Computing discrete logarithms in $\mathbb{F}_{3^{6\cdot137}}$ and $\mathbb{F}_{3^{6\cdot163}}$ using Magma

Gora Adj[1], Alfred Menezes[2], Thomaz Oliveira[1], and Francisco Rodríguez-Henríquez[1]

[1] Computer Science Department, CINVESTAV-IPN
gora.adj@gmail.com, thomaz.figueiredo@gmail.com,
francisco@cs.cinvestav.mx
[2] Department of Combinatorics & Optimization, University of Waterloo
ajmeneze@uwaterloo.ca

Abstract. We show that a Magma implementation of Joux's $L[1/4 + o(1)]$ algorithm can be used to compute discrete logarithms in the 1303-bit finite field $\mathbb{F}_{3^{6\cdot137}}$ and the 1551-bit finite field $\mathbb{F}_{3^{6\cdot163}}$ with very modest computational resources. Our $\mathbb{F}_{3^{6\cdot137}}$ implementation was the first to illustrate the effectiveness of Joux's algorithm for computing discrete logarithms in small-characteristic finite fields that are not Kummer or twisted-Kummer extensions.

1 Introduction

Let $\mathbb{F}_Q$ denote the finite field of order Q. The discrete logarithm problem (DLP) in $\mathbb{F}_Q$ is that of determining, given a generator g of $\mathbb{F}_Q^*$ and an element $h \in \mathbb{F}_Q^*$, the integer $x \in [0, Q-2]$ satisfying $h = g^x$. In the remainder of the paper, we shall assume that the characteristic of $\mathbb{F}_Q$ is 2 or 3.

Until recently, the fastest general-purpose algorithm known for solving the DLP in $\mathbb{F}_Q$ was Coppersmith's 1984 index-calculus algorithm [9] with a running time of $L_Q[\frac{1}{3}, (32/9)^{1/3}] \approx L_Q[\frac{1}{3}, 1.526]$, where as usual $L_Q[\alpha, c]$ with $0 < \alpha < 1$ and $c > 0$ denotes the expression $\exp\left((c + o(1))(\log Q)^{\alpha}(\log\log Q)^{1-\alpha}\right)$ that is subexponential in $\log Q$. In February 2013, Joux [23] presented a new DLP algorithm with a running time of $L_Q[\frac{1}{4} + o(1), c]$ (for some undetermined c) when $Q = q^{2n}$ and $q \approx n$. Shortly thereafter, Barbulescu, Gaudry, Joux and Thomé [4] presented an algorithm with *quasi-polynomial* running time $(\log Q)^{O(\log\log Q)}$ when $Q = q^{2n}$ with $q \approx n$.

These dramatic developments were accompanied by some striking computational results. For example, Göloğlu et al. [16] computed logarithms in $\mathbb{F}_{2^{8\cdot3\cdot255}} = \mathbb{F}_{2^{6120}}$ in only 750 CPU hours, and Joux [24] computed logarithms in $\mathbb{F}_{2^{8\cdot3\cdot257}} = \mathbb{F}_{2^{6168}}$ in only 550 CPU hours. The small computational effort expended in these experiments depends crucially on the special nature of the fields $\mathbb{F}_{2^{6120}}$ and $\mathbb{F}_{2^{6168}}$ — namely that $\mathbb{F}_{2^{6120}}$ is a degree-255 extension of $\mathbb{F}_{2^{8\cdot3}}$ with $255 = 2^8 - 1$ (a Kummer extension), and $\mathbb{F}_{2^{6168}}$ is a degree-257 extension of $\mathbb{F}_{2^{8\cdot3}}$ with $257 = 2^8 + 1$ (a twisted Kummer extension). Adj et al. [1] presented a concrete analysis of the new algorithms and demonstrated that logarithms in $\mathbb{F}_{3^{6\cdot 509}}$ can be computed in approximately 2^{82} time, which is considerably less than the 2^{128} time required by Coppersmith's algorithm. Adj et al. [2] also showed how a modification of the new algorithms by Granger and Zumbrägel [21] can be used to compute logarithms in $\mathbb{F}_{3^{6\cdot 1429}}$ in approximately 2^{96} time, which is considerably less than the 2^{192} time required by Coppersmith's algorithm. Unlike the aforementioned experimental results, the analysis by Adj et al. does not exploit any special properties of the fields $\mathbb{F}_{3^{6\cdot 509}}$ and $\mathbb{F}_{3^{6\cdot 1429}}$.

The purpose of this paper is to demonstrate that, with modest computational resources, the new algorithms can be used to solve instances of the discrete logarithm problem that remain beyond the reach of classical algorithms. The first target field is the 1303-bit field $\mathbb{F}_{3^{6\cdot 137}}$; this field does not enjoy any Kummer-like properties. More precisely, we are interested in solving the discrete logarithm problem in the order-r subgroup $\mathcal{G}$ of $\mathbb{F}_{3^{6\cdot 137}}^*$, where $r = (3^{137} - 3^{69} + 1)/7011427850892440647$ is a 155-bit prime. The discrete logarithm problem in this group is of cryptographic interest because the values of the bilinear pairing derived from the supersingular elliptic curve $E : y^2 = x^3 - x + 1$ over $\mathbb{F}_{3^{137}}$ lie in $\mathcal{G}$.[1] Consequently, if logarithms in $\mathcal{G}$ can be computed efficiently then the associated bilinear pairing is rendered cryptographically insecure. Note that since r is a 155-bit prime, Pollard's rho algorithm [29] for computing logarithms in $\mathcal{G}$ is infeasible. Moreover, recent work on computing logarithms in the 809-bit field $\mathbb{F}_{2^{809}}$ [3] suggests that Coppersmith's algorithm is infeasible for computing logarithms in $\mathcal{G}$, whereas recent work on computing logarithms in the 923-bit field $\mathbb{F}_{3^{6\cdot 97}}$ [22] (see also [30]) indicates that computing logarithms in $\mathcal{G}$ using the Joux-Lercier algorithm [25] would be a formidable challenge. In contrast, we show that Joux's algorithm can be used to compute logarithms in $\mathcal{G}$ in just a few days using a small number of CPUs; more precisely, our computation consumed a total of 888 CPU hours. The computational effort expended in our experiment is relatively small, despite the fact that our implementation was done using the computer algebra system Magma V2.20-2 [27] and is far from optimal.

The second target field is the 1551-bit field $\mathbb{F}_{3^{6\cdot 163}}$; this field does not enjoy any Kummer-like properties. More precisely, we are interested in solving the discrete logarithm problem in the order-r subgroup $\mathcal{G}$ of $\mathbb{F}_{3^{6\cdot 163}}^*$, where $r = 3^{163} + 3^{82} + 1$ is a 259-bit prime. The discrete logarithm problem in this group is of cryptographic interest because the values of the bilinear pairing derived from the supersingular elliptic curve $E : y^2 = x^3 - x - 1$ over $\mathbb{F}_{3^{163}}$ lie in $\mathcal{G}$. This bilinear pairing was first considered by Boneh, Lynn and Shacham in their landmark paper on short signature schemes [8]; see also [20]. Furthermore, the bilinear pairing derived from the quadratic twist of E was one of the pairings implemented by Galbraith, Harrison and Soldera [14]. Again, we show that Joux's algorithm can be used to compute logarithms in $\mathcal{G}$ in just a few days using a small number of CPUs; our computation used 1201 CPU hours.

[1] We note that the supersingular elliptic curves $y^2 = x^3 - x \pm 1$ over $\mathbb{F}_{3^n}$ have embedding degree 6 and were proposed for cryptographic use in several early papers on pairing-based cryptography [8, 5, 14, 19].

After we had completed the $\mathbb{F}_{3^{6\cdot 137}}$ discrete logarithm computation, Granger, Kleinjung and Zumbrägel [18] presented several practical improvements and refinements of Joux's algorithm. These improvements allowed them to compute logarithms in the 4404-bit field $\mathbb{F}_{2^{12\cdot 367}}$ in approximately 52,240 CPU hours, and drastically lowered the estimated time to compute logarithms in the 4892-bit field $\mathbb{F}_{2^{4\cdot 1223}}$ to 2^{59} modular multiplications. More recently, Joux and Pierrot [26] presented a more efficient algorithm for computing logarithms of factor base elements. The new algorithm was used to compute logarithms in the 3796-bit characteristic-three field $\mathbb{F}_{3^{5\cdot 479}}$ in less than 8600 CPU hours.

The remainder of the paper is organized as follows. In §2, we review Joux's algorithm for computing logarithms in $\mathbb{F}_{q^{3n}}$; the algorithm uses the polynomial representation (selection of h_0 and h_1) of Granger and Zumbrägel [21]. Our experimental results with computing logarithms in $\mathbb{F}_{3^{6\cdot 137}}$ and $\mathbb{F}_{3^{6\cdot 163}}$ are reported in §3 and §4, respectively. In §5, we use the aforementioned improvements from [18] and [26] to derive improved upper bounds for discrete logarithm computations in $\mathbb{F}_{3^{6\cdot 509}}$ and $\mathbb{F}_{3^{6\cdot 1429}}$. We draw our conclusions in §6.

2 Joux's $L[1/4 + o(1)]$ algorithm

Let $\mathbb{F}_{q^{3n}}$ be a finite field where $n \leq 2q+1$.[2] The elements of $\mathbb{F}_{q^{3n}}$ are represented as polynomials of degree at most $n-1$ over $\mathbb{F}_{q^3}$. Let $N = q^{3n}-1$, and let r be a prime divisor of N. In this paper, we are interested in the discrete logarithm problem in the order-r subgroup of $\mathbb{F}_{q^{3n}}^*$. More precisely, we are given two elements α, β of order r in $\mathbb{F}_{q^{3n}}^*$ and we wish to find $\log_\alpha \beta$. Let g be an element of order N in $\mathbb{F}_{q^{3n}}^*$. Then $\log_\alpha \beta = (\log_g \beta)/(\log_g \alpha) \bmod r$. Thus, in the remainder of this section we will assume that we need to compute $\log_g h \bmod r$, where h is an element of order r in $\mathbb{F}_{q^{3n}}^*$.

The algorithm proceeds by first finding the logarithms (mod r) of all degree-one elements in $\mathbb{F}_{q^{3n}}$ (§2.2). Then, in the *descent stage*, $\log_g h$ is expressed as a linear combination of logarithms of degree-one elements. The descent stage proceeds in several steps, each expressing the logarithm of a degree-D element as a linear combination of the logarithms of elements of degree $\leq m$ for some $m < D$. Four descent methods are employed; these are described in §2.3–§2.6.

Notation. $N_{q^3}(m,n)$ denotes the number of monic m-smooth degree-n polynomials in $\mathbb{F}_{q^3}[X]$, $A_{q^3}(m,n)$ denotes the average number of distinct monic irreducible factors among all monic m-smooth degree-n polynomials in $\mathbb{F}_{q^3}[X]$, and $S_{q^3}(m,d)$ denotes the cost of testing m-smoothness of a degree-d polynomial in $\mathbb{F}_{q^3}[X]$. Formulas for $N_{q^3}(m,n)$, $A_{q^3}(m,n)$ and $S_{q^3}(m,n)$ are given in [1]. For $\gamma \in \mathbb{F}_{q^3}$, $\overline{\gamma}$ denotes the element γ^{q^2}. For $P \in \mathbb{F}_{q^3}[X]$, $\overline{P}$ denotes the polynomial obtained by raising each coefficient of P to the power q^2. The cost of an integer addition modulo r is denoted by A_r, and the cost of a multiplication in $\mathbb{F}_{q^3}$

[2] More generally, one could consider fields $\mathbb{F}_{q^{kn}}$ where $n \leq 2q+1$. We focus on the case $k=3$ since our target fields are $\mathbb{F}_{3^{6n}}$ with $n \in \{137, 163\}$, which we will embed in $\mathbb{F}_{(3^4)^{3\cdot n}}$.

is denoted by M_{q^3}. The projective general linear group of degree 2 over $\mathbb{F}_q$ is denoted $\mathrm{PGL}_2(\mathbb{F}_q)$. $\mathcal{P}_q$ is a set of distinct representatives of the left cosets of $\mathrm{PGL}_2(\mathbb{F}_q)$ in $\mathrm{PGL}_2(\mathbb{F}_{q^3})$; note that $\#\mathcal{P}_q = q^6 + q^4 + q^2$. A matrix $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right) \in \mathcal{P}_q$ is identified with the quadruple (a, b, c, d).

2.1 Setup. Select polynomials $h_0, h_1 \in \mathbb{F}_{q^3}[X]$ of small degree so that

$$X \cdot h_1(X^q) - h_0(X^q) \tag{1}$$

has an irreducible factor I_X of degree n in $\mathbb{F}_{q^3}[X]$; we will henceforth assume that $\max(\deg h_0, \deg h_1) = 2$, whence $n \leq 2q + 1$. Note that

$$X \equiv \frac{h_0(X^q)}{h_1(X^q)} \equiv \left(\frac{\overline{h}_0(X)}{\overline{h}_1(X)}\right)^q \pmod{I_X}. \tag{2}$$

The field $\mathbb{F}_{q^{3n}}$ is represented as $\mathbb{F}_{q^{3n}} = \mathbb{F}_{q^3}[X]/(I_X)$ and the elements of $\mathbb{F}_{q^{3n}}$ are represented as polynomials in $\mathbb{F}_{q^3}[X]$ of degree at most $n - 1$. Let g be a generator of $\mathbb{F}^*_{q^{3n}}$.

2.2 Finding logarithms of linear polynomials. Let $\mathcal{B}_1 = \{X + a \mid a \in \mathbb{F}_{q^3}\}$, and note that $\#\mathcal{B}_1 = q^3$. To compute the logarithms of $\mathcal{B}_1$-elements, we first generate linear relations of these logarithms. Let $(a, b, c, d) \in \mathcal{P}_q$. Substituting $Y \mapsto (aX + b)/(cX + d)$ into the systematic equation

$$Y^q - Y = \prod_{\alpha \in \mathbb{F}_q} (Y - \alpha) \tag{3}$$

and using (2) yields

$$\Big((aX + b)(\overline{c}\overline{h}_0 + \overline{d}\,\overline{h}_1) - (\overline{a}\overline{h}_0 + \overline{b}\,\overline{h}_1)(cX + d)\Big)^q \tag{4}$$
$$\equiv \overline{h}_1^q \cdot (cX + d) \cdot \prod_{\alpha \in \mathbb{F}_q} [(a - \alpha c)X + (b - \alpha d)].$$

If the polynomial on the left side of (4) is 1-smooth, then taking logarithms (mod r) of both sides of (4) yields a linear relation of the logarithms of $\mathcal{B}_1$-elements and the logarithm of $\overline{h}_1$. The probability that the left side of (4) is 1-smooth is $N_{q^3}(1, 3)/q^9 \approx \frac{1}{6}$. Thus, after approximately $6q^3$ trials one expects to obtain q^3 relations. The cost of the relation generation stage is $6q^3 \cdot S_{q^3}(1, 3)$. The logarithms can then be obtained by using Wiedemann's algorithm for solving sparse systems of linear equations [31, 10]. The expected cost of the linear algebra is $q^7 \cdot A_r$ since each equation has approximately q nonzero terms.

2.3 Continued-fractions descent. Recall that we wish to compute $\log_g h \bmod r$, where $h \in \mathbb{F}_{q^{3n}} = \mathbb{F}_{q^3}[X]/(I_X)$ has order r. We will henceforth assume that $\deg h = n-1$. The descent stage begins by multiplying h by a random power of g. The extended Euclidean algorithm is used to express the resulting field element h' in the form $h' = w_1/w_2$ where $\deg w_1, \deg w_2 \approx n/2$ [7]; for simplicity, we shall assume that n is odd and $\deg w_1 = \deg w_2 = (n-1)/2$. This process is repeated until both w_1 and w_2 are m-smooth for some chosen $m < (n-1)/2$. This gives $\log_g h'$ as a linear combination of logarithms of polynomials of degree at most m. The expected cost of this continued-fractions descent step is approximately

$$\left(\frac{(q^3)^{(n-1)/2}}{N_{q^3}(m,(n-1)/2)}\right)^2 \cdot S_{q^3}(m,(n-1)/2). \tag{5}$$

The expected number of distinct irreducible factors of w_1 and w_2 is $2A_{q^3}(m,(n-1)/2)$. In the concrete analysis, we shall assume that each of these irreducible factors has degree exactly m. The logarithm of each of these degree-m polynomials is then expressed as a linear combination of logarithms of smaller degree polynomials using one of the descent methods described in §2.4, §2.5 and §2.6.

2.4 Classical descent. Let p be the characteristic of $\mathbb{F}_q$, and let $q = p^\ell$. Let $s \in [0,\ell]$, and let $R \in \mathbb{F}_{q^3}[X,Y]$. Then it can be seen that

$$\left[R(X,(\overline{h}_0/\overline{h}_1)^{p^{\ell-s}})\right]^{p^s} \equiv R'(X^{p^s},X) \pmod{I_X} \tag{6}$$

where R' is obtained from R by raising all its coefficients to the power p^s. Let $\mu = \deg_Y R$. Then multiplying both sides of (6) by $\overline{h}_1^{q\mu}$ gives

$$\left[\overline{h}_1^{p^{\ell-s}\cdot\mu} \cdot R(X,(\overline{h}_0/\overline{h}_1)^{p^{\ell-s}})\right]^{p^s} \equiv \overline{h}_1^{q\mu} \cdot R'(X^{p^s},X) \pmod{I_X}. \tag{7}$$

Let $Q \in \mathbb{F}_{q^3}[X]$ with $\deg Q = D$, and let $m < D$. In the Joux-Lercier descent method [25], as modified by Göloğlu et al. [15], one selects $s \in [0,\ell]$ and searches for a polynomial $R \in \mathbb{F}_{q^3}[X,Y]$ such that (i) $Q \mid R_2$ where $R_2 = R'(X^{p^s},X)$; (ii) $\deg R_1$ and $\deg R_2/Q$ are appropriately balanced where $R_1 = \overline{h}_1^{p^{\ell-s}\mu} R(X,(\overline{h}_0/\overline{h}_1)^{p^{\ell-s}})$; and (iii) both R_1 and R_2/Q are m-smooth. Taking logarithms of both sides of (7) then gives an expression for $\log_g Q$ in terms of the logarithms of polynomials of degree at most m.

A family of polynomials R satisfying (i) and (ii) can be constructed by finding a basis $\{(u_1,u_2),(v_1,v_2)\}$ of the lattice

$$L_Q = \{(w_1,w_2) \in \mathbb{F}_{q^3}[X] \times \mathbb{F}_{q^3}[X] \ : \ Q \mid (w_1(X) - w_2(X)X^{p^s})\}$$

where $\deg u_1$, $\deg u_2$, $\deg v_1$, $\deg v_2 \approx D/2$. By writing $(w_1,w_2) = a(u_1,u_2) + b(v_1,v_2) = (au_1+bv_1, au_2+bv_2)$ with $a \in \mathbb{F}_{q^3}[X]$ monic of degree δ and $b \in \mathbb{F}_{q^3}[X]$ of degree $\delta - 1$, the points (w_1,w_2) in L_Q can be sampled to obtain polynomials $R(X,Y) = w_1''(Y) - w_2''(Y)X$ satisfying (i) and (ii) where w'' is obtained from w by raising all its coefficients to the power p^{-s}. The number of lattice points to consider is therefore $(q^3)^{2\delta}$. We have $\deg w_1, \deg w_2 \approx D/2 + \delta$, so $\deg R_1 = t_1 \approx 2(D/2+\delta)p^{\ell-s} + 1$ and $\deg R_2 = t_2 \approx (D/2+\delta) + p^s$. In order to ensure that there are sufficiently many such lattice points to generate a polynomial R for which both R_1 and R_2/Q are m-smooth, the parameters s and δ must be selected so that

$$q^{6\delta} \gg \frac{q^{3t_1}}{N_{q^3}(m,t_1)} \cdot \frac{q^{3(t_2-D)}}{N_{q^3}(m,t_2-D)}. \tag{8}$$

Ignoring the time to compute a balanced basis of L_Q, the expected cost of finding a polynomial R satisfying (i)–(iii) is

$$\frac{q^{3t_1}}{N_{q^3}(m,t_1)} \cdot \frac{q^{3(t_2-D)}}{N_{q^3}(m,t_2-D)} \cdot \min(S_{q^3}(m,t_1), S_{q^3}(m,t_2-D)). \tag{9}$$

The expected number of distinct irreducible factors of R_1 and R_2/Q is $A_{q^3}(m,t_1) + A_{q^3}(m,t_2-D)$.

2.5 Gröbner bases descent. Let $Q \in \mathbb{F}_{q^3}[X]$ with $\deg Q = D$. Let $m = \lceil (D+1)/2 \rceil$, and suppose that $3m < n$. In Joux's new descent method [23, §5.3], one finds degree-m polynomials $k_1, k_2 \in \mathbb{F}_{q^3}[X]$ such that $G = k_1\widetilde{k}_2 - \widetilde{k}_1 k_2 = QR$, where $\widetilde{k}_1 = \overline{h}_1^m \overline{k}_1(\overline{h}_0/\overline{h}_1)$ and $\widetilde{k}_2 = \overline{h}_1^m \overline{k}_2(\overline{h}_0/\overline{h}_1)$, and $R \in \mathbb{F}_{q^3}[X]$. Note that $\deg R = 3m - D$. If R is m-smooth, then we obtain a linear relationship between $\log_g Q$ and logs of degree-m polynomials (see [2, §3.7]):

$$\overline{h}_1^{mq} \cdot k_2 \cdot \prod_{\alpha \in \mathbb{F}_q} (k_1 - \alpha k_2) \equiv (Q(X)R(X))^q \pmod{I_X}. \tag{10}$$

To determine (k_1, k_2, R) that satisfy

$$k_1\widetilde{k}_2 - \widetilde{k}_1 k_2 = QR, \tag{11}$$

one can transform (11) into a system of multivariate quadratic equations over $\mathbb{F}_q$. Specifically, each coefficient of k_1, k_2 and R is written using three variables over $\mathbb{F}_q$. The coefficients of $\widetilde{k}_1$ and $\widetilde{k}_2$ can then be written in terms of the coefficients of k_1 and k_2. Hence, equating coefficients of X^i of both sides of (11) yields $3m+1$ quadratic equations. Equating $\mathbb{F}_q$-components of these equations then yields $9m+3$ bilinear equations in $15m - 3D + 9$ variables over $\mathbb{F}_q$. This system of equations can be solved by finding a Gröbner basis for the ideal it generates. Finally, solutions (k_1, k_2, R) are tested until one is found for which R is m-smooth. This yields an expression for $\log_g Q$ in terms of the logarithms of approximately $q + 1 + A_{q^3}(m, 3m-D)$ polynomials of degree (at most) m; in the concrete analysis we shall assume that each of the polynomials has degree exactly m.

2.6 2-to-1 descent. The Gröbner bases descent methodology of §2.5 can be employed in the case $(D, m) = (2, 1)$. However, as also reported by Joux in his $\mathbb{F}_{2^{6168}}$ discrete log computation [24], we found the descent to be successful for only about 50% of all irreducible quadratic polynomials. Despite this, some strategies can be used to increase this percentage.

Let $Q(X) = X^2 + uX + v \in \mathbb{F}_{q^3}[X]$ be an irreducible quadratic polynomial for which the Gröbner bases descent method failed.

Strategy 1. Introduced by Joux [24] and Göloğlu et al. [16], this strategy is based on the systematic equation derived from $Y^{q'} - Y$ where $q' < q$ and $\mathbb{F}_{q'}$ is a proper subfield of $\mathbb{F}_{q^3}$ instead of the systematic equation (3) derived from $Y^q - Y$. Let p be the characteristic of $\mathbb{F}_q$, and let $q = p^\ell$, $q' = p^{\ell'}$, and $s = \ell - \ell'$. Then $q = p^s \cdot q'$. Now, one searches for $a, b, c, d \in \mathbb{F}_{q^3}$ such that

$$G = (aX + b)(\overline{c}\overline{h}_0 + \overline{d}\,\overline{h}_1)^{p^s} - (\overline{a}\overline{h}_0 + \overline{b}\,\overline{h}_1)^{p^s}(cX + d) = QR$$

with $R \in \mathbb{F}_{q^3}[X]$. Note that $\deg R = 2p^s - 1$.[3] If R is 1-smooth, then we obtain a linear relationship between $\log_g Q$ and logs of linear polynomials since

$$G^q \equiv \overline{h}_1^{p^s q} \cdot (cX + d)^{p^s} \cdot \prod_{\alpha \in \mathbb{F}_{q'}} \Big((aX + b)^{p^s} - \alpha (cX + d)^{p^s} \Big) \pmod{I_X},$$

as can be seen by making the substitution $Y \mapsto (aX + b)^{p^s}/(cX + d)^{p^s}$ into the systematic equation derived from $Y^{q'} - Y$.

Unfortunately, in all instances we considered, the polynomial R never factors completely into linear polynomials. However, it hopefully factors into a quadratic polynomial Q' and $2p^s - 3$ linear polynomials, thereby yielding a relation between Q and another quadratic which has a roughly 50% chance of descending using Gröbner bases descent. Combined with the latter, this strategy descends about 95% of all irreducible quadratic polynomials in the fields $\mathbb{F}_{3^{6\cdot 137}}$ and $\mathbb{F}_{3^{6\cdot 163}}$.

Strategy 2. We have

$$\begin{aligned} \overline{h}_1^{2q} Q(X) \equiv \overline{h}_1^{2q} Q((\overline{h}_0/\overline{h}_1)^q) &= \overline{h}_0^{2q} + u\overline{h}_0^q \overline{h}_1^q + v\overline{h}_1^{2q} \\ &= (\overline{h}_0^2 + \overline{u}\overline{h}_0\overline{h}_1 + \overline{v}\,\overline{h}_1^2)^q \pmod{I_X}. \end{aligned} \tag{12}$$

It can be seen that the degree-4 polynomial $f_Q(X) = \overline{h}_0^2 + \overline{u}\overline{h}_0\overline{h}_1 + \overline{v}\,\overline{h}_1^2$ is either a product of two irreducible quadratics or itself irreducible. In the former case, we apply the standard Gröbner bases descent method to the two irreducible quadratics. If both descents are successful, then we have succeeded in descending the original Q.

The strategies are combined in the following manner. For an irreducible quadratic $Q \in \mathbb{F}_{q^3}[X]$, we first check if the Gröbner bases descent is successful. If the descent fails, we apply Strategy 2 to Q. In the case where f_Q factors

[3] For our $\mathbb{F}_{3^{6\cdot 137}}$ and $\mathbb{F}_{3^{6\cdot 163}}$ computations, we have $q = 3^4$ and used $q' = 3^3$, so $s = 1$ and $\deg R = 5$.

into two irreducible quadratics, and at least one of them fails to descent with Gröbner bases descent, we apply Strategy 1 to Q. If Strategy 1 fails on Q, we apply it to the two quadratic factors of f_Q. In the case where f_Q is irreducible, we apply Strategy 1 to Q.

If none of the attempts succeed, we declare Q to be "bad", and avoid it in the higher-degree descent steps by repeating a step until all the quadratics encountered are "good". In our experiments with $\mathbb{F}_{3^{6\cdot 137}}$ and $\mathbb{F}_{3^{6\cdot 163}}$, we observed that approximately 97.2% of all irreducible quadratic polynomials Q were "good".

To see that this percentage is sufficient to complete the descent phase in these two fields, consider a 3-to-2 descent step where the number of resulting irreducible quadratic polynomials is 42 on average (cf. equation (10)). Then the probability of descending a degree-3 polynomial after finding one useful solution (k_1, k_2, R) in Gröbner bases descent is $0.972^{42} \approx 0.3$. Therefore, after at most four trials we expect to successfully descend a degree-3 polynomial. Since the expected number of distinct solutions of (11) is approximately q^3 (according to equation (10) in [18]), one can afford this many trials.

3 Computing discrete logarithms in $\mathbb{F}_{3^{6\cdot 137}}$

The supersingular elliptic curve $E : y^2 = x^3 - x + 1$ has order $\#E(\mathbb{F}_{3^{137}}) = cr$, where

$$c = 7 \cdot 4111 \cdot 5729341 \cdot 42526171$$

and

$$r = (3^{137} - 3^{69} + 1)/c = 33098280119090191028775580055082175056428495623$$

is a 155-bit prime. The Weil and Tate pairing attacks [28, 13] efficiently reduce the discrete logarithm problem in the order-r subgroup $\mathcal{E}$ of $E(\mathbb{F}_{3^{137}})$ to the discrete logarithm problem in the order-r subgroup $\mathcal{G}$ of $\mathbb{F}^*_{3^{6\cdot 137}}$.

Our approach to computing logarithms in $\mathcal{G}$ is to use Joux's algorithm to compute logarithms in the quadratic extension $\mathbb{F}_{3^{12\cdot 137}}$ of $\mathbb{F}_{3^{6\cdot 137}}$ (so $q = 3^4$ and $n = 137$ in the notation of §2). More precisely, we are given two elements α, β of order r in $\mathbb{F}^*_{3^{12\cdot 137}}$ and we wish to find $\log_\alpha \beta$. Let g be a generator of $\mathbb{F}^*_{3^{12\cdot 137}}$. Then $\log_\alpha \beta = (\log_g \beta)/(\log_g \alpha) \bmod r$. Thus, in the remainder of the section we will assume that we need to compute $\log_g h \bmod r$, where h is an element of order r in $\mathbb{F}^*_{3^{12\cdot 137}}$.

The DLP instance we solved is described in §3.1. The concrete estimates from §2 for solving the DLP instances are given in §3.2. These estimates are only upper bounds on the running time of the algorithm. Nevertheless, they provide convincing evidence for the feasibility of the discrete logarithm computations. Our experimental results are presented in §3.3.

3.1 Problem instance. Let N denote the order of $\mathbb{F}^*_{3^{12\cdot 137}}$. Using the tables from the Cunningham Project [11], we determined that the factorization of N

is $N = p_1^4 \cdot \prod_{i=2}^{31} p_i$, where the p_i are the following primes (and $r = p_{25}$):

$p_1 = 2$ $p_2 = 5$ $p_3 = 7$ $p_4 = 13$ $p_5 = 73$ $p_6 = 823$ $p_7 = 4111$ $p_8 = 4933$
$p_9 = 236737$ $p_{10} = 344693$ $p_{11} = 2115829$ $p_{12} = 5729341$ $p_{13} = 42526171$
$p_{14} = 217629707$ $p_{15} = 634432753$ $p_{16} = 685934341$ $p_{17} = 82093596209179$
$p_{18} = 4354414202063707$ $p_{19} = 18329390240606021$ $p_{20} = 46249052722878623693$
$p_{21} = 201820452878622271249$ $p_{22} = 113938829134880224954142892526477$
$p_{23} = 51854546646328186791017417700430486396513$
$p_{24} = 273537065683369412556888964042827802376371$
$p_{25} = 3309828011909019102877558005508217505642849562\,3$
$p_{26} = 706712258201940254667826642673008768387229115048379$
$p_{27} = 1080818097738399951882568004991415436843930354503505 51$
$p_{28} = 9132197459566276133922227162624796611612645016288069258858718395223 7$
$p_{29} = 3948753114977348953209699629336837018295752625798857387703105447724 9$
$\quad 393549$
$p_{30} = 4018986002238485004425485479656118254755307273073882386698630080761 3$
$\quad 29207749418522920289$
$p_{31} = 1906432315382527207280368587080395562283428652313903740358075231082 2$
$\quad 78966446469840637369426240662274068981321133662265931584644197 13.$

We chose $\mathbb{F}_{3^4} = \mathbb{F}_3[U]/(U^4 + U^2 + 2)$ and $\mathbb{F}_{3^{12}} = \mathbb{F}_{3^4}[V]/(V^3 + V + U^2 + U)$, and selected $h_0(X) = V^{326196}X^2 + V^{35305}X + V^{204091} \in \mathbb{F}_{3^{12}}[X]$ and $h_1 = 1$. Then $I_X \in \mathbb{F}_{3^{12}}[X]$ is the degree-137 monic irreducible factor of $X - h_0(X^{3^4})$; the other irreducible factor has degree 25.

We chose the generator $g = X + V^{113713}$ of $\mathbb{F}^*_{3^{12 \cdot 137}}$. To generate an order-$r$ discrete logarithm challenge h, we computed

$$h' = \sum_{i=0}^{136} \left(V^{\lfloor \pi \cdot (3^{12})^{i+1} \rfloor \bmod 3^{12}} \right) X^i$$

and then set $h = (h')^{N/r}$. The discrete logarithm $\log_g h \bmod r$ was found to be

$$x = 27339619076975093920245515973214186963025656559.$$

This can be verified by checking that $h = (g^{N/r})^y$, where $y = x \cdot (N/r)^{-1} \bmod r$ (cf. Appendix A).

3.2 Estimates. The factor base $\mathcal{B}_1$ has size $3^{12} \approx 2^{19}$. The cost of the relation generation is approximately $2^{29.2} M_{q^3}$, whereas the cost of the linear algebra is approximately $2^{44.4} A_r$. Figure 1 shows the estimated running times for the descent stage. Further information about the parameter choices are provided below.

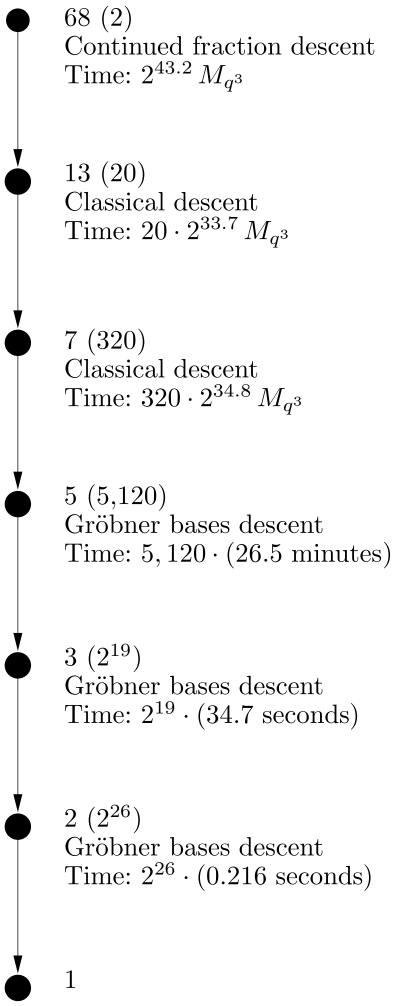
68 (2)
Continued fraction descent
Time: $2^{43.2}\,M_{q^3}$
13 (20)
Classical descent
Time: $20 \cdot 2^{33.7}\,M_{q^3}$
7 (320)
Classical descent
Time: $320 \cdot 2^{34.8}\,M_{q^3}$
5 (5,120)
Gröbner bases descent
Time: $5,120 \cdot$ (26.5 minutes)
3 (2^{19})
Gröbner bases descent
Time: $2^{19} \cdot$ (34.7 seconds)
2 (2^{26})
Gröbner bases descent
Time: $2^{26} \cdot$ (0.216 seconds)
1

Fig. 1. A typical path of the descent tree for computing an individual logarithm in $\mathbb{F}_{3^{12\cdot 137}}$ ($q = 3^4$). The numbers in parentheses next to each node are the expected number of nodes at that level. 'Time' is the expected time to generate all nodes at a level.

1. For the continued-fractions descent stage, we selected $m = 13$. The expected cost of this descent is $2^{43.2}M_{q^3}$, and the expected number of irreducible factors of degree (at most) 13 obtained is $2A_{3^{12}}(68, 13) \approx 20$.
2. Two classical descent stages are employed. In the first stage, we have $D = 13$ and select $m = 7$, $s = 3$, $\delta = 1$, which yield $t_1 = 43$ and $t_2 = 34$. The expected cost of the descent for each of the 20 degree-13 polynomials is approximately $2^{33.7}M_{q^3}$. The expected total number of distinct irreducible polynomials of degree (at most) 7 obtained is approximately 320.
   In the second classical descent stage, we have $D = 7$ and select $m = 5$, $s = 3$, $\delta = 1$, which yield $t_1 = 25$ and $t_2 = 31$. The expected cost of the descent for each of the 320 degree-7 polynomials is approximately $2^{34.8}M_{q^3}$. The expected total number of distinct irreducible polynomials of degree (at most) 5 obtained is approximately $5, 120$.
3. Our implementation of the Gröbner bases descent stage used Magma's implementation of Faugére's F4 algorithm [12] and took 26.5 minutes on average for a 5-to-3 descent, 34.7 seconds for a 3-to-2 descent, and 0.216 seconds for a 2-to-1 descent. The total expected running time for each of these stages is 94, 211 and 168 days, respectively.

Since all the descent stages can be effectively parallelized, our estimates suggest that a discrete logarithm can be computed in a week or so given a few dozen processors. In fact (and as confirmed by our experimental results), the actual running time is expected to be significantly less than the estimated running time since the estimates are quite conservative; for example, our estimates for the number of branches in a descent step assumes that each distinct irreducible polynomial has degree exactly m, whereas in practice many of these polynomials will have degree significantly less than m.

3.3 Experimental results. Our experiments were run on an Intel i7-2600K 3.40 GHz machine (Sandy Bridge), and on an Intel i7-4700MQ 2.40 GHz machine (Haswell).

Relation generation took 1.05 CPU hours (Sandy Bridge, 1 core). The resulting sparse linear system of linear equation was solved using Magma's multithreaded parallel version of the Lanczos algorithm; the computation took 556.8 CPU hours (Sandy Bridge, 4 cores).

In the continued-fractions descent stage, the first degree-68 polynomial yielded 9 irreducible factors of degrees 12, 12, 11, 10, 8, 6, 6, 2, 1, and the second degree-68 polynomial yielded 11 irreducible factors of degrees 13, 12, 10, 10, 7, 6, 5, 2, 1, 1, 1. The computation took 22 CPU hours (Haswell, 4 cores).

Classical descent was used on the 9 polynomials of degree ≥ 8 to obtain polynomials of degree ≤ 7, and then on the 23 polynomials of degree 7 and 23 polynomials of degree 6 to obtain polynomials of degree ≤ 5. These computations took 80 CPU hours (Haswell, 4 cores).

Finally, we used 5-to-3, 4-to-3, 3-to-2 and 2-to-1 Gröbner bases descent procedures. The average time for a 4-to-3 descent was 33.8 seconds; the other average times are given in Figure 1. In total, we performed 233 5-to-3 descents, 174 4-to-3 descents, and 11573 3-to-2 descents. These computations took 115.2 CPU hours, 1.5 CPU hours, and 111.2 CPU hours, respectively (Haswell, 4 cores). We also performed 493537 2-to-1 descents; their running times are incorporated into the running times for the higher-level descents.

4 Computing discrete logarithms in $\mathbb{F}_{3^{6\cdot 163}}$

The supersingular elliptic curve $E : y^2 = x^3 - x - 1$ has order $\#E(\mathbb{F}_{3^{163}}) = 3^{163} + 3^{82} + 1 = r$, where r is the following 259-bit prime:

$$
\begin{aligned}
r = {} & 5898811514266587408542277255807363488506406322973734140917909955057 56\\
& 623268837.
\end{aligned}
$$

The Weil and Tate pairing attacks [28, 13] efficiently reduce the discrete logarithm problem in the order-r group $\mathcal{E} = E(\mathbb{F}_{3^{163}})$ to the discrete logarithm problem in the order-r subgroup $\mathcal{G}$ of $\mathbb{F}^*_{3^{6\cdot 163}}$.

As in §3, we will compute logarithms in $\mathcal{G}$ by using Joux's algorithm to compute logarithms in the quadratic extension $\mathbb{F}_{3^{12\cdot 163}}$ of $\mathbb{F}_{3^{6\cdot 163}}$ (so $q = 3^4$ and $n = 163$ in the notation of §2). We will compute $\log_g h \bmod r$, where g is a generator of $\mathbb{F}^*_{3^{12\cdot 163}}$ and h is an element of order r in $\mathbb{F}^*_{3^{12\cdot 163}}$.

4.1 Problem instance. Let N denote the order of $\mathbb{F}^*_{3^{12\cdot 163}}$. Using the tables from the Cunningham Project [11], we partially factored N as $N = C\cdot p_1^4\cdot\prod_{i=2}^{22} p_i$, where the p_i are the following primes (and $r = p_{20}$):

$$
\begin{aligned}
& p_1 = 2 \quad p_2 = 5 \quad p_3 = 7 \quad p_4 = 13 \quad p_5 = 73 \quad p_6 = 653 \quad p_7 = 50857\\
& p_8 = 107581 \quad p_9 = 489001 \quad p_{10} = 105451873 \quad p_{11} = 380998157\\
& p_{12} = 8483499631 \quad p_{13} = 5227348213873 \quad p_{14} = 8882811705390167\\
& p_{15} = 4956470591980320134353 \quad p_{16} = 23210817035829275705929\\
& p_{17} = 350717106095718676799491213620033381468 9659449\\
& p_{18} = 635188514196405741125949952661184862607 2045955243\\
& p_{19} = 8426873591809410583631824651153376412114001048113074106744307110314 8\\
& \qquad 817701717\\
& p_{20} = 5898811514266587408542277255807363488506406322973734140917909955057 5\\
& \qquad 6623268837\\
& p_{21} = 1326290578404372337003402566761812108154043828317726868004518688485 3\\
& \qquad 2620412724278105428771691382890569577153531961762590484982180238880 1\\
& p_{22} = 2487998472767501120519871818305554760112258297437457690889886964157 0\\
& \qquad 0926912242398570439592596492295941044800988653984249495592713645064 3\\
& \qquad 31019158574269,
\end{aligned}
$$

and C is the following 919-bit composite number

$$\begin{aligned}C = {} & 287332203665612050739450194991228343672298354626595155150763295732576 7\\ & 0275216328747773792566523729655097848102113488795698936768394494992621\\ & 2312022819011019340957620502000045691081669475648919901346991751981450\\ & 83115345709455585222288272983378262150437440948615147544541514931 77.\end{aligned}$$

We verified that $\gcd(C, N/C) = 1$ and that C is not divisible by any of the first 10^7 primes. Consequently, if an element g is selected uniformly at random from $\mathbb{F}^*_{3^{12 \cdot 163}}$, and g satisfies $g^{(N-1)/p_i} \neq 1$ for $1 \leq i \leq 22$, then g is a generator with very high probability.[4]

We chose $\mathbb{F}_{3^4} = \mathbb{F}_3[U]/(U^4 + U^2 + 2)$ and $\mathbb{F}_{3^{12}} = \mathbb{F}_{3^4}[V]/(V^3 + V + U^2 + U)$, and selected $h_0(X) = 1$ and

$$h_1(X) = X^2 + V^{530855} \in \mathbb{F}_{3^{12}}[X].$$

Then $I_X \in \mathbb{F}_{3^{12}}[X]$ is the degree-163 irreducible polynomial $X \cdot h_1(X^{3^4}) - 1$:

$$I_X = X^{163} + V^{530855} X + 2.$$

We chose $g = X + V^2$, which we hope is a generator of $\mathbb{F}^*_{3^{12 \cdot 163}}$.

To generate an order-r discrete logarithm challenge h, we computed

$$h' = \sum_{i=0}^{162} \left(V^{\lfloor \pi \cdot (3^{12})^{i+1} \rfloor \bmod 3^{12}} \right) X^i$$

and then set $h = (h')^{N/r}$. The discrete logarithm $\log_g h \bmod r$ was found to be

$$\begin{aligned}x = {} & 4263959514982791937132913919534490007325925542511325256720397843560 5\\ & 4526194343.\end{aligned}$$

This can be verified by checking that $h = (g^{N/r})^y$, where $y = x \cdot (N/r)^{-1} \bmod r$ (cf. Appendix B).

4.2 Experimental results. Our experiments were run on an Intel i7-2600K 3.40 GHz machine (Sandy Bridge), and on an Intel Xeon E5-2650 2.00 GHz machine (Sandy Bridge-EP). The descent strategy was similar to the one used for the $\mathbb{F}_{3^{6 \cdot 137}}$ computation.

Relation generation took 0.84 CPU hours (Sandy Bridge, 1 core). The resulting sparse system of linear equations was solved using Magma's multi-threaded parallel version of the Lanczos algorithm; the computation took 852.5 CPU hours (Sandy Bridge, 4 cores).

[4] More precisely, since C has at most 34 prime factors, each of which is greater than the ten-millionth prime $p = 179424673$, the probability that g is a generator is at least $(1 - \frac{1}{p})^{34} > 0.99999981$.

In the continued-fractions descent stage with $m = 15$, the first degree-81 polynomial yielded 8 irreducible factors of degrees 15, 15, 14, 14, 10, 7, 5, 1, and the second degree-81 polynomial yielded 12 irreducible factors of degrees 12, 10, 9, 9, 9, 8, 6, 6, 6, 4, 1, 1. The computation took 226.7 CPU hours (Sandy Bridge-EP, 16 cores).

Classical descent was used on the 11 polynomials of degree ≥ 8 to obtain polynomials of degree ≤ 7, and then a variant of classical descent (called the "alternative" method in §3.5 of [2]) was used on the 15 polynomials of degree 7 and 30 polynomials of degree 6 to obtain polynomials of degree ≤ 5. These computations took 51.0 CPU hours (Sandy Bridge-EP, 16 cores).

Finally, we used 5-to-3, 4-to-3 and 3-to-2 Gröbner bases descent procedures. The descent was sped up by writing the coefficients of R (cf. equation (11)) in terms of the coefficients of k_1 and k_2; this reduced the number of variables in the resulting bilinear equations from $15m - 3D + 9$ to $9m + 3$. In total, we performed 213 5-to-3 descents, 187 4-to-3 descents, and 11442 3-to-2 descents. These computations took 24.0 CPU hours (Sandy Bridge-EP 16 cores), 0.8 CPU hours (Sandy Bridge, 4 cores), and 44.8 CPU hours (Sandy Bridge, 4 cores), respectively. The running times of the 2-to-1 descents were incorporated into the running times for the higher-level descents.

5 Higher extension degrees

As mentioned in §1, there have been several practical improvements and refinements in discrete logarithm algorithms since Joux's $L[\frac{1}{4} + o(1)]$ algorithm. Most notably, Granger, Kleinjung and Zumbrägel [18] presented several refinements that allowed them to compute logarithms in the 4404-bit characteristic-two field $\mathbb{F}_{2^{12 \cdot 367}}$, and Joux and Pierrot [26] presented a faster algorithm for computing logarithms of factor base elements and used it to compute logarithms in the 3796-bit characteristic-three field $\mathbb{F}_{3^{5 \cdot 479}}$.

In §5.1, we show that the techniques from [26] and [18] can be used to lower the estimate from [1] for computing discrete logarithms in the 4841-bit characteristic-three field $\mathbb{F}_{3^{6 \cdot 509}}$ from $2^{81.7} M_{q^2}$ to $2^{58.9} M_q$ (where $q = 3^6$). In §5.2, we use techniques from [18] to lower the estimate from [2] for computing discrete logarithms in the 13590-bit characteristic-three field $\mathbb{F}_{3^{6 \cdot 1429}}$ from $2^{95.8} M_{q^2}$ to $2^{78.8} M_{q^2}$ (where $q = 3^6$). We emphasize that these estimates are *upper bounds* on the running times of known algorithms for computing discrete logarithms. Of course, it is possible that these upper bounds can be lowered with a more judicious choice of algorithm parameters, or with a tighter analysis, or with improvements to the algorithms themselves.

5.1 Computing discrete logarithms in $\mathbb{F}_{3^{6 \cdot 509}}$. As in §4 of [1], we are interested in computing discrete logarithms in the order r-subgroup of $\mathbb{F}^*_{3^{6 \cdot 509}}$, where $r = (3^{509} - 3^{255} + 1)/7$ is an 804-bit prime.

We use the algorithm developed by Joux and Pierrot [26], whence $q = 3^6$ and $k = 1$. The field $\mathbb{F}_{3^6}$ is represented as $\mathbb{F}_3[u]/(u^6 + 2u^4 + u^2 + 2u + 2)$. The field $\mathbb{F}_{3^{6\cdot 509}}$ is represented as $\mathbb{F}_{3^6}[X]/(I_X)$, where I_X is the degree-509 irreducible factor of $h_1(X)X^q - h_0(X)$ with $h_0(X) = u^{46}X + u^{219}$ and $h_1(X) = X(X+u^{409})$. Joux and Pierrot [26] exploit the special form of $h_0(X)$ and $h_1(X)$ to accelerate the computation of logarithms of polynomials of degree ≤ 4; the dominant step is the computation of logarithms of degree-3 polynomials, where q linear algebra problems are solved each taking time approximately $q^5/27\,A_r$. The continued-fractions, classical and Gröbner bases descents are all performed over $\mathbb{F}_q$.

The new cost estimates are presented in Table 1. We used the estimates for smoothness testing from [17], and the 'bottom-top' approach from [18] for estimating the cost of Gröbner bases descent from degree 15 to degree 4. We assume that 2^{27} multiplications in $\mathbb{F}_{3^6}$ can be performed in 1 second; we achieved this performance using a look-up table approach. The timings for Gröbner bases descent and $\mathbb{F}_{3^6}$ multiplications were obtained on an Intel i7-3930K 3.2 GHz machine. In a non-optimized C implementation, we have observed an A_r cost of 43 clock cycles, where lazy reduction is used to amortize the cost of a modular reduction among many integer additions. This yields the cost ratio $A_r/M_q \approx 2$.

The main effect of the improvements is the removal of the QPA descent stage from the estimates in [1]. The overall running time is $2^{58.9}M_q$, a significant improvement over the $2^{81.7}M_{q^2}$ estimate from [1]. In particular, assuming the availability of processors that can perform 2^{27} $\mathbb{F}_{3^6}$-multiplications per second, the estimated running time is approximately 127 CPU years — this is a feasible computation if one has access to a few hundred cores.

Finding logarithms of polynomials of degree ≤ 4		
Linear algebra	$2^{52.3}A_r$	$2^{53.3}M_q$
Descent		
Continued-fractions (254 to 40)	$2^{56.9}M_q$	$2^{56.9}M_q$
Classical (40 to 21)	$12.7 \times 2^{54.2}M_q$	$2^{57.9}M_q$
Classical (21 to 15)	$159 \times 2^{49.4}M_q$	$2^{56.7}M_q$
Gröbner bases (15 to 4)	1924×8249 seconds	$2^{50.9}M_q$

Table 1. Estimated costs of the main steps for computing discrete logarithms in $\mathbb{F}_{3^{6\cdot 509}}$ ($q = 3^6$). A_r and M_q denote the costs of an addition modulo the 804-bit prime $r = (3^{509} - 3^{255} + 1)/7$ and a multiplication in $\mathbb{F}_{3^6}$. We use the cost ratio $A_r/M_q = 2$, and also assume that 2^{27} multiplications in $\mathbb{F}_{3^6}$ can be performed in 1 second.

Remark 1. The strategy for computing logarithms in $\mathbb{F}_{3^{6\cdot 509}}$ can be employed to compute logarithms in $\mathbb{F}_{3^{6\cdot 239}}$. The latter problem is of cryptographic interest because the prime-order elliptic curve $y^2 = x^3 - x - 1$ over $\mathbb{F}_{3^{239}}$ has embedding degree 6 and has been considered in several papers including [20] and [6]. One could use continued-fractions descent from degree 119 to degree 20 with an estimated cost of $2^{50}M_q$, followed by a classical descent stage from degree 20 to degree 15 at a cost of $2^{53.2}M_q$, and finally Gröbner bases descent to degree 4 at a cost of $2^{47.2}M_q$. The total computational effort is $2^{54.3}M_q$, or approximately 5.2 CPU years.

5.2 Computing discrete logarithms in $\mathbb{F}_{3^{6\cdot 1429}}$. As in §4 of [2], we are interested in computing discrete logarithms in the order r-subgroup of $\mathbb{F}^*_{3^{6\cdot 1429}}$, where $r = (3^{1429} - 3^{715} + 1)/7622150170693$ is a 2223-bit prime. To accomplish this, we embed $\mathbb{F}_{3^{6\cdot 1429}}$ in its quadratic extension $\mathbb{F}_{3^{12\cdot 1429}}$. Let $q = 3^6$ and $k = 2$. The field $\mathbb{F}_{3^{12\cdot 1429}}$ is represented as $\mathbb{F}_{q^2}[X]/(I_X)$, where I_X is a monic degree-1429 irreducible factor of $h_1(X^q) \cdot X - h_0(X^q)$ with $h_0, h_1 \in \mathbb{F}_{q^2}[X]$ and $\max(\deg h_0, \deg h_1) = 2$.

The techniques from [18] employed to improve the estimates of [2] are the following:

1. Since logarithms are actually sought in the field $\mathbb{F}_{3^{6\cdot 1429}}$, the continued fractions and classical descent stages are performed over $\mathbb{F}_q$ (and not $\mathbb{F}_{q^2}$).
2. In the final classical descent stage to degree 11, one permits irreducible factors over $\mathbb{F}_q$ of even degree up to 22; any factors of degree $2t \geq 12$ that are obtained can be written as a product of two degree-t irreducible polynomials over $\mathbb{F}_{q^2}$.
3. The number of irreducible factors of an m-smooth degree-t polynomial is estimated as t/m.
4. The smoothness testing estimates from Appendix B of [17] were used.

The remaining steps of the algorithm, namely finding logarithms of linear polynomial, finding logarithms of irreducible quadratic polynomials, QPA descent, and Gröbner bases descent, are as described in [2].

The new cost estimates are presented in Table 2. The main effect of the techniques from [18] is the removal of one QPA descent stage. The overall running time is $2^{78.8}M_{q^2}$, a significant improvement over the $2^{95.8}M_{q^2}$ estimate from [2].

6 Conclusions

We used Joux's algorithm to solve instances of the discrete logarithm problem in the 1303-bit finite field $\mathbb{F}_{3^{6\cdot 137}}$ and the 1551-bit finite field $\mathbb{F}_{3^{6\cdot 163}}$. We emphasize that these fields are 'general' in that they do not enjoy any Kummer-like properties. The computations took only 888 CPU hours and 1201 CPU hours, respectively, using modest computer resources despite our implementation being in Magma and far from optimal, unlike the substantial resources (approximately 800,000 CPU hours) that were consumed in [22] for computing a logarithm in the 923-bit field $\mathbb{F}_{3^{6\cdot 97}}$ with the Joux-Lercier algorithm. We also used newer techniques from [26] and [18] to lower the estimates for computing discrete logarithms in $\mathbb{F}_{3^{6\cdot 509}}$ and $\mathbb{F}_{3^{6\cdot 1429}}$ to $2^{58.9}M_q$ and $2^{78.8}M_{q^2}$ (where $q = 3^6$). Our computational results add further weight to the claim that Joux's algorithm and its quasi-polytime successor [4] render bilinear pairings derived from the supersingular elliptic curves $E : y^2 = x^3 - x \pm 1$ over $\mathbb{F}_{3^n}$ unsuitable for pairing-based cryptography.

Finding logarithms of linear polynomials		
Relation generation	$2^{28.6} M_{q^2}$	$2^{28.6} M_{q^2}$
Linear algebra	$2^{47.5} A_r$	$2^{49.5} M_{q^2}$
Finding logarithms of irreducible quadratic polynomials		
Relation generation	$3^{12} \times 2^{37.6} M_{q^2}$	$2^{56.6} M_{q^2}$
Linear algebra	$3^{12} \times 2^{47.5} A_r$	$2^{68.5} M_{q^2}$
Descent		
Continued-fractions (714 to 88)	$2^{77.6} M_q$	$2^{77.6} M_q$
Classical (88 to 29)	$16.2 \times 2^{73.5} M_q$	$2^{77.5} M_q$
Classical (29 to 11)	$267.3 \times 2^{70.8} M_q$	$2^{78.9} M_q$
QPA (11 to 7)	$2^{13.9} \times (2^{44.4} M_{q^2} + 2^{47.5} A_r)$	$2^{63.4} M_{q^2}$
Gröbner bases (7 to 4)	$2^{35.2} \times$ (76.9 seconds)	$2^{67.5} M_{q^2}$
Gröbner bases (4 to 3)	$2^{44.7} \times$ (0.03135 seconds)	$2^{65.7} M_{q^2}$
Gröbner bases (3 to 2)	$2^{54.2} \times$ (0.002532 seconds)	$2^{71.6} M_{q^2}$

Table 2. Estimated costs of the main steps for computing discrete logarithms in $\mathbb{F}_{3^{12 \cdot 1429}}$ ($q = 3^6$). A_r, M_q, and M_{q^2} denote the costs of an addition modulo the 2223-bit prime r, a multiplication in $\mathbb{F}_{3^6}$, and a multiplication in $\mathbb{F}_{3^{12}}$. We use the cost ratio $A_r/M_{q^2} = 4$, and also assume that 2^{26} (resp. 2^{27}) multiplications in $\mathbb{F}_{3^{12}}$ (resp. $\mathbb{F}_{3^6}$) can be performed in 1 second (cf. §5.1).

References

1. G. Adj, A. Menezes, T. Oliveira and F. Rodríguez-Henríquez, "Weakness of $\mathbb{F}_{3^{6 \cdot 509}}$ for discrete logarithm cryptography", *Pairing-Based Cryptography — Pairing 2013*, LNCS 8365 (2014), 20–44.
2. G. Adj, A. Menezes, T. Oliveira and F. Rodríguez-Henríquez, "Weakness of $\mathbb{F}_{3^{6 \cdot 1429}}$ and $\mathbb{F}_{2^{4 \cdot 3041}}$ for discrete logarithm cryptography", *Finite Fields and Their Applications*, to appear.
3. R. Barbulescu, C. Bouvier, J. Detrey, P. Gaudry, H. Jeljeli, E. Thomé, M. Videau and P. Zimmermann, "Discrete logarithm in $GF(2^{809})$ with FFS", *Public Key Cryptography — PKC 2014*, LNCS 8383 (2014), 221–238.
4. R. Barbulescu, P. Gaudry, A. Joux and E. Thomé, "A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic: Improvements over FFS in small to medium characteristic", *Advances in Cryptology — EUROCRYPT 2014*, LNCS 8441 (2014), 1–16.
5. P. Barreto, H. Kim, B. Lynn and M. Scott, "Efficient algorithms for pairing-based cryptosystems", *Advances in Cryptology — CRYPTO 2002*, LNCS 2442 (2002), 354–368.
6. J. Beuchat, J. Detrey, N. Estibals, E. Okamoto and F. Rodríguez-Henríquez, "Fast architectures for the η_T pairing over small-characteristic supersingular elliptic curves", *IEEE Transactions on Computers*, 60 (2011), 266–281.
7. I. Blake, R. Fuji-Hara, R. Mullin and S. Vanstone, "Computing logarithms in finite fields of characteristic two", *SIAM Journal on Algebraic and Discrete Methods*, 5 (1984), 276–285.
8. D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing", *Journal of Cryptology*, 17 (2004), 297–319.
9. D. Coppersmith, "Fast evaluation of logarithms in fields of characteristic two", *IEEE Transactions on Information Theory*, 30 (1984), 587–594.

10. D. Coppersmith, "Solving homogeneous linear equations over $GF(2)$ via block Wiedemann algorithm", *Mathematics of Computation*, 62 (1994), 333–350.
11. The Cunningham Project, http://homes.cerias.purdue.edu/~ssw/cun/.
12. J. Faugère, "A new efficient algorithm for computing Gröbner bases (F_4)", *Journal of Pure and Applied Algebra*, 139 (1999), 61–88.
13. G. Frey and H. Rück, "A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves", *Mathematics of Computation*, 62 (1994), 865–874.
14. S. Galbraith, K. Harrison and D. Soldera, "Implementing the Tate pairing", *Algorithmic Number Theory — ANTS 2002*, LNCS 2369 (2002), 324–337.
15. F. Göloğlu, R. Granger, G. McGuire and J. Zumbrägel, "On the function field sieve and the impact of higher splitting probabilities: Application to discrete logarithms in $\mathbb{F}_{2^{1971}}$", *Advances in Cryptology — CRYPTO 2013*, LNCS 8043 (2013), 109–128.
16. F. Göloğlu, R. Granger, G. McGuire and J. Zumbrägel, "Solving a 6120-bit DLP on a desktop computer", *Selected Areas in Cryptography — SAC 2013*, LNCS 8282 (2014), 136–152.
17. R. Granger, T. Kleinjung and J. Zumbrägel, "Breaking '128-bit secure' supersingular binary curves (or how to solve discrete logarithms in $\mathbb{F}_{2^{4\cdot 1223}}$ and $\mathbb{F}_{2^{12\cdot 367}}$)", available at http://eprint.iacr.org/2014/119.
18. R. Granger, T. Kleinjung and J. Zumbrägel, "Breaking '128-bit secure' supersingular binary curves (or how to solve discrete logarithms in $\mathbb{F}_{2^{4\cdot 1223}}$ and $\mathbb{F}_{2^{12\cdot 367}}$)", *Advances in Cryptology — CRYPTO 2014*, Part II, LNCS 8617 (2014), 126–145.
19. R. Granger, D. Page and M. Stam, "Hardware and software normal basis arithmetic for pairing based cryptography in characteristic three", *IEEE Transactions on Computers*, 54 (2005), 852–860.
20. R. Granger, D. Page and M. Stam, "On small characteristic algebraic tori in pairing-based cryptography", *LMS Journal of Computation and Mathematics*, 9 (2006), 64–85.
21. R. Granger and J. Zumbrägel, "On the security of supersingular binary curves", presentation at ECC 2013, September 16 2013.
22. T. Hayashi, T. Shimoyama, N. Shinohara and T. Takagi, "Breaking pairing-based cryptosystems using η_T pairing over $GF(3^{97})$", *Advances in Cryptology — ASIACRYPT 2012*, LNCS 7658 (2012), 43–60.
23. A. Joux, "A new index calculus algorithm with complexity $L(1/4 + o(1))$ in very small characteristic", *Selected Areas in Cryptography — SAC 2013*, LNCS 8282 (2014), 355–379.
24. A. Joux, "Discrete logarithm in $GF(2^{6128})$", Number Theory List, May 21 2013.
25. A. Joux and R. Lercier, "The function field sieve in the medium prime case" *Advances in Cryptology — EUROCRYPT 2006*, LNCS 4004 (2006), 254–270.
26. A. Joux and C. Pierrot, "Improving the polynomial time precomputation of Frobenius representation discrete logarithm algorithms", *Advances in Cryptology — ASIACRYPT 2014*, to appear.
27. Magma v2.19-7, http://magma.maths.usyd.edu.au/magma/.
28. A. Menezes, T. Okamoto and S. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field", *IEEE Transactions on Information Theory*, 39 (1993), 1639–1646.
29. J. Pollard, "Monte Carlo methods for index computation mod p", *Mathematics of Computation*, 32 (1978), 918–924.
30. N. Shinohara, T. Shimoyama, T. Hayashi and T. Takagi, "Key length estimation of pairing-based cryptosystems using η_T pairing", *Information Security Practice and Experience — ISPEC 2012*, LNCS 7232 (2012), 228–244.

31. D. Wiedemann, “Solving sparse linear equations over finite fields”, *IEEE Transactions on Information Theory*, 32 (1986), 54–62.

A Magma script for verifying the $\mathbb{F}_{3^{6\cdot 137}}$ discrete log

```
//Definition of the extension fields Fq := F3(U) and Fq3 := Fq(V)
q         := 3^4;
F3        := FiniteField(3);
P3<u>     := PolynomialRing(F3);
poly      := u^4 + u^2 + 2;
Fq<U>     := ext<F3|poly>;
Pq<v>     := PolynomialRing(Fq);
poly      := v^3 + v + U^2 + U;
Fq3<V>    := ext<Fq|poly>;
Pq3<Z>    := PolynomialRing(Fq3);
r         := 33098280119090191028775580055082175056428495623;
Fr        := GF(r);

h0        := V^326196*Z^2 + V^35305*Z + V^204091;
h0q       := Evaluate(h0,Z^q);
F         := Z - h0q;
Ix        := Factorization(F)[2][1];
Fn<X>     := ext<Fq3|Ix>;
N         := #Fn - 1;

// Generator of GF(3^{12*137})^*
g         := X + V^113713;

// Encoding pi
Re        := RealField(2000);
pival     :=Pi(Re);
hp        := 0;
for i := 0 to 136 do
          hp := hp + V^(Floor(pival*(#Fq3)^(i+1)) mod #Fq3)*(X^i);
end for;

// This is the logarithm challenge
cofactor := N div r;
h         := hp^cofactor;

// log_g(h) mod r is:
x         := 27339619076975093920245515973214186963025656559;

// Define the exponent y to be used in the verification:
y         := IntegerRing()!(Fr!(x/cofactor));

// Check that h = (g^cofactor)^y
h eq (g^cofactor)^y;
```

B Magma script for verifying the $\mathbb{F}_{3^{6\cdot 163}}$ discrete log

```
//Definition of the extension fields Fq := F3(U) and Fq3 := Fq(V)
q         := 3^4;
F3        := FiniteField(3);
P3<u>     := PolynomialRing(F3);
poly      := u^4 + u^2 + 2;
Fq<U>     := ext<F3|poly>;
Pq<v>     := PolynomialRing(Fq);
poly      := v^3 + v + U^2 + U;
Fq3<V>    := ext<Fq|poly>;
Pq3<Z>    := PolynomialRing(Fq3);
r         := 58988115142665874085422772558073634885064063229737341409179O
             995505756623268837;
Fr        := GF(r);

h1        := Z^2 + V^530855;
h1q       := Evaluate(h1,Z^q);
Ix        := h1q*Z - 1;
Fn<X>     := ext<Fq3|Ix>;
N         := #Fn - 1;

// Generator of GF(3^{12*163})^*
g         := X + V^2;

// Encoding pi
Re        := RealField(2000);
pival     :=Pi(Re);
hp        := 0;
for i := 0 to 162 do
          hp := hp + V^(Floor(pival*(#Fq3)^(i+1)) mod #Fq3)*(X^i);
end for;

// This is the logarithm challenge
cofactor := N div r;
h         := hp^cofactor;

// log_g(h) mod r is:
x         := 4263959514982791937132913919534490007325925542511325256720З
             9784356054526194343;

// Define the exponent y to be used in the verification:
y         := IntegerRing()!(Fr!(x/cofactor));

// Check that h = (g^cofactor)^y
h eq (g^cofactor)^y;
```