_q is 2 or 3, then a curve over F_q is supersingular if and only if it has j-invariant equal to 0. The curve E can be viewed as an elliptic curve over any extension field F_{q^k} of F_q; $E(F_q)$ is a subgroup of $E(F_{q^k})$. The Weil conjecture (which was proved for elliptic curves in 1934 by Hasse) enables $\# E(F_{q^k})$ to be computed from $\# E(F_q)$ as follows. Let $t = q + 1 - \# E(F_q)$. Then $\# E(F_{q^k}) = q^k + 1 - \alpha^k - \beta^k$, where α, β are complex numbers determined from the factorization of $1 - tT + qT^2 = (1 - \alpha T)(1 - \beta T)$.

A random point P in E can be selected by randomly choosing an element $x_1 \in F_q$, and solving (1), (2), or (3) for y. By Hasse's theorem, the probability that x_1 is the x-coordinate of a point in E is roughly 1/2. The order of P can be computed in polynomial time if the factorization of $\# E$ is known.

3. The Elliptic Curve Logarithm Problem

The discrete logarithm problem for a general group G is the following: given α, $\beta \in G$, determine an integer x such that $\beta = \alpha^x$, provided that such an integer exists. The integer x is called the discrete logarithm of β to the base α, and is uniquely determined modulo the order of α. For the elliptic curve discrete logarithm problem, we replace G by the group of points of an elliptic curve E, write the group law additively rather than multiplicatively, and replace α by P, an element of E. The security of the elliptic curve cryptosystems, to be discussed later, is based on the presumed intractability of this problem.

The best general-purpose algorithm for computing elliptic curve logarithms is the combination of Shanks' exponential baby-step giant-step method (for example, see [24]) and the Pohlig–Hellman method [26], and has a running time that is proportional to the square root of the largest prime divisor of $\# G$. The more powerful index-calculus attacks that are used to compute logarithms in the multiplicative group of a finite field do not appear to extend to elliptic curve groups, as argued by Miller in [21].

Recently, a method was discovered for reducing the logarithm problem in $E(F_q)$ to the logarithm problem in the finite field F_{q^k} for some integer k (MOV) [18], for the case $\gcd(\# E(F_q), q) = 1$. The MOV reduction uses the Weil pairing and yields a subexponential algorithm for computing logarithms in $E(F_q)$, provided that k is small.

In [18] it is shown that if E is a supersingular curve, then $k \leq 6$. More precisely, if $\# E(F_q) = q + 1 - t$, then $k = 2, 3, 4, 6, 1$ when $t^2 = 0, q, 2q, 3q, 4q$, respectively. In this case, to preclude the MOV attack, it is necessary to select an underlying field

Table 1. Orders of supersingular elliptic curves over F_{2^m}, where m is odd.

Curve	m	Order	Group type	k
$y^2 + y = x^3$	Odd	$q + 1$	Cyclic	2
$y^2 + y = x^3 + x$	$m \equiv 1, 7 \pmod 8$	$q + 1 + \sqrt{2q}$	Cyclic	4
	$m \equiv 3, 5 \pmod 8$	$q + 1 - \sqrt{2q}$	Cyclic	4
$y^2 + y = x^3 + x + 1$	$m \equiv 1, 7 \pmod 8$	$q + 1 - \sqrt{2q}$	Cyclic	4
	$m \equiv 3, 5 \pmod 8$	$q + 1 + \sqrt{2q}$	Cyclic	4

F_q of a sufficiently large size in order that the discrete logarithm problem in F_{q^k} be intractable using the best algorithms known for the latter problem [7], [8], [11]. It appears (see Section 5) that the supersingular curves over F_{2^m} are particularly convenient for implementation of elliptic curve cryptosystems, but some care must be exercised when selecting such a curve in light of the preceding result. In Tables 1 and 2, we list, for m odd and even, a representative curve from each of the isomorphism classes of supersingular curves over F_{2^m}, together with the order, group structure and value of k. We write q for 2^m, γ, α, β, δ, ω are any elements in F_{2^m} such that γ is a noncube, $Tr(\gamma^{-2}\alpha) = 1$, $Tr(\gamma^{-4}\beta) = 1$, $Te(\delta) \neq 0$, and $Tr(\omega) = 1$. For more details, consult [19].

If a nonsupersingular curve is desired, then the MOV attack can be avoided by simply choosing a curve $E(F_q)$ such that the corresponding k value is sufficiently large. (By sufficiently large we mean that $k \geq c$, where the discrete logarithm problem in F_{q^c} is considered intractable.) Let $E(F_q)$ be of type (n_1, n_2). We assume that n_1 is divisible by a large prime v. We further assume that the base point P has order also divisible by v. It can then be ensured that $k \neq l$ by simply checking that either v does not divide $q^l - 1$ or else v^2 does not divide $\#E(F_{q^l})$. To verify that

Table 2. Orders of supersingular elliptic curves over F_{2^m}, where m is even.

Curve	m	Order	Group type	k
$y^2 + \gamma y = x^3$	$m \equiv 0 \pmod 4$	$q + 1 + \sqrt{q}$	Cyclic	3
	$m \equiv 2 \pmod 4$	$q + 1 - \sqrt{q}$	Cyclic	3
$y^2 + \gamma y = x^3 + \alpha$	$m \equiv 0 \pmod 4$	$q + 1 - \sqrt{q}$	Cyclic	3
	$m \equiv 2 \pmod 4$	$q + 1 + \sqrt{q}$	Cyclic	3
$y^2 + \gamma^2 y = x^3$	$m \equiv 0 \pmod 4$	$q + 1 + \sqrt{q}$	Cyclic	3
	$m \equiv 2 \pmod 4$	$q + 1 - \sqrt{q}$	Cyclic	3
$y^2 + \gamma^2 y = x^3 + \beta$	$m \equiv 0 \pmod 4$	$q + 1 - \sqrt{q}$	Cyclic	3
	$m \equiv 2 \pmod 4$	$q + 1 + \sqrt{q}$	Cyclic	3
$y^2 + y = x^3 + \delta x$	m even	$q + 1$	Cyclic	2
$y^2 + y = x^3$	$m \equiv 0 \pmod 4$	$q + 1 - 2\sqrt{q}$	$\mathbb{Z}_{\sqrt{q}-1} \oplus \mathbb{Z}_{\sqrt{q}-1}$	1
	$m \equiv 2 \pmod 4$	$q + 1 + 2\sqrt{q}$	$\mathbb{Z}_{\sqrt{q}+1} \oplus \mathbb{Z}_{\sqrt{q}+1}$	1
$y^2 + y = x^3 + \omega$	$m \equiv 0 \pmod 4$	$q + 1 + 2\sqrt{q}$	$\mathbb{Z}_{\sqrt{q}+1} \oplus \mathbb{Z}_{\sqrt{q}+1}$	1
	$m \equiv 2 \pmod 4$	$q + 1 - 2\sqrt{q}$	$\mathbb{Z}_{\sqrt{q}-1} \oplus \mathbb{Z}_{\sqrt{q}-1}$	1

$k > c$, check that $k \neq l$, for each l, $1 \leq l \leq c$. The quantity $\# E(F_{q^l})$ can be easily obtained from $\# E(F_q)$ by applying the Weil conjecture as described in Section 2. For most nonsupersingular curves, the value of k will be too large for the MOV reduction to the useful. This statement is made precise in [16].

4. Field Arithmetic in F_{2^m}

Since we are most interested in elliptic curves over finite fields of characteristic 2, we briefly discuss how the arithmetic in F_{2^m} can be efficiently accomplished.

The field F_{2^m} can be viewed as a vector space of dimension m over F_2. Once a basis of F_{2^m} over F_2 has been chosen, the elements of F_{2^m} can be conveniently represented as 0–1 vectors of length m. In hardware, a field element is stored in a shift register of length m. Addition of field elements is performed by bitwise XOR-ing the vector representations, and takes one clock cycle. A normal basis of F_{2^m} over F_2 is a basis of the form

$$\{\beta, \beta^2, \beta^{2^2}, \ldots, \beta^{2^{m-1}}\},$$

where $\beta \in F_{2^m}$. Given any $\alpha \in F_{2^m}$, we can write $\alpha = \sum_{i=0}^{m-1} a_i \beta^{2^i}$, where $a_i \in F_2$. Notice that

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

with indices reduced modulo m. Hence a normal basis representation of F_{2^m} is preferred because squaring a field element can then be accomplished by a simple rotation of the vector representation, an operation that is easily implemented in hardware; squaring an element also takes one clock cycle.

To minimize the hardware complexity in multiplying field elements (i.e., to minimize the number of connections between the cells of the shift registers holding the multiplicands), the normal basis chosen has to belong to a special class called optimal normal bases. A description of these special normal bases can be found in [23], where constructions are given, together with a list of fields for which these bases exist. An associated architecture for a hardware implementation is given in [2]. Using this architecture, a multiplication can be performed in m clock cycles. For fields for which optimal normal bases do not exist, the so-called low complexity normal bases described in [5] may be useful.

Finally, the most efficient technique, from the point of view of minimizing the number of multiplications, to compute an inverse was proposed by Itoh, Teechai, and Tsujii, and is described in [1]. The method requires exactly $\lfloor \log_2(m-1) \rfloor + \omega(m-1) - 1$ field multiplications, where $\omega(m-1)$ denotes the Hamming weight of the binary representation of $m - 1$. However, it is costly in terms of hardware implementation in that it requires the storage of several intermediate results. An alternate method for inversion which is slower but which does not require the storage of such intermediate results is also described in [1].

Recently Newbridge Microsystems Inc., in conjunction with Cryptech Systems Inc. (Canada), has manufactured a single chip device that implements various public and conventional key cryptosystems based on arithmetic in the field $F_{2^{593}}$. Since the field size is quite large, a slower two-pass multiplication technique was used in order to reduce the number of cell interconnections (see [2] or [27]). Also, to reduce the number of registers, the slower method mentioned in the previous paragraph to compute inverses was used. Multiplication of two elements takes 1300 clock cycles, while an inverse computation takes 50,000 clock cycles. The chip has a clock rating of 20 MHz, and so the multiplication and inverse computation take 0.065 ms and 2.5 ms, respectively.

More recently, a custom gate array device has been constructed [4] to do field operations in $F_{2^{155}}$. This chip was explicitly designed to perform the elliptic curve point additions efficiently. The chip is of relatively low complexity having about 11,000 gates and has a clock rate of 40 MHz.

5. Selecting a Curve and Field K

From the addition formulae in Section 2, we see that two distinct points on an elliptic curve can be added by means of three multiplications and one inversion of field elements in the underlying field K, while a point can be doubled in one inversion and four multiplications in K. This is true regardless of whether the curve has equation (1), (2), or (3). Additions and subtractions are not considered in this count, since these operations are relatively inexpensive. Our intention is to select a curve and field K so as to minimize the number of field operations involved in adding two points. Curves over $K = F_{2^m}$ are very attractive for the following reasons:

(i) The arithmetic in F_{2^m} is easier to implement in computer hardware than the arithmetic in finite fields of characteristic greater that 2.
(ii) When using a normal basis representation for the elements of F_{2^m}, squaring a field element becomes a simple cyclic shift of the vector representation, and thus reduces the multiplication count in adding two points.
(iii) A third reason applies to supersingular curves. For supersingular curves over F_{2^m}, the inverse operation in doubling a point can be eliminated by choosing $a_3 = 1$, further reducing the operation count.

For these reasons we first consider curves over F_{2^m} of the form $y^2 + y = x^3 + a_4x + a_6$. A further advantage of using these curves is that it is then easy to recover the y-coordinate of a point given its x-coordinate plus a single bit of the y-coordinate. This is useful in message embedding, and in reducing message expansion in the ElGamal scheme, as is explained in Section 8. The implementation of nonsupersingular curves over F_{2^m} is considered in Section 10.

From Table 1, we see that there are precisely three isomorphism classes of supersingular elliptic curves over F_{2^m}, m odd. A representative curve from each class is

$$E_1:\quad y^2 + y = x^3,$$

$$E_2:\quad y^2 + y = x^3 + x,$$

$$E_3:\quad y^2 + y = x^3 + x + 1.$$

The addition formula for E_1 simplifies to

$$x_3 = \begin{cases} \left(\dfrac{y_1 + y_2}{x_1 + x_2}\right)^2 + x_1 + x_2, & P \neq Q, \\ x_1^4, & P = Q, \end{cases}$$

and

$$y_3 = \begin{cases} \left(\dfrac{y_1 + y_2}{x_1 + x_2}\right)(x_1 + x_3) + y_1 + 1, & P \neq Q, \\ y_1^4 + 1, & P = Q. \end{cases}$$

The addition formulae for curves E_2 and E_3 is similar to that for E_1, except that the formula for doubling a point becomes

$$x_3 = x_1^4 + 1,$$
$$y_3 = y_1^4 + x_1^4.$$

If a normal basis representation is chosen for the elements of F_{2^m}, we see that doubling a point in E_1, E_2, or E_3 is "free," while adding two distinct points can be accomplished in two multiplications and one inversion. The multiple kP of the point P is computed by the repeated doubling and add method. If $\omega(k) = t + 1$, then the exponentiation takes $2t$ multiplications and t inversions.

6. Projective Coordinates

Even though there are special techniques for computing inverses in F_{2^m}, a field inversion is still far more expensive than a field multiplication (see Section 4). The inverse operation needed when adding two points can be eliminated by resorting to projective coordinates.

Let E be either E_1, E_2, or E_3. The curve E can be equivalently viewed as the set of all points in $\mathbb{P}^2(K)$ which satisfy the homogeneous cubic equation $y^2z + yz^2 = x^3$ (or $y^2z + yz^2 = x^3 + xz^2$, or $y^2z + yz^2 = x^3 + xz^2 + z^3$). Here $\mathbb{P}^2(K)$ denotes the projective plane over K. The points of $\mathbb{P}^2(K)$ are all of the equivalence classes of nonzero triples in K^3 under the equivalence relation $\sim$, where $(x, y, z) \sim (x', y', z')$ if and only if there exists $\alpha \in K^*$ such that $x' = \alpha x$, $y' = \alpha y$, and $z' = \alpha z$. The representative of an equivalence class containing (x, y, z) is denoted by $(x : y : z)$. Note that the only projective point in E with z-coordinate equal to 0 is the point $(0 : 1 : 0)$; this point is the point at infinity $\mathcal{O}$ of E. If $\mathcal{O} \neq (x : y : z) \in E$, then $(x : y : z) = (x/z : y/z : 1)$, and so the projective point $(x : y : z)$ corresponds uniquely to the affine point $(x/z, y/z)$.

Let $P = (x_1 : y_1 : 1) \in E$, $Q = (x_2 : y_2 : z_2) \in E$, and suppose that $P, Q \neq \mathcal{O}$, $P \neq Q$, and $P \neq -Q$. Since $Q = (x_2/z_2 : y_2/z_2 : 1)$ we can use the addition formula for E in affine coordinates to find $P + Q = (x_3' : y_3' : 1)$. We obtain

$$x_3' = \frac{A^2}{B^2} + x_1 + \frac{x_2}{z_2},$$
$$y_3' = 1 + y_1 + \frac{A}{B}\left(\frac{A^2}{B^2} + \frac{x_2}{z_2}\right),$$

where $A = (y_1 z_2 + y_2)$ and $B = (x_1 z_2 + x_2)$.

To eliminate the denominators of the expressions for x_3' and y_3', we set $z_3 = B^3 z_2$, $x_3 = x_3' z_3$, and $y_3 = y_3' z_3$, to obtain $P + Q = (x_3 : y_3 : z_3)$, where

$$\begin{aligned} x_3 &= A^2 B z_2 + B^4, \\ y_3 &= (1 + y_1) z_3 + A^3 z_2 + A B^2 x_2, \\ z_3 &= B^3 z_2. \end{aligned}$$

This addition formula can be done in nine multiplications of field elements, which is more that the two multiplications required when using affine coordinates. We save by not having to peform a costly inversion. The gain occurs at the expense of space, however, as we now need extra registers to store P and Q, and also to store intermediate results when doing the addition.

The multiple kP, where P is the affine point $(x_1, y_1, 1)$, can now be computed by repeatedly doubling P, and adding the result into an accumulator. The result $kP = (x_3, y_3, z_3)$ can be converted back into affine coordinates by multiplying each coordinate by z_3^{-1}. If $\omega(k) = t + 1$, then the total operation count to compute kP is $9t + 2$ field multiplications and one inversion.

7. Montgomery's Method

To reduce the number of registers needed to add points on an elliptic curve, a method for addition that is similar to that used by Montgomery in Section 10.3.1 of [22] may be used.

Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ be two distinct and nonzero points on E, with $P \neq -Q$. Then $P + Q = (x_3, y_3)$ satisfies

$$x_3 = \frac{(y_1 + y_2)^2}{(x_1 + x_2)^2} + x_1 + x_2. \tag{4}$$

Similarly, since $-Q = (x_2, y_2 + 1)$, $P - Q = (x_4, y_4)$ satisfies

$$x_4 = \frac{(y_1 + y_2)^2 + 1}{(x_1 + x_2)^2} + x_1 + x_2. \tag{5}$$

Adding (4) and (5), we obtain

$$x_3 = x_4 + \frac{1}{(x_1 + x_2)^2}. \tag{6}$$

Notice that to compute the x-coordinate x_3 of $P + Q$, we only need the x-coordinates of P, Q, and $P - Q$, and this can be accomplished with a single inversion.

We can now compute kP from P using the double and add method. First $2P$ is computed, and then we repeatedly compute either $(2mP, (2m + 1)P)$ or $((2m + 1)P, (2m + 2)P)$ from $(mP, (m + 1)P)$, depending on whether the corresponding bit in the binary representation of k is 0 or 1. Notice, however, that we have to use the addition formula (6) each time a new pair of points is computed, and this is done $\log_2 k$ times. In the methods of Sections 4 and 6, the corresponding addition formulae were only used t times when computing kP, where $\omega(k) = t + 1$. Thus the improvement in storage requirements when using the Montgomery method is at a considerable expense of speed.

8. ElGamal Cryptosystem

Let E be the curve E_1, E_2, or E_3 over F_{2^m}, m odd, and let P be a publicly known point on E, preferably a generator of E. The elements of F_{2^m} are represented with respect to a normal basis. User A randomly chooses an integer a and makes public the point aP, while keeping a itself secret. We assume that messages are ordered pairs of elements in F_{2^m}. To transmit the message (M_1, M_2) to user A, sender B chooses a random integer k and computes the poins kP and $akP = (\bar{x}, \bar{y})$. Assuming $\bar{x}, \bar{y} \neq 0$ (the event $\bar{x} = 0$ or $\bar{y} = 0$ occurs with very small probability for random k), B then sends A the point kP, and the field elements $M_1\bar{x}$ and $M_2\bar{y}$. (We multiply by M_1 and M_2 rather than add because if $M_1 + \bar{x}$ were sent, then it is more likely that a third party can change some bits of the message without being detected.) To read the message, A multiplies the point kP by her secret key a to obtain $(\bar{x}, \bar{y})$, from which she can recover M_1 and M_2 in two divisions.

In the above scheme, four field elements are transmitted in order to convey a message consisting of two field elements. We say that there is message expansion by a factor of 2. The message expansion factor can be reduced to $\frac{3}{2}$ by only sending the x-coordinate x_1 of kP and a single bit of the y-coordinate y_1 of kP. y_1 can easily be recovered from this information as follows. First $\alpha = x_1^3$, $x_1^3 + x_1$ or $x_1^3 + x_1 + 1$ is computed, depending on whether $E = E_1$, E_2, or E_3, respectively, by a single multiplication of x_1 and x_1^2. Since the trace of α must be 0, we have that either

$$y_1 = \alpha + \alpha^{2^2} + \alpha^{2^4} + \cdots + \alpha^{2^{m-1}}$$

or else

$$y_1 = \alpha + \alpha^{2^2} + \alpha^{2^4} + \cdots + \alpha^{2^{m-1}} + 1.$$

The identity 1 is represented by the vector of all 1's, and so the single bit of y_1 that was sent enables the correct choice for y_1 to be made. Notice that the computation of y_1 is inexpensive, since the terms in the formula for y_1 may be obtained by successively squaring α.

A drawback of the method described above is that if an intruder happens to know M_1 (or M_2), he can then easily obtain M_2 (or M_1). This attack can be prevented by only sending $(kP, M_1\bar{x})$, or by embedding M_1 on the curve. If the user wishes to embed messages on the elliptic curve, the following deterministic scheme may be used for the curve $E = E_1$. We assume that messages are $(m - 1)$-bit strings $M = (M_0, M_1, \ldots, M_{m-2})$. We can consider M as an element of F_{2^m} (where $M_{m-1} = 0$). To embed M on the curve, M^3 is first computed and then the trace of M^3 is evaluated. If $Tr(M^3) = 0$, then we set $x_M = M$, otherwise we set $x_M = M + 1$. In either case, we have that $Tr(x_M^3) = 0$. As in the preceding paragraph, y_M such that $P_M = (x_M, y_M)$ is a point on E can be easily found. Sender B can now transmit to A the pair of points $(kP, akP + P_M)$. With this scheme the message expansion is by a factor of 4. The message expansion factor can be reduced to 2 by sending only the x-coordinate and a single bit of the y-coordinate of each point. Note that after user A recovers x_M, she can decide whether the message sent is x_M or $x_M + 1$, by simply checking whether the last bit of x_M is 0 or 1, respectively.

9. Implementation

We estimate the throughput rate of encryption using the elliptic curve analogue of the ElGamal public-key cryptosystem. We choose the curves E_2 and E_3 over F_{2^m}, where m is odd. The elements of F_{2^m} are represented with respect to an optimal normal basis. We assume that a multiplication in F_{2^m} takes m clock cycles, while an inversion takes $I(m) = \lfloor \log_2(m-1) \rfloor + \omega(m-1) - 1$ multiplications. For simplicity, we ignore the cost of field additions and squarings.

It was noted in Section 3 that computing logarithms in E_2 or E_3 is believed to be as hard as computing logarithms in $F_{2^{4m}}$. We can thus achieve a high level of security using the elliptic curve ElGamal cryptosystem, but by using a significantly smaller field than is necessary for a secure implementation of the ElGamal cryptosystem over a finite field. Since the field size is small, we can assume that the number of registers used is not a crucial factor in an efficient implelementation. We thus represent points using projective coordiates.

In the ElGamal system the computation of $kP$and kaP requires m additions of points on average, for a randomly chosen k. To increase the speed of the system, and to place an upper bound on the time for encryption, we limit the Hamming weight of k to some integer d, where $d \leq m$. A similar technique is used in RSA (see [12]) and in [2]. The integer d should be selected so that $\binom{m}{d/2}$ is large in order to prevent the (close to) square-root methods [25]. For the present discussion, we choose $d = 30$.

The computation of kP and kaP takes 58 additions of points, 2 field inversions, and 4 field multiplications. Computing $m_1\bar{x}$ and $m_2\bar{y}$, where $kaP = (\bar{x}, \bar{y})$, takes another two multiplications. Thus two field elements can be encrypted in $528 + 2I(m)$ field multiplications. For concreteness we select the curve E_3 over $F_{2^{239}}$. This choice is appropriate because an optimal normal basis exists in $F_{2^{239}}$. Also since $\# E_3(F_{2^{239}})$ is a 72-digit prime, the square root attacks for computing elliptic curve logarithms do not apply. Finally, noting that $I(239) = 12$, and assuming a clock rate of 40 MHz, we get an encryption rate of 145K bits/s.

Table 3 lists some fields F_{2^m} for which an optimal norma basis exists, and where either $\# E_2(F_{2^m})$ or $\# E_3(F_{2^m})$ contains a large prime factor, precluding a square-root attack. The factorizations of the order of curves was obtained from [6]. The approximate running time for an index calculus attack in $F_{2^{4m}}$ is also included, using the asymptotic running time estimate of

$$e^{(1.35)n^{1/3}(\ln n)^{2/3}}$$

operations for computing discrete logarithms in F_{2^n} [24].

10. Using Nonsupersingular Curves

This discussion in this section is restricted to elliptic curves over fields of characteristic 2. However, it should be pointed out that nonsupersingular curves over fields of odd characteristic, and in particular prime fields, are also attractive for implementation.

Table 3. Some suitable supersingular curves of F_{2^m}, m odd.

m	Curve	Order of curve over F_{2^m}	Rough estimate of the operation count for index-calculus attack in $F_{2^{4m}}$
173	E_2	$5 \cdot 13625405957 \cdot$ P42	1.4×10^{18}
173	E_3	$7152893721041 \cdot$ P40	1.4×10^{18}
179	E_3	$1301260549 \cdot$ P45	2.5×10^{18}
191	E_2	$5 \cdot 3821 \cdot 89618875387061 \cdot$ P40	8.6×10^{18}
191	E_3	$25212001 \cdot 5972216269 \cdot$ P41	8.6×10^{18}
233	E_2	$5 \cdot 3108221 \cdot$ P63	4.3×10^{20}
239	E_2	$5 \cdot 77852679293 \cdot$ P61	7.2×10^{20}
239	E_3	P72	7.2×10^{20}
281	E_3	$91568909 \cdot$ PRP77	2.3×10^{22}
323	E_3	$137 \cdot 953 \cdot 525313 \cdot$ P87	5.3×10^{23}

There are $2(q-1)$ isomorphism classes of nonsupersingular elliptic curves over F_q, where $q = 2^m$ (and m is either even or odd). A set of representative curves, one from each class, is

$$y^2 + xy = x^3 + a_2 x^3 + a_6, \tag{7}$$

where $a_6 \in F_q \backslash \{0\}$, $a_2 \in \{0, \gamma\}$, and γ is an element in F_q of trace 1. If E is the curve $y^2 + xy = x^3 + a_6$, then its twist is the curve $\tilde{E}$: $y^2 + xy = x^3 + a_2 x^2 + a_6$. Note that $\# E(F_q) + \# \tilde{E}(F_q) = 2q + 2$, and that $\# E(F_q) \equiv 0 \pmod 4$.

As mentioned in Section 3, the best algorithms known for the logarithm problem in nonsupersingular elliptic curves is the baby-step giant-step algorithm. A nonsupersingular curve that is suitable for cryptographic applications is one whose order is divisible by a large prime factor, say a prime factor of at least 40 decimal digits. Consequently, the underlying field should be of size at least 2^{130}. The underlying field should also have an optimal normal basis, in order to achieve efficient field arithmetic. In addition, we prefer a curve whose group is cyclic; this will be the case if $\# E(F_q)$ has no repeated prime factors. From the addition formulae in Section 2, we see that adding two distinct points takes two field multiplications and one inversion, while doubling a point takes three multiplications and one inversion. (Recall that doubling a point in a supersingular curve was for "free.") The need for computing inverses may be eliminated by resorting to projective coordinates. We include the addition formulae for projective coordinates below:

Let $P = (x_1 : y_1 : z_1)$, $Q = (x_2 : y_2 : 1)$, with $P, Q \neq \mathcal{O}$, $P \neq -Q$, and let $P + Q = (x_3 : y_3 : z_3)$.

If $P \neq Q$, then

$$x_3 = AD,$$

$$y_3 = CD + A^2(Bx_1 + Ay_1),$$

$$z_3 = A^3 z_1,$$

where $A = x_2 z_1 + x_1$, $B = y_2 z_1 + y_1$, $C = A + B$, and $D = A^2(A + a_2 z_1) + z_1 BC$. Computing $P + Q$ can be done in 13 multiplications.

If $2P = (x_3 : y_3 : z_3)$, then

$$x_3 = AB,$$
$$y_3 = x_1^4 A + B(x_1^2 + y_1 z_1 + A),$$
$$z_3 = A^3,$$

where $A = x_1 z_1$ and $B = a_6 z_1^4 + x_1^4$. Computing $2P$ can be done in seven multiplications.

Of course, the nonsupersingular curves may also be used to implement the ElGamal cryptosystem as in Section 8. The advantage of using a nonsupersingular curve is that the same security level can be attained as with a supersingular curve, but with a much smaller underlying field. This results in smaller key lengths, faster field arithmetic, and a smaller processor for performing the arithmetic. Another advantage of using nonsupersingular curves is that each user of the system may select a different curve E, even though all users use the same underlying field F_q. Thus, all users require the same hardware for performing the field arithmetic.

If a random elliptic curve E is required, then $\# E(F_q)$ can be computed in polynomial time by Schoof's algorithm [29], as suitably adapted by Koblitz to curves over fields of characteristic 2 [15]. The algorithm has a running time of $O((\log q)^8)$ bit operations, however, it is practical for computing the order of curves over F_{2^m} for m up to 155 [20]. Using heuristic arguments, Koblitz [15] showed that the probability of a random nonsupersingular curve $E(F_q)$ having the property that $N = \# E(F_q)$ is divisible by a prime factor $\geq N/B$ is about $(1/m)\log_2(B/2)$. Thus, for example, the probability that $\# E(F_{2^{155}})$ is divisible by a 40-digit prime is approximately

$$\frac{1}{155}\log_2\left(\frac{2^{155}}{2 \cdot 10^{40}}\right) \approx 0.136,$$

and so one can expect to try seven curves before a suitable one is found.

An alternative method for selecting curves is to choose a curve E defined over F_q, where q is small enough so that $\# E(F_q)$ can be computed directly, and then using the group $E(F_{q^n})$ for suitable n. Note that $\# E(F_{q^n})$ can easily be computed from $\# E(F_q)$. Observe also that if l divides n, then $\# E(F_{q^l})$ divides $\# E(F_{q^n})$, and so we should select n such that it is prime, or else a product of a small factor and a large prime.

In [17] Koblitz observed that if exponents k of a small Hamming weight are used, then doubling of points "almost $\frac{3}{4}$ for free" are obtained for the nonsupersingular curves $y^2 + xy = x^3 + 1$ and $y^2 + xy = x^3 + x^2 + 1$ when computing kP. Also in [17] is a list of curves defined over F_2 (respectively F_4, F_8, and F_{16}) such that $\# E(F_{q^n})$ has a prime factor of at least 30 digits, there exists an optimal normal basis in F_{q^n}, and any string of ≤ 4 zeros (respectively exactly 2, 3, 4 zeros) can be handled with a single addition of points.

When using the curve (7), message expansion can be reduced by sending x_1 and a single bit of y_1/x (if $x_1 \neq 0$), instead of sending the point $P = (x_1, y_1)$. y_1 can then be recovered by using the following method. First, if $x_1 = 0$, then $y_1 = \sqrt{a_6}$. If $x_1 \neq 0$, then the change of variables $(x, y) \to (x, xz)$ transforms (7) to $z^2 + z =$

$x + a_2 + a_6 x^{-2}$. Compute $\alpha = x_1 + a_2 + a_6 x_1^{-2}$. To solve the quadratic equation $z^2 + z = \alpha$, let $z = (z_0, z_1, \ldots, z_{m-1})$ and $\alpha = (\alpha_0, \alpha_1, \ldots, \alpha_{m-1})$ be the vector representations of z and α, respectively. Then $z^2 + z = (z_0 + z_{m-1}, z_0 + z_1, \ldots, z_{m-2} + z_{m-1})$. Each choice $z_0 = 0$ or $z_0 = 1$ uniquely determines a solution $\bar{z}$ to $z^2 + z = \alpha$, by comparing the components of $z^2 + z$ and α. The correct solution $\bar{z}$ is selected by comparison with the corresponding bit of y_1/x_1 that was transmitted. Finally, y_1 is recovered as $y_1 = x_1 \bar{z}$.

11. Digital Signatures

One of the true advantages of public-key cryptography is the digital signature. In 1985 ElGamal [10] established the existence of such signatures in discrete exponentiation systems based on the multiplicative cyclic group of a finite field. It is a straightforward matter to see that the concept carries over to a discrete exponentiation system based on any cyclic group. For completeness, we briefly describe how this is done.

Let G be a cyclic group of order n, and let α be a generating element. Let $\mathcal{M}$ denote the message space, where we suppose that $\#\mathcal{M} = n$. Let f and g be bijections from $\mathcal{M}$ and G, respectively, to the set of integers $\{0, 1, 2, \ldots, n-1\}$. Suppose person A has private key a and public key α^a and that A wants to sign a message $M \in \mathcal{M}$.

Creating Signatures. A does the following:

- Generate a random integer k such that $\gcd(k, n) = 1$.
- Compute the group element $r = \alpha^k$.
- Solve the congruence
$$f(M) \equiv ag(r) + sk \pmod{n} \tag{8}$$
for s.

The signature for M is the pair (r, s).

Checking Signatures. Given M and the signature (r, s), we verify as follows:

- Compute $r^s = \alpha^{ks}$ and $(\alpha^a)^{g(r)}$.
- Compute $(\alpha^{ag(r)})(\alpha^{ks})$ and $\alpha^{f(M)}$ and verify that they are the same group element.

Note that in computing the ElGamal signature $k^{-1} \pmod{n}$ must be computed. An easy modification avoids this situation. Instead of solving (8), solve

$$f(M) \equiv kg(r) + sa \pmod{n}.$$

This has the advantage that a is fixed and a^{-1} can be computed once and for all. The security of this modification relies partially on the intractability of finding a solution to the equation

$$u(x) = x^{g(x)}$$

in the group G. For more details, the interested reader is referred to [3].

Another modification of the ElGamal scheme is one given by Schnorr in [28]. This method requires a hash function $h: \mathcal{M} \times G \to \mathbb{Z}$.

Creating Signatures. To sign message M, person A does the following:

- Compute group element $r = \alpha^k$ for some random integer k.
- Compute the hash value of M and r, i.e., $e = h(M, r)$.
- Compute $s \equiv ae + k \pmod{n}$.

The signature for message M is (s, e).

Checking Signatures. Given M and the signature (s, e) we verify as follows:

- Compute α^s, $(\alpha^a)^e$, and thus $\alpha^s\alpha^{-ae} = b$.
- Verify that $h(M, b)$ equals e.

This method, although it requires a hash function, has the advantage that signatures can be smaller.

For clarity, we describe one method of applying the ElGamal signature scheme to elliptic curves over F_{2^m}.

Let $P = (x_0, y_0)$ be a generator for a cyclic subgroup G of the group of points of an elliptic curve over F_{2^m}, and let $n = \#G$. We take messages to be elements of F_{2^m}. Define a mapping $f: F_{2^m} \to \{0, 1, \ldots, 2^m - 1\}$ as follows: if $M = (M_0, M_1, \ldots, M_{m-1}) \in F_{2^m}$, then $f(M) = \sum_{i=0}^{m-1} M_i 2^i$. In general, f will not be a bijection from $\mathcal{M}$ to $\{0, 1, \ldots, n - 1\}$ because $n \neq 2^m$, but, in practice, this causes no problem as we can choose a curve E with $\#E(F_{2^m}) > 2^m$. Finally, we take g to be the map $g((x, y)) = y$ for all $(x, y) \in G$. Note that g is not a bijection, however, this is not a problem in practice.

Acknowledgements

We would like to thank the referees for carefully reading this paper, and for their comments which improved the presentation of the paper.

References

[1] G. Agnew, T. Beth, R. Mullin, and S. Vanstone, Arithmetic operations in $GF(2^m)$, *Journal of Cryptology*, **6** (1993), 3–13.

[2] G. Agnew, R. Mullin, I. Onyszchuk, and S. Vanstone, An implementation for a fast public-key cryptosystem, *Journal of Cryptology*, **3** (1991), 63–79.

[3] G. Agnew, R. Mullin, and S. Vanstone, Improved digital signature scheme based on discrete exponentiation, *Electronics Letters*, **26** (1990), 1024–1025.

[4] G. Agnew, R. Mullin, and S. Vanstone, An implementation of elliptic curve cryptosystems over $F_{2^{155}}$, *IEEE Journal on Selected Areas in Communications*, to appear.

[5] D. Ash, I. Blake, and S. Vanstone, Low complexity normal bases, *Discrete Applied Mathematics*, **25** (1989), 191–210.

[6] J. Brillhart, D. Lehmer, J. Selfridge, B. Tuckerman, and S. Wagstaff, Factorizations of $b^n \pm 1$, $b = 2, 3, 5, 6, 7, 10, 11, 12$ up to high powers, *Contemporary Mathematics*, **22**, 1983.

[7] D. Coppersmith, Fast evaluation of logarithms in fields of characteristic two, *IEEE Transactions on Information Theory*, **30** (1984), 587–594.

[8] D. Coppersmith, A. Odlyzko, and R. Schroeppel, Discrete logarithms in $GF(p)$, *Algorithmica*, **1** (1986), 1–15.

[9] W. Diffie and M. Hellman, New directions in cryptography, *IEEE Transactions on Information Theory*, **22** (1976), 644–654.

[10] T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, *IEEE Transactions on Information Theory*, **31** (1985), 469–472.

[11] T. ElGamal, A subexponential-time algorithm for computing discrete logarithms over $GF(p^2)$, *IEEE Transactions on Information Theory*, **31** (1985), 473–481.

[12] J. Hastad, On using RSA with low exponent in a public key network, *Advances in Cryptology: Proceedings of Crypto '85*, Lecture Notes in Computer Science, Vol. 218, Springer-Verlag, Berlin, 1986, pp. 403–408.

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

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

[15] N. Koblitz, Constructing elliptic curve cryptosystems in characteristic 2, *Advances in Cryptology: Proceedings of Crypto '90*, Lecture Notes in Computer Science, Vol. 537, Springer-Verlag, Berlin, 1991, pp. 156–167.

[16] N. Koblitz, Elliptic curve implementation of zero-knowledge blobs, *Journal of Cryptology*, **4** (1991), 207–213.

[17] N. Koblitz, CM-curves with good cryptographic properties, *Advances in Cryptology: Proceedings of Crypto '91*, Lecture Notes in Computer Science, Vol. 576, Springer-Verlag, Berlin, 1992, pp. 279–287.

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

[19] A. Menezes and S. Vanstone, Isomorphism classes of elliptic curves over finite fields of characteristic 2, *Utilitas Mathematica*, **38** (1990), 135–154.

[20] A. Menezes, S. Vanstone, and R. Zuccherato, Counting points on elliptic curves over F_{2^m}, *Mathematics of Computation*, **60** (1993), 407–420.

[21] V. Miller, Uses of elliptic curves in cryptography, *Advances in Cryptology: Proceedings of Crypto '85*, Lecture Notes in Computer Science, Vol. 218, Springer-Verlag, Berlin, 1986, pp. 417–426.

[22] P. Montgomery, Speeding the Pollard and elliptic curve methods of factorization, *Mathematics of Computation*, **48** (1987), 243–264.

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

[24] A. Odlyzko, Discrete logarithms and their cryptographic significance, *Advances in Cryptology: Proceedings of Eurocrypt '84*, Lecture Notes in Computer Science, Vol. 209, Springer-Verlag, Berlin, 1985, pp. 224–314.

[25] A. Odlyzko, Personal communication, 1986.

[26] S. Pohlig and M. Hellman, An improved algorithm for computing logarithms over $GF(p)$ and its crptographic significance, *IEEE Transactions on Information Theory*, **24** (1978), 106–110.

[27] T. Rosati, A high speed data encryption processor for public key cryptography, *Proceedings of IEEE Custom Integrated Circuits Conference*, San Diego, 1989, pp. 12.3.1–12.3.5.

[28] C. Schnorr, Efficient identification and signatures for smart cards, *Advances in Cryptology: Proceedings of Crpto '89*, Lecture Notes in Computer Science, Vol. 435, Springer-Verlag, Berlin, 1990, pp. 239–252.

[29] R. Schoof, Elliptic curves over finite fields and the computation of square roots mod p, *Mathematics of Computation*, **44** (1985), 483–494.

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