Solving discrete logarithms on a 170-bit MNT curve by pairing reduction [*]

Aurore Guillevic[5,6], François Morain[4,1], and Emmanuel Thomé[1,2,3]

[1] Institut national de recherche en informatique et en automatique (INRIA),
Villers-lès-Nancy & Saclay, France
[2] Université de Lorraine, Loria, UMR 7503, Vandoeuvre-lès-Nancy, France
[3] CNRS, Loria, UMR 7503, Vandoeuvre-lès-Nancy, France
[4] École Polytechnique/LIX, CNRS UMR 7161, Palaiseau, France
[5] University of Calgary, Alberta, Canada
[6] Pacific Institute for the Mathematical Sciences, CNRS UMI 3069, Canada
emmanuel.thome@inria.fr
morain@lix.polytechnique.fr
aurore.guillevic@ucalgary.ca

Abstract. Pairing based cryptography is in a dangerous position following the breakthroughs on discrete logarithms computations in finite fields of small characteristic. Remaining instances are built over finite fields of large characteristic and their security relies on the fact the embedding field of the underlying curve is relatively large. How large is debatable. The aim of our work is to sustain the claim that the combination of degree 3 embedding and too small finite fields obviously does not provide enough security. As a computational example, we solve the DLP on a 170-bit MNT curve, by exploiting the pairing embedding to a 508-bit, degree-3 extension of the base field.

Keywords: Discrete logarithm, finite field, number field sieve, MNT elliptic curve

1 Introduction

Pairings were introduced as a constructive cryptographic tool in 2000 by Joux [31], who proposed a one-round three participants key-exchange. Numerous protocols also based on pairings have been developed since. Beyond efficient broadcast protocols, prominent applications include Identity-Based Encryption [35,36,13], or short signatures [14].

The choice of appropriate curves and pairing definitions in the context of pairing-based cryptography has been the topic of many research articles. An important invariant is the degree of the embedding field, which measures the complexity of evaluating pairings, but is also related to the security of systems

---

[*] This is the authors' version. The final post-conference proceedings version will appear in the LNCS series at Springer http://link.springer.com/bookseries/558

(see Section 2 for more precisions). The first cryptographic setups proposed used pairings on supersingular curves of embedding degree 2 defined over a prime field $\mathbb{F}_p$, where p is 512-bit long, so that the pairing embeds into a 1024-bit finite field $\mathbb{F}_{p^2}$. Another early curve choice is a supersingular elliptic curve in characteristic three, defined over $\mathbb{F}_{3^{97}}$, of embedding degree 6 (used e.g. in [14], as well as various implementation proposals, e.g. [10]). More recent proposals define *pairing-friendly* ordinary curves over large characteristic fields, where constraining the embedding degree to selected values is a desired property [42,16,19,15,25,22,9,23].

Cryptanalysis of pairings can be attempted via two distinct routes. Either attack the discrete logarithm problem *on the curve*, or in the embedding field of the pairing considered. The former approach is rarely successful, given that it is usually easy to choose curves which are large enough to thwart $O(\sqrt{N})$ attacks such as parallel collision search or Pollard rho. Note however that derived problems such as the discrete logarithm *with auxiliary inputs* are much easier to handle, as shown by [46].

Attacking pairings via the embedding field is a strategy known as the Menezes–Okamoto–Vanstone [41] or Frey–Rück [24] attack, depending on which pairing is considered. Successful cryptanalyses that follow this strategy have been described in small characteristic. In [29], for a supersingular curve over $\mathbb{F}_{3^{97}}$, the small characteristic allowed the use of the Function Field Sieve algorithm [1], and the composite extension degree was also a very useful property. More recently, following recent breakthroughs for discrete logarithm computation in small characteristic finite fields [7,27], a successful attack has been reported on a supersingular curve over $\mathbb{F}_{2^{1223}}$, with degree-4 embedding [27]. The outcome of these more recent works is that curves in small characteristic are now definitively avoided for pairing-based cryptography.

As far as we know, there is no major record computation of discrete logarithms over pairing-friendly curves in large characteristic using a pairing reduction in the finite field. The pairing-friendly curves used in practice have a large embedding field of more than 1024 bits, where computing a discrete logarithm is still very challenging. A few curves in large characteristic have comparatively small embedding fields, and were identified as weak to this regard, although no practical computation to date demonstrated the criticality of this weakness. This includes the so-called MNT curves defined by Miyaji–Nakabayashi–Takano, e.g. [42, Example 1], an elliptic curve defined over a 170-bit prime p, and of 508-bit embedding field $\mathbb{F}_{p^3}$.

Despite the academic agreement on the fact that the pairing embedding fields for 170-bit MNT curves in general, and the one just mentioned in particular, are too small for cryptographic use, recent work like [2] has shown how cryptography relying on overly optimistic hardness assumptions can linger almost indefinitely in the wild. Demonstrating a practical break is key to really phasing out such outdated cryptographic choices. As far as we know, an MNT curve of low embedding degree 3 was never used in pairing-based cryptography, but was never attacked by a pairing reduction either. In this article, we present our attack over

the weak[7] MNT curve [42, Example 1], with p of 170 bits and $n = 3$. We report a discrete logarithm computation in the group of points of this curve by a pairing reduction, using only a moderate amount of computing power.

In order to attack the discrete logarithm problem in the embedding field, appropriate variants of the Number Field Sieve must be used. The crucial point is the adequate choice of a polynomial pair defining the Number Field Sieve setup, among the various choices proposed in the literature [32,40,33,6,8]. It is also important to arrange for the computation to take advantage of Galois automorphisms when available, both within sieving and linear algebra. Last, some care is needed in order to efficiently compute individual logarithms of arbitrary field elements.

This article is organized as follows. Section 2 reviews some background and notations for MNT curves on the one hand, and the Number Field Sieve (NFS) as a general framework on the other hand. Section 3 discusses in more detail the various possible choices of polynomial selection techniques for NFS. Section 4 discusses the details of the discrete logarithm computation with NFS, while Section 4.3 defines and solves an arbitrary challenge on the MNT curve.

2 Background and notations

2.1 Using pairing embedding to break DLP

We follow [12, chap. IX]. To fix notations, pairings are defined as follows, the map being bilinear, non-degenerate and computable in polynomial time in the size of the inputs.

$$e : \begin{cases} E(\mathbb{F}_p)[\ell] \times E(\mathbb{F}_{p^n})[\ell] \to \mu_\ell \subset \mathbb{F}_{p^n}^* \\ (P, Q) \mapsto e(P, Q). \end{cases} \quad (1)$$

Here, μ_ℓ is the subgroup of ℓ-th roots of unity, i.e. an element $u \in \mu_\ell$ satisfies $u^\ell = 1 \in \mathbb{F}_{p^n}^*$. The integer n is the so-called *embedding degree*, that is the smallest integer i for which the ℓ-torsion is contained in $\mathbb{F}_{p^i}$. It has a major impact on evaluating the difficulty of solving the DLP on the curve.

Let G_1 be a generator of $E(\mathbb{F}_p)[\ell]$ and P in the same group, whose discrete logarithm u is sought (so that $P = [u]G_1$). We choose a generator G_2 for $E(\mathbb{F}_{p^n})[\ell]$. We observe that

$$e(P, G_2) = e(G_1, G_2)^u$$

so that u can be recovered as the logarithm of $U = e(P, G_2)$ in base $T = e(G_1, G_2)$, where both elements belong to the subgroup of order ℓ of $\mathbb{F}_{p^n}^*$. Note that by construction, $\ell = O(p)$, so that the Number Field Sieve linear algebra phase has to be considered modulo ℓ, which is a priori much smaller than the largest prime order subgroup of $\mathbb{F}_{p^n}^*$, which has size $O(p^{\phi(n)})$.

---

[7] already described as weak in the paper by the authors

2.2 MNT curves

The Miyaji–Nakabayashi–Takano curves were designed in 2000 in [42] as the first example of *ordinary* curves with low embedding degree $n = 3$, 4, or 6. The curves were presented as a weak instance of ordinary elliptic curves that should be avoided in elliptic-curve cryptography because of the Menezes–Okamoto–Vanstone and Frey–Rück attacks [41,24] that embed the computation of a discrete logarithm from the group of points of the curve to the embedding field $\mathbb{F}_{p^n}$. At the 80-bit security level which was used in the 2000's, an elliptic curve of 160-bit prime order was considered safe, and of at least the same security as an 1024-bit RSA modulus. However for MNT curves over prime fields of 160 bits, the MOV and FR reduction attacks embed to finite fields of size 480, 640, or 960 bits, none of which should be considered as having a hard enough DLP. For these three cases and most of all for $n = 3$, computing a discrete logarithm in the embedding field is considerably easier than over the elliptic curve. The conclusion of the MNT paper was to advise developers to systematically check that the embedding degree of an elliptic curve is large enough, to avoid pairing reduction attacks. The authors also mentioned as a constructive use of their curves the prequel work of Kasahara, Ohgishi, and Sakai on identity-based encryption using pairings [35,36]. Some implementations using MNT curves exist, for example the Miracl Library proposes software on an MNT curve over a 170-bit prime, with embedding degree $n = 6$, providing a 80-bit security level.

embedding degree n	$\log_2 p$ $(\#E(\mathbb{F}_p))$	$n \log_2 p$ $(\#\mathbb{F}_{p^n})$	80-bit security
3	170	510	no
4	170	680	no
6	170	1020	yes

Table 1. MNT curves as pairing-friendly curves in the 2000's

Construction of MNT curves The parameters p, τ, ℓ (base field, trace, and number of points) of the curve are given by polynomials of degree at most two. For $n = 3$, 4, or 6, these are

embedding degree n	$p = P(x)$	$\tau = \mathrm{Tr}(x)$	$\#E(\mathbb{F}_p) = p + 1 - \tau$
3	$12x^2 - 1$	$\pm 6x - 1$	$12x^2 \mp 6x + 1$
4	$x^2 + x + 1$	$-x,$ or $x + 1$	$x^2 + 2x + 2$ or $x^2 + 1$
6	$4x^2 + 1$	$1 \pm 2x$	$x^2 \mp 2x + 1$

To generate a curve, one needs to find an integer y of the appropriate size, such that $p = P(y)$ is prime and $\#E(\mathbb{F}_p)$ is also prime, or equal to a small cofactor times a large prime. To compute the coefficients of the curve equation, a Pell equation needs to be solved.

4

The target curve Our target will be the MNT curve given in [42, Example 1]. We recall that the curve parameters satisfy

$$y = -\texttt{0x732c8cf5f983038060466}$$
$$p = 12y^2 - 1 = \texttt{0x26dccacc5041939206cf2b7dec50950e3c9fa4827af} \text{ of } 170 \text{ bits}$$
$$\tau = 6y - 1 \text{ where } \tau \text{ is the trace of the curve}$$
$$\#E(\mathbb{F}_p) = p + 1 - \tau = 7^2 \cdot 313 \cdot \ell \text{ where } \ell \text{ is a 156-bit prime}$$
$$\ell = \texttt{0xa60fd646ad409b3312c3b23ba64e082ad7b354d}$$

The pairing embeds into the prime order ℓ subgroup of the cyclotomic subgroup of $\mathbb{F}_{p^3}$, where ℓ divides $p^2 + p + 1$.

2.3 A brief overview of NFS-DL

Our target field is $\mathbb{F}_{p^n}$. NFS-DL starts by selecting two irreducible integer polynomials f and g such that $\varphi = \gcd(f \bmod p, g \bmod p)$ is irreducible of degree n (construction of f and g is discussed in Section 3). We use the representation $\mathbb{F}_{p^n} = \mathbb{F}_p[x]/(\varphi(x))$. Let $K_f = \mathbb{Q}[x]/(f(x)) = \mathbb{Q}(\alpha)$, and $\mathcal{O}_f$ be its ring of integers. Note that because f is not necessarily monic, α might not be an algebraic integer. Let ρ_f be the map from K_f to $\mathbb{F}_{p^n}$, sending α to $T \bmod (p, \varphi(T))$. We define likewise $K_g = \mathbb{Q}(\beta)$, together with $\mathcal{O}_g$ and ρ_g. This installs the (typical) commutative diagram in Figure 1.
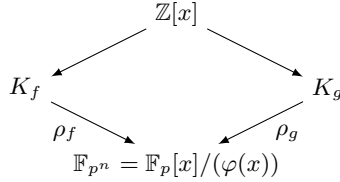
Fig. 1. NFS-DL diagram for $\mathbb{F}_{p^n}$

Given f and g, we choose a smoothness bound B and build factor bases $\mathcal{F}_f$ (resp. $\mathcal{F}_g$) consisting of prime ideals in $\mathcal{O}_f$ (resp. $\mathcal{O}_g$) of norm less than B, to which we add prime ideals dividing $\text{lc}(f)$ (resp. $\text{lc}(g)$) to take into account the fact that α and β are not algebraic integers. Then, we collect relations, that is polynomials $\phi(x) \in \mathbb{Z}[x]$ such that both ideals $\langle \phi(\alpha) \rangle$ and $\langle \phi(\beta) \rangle$ are *smooth*, namely factor completely over $\mathcal{F}_f$ (resp. $\mathcal{F}_g$). Smoothness is related to $\text{Norm}(\phi(\alpha))$, and in turn to $\text{Res}(f, \phi)$ since we have

$$\pm \text{lc}(f)^{\deg(\phi)} \text{Norm}(\phi(\alpha)) = \text{Res}(f, \phi).$$

5

When ϕ is such that the integers $\mathrm{Res}(f, \phi)$ and $\mathrm{Res}(g, \phi)$ are B-smooth (only prime factors below B), we have a relation:

$$\begin{cases} \phi(\alpha)\mathcal{O}_f = \prod_{\mathfrak{q} \in \mathcal{F}_f} \mathfrak{q}^{\mathrm{val}_{\mathfrak{q}}(\phi(\alpha))}, \\ \phi(\beta)\mathcal{O}_g = \prod_{\mathfrak{r} \in \mathcal{F}_g} \mathfrak{r}^{\mathrm{val}_{\mathfrak{r}}(\phi(\beta))} \end{cases}$$

that are transformed as linear relation between virtual logarithms of ideals [52], to which are added the so-called Schirokauer maps [51], labelled $\lambda_{f,i}$ for $1 \leq i \leq r_f$ where r_f is the unit rank of K_f (and the same for g).

To overcome the problem of dealing with fractional ideals instead of integral ideals, we use the following result from [43] (see also [20]).

Proposition 1. *Let $f(X) = \sum_{i=0}^{d} c_i X^i$ with coprime integer coefficients and α a root of f. Let*

$$J_f = \langle c_d, c_d\alpha + c_{d-1}, c_d\alpha^2 + c_{d-1}\alpha + c_{d-2}, \ldots, c_d\alpha^{d-1} + c_{d-1}\alpha^{d-2} + \cdots + c_1 \rangle.$$

Then $\langle 1, \alpha \rangle J_f = (1)$, J_f has norm $|c_d|$, and $J_f\langle a - b\alpha \rangle$ is an integral ideal for integers a and b.

If $\phi(X)$ has degree $k - 1$, we have $\mathrm{Norm}(J_f^{k-1}\langle\phi(\alpha)\rangle) = \pm\mathrm{Res}(f, \phi)$, so that we can read off the factorization of the integral $J_f^{k-1}\langle\phi(\alpha)\rangle$ directly from the factorization of its norm. A relation can now be written as:

$$(k-1)\,\mathrm{vlog}(J_f) + \sum_{\mathfrak{q} \in \mathcal{F}_f} \mathrm{val}_{\mathfrak{q}}(\phi(\alpha))\,\mathrm{vlog}(\mathfrak{q}) + \sum_{i=1}^{r_f} \lambda_{f,i}(\phi(\alpha))\,\mathrm{vlog}(\lambda_{f,i})$$

$$\equiv (k-1)\,\mathrm{vlog}(J_g) + \sum_{\mathfrak{r} \in \mathcal{F}_g} \mathrm{val}_{\mathfrak{r}}(\phi(\beta))\,\mathrm{vlog}(\mathfrak{r}) + \sum_{i=1}^{r_g} \lambda_{g,i}(\phi(\beta))\,\mathrm{vlog}(\lambda_{g,i}) \bmod \ell.$$

We select as many $\phi(x)$ of degree at most $k - 1$ (for $k \geq 2$ and very often $k = 2$) as needed to find $\#\mathcal{F}_f + \#\mathcal{F}_g + r_f + r_g + 2$ relations. Note that J_f and J_g are not always prime ideals. Nevertheless since all their prime divisors have a grouped contribution for each relation, we may count them as single columns. We may even replace the two columns by one, corresponding to $\mathrm{vlog}(J_f) - \mathrm{vlog}(J_g)$ (e.g. this is done in `cado-nfs`).

Given sufficiently many equations, the linear system in the virtual logarithms can be solved using sparse linear algebra techniques such as the Block Wiedemann algorithm [18]. When we want to compute the logarithm of a given target, we need to rewrite some power (or some multiple) of the target as a multiplicative combination of the images in $\mathbb{F}_{p^n}$ of the factor base ideals, and use the precomputed data base of computed logarithms. Section 4 will briefly discusses algebraic factorization in practice.

3 Polynomial Selection

The polynomial selection is the first step of the NFS algorithm. Polynomial selection is rather cheap, but care is needed since the quality of the polynomial pair it outputs conditions the running time of the three next steps. Section 3.1 below explains the two phases of polynomial selection. In a nutshell, we first decide from which family the polynomials are chosen, and then we search among possible solutions for "exceptionally good" polynomials. Note that because all degree n irreducible polynomials correspond to isomorphic finite fields $\mathbb{F}_{p^n}$, we are not constrained in the choice of $\mathrm{Res}(f, g)$. This degree of freedom allows to select good polynomials.

As of 2016, the available polynomial selection algorithms are:

- the Conjugation method (Conj) [6, §. 3.3], explained in Algorithm 1;
- the Generalized Joux–Lercier method (GJL) [6, §. 3.2] and [40] that produces polynomials of unbalanced coefficient sizes;
- the Joux–Lercier–Smart–Vercauteren method (JLSV$_1$) [32, §. 2.3], that produces two polynomials of degree n and coefficient size in $O(\sqrt{p})$ for both polynomials;
- the second proposition (JLSV$_2$) of the same paper [32, §. 3.2];
- the Joux–Pierrot (JP) method for pairing-friendly curves [33] which produces polynomials equivalent to the Conjugation method for MNT curves;
- the Tower-NFS method (TNFS) of Barbulescu, Gaudry and Kleinjung [8];
- the Sarkar–Singh method that combines and generalizes the GJL and Conjugation methods [49].

Remark 1 (Non-applicable methods.). The Extended-TNFS method of Kim and Kim-Barbulescu and its numerous variants [37,38,50,47,48,30] do not apply to finite fields of prime extension degree n such as $\mathbb{F}_{p^3}$. The TNFS method is not better than the best above methods for our practical case study, as shown in the paper [8, §5]. The Sarkar–Singh method [49] has two parameters (d, r): d is a divisor of n and $r \geq n/d$. Since n is prime, the pair $(d = 1, r \geq n)$ corresponds to the GJL method and the pair $(d = n, r = 1)$ to the Conjugation method. The pair $(d = n, r = 2)$ produces a polynomial f of degree 9 and small coefficients, and a polynomial g of degree 6 and coefficients in $O(p^{1/3})$. This is not competitive for our size of parameters $n = 3$ and p of 170 bits: the cross-over point between the Conjugation $(r = 1)$ and their method $(r = 2)$ is at $\log_2 p^3 = 9592$ bits.

Algorithm 1 presents the Conjugation method, which eventually provided the best yield. Pseudo-code describing the other methods can be found in Appendix A.

3.1 A First Comparison

The various methods above yield polynomial pairs whose characteristics differ significantly. Table 2 gives the expected degrees and coefficient sizes. From this data, we can derive bounds on the resultants on both sides of a relation (either using the coarse bound $(\deg f + \deg \phi)! \|f\|_\infty^{\deg \phi} \|\phi\|_\infty^{\deg f}$, or finer bounds such

---
Algorithm 1: EC:BGGM15

1]Polynomial selection with the Conjugation method [6, §3.3] **Input**: p prime and
   $\quad\quad n$ integer

   Output: f, g, ψ with $f, g \in \mathbb{Z}[x]$ irreducible and $\psi = \gcd(f \bmod p, g \bmod p)$ in
   $\quad\quad\quad\;\; \mathbb{F}_p[x]$ irreducible of degree n

2 repeat

3 $\quad$ Select $g_1(x), g_0(x)$, two polynomials with small integer coefficients,
   $\quad\quad \deg g_1 < \deg g_0 = n$

4 $\quad$ Select $a(y)$ a quadratic, monic, irreducible polynomial over $\mathbb{Z}$ with small
   $\quad\quad$ coefficients

5 until $a(y)$ *has a root* y *in* $\mathbb{F}_p$ *and* $\psi(x) = g_0(x) + yg_1(x)$ *is irreducible in* $\mathbb{F}_p[x]$

6 $f \leftarrow \mathrm{Res}_y(a(y), g_0(x) + yg_1(x))$

7 $(u, v) \leftarrow$ a rational reconstruction of y

8 $g \leftarrow vg_0 + ug_1$

9 return (f, g, ψ)
---

as [11, Th. 7], as used in [8, §. 3.2]). These norms should be minimized in order to obtain the best running-time for the NFS algorithm. We obtain the plot of Figure 2 for the bit-size of the product of norms, similar to [6, Fig. 3].

method	$\deg f$	$\|f\|_\infty$	$\deg g$	$\|g\|_\infty$
GJL	$D + 1 \geq n + 1$	$O(\log p)$	$D \geq n$	$O(Q^{1/(D+1)})$
JP or Conj	$2n$	$O(\log p)$	n	$O(Q^{1/(2n)})$
JLSV_1	n	$O(Q^{1/(2n)})$	n	$O(Q^{1/(2n)})$
JLSV_2	$D \geq n + 1$	$O(Q^{1/(D+1)})$	n	$O(Q^{1/(D+1)})$

Table 2. Norm bound w.r.t. Q

Figure 2 suggests that the GJL method yields the smallest norms for $\log_2 Q = 508$. The norms produced with the Conjugation and JLSV_1 methods are not very far however so we compared more precisely these three methods for our 170-bit parameters. This entails finding competitive polynomial pairs for each method, and comparing their merits. Estimated bounds as well as experimental values for the products of norms for $\log_2 Q = 508$ are reported in Table 3. Results of sieving on one slide of special-q is reported in Table 4. The algorithms and computed polynomials are given in Appendix A. The theoretical bound $\|f\|_\infty$ equals one bit in the Conjugation and GJL methods whereas in practice to improve the smoothness properties of f, we have chosen a polynomial with moderately larger coefficients, and with better α and Murphy's E values (see [44, §5.2 eq. (5.7)] on Murphy's E value). The coefficient size of g selected with the GJL, Conj and JLSV_1methods is a few bits larger than the theoretical bound because we computed linear combinations of two distinct g, and of f and the initial g in the JLSV_1 case (since they are of same degree). The advantage of the hybrid
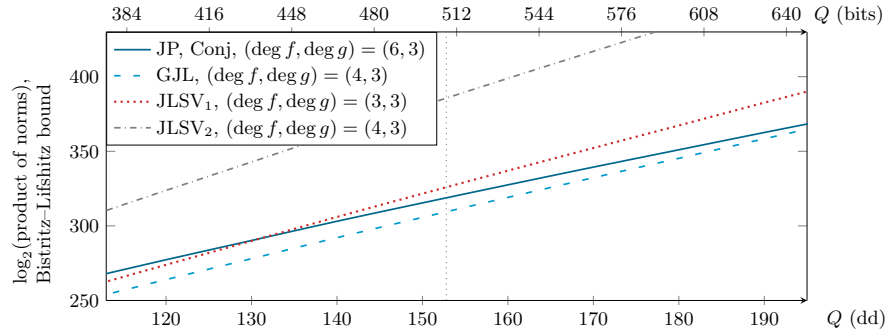
Fig. 2. Norm bound for four polynomial selection methods for $\mathbb{F}_{p^3}$

Joux–Pierrot method (Algorithm 2) in the MNT case is that g can be monic, which does not allow for linear combinations.

method	$\|f\|_\infty$		$\|g\|_\infty$		Norm bound f		Norm bound g		product	
	bound	exp.	bound	exp.	bound	exp.	bound	exp.	bound	exp.
GJL	1	2	127	130	106	107	206	208	311	314
Conj	1	9	85	86	157	165	163	164	320	328
hybrid JP	1	12	85	85	157	168	163	164	320	331
JLSV$_1$	85	85	85	86	163	163	163	164	326	327
JLSV$_2$	102	–	102	–	206	–	180	–	386	–

Table 3. Norm bounds in bits for $\log Q = 508$ and $\log E = 25.25$: estimates based on Table 2, compared to experimental values with our selected polynomials.

Galois actions. For small extension degrees $n \in \{3, 4, 6\}$ there exist families of polynomials producing number fields with cyclic Galois groups, and an easy-to-compute automorphism [21, Prop. 1.2]. Taking polynomials from these families yields a speed-up in the sieving part as well as in the linear algebra part for the JLSV$_1$ and Conjugation methods. We take $g = x^3 - y_0 x^2 - (y_0 + 3)x - 1$ for the Conjugation method, i.e. $g_0 = x^3 - 3x - 1$ and $g_1 = -x^2 - x$ in Algorithm 1. The Galois action is $\sigma(x) = (-x - 1)/x$ which is independent of the parameter y_0. In that case, given the factorization for $\langle a - b\alpha \rangle$, we can deduce that of

$$\sigma(\langle a - b\alpha \rangle) = \langle a - b\sigma(\alpha) \rangle = -\frac{1}{\alpha}(b - (-a - b)\alpha).$$

The same holds on the f side.

Forming a database of good polynomials f. For the Conjugation method (and similarly for the competing methods), the early steps in Algorithm 1 can be

tabulated in some way, depending only on the extension degree n (and for JLSV$_1$, also on the size of p, but not its value): we can store a database of f's with good smoothness properties (low α and high Murphy's E values). Actually we searched over $a(y) = a_2y^2 + a_1y + a_0$, where $0 < a_2 < 32$, $|a_1| < 32$ and $|a_0| < 512$, and computed $f = \mathrm{Res}_y(a(y), x^3 - yx^2 - (y+3)x - 1)$. Later, depending on p, we can continue Algorithm 1 for these precomputed polynomials (test whether a has a root modulo p).

Note also that in Algorithm 1, the rational reconstruction step naturally produces several quotients u/v, which yield several candidate polynomials g. Small linear combinations of these polynomials can be tried, in order to improve on the Murphy's E value.

3.2   Probing the sieving yield

To finalize the comparison between the polynomials, we compared the relation yield for small special-q ranges sampled over the complete special-q space. Because the JLSV$_1$ and Conjugation methods feature balanced norms (see Tab. 3), we used similar large prime bounds (27 bits) on both sides in both cases, and allowed two large prime on each side. In contrast, for the GJL method, we allowed 28-bit large primes on the g side, and chose q to be only on that side. The Conjugation method (polynomial below) appeared as the best option based on the seconds/relation measure, given that the overall yield was sufficient. Results of this test are reported on Table 4.

$$
\begin{aligned}
f &= 28x^6 + 16x^5 - 261x^4 - 322x^3 + 79x^2 + 152x + 28 \\
\alpha(f) &= -2.94 \\
\log_2 \|f\|_\infty &= 8.33 \\
g &= 2475781518663919737044 2122\, x^3 + 408068970402536804717 75183\, x^2 \\
&\quad - 33466548519663911639551183 x - 2475781518663919737044 2122 \\
\alpha(g) &= -4.16 \\
\log_2 \|g\|_\infty &= 85.08, \text{ the optimal being } \tfrac{1}{2}\log_2 p = 85 \\
\mathrm{E}(f,g) &= 1.31 \cdot 10^{-12}
\end{aligned}
\tag{2}
$$

Method	seconds/relation	relations/special-q	remarks
Generalized Joux–Lercier	3.48	4.96	0+3 large primes below 2^{28}
JLSV$_1$	1.31	4.24	2+2 large primes below 2^{27}, orbits
Conjugation	**0.91**	5.93	of three special-q batched together

Table 4. Probed yield for special-q ranges. Cpu time on Intel Xeon E5520 (2.27GHz).

10

4 Solving DLP over $\mathbb{F}_{p^3}$

4.1 Sieving and linear algebra

We took a smoothness bound of 50×10^6 on both sides; and all special-q in $[50 \times 10^6, 2^{27}]$, on both sides. This took roughly 660 core-days, normalized on the most common hardware used, namely 4-core Intel Xeon E5520 CPUs (2.27GHz). We collected 57070251 relations, out of which 34740801 were non duplicate. Filtering produced a 1982791×1982784 matrix M with weight 396558692. Taking into account the block of 7 Schirokauer maps S, the matrix $M\|S$ is square.

We computed 8 sequences in the Block Wiedemann algorithm, using the trick mentioned in [18, §8], as programmed in `cado-nfs` (rediscovered and further analyzed in [34]). All these sequences can be computed independently. Computation time for the 8 Krylov sequence was about 250 core-days (Xeon E5-2650, 2.4GHz, using four 16-core nodes per sequence). Finding the linear (matrix) generator for the matrices took 75 core-hours, parallelized over 64 cores. Building the solution cost some more 170 core-days. We reconstructed virtual logarithms for 15196345 out of the 15206761 factor base elements (99.9%). This was good enough to start looking for individual logarithms.

4.2 Computing individual discrete logarithms in $\mathbb{F}_{p^3}$

From the linear algebra step, we know how to compute the logarithm modulo ℓ of any element of $\mathbb{F}_{p^3}$ whose lift in either K_f or K_g factors completely over the factor base. Lifting in K_f is often convenient because norms are smaller.

The tiny case. A particular element which lifts conveniently in K_f is the common root t of both polynomials. By construction, its lift $\alpha \in K_f$ generates a principal (fractional) ideal that factors as J_f^{-1} (see Proposition 1) times prime ideals of norm dividing 28, namely: $(\alpha) = I_{2,0}^2 I_{2,\infty}^{-2} I_{7,0} I_{7,\infty}^{-1}$, where $I_{2,\infty}^2 I_{7,\infty}$ corresponds to J_f and the prime ideals in the right-hand side can be made explicit. Its logarithm therefore writes as [8]

$$\log(t) = 2\,\mathrm{vlog}\,I_{2,0} - 2\,\mathrm{vlog}\,I_{2,\infty} + \mathrm{vlog}\,I_{7,0} - \mathrm{vlog}\,I_{7,\infty} + \sum_{i=1}^{5} \lambda_{f,i}(\alpha)\,\mathrm{vlog}(\lambda_{f,i}).$$

$\lambda_{f,1}(\alpha) = \text{0x3720106a3d368d7f731a0757b905778050ae327}, \lambda_{f,2}(\alpha) = \text{0x1dbeace7d0ec187712ae8afcd6ccdc4db06f781},$

$\lambda_{f,3}(\alpha) = \text{0x9c3109f7741d625869f135706be03fc09375450}, \lambda_{f,4}(\alpha) = \text{0x1e46653b287d99c502a5c6e12ab17a3dd10988c},$

$\lambda_{f,5}(\alpha) = \text{0x31628f3e0b491e622946b32f66292c1389a7427}.$

By construction the value $\log(t)$ above is invertible modulo ℓ, and we can freely normalize our virtual logarithm values so that it is equal to one.

---

[8] The convention in `cado-nfs` is to take coefficients of largest degree first in the Schirokauer maps computation $z \mapsto \frac{1}{\ell}(z^{\ell^m - 1} - 1)$ where $m = \mathrm{lcm}_{\mathfrak{l}\ \mathrm{prime},\ \mathfrak{l}|\ell}[\mathfrak{l} : \ell]$. Here we have $m = 1$.

The tame case. Elements whose lifts do not factor completely over any of the factor base but have only moderate-size outstanding factors can be dealt with using a classical *descent* procedure. This finds recursively new relations involving smaller and smaller primes, until all primes involved belong to the factor base. Software achieving this exists, such as the `las_descent` program in `cado-nfs`.

The general case. For computing individual logarithms of arbitrary elements, we used the boot technique described in [28]. For each target, we compute a preimage in $\mathbb{Z}[x]$ represented by a polynomial of degree at most 5 and coefficients bounded by $p^{1/3}$. The norm in K_f of the preimage is $O(p^2) = O(Q^{2/3})$, of approximately 340 bits. The asymptotic complexity of this step is $L_Q[1/3, 1.26]$, and would be $L_Q[1/3, 1.132]$ with one early-abort test (see e.g. [45, §4.3] or [3, Ch. 4]). The optimal size of largest prime factors in the decomposition is given by the formula $L_Q[2/3, (e^2/3)^{1/3} \approx 0.529]$, where $e = 2/3$ (see [17, §4]). Applying it for $\log_2 Q = 508$ gives a bound of 68 bits and a running-time of approximately 2^{42} tests. In practice we found very easily initial splittings where B_1 is less than 64 bits, which eased the descent.

4.3 Solving the challenge

Our main use case for individual logarithm computation in $\mathbb{F}_{p^3}$ is to solve a DLP challenge on the curve. The challenge definition procedure (described in the appendix[9], the Magma code is also available[10]) gives:

$$G_1 = \big(\text{0x106b415d7b4a2d71659ae97440cbb20a6de42d76d69}, \text{0x16d74a2a88e817f1821a1c40e220d34eec93e33cb83}\big),$$

$$P = \big(\text{0x15052ba45717710e6b0cbf8ed89c5c1a0a279480e26}, \text{0x8050f05a231ae1f13e56de1171c108294656052339}\big)$$

From Section 2.1, we need to compute $\log(G_T)$ and $\log(S)$, where $G_T = e(G_1, G_2)$ and $S = e(P, G_2)$ are given in the Magma verification script[10]. We searched for randomized values G_T^r and $G_T^{r'}S$ which were amenable to the descent procedure. After 32 core-hours looking in the range $r \in [1, 64000]$, we selected the following element

$$G_T^{52154} = -\text{0x21d517d51512e9} - \text{0x95233b3af1b3c7}\, x + \text{0x8d324ebc7849bb}\, x^2$$
$$+ \text{0x18ff0d5ae0b52b}\, x^3 + \text{0x13f711fe92d63cd}\, x^4 - \text{0x15c778630d36920}\, x^5$$

whose straightforward lift in K_f has 59-bit smooth norm (resultant with f, more precisely):

0x87ac1a057df9772d1e08d4de56b3e6b5f208710437b5f92ac4a494c318c9781107e00364934e34efa87b26597771c

$$= 2^2 \cdot 5 \cdot 7^2 \cdot 31 \cdot 193 \cdot 277 \cdot 1787 \cdot 12917 \cdot 125789 \cdot 142301513 \cdot 380646221 \cdot 2256567883$$

$$\cdot 132643203397 \cdot 138019432565816569 \cdot 603094914193031251 \cdot 801060739300538627$$

---

[9] §B.1 and §B.2 of the pre-proceedings version available at `https://hal.inria.fr/hal-01320496`

[10] `http://www.lix.polytechnique.fr/~guillevic/discrete-log/SAC2016-mnt170-verification-script.mag`

Virtual logarithms for primes below $50 \cdot 10^6$ (25.57 bits) were known. The descent procedure took 13.4 hours. Once all logarithms were computed, the value of $\log(G_T)$ could be deduced:

$$\log(G_T) = \text{0x8c58b66f0d8b2e99a1c0530b2649ec0c76501c3} \text{ (normalized to } \log t = 1\text{)}.$$

Similarly, we selected

$$G_T^{35313} S \mapsto \quad \text{0x457449569db669} + \text{0x88c32ec54242fd}\, x - \text{0x2370c0f5914ba9}\, x^2$$
$$+ \text{0x14c7ccbafc20e2}\, x^3 + \text{0xde2e21c5f1a4c4}\, x^4 - \text{0x10b6bfd826db49c}\, x^5$$

whose lift in K_f has norm

$$-\text{0x44dafd6ec57c91e64567fa045187100da9a98c5c509b388cb61759f345b3ce27226a5e8520be0bd4559acbd538b90}$$

$$= -2^4 \cdot 5^2 \cdot 7 \cdot 643 \cdot 1483 \cdot 2693 \cdot 95617 \cdot 9573331 \cdot 33281579 \cdot 1608560119 \cdot 48867401441$$

$$\cdot 516931716361 \cdot 896237937459937 \cdot 16606283628226811 \cdot 19530910835315983$$

the largest factor having 54 bits, a very small size indeed (compared to the 68 bits predicted by theory). The descent procedure for other primes took 10.7 hours. We found that

$$\log(S) = \text{0x48a6bcf57cacca997658c98a0c196c25116a0aa} \text{ (normalized to } \log t = 1\text{)}.$$

We eventually found that

$$\log_{G_1}(P) = \text{0x711d13ed75e05cc2ab2c9ec2c910a98288ec038} \bmod \ell.$$

5 Conclusion and future work

5.1 Consequences for pairing-based cryptography

Our work showed that the choice of embedding degree n and finite field size $\log p^n$ should be done carefully. The size of $\mathbb{F}_{p^n}$ should be large enough to provide the desired level of security. We recall these sizes for $\mathbb{F}_{p^3}$. The recent improvements of Kim and Kim–Barbulescu [37,38] do not apply to $\mathbb{F}_{p^n}$ where n is prime, so $\mathbb{F}_{p^3}$ is not affected. The asymptotic complexity of the NFS algorithm for $\mathbb{F}_{p^3}$ is $\exp\left((c + o(1))(\log p^n)^{1/3}(\log \log p^n)^{2/3}\right) = L_{p^3}[1/3, (64/9)^{1/3}]$. Since there is a polynomial factor hidden in the notation $c + o(1)$, taking $\log_2 L_{p^3}[1/3, (64/9)^{1/3}]$ does not give the exact security level but only an approximation. We may compare our present record with previous records of same size for prime fields $\mathbb{F}_p$ and quadratic fields $\mathbb{F}_{p^2}$. Kleinjung in 2007 announced a record computation in a prime field $\mathbb{F}_p$ of 530 bits (160 decimal digits) [39]. Barbulescu, Gaudry, Guillevic and Morain in 2014 announced a record computation in $\mathbb{F}_{p^2}$ of 529 bits (160 decimal digits) [4]. We compare the timings in Table 5. The timings of relation collection and linear algebra were not balanced in Kleinjung record: 3.3 years compared to 14 years and moreover, this is a quite old record so it is not really possible to compare our record with this one directly. We can compare our record with the 529-bit $\mathbb{F}_{p^2}$ record computation of 2014 [4]. Our total running-time is 15.5 times longer whereas the finite field is 21 bit smaller.

record	relation collection	linear algebra	individual log	total
Kleinjung [39] 530-bit field $\mathbb{F}_p$, 2007	3.3 CPU-years 3.2 GHz Xeon64	14 years 3.2 GHz Xeon64	few hours 3.2 GHz Xeon64	17.3 years
BGGM [4] 529-bit field $\mathbb{F}_{p^2}$, 2014	68 core-days=0.19y 2.0 GHz E5-2650	30.3 hours NVidia GTX 680 graphic card	few hours 2.0 GHz E5-2650	70 days = 0.2 year
BGGM [5] 512-bit field $\mathbb{F}_{p^3}$, 2015	850 core-days = 2.33 years	5500 core-days = 15 years	few days	17.3 years
	2.4 GHz Xeon E5-2650			
this work 508-bit field $\mathbb{F}_{p^3}$, 2016	660 core-days =1.81 years 2.27GHz 4-core Xeon E5520	423 days = 1.16 years 2.4 GHz Xeon E5-2650	2 days 2.27GHz 4-core Xeon E5520	1085 days = 2.97 years

Table 5. Comparison of running-time for Discrete Logarithm records in $\mathbb{F}_p$, $\mathbb{F}_{p^2}$ and $\mathbb{F}_{p^3}$ of 530, 529, 512 and 508 bits.

5.2 Future work

We have computed a DLP on an MNT curve with embedding degree 3. What are the next candidates? We could continue the series in two directions: increasing the size of p^n to 600 bits, in order to compare this new record to the previous records of the same size, in particular the $\mathbb{F}_{p^2}$ record of 600 bits [6]. We could conjecture, according to the present record and the size of the norms, that a DLP record in $\mathbb{F}_{p^3}$ of 600 bits will be more than 15 times harder than in a 600-bit field $\mathbb{F}_{p^2}$.

The second direction would be to continue the series of MNT curves, with $n = 4$. We found an MNT curve of embedding degree 4 in Miracl (file `k4mnt.ecs`). The curve was generated by Drew Sutherland for Mike Scott a long time ago.

$$y = \texttt{0xf19192168b16c1315d33}$$
$$p = y^2 + y + 1 = \texttt{0xe3f367d542c82027f33dc5f3245769e676a5755d}$$
$$\ell = \texttt{0x6b455e0a014f1e30eaef7300bd4bb4258290fc5}$$
$$\tau = y + 1 = \texttt{0xf19192168b16c1315d34}$$
$$\#E(\mathbb{F}_p) = y^2 + 1 = p + 1 - \tau = 2 \cdot 17 \cdot \ell$$

Since n is a prime power, we have to adapt the Kim–Barbulescu technique (dedicated to non-prime power n) to prime-power extension degrees[11]. We construct $\mathbb{F}_{p^4}$ as $\mathbb{F}_{p^2}[x]/(\varphi(x))$, where $\mathbb{F}_{p^2} = \mathbb{F}_p[s]/(h_1(s))$ and both h_1 and φ are of degree 2, and φ has coefficients in $\mathbb{F}_{p^2}$. As a consequence, the polynomials f and

---

[11] right after the submission, several variants of Kim's Extended TNFS where proposed, that deal with any composite n, in particular prime power n, and generalize the Sarkar–Singh method [50,47,48,30].

g will have coefficients in $\mathbb{Z}[s]/(h_1(s))$ instead of $\mathbb{Z}$. For example, one could take

$$
\begin{aligned}
h_1(s) &= s^2 + 2, \\
h_2(x, t_0, s) &= x^2 + s + t_0, \\
P(t_0) &= t_0^2 + t_0 + 1, \\
f &= \mathrm{Res}_{t_0}(P(t_0), h_2(x, t_0, s)) = x^4 + (2s-1)x^2 - s - 1, \\
g &= h_2(x, y, s) = x^2 + s + \texttt{0xf19192168b16c1315d33}.
\end{aligned}
$$

The major difference is that to be efficient, we have to sieve polynomials of degree 1 with coefficients in $\mathbb{Z}[s]/(h_1(s))$, that is elements of the form $(a_0+a_1 s)+(b_0+b_1 s)x$ where the a_i's and b_i's are small rational integers, say $|a_i|, |b_i| \leq A$. For instance, taking $\log_2(E) = 1.1(\log Q)^{1/3}(\log \log Q)^{2/3} \approx 28$, we obtain $A = E^{2/(2\deg h)}$ of 14 bits. The upper bound on the norm would be of 120 bits on f-side and 219 bits on g-side, the total being roughly of 339 bits. This is 11 bits more than our present record for the 508-bit $n = 3$ MNT curve (328 bits, Table 3), but by far much less than with any previous technique applied to that $\mathbb{F}_{p^4}$. Norm estimates are provided in Table 6. From a practical point of view, we would need extensions of the work [26].

Table 6. Norm bound estimates for $\mathbb{F}_{p^4}$ of 640 bits.

method	$\|f\|_\infty$	$\|g\|_\infty$	NB_f	NB_g	$NB_f + NB_g$
Extended TNFS+hybrid JP	1	80	120	219	339
GJL	1	128	144	243	387
JLSV$_1$	80	80	195	195	390
Sarkar-Singh, $d = 2, r = 2$	1	107	172	222	394
Hybrid JP–Conj	1	80	159	240	399
JLSV$_2$, $D = 6$ (D best choice)	91	91	264	206	470

References

1. L. M. Adleman and M.-D. Huang. Function field sieve methods for discrete logarithms over finite fields. *Inform. and Comput.*, 151(1):5–16, 1999.
2. D. Adrian, K. Bhargavan, Z. Durumeric, P. Gaudry, M. Green, J. A. Halderman, N. Heninger, D. Springall, E. Thomé, L. Valenta, B. VanderSloot, E. Wustrow, S. Z. Béguelin, and P. Zimmermann. Imperfect forward secrecy: How Diffie-Hellman fails in practice. In I. Ray, N. Li, and C. Kruegel:, editors, *ACM CCS 15*, pages 5–17. ACM Press, Oct. 2015.
3. R. Barbulescu. *Algorithmes de logarithmes discrets dans les corps finis*. PhD thesis, Université de Lorraine, 2013.
4. R. Barbulescu, P. Gaudry, A. Guillevic, and F. Morain. Discrete logarithms in $\mathrm{GF}(p^2)$ — 160 digits. Announcement on the Number Theory List, Jun 2014. `https://listserv.nodak.edu/cgi-bin/wa.exe?A2=NMBRTHRY;2ddabd4c.1406`.

5. R. Barbulescu, P. Gaudry, A. Guillevic, and F. Morain. Discrete logarithms in $GF(p^3)$ — 512 bits. Announcement at the CATREL workshop, Oct 2015. https://webusers.imj-prg.fr/~razvan.barbaud/p3dd52.pdf.

6. R. Barbulescu, P. Gaudry, A. Guillevic, and F. Morain. Improving NFS for the discrete logarithm problem in non-prime finite fields. In E. Oswald and M. Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 129–155. Springer, Heidelberg, Apr. 2015.

7. R. Barbulescu, P. Gaudry, A. Joux, and E. Thomé. A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In P. Q. Nguyen and E. Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 1–16. Springer, Heidelberg, May 2014.

8. R. Barbulescu, P. Gaudry, and T. Kleinjung. The tower number field sieve. In T. Iwata and J. H. Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 31–55. Springer, Heidelberg, Nov. / Dec. 2015.

9. P. S. L. M. Barreto and M. Naehrig. Pairing-friendly elliptic curves of prime order. In B. Preneel and S. Tavares, editors, *SAC 2005*, volume 3897 of *LNCS*, pages 319–331. Springer, Heidelberg, Aug. 2006.

10. J. Beuchat, N. Brisebarre, J. Detrey, E. Okamoto, M. Shirase, and T. Takagi. Algorithms and arithmetic operators for computing the η_T pairing in characteristic three. *IEEE Trans. Computers*, 57(11):1454–1468, 2008.

11. Y. Bistritz and A. Lifshitz. Bounds for resultants of univariate and bivariate polynomials. *Linear Algebra and its Applications*, 432(8):1995 – 2005, 2010. Special issue devoted to the 15th ILAS Conference at Cancun, Mexico, June 16-20, 2008.

12. I. F. Blake, G. Seroussi, and N. Smart. *Advances in Elliptic Curve Cryptography*, volume 317 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 2005.

13. D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, Heidelberg, Aug. 2001.

14. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In C. Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 514–532. Springer, Heidelberg, Dec. 2001.

15. F. Brezing and A. Weng. Elliptic curves suitable for pairing based cryptography. *Des. Codes Cryptogr.*, 37(1):133–141, 2005.

16. C. Cocks and R. G. Pinch. ID-based cryptosystems based on the Weil pairing, 2001. Unpublished manuscript.

17. A. Commeine and I. Semaev. An algorithm to solve the discrete logarithm problem with the number field sieve. In M. Yung, Y. Dodis, A. Kiayias, and T. Malkin, editors, *PKC 2006*, volume 3958 of *LNCS*, pages 174–190. Springer, Heidelberg, Apr. 2006.

18. D. Coppersmith. Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. *Math. Comp.*, 62(205):333–350, 1994.

19. R. Dupont, A. Enge, and F. Morain. Building curves with arbitrary small MOV degree over finite prime fields. *Journal of Cryptology*, 18(2):79–89, Apr. 2005.

20. R. M. Elkenbracht-Huizing. An implementation of the number field sieve. *Experiment. Math.*, 5(3):231–253, 1996.

21. K. Foster. HT90 and "simplest" number fields. *Illinois J. Math.*, 55(4):1621–1655, 2011.

22. D. Freeman. Constructing pairing-friendly elliptic curves with embedding degree 10. In F. Hess, S. Pauli, and M. E. Pohst, editors, *Algorithmic Number Theory,*

7th International Symposium, ANTS-VII, Berlin, Germany, July 23-28, 2006, Proceedings, volume 4076 of *Lecture Notes in Comput. Sci.*, pages 452–465. Springer, 2006.

23. D. Freeman, M. Scott, and E. Teske. A taxonomy of pairing-friendly elliptic curves. *Journal of Cryptology*, 23(2):224–280, Apr. 2010.

24. G. Frey and H. G. Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Math. Comp.*, 62(206):865–874, 1994.

25. S. D. Galbraith, J. F. McKee, and P. C. Valença. Ordinary abelian varieties having small embedding degree. *Finite Fields and Their Applications*, 13(4):800–814, 2007.

26. P. Gaudry, L. Grémy, and M. Videau. Collecting relations for the number field sieve in $\mathrm{GF}(p^6)$. *LMS Journal of Computation and Mathematics, Special issue ANTS-XII*, August 2016. To appear, preprint available at `https://eprint.iacr.org/2016/124`.

27. R. Granger, T. Kleinjung, and J. Zumbrägel. Breaking '128-bit secure' supersingular binary curves - (or how to solve discrete logarithms in $F_{2^{4\cdot1223}}$ and $F_{2^{12\cdot367}}$). In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 126–145. Springer, Heidelberg, Aug. 2014.

28. A. Guillevic. Computing individual discrete logarithms faster in $\mathrm{GF}(p^n)$ with the NFS-DL algorithm. In T. Iwata and J. H. Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 149–173. Springer, Heidelberg, Nov. / Dec. 2015.

29. T. Hayashi, T. Shimoyama, N. Shinohara, and T. Takagi. Breaking pairing-based cryptosystems using η_T pairing over $gf(3^{97})$. In X. Wang and K. Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 43–60. Springer, Heidelberg, Dec. 2012.

30. J. Jeong and T. Kim. Extended tower number field sieve with application to finite fields of arbitrary composite extension degree. Cryptology ePrint Archive, Report 2016/526, 2016. `http://eprint.iacr.org/2016/526`.

31. A. Joux. A one round protocol for tripartite Diffie-Hellman. In W. Bosma, editor, *ANTS-IV*, volume 1838 of *Lecture Notes in Comput. Sci.*, pages 385–394. Springer, 2000.

32. A. Joux, R. Lercier, N. Smart, and F. Vercauteren. The number field sieve in the medium prime case. In C. Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 326–344. Springer, Heidelberg, Aug. 2006.

33. A. Joux and C. Pierrot. The special number field sieve in $\mathbb{F}_{p^n}$ - application to pairing-friendly constructions. In Z. Cao and F. Zhang, editors, *PAIRING 2013*, volume 8365 of *LNCS*, pages 45–61. Springer, Heidelberg, Nov. 2014.

34. A. Joux and C. Pierrot. Nearly sparse linear algebra. Cryptology ePrint Archive, Report 2015/930, 2015. `http://eprint.iacr.org/2015/930`.

35. M. Kasahara, K. Ohgishi, and R. Sakai. Notes on ID-based key sharing systems on elliptic curve. Technical report, IEICE, Nov. 1999.

36. M. Kasahara, K. Ohgishi, and R. Sakai. Cryptosystems based on pairing. In *The 2000 Symposium on Cryptography and Information Security*, volume SCIS2000-C20, Jan. 2000.

37. T. Kim. Extended tower number field sieve: A new complexity for medium prime case. Cryptology ePrint Archive, Report 2015/1027, version 1, October 2015. `http://eprint.iacr.org/eprint-bin/getfile.pl?entry=2015/1027&version=20151026:154834&file=1027.pdf`.

38. T. Kim and R. Barbulescu. Extended tower number field sieve: A new complexity for the medium prime case. In M. Robshaw and J. Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 543–571. Springer, Heidelberg, Aug. 2016.

39. T. Kleinjung. Discrete logarithms in GF(p) — 160 digits. Announcement on the Number Theory List, Feb 2007. `https://listserv.nodak.edu/cgi-bin/wa.exe?A2=NMBRTHRY;1c737cf8.0702`.

40. D. Matyukhin. Effective version of the number field sieve for discrete logarithms in the field GF(p^k) (in Russian). *Tr. Diskr. Mat.*, 9:121–151, 2006. `http://m.mathnet.ru/php/archive.phtml?wshow=paper&jrnid=tdm&paperid=144&option_lang=eng`.

41. A. Menezes, T. Okamoto, and S. A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Transactions on Information Theory*, 39(5):1639–1646, 1993.

42. A. Miyaji, M. Nakabayashi, and S. Takano. Characterization of elliptic curve traces under FR-reduction. In D. Won, editor, *ICISC 00*, volume 2015 of *LNCS*, pages 90–108. Springer, Heidelberg, Dec. 2001.

43. P. L. Montgomery. Square roots of products of algebraic numbers, 1997. Unpublished draft, dated May 16, 1997.

44. B. A. Murphy. *Polynomial Selection for the Number Field Sieve Integer Factorisation Algorithm*. PhD thesis, Australian National University, 1999.

45. C. Pomerance. Analysis and comparison of some integer factoring algorithms. In H. W. J. Lenstra and R. Tijdeman, editors, *Computational methods in number theory, part I*, volume 154 of *Mathematical Centre Tracts*, pages 89–139. Mathematisch Centrum, Amsterdam, 1982. available in pdf at `http://oai.cwi.nl/oai/asset/19571/19571A.pdf`.

46. Y. Sakemi, G. Hanaoka, T. Izu, M. Takenaka, and M. Yasuda. Solving a discrete logarithm problem with auxiliary input on a 160-bit elliptic curve. In M. Fischlin, J. Buchmann, and M. Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 595–608. Springer, Heidelberg, May 2012.

47. P. Sarkar and S. Singh. A general polynomial selection method and new asymptotic complexities for the tower number field sieve algorithm. In J. H. Cheon and T. Takagi, editors, *ASIACRYPT 2016*, Lecture Notes in Comput. Sci., Dec 2016. to appear, preprint at `http://eprint.iacr.org/2016/485`.

48. P. Sarkar and S. Singh. A generalisation of the conjugation method for polynomial selection for the extended tower number field sieve algorithm. Cryptology ePrint Archive, Report 2016/537, 2016. `http://eprint.iacr.org/2016/537`.

49. P. Sarkar and S. Singh. New complexity trade-offs for the (multiple) number field sieve algorithm in non-prime fields. In M. Fischlin and J.-S. Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 429–458. Springer, Heidelberg, May 2016.

50. P. Sarkar and S. Singh. Tower number field sieve variant of a recent polynomial selection method. Cryptology ePrint Archive, Report 2016/401, 2016. `http://eprint.iacr.org/2016/401`.

51. O. Schirokauer. Discrete logarithms and local units. *Philos. Trans. Roy. Soc. London Ser. A*, 345(1676):409–423, 1993.

52. O. Schirokauer. Virtual logarithms. *J. Algorithms*, 57(2):140–147, 2005.

A Polynomial selection methods

We provide in this section the polynomials computed for our $\mathbb{F}_{p^3}$ record with the other competitive polynomial selection methods that we compared in Section 3.

Generalized Joux–Lercier method. The first step of the GJL polynomial selection algorithm is to choose a polynomial f of degree 4 in our context. We need f to factor as a linear polynomial times a degree 3 polynomial modulo p, hence we cannot allow for a degree two subfield, or any of the Galois groups C4, V4 or D4. We extracted from the Magma number field database the list of irreducible polynomials of degree 4 and Galois group A4 (of order 12), class number one and signature $(0, 2)$ (592 polynomials) and $(4, 0)$ (3101 polynomials).

In the GJL method, the LLL algorithm outputs four polynomials g_1, g_2, g_3 and g_4 with small coefficients. To obtain the smallest possible coefficients, we set the LLL parameters to $\delta = 0.99999$ and $\eta = 0.50001$. We compute linear combinations $g = \sum_{i=1}^4 \lambda_i g_i$ with $|\lambda_i|\|g_i\|_\infty \leq 2^5 \cdot \min_{1 \leq i \leq 4} \|g_i\|_\infty$ (roughly speaking, $|\lambda_i| \leq 32$) so that the size of the coefficients of g do not increase too much, while we can obtain a polynomial g with a better Murphy's $\mathbb{E}$ value.

Then we run the GJL method with our modified post-LLL step for each polynomial f in our database and we selected the pair with the highest Murphy's $\mathbb{E}$ value. We obtained

$f \qquad = x^4 - 2x^3 + 2x^2 + 4x + 2$

$\alpha(f) \qquad = 1.2$

$\log_2 \|f\|_\infty = 2$

$g = 133714102332614336563681181193704960555\, x^3 + 173818706907699496668994559342802299969\, x^2$
$\qquad + 878019651910536420352249995702628405053\, x - 185403948115503498471378323785210605885$

$\alpha(g) \qquad = -2.1$

$\log_2 \|g\|_\infty = 129.37$, the optimal being $\frac{3}{4} \log_2 p = 127.5$

$\mathrm{E}(f,g) \qquad = 5.08 \cdot 10^{-13}$

Joux-Lercier-Smart-Vercauteren method. The Joux-Lercier-Smart-Vercauteren method (JLSV1) is possibly the most straighforward polynomial selection method adapted to non-prime finite fields. It is possible to force this method to pick polynomials f within a specific family, in order to force nice Galois properties. For example, we may use the form $\psi = x^3 - tx^2 - (t + 3)x - 1$.

The enumeration was the largest for the JLSV_1 method: we searched over 2^{25} polynomials f in the cyclic family $x^3 - t_0 x^2 - (t_0 + 3)x - 1$, with a parameter t_0 of 84 up to 85 bits. We kept the polynomials whose α value was less than -3.0. We continued the JLSV_1 polynomial selection algorithm selectively for these good precomputed polynomials. The "initial" g (say g_0) produced by the method can be improved by using instead any linear combination $g = \lambda f + \mu g_0$ for small λ and μ, thereby improving the Murphy's E value. We set $|\lambda|, |\mu| \leq 2^5$.

$$
\begin{aligned}
f \qquad &= x^3 - 301456631008579392963434446\, x^2 - 301456631008579392963434449\, x - 1 \\
\alpha(f) \qquad &= -3.0 \\
\log_2 \|f\|_\infty &= 84.64 \\
g \qquad &= 301456631008579392996699540\, x^3 + 468452741444959805783316407\, x^2 \\
&\quad -435917151580778373207822213\, x - 301456631008579392996699540 \\
\alpha(g) \qquad &= -2.8 \\
\log_2 \|g\|_\infty &= 85.28, \text{ the optimal being } \tfrac{1}{2} \log_2 p = 85 \\
\mathrm{E}(f,g) \qquad &= 1.02 \cdot 10^{-12}
\end{aligned}
\tag{3}
$$

Conjugation and Joux–Pierrot methods. The Joux-Pierrot method produces polynomials with the same degree and coefficient properties as the Conjugation method for MNT curves and that are moreover monic. The polynomials constructed with the Conjugation method allow a factor two speed-up thanks to a Galois automorphism. We propose here a hybrid variant in Algorithm 2 for pairing-friendly curves. The conjugation method, in Algorithm 1, is the one which eventually produced the best polynomial pair.

For the Conjugation method as well as the hybrid method of Algorithm 2, and similarly to the JLSV$_1$ method, it is possible to choose polynomials g of the form $\psi = x^3 - tx^2 - (t+3)x - 1$ to allow a Galois automorphism of degree 3.

Algorithm 2: Variant of Joux–Pierrot and Conjugation methods

Input: p prime, $p = P(y)$ where $\deg P \geq 2$ and P of tiny coefficients, n integer
Output: $f, g \in \mathbb{Z}[x]$ irreducible and $\psi = \gcd(f \bmod p, g \bmod p)$ in $\mathbb{F}_p[x]$ irreducible of degree n

1 repeat
2 $\quad$ Select $g_1(x), g_0(x)$, two polynomials with small integer coefficients, $\deg g_1 < \deg g_0 = n$
3 $\quad$ Select small integers a, b, c, d
4 $\quad$ $\psi(x) = g_0(x) + \left(\dfrac{a + by}{c + dy} \bmod p \right) g_1(x)$
5 $\quad$ $f \leftarrow \mathrm{Res}_Y \left(P(Y), (c + dY)g_0(x) + (a + bY)g_1(x) \right)$
6 $\quad$ $g \leftarrow (c + dy)g_0(x) + (a + by)g_1(x)$ $\qquad$ *// $g \equiv (c + dy)\psi(x) \bmod p$*
7 until *$\psi(x)$ is irreducible in $\mathbb{F}_p[x]$ and f, g are irreducible in $\mathbb{Z}[x]$*
8 return (f, g, ψ)

In practice, in Algorithm 2 one might prefer to constrain $d = 0$, so that g has small leading coefficient c. Going further and requiring $c = 1$ so that g is monic reduces however too much the possibilities to find a good pair of polynomials.

The following example has been obtained with Algorithm 2, searching over all $(a + by)/c$ with $|a|, |b|, |c| \leq 256$.

$y = -8702303353090049898316902$ is the targeted MNT curve parameter
$f = 108x^6 + 1116x^5 + 3347x^4 + 2194x^3 - 613x^2 - 468x + 108$
$g = 6x^3 + 34809213412360199593267639\ x^2 + 34809213412360199593267621\ x - 6$
$\quad = 6x^3 - (4y - 31)x^2 - (4y - 13)x - 6$
$\varphi = \frac{1}{6}g \bmod p = x^3 + 15146016729840465134625816509459896150600476996 6481\ x^2$
$\quad\quad + 15146016729840465134625816509459896150600476996 6478\ x - 1$