Efficient Algorithms for Pairing-Based Cryptosystems

Paulo S. L. M. Barreto[1], Hae Y. Kim[1],
Ben Lynn[2], and Michael Scott[3]

[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] Computer Science Department, Stanford University, USA.
blynn@cs.stanford.edu
[3] 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 about 55 compared to previously known methods in characteristic 3, and attain performance comparable to that of RSA in larger characteristics. We also propose faster algorithms for scalar multiplication in characteristic 3 and square root extraction over $\mathbb{F}_{p^m}$, the latter technique being also useful in contexts other than that of pairing-based cryptography.

1 Introduction

The recent discovery [11] 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 [11], Bilinear Diffie-Hellman [2], or Tate-Diffie-Hellman [6] class, has given rise to the development of a new, ever expanding family of cryptosystems based on pairings, such as:

- Short signatures [3].
- Identity-based encryption and escrow ElGamal encryption [2].
- Identity-based authenticated key agreement [29].
- Identity-based signature schemes [8, 22, 24].
- Tripartite Diffie-Hellman [10].
- Self-blindable credentials [33].

The growing interest and active research in this branch of cryptography has led to new analyses of the associated security properties and to extensions to more general (e.g. hyperelliptic and superelliptic) algebraic curves [6, 23].

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 are used to achieve the best possible ratio between security level and space requirements for supersingular curves, but such curves have received considerably less attention than their even or (large) prime characteristic counterparts. Our goal is to make such systems entirely practical and contribute to fill the theoretical gap in the study of the underlying family of curves, and to this end we propose several efficient algorithms for the arithmetic operations involved.

The contributions of this paper are:

- The definition of *point tripling* for supersingular elliptic curves over $\mathbb{F}_{3^m}$, that is, over fields of characteristic 3. A point tripling operation can be done in $O(m)$ steps (or essentially for free in hardware), 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 algorithms for square root extraction under these conditions take $O(m^3)$ 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 whenever one of the pairing's arguments is restricted to a base field (as opposed to having both in an extension field). Besides, in characteristics 2 and 3 both the underlying scalar multiplication and the final powering in the Tate pairing experience a complexity reduction from $O(m^3)$ to $O(m^2)$ steps.

All of these improvements are very practical and result in surprisingly faster implementations. Independent results on this topic have been obtained by Galbraith, Harrison and Soldera, and are reported in [7]; in particular, they provide a very clear and nice description of the Tate pairing.

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

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 the context or irrelevant for the discussion. We also write $\mathbb{F}_q^* \equiv \mathbb{F}_q - \{0\}$.

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 $k > 0$.

There exists an abelian group law on E. Explicit formulas for computing the coordinates of a point $P_3 = P_1 + P_2$ from the coordinates of P_1 and P_2 are given in [27, algorithm 2.3]; we shall present in section 3 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 Frobenius* (for brevity, we will call it simply 'trace'). Of particular interest to us are *supersingular* curves, which are curves whose trace t is a multiple of the characteristic p.

Let $n = \#E(\mathbb{F}_q)$. The order of a point $P \in E$ is the least nonzero integer r such that $rP = O$. The set of all points of order r in E is denoted $E[r]$, or $E(K)[r]$ to stress the particular subgroup $E(K)$ for a field K. The order of a point always divides the curve order. It follows that $\langle P \rangle$ is a subgroup of $E[r]$, which in turn is a subgroup of $E[n]$.

Let P be a point on E of prime order r where $r^2 \nmid n$. The subgroup $\langle P \rangle$ is said to have *security multiplier* k for some $k > 0$ if $r \mid q^k - 1$ and $r \nmid q^s - 1$ for any $0 < s < k$. If E is supersingular, the value of k is bounded by $k \leqslant 6$ [16]. This bound is attained in characteristic 3 but not in characteristic 2, where the maximum achievable value is $k = 4$ [15, section 5.2.2].

The group $E(\mathbb{F}_q)$ is (isomorphic to) 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.

We will consider in detail the curves listed in table 1, where k is the security multiplier, both m and p are prime numbers, and either $p \equiv 2 \pmod 3$ or $p \equiv 3 \pmod 4$. The curve orders are explicitly computed in [15, section 5.2.2].

Table 1. Some cryptographically interesting supersingular elliptic curves

curve equation	underlying field	curve order	k
$E_{1,b} : y^2 = x^3 + (1-b)x + b,\ b \in \{0,1\}$	$\mathbb{F}_p$	$p + 1$	2
$E_{2,b} : y^2 + y = x^3 + x + b,\ b \in \{0,1\}$	$\mathbb{F}_{2^m}$	$2^m + 1 \pm 2^{(m+1)/2}$	4
$E_{3,b} : y^2 = x^3 - x + b,\ b \in \{-1,1\}$	$\mathbb{F}_{3^m}$	$3^m + 1 \pm 3^{(m+1)/2}$	6

For our purposes, a *divisor* is a formal sum of points on the curve $E(\mathbb{F}_{q^m})$, $m > 0$. 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 let $\mathcal{A} = \sum_P a_P(P)$ 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 [15, 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 let $\mathcal{A}_P$ 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 $(f_P) = n\mathcal{A}_P = n(P) - n(O)$.

Let ℓ be a natural number coprime to q. The *Tate pairing* of order ℓ is the map $e_\ell : E(\mathbb{F}_q)[\ell] \times E(\mathbb{F}_{q^k})[\ell] \to \mathbb{F}_{q^k}^*$ defined[4] as $e_\ell(P, Q) = f_P(\mathcal{A}_Q)^{(q^k-1)/\ell}$. It satisfies the following properties:

- (Bilinearity) $e_\ell(P_1 + P_2, Q) = e_\ell(P_1, Q) \cdot e_\ell(P_2, Q)$ and $e_\ell(P, Q_1 + Q_2) = e_\ell(P, Q_1) \cdot e_\ell(P, Q_2)$ for all $P, P_1, P_2 \in E(\mathbb{F}_q)[\ell]$ and all $Q, Q_1, Q_2 \in E(\mathbb{F}_{q^k})[\ell]$. It follows that $e_\ell(aP, Q) = e_\ell(P, aQ) = e_\ell(P, Q)^a$ for all $a \in \mathbb{Z}$.
- (Non-degeneracy) If $e_\ell(P, Q) = 1$ for all $Q \in E(\mathbb{F}_{q^k})[\ell]$, then $P = O$. Alternatively, for each $P \neq O$ there exists $Q \in E(\mathbb{F}_{q^k})[\ell]$ such that $e_\ell(P, Q) \neq 1$.
- (Compatibility) Let $\ell = h\ell'$. If $P \in E(\mathbb{F}_q)[\ell]$ and $Q \in E(\mathbb{F}_{q^k})[\ell']$, then $e_{\ell'}(hP, Q) = e_\ell(P, Q)^h$.

Notice that, because $P \in E(\mathbb{F}_q)$, f_P is a rational function with coefficients in $\mathbb{F}_q$.

3 Scalar Multiplication in Characteristic 3

Arithmetic on the curve $E_{3,b}$ is performed according to the following rules. Let $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$, $P_3 = P_1 + P_2 = (x_3, y_3)$. By definition, $-O = O$, $-P_1 = (x_1, -y_1)$, $P_1 + O = O + P_1 = P_1$. Furthermore,

$$\begin{aligned}
P_1 &= -P_2 && \Rightarrow P_3 = O. \\
P_1 &= P_2 && \Rightarrow \lambda \equiv 1/y_1,\ x_3 = x_1 + \lambda^2,\ y_3 = -(y_1 + \lambda^3). \\
P_1 &\neq -P_2,\ P_2 && \Rightarrow \lambda \equiv \frac{y_2 - y_1}{x_2 - x_1},\ x_3 = \lambda^2 - (x_1 + x_2),\ y_3 = y_1 + y_2 - \lambda^3.
\end{aligned}$$

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 $V = kP \equiv P + P + \cdots + P$ (with k terms) proceeds as follows.

[4] This definition differs from those given in [5, 6] in that we restrict the first argument of e_ℓ to $E(\mathbb{F}_q)[\ell]$ and the second argument to $E(\mathbb{F}_{q^k})[\ell]$ instead of $E(\mathbb{F}_{q^k})[\ell]$ and $E(\mathbb{F}_{q^k})/\ell E(\mathbb{F}_{q^k})$ respectively, and we raise $f_P(\mathcal{A}_Q)$ to the power $(q^k - 1)/\ell$, so that e_ℓ maps to certain uniquely determined coset representatives. However, our definition keeps the properties listed above unchanged, and captures the essential properties needed in practice for cryptographical purposes.

Double-and-add scalar multiplication:

```
set V ← P
for i ← t − 1, t − 2, ..., 1, 0 do {
    set V ← 2V
    if k_i = 1 then set V ← V + P
}
return V
```

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

Several improvements to this basic algorithm are well known [1, 17]. However, one can do much better than this, as we will now see.

3.1 Point Tripling

In characteristic 3, *point tripling* for the supersingular curve $E_{3,b}$ can be done in time $O(m)$ in polynomial basis, or simply $O(1)$ in hardware using 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:

$$x_3 = (x^3)^3 - b$$
$$y_3 = -(y^3)^3$$

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

The linearity of point tripling corresponds to that of point doubling for supersingular curves in characteristic 2, as discovered by Menezes and Vanstone [18], and it leads to a triple-and-add scalar multiplication algorithm much faster than the double-and-add method. Let the signed ternary representation of k be $k = (k_t \ldots k_1 k_0)_2$ where $k_i \in \{-1, 0, 1\}$ and $k_t \neq 0$. Computation of $V = kP$ proceeds as follows.

Triple-and-add scalar multiplication:

```
set V ← P if k_t = 1, or V ← −P if k_t = −1
for i ← t − 1, t − 2, ..., 1, 0 do {
    set V ← 3V
    if k_i =  1 then set V ← V + P
    if k_i = −1 then set V ← V − P
}
return V
```

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 [12] describes a method to add curve points in 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:

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

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[5] 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, 17] to compute a square root executes $O(m^3)$, or more precisely $O(m^3 \log p)$, $\mathbb{F}_p$ 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 [9], 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

[5] In certain cryptographic applications one can simply discard y. This happens, for instance, in BLS signatures [3], 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.

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 a conventional exponentiation algorithm[6]. 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 [21, 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.

The general case is unfortunately not so easy. Neither the Tonelli-Shanks algorithm [4] nor Lehmer's algorithm [21, 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. [12, section 6]).

5 Computing the Tate Pairing

In this section we propose several improvements to Miller's algorithm [19] for computing the Tate pairing in the cases of cryptographical interest. Let $P \in E(\mathbb{F}_q)[\ell]$ and $Q \in E(\mathbb{F}_{q^k})[\ell]$ be linearly independent points, and let $n \equiv \#E(\mathbb{F}_q)$. As we saw in section 2, the Tate pairing is defined as $e_\ell(P,Q) = f_P(\mathcal{A}_Q)^{(q^k-1)/\ell}$, where $\mathcal{A}_Q \sim (Q) - (O)$ and $(f_P) = \ell(P) - \ell(O)$. Computation of the Tate pairing is helped by the following observations.

Lemma 1. *The value $q-1$ is a factor of $(q^k-1)/r$ for any factor r of n, for all curves on table 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 [15, section 5.2.2] that $\#E_{1,b} = q+1$, $\#E_{2,b} = q+1 \pm \sqrt{2q}$, and $\#E_{3,b} = q+1 \pm \sqrt{3q}$. In all cases, $\gcd(n, q-1) = 1$, and hence no factor r of n divides $q-1$. Therefore, $(q^k-1)/r$ contains a factor $q-1$. □

Theorem 1. *Let r be a factor of n. Then $e_r(P,Q) = f_P(Q)^{(q^k-1)/r}$ for $Q \neq O$ and for all curves on table 1.*

Proof. Suppose $R \notin \{O, -P\}$ is some point on the curve. Let f'_P be a function with divisor $(f'_P) = r(P+R) - r(R) \sim (f_P)$, so that $e_r(P,Q) = f'_P((Q) - (O))^{(q^k-1)/r}$. Since P has coordinates in $\mathbb{F}_p$, and because f'_P does not have a zero

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

or pole at O, we know that $f'_P(O) \in \mathbb{F}_q^*$. Thus $f'_P((Q)-(O)) = f'_P(Q)/f'_P(O)$. By Fermat's Little Theorem for finite fields [13, lemma 2.3], $f'_P(O)^{q-1} = 1$. Lemma 1 then ensures that $f'_P(O)^{(q^k-1)/r} = 1$. Hence, $f'_P(O)$ is an irrelevant factor and can be omitted from the Tate pairing computation, i.e. $e_r(P,Q) = f'_P(Q)^{(q^k-1)/r}$. Now consider P, Q to be fixed and R to be variable. Since the above statement holds for all $R \notin \{O, -P\}$ we have that $f'_P(Q)$ is a constant when viewed as a function of R, coinciding with the value of $f_P(Q)$. Therefore, $e_r(P,Q) = f_P(Q)^{(q^k-1)/r}$. □

Corollary 1. *One can freely multiply or divide $f_P(Q)$ by any nonzero $\mathbb{F}_q$ factor without affecting the pairing value.*

The above corollary is not the same property that enables one to replace (f) by (cf); in particular, it does not hold for the Weil pairing. Notice that the special case $Q = O$ where the theorem does not apply is trivially handled, since then $e_r(P,Q) = 1$.

In the next theorem, for each pair $U, V \in E(\mathbb{F}_q)$ we define $g_{U,V} : E(\mathbb{F}_{q^k}) \to \mathbb{F}_{q^k}$ to be (the equation of) the line through points U and V (if $U = V$, then $g_{U,V}$ is the tangent to the curve at U, and if either one of U, V is the point at infinity O, then $g_{U,V}$ is the vertical line at the other point). The shorthand g_U stands for $g_{U,-U}$: if $U = (u,v)$ and $Q = (x,y)$, then $g_U(Q) = x - u$.

Theorem 2 (Miller's formula). *Let P be a point on $E(\mathbb{F}_q)$ and f_c be a function with divisor $(f_c) = c(P) - (cP) - (c-1)(O)$, $c \in \mathbb{Z}$. For all $a, b \in \mathbb{Z}$, $f_{a+b}(Q) = f_a(Q) \cdot f_b(Q) \cdot g_{aP,bP}(Q)/g_{(a+b)P}(Q)$.*

Proof. The divisors of the line functions satisfy:

$$\begin{aligned}(g_{aP,bP}) &= (aP) + (bP) - (-(a+b)P) - 3(O),\\ (g_{(a+b)P}) &= ((a+b)P) + (-(a+b)P) - 2(O).\end{aligned}$$

Hence, $(g_{aP,bP}) - (g_{(a+b)P}) = (aP) + (bP) - ((a+b)P) - (O)$. From the definition of f_c we see that:

$$\begin{aligned}(f_{a+b}) &= (a+b)(P) - ((a+b)P) - (a+b-1)(O)\\ &= a(P) - (aP) - (a-1)(O)\\ &+ b(P) - (bP) - (b-1)(O)\\ &+ (aP) + (bP) - ((a+b)P) - (O)\\ &= (f_a) + (f_b) + (g_{aP,bP}) - (g_{(a+b)P}).\end{aligned}$$

Therefore $f_{a+b}(Q) = f_a(Q) \cdot f_b(Q) \cdot g_{aP,bP}(Q) \,/\, g_{(a+b)P}(Q)$. □

Notice that $(f_0) = (f_1) = 0$, so that $f_0(Q) = f_1(Q) = 1$. Furthermore, $f_{a+1}(Q) = f_a(Q) \cdot g_{aP,P}(Q)/g_{(a+1)P}(Q)$ and $f_{2a}(Q) = f_a(Q)^2 \cdot g_{aP,aP}(Q)/g_{2aP}(Q)$.

Let the binary representation of $\ell \geqslant 0$ be $\ell = (\ell_t, \ldots, \ell_1, \ell_0)$ where $\ell_i \in \{0,1\}$ and $\ell_t \neq 0$. Miller's algorithm computes $f_P(Q) = f_\ell(Q), Q \neq O$ by coupling the above formulas with the double-and-add method to calculate ℓP:

Miller's algorithm:

set $f \leftarrow 1$ and $V \leftarrow P$
for $i \leftarrow t-1, t-2, \ldots, 1, 0$ do {
  set $f \leftarrow f^2 \cdot g_{V,V}(Q)/g_{2V}(Q)$ and $V \leftarrow 2V$
  if $\ell_i = 1$ then set $f \leftarrow f \cdot g_{V,P}(Q)/g_{V+P}(Q)$ and $V \leftarrow V + P$
}
return f

5.1 Irrelevant denominators

We will now show that, when computing $e_n(P, \phi(Q))$ where $Q \in E(\mathbb{F}_q)$ and where ϕ is a distortion map [32], the g_{2V} and g_{V+P} denominators in Miller's algorithm can be discarded. The choice of parameters is important, and is summarized in table 2. Notice that there is no entry for $E_{1,1}$.

Table 2. Choice of distortion maps

curve (see table 1)	underlying field	distortion map	conditions
$E_{1,0}$	$\mathbb{F}_p$, $p > 3$	$\phi_1(x, y) = (-x, iy)$	$i \in \mathbb{F}_{p^2}$, $i^2 = -1$
$E_{2,b}$, $b \in \{0, 1\}$	$\mathbb{F}_{2^m}$	$\phi_2(x, y) = (x + s^2, y + sx + t)$	$s, t \in \mathbb{F}_{2^{4m}}$, $s^4 + s = 0$, $t^2 + t + s^6 + s^2 = 0$
$E_{3,b}$, $b \in \{-1, 1\}$	$\mathbb{F}_{3^m}$	$\phi_3(x, y) = (-x + r_b, iy)$	$r_b, i \in \mathbb{F}_{3^{6m}}$ $r_b^3 - r_b - b = 0$, $i^2 = -1$

Theorem 3. *With the settings listed in table 2, the denominators in Miller's formula can be discarded altogether without changing the value of $e_n(P, Q)$.*

Proof. We will show that the denominators become unity at the final powering in the Tate pairing.

- (Characteristic 2) Let $q \equiv 2^m$. From the defining condition $s^4 = s$ it follows by induction that $s^{4^t} = s$ for all $t \geqslant 0$; in particular, $s^{q^2} = s^{2^{2m}} = s$, and hence $(s^2)^{q^2} = s^2$. The denominators in Miller's formula have the form $g_U(\phi(Q)) \equiv x + s^2 + c$, where $x \in \mathbb{F}_q$ is the abscissa of Q and $c \in \mathbb{F}_q$, so that $x^{q^2} = x$ and $c^{q^2} = c$. Hence, $g_U(\phi(Q))^{q^2} = x^{q^2} + (s^2)^{q^2} + c^{q^2} = x + s^2 + c = g_U(\phi(Q))$, using the linearity of raising to powers of q in $\mathbb{F}_q$. It follows that $g_U(\phi(Q))^{q^2-1} = 1$. Now the exponent of the final powering in the Tate pairing has the form $z = (q^4 - 1)/n = (q + 1 \pm \sqrt{2q})(q^2 - 1)$, i.e. $q^2 - 1 \mid z$. Therefore, $g_U(\phi(Q))^z = 1$.

- (Characteristic 3) Let $q \equiv 3^m$. From the defining condition $r_b^3 - r_b - b = 0$ it follows by induction that $r_b^{3^t} = r_b + b(t \bmod 3)$ for all $t \geqslant 0$; in particular, $r_b^{q^3} = r_b^{3^{3m}} = r_b$. The denominators in Miller's formula have the form $g_U(\phi(Q)) \equiv r_b - x - c$, where $x \in \mathbb{F}_q$ is the abscissa of Q and $c \in \mathbb{F}_q$, so that $x^{q^t} = x$ and $c^{q^t} = c$ for all $t \geqslant 0$. Hence, $g_U(\phi(Q))^{q^3} = r_b^{q^3} - x^{q^3} - c^{q^3} = r_b - x - c = g_U(\phi(Q))$, using the linearity of raising to powers of q in $\mathbb{F}_q$. It follows that $g_U(\phi(Q))^{q^3-1} = 1$. Now the exponent of the final powering in the Tate pairing has the form $z = (q^6-1)/n = (q+1 \pm \sqrt{3q})(q^3-1)(q+1)$, i.e. $q^3 - 1 \mid z$. Therefore, $g_U(\phi(Q))^z = 1$.
- (Characteristic $p > 3$) The denominators in Miller's formula have the form $g_U(\phi(Q)) \equiv -x - c$, where $x \in \mathbb{F}_p$ is the abscissa of Q and $c \in \mathbb{F}_p$. Hence, $g_U(\phi(Q))^p = -x^p - c^p = -x - c = g_U(\phi(Q))$, using the linearity of raising to p in $\mathbb{F}_p$. It follows that $g_U(\phi(Q))^{p-1} = 1$. Now the exponent of final powering in the Tate pairing is precisely $z = (p^2-1)/n = p - 1$. Therefore, $g_U(\phi(Q))^z = 1$.

□

One can alternatively couple the evaluation of f_n with the more efficient triple-and-add method in characteristic 3. To this end one needs a recursive formula for $f_{3a}(Q)$, which is easy to obtain from Miller's formula: the divisor of f_{3a} is $(f_{3a}) = 3(f_a) + (g_{aP,aP}) + (g_{2aP,aP}) - (g_{2aP}) - (g_{3aP})$, hence discarding the irrelevant denominators one obtains:

$$f_{3b}(Q) = f_b^3(Q) \cdot g_{aP,aP}(Q) \cdot g_{2aP,aP}(Q).$$

Notice that it is not necessary to actually compute $2aP$, because the coefficients of $g_{2aP,aP}$ can be obtained from aP and $3aP$.

In characteristic 3, 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 $\log_3 2$ times compared to the doubling case. Furthermore, for the Tate pairing of order $n = (3^{(m-1)/2} \pm 1)3^{(m+1)/2} + 1$ the contribution of the underlying scalar multiplication to the complexity of Miller's algorithm is only $O(m^2)$ instead of $O(m^3)$, as it involves only two additions or one addition and one subtraction. An analogous observation holds for supersingular elliptic curves in characteristic 2.

An interesting observation is that, even if Miller's algorithm computes $f_r(Q)$ 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.

5.2 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 where calculations are performed. Instead of choosing a random subfield prime, use a Solinas prime [30] 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.

5.3 Speeding up the Final Powering

Evaluation of the Tate pairing $e_n(P, Q)$, where $n \equiv \#E(\mathbb{F}_{p^m})$, 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. As it turns out, it is actually possible to compute the power in only $O(m^2)$ steps, by carefully exploiting the structure of the exponent. Details of this process are given in appendix A.2.

5.4 Fixed-base Pairing Precomputation

Actual pairing-based cryptosystems often need to compute pairings $e_n(P, Q)$ 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 line functions $g_U(Q)$. The speedup resulting from this technique is more prominent for characteristic $p > 3$.

5.5 MNT curves

Until recently, the only elliptic curves known to have embedding degree $k \leqslant 6$ were supersingular like $E_{2,b}$ and $E_{3,b}$. As it turns out, it is possible to construct ordinary (non-supersingular) curves with $k \in \{3, 4, 6\}$. Such curves were first described by Miyaji, Nakabayashi and Takano in [20]; we call them MNT curves.

Briefly, MNT curves are built with the complex multiplication (CM) method [1, chapter VIII]. The idea is to impose certain constraints on the form of the underlying finite field $\mathbb{F}_q$, the curve order n, and the trace of Frobenius t, which are linked to each other by the relation $n = q + 1 - t$. These in turn lead to further constraints on the form of the CM equation $DV^2 = 4q - t^2$, which for $k \in \{3, 4, 6\}$ reduces to a Pell equation[7], whose solution is well known [28].

MNT curves address concerns that supersingular curves may not be as secure as ordinary curves. They are suitable for variants of pairing-based cryptosystems that do not involve distortion maps, like the BLS variant of [3, section 3.5] or

[7] There is reason to believe that one can effectively construct MNT-like curves with $k \in \{5, 8, 10, 12\}$, for which the CM equation reduces to a quartic elliptic Diophantine equation [31]. However, we refrain from further investigating this possibility here.

the general IBE variants of [2, section 4] and [6, section 3]. In such systems, the pairings have the form $e_\ell(P,Q)$ where $P \in E(\mathbb{F}_q)$ and $Q \in E(\mathbb{F}_{q^k})$, and both are chosen so that $e_\ell(P,Q) \neq 1$.

An important property of the MNT criteria is that $n \mid \Phi_k(q)$ but $n \nmid (q^k - 1)/\Phi_k(q)$, where Φ_k is the k-th cyclotomic polynomial. Due to this property, lemma 1 holds for MNT curves as well, and consequently, so do theorem 1 and corollary 1. Therefore, the deterministic version of Miller's algorithm presented in section 5 is equally valid for the MNT case. Furthermore, for even k it often happens that the point $Q = (x, y)$ in the variant cryptosystems can be chosen so that $x \in \mathbb{F}_{q^{k/2}}$ but $y \notin \mathbb{F}_{q^{k/2}}$; with this setting[8], denominator elimination as suggested in section 5.1 is also applicable.

6 Experimental Results

The heaviest operation in any pairing-based cryptosystem is the pairing computation. We give our timings for these operations in table 3.

Table 3. Tate pairing computation times (in ms) on a PIII 1 GHz.

underlying base field	timing
$\mathbb{F}_{3^{97}}$	26.2
$\mathbb{F}_{2^{271}}$	23.0
$\mathbb{F}_p$, $\lvert p\rvert = 512$ bits	20.0
$\mathbb{F}_p$ with preprocessing	8.6

Boneh-Lynn-Shacham (BLS) signature generation is comparable to RSA or DSA signing at the same security level. Table 4 compares the signing times for the RSA, DSA (without precomputation), ECDSA (without precomputation), and BLS signature schemes. We consider two BLS implementations, namely, one using the curve $E_{3,b}$ and one using an MNT curve.

Timings for BLS verification and Boneh-Franklin identity-based encryption (IBE) are listed in table 5. BLS signature verification speed for $\mathbb{F}_{3^{97}}$ shows an improvement by a factor of about 55 over published timings. The performance of IBE is also comparable to other cryptosystems; the data refers to a curve over $\mathbb{F}_p$ where $|p| = 512$ bits, using a subgroup of order q where q is a Solinas prime and $|q| = 160$ bits.

The implementations in this section were written in C/C++ and based on the MIRACL [26] library.

[8] Representing $\mathbb{F}_{q^k}$ in polynomial basis as $\mathbb{F}_q[t]/R_k(t)$ and carefully choosing $R_k(t)$, it is quite easy to find a point Q satisfying these constraints. For instance, if $R_k(t) = t^k + t^2 + c$ for some $c \in \mathbb{F}_q$, one can show that a suitable Q can be found by restricting the coordinates to the form $x = a_{k-2}t^{k-2} + a_{k-4}t^{k-4} + \cdots + a_2t^2 + a_0$ and $y = b_{k-1}t^{k-1} + b_{k-3}t^{k-3} + \cdots + b_3t^3 + b_1t$.

Table 4. Comparison of signing and verification times (in ms) on a PIII 1 GHz.

algorithm	signing	verification
RSA, $\|n\| = 1024$ bits, $\|d\| = 1007$ bits	7.90	0.40
DSA, $\|p\| = 1024$ bits, $\|q\| = 160$ bits	4.09	4.87
$\mathbb{F}_p$ ECDSA, $\|p\| = 160$ bits	4.00	5.17
$\mathbb{F}_{2^{160}}$ ECDSA	5.77	7.15
$\mathbb{F}_{3^{97}}$ BLS (supersingular)	3.57	53.0
$\mathbb{F}_p$ BLS (MNT), $\|p\| = 157$ bits	2.75	81.0

Table 5. BLS and IBE times (in ms) on a PIII 1 GHz.

operation	original [3, 14]	ours
BLS verification	2900	53
IBE encryption	170	48 (preprocessed: 36)
IBE decryption	140	30 (preprocessed: 19)

7 Conclusions and Acknowledgements

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. Investigating the conditions leading to composition operations computable in linear time in abelian varieties would also be of great interest.

We are very grateful to Dan Boneh, Steven Galbraith, Antoine Joux, Frederik Vercauteren, and the anonymous referees for their valuable comments and/or feedback regarding this work.

References

1. I. Blake, G. Seroussi and N. Smart, "Elliptic Curves in 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," Asiacrypt'2001, Lecture Notes in Computer Science **2248**, pp. 514–532, Springer-Verlag, 2002.
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," Asiacrypt'2001, Lecture Notes in Computer Science **2248**, pp. 495–513, Springer-Verlag, 2002.
7. S. Galbraith, K. Harrison and D. Soldera, "Implementing the Tate pairing," Algorithm Number Theory Symposium – ANTS V, Lecture Notes in Computer Science **2369**, Springer-Verlag, to appear.
8. F. Hess, "Exponent Group Signature Schemes and Efficient Identity Based Signature Schemes Based on Pairings," Cryptology ePrint Archive, Report 2002/012, available at `http://eprint.iacr.org/2002/012/`.
9. T. Itoh, O. Teechai and S. Tsujii, "A fast algorithm for computing multiplicative inverses in $\mathrm{GF}(2^m)$ using normal bases," *Information and Computation* **78**, pp. 171–177, 1988.
10. A. Joux, "A one-round protocol for tripartite Diffie-Hellman," Algorithm Number Theory Symposium – ANTS IV, Lecture Notes in Computer Science **1838**, pp. 385–394, Springer-Verlag, 2000.
11. 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/`.
12. 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.
13. R. Lidl and H. Niederreiter, "Finite Fields," Encyclopedia of Mathematics and its Applications **20**, 2nd Ed. Cambridge University Press, 1997.
14. B. Lynn, "Stanford IBE library," available at `http://crypto.stanford.edu/ibe/`.
15. A.J. Menezes, "Elliptic Curve Public Key Cryptosystems," Kluwer International Series in Engineering and Computer Science, 1993.
16. 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.
17. A.J. Menezes, P.C. van Oorschot and S.A. Vanstone, "Handbook of Applied Cryptography," CRC Press, 1997.
18. 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.
19. V. Miller, "Short Programs for Functions on Curves," unpublished manuscript, 1986.
20. A. Miyaji, M. Nakabayashi, and S. Takano, "New explicit conditions of elliptic curve traces for FR-reduction," IEICE Trans. Fundamentals, Vol. E84 A, no. 5, May 2001.
21. IEEE Std 2000–1363, "Standard Specifications for Public Key Cryptography," 2000.
22. 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/`.
23. K. Rubin and A. Silverberg, "Supersingular abelian varieties in cryptology," Advances in Cryptology – Crypto'2002, these proceedings.
24. 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.

25. R. Schroeppel, H. Orman, S. O'Malley, O. Spatscheck, "Fast Key Exchange with Elliptic Curve Systems," Advances in Cryptology – Crypto '95, Lecture Notes in Computer Science **963**, pp. 43–56, Springer-Verlag, 1995.
26. M. Scott, "Multiprecision Integer and Rational Arithmetic C/C++ Library (MIRACL)," available at `http://indigo.ie/~mscott/`.
27. J.H. Silverman, "The Arithmetic of Elliptic Curves," Graduate Texts in Mathematics, vol. 106, Springer-Verlag, 1986.
28. N.P. Smart, "The Algorithmic Resolution of Diophantine Equations," London Mathematical Society Student Text **41**, Cambridge University Press, 1998.
29. N.P. Smart, "An Identity Based Authenticated Key Agreement Protocol Based on the Weil Pairing," Electronics Letters, to appear.
30. 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/`.
31. N. Tzanakis, "Solving elliptic diophantine equations by estimating linear forms in elliptic logarithms. The case of quartic equations," *Acta Arithmetica* **75** (1996), pp. 165–190.
32. E. Verheul, "Evidence that XTR is more secure than supersingular elliptic curve cryptosystems," Advances in Cryptology – Eurocrypt'2001, Lecture Notes in Computer Science **2045** (2001), pp. 195–210.
33. E. Verheul, "Self-blindable Credential Certificates from the Weil Pairing," Asiacrypt'2001, Lecture Notes in Computer Science **2248**, pp. 533–551, Springer-Verlag, 2002.

A Implementation Issues

A.1 Field Representation

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 $\tau_6(x)$ (for instance, $\tau_6(x) = x^6 + x - 1$) leads to better performance for practical values of m; moreover, both signing and verification benefit at once from any improvement made to the implementation of $\mathbb{F}_{3^m}$. Karatsuba multiplication can also be used to great effect, as one $\mathbb{F}_{3^{6m}}$ multiplication can be implemented with only 18 $\mathbb{F}_{3^m}$ multiplications. Similar observations apply to characteristic 2, where one $\mathbb{F}_{2^{4m}}$ multiplication takes 9 $\mathbb{F}_{2^m}$ multiplications.

As it turns out, however, Karatsuba is *not* the fastest multiplication technique in all circumstances. As seen in section 5.1, it is often the case that the actual pairing to be computed is $e_n(P, \phi(Q))$ where both P and Q are on the curve over $\mathbb{F}_q$ (rather than the curve over the extension field $\mathbb{F}_{q^k}$), and the pairing algorithm can explicitly use the form of the ϕ distortion map to reduce the number of $\mathbb{F}_q$ products involved in Miller's formula down to only two per line equation evaluation.

A.2 Speeding up the Final Powering in the Tate Pairing

The exponentiation needed by the Tate pairing $e_n(P, Q) = f_P(Q)^z$ where $z = (q^k - 1)/n$ can be efficiently computed with the following observations:

1. (Characteristic $p > 3$) Assume that $p \equiv 2 \pmod 3$ and $p \equiv 3 \pmod 4$. The order of a curve $E_{1,b}$ is $n = p + 1$. Let the order of the curve subgroup of interest be r, and notice that $r \mid p + 1$. Consider the scenario where the representation of a point $t \in \mathbb{F}_{p^2}$ is $t = u + iv$ where $u, v \in \mathbb{F}_p$ and i satisfies $i^2 + 1 = 0$. The Tate exponent is $z = (p^2 - 1)/r = ((p + 1)/r) \cdot (p - 1)$. To calculate $s = w^z \bmod p$, compute $t = w^{(p+1)/r} \equiv u + iv$ and set $s = (u + iv)^{p-1} = (u + v)^p/(u + iv) = (u - v)/(u + iv)$, using the linearity of raising to p and the fact that $i^p = -i$ for $p \equiv 3 \pmod 4$. We can further simplify to obtain $s = (u^2 - v^2)/(u^2 + v^2) - 2uvi/(u^2 + v^2)$.
2. (Characteristic 2) Let $q = 2^m$. As we saw in the proof of theorem 3, the Tate exponent is of form $z = (q + 1 \pm \sqrt{2q})(q^2 - 1)$. Therefore, to calculate $s = w^z$ one computes $t = w^q \cdot w \cdot w^{\pm\sqrt{2q}}$ and $s = t^{q^2}/t$. Raising to the exponents q, $\sqrt{2q}$ and q^2 can be done in $O(m)$ steps using normal basis, or in $O(m^2)$ steps using polynomial basis with a careful choice of the reduction polynomial (see [25], for instance), while the small (and constant) number of multiplications and inversions can be done in $O(m^2)$ steps. Therefore, the complete operation takes time $O(m^2)$.
3. (Characteristic 3) Let $q = 3^m$. As we saw in the proof of theorem 3, the Tate exponent is of form $z = (q + 1 \pm \sqrt{3q})(q^3 - 1)(q + 1)$. Therefore, to calculate $s = w^z$ one computes $u = w^q \cdot w \cdot w^{\pm\sqrt{3q}}$, $t = u^{q^3}/u$, and $s = t^q \cdot t$. Raising to the exponents q, $\sqrt{3q}$ and q^3 can be done in $O(m)$ steps using normal basis, or in $O(m^2)$ steps using polynomial basis with a careful choice of the reduction polynomial (see [25], for instance), while the small (and constant) number of multiplications and inversions can be done in $O(m^2)$ steps. Therefore, the complete operation takes time $O(m^2)$.