COMPUTING DISCRETE LOGARITHMS IN CRYPTOGRAPHICALLY-INTERESTING CHARACTERISTIC-THREE FINITE FIELDS

GORA ADJ, ISAAC CANALES-MARTÍNEZ, NARELI CRUZ-CORTÉS, ALFRED MENEZES, THOMAZ OLIVEIRA, LUIS RIVERA-ZAMARRIPA, AND FRANCISCO RODRÍGUEZ-HENRÍQUEZ

ABSTRACT. Since 2013 there have been several developments in algorithms for computing discrete logarithms in small-characteristic finite fields, culminating in a quasi-polynomial algorithm. In this paper, we report on our successful computation of discrete logarithms in the cryptographically-interesting characteristic-three finite field $\mathbb{F}_{3^{6\cdot 509}}$ using these new algorithms; prior to 2013, it was believed that this field enjoyed a security level of 128 bits. We also show that a recent idea of Guillevic can be used to compute discrete logarithms in the cryptographically-interesting finite field $\mathbb{F}_{3^{6\cdot 709}}$ using essentially the same resources as we expended on the $\mathbb{F}_{3^{6\cdot 509}}$ computation. Finally, we argue that discrete logarithms in the finite field $\mathbb{F}_{3^{6\cdot 1429}}$ can feasibly be computed today; this is significant because this cryptographically-interesting field was previously believed to enjoy a security level of 192 bits.

1. INTRODUCTION

Let $\mathbb{F}_q$ denote a finite field of order q. The discrete logarithm problem (DLP) in $\mathbb{F}_q$ is the following: given an element $g \in \mathbb{F}_q^*$ of order r, and $h \in \langle g \rangle$, find the integer $x \in [0, r-1]$ such that $h = g^x$. The integer x is called the discrete logarithm of h to the base g and is denoted by $\log_g h$. Before 2013, the fastest general-purpose algorithm known for solving the DLP in the case where q is a prime was the Number Field Sieve [21, 35] with running time $L_q[\frac{1}{3}, 1.923]$, and the fastest general-purpose algorithm known for solving the DLP in the case where q is a power of 2 or 3 was Coppersmith's algorithm [13] with running time $L_q[\frac{1}{3}, 1, 526]$. Here, $L_q[\alpha, c]$ 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 $\log q$.

Since the DLP in small-characteristic fields was a little easier than the DLP in prime-order fields, the vast majority of early research and development of discrete-logarithm cryptographic protocols used prime-order fields. This situation changed at the beginning of the present century with the introduction of pairing-based cryptography (see [10]), which employs non-degenerate bilinear pairings derived from elliptic curves and genus-2 hyperelliptic curves. In particular, some influential early papers [11, 17, 8, 18, 7] considered symmetric pairings $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ derived from supersingular elliptic curves E and genus-2 hyperelliptic curves C defined over finite fields $\mathbb{F}_q$ of characteristic 2 or 3. Here, $\mathbb{G}$ is a subgroup of prime-order r of $E(\mathbb{F}_q)$, the group of $\mathbb{F}_q$-rational points on E, or of $\mathrm{Jac}_C(\mathbb{F}_q)$, the jacobian of C over $\mathbb{F}_q$; $\mathbb{G}_T$ is the order-r subgroup of the multiplicative group of $\mathbb{F}_{q^k}$;

Date: September 20, 2016.

and the embedding degree k is the smallest positive integer such that $r \mid (q^k - 1)$. Three symmetric pairings that were widely studied and implemented are:

(1) the $k = 6$ pairings derived from supersingular elliptic curves $Y^2 = X^3 - X + 1$ and $Y^2 = X^3 - X - 1$ over $\mathbb{F}_q$ with $q = 3^n$;
(2) the $k = 4$ pairings derived from supersingular elliptic curves $Y^2 + Y = X^3 + X$ and $Y^2 + Y = X^3 + X + 1$ over $\mathbb{F}_q$ with $q = 2^n$;
(3) the $k = 12$ pairings derived from supersingular genus-2 curves $Y^2 + Y = X^5 + X^3$ and $Y^2 + Y = X^5 + X^3 + 1$ over $\mathbb{F}_q$ with $q = 2^n$.

In all cases, n is chosen to be a prime such that $\#E(\mathbb{F}_q)$ or $\#\mathrm{Jac}_C(\mathbb{F}_q)$ is divisible by a large prime r. A necessary condition for the security of these pairings is the intractability of the DLP in $\mathbb{G}_T$, and thus also in $\mathbb{F}_{q^k}$ [34, 16]. Consequently, the DLP in such finite fields $\mathbb{F}_{3^{6n}}$, $\mathbb{F}_{2^{4n}}$ and $\mathbb{F}_{2^{12n}}$ became especially important from a cryptographic point of view; we will henceforth say that these fields are 'cryptographically interesting'.

In 2013, there were several spectacular developments in algorithms for computing discrete logarithms in small-characteristic finite fields, culminating in a quasi-polynomial time algorithm [26, 19, 6]. These developments were accompanied by some striking computational results such as the computation of discrete logarithms in the 6120-bit field $\mathbb{F}_{2^{8\cdot 3\cdot 255}}$ in only 550 CPU hours [20]; see [29] for a complete list. In 2014, Granger, Kleinjung and Zumbrägel [22] reported the first computation of discrete logarithms in one of the cryptographically-interesting finite fields $\mathbb{F}_{3^{6n}}$, $\mathbb{F}_{2^{4n}}$, $\mathbb{F}_{2^{12n}}$ that was believed to offer 128 bits of security against Coppersmith's attack, namely the 4404-bit field $\mathbb{F}_{2^{12\cdot 367}}$.

Let $q = 3^6$. In this paper, we shall focus on the DLP in cryptographically-interesting fields $\mathbb{F}_{q^n} = \mathbb{F}_{3^{6n}}$. In [3] and [4], Adj et al. showed that the new algorithms can in principle be used to compute logarithms in $\mathbb{F}_{3^{6\cdot 509}}$ and $\mathbb{F}_{3^{6\cdot 1429}}$ in $2^{81.7}M_{q^2}$ and $2^{95.8}M_{q^2}$ time, respectively, where M_{q^2} denotes the time to perform one multiplication in $\mathbb{F}_{q^2}$. These results were cryptographically significant because the fields $\mathbb{F}_{3^{6\cdot 509}}$ and $\mathbb{F}_{3^{6\cdot 1429}}$ were believed to offer 128 and 192 bits of security against Coppersmith's attack (see [31]). However, the computations were still infeasible using existing computer technology. Then, in [5], Adj et al. used ideas from [28] and [22] to improve their estimates for discrete logarithm computations in $\mathbb{F}_{3^{6\cdot 509}}$ and $\mathbb{F}_{3^{6\cdot 1429}}$ to $2^{58.9}M_q$ and $2^{78.8}M_{q^2}$, respectively, where M_q denotes the time to perform one multiplication in $\mathbb{F}_q$.

In §3, we describe our computation of discrete logarithms in the 4841-bit field $\mathbb{F}_{3^{6\cdot 509}}$. This is the second computation of discrete logarithms in a cryptographically-interesting finite field that was purported to provide 128 bits of security against Coppersmith's attack. Then, in §4, we show that a recent idea of Guillevic [25] can be used to compute discrete logarithms in $\mathbb{F}_{3^{6\cdot 709}}$ using essentially the same resources as we expended on the $\mathbb{F}_{3^{6\cdot 509}}$ computation. Furthermore, in §5 we lower the estimates for discrete logarithm computations in $\mathbb{F}_{3^{6\cdot 1429}}$ to $2^{63.4}M_q$. We argue that this computation is feasible today, even though it is just beyond the reach of the computer resources available to the authors of this paper. This is the first demonstration that pairing-based cryptosystems originally believed to offer 192 bits of security can be broken in practice today.

We begin in §2 by providing an overview of the key ingredients in the DLP algorithm.

2. Overview of the DLP algorithm

For the sake of concreteness, we focus on DLP instances in cryptographically-interesting finite fields $\mathbb{F}_{3^{6n}}$. Let $q = 3^6$ and let n be prime. Let r be a large prime divisor of $3^n \pm \sqrt{3^{n+1}} + 1$, whence $r \mid (q^n - 1)$. Let g be a generator of $\mathbb{F}_{q^n}^*$, and let h be an element of the order-r subgroup of $\mathbb{F}_{q^n}^*$. We wish to determine $x = \log_g h \bmod r$.

The elements of $\mathbb{F}_{q^n}$ are represented as polynomials of degree at most $n-1$ over $\mathbb{F}_q$. The algorithm begins by building a factor base of logarithms of all degree-one, degree-two and degree-three polynomials over $\mathbb{F}_q$, and a proportion of degree-four polynomials. Then, in the descent stage, various techniques are used to recursively express $\log_g h$ as a linear combination of logarithms of smaller-degree polynomials until all these polynomials belong to the factor base.

Notation. $N_q(m, n)$ denotes the number of monic m-smooth degree-n polynomials in $\mathbb{F}_q[X]$, and $S_q(m, d)$ denotes the cost of testing m-smoothness of a degree-d polynomial in $\mathbb{F}_q[X]$. Formulas for $N_q(m, n)$ and $S_q(m, n)$ are given in [3] and [23], respectively.

2.1. **Frobenius representation.** Let $h_0, h_1 \in \mathbb{F}_q[X]$ be polynomials such that $h_1 X^q - h_0$ has a degree-n irreducible factor I_X in $\mathbb{F}_q[X]$ and $\max(\deg h_0, \deg h_1) = 2$. The field $\mathbb{F}_{q^n}$ is represented as $\mathbb{F}_q[X]/(I_X)$. This *Frobenius representation*, introduced by Joux [26], has the useful property that

$$X^q \equiv h_0/h_1 \pmod{I_X}.$$

2.2. **Small degrees.** We use the Joux-Pierrot [28] method for computing logarithms of small-degree polynomials. The main idea is to partition the set of irreducible cubics and quartics into smaller families, and to exploit the special form of h_0 and h_1 to find relations of logarithms of elements within a family. The Joux-Pierrot method requires that h_0 and h_1 have the form $h_0(X) = \alpha_0 X + \alpha_1$ and $h_1(X) = X^2 + \alpha_2 X$.

2.2.1. *Linear and quadratic polynomials.* The dominant cost of finding logarithms of elements in $\mathcal{B}_2$, the set of all linear and irreducible quadratic polynomials over $\mathbb{F}_q$, is the solution of a linear system of size $\approx q^2/2 \times q^2/2$ and row density $\approx 3q/2$. The linear system can be solved using Wiedemann's algorithm [36] at a cost of approximately $9q^5/8A_r$, where A_r denotes the cost of an addition modulo the integer r.[1]

2.2.2. *Cubic polynomials.* The set $\mathcal{B}_3$ of irreducible cubics over $\mathbb{F}_q$ is partitioned into $q-1$ families $\mathcal{B}_{3,\gamma} = \{X^3 + aX^2 + bX + \gamma\}$, each of size exactly $(q^2+q)/3$. The dominant cost of finding logarithms of polynomials in a family $\mathcal{B}_{3,\gamma}$ is the solution of a linear system of size $\approx q^2/3 \times q^2/3$ and row density $\approx q/3$; the total cost of solving the $q-1$ linear systems is $\approx q^6/9A_r$. Note that the $q-1$ linear systems can be generated and solved independently of each other.

[1]The dominant cost of Wiedemann's algorithm for solving an $N \times N$ system of linear equations with row density λ and where the nonzero entries are small (e.g., $0, \pm1, \pm2$) is $3\lambda N^2 A_r$.

2.2.3. *Quartic polynomials.* The irreducible quartics over $\mathbb{F}_q$ are partitioned into $q-1$ families $\mathcal{B}_{4,\gamma} = \{X^4 + aX^3 + bX^2 + \delta X + \gamma\}$, each of size approximately $q^3/4$. Each family $\mathcal{B}_{4,\gamma}$ is further partitioned into q subfamilies $\mathcal{B}_{4,\gamma,\delta}$ according to the coefficient δ of the linear term in a polynomial; we have $|\mathcal{B}_{4,\gamma,\delta}| \approx q^2/4$.

To compute logarithms of all quartics Q in a subfamily $\mathcal{B}_{4,\gamma,\delta}$, one first attempts a 4-to-3 Gröbner bases descent on Q (see §2.5). This is expected to be successful about 50% of the time. If unsuccessful, then the 'Frobenius strategy' described in [5] is employed. Namely, one has

$$\tilde{Q}(X) = h_1^4 Q(X^q) = h_1^4 Q(h_0/h_1) = h_0^4 + a h_0^3 h_1 + b h_0^2 h_1^2 + \delta h_0 h_1^3 + \gamma h_1^4.$$

The polynomial $\tilde{Q}$ is either irreducible or a product of two irreducible quartics, the latter occurring with probability[2] approximately $\frac{1}{2}$ (see Lemma 2 in [28]). In the latter case, a 4-to-3 Gröbner bases/Frobenius descent is recursively attempted on each of the two quartic factors of $\tilde{Q}$. If both are successful, then we have succeeded in descending Q.

About 58.6% of all quartics Q can be descended in this way [28]. Thus, one expects there to be $(1-0.586)\cdot q^2/4 \le 0.11q^2$ quartics in $\mathcal{B}_{4,\gamma,\delta}$ whose logarithms are yet to be determined. The Joux-Pierrot technique is then used to generate relations of logarithms of these quartics, yielding a linear system of size $\approx 0.11q^2 \times 0.11q^2$ and row density $\approx 0.11q$. The $0.00399q^5 A_r$ cost of solving this linear system dominates the cost of computing logarithms of all polynomials in a subfamily. Thus, the cost of computing logarithms of all polynomials in a family is $0.00399q^6 A_r$, and the cost of computing logarithms of all quartics is $0.00399q^7 A_r$.

Note that the $q(q-1)$ linear systems can be generated and solved independently of each other. While sacrificing some parallelizability, Joux and Pierrot [28] described how the linear systems associated with families can be recursively halved in dimension using the family-based Gröbner bases descent method described in §2.2.4. With this modification, the dominant cost of computing logarithms of irreducible quartics is the cost of the linear algebra for the first few of the $q-1$ families.

2.2.4. *Family-based Gröbner bases descent for quartics.* To avoid the high storage and lookup costs for the factor base of logarithms of all $(q^4-q^2)/4$ irreducible quartics, one can instead compute and store the logarithms of quartics in a relatively small number of families, say $\mathcal{B}_{4,\gamma_i}$ for $i = 1, 2, \ldots, s$. Suppose that $Q \in \mathbb{F}_q[X]$ is an irreducible quartic that is not in one of these s families. Then $\log_g Q$ can be computed on-the-fly as follows.

We first attempt a 4-to-3 Gröbner bases descent and the Frobenius strategy on Q. If this fails, we consider the first family $\mathcal{B}_{4,\gamma_1}$. Our goal is to find polynomials $k_1(X) = X^4 + a_2X^2 + a_1X + \gamma_1$ and $k_2(X) = X^3 + b_2X^2 + b_1X \in \mathbb{F}_q[X]$ such that $Q \mid G$, where

$$G = h_1^4(k_1^q k_2 - k_1 k_2^q) \bmod I_X.$$

In this case, we have

$$G(X) \equiv h_1^4 \cdot (X^3 + b_2X^2 + b_1X) \cdot \prod_{\alpha\in\mathbb{F}_q} \left(X^4 - \alpha X^3 + (a_2 - \alpha b_2)X^2 + (a_1 - \alpha b_1)X + \gamma_1\right) \pmod{I_X} \tag{1}$$

[2]All statements in this paper about probability are heuristic, and assume that polynomials generated in the course of the algorithm are distributed uniformly at random from a certain set of polynomials.

as can be seen by making the substitution $Y \mapsto k_1/k_2$ into the systematic equation

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

and clearing denominators. It is clear that all the irreducible quartics appearing in the right side of (1) are $\mathcal{B}_{4,\gamma_1}$-elements. Note that

$$G = (h_0^4 + a_1 h_0^2 h_1^2 + a_2 h_0 h_1^3 + \gamma_1 h_1^4)(X^3 + b_2 X^2 + b_1 X) \\ - (X^4 + a_2 X^2 + a_1 X + \gamma_1)(h_0^3 h_1 + b_2 h_0^2 h_1^2 + b_1 h_0 h_1^3),$$

which is a degree-11 polynomial divisible by $X(h_1 X - h_0)$ and Q. The cofactor of $X \cdot (h_1 X - h_0) \cdot Q$ in G is a degree-3 polynomial. Thus, equation (1) yields an expression for $\log_g Q$ in terms of logarithms of polynomials of degree ≤ 3 and polynomials in $\mathcal{B}_{4,\gamma_1}$. Since these logarithms are all known, we can determine $\log_g Q$.

To find polynomials (k_1, k_2) such that $Q \mid G$, one proceeds as in the classical Gröbner bases descent (see §2.5), with the same computational cost, whereby a system of multivariate bilinear equations is solved using a Gröbner basis finding algorithm.

As discussed in §2.2.3, this descent method, together with the Frobenius strategy, is successful for only about 58.6% of all irreducible quadratics Q not in $\mathcal{B}_{4,\gamma_1}$. If the descent fails, then the procedure is iterated with the other families $\mathcal{B}_{4,\gamma_i}$ for $i = 2, \ldots, s$. The probability that the descent fails after s iterations is $(1 - 0.586)^{s+1}$.

2.3. **Continued-fractions descent.**

Suppose that $\deg h = n - 1$. The descent 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 $\deg w_1, \deg w_2 \approx n/2$ [9]. This process is repeated until both w_1 and w_2 are m-smooth for some chosen $m < (n-1)/2$, thus giving $\log_h h$ as a linear combination of logarithms of polynomials of degree at most m. The expected cost of this continued-fractions descent is approximately

$$\left(\frac{q^{(n-1)/2}}{N_q(m, (n-1)/2)} \right)^2 \cdot S_q(m, (n-1)/2). \tag{2}$$

The logarithms of the polynomials of degree $\leq m$ are then expressed as linear combinations 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.**

The classical descent method has its origins in the work of Joux and Lercier [27]. We follow the description in §5.5 of [4] and refer to that paper for further details.

Suppose that we wish to determine $\log_g Q$, where $\deg Q = D$. One selects parameters $m < D$, $s \in [0, 6]$, and $\delta \geq 1$. The classical descent method yields candidate polynomials (R_1, R_2) with $Q \mid R_1$, $\deg R_1 = t_1 \approx (\lfloor D/2 \rfloor + \delta) + 3^{6-s}$, $\deg R_2 = t_2 \approx (\lfloor D/2 \rfloor + \delta) \cdot 3^s + 2$, and such that $\log_g Q$ can be written in terms of $\log_g(R_1/Q)$ and $\log_g R_2$. Pairs (R_1, R_2) are generated until one is found where both R_1/Q and R_2 are m-smooth. In order to ensure that there are sufficiently many candidates (R_1, R_2), the parameters m, s and δ must be selected so that

$$q^{2\delta} \gg \frac{q^{t_1 - D}}{N_q(m, t_1 - D)} \cdot \frac{q^{t_2}}{N_q(m, t_2)}. \tag{3}$$

If this condition is satisfied, then the expected cost of the D-to-m classical descent is

$$\frac{q^{t_1-D}}{N_q(m,t_1-D)}\cdot\frac{q^{t_2}}{N_q(m,t_2)}\cdot\min(S_q(m,t_1-D),S_q(m,t_2)). \tag{4}$$

2.5. **Gröbner bases descent.** Let $Q\in\mathbb{F}_q[X]$ be an irreducible polynomial of degree D, and let $m\geq 1$. In Joux's D-to-m descent [26] (see also §2.5 of [5]), one obtains a system of $3m+1$ bilinear equations in $5m-D+3$ variables over $\mathbb{F}_q$. The system of equations can be solved by finding a Gröbner basis for the ideal it generates. Provided that the condition

$$q^{2m+1-D}\gg q^{3m-D}/N_q(m,3m-D) \tag{5}$$

is satisfied [22], one expects to obtain an expression for $\log_g Q$ in terms of the logarithms of slightly more than q (not necessarily irreducible) polynomials of degree m.

2.6. **Zigzag descent.** Let $Q\in\mathbb{F}_q[X]$ be an irreducible polynomial of degree $2m$, $m\geq 3$. In [24], one begins by lifting Q to $\mathbb{F}_{q^m}[X]$, where it factors into m irreducible quadratics. The factors Q_i, where $0\leq i<m$, are conjugates and can be ordered so that $Q_i=Q_0^{[i]}$ where $Q_0^{[i]}$ denotes the polynomial obtained by raising each coefficient of Q_0 to the power q^i.

Next, the 2-to-1 on-the-fly descent method [19, 20] is employed to obtain a relation involving Q_0 and slightly more than q linear polynomials over $\mathbb{F}_{q^m}$. The descent is always expected to be successful if $m\geq 4$. In contrast, when $m=3$ only 50% of the irreducible quadratics over $\mathbb{F}_{q^m}$ are expected to descend.

Suppose now that we have a relation

$$Q_0\cdot\prod_s F_s=\prod_t G_t,$$

where the F_s and G_t are linear polynomials over $\mathbb{F}_{q^m}$. Then, for each $0\leq i<m$, we have $Q_i\cdot\prod_s F_s^{[i]}=\prod_t G_t^{[i]}$. This gives

$$Q\cdot\prod_s\left(F_s^{[0]}\cdots F_s^{[m-1]}\right)=\prod_t\left(G_t^{[0]}\cdots G_t^{[m-1]}\right).$$

Since for every pair of indexes (s,t), the products $F_s^{[0]}\cdots F_s^{[m-1]}$ and $G_t^{[0]}\cdots G_t^{[m-1]}$ are nothing more than the respective polynomial norms of the linear polynomials F_s and G_t over $\mathbb{F}_q$ and, therefore, are degree-m polynomials in $\mathbb{F}_q[X]$, we get an expression for $\log_g Q$ in terms of the logarithms of polynomials of degree (at most) m over $\mathbb{F}_q$.

3. Discrete logarithms in $\mathbb{F}_{3^{6\cdot 509}}$

The DLP instance we solved is described in §3.1. Some details of our implementation are presented in §3.2.

3.1. **Problem instance.** Let N denote the order of $\mathbb{F}_{3^{6\cdot 509}}^*$. Using the tables from the Cunningham Project [14], we partially factored N as $N = C \cdot p_1^3 \cdot \prod_{i=2}^{21} p_i$, where the p_i are the following primes (and $r = p_{21}$):

$p_1 = 2$ $p_2 = 7$ $p_3 = 13$ $p_4 = 1019$ $p_5 = 7127$ $p_6 = 21379$ $p_7 = 54973$ $p_8 = 97729$

$p_9 = 14495303$ $p_{10} = 39115633$ $p_{11} = 324927277$ $p_{12} = 1644550169868135799$

$p_{13} = 59561824373572167761652488341$ $p_{14} = 1408323592065265621229603282020508687$

$p_{15} = 19724128725821325379688781664270351435664812399$

$p_{16} = 445822414421517590127833782065296611184663760610930675526963377000813$

$p_{17} = 746958920898165755935823445465202971333729069876913071203956406484495983074 3$

$p_{18} = 316301639905466145321616735671363539633419684190527002790777119629131087134 6$
$2585889709129767869836324 9$

$p_{19} = 2321404946852574407463368383953127746140259615234678007610408233057803930650$
$588927817902650339528272898911095769691793236585093530887374961583727 3$

$p_{20} = 4481615016192797792064736092967441840962676013893578289418828353437587210956$
$6942820514222439432850646355157098126051586360410836283973747433357183937067$
44237385203

$p_{21} = 1022399462025868524098098874180930214571506124952557066147330033275262790815$
$6368783078274830574618706026498586928352444181958959275099808618631525078106$
$7131293823177124077445718802216415539934838376431091001197641295264650596195$
201747790167311

and $C = (3^{1018} + 3^{509} + 1)/13$ is a 1610-bit composite number.

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 $\mathbb{F}_{3^{6\cdot 509}}^*$, and g satisfies $g^{N/p_i} \neq 1$ for $1 \leq i \leq 21$, then g is a generator of $\mathbb{F}_{3^{6\cdot 509}}^*$ with very high probability.

We chose the representation $\mathbb{F}_{3^6} = \mathbb{F}_3[u]/(u^6 + 2u^4 + u^2 + 2u + 2)$, with u generating $\mathbb{F}_{3^6}^*$. 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 $h_0(X) = u^{316}X + u^{135}$ and $h_1(X) = X^2 + u^{424}X$.

We chose the (presumed) generator $g = X + u^2$ of $\mathbb{F}_{3^{6\cdot 509}}^*$. To generate an order-$r$ discrete logarithm challenge h, we computed

$$h' = \sum_{i=0}^{508} \left(u^{\lfloor \pi \cdot (3^6)^{i+1} \rfloor \bmod 3^6} \right) X^i$$

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

$x = 1491873998603182663602166336932969933774562818915699213253138178777703786430493 06$
$6480809525652983536765790074515932016074643909955366015387428992219846518967940 08$
$8550218976628414869296477962794457122205313359696590735777748798909231235385145 6.$

This can be verified by checking that $h = (g^{N/r})^y$, where $y = x \cdot (N/r)^{-1} \bmod r$. Magma script for this verification is available at http://tinyurl.com/GF3-6-509.

3.2. **Experimental results.** The computation described in this section was done using clusters from Cinvestav's ABACUS supercomputer [1] (Intel Xeon E5-2697 v3 2.60 GHz cores, 5096 CPU cores used), Cinvestav's Computer Science Department (290 CPU cores used), and University of Waterloo's Faculty of Mathematics (120 CPU cores used). We used Magma's implementation of Faugère's F4 algorithm for Gröbner bases finding [15, 32]. Polynomial smoothness testing was implemented in C [2, 12]. Table 1 gives the number of CPU years that were expended on each stage of the computation. The CPU frequency column lists the average clock speed of the cores used.

Computation stage	**CPU time (years)**	**CPU frequency (GHz)**
Finding logarithms of quadratic polynomials		
Relation generation	0.0003	3.20
Linear algebra	0.49	2.40
Finding logarithms of cubics		
Relation generation	0.14	3.20
Linear algebra	43.28	2.60
Finding logarithms of quartics (29 families)		
Relation generation	4.01	2.60
Linear algebra	94.70	2.60
Descent		
Continued-fractions (508 to 40)	51.00	2.87
Classical (40 to 21)	9.85	2.66
Classical (21 to 15)	10.10	2.66
Small degree (15 to 4)	6.18	3.00
Total CPU time (years)	219.75	

TABLE 1. CPU times of each stage of the computation of discrete logarithms in $\mathbb{F}_{3^{6\cdot 509}}$.

3.2.1. *Quadratics.* There are $266,086 \approx 2^{18}$ linear and irreducible quadratics. Relation generation took 2.4 CPU hours using Magma on an Intel i7-3930K 3.20GHz CPU core. The resulting sparse system of linear equations was solved using our C implementation of Wiedemann's algorithm; the computation took $4,320$ CPU hours on Intel Xeon E5-2658 v2 2.40 GHz CPU cores.

3.2.2. *Cubics.* For every $\gamma \in \mathbb{F}_q^*$, $\mathcal{B}_{3,\gamma}$ has size exactly $177,390$. The total relation generation running time is $1,232$ CPU hours using Magma on Intel i7-3930K 3.20GHz CPU cores. The resulting 728 sparse systems of linear equations were solved using our C implementation of Wiedemann's algorithm. Each linear system was solved in parallel on 7 ABACUS cores.[3] The 728 linear systems were solved simultaneously using 5096 ABACUS cores. The total execution time was $379,142$ CPU hours. This time, and also the time for the linear algebra for the quartics (see §3.2.3), was more than expected in part because ABACUS was still running in an experimental phase and the machine was under-clocked

[3]The dominant operation in Wiedemann's algorithm is the computation of Av where A is a sparse matrix with small entries and v is a vector. This operation can be parallelized on k cores by partitioning the rows of A into k submatrices $A_1, A_2, \dots, A_k$ and computing A_iv on the ith core.

to prevent over-heating. The increased CPU time did not have a significant impact on the total calendar time for the discrete log computation because of the large number of cores that we were at our disposal.

The logarithms were stored in files whose total size is 26.4 gigabytes.

3.2.3. *Quartics.* We computed logarithms of $s = 29$ families $\mathcal{B}_{4,u^i}$, $0 \le i \le 28$. We elected not to used the technique of iteratively decreasing the size of the corresponding linear systems. Consequently, each subfamily of quartics yielded a linear system of dimension approximately 55,050. The total relation generation running time was $35,118$ CPU hours using Magma on Intel Xeon E5-2650 v2 2.60GHz CPU cores. The resulting $29 \times 729 = 21,141$ sparse systems of linear equations were solved using our C implementation of Wiedemann's algorithm in $829,573$ CPU hours on ABACUS. Each linear system was solved in parallel on 2 cores. We used approximately 5000 cores to solve all 21,141 linear systems. The logarithms of a family were stored in files whose total size is 20.4 gigabytes. The total size of the files of factor base logarithms is 618 gigabytes. Note that the total size of the files for logarithms of all polynomials of degree ≤ 4 would be about 14.9 terabytes.

Theorem 4 of [30] shows that the expected number of degree-d irreducible factors of a randomly selected degree-n polynomial over $\mathbb{F}_q$ is approximately $1/d$. Using this result, we computed the expected number of degree-4 elements obtained after a descent of a polynomial of degree in the interval $[5, 15]$. We then used Table 4 to estimate the expected number of irreducible quartics that result from all the descent steps. These estimates are shown in Table 2; the total number of irreducible quartics is $2^{31.15}$.

Degree	5	6	7	8	9	10	11	12	13	14	15
Number of degree-4 polynomials	$2^{14.18}$	$2^{14.26}$	$2^{21.27}$	$2^{16.13}$	$2^{22.11}$	$2^{23.88}$	$2^{28.54}$	$2^{23.97}$	$2^{28.74}$	$2^{24.88}$	$2^{30.45}$

TABLE 2. Expected number of irreducible quartics resulting from all the Gröbner bases and zigzag descent steps for each degree in $[5, 15]$.

To compute the logarithm of a polynomial in $\mathcal{B}_{4,u^i}$, $i > 28$, we used the family-based descent method described in §2.2.4. The descent considered the families in the order $\mathcal{B}_3$, $\mathcal{B}_{4,u^0}$, $\mathcal{B}_{4,u^1}, \ldots, \mathcal{B}_{4,u^{28}}$. Table 3 shows, for each $i \in [0, 28]$, the inverse of the probability that a randomly-selected irreducible quartic descends based on that family. In particular, note that the probability that a quartic fails to descend is less than $2^{-38.2}$. Since the reciprocal of the failure probability is $\gg 2^{31.15}$, the number of quartics encountered that fail to descend is expected to be very small. In the event that a quartic fails to descent, the computation that yielded that quartic is repeated (with different parameters).

The computers available to us had at most 256 gigabytes of RAM. Thus, only the logarithms of the cubics and the logarithms of the first 10 quartics families $\mathcal{B}_{4,u^0}, \mathcal{B}_{4,u^1}, \ldots, \mathcal{B}_{4,u^9}$ were placed in RAM, and the logarithms of the remaining quartic families were stored in hard disk (HD), which is much slower to access than virtual memory (VM). Since many copies of the Magma code will be executed in parallel, each of which will be accessing the same logarithm files, the memory accesses have to be carefully scheduled to avoid traffic congestion. In addition, we had to deal with some restrictions on Magma's file reading capabilities (for example, whether the files are stored in hexadecimal encoding or binary

$\mathcal{B}_3$	$2^{00.8297}$	$\mathcal{B}_{4,u^0}$	$2^{02.1020}$	$\mathcal{B}_{4,u^1}$	$2^{03.3742}$	$\mathcal{B}_{4,u^2}$	$2^{04.6466}$	$\mathcal{B}_{4,u^3}$	$2^{05.9189}$
$\mathcal{B}_{4,u^4}$	$2^{07.1912}$	$\mathcal{B}_{4,u^5}$	$2^{08.4635}$	$\mathcal{B}_{4,u^6}$	$2^{09.7358}$	$\mathcal{B}_{4,u^7}$	$2^{11.0081}$	$\mathcal{B}_{4,u^8}$	$2^{12.2803}$
$\mathcal{B}_{4,u^9}$	$2^{13.5526}$	$\mathcal{B}_{4,u^{10}}$	$2^{14.8250}$	$\mathcal{B}_{4,u^{11}}$	$2^{16.0972}$	$\mathcal{B}_{4,u^{12}}$	$2^{17.3695}$	$\mathcal{B}_{4,u^{13}}$	$2^{18.6418}$
$\mathcal{B}_{4,u^{14}}$	$2^{19.9141}$	$\mathcal{B}_{4,u^{15}}$	$2^{21.1864}$	$\mathcal{B}_{4,u^{16}}$	$2^{22.4587}$	$\mathcal{B}_{4,u^{17}}$	$2^{23.7310}$	$\mathcal{B}_{4,u^{18}}$	$2^{25.0033}$
$\mathcal{B}_{4,u^{19}}$	$2^{26.2756}$	$\mathcal{B}_{4,u^{20}}$	$2^{27.5479}$	$\mathcal{B}_{4,u^{21}}$	$2^{28.8202}$	$\mathcal{B}_{4,u^{22}}$	$2^{30.0925}$	$\mathcal{B}_{4,u^{23}}$	$2^{31.3648}$
$\mathcal{B}_{4,u^{24}}$	$2^{32.6371}$	$\mathcal{B}_{4,u^{25}}$	$2^{33.9094}$	$\mathcal{B}_{4,u^{26}}$	$2^{35.1817}$	$\mathcal{B}_{4,u^{27}}$	$2^{36.4540}$	$\mathcal{B}_{4,u^{28}}$	$2^{37.7263}$

TABLE 3. For every family from $\mathcal{B}_3$, $\mathcal{B}_{4,u^i}$, $i \in [0, 28]$, the inverse of the probability for a random irreducible quartic to descend based on that family.

encoding) and with limits on the total number of open files permitted on Linux. In the end, the average time to descend a randomly selected quartic was found to be 0.0614 seconds of CPU time (0.0640 seconds of real time) on a 20-core Intel Xeon E5-2658 v2 2.40GHz machine with 256 gigabytes of RAM. For further details, see [2].

3.2.4. *Continued-fractions descent.* The two degree-254 polynomials yielded 22 irreducible factors with 2 of degree 40, 1 of degree 39, 1 of degree 38, 1 of degree 37, and 7 of degree in the interval $[22, 35]$. The computation took $446,768$ CPU hours on CPU cores with average frequency 2.87 GHz (270 cores of different frequencies were used in this stage).

3.2.5. *Classical descent.* In the first classical descent phase, 255 polynomials of degree ≤ 21 were obtained from the 12 polynomials of degree ≥ 22. These computations took $86,323$ CPU hours on CPU cores with average frequency 2.66 GHz (390 cores of different frequencies were used in the two phases of the classical descent).

The second classical descent phase was used on the 84 polynomials of degree ≥ 16 arising from the first phase, to obtain polynomials of degree ≤ 15. These computations took $88,452$ CPU hours.

The number of polynomials of each degree in $[5, 15]$ that were obtained from the continued-fractions and classical descents is shown in Table 4.

Degree	5	6	7	8	9	10	11	12	13	14	15
Number of polynomials	101	107	94	98	92	116	137	123	155	173	213

TABLE 4. Number of polynomials of each degree in the interval $[5, 15]$ obtained after the continued-fractions and classical descents. The total number of polynomials is 1409.

3.2.6. *Small-degree descent.* In the last descent stage, the 1409 polynomials of degrees in $[5, 15]$ that resulted from the continued-fractions and classical descents should have their logarithms expressed in terms of logarithms of elements in the factor base, namely, in $\mathcal{B}_2$, $\mathcal{B}_3$ and $\mathcal{B}_{4,u^i}$, $i \in [0, 28]$.

The Gröbner bases descent expresses the logarithm of a degree-D element, $D \in [5, 15]$, as a linear combination of logarithms of polynomials of degree $\le d$ where $d = \lfloor D/2 \rfloor + 2$. This is the best that can be done because of condition (5).

Gröbner bases descent was used on polynomials of odd degree. For those of even degree the zigzag descent was employed (except for degree-14 polynomials, see below) because of its more aggressive descent character. Indeed, a polynomial of degree $2d$, $d > 2$, is related with polynomials of degree d using the zigzag descent instead of polynomials of degree $d + 2$ when using the Gröbner bases descent. The degree-4 polynomials which are not in the factor base are descended using the classical or family-based Gröbner bases descent combined with the Frobenius strategy.

As mentioned in §2.6, the zigzag descent is successful for only 50% of degree-6 polynomials. For the remainder, we used a hybrid Gröbner bases-zigzag descent. In this hybrid descent, a degree-6 polynomial is lifted to the quadratic extension of $\mathbb{F}_{3^6}$, where it splits into two cubics. Over $\mathbb{F}_{3^{12}}$, we adapted the Gröbner bases descent in §2.5 and used it to perform a 3-to-2 descent on one of the two degree-3 polynomials. Then, using the polynomial norm as in §2.6, we obtained the logarithm of the degree-6 polynomial expressed in term of logarithms of polynomials of degree (at most) 4. This strategy allowed us to avoid the more costly 6-to-5 and then 5-to-4 Gröbner bases descent steps (recall that each of these descents has a branching factor of q).

We also employed the hybrid descent on the degree-14 polynomials to perform 14-to-8 descents instead of zigzag 14-to-7 descents. In fact, a complete descent is more costly on a degree-7 polynomial than on a degree-8 polynomial since in the former two Gröbner bases descent stages, 7-to-5 then 5-to-4, are needed whereas only one zigzag 8-to-4 descent stage is needed in the latter.

Table 5 lists the (scaled) times for computing the logarithms of all polynomials of degrees in the interval $[5, 15]$ that arose from the continued-fractions and classical descents stages.

Degree	5	6	7	8	9	10	11	12	13	14	15
Total	$2^{10.21}$	$2^{10.29}$	$2^{17.30}$	$2^{12.16}$	$2^{18.14}$	$2^{19.92}$	$2^{24.57}$	$2^{20.00}$	$2^{24.78}$	$2^{20.91}$	$2^{26.48}$
Average	$2^{3.55}$	$2^{3.55}$	$2^{10.69}$	$2^{5.64}$	$2^{11.62}$	$2^{13.06}$	$2^{17.47}$	$2^{13.06}$	$2^{17.50}$	$2^{13.48}$	$2^{18.75}$

TABLE 5. Total and average CPU times in seconds to obtain the logarithms of all the polynomials of degrees in $[5, 15]$ that resulted from the continued-fractions and classical descents. The times assume that an Intel Xeon E5-2658 v2 2.40 GHz machine with 256 gigabytes of RAM is used.

Remark 1. In hindsight, the total running time for computing logarithms in $\mathbb{F}_{3^{6\cdot 509}}$ can be reduced substantially with two modifications to the algorithm. First, as mentioned at the end of §2.2.3, the linear systems associated with families of quartics can be reduced after discrete logarithms of a few families have been computed. Second, a 508-to-32 Guillevic descent (see §4.1) could be used instead of the 508-to-40 continued-fractions descent. The estimated cost of the 508-to-32 Guillevic descent is only 0.05 CPU years as compared to the estimated cost of 31.8 CPU years for the 508-to-40 continued-fractions descent. Moreover, the estimated costs of the subsequent 32-to-19 and 19-to-15 classical descents and the 15-to-4 small-degree descents are also substantially lower — 0.93, 0.99 and 2.13 CPU years, respectively (cf. Table 1).

4. Discrete logarithms in $\mathbb{F}_{3^{6\cdot 709}}$

Recall that the Frobenius representation of $\mathbb{F}_{3^{6n}}$ requires a degree-n irreducible factor of $h_1(X)\cdot X^q - h_0(X)$ over $\mathbb{F}_q$ where $\max(\deg h_0, \deg h_1) = 2$. Now, $n = 709$ is the largest prime ≤ 731 for which there is a supersingular elliptic curve E over $\mathbb{F}_{3^n}$ with $r = \#E(\mathbb{F}_{3^n})$ a prime. More precisely, we have $r = 3^{709} - 3^{355} + 1 = \#E(\mathbb{F}_{3^{709}})$ where E is the supersingular elliptic curve $Y^2 = X^3 - X - 1$ defined over $\mathbb{F}_3$. The Weil and Tate pairings can be used to embed $E(\mathbb{F}_{3^{709}})$ in the multiplicative group of the 6743-bit field $\mathbb{F}_{3^{6\cdot 709}}$. Thus, we are interested in computing $x = \log_g h \bmod r$, where g is a generator of $\mathbb{F}^*_{3^{6\cdot 709}}$ and h is an element of the order-r subgroup of $\mathbb{F}^*_{3^{6\cdot 709}}$.

In §4.1 we describe a slight modification of a descent method proposed by Guillevic [25] that is considerably more effective than the continued-fractions descent. Then, in §4.2, we demonstrate that Guillevic's descent method can be utilized to compute discrete logarithms in the cryptographically-interesting field $\mathbb{F}_{3^{6\cdot 709}}$ with essentially the same resources as we expended on the $\mathbb{F}_{3^{6\cdot 509}}$ discrete logarithm computation. Thus, we conclude that discrete logarithms in the cryptographically-interesting field $\mathbb{F}_{3^{6\cdot 709}}$ can be feasibly computed today.

4.1. **Guillevic descent.**

Let $q = 3^6$ and let r be a prime divisor of $\Phi_6(3^n)$, where $\Phi_6(X)$ denotes the 6th cyclotomic polynomial. Suppose that elements of the finite field $\mathbb{F}_{q^n}$ are represented as polynomials of degree at most $n-1$ over $\mathbb{F}_q$, with multiplication performed modulo a degree-n irreducible polynomial. Let g be a generator of $\mathbb{F}^*_{q^n}$, and let $h \in \mathbb{F}^*_{q^n}$. We wish to determine $x = \log_g h \bmod r$.

Since h is an arbitrary element of $\mathbb{F}^*_{q^n}$, its degree can be as high as $n-1$. Without loss of generality, we can suppose that h has degree exactly $n-1$. Guillevic observed that if $h' = hv$, where v is an element of the proper subfield $\mathbb{F}_{3^{3n}}$ of $\mathbb{F}_{q^n}$, then

$$\log_g h' \equiv \log_g h \pmod{r}.$$

Her descent method consists of searching for v until h' has degree $n' \approx n/2$ and is smooth with respect to some smoothness bound m.

Let $n' = \lfloor n/2 \rfloor + c$, where c is chosen so that $3^{6n'-3n} \gg q^{n'}/N_q(m, n')$, the right hand side of the inequality being the reciprocal of the proportion of degree-n' polynomials over $\mathbb{F}_q$ that are m-smooth. Let $\{1, w, w^2, ..., w^{3n-1}\}$ be a basis for $\mathbb{F}_{3^{3n}}$ over $\mathbb{F}_3$. Thus, we can write $v = v_0 + v_1 w + \cdots + v_{3n-1}w^{3n-1}$, where $v_i \in \mathbb{F}_3$. Let H be the $6n \times 3n$ matrix over $\mathbb{F}_3$ whose columns are the coefficients of $h, wh, w^2h, \ldots, w^{3n-1}h$. Here, if $w^i h = h_0 + h_1 X + \cdots + h_{n-1}X^{n-1}$ with $h_j \in \mathbb{F}_q$, then the column vector corresponding to $w^i h$ is $(h_0, h_1, \ldots, h_{n-1})$ where each h_j is written as a length-6 vector over $\mathbb{F}_3$. Thus, we wish to find vectors v such that $h' = Hv$ is an $\mathbb{F}_3$-vector corresponding to a monic polynomial of degree n' over $\mathbb{F}_q$.

Now, let H' be the $6(n-n') \times 3n$ matrix consisting of the last $6(n-n')$ rows of H. We expect that H' has full row rank (otherwise we can randomize h and repeat). Thus, each of the $3^{6n'-3n}$ solutions v to the matrix equation $H'v = e$, with e being the unit vector having a 1 in it first position, yields a monic polynomial $h' = Hv$ of degree n' over $\mathbb{F}_q$. These polynomials are tested for m-smoothness until an m-smooth polynomial is found.

The expected running time is $S_q(m,n')\cdot q^{n'}/N_q(m,n')$, the cost of the linear algebra being negligible.

As proof-of-concept, we used the new descent method to write a degree-708 polynomial $h \in \mathbb{F}_{3^{6\cdot 709}}$ in terms of a 52-smooth degree-358 monic polynomial h' (so $n = 709$, $c = 4$, $n' = 358$, and $m = 52$). The estimated cost of finding such an h' is only $2^{43.8}M_q$. We found a 52-smooth h' in about 245 CPU hours, after testing about $2^{19.6}$ candidates h' (the search was implemented in Magma using a sub-optimal procedure for smoothness testing). In contrast, the expected cost of continued-fractions descent to write h as w_1/w_2 where each w_i is 52-smooth and has degree approximately 354 is $2^{62.5}M_q$; this computation is feasible but only with a considerable effort.

4.2. **Estimates.** Let $\mathbb{F}_{3^6} = \mathbb{F}_3[u]/(u^6 + 2u^4 + u^2 + 2u + 2)$, $h_0(X) = u^{10}X + 1$, and $h_1(X) = X^2 + u^{110}X$. Then $h_1(X)\cdot X^q - h_0(X)$ has a degree-709 irreducible factor I_X, and the cofactor is an irreducible polynomial of degree 22. The field $\mathbb{F}_{3^{6\cdot 709}}$ can be represented as $\mathbb{F}_{3^6}[X]/(I_X)$.

To find $x = \log_g h \bmod r$, we first use Guillevic's descent to find a degree-358 monic polynomial h' that is 40-smooth. The expected cost of this step is $2^{53.3}M_q$. (In contrast, the expected cost of continued-fractions descent to express h as the ratio of two 40-smooth degree-354 polynomials is $2^{81.3}M_q$.) Thus, the expected cost of the 708-to-40 Guillevic descent is less than the expected cost of the 508-to-40 continued-fractions descent. Furthermore, the 709-to-40 Guillevic descent will yield fewer polynomials of degree ≤ 40 than the 508-to-40 continued-fractions descent.

The remainder of the discrete logarithm computation in $\mathbb{F}_{3^{6\cdot 709}}$ proceeds in the same way as the $\mathbb{F}_{3^{6\cdot 509}}$ discrete logarithm computation, except that we have to work modulo the 1124-bit prime r instead of an 804-bit prime. The larger r will only have a slight impact on the cost of the linear algebra. Hence, we can conclude that discrete logarithms in $\mathbb{F}_{3^{6\cdot 709}}$ can be computed using essentially the same resources as we expended in the $\mathbb{F}_{3^{6\cdot 509}}$ computation.

5. Discrete logarithms in $\mathbb{F}_{3^{6\cdot 1429}}$

The supersingular elliptic curve $E : Y^2 = X^3 - X - 1$ defined over $\mathbb{F}_3$ has $\#E(\mathbb{F}_{3^{1429}}) = cr$, where $c = 7622150170693$ is a 43-bit cofactor and $r = (3^{1429} - 3^{715} + 1)/c$ is a 2223-bit prime. The Weil and Tate pairings can be used to embed the order-r subgroup of $E(\mathbb{F}_{3^{1429}})$ in the multiplicative group of the 13590-bit field $\mathbb{F}_{3^{6\cdot 1429}}$. Thus, we are interested in computing $x = \log_g h \bmod r$, where g is a generator of $\mathbb{F}^*_{3^{6\cdot 1429}}$ and h is an element of the order-r subgroup of $\mathbb{F}^*_{3^{6\cdot 1429}}$.

In §5.1, we show that discrete logarithms in the order-r subgroup of $\mathbb{F}^*_{3^{6\cdot 1429}}$ can be computed in time $2^{63.4}M_q$. In §5.2 we present our arguments that this computation is feasible using existing computer technology.

5.1. **Estimates.** Let $\mathbb{F}_{3^6} = \mathbb{F}_3[u]/(u^6 + 2u^4 + u^2 + 2u + 2)$, $h_0(X) = X + u^{28}$, and $h_1(X) = X^2 + u^{420}X$. Then $h_1(X^q)\cdot X - h_0(X^q)$ has a degree-1429 irreducible factor I_X, the cofactor being the product of six irreducible polynomials of degrees 1, 1, 1, 3, 12 and 12. The field $\mathbb{F}_{3^{6\cdot 1429}}$ can be represented as $\mathbb{F}_{3^6}[X]/(I_X)$. This *dual Frobenius* *representation*, introduced in [22], has the useful property that

$$X \equiv \left(\frac{h_0(X)}{h_1(X)}\right)^q \pmod{I_X}.$$

The estimated costs of computing discrete logarithms in $\mathbb{F}_{3^{6\cdot 1429}}$ are given in Table 6 and explained in §§5.1.1–5.1.5.

Finding logarithms of polynomials of degree ≤ 4	
Degrees 1 and 2	$2^{50.7}M_q$
Degree 3	$2^{56.9}M_q$
Degree 4 (36 families)	$2^{56.3}M_q$
Descent	
Guillevic (1428 to 71)	$2^{62.4}M_q$
Classical (71 to 32)	$2^{61.8}M_q$
Classical (31 to $\{1,\dots,16,18,20,22,24,28,32\}$)	$2^{59.2}M_q$
Small degree ($\{5,\dots,16,18,20,22,24,28,32\}$ to 4)	$2^{60.0}M_q$
Total cost	$2^{63.4}M_q$

TABLE 6. Estimated costs of the main steps for computing discrete logarithms in $\mathbb{F}_{3^{6\cdot 1429}}$.

Remark 2. To gauge the accurateness of our $\mathbb{F}_{3^{6\cdot 1429}}$ estimates, we generated estimates for the $\mathbb{F}_{3^{6\cdot 509}}$ computation using the same methodology as for the $\mathbb{F}_{3^{6\cdot 1429}}$ estimates. The cost estimates (in CPU years) for the main steps in the $\mathbb{F}_{3^{6\cdot 509}}$ computation are 15.7 (linear algebra for cubics), 16.4 (linear algebra for 29 families of quartics), 31.8 (continued-fractions descent), 7.4 (first classical descent), 7.9 (second classical descent), and 4.7 (small-degree descent). These estimates compare well with the observed times in Table 1 with the exception of the linear algebra as explained in §3.2.2.

5.1.1. *Degrees 1, 2, 3, 4.* The logarithms of irreducible polynomials of degrees 1, 2, 3 and 4 (36 families) are obtained using the Joux-Pierrot approach as described in §2.2. The cost ratio $A_r/M_q = 8$ yields the cost estimates in Table 6.

We chose 36 quartic families to ensure that the probability of an irreducible quartic failing to descend using the family-based Gröbner bases descent method (see §2.2.4) is small. If we precompute logarithms of 36 quartic families, then this failure probability is less than $2^{-47.15}$. This can be considered to be sufficiently small since we expect to descend about $2^{36.94}$ irreducible quartics during the entire computation.

In fact, we only need to compute the logarithms of 18 families of quartics since we then get the logarithms of another 18 families for free. To see this, observe that the coefficients of $h_0(X)$ and $h_1(X)$ are elements of $\mathbb{F}_{3^3}$. Thus,

$$X^{3^{3\cdot 1429}} \equiv X \pmod{I_X}$$

and so the order-2 $\mathbb{F}_{3^{6\cdot 1429}}$-automorphism $\sigma : \alpha \mapsto \alpha^{3^{3\cdot 1429}}$ fixes X. Now, if $\gamma \in \mathbb{F}_{3^6} \setminus \mathbb{F}_{3^3}$, then σ gives a one-to-one correspondence between elements of the quartic families $\mathcal{B}_{4,\gamma}$

and $\mathcal{B}_{4,\sigma(\gamma)}$. Hence, if we compute the logarithms of all elements in $\mathcal{B}_{4,\gamma}$, then we can obtain the logarithms of elements $f \in \mathcal{B}_{4,\sigma(\gamma)}$ for free via

$$\log_g f \equiv 3^{3\cdot 1429} \log_g \sigma(f) \pmod{r}$$

since $\sigma(f) \in \mathcal{B}_{4,\gamma}$.

5.1.2. *Guillevic descent (1428 to 71).* Guillevic descent, as described in §4.1, is used to express the target element h in terms of a 71-smooth degree-720 polynomial h'. The matrix H' has dimensions 4254×4287, and so solving $H'v = e$ takes negligible time. The expected number of candidates h' to be tested for 71-smoothness is $2^{35.5}$. Since the cost of testing a degree-720 polynomial for 71-smoothness is $S_q(71, 720) = 2^{26.9} M_q$, the expected cost of the Guillevic descent is $2^{62.4} M_q$.

In order to obtain a tighter estimate for the running time of the entire descent, we undertake a top-down analysis of the expected number of polynomials of each degree that are produced after each descent step. For this analysis, we use the generating function $F_{k,m}(u, z)$ for m-smooth monic polynomials over $\mathbb{F}_q$, where z marks the degree of a polynomial and u marks distinct degree-k monic irreducible factors of the polynomial. It is easy to see that

$$F_{k,m}(u,z) = \left(\prod_{\substack{i=1 \\ i \neq k}}^{m} \left(\frac{1}{1 - z^i} \right)^{I_i(q)} \right) \left(1 + \frac{u z^k}{1 - z^k} \right)^{I_k(q)},$$

where $I_i(q)$ denotes the number of monic irreducible polynomials of degree i over $\mathbb{F}_q$. Then the average number of distinct degree-k monic irreducible factors of an m-smooth degree-n monic polynomial over $\mathbb{F}_q$ is

$$c_{k,m,n} = \frac{[z^n] \left. \frac{\partial F_{k,m}}{\partial u} \right|_{u=1}}{[z^n] F_{k,m}(1, z)}, \tag{6}$$

where $[\,]$ denotes the coefficient operator. For any given k, m and n, $c_{k,m,n}$ can be obtained by using a symbolic algebra package such as Maple [33]. Thus, for each $k \in [1, 71]$, we can deduce the average number $c_{k,71,720}$ of degree-k polynomials obtained after the 1428-to-71 Guillevic descent.

5.1.3. *Classical descent (71 to 32).* Suppose that one wishes to express a degree-D polynomial Q over $\mathbb{F}_q$ in terms of irreducible polynomials of degrees at most m. In the alternative classical descent described in [4, §3.5], parameters $s \in [0, 6]$ and $\delta \geq 1$ are selected. One then searches for a pair of polynomials (R_1, R_2) such that $Q \mid R_2$, $\deg R_1 = t_1 \approx 2(\lfloor D/2 \rfloor + \delta) 3^{6-s} + 1$, $\deg R_2 = t_2 \approx (\lfloor D/2 \rfloor + \delta) + 3^s$, and both R_1 and R_2/Q are m-smooth. The expected cost of the classical descent is

$$CD1_{D,m} = \frac{q^{t_1}}{N_q(m, t_1)} \cdot \frac{q^{t_2 - D}}{N_q(m, t_2 - D)} \cdot \min(S_q(m, t_1), S_q(m, t_2 - D)).$$

For the 71-to-32 classical descent, we selected $s = 5$ and $\delta = 3$. Then, the expected cost of the 71-to-32 classical descent is

$$\sum_{D=33}^{71} (CD1_{D,32} \cdot c_{D,71,720}) \approx 2^{61.8} M_q.$$

We also use the expression (6) to estimate the expected number d_k of polynomials of each degree $k \in [1, 32]$ at the conclusion of this descent.

5.1.4. *Classical descent (31 to S).* Let $S = \{5, 6, \ldots, 16, 18, 20, 22, 24, 28, 32\}$, and $D \in \mathcal{D}$ where $\mathcal{D} = \{17, 19, 21, 23, 25, 26, 27, 29, 30, 31\}$. Rationale for the choice of S is provided in §5.1.5. Suppose that one wishes to express a degree-D polynomial Q over $\mathbb{F}_q$ in terms of irreducible polynomials with degrees in S. In the classical descent described in [4, §3.5], parameters $s \in [0, 6]$ and $\delta \geq 1$ are selected. One then searches for a pair of polynomials (R_1, R_2) such that $Q \mid R_1$, $\deg R_1 = t_1 \approx (\lfloor D/2 \rfloor + \delta) + 2 \cdot 3^{6-s}$, $\deg R_2 = t_2 \approx (\lfloor D/2 \rfloor + \delta)3^s + 1$, and both R_1/Q and R_2 are S-smooth (i.e., all irreducible factors of R_1/Q and R_2 have degrees in S). The expected cost of the classical descent is

$$CD2_{D,S} = \frac{q^{t_1 - D}}{N_q(S, t_1 - D)} \cdot \frac{q^{t_2}}{N_q(S, t_2)} \cdot \min(S_q(32, t_1 - D), S_q(32, t_2)),$$

where

$$N_q(S, n) = [z^n] \prod_{i \in S} (1 - z^i)^{-I_i(q)}$$

is the number of S-smooth degree-n monic polynomials over $\mathbb{F}_q$.

We selected $s = 2$ and $\delta = 2$. Then, the expected cost of the 31-to-S classical descent is

$$\sum_{D \in \mathcal{D}} (CD2_{D,S} \cdot d_D) \approx 2^{59.2} M_q.$$

5.1.5. *Small-degree descent (S to 4).* We used the expression (6) to estimate the expected number of polynomials e_D of each degree $D \in S$ at the conclusion of the second classical descent. These numbers are listed Table 7. For each polynomial of degree $D \in [5, 15]$,

D	5	6	7	8	9	10	11	12	13
e_D	73	68	65	63	63	63	64	65	67
$\log_2$(cost)	36.7	36.6	43.7	38.6	44.6	46.0	50.5	46.1	50.6
D	14	15	16	18	20	22	24	28	32
e_D	69	72	75	83	92	103	117	151	200
$\log_2$(cost)	46.6	51.9	48.4	54.5	56.1	56.4	56.4	57.2	59.3

TABLE 7. M_q costs of performing all the D-to-4 descents for each $D \in S$.

the D-to-4 descent is performed using the strategies in §3.2.6. For each polynomial of degree $D \in \{16, 18, 20, 24, 28, 32\}$, one first performs a D-to-$D/2$ zigzag descent; in the analysis we assume that the descent yields q irreducible polynomials of degree $D/2$. A degree-22 polynomial is lifted to $\mathbb{F}_{q^2}$, resulting in a degree-11 irreducible polynomial and its conjugate. An 11-to-8 Gröbner-bases descent is performed, and the resulting polynomials are projected down to $\mathbb{F}_q$. One expects to obtain q/i polynomials of degree $2i$, for each $i \in 8$ [30]. The even degrees 26 and 30 are omitted from S because of the relative high cost of performing 13-to-4 and 15-to-4 descents (see Table 5). The costs of all the descents are given in Table 7. The total cost is $2^{60.0} M_q$.

5.2. **Feasibility.** In this section we argue that the computation outlined in §5.1 is feasible today, even though it is just beyond the reach of the computer resources available to the authors of this paper.

We assume that we have access to a 9000-core machine $\mathcal{A}$ such as ABACUS [1], where each core has 16 gigabytes of RAM. In addition, we assume that we have access to a 1500-core machine $\mathcal{B}$ with 1 terabyte of shared RAM. We further assume that both machines can execute $2^{27} M_q$ per second; we achieved these speeds in our experiments using a look-up table approach.

Table 8 shows the estimated calendar time for computing a discrete logarithm in $\mathbb{F}_{3^{6\cdot 1429}}$. All computations are performed on machine $\mathcal{A}$ except for the small-degree descents which

Computation	# cores	# days
Degree-3 logarithms	5824	2
Degree-4 logarithms	9000	1
Guillevic descent	9000	59
First classical descent	9000	39
Second classical descent	9000	7
Small degree descent	1500	65
Total time		173

Table 8. Estimated calendar time for computing a discrete logarithm in $\mathbb{F}_{3^{6\cdot 1429}}$ using machines $\mathcal{A}$ and $\mathcal{B}$.

are performed on machine $\mathcal{B}$. The 728 matrices for degree-3 logarithms are solved simultaneously, with each linear system assigned to 8 cores. Then, 4,500 of the 13,122 matrices for degree-4 logarithms are solved simultaneously using 2 cores per matrix. The next 4,500 matrices are solved after that, and finally the remaining 4,122 matrices. The estimated size of the files containing logarithms of cubics is 38.3 gigabytes (only half the logarithms have to be stored thanks to the automorphism σ). The estimated size of the files containing logarithms of one family of quartics is 59.2 gigabytes. Thus, the logarithms of cubics and 15 quartic families, whose total size is 926.3 gigabytes, can be stored in shared RAM. Since the family-based Gröbner bases descent can be performed with respect to 30 quartic families without resorting to the quartic families stored in hard disk, we can reasonably expect that the time to compute the logarithm of a randomly-selected irreducible quartic to be no mor than the time for this operation in the $\mathbb{F}_{3^{6\cdot 509}}$ computation (where only 10 quartic families were stored in RAM), namely 0.064 seconds. Since we expect to perform $2^{36.94}$ such descents in total, we obtain an upper bound of 65 days on the time for all the small-degree descents.

The total estimated calendar time for computing a logarithm in $\mathbb{F}_{3^{6\cdot 1429}}$ is 173 days. It would be worthwhile to consider alternate descent strategies to reduce the expected time to the extent that the computation could be performed with relatively modest computational resources.

Acknowledgements

We would like to thank the ABACUS-CINVESTAV staff especially Isidoro Gitler, Daniel Ortiz-Gutiérrez and Omar Nieto-Ayalo, and the system managers Santiago Domínguez-Domínguez (CINVESTAV) and Christopher Calzonetti (University of Waterloo), for granting us computer access and for their technical support. The authors of this paper affiliated to CINVESTAV would like to acknowledge partial support from CONACyT project 180421.

References

[1] ABACUS Supercomputer – Cinvestav, http://www.abacus.cinvestav.mx/.

[2] G. Adj, "Logaritmo discreto en campos finitos de característica pequeña: atacando la criptografía basada en emparejamientos de Tipo 1", Ph.D. thesis, CINVESTAV-IPN, 2016. Available at https://www.cs.cinvestav.mx/TesisGraduados/2016/TesisGoraAdj.pdf.

[3] 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.

[4] 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*, 32 (2015), 148–170.

[5] G. Adj, A. Menezes, T. Oliveira and F. Rodríguez-Henríquez, "Computing discrete logarithms in $\mathbb{F}_{3^{6\cdot 137}}$ and $\mathbb{F}_{3^{6\cdot 163}}$ using Magma", *WAIFI 2014*, LNCS 9061 (2015), 3–22.

[6] 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.

[7] P. Barreto, S. Galbraith, C. Ó hÉigeartaigh and M. Scott, "Efficient pairing computation on supersingular abelian varieties", *Designs, Codes and Cryptography*, 42 (2007), 239–271.

[8] 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.

[9] 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.

[10] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing", *Advances in Cryptology – CRYPTO 2001*, Lecture Notes in Computer Science, 2139 (2001), 213–229.

[11] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing", *Journal of Cryptology*, 17 (2004), 297–319.

[12] I. Canales-Martínez, "Implementación eficiente de prueba de suavidad para polinomios", Tesis de Maestría, CINVESTAV-IPN, 2015. Available at http://delta.cs.cinvestav.mx/~francisco/Thesis_IAC.pdf.

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

[14] The Cunningham Project, http://homes.cerias.purdue.edu/~ssw/cun/.

[15] J. Faugère, "A new efficient algorithm for computing Gröbner bases (F_4)", *Journal of Pure and Applied Algebra*, 139 (1999), 61–88.

[16] 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.

[17] S. Galbraith, "Supersingular curves in cryptography", *Advances in Cryptology – ASIACRYPT 2001*, LNCS 2248 (2001), 495–513.

[18] S. Galbraith, K. Harrison and D. Soldera, "Implementing the Tate pairing", *Algorithmic Number Theory – ANTS 2002*, LNCS 2369 (2002), 324–337.

[19] 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.

[20] F. Göloğlu, R. Granger, G. McGuire and J. Zumbrägel, "Solving a 6120-bit DLP on a desktop computer", *Selected Areas in Cryptography – SAC 2014*, LNCS 8282 (2014), 136–152.

[21] D. Gordon, "Discrete logarithms in $GF(p)$ using the number field sieve", *SIAM Journal on Discrete Mathematics*, 6 (1993), 124–138.

[22] 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.

[23] 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 $\mathbb{F}_{2^{12\cdot 367}}$)", Cryptology ePrint Archive: Report 2014/119, http://eprint.iacr.org/2014/119.

[24] R. Granger, T. Kleinjung and J. Zumbrägel, "On the powers of 2", Cryptology ePrint Archive: Report 2014/300, https://eprint.iacr.org/2014/300.

[25] A. Guillevic, "Faster individual discrete logarithms in non-prime finite fields with the NFS and FFS algorithms", Cryptology ePrint Archive: Report 2016/684, https://eprint.iacr.org/2016/684.

[26] 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.

[27] A. Joux and R. Lercier, "The function field sieve in the medium prime case", *Advances in Cryptology – EUROCRYPT 2006*, LNCS 4004 (2006), 254–270.

[28] A. Joux and C. Pierrot, "Improving the polynomial time precomputation of Frobenius representation discrete logarithm algorithms", *Advances in Cryptology – ASIACRYPT 2014*, LNCS 8873 (2014), 378–397.

[29] A. Joux and C. Pierrot, "Technical history of discrete logarithms in small characteristic finite fields", *Designs, Codes and Cryptography*, 78 (2016), 73–85.

[30] A. Knopfmacher and J. Knopfmacher, "Counting irreducible factors of polynomials over a finite fields", *Discrete Mathematics*, 112 (1993), 103–118.

[31] A. Lenstra, "Unbelievable security: Matching AES security using public key systems", *Advances in Cryptology – ASIACRYPT 2001*, LNCS 2248 (2001), 67–86.

[32] Magma v2.19-7, http://magma.maths.usyd.edu.au/magma/.

[33] Maple 2016, http://www.maplesoft.com/products/maple/.

[34] 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.

[35] O. Schirokauer, "Discrete logarithms and local units", *Philosophical Transactions of the Royal Society London A*, 345 (1993), 409–423.

[36] D. Wiedemann, "Solving sparse linear equations over finite fields", *IEEE Transactions on Information Theory*, 32 (1986), 54–62.

Computer Science Department, CINVESTAV-IPN
E-mail address: `gora.adj@gmail.com`

Computer Science Department, CINVESTAV-IPN
E-mail address: `icanales@computacion.cs.cinvestav.mx`

Centro de Investigación en Computación del Instituto Politécnico Nacional
E-mail address: `nareli@cic.ipn.mx`

Department of Combinatorics & Optimization, University of Waterloo
E-mail address: `ajmeneze@uwaterloo.ca`

Computer Science Department, CINVESTAV-IPN
E-mail address: `thomaz.figueiredo@gmail.com`

Centro de Investigación en Computación del Instituto Politécnico Nacional
E-mail address: `lriveraz@gmail.com`

Computer Science Department, CINVESTAV-IPN
E-mail address: `francisco@cs.cinvestav.mx`