Efficient algorithms for pairing-based cryptosystems

Paulo S.L.M. Barreto[1], Hae Y. Kim[1], and Michael Scott[2]

[1] Universidade de São Paulo, Escola Politécnica.
Av. Prof. Luciano Gualberto, tr. 3, 158
BR 05508-900, São Paulo(SP), Brazil.
pbarreto@larc.usp.br, hae@lps.usp.br
[2] School of Computer Applications
Dublin City University
Ballymun, Dublin 9, Ireland.
mscott@indigo.ie

Abstract. We describe fast new algorithms to implement recent cryptosystems based on the Tate pairing. In particular, our techniques improve pairing evaluation speed by a factor of 20 compared to previously known methods. We also propose much faster algorithms for scalar multiplication and square root extraction, the latter technique being also useful in contexts other than that of pairing-based cryptography.

Keywords: elliptic curve cryptosystem, pairing-based cryptosystem.

1 Introduction

The recent discovery [8] of groups where the Decision Diffie-Hellman (DDH) problem is easy while the Computational Diffie-Hellman (CDH) problem is hard, and the subsequent definition of a new class of problems variously called the Gap Diffie-Hellman [8], Bilinear Diffie-Hellman [2], or Tate-Diffie-Hellman [6] class, has given rise to the development of several new cryptosystems based on pairings:

- Boneh-Lynn-Shacham [3] (BLS) short signatures.
- Boneh-Franklin identity-based encryption [2].
- Smart's identity-based authenticated key agreement protocol [21].
- Identity-based signatures schemes [17, 19].

The growing interest in this area, which relies on the use of supersingular elliptic curves, has led to new analyses of the associated security properties [6, 18], as well as to extensions to more general (e.g. hyperelliptic and superelliptic) algebraic curves [6].

However, a central operation in these systems is computing a bilinear pairing (e.g. the Weil or the Tate pairing), which are computationally expensive. Moreover, it is often the case that curves over fields of characteristic 3 must be used to achieve the maximum possible security level for supersingular curves.

Our goal is to make such systems entirely practical, and to this end we propose several efficient algorithms for the underlying arithmetic operations.

The contributions of this paper are:

- The definition of *point tripling* for supersingular elliptic curves over fields of characteristic 3, an operation that can be done in $O(m)$ steps in polynomial basis and $O(1)$ in normal basis, as opposed to conventional point doubling that takes $O(m^2)$ steps. Furthermore, a faster point addition algorithm is proposed for normal basis representation. These operations lead to a noticeably faster scalar multiplication algorithm in characteristic 3.
- An algorithm to compute square roots over $\mathbb{F}_{p^m}$ in $O(m^2 \log m)$ steps, where m is odd and $p \equiv 3 \pmod 4$ or $p \equiv 5 \pmod 8$. The best previously known algorithm for square root extraction under these conditions takes $O(m^3)$ steps. This operation is important for the point compression technique, whereby a curve point $P = (x, y)$ is represented by its x coordinate and one bit of its y coordinate, and its usefulness transcends pairing-based cryptography.
- A deterministic variant of Miller's algorithm to compute the Tate pairing that avoids many irrelevant operations present in the conventional algorithm, and reduces the contribution of the underlying scalar multiplication to the computational complexity from $O(m^3)$ to $O(m^2)$. The computations involved can be done in parallel with triple-and-add scalar multiplication. Besides, evaluation of the final powering in the Tate pairing benefits from the same technique we use to speedup the extraction of square roots.

All of these improvements are very practical and result in noticeably faster implementations.

This paper is organized as follows. Section 2 summarizes the mathematical concepts we will use in the remainder of the paper. Section 3 describes point tripling and derives a fast scalar multiplication algorithm for characteristic 3. Section 4 introduces a fast method to compute square roots that works for half of all finite fields, and an extension to half of the remaining cases. Section 5 presents our improvements for Tate pairing computation. Section 6 discusses experimental results. We conclude in section 7.

2 Mathematical preliminaries

2.1 Elliptic curve arithmetic

Let p be a prime number, m a positive integer and $\mathbb{F}_{p^m}$ the finite field with p^m elements; p is said to be the *characteristic* of $\mathbb{F}_{p^m}$, and m is its *extension degree*. We simply write $\mathbb{F}_q$ with $q = p^m$ when the characteristic or the extension degree are known from context or irrelevant for the discussion.

An *elliptic curve* $E(\mathbb{F}_q)$ is the set of solutions (x, y) over $\mathbb{F}_q$ to an equation of form $E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$, where $a_i \in \mathbb{F}_q$, together with an additional *point at infinity*, denoted O. The same equation defines curves over $\mathbb{F}_{q^k}$ for $k > 0$.

It is possible to impose an abelian group law on E. Explicit formulas for computing the coordinates of a point $P_3 = P_1 + P_2 = (x_3, y_3)$ from the coordinates of $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ are given in [20, algorithm 2.3]; we shall present below a subset of those formulas. The number of points of an elliptic curve $E(\mathbb{F}_q)$, denoted $\#E(\mathbb{F}_q)$, is called the *order* of the curve over the field $\mathbb{F}_q$. The *Hasse bound* states that $\#E(\mathbb{F}_q) = q+1-t$, where $|t| \leqslant 2\sqrt{q}$. The quantity t is called the *trace* of E. Of particular interest to us are *supersingular* curves, which are curves whose trace t is a multiple of the characteristic p.

The order of a point $P \in E(\mathbb{F}_q)$ is the least nonzero integer r such that $rP = O$. The order of a point always divides the curve order.

2.2 Bilinear maps and pairings

We follow the presentation of [2, section 3]. Let $\mathbb{G}_1$ be a cyclic additive group of order n and $\mathbb{G}_2$ a cyclic multiplicative group of the same order n, such that the Computational Diffie-Hellman problem is hard in both groups. A *bilinear map* is a function $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ satisfying the properties:

1. (Bilinearity) $e(aP, Q) = e(P, aQ) = e(P, Q)^a$ for all $P, Q \in \mathbb{G}_1$ and all $a \in \mathbb{Z}$.
2. (Non-degeneracy) If $e(P, Q) = 1$ for all $Q \in \mathbb{G}_1$, then $P = O$ (the identity element in $\mathbb{G}_1$). This means that e does not map all pairs in $\mathbb{G}_1 \times \mathbb{G}_1$ to 1 (the identity element in $\mathbb{G}_2$).
3. (Computability) There is an efficient algorithm to compute $e(P, Q)$ for all $P, Q, \in \mathbb{G}_1$.

Let E be an elliptic curve over $\mathbb{F}_q$ of order n, and let P be a point on E of prime order r where $r^2 \not| n$. The subgroup $\langle P \rangle$ is said to have *security multiplier* k for some $k > 0$ if $r \mid q^k - 1$ and $r \not| q^s - 1$ for any $s < k$. If E is supersingular, the value of k is bounded by $k \leqslant 6$ [12]. This bound is met by curves of characteristic 3 and no other characteristic; for instance, in characteristic 2 the maximum achievable value is $k = 4$ [11, section 5.2.2].

The group $E(\mathbb{F}_q)$ is a subgroup of $E(\mathbb{F}_{q^k})$. Let $P \in E(\mathbb{F}_q)$ be a point of order r such that $\langle P \rangle$ has security multiplier k. Then $E(\mathbb{F}_{q^k})$ contains a point Q of the same order r but linearly independent of P. Notice that, since $r \mid n$, both $nP = nQ = O$.

The set of all points of order r in $E(\mathbb{F}_{q^k})$, denoted $E[r]$, is a subgroup of $E(\mathbb{F}_{q^k})$ called the group of r-torsion points of E. Notice that $\langle P \rangle$ is a subgroup of $E[r]$, which in turn is a subgroup of $E[n]$ where n is the order of $E(\mathbb{F}_q)$.

A *divisor* is a formal sum of points on the curve $E(\mathbb{F}_{q^k})$. The *degree* of a divisor $\mathcal{A} = \sum_P a_P(P)$ is the sum $\sum_P a_P$. An abelian group structure is imposed on the set of divisors by the addition of corresponding coefficients in their formal sums; in particular, $n\mathcal{A} = \sum_P (na_P)(P)$.

Let $f : E(\mathbb{F}_{q^k}) \to \mathbb{F}_{q^k}$ be a function on the curve and $\mathcal{A} = \sum_P a_P(P)$ be a divisor of degree 0. We define $f(\mathcal{A}) \equiv \prod_P f(P)^{a_P}$. Note that, since $\sum_P a_P = 0$, $f(\mathcal{A}) = (cf)(\mathcal{A})$ for any factor $c \in \mathbb{F}^*_{q^k}$. The divisor of a function f is $(f) \equiv \sum_P \mathrm{ord}_P(f)(P)$ where $\mathrm{ord}_P(f)$ is the order of the zero or pole of f at P (if f

has no zero or pole at P, then $\mathrm{ord}_P(f) = 0$). A divisor $\mathcal{A}$ is called *principal* if $\mathcal{A} = (f)$ for some function (f). It is known [11, theorem 2.25] that a divisor $\mathcal{A} = \sum_P a_P(P)$ is principal if and only if the degree of $\mathcal{A}$ is zero and $\sum_P a_P P = O$. Two divisors $\mathcal{A}$ and $\mathcal{B}$ are equivalent, and we write $\mathcal{A} \sim \mathcal{B}$, if their difference $\mathcal{A} - \mathcal{B}$ is a principal divisor. Let $P \in E[n]$ where n is coprime to q and $\mathcal{A}_P$ be a divisor equivalent to $(P) - (O)$; under these circumstances the divisor $n\mathcal{A}_P$ is principal, and hence there is a function f_P such that $n\mathcal{A}_P = (f_P)$. The *Tate pairing* [5] is the bilinear map defined as $e_n(P, Q) = f_P(\mathcal{A}_Q)^{(q^k-1)/n}$.

3 Scalar multiplication in characteristic 3

The equation of a supersingular elliptic curve in characteristic 3 can always be written in the form $E : y^2 = x^3 + a_4x + a_6$, $a_4 \neq 0$ [20, prop. 1.1]. Arithmetic on such a curve is governed by the following rules, where $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$, $P_3 = P_1 + P_2 = (x_3, y_3)$. By definition, $-O = O$, $-P_1 = (x_1, -y_1)$, $P_1 + O = O + P_1 = P_1$; furthermore:

$$
\begin{aligned}
&P_1 = -P_2 && \Rightarrow P_3 = O.\\
&P_1 = \ \ P_2 && \Rightarrow \lambda \equiv -a_4/y_1,\ x_3 = x_1 + \lambda^2,\ y_3 = -(y_1 + \lambda^3).\\
&P_1 \neq -P_2,\ P_2 && \Rightarrow \lambda \equiv \frac{y_2 - y_1}{x_2 - x_1},\ x_3 = \lambda^2 - (x_1 + x_2),\ y_3 = y_1 + y_2 - \lambda^3.
\end{aligned}
$$

These rules in turn give rise to the *double-and-add* method to compute scalar multiples $V = kP$, $k \in \mathbb{Z}$. Let the binary representation of $k > 0$ be $k = (k_t \ldots k_1 k_0)_2$ where $k_i \in \{0, 1\}$ and $k_t \neq 0$. Computation of $V = kP$ proceeds as follows.

Double-and-add scalar multiplication:

set $V \leftarrow P$;
for $i \leftarrow t-1, t-2, \ldots, 1, 0$ do {
    set $V \leftarrow 2V$;
    if $k_i = 1$ then set $V \leftarrow V + P$;
}

By extension, one defines $0P = O$ and $(-k)P = k(-P) = -(kP)$.

Several improvements to this basic algorithm are well known [13]. However, one can do much better than this if $a_4, a_6 \in \mathbb{F}_3$, as we will now see.

3.1 Point tripling

In characteristic 3, *point tripling* for the supersingular curve $E : y^2 = x^3 + a_4x + a_6$ where $a_4, a_6 \in \mathbb{F}_3$, $a_4 \neq 0$, can be done in time $O(m)$ in polynomial basis,

or simply $O(1)$ in normal basis. Indeed, since the cubing operation is linear in characteristic 3, given $P = (x, y)$ one computes $3P = (x_3, y_3)$ with the formulas:

$$x_3 = (x^3)^3 + a_6(1 - a_4)$$
$$y_3 = -(y^3)^3$$

These formulas are derived from the basic arithmetic formulas above in a straightforward way.

The linearity of point tripling is similar to point doubling for supersingular curves in characteristic 2, as discovered by Menezes and Vanstone [14], and it leads to a triple-and-add scalar multiplication algorithm much faster than the double-and-add method to compute $V = kP$. Let the ternary representation of k be $k = (k_t \dots k_1 k_0)_2$ where $k_i \in \{0, 1, 2\}$ and $k_t \neq 0$. Computation of $V = kP$ proceeds as follows.

Triple-and-add scalar multiplication:

set $V \leftarrow P$ if $k_t = 1$, or $V \leftarrow -P$ if $k_t = 2$;
for $i \leftarrow t-1, t-2, \dots, 1, 0$ do {
    set $V \leftarrow 3V$;
    if $k_i = 1$ then set $V \leftarrow V + P$;
    if $k_i = 2$ then set $V \leftarrow V - P$;
}

Obviously, the same advanced techniques used for the double-and-add method can be easily applied to triple-and-add.

3.2 Projective coordinates

Koblitz [9] describes a method to add curve points over characteristic 3 in projective coordinates with 10 multiplications. Actually, point addition can be done with only 9 multiplications. Let $P_1 = (x_1, y_1, z_1)$, $P_2 = (x_2, y_2, 1)$; one computes $P_3 = P_1 + P_2 = (x_3, y_3, z_3)$ as:

$$A = x_2 z_1 - x_1,$$
$$B = y_2 z_1 - y_1,$$
$$C = A^3,$$
$$D = C - z_1 B^2,$$
$$x_3 = x_1 C - AD,$$
$$y_3 = BD - y_1 C,$$
$$z_3 = z_1 C.$$

To recover P_3 in affine coordinates one just sets $P_3 = (x_3/z_3,\ y_3/z_3)$. This involves one single inversion, which is usually only performed at the end of a scalar multiplication.

4 Square root extraction

One can use the elliptic curve equation $E : y^2 = f(x)$ over $\mathbb{F}_q$, where $f(x)$ is a cubic polynomial, to obtain a compact representation of curve points. The idea is to use a single bit from the ordinate y as a selector[1] between the two solutions of the equation $y^2 = f(x)$ for a given x.

In a finite field $\mathbb{F}_{p^m}$ where $p \equiv 3 \pmod 4$ and odd m, the best algorithm known [4, 13] to compute a square root executes $O(m^3)$ $\mathbb{F}_p$ operations. By that method, a solution of $x^2 = a$ is given by $x = a^{(p^m+1)/4}$, assuming a is a quadratic residue.

We first notice that, if $m = 2k + 1$ for some k:

$$\frac{p^m + 1}{4} = \frac{p+1}{4}\left[p(p-1)\sum_{i=0}^{k-1}(p^2)^i + 1\right],$$

so that

$$a^{(p^m+1)/4} = [(a^{\sum_{i=0}^{k-1}(p^2)^i})^{p(p-1)} \cdot a]^{(p+1)/4}.$$

These relations can be verified by straightforward induction. The quantity $a^{\sum_{i=0}^{k-1} u^i}$ where $u = p^2$ can be efficiently computed in an analogous fashion to Itoh-Teechai-Tsujii inversion [7], based on the Frobenius map in characteristic p:

$$a^{1+u+\cdots+u^{k-1}} = \begin{cases} (a^{1+u+\cdots+u^{\lfloor k/2\rfloor-1}}) \cdot (a^{1+u+\cdots+u^{\lfloor k/2\rfloor-1}})^{u^{\lfloor k/2\rfloor}}, & k \text{ even}, \\ ((a^{1+u+\cdots+u^{\lfloor k/2\rfloor-1}}) \cdot (a^{1+u+\cdots+u^{\lfloor k/2\rfloor-1}})^{u^{\lfloor k/2\rfloor}})^u \cdot a, & k \text{ odd}. \end{cases}$$

Notice that raising to a power of p is a linear operation in characteristic p (and almost for free in normal basis representation). It can be easily verified by induction that this method requires $\lfloor \lg k \rfloor + \omega(k) - 1$ field multiplications, where $\omega(k)$ is the Hamming weight of the binary representation of k. Additional $O(\log p)$ multiplications are needed to complete the square root evaluation due to the extra multiplication by a and to the raisings to $p - 1$ and $(p + 1)/4$, which can be done with the conventional exponentiation algorithm[2]. The overall cost is $O(m^2(\log m + \log p))$ $\mathbb{F}_p$ operations to compute a square root. If the characteristic p is fixed and small compared to m, the complexity is simply $O(m^2 \log m)$ $\mathbb{F}_p$ operations.

Similar recurrence relations hold for a variant of Atkin's algorithm [16, section A.2.5] for computing square roots in $\mathbb{F}_{p^m}$ when $p \equiv 5 \pmod 8$ and odd m, with the same $O(m^2(\log m + \log p))$ complexity. The details are left to the reader.

[1] In certain cryptographic applications one can simply discard y. This is the case, for instance, of the BLS signature scheme, where one only keeps the abscissa x as signature representative. Notice that one could discard the ordinates of public keys as well, without affecting the security level.

[2] If p is large, it may be advantageous to compute z^{p-1} as z^p/z, trading $O(\log p)$ multiplication by one inversion.

The general case is unfortunately not so easy. Neither the Tonelli-Shanks algorithm [4] nor Lehmer's algorithm [16, section A.2.5] can benefit entirely from the above technique, although partial improvements that don't change the overall complexity are possible.

The above improvements are useful not only for pairing-based cryptosystems, but for more conventional schemes as well (see e.g. [9, section 6]).

4.1 Abscissas vs. ordinates

As we saw above, solving a quadratic equation over $\mathbb{F}_{3^m}$ takes $O(m^2 \log m)$ time. In characteristic 3, one can alternatively solve for x the cubic equation $y^2 = f(x)$ in only $O(m^2)$ time, and represent a point $V = (x, y)$ by its ordinate y and one $\mathbb{F}_3$ component of x chosen to select among the three equation solutions. If x_0 is a solution, the other two are $x_1 = x_0 + 1$ and $x_2 = x_0 + 2$.

Suppose that a pairing $e(P, V)$ must be computed in a certain cryptosystem. In schemes where one only knows the abscissa x (i.e. where either V or $-V$ are equally acceptable for pairing evaluation), one avoids the burden of calculating two pairings by using the property that $e(P, -V) = e(P, V)^{-1}$, as noticed in [3, section 5.1]. On the other hand, it would seem that one must compute three pairings $e(P, V_i)$, $i \in \{0, 1, 2\}$, in settings where one only knows the ordinate y. However, one easily verifies that $V_0 + V_1 + V_2 = O$, and hence $e(P, V_2) = [e(P, V_0) \cdot e(P, V_1)]^{-1}$, and because P is the same and both V_0 and V_1 share a common ordinate y, these two pairings can be computed together, with only a modest performance penalty as compared to computing only one pairing.

5 Computing the Tate pairing

In this section we propose several improvements to Miller's algorithm for computing the Tate pairing, as well as to some features of the Tate pairing itself.

Let $P, Q, \in E[n]$ be linearly independent points. As we saw in section 2.2, the Tate pairing is defined as $e_n(P, Q) = f_P(\mathcal{A}_Q)^{(q^k-1)/n}$, where $\mathcal{A}_Q \sim (Q) - (O)$. Evaluation of $f_P(\mathcal{A}_Q)$ proceeds iteratively. Let $R_1, R_2 \in E[n]$ be two random curve points, and define $\mathcal{A}_b \equiv b(P + R_1) - b(R_1) - (bP) + (O)$. Since $\mathcal{A}_b$ is a principal divisor, there exists a function f_b such that $(f_b) = \mathcal{A}_b$; furthermore, $(f_n) = (f_P)$, so that $f_P(\mathcal{A}_Q) = f_n(\mathcal{A}_Q)$.

Consider functions $g_1(x, y) = a_1 x + b_1 y + c_1$ and $g_2(x, y) = x + c_2$ defined so that the secant line through points bP and cP (or the tangent at bP if $bP = cP$) satisfies $g_1(x, y) = 0$, and the vertical line through $(b+c)P$ satisfies $g_2(x, y) = 0$. It can be shown [2, appendix C] that the following relation (*Miller's formula*) holds:

$$f_{b+c}(\mathcal{A}_Q) = f_b(\mathcal{A}_Q) \cdot f_c(\mathcal{A}_Q) \cdot \frac{g_1(\mathcal{A}_Q)}{g_2(\mathcal{A}_Q)}.$$

Actual calculations use the divisor $(Q + R_2) - (R_2) \sim \mathcal{A}_Q$:

$$f_{b+c}(\mathcal{A}_Q) = f_b(\mathcal{A}_Q) \cdot f_c(\mathcal{A}_Q) \cdot \frac{g_1(Q + R_2)g_2(R_2)}{g_2(Q + R_2)g_1(R_2)},$$

assuming none of the g_i values above is zero (otherwise Miller's algorithm is said to fail for this choice of R_1 and R_2, and one should pick a new random pair).

It remains to compute $f_1(\mathcal{A}_Q)$, which is easy since $(f_1) \sim (P + R_1) - (R_1) - (P) + (O)$. Define functions $\gamma_1(x, y) = a_1 x + b_1 y + c_1$ and $\gamma_2(x, y) = x + c_2$ so that the secant through points P and R_1 satisfies $\gamma_1(x, y) = 0$ and the vertical through $P + R_1$ satisfies $\gamma_2(x, y) = 0$. In this case Miller's formula reads:

$$f_1(\mathcal{A}_Q) = \frac{\gamma_2(\mathcal{A}_Q)}{\gamma_1(\mathcal{A}_Q)} = \frac{\gamma_2(Q + R_2)\gamma_1(R_2)}{\gamma_1(Q + R_2)\gamma_2(R_2)}.$$

These formulas are used in Miller's algorithm, defined below.

Miller's algorithm:

compute $f_1(\mathcal{A}_Q)$;
set $V \leftarrow P$;
for $i \leftarrow t - 1, t - 2, \ldots, 1, 0$ do {
    // N.B. $V \equiv bP$ for some b;
    compute $f_{2b}(\mathcal{A}_Q)$ from $f_b(\mathcal{A}_Q)$, V and $2V$;
    set $V \leftarrow 2V$;
    if $k_i = 1$ then {
        compute $f_{b+c}(\mathcal{A}_Q)$ from $f_b(\mathcal{A}_Q)$, $f_c(\mathcal{A}_Q)$, V and $V + P$;
        set $V \leftarrow V + P$;
    }
}

5.1 Irrelevant factors

Let the curve order be $n = hr$ where r is prime, $h \neq 1$, and $\gcd(h, r) = 1$. For P, Q of order r, the Tate pairing $e_n(P, Q)$ can be made deterministic by choosing distinct finite points R_1 and R_2 from the subgroup of points of order h.

We first observe that, in general, the Tate pairing is computed on a pair of points (P, Q) where $P \in E(\mathbb{F}_q)$ and $Q \in E(\mathbb{F}_{q^k})$; hence, the coordinates of P are in $\mathbb{F}_q$, and so are the coordinates of R_1 and $P + R_1$ with the proposed choice of R_1. Hence, the coefficients of g_1, g_2, γ_1 and γ_2 are all in $\mathbb{F}_q$ as well.

We will need the following lemmas:

Lemma 1. *All factors contributed by R_2 to Miller's formula are in $\mathbb{F}_q$.*

Proof. The factors contributed by R_2 are $g_1(R_2)$, $g_2(R_2)$, $\gamma_1(R_2)$, and $\gamma_2(R_2)$. The choice of R_2 implies that its coordinates are in $\mathbb{F}_q$. Since the coefficients of g_1, g_2, γ_1 and γ_2 are also in $\mathbb{F}_q$, the factors contributed by R_2 are in $\mathbb{F}_q$. □

Lemma 2. *The factors contributed by R_2 and $Q + R_2$ to Miller's formula are always nonzero (in other words, Miller's algorithm will never fail with this choice of R_1 and R_2).*

Proof. These factors consist of the g_i and γ_i functions evaluated at R_2 and $Q + R_2$, $i = 1, 2$. Consider that:

- the line $g_1(x, y) = 0$ intercepts the curve only at bP, cP, and $-(b + c)P$;
- the line $g_2(x, y) = 0$ intercepts the curve only at $\pm(b + c)P$ and O;
- the line $\gamma_1(x, y) = 0$ intercepts the curve only at P, R_1, and $-(P + R_1)$;
- the line $\gamma_2(x, y) = 0$ intercepts the curve only at $\pm(P + R_1)$ and O.

It follows that R_2 cannot be any of these points, since by choice R_2 and P are in different subgroups, $R_2 \neq R_1$, and $R_2 \neq O$ (notice that, if $P + R_1 = \pm R_2$, then $P = R_1 \pm R_2$, so all these points would be in the same subgroup). Similarly, $Q + R_2$ cannot be any of these points, as otherwise R_2 would be in the subgroup of order r rather than h, or Q would be in the subgroup of order h rather than r. Therefore, the equations of these lines are not satisfied by either R_2 or $Q + R_2$, and all factors contributed by these points are nonzero. □

Lemma 3. *The value $q-1$ is a factor of $(q^k-1)/r$ for any factor r of the order n for a supersingular elliptic curve over $\mathbb{F}_q$ with security multiplier $k > 1$.*

Proof. Since $\mathbb{F}_q^*$ is a multiplicative subgroup of $\mathbb{F}_{q^k}^*$, it follows that $\#\mathbb{F}_q^* \mid \#\mathbb{F}_{q^k}^*$, i.e. $q-1 \mid q^k-1$. On the other hand, it is known [11, section 5.2.2] that the order n of a supersingular curve with security multiplier $k > 1$ does not divide $q - 1$, and hence no factor r of n does. Therefore $(q^k - 1)/r$ contains a factor $q-1$. □

These lemmas immediately lead to the following result:

Theorem 1. *As long as $k > 1$, the factors contributed by R_2 to Miller's formula are irrelevant to the computation of the Tate pairing $e_r(P, Q)$ where $r \mid n$.*

Proof. Since these factors are nonzero, evaluation of Miller's formula cannot fail due to them; moreover, by Fermat's Little Theorem for finite fields [10, lemma 2.3] these $\mathbb{F}_q$ factors disappear under the final raising to the power of $q - 1$, which originates as a factor of $(q^k - 1)/r$. Therefore they are irrelevant and can be omitted from the Tate pairing computation. □

5.2 Coupling pairing evaluation with point tripling

Miller's composition formula can be extended to work with point tripling, so as to obtain $f_{3b}(\mathcal{A}_Q)$ from $f_b(\mathcal{A}_Q)$ and points bP and $(3b)P$. Discarding the irrelevant factors, one obtains:

$$f_{3b}(\mathcal{A}_Q) = f_b^3(\mathcal{A}_Q) \cdot \frac{g_1(Q + R_2)}{g_2(Q + R_2)} \cdot \frac{g_3(Q + R_2)}{g_4(Q + R_2)},$$

where $g_1(x, y) = a_1x + b_1y + c_1$, $g_2(x, y) = x + c_2$, $g_3(x, y) = a_3x + b_3y + c_3$, and $g_4(x, y) = x + c_4$ are defined so that the tangent at bP satisfies $g_1(x, y) = 0$, the vertical through $(2b)P$ satisfies $g_2(x, y) = 0$, the secant through points $(2b)P$ and bP satisfies $g_3(x, y) = 0$, and the vertical through $(3b)P$ satisfies $g_4(x, y) = 0$.

Notice that it is not necessary to actually compute $(2b)P$: the coefficients of $g_3(x, y)$ can be obtained from bP and $-(3b)P$, and the inversion needed to compute the abscissa of $(2b)P$ needed for $g_2(x, y)$ is avoided with a scale factor.

The tripling formula is by itself more efficient than the doubling formula, since the squaring operation, which takes $O(m^2)$ time, is replaced by cubing, which has only linear complexity at most. Besides, it is invoked only a fraction $\log_3 2$ compared to the doubling case.

Furthermore, it is known [11, section 5.2.2] that the curve order over $\mathbb{F}_{3^m}$ has the form $n = 3^m \pm 3^{(m+1)/2} + 1$. Hence, computing $f_P(\mathcal{A}_Q)$ as $f_n(\mathcal{A}_Q)$ (as opposed to, say, $f_r(\mathcal{A}_Q)$, where $r \mid n$) is very efficient with point tripling, since only two additions or one addition and one subtraction are needed to compute $n\mathcal{A}_Q$ as $((3^{(m-1)/2} \pm 1)3^{(m+1)/2} + 1)\mathcal{A}_Q$.

An interesting observation is that, even if Miller's algorithm computes $f_r(\mathcal{A}_Q)$ for $r \mid n$, it is often the case that a technique similar to that used for square root extraction can be applied, reducing the number of point additions or subtractions from $O(m)$ down to $O(\log m)$. However, we won't elaborate on this possibility, as the above choice is clearly faster.

5.3 Choice of the subgroup order

Pairing evaluation over fields $\mathbb{F}_{p^2}$ of general characteristic (as used, for instance, in the Boneh-Franklin identity-based cryptosystem [2]) with Miller's algorithm can benefit from the above observations with a careful choice of parameters, particularly the size q of the subfield $\mathbb{F}_q$ of $\mathbb{F}_p$ where calculations are performed. Instead of choosing a random subfield prime, use a Solinas prime [22] of form $q = 2^\alpha \pm 2^\beta \pm 1$ (it is always possible to find such primes for practical subgroup sizes), since $qP = (2^\beta(2^{\alpha-\beta} \pm 1) \pm 1)P$ involves only two additions or subtractions plus α doublings.

With this technique, the contribution of the underlying scalar multiplication to the complexity of Miller's algorithm is only $O(m^2)$ instead of $O(m^3)$.

5.4 Speeding up the final powering

Evaluation of the Tate pairing $e_n(P, Q)$, where n is the curve order over $\mathbb{F}_{p^m}$, includes a final raising to the power of $(p^{km} - 1)/n$. The powering is usually computed in $O(m^3)$ steps. However, this exponent shows a rather periodical structure in base p. One can exploit this property in a fashion similar to the square root algorithm of section 4, reducing the computational effort to $O(m^2 \log m)$ steps.

5.5 Fixed-base pairing precomputation

Actual pairing-based cryptosystems often need to compute pairings $e(P, Q)$ where P is either fixed (e.g. the base point on the curve) or used repeatedly (e.g. a public key). In these cases, the underlying scalar multiplication in Miller's algorithm can be executed only once to precompute the coefficients of the g_i and

γ_i functions that appear in Miller's formula. One can also write factors of form $g(x, y) = ax + by + c$ where $a \neq -1, 0, 1$ as $a(x + b'y + c')$ where $b' = b/a$ and $c' = c/a$; the factored a is cancelled by the final powering in the Tate pairing as described in section 5.1, and can therefore be omitted altogether, thus saving one multiplication per $g(x, y)$ factor.

6 Experimental results

BLS signature generation is faster than RSA or DSA signing at the same security level. Table 1 compares the signing times for the RSA, DSA, ECSA, and BLS signature schemes. Implementations were based on the MIRACL library.

Table 1. Comparison of signing times on a PIII 1 GHz.

algorithm	signing time
RSA, $\|n\| = 1024$ bits, $\|d\| = 1007$ bits	7.90 ms
DSA, $\|p\| = 1024$ bits, $\|q\| = 160$ bits	4.09 ms
$\mathbb{F}_p$ ECDSA, $\|p\| = 160$ bits	4.00 ms
$\mathbb{F}_{2^{160}}$ ECDSA	5.77 ms
$\mathbb{F}_{3^{97}}$ BLS	3.54 ms

BLS signature verification speed shows an improvement by a factor of about 14 without preprocessing and about 20 with preprocessing, as listed in table 2.

Table 2. BLS signature verification times on a PIII 1 GHz.

underlying field	published [3]	without preprocessing	with preprocessing
$\mathbb{F}_{3^{97}}$	2900 ms	210 ms	142 ms

The performance of Boneh-Franklin identity-based encryption (IBE) is also comparable to other cryptosystems, as shown in table 3.

Table 3. IBE decryption times on a PIII 1 GHz.

system parameters	without preprocessing	with preprocessing
$\|p\| = 512$ bits, $\|q\| = 160$ bits	35 ms	21 ms

6.1 Implementation issues

Originally, the authors of the BLS scheme suggest representing $\mathbb{F}_{3^{6m}}$ as $\mathbb{F}_{3^6}[x]/\tau_m(x)$ for a suitable irreducible polynomial $\tau_m(x)$ [3, section 5.1]. It is our experience that the alternative representation as $\mathbb{F}_{3^m}[x]/\tau_6(x)$ using an irreducible trinomial $\tau_6(x)$ leads to better performance; moreover, both signing and verification benefit at once from any improvement made to the implementation of $\mathbb{F}_{3^m}$. Karatsuba multiplication with fairly low overhead is also possible.

7 Conclusion

We have proposed several new algorithms to implement pairing-based cryptosystems. Our algorithms are all practical and lead to significant improvements, not only for the pairing evaluation process but to other operations as well, such as elliptic curve scalar multiplication and square root extraction.

An interesting line of further research is the application of these techniques to more general algebraic curves; for instance, a fast n-th root algorithm in the lines of the square root algorithm presented here would be useful for superelliptic curves.

References

1. I. Blake, G. Seroussi and N. Smart, "Elliptic Curves un Cryptography," Cambridge University Press, 1999.
2. D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," Advances in Cryptology – Crypto'2001, Lecture Notes in Computer Science **2139**, pp. 213–229, Springer-Verlag, 2001.
3. D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing," Proceedings of Asiacrypt'2001, to appear. Preprint available online at `http://crypto.stanford.edu/dabo/abstracts/weilsigs.html`.
4. H. Cohen, "A Course in Computational Algebraic Number Theory," Springer-Verlag, 1993.
5. G. Frey, M. Müller, and H. Rück, "The Tate Pairing and the Discrete Logarithm Applied to Elliptic Curve Cryptosystems," IEEE Transactions on Information Theory **45(5)**, pp. 1717–1719, 1999.
6. S. Galbraith, "Supersingular curves in cryptography," Proceedings of Asiacrypt'2001, to appear.
7. T. Itoh, O. Teechai and S. Tsujii, "A fast algorithm for computing multiplicative inverses in GF(2^m) using normal bases," *Information and Computation* **78**, pp. 171–177, 1988.
8. A. Joux and K. Nguyen, "Separating Decision Diffie-Hellman from Diffie-Hellman in Cryptographic Groups," Cryptology ePrint Archive, Report 2001/003, available at `http://eprint.iacr.org/2001/003/`.
9. N. Koblitz, "An Elliptic Curve Implementation of the Finite Field Digital Signature Algorithm," Advances in Cryptology – Crypto'98, Lecture Notes in Computer Science **1462**, pp. 327–337, Springer-Verlag, 1998.

10. R. Lidl and H. Niederreiter, "Finite Fields," Encyclopedia of Mathematics and its Applications **20**, 2nd Ed. Cambridge University Press, 1997.
11. A.J. Menezes, "Elliptic Curve Public Key Cryptosystems," Kluwer International Series in Engineering and Computer Science, 1993.
12. A.J. Menezes, T. Okamoto and S.A. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field," IEEE Transactions on Information Theory *39*, pp. 1639–1646, 1993.
13. A.J. Menezes, P.C. van Oorschot and S.A. Vanstone, "Handbook of Applied Cryptography," CRC Press, 1997.
14. A.J. Menezes and S.A. Vanstone, "The implementation of elliptic curve cryptosystems," Advances in Cryptology – Auscrypt'90, Lecture Notes in Computer Science **453**, pp. 2–13, Springer-Verlag, 1990.
15. V. Miller, "Short Programs for Functions on Curves," unpublished manuscript, 1986.
16. IEEE Std 2000–1363, "Standard Specifications for Public Key Cryptography," 2000.
17. K.G. Paterson, "ID-based signatures from pairings on elliptic curves," Cryptology ePrint Archive, Report 2002/004, available at `http://eprint.iacr.org/2002/004/`.
18. K. Rubin and A. Silverberg, "The best and worst of supersingular abelian varieties in cryptology," Cryptology ePrint Archive, Report 2002/006, available at `http://eprint.iacr.org/2002/006/`.
19. R. Sakai, K. Ohgishi and M. Kasahara, "Cryptosystems based on pairing," 2000 Symposium on Cryptography and Information Security (SCIS2000), Okinawa, Japan, Jan. 26–28, 2000.
20. J.H. Silverman, "The Arithmetic of Elliptic Curves," Graduate Texts in Mathematics, vol. 106, Springer-Verlag, 1986.
21. N.P. Smart, "An Identity Based Authenticated Key Agreement Protocol Based on the Weil Pairing," Cryptology ePrint Archive, Report 2001/111, available at `http://eprint.iacr.org/2001/111/`.
22. J. Solinas, "Generalized Mersenne numbers," technical report CORR-39, Department of C&O, University of Waterloo, 1999, available at `http://www.cacr.math.uwaterloo.ca/`.