A Note on the Galbraith–Petit–Silva Cryptosystem

Benjamin E. Diamond
J.P. Morgan

Abstract

We describe a handful of practical enhancements to the cryptosystem of Galbraith, Petit, and Silva [GPS20], which improve the scheme's concrete efficiency. Our primary innovation is the application of an algorithm of Elkies [Elk98] to the generation of torsion bases. This technique may be of independent interest. We finally describe a full implementation of our improved protocol.

1 Introduction

The identification and signature scheme of Galbraith, Petit and Silva [GPS20] represents an appealing option among isogeny-based cryptosystems. As its authors mention, this scheme—as compared with that of De Feo, Jao, and Plût [DFJP14]—invokes a more "standard" hardness assumption (namely, the endomorphism ring computation problem), and in particular, dispenses with the sending of auxiliary points and the use of a special prime. The best known attacks on this protocol are of exponential complexity, even for quantum adversaries. For further details, we refer to the original paper [GPS20].

Unfortunately, the GPS scheme is not at present efficient enough for practical use. In this short note, we describe a number of measures which shrink—although do not eliminate—this gap.

We follow the notation of Blake, Seroussi, and Smart [BSS99, §VII]. In particular, we fix a finite field $\mathbb{F}_q$ with characteristic $p > 3$, as well as an elliptic curve E_0 over $\mathbb{F}_q$ in Weierstrass form. We do *not* require that q be prime. In fact, following [GPS20, §2.2], we assume in what follows that E_0 is supersingular, and set $q = p^2$.

2 Elkies' Algorithm

We begin with our main technique.

2.1 Overview, and relevance to [GPS20]

We recall the algorithm of Elkies [Elk98] (see also Schoof [Sch95, §7]), following the treatment of [BSS99, §VII.2]. We furthermore take for granted the improvements of Müller and Maurer [MM01], discussed also in [BSS99, §VII.4.2].

Elkies' algorithm concerns a prime l, unequal to p, for which the characteristic equation $\mathcal{F}_l(u)$ splits over $\mathbb{F}_l$. For any such l, the l-torsion subgroup $E_0[l]$ of E_0 (which is isomorphic to $\mathbb{Z}/l\mathbb{Z} \times \mathbb{Z}/l\mathbb{Z}$) contains at least one subspace C invariant under the action of the Frobenius endomorphism φ_q. Elkies' highly nontrivial algorithm (see e.g., [BSS99, §VII.4]) yields a polynomial $F_l(x)$, of degree $\frac{l-1}{2}$, whose roots consist exactly of the x-coordinates of the non-identity points of C. In particular, $F_l(x)$ divides the division polynomial $f_l(x)$.

The utility of the factor $F_l(x)$ for our purposes resides in its ability to concretely deliver elements of $E_0[l]$. (Any root of $F_l(x)$ over its splitting field will yield such a point's x-coordinate.) Elements of $E[l]$ are demanded in line 2 of [GPS20, Alg. 2], which stipulates that a basis $\{P_{i,1}, P_{i,2}\}$ of the l_i-torsion subgroup of E_0 be constructed. (We use only primes l_i, and not prime powers, in order to maintain compatibility with Elkies' algorithm.)

The proof of [GPS20, Lem. 5] suggests that the factors $F_l(x)$ (and hence the required basis elements) be obtained by factoring the order-$l^2 - 1$ *division polynomial* $f_l(x)$. This is exactly the inefficient step which Elkies' improvement to Schoof's algorithm serves to obviate. To drive home this point, we directly quote [BSS99, §VII.2.1]:

> In particular, the straightforward approach of attempting to factor $f_l(x)$ to obtain $F_l(x)$ does not seem to work, as one of the first steps in such a factorization would involve a computation of the type $x^q \pmod{f_l(x)}$, precisely what we are trying to avoid.

In practice, the factorization of the division polynomials $f_l(x)$ is by far the slowest part of [GPS20] as written, at least in concrete terms (the analysis of [GPS20, Lem. 5] suggests that finding roots of the factors $F_l(x)$ is asymptotically slower).

2.2 Applicability in the supersingular setting

We justify our use of Elkies' algorithm in this setting. We note first that, because each "base curve" E_0 at issue is *supersingular*, with $\#E_0(\mathbb{F}_{p^2}) = (p-1)^2$, the characteristic equation of its q^{th}-power Frobenius φ_q splits *over* $\mathbb{Z}$ as

$$0 = \varphi_q^2 - -2p\varphi_q + p^2 = (\varphi_q + p)^2.$$

(We refer to Silverman [Sil09, Thm. 2.3.1] for preliminaries.) This equation's reduction modulo l thus also splits over $\mathbb{F}_l$, regardless of l. Put differently, the q^{th}-power Frobenius φ_q acts as a scalar endomophism—that is, as multiplication by $-p$—on E_0, and hence also on each torsion subgroup $E_0[l]$. We conclude that (in this supersingular setting) *each* prime l is "Elkies", and in fact that every subspace $C \subset E_0[l]$ is invariant under φ_q (for each l).

We now observe that the subspace $C \subset E_0[l]$ to which the Elkies factor $F_l(x)$ corresponds is controlled by the initial j-invariant $\tilde{j} \in \mathbb{F}_{p^2}$ selected in step (i) of [BSS99, p. 123]. Because E_0 is supersingular, $\Phi_l(x, j(E_0))$ splits completely over $\mathbb{F}_{p^2}$; any of its roots $\tilde{j}$ will do. In practice, we use instead the Müller modular polynomials $G_l(x, j(E_0))$ (as in [BSS99, VII.4.2]) for which the same reasoning applies. By (arbitrarily) varying $\tilde{j}$, we thus likewise vary C, and in turn generate using the Elkies procedure (two) *distinct* factors $F_l(x)$ of $f_l(x)$. These factors in turn yield the required $\mathbb{F}_l$-independent elements of $E_0[l]$.

2.3 Multiple roots

An issue arises from the fact that E_0 as defined in [GPS20, §4.1]—that is, using the equation $y^2 = x^3 + x$ over $\mathbb{F}_{p^2}$, for $p \equiv 3 \mod 4$—satisfies $j(E_0) = 1728$, a j-invariant for which the specialization $\Phi_l(x, j(E_0))$ has multiple roots. For any such E_0, the method of Elkies necessarily fails; we refer for example to Galbraith [Gal12, §25.2.1], which assumes "that $\tilde{j}$ is a simple root" of $\Phi_l(x, j(E_0))$. I would also like to thank Andrew Sutherland for helping to explain this.

Our solution is to replace E_0 as in [GPS20, §4.1] by a differently chosen base curve, whose endomorphism ring in particular resides in the *final* row of the classification of Pizer [Piz80, Prop. 5.2] (see also [EHL+18, Prop. 1]). In particular, we choose p for which $p \equiv 1 \mod 8$, and produce E_0 using an algorithm of Eisenträger, Hallgren, Lauter, Morrison and Petit [EHL+18, Alg. 3], which in turn invokes Bröker [Brö09]. In fact, we require furthermore that $p \equiv 1 \mod 3$, so that $\left(\frac{-3}{p}\right) \neq -1$; this in turn guarantees that [Brö09, Alg. 2.4] does *not* set $q = 3$ in step 3., and so terminate in step 6. by returning E_0 as $y^2 = x^3 - 1$. (This latter curve satisfies $j(E_0) = 0$, and so too would yield multiple roots.) In particular, we require that $p \equiv 1 \mod 24$. Curves E_0 chosen in this way yield specializations $\Phi_l(x, j(E_0))$ which lack multiple roots, at least with high probability, and are suitable for the use of Elkies' algorithm.

We note that a suitable base curve E_0, could, alternatively, be obtained by *starting* at $y^2 = x^3 + x$ (with $p \equiv 3 \mod 4$) and then taking a short, arbitrary walk (even of one step). This approach would convey the benefit of admitting the use of "more" primes p (we discuss this matter further below). On the other hand, it would make the endomorphism $\text{End}(E_0)$—and in particular, the four self-maps generating it—harder to represent and compute. In particular, these would depend on the initial walk, and something like [EHL+18, Alg. 5] would become necessary.

3 Preprocessing, and Implications

As [GPS20, p. 166] observe, the computation of the bases $\{P_{i,1}, P_{i,2}\}$ can be outsourced to a preprocessing step. We adopt this measure as well.

One implication of this recourse, not discussed in [GPS20], is that the lines 10 and 14 of [GPS20, Alg. 1] cannot proceed as written. Indeed, adding small cofactors to S_1 and S_2 will subvert the execution of [GPS20, Alg. 2], which assumes (in the preprocessing setting) that torsion bases $\{P_{i,1}, P_{i,2}\}$ for *each* l_i dividing S_1 and S_2 have already been constructed. (One approach, which we have not explored, would involve constructing "spare" torsion bases to accommodate these cofactors.)

Instead, we slightly modify [GPS20, Alg. 1]. If either of the loops 9–11 or 16–18 fail sufficiently many times—that is, if either of the attempts to run Cornacchia's algorithm repeatedly fail, on negative, or non-prime arguments—then we restart the entire algorithm, with a fresh choice of N. Similarly, we discard, and obtain afresh, even a successfully obtained element $\beta_1 = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$ (as in line 12) if the resulting element $\beta_2 = C\mathbf{j} + D\mathbf{k}$ (of line 14) fails to immediately satisfy $\left(\frac{(pC^2+pqD^2)\cdot S_2}{N}\right) = 1$. (We note that the reduced norm map, in our setting—in light of our unusually chosen E_0, discussed above—is given instead by $(a, b, c, d) \mapsto a^2 + qb^2 + pc^2 + pqd^2$, where q is as in [EHL$^+$18, Prop. 1].) We leave for future work a rigorous argument that this modification does not impact the security analysis of [GPS20, Lem. 3].

4 Choice of p

We now discuss how an appropriately chosen characteristic can *further* improve efficiency. In short, we describe a technique to choose p so as to minimize the degrees of the respective field extensions over which the torsion subgroups $E[l_i]$ are defined.

These field-extension degrees control the efficiency of the algorithm in multiple important respects. For one, when torsion bases are first *constructed*—that is, during the preprocessing phase—roots must be found in these field extensions (that is, of the Elkies polynomials, which yield the basis elements' respective x-coordinates, from which the y-coordinates in turn arise as square roots). More importantly, they also control the rate-limiting steps of the algorithm's online portion. The main source of *online* inefficiency of [GPS20], indeed, resides in the loop of lines 7–10 of [GPS20, Alg. 3] (and in particular the check of line 10), as well as, to a lesser extent, line 12 of [GPS20, Alg. 2]. Both of these depend heavily on the field-extension degrees of the elements of $E[l_i]$.

Our important observation here—which evokes comments made in CSIDH [CLM$^+$18]—is that the extension degree of $E[l_i]$ is given exactly by the multiplicative order of $-p$ modulo l_i. This follows from the fact that φ_q acts as multiplication by $-p \pmod{l_i}$ on $E[l_i]$, together with a Galois-theoretic argument. Indeed, we denote the field at issue by K_{E,l_i}. The natural representation $\mathrm{Gal}(\overline{K}/\mathbb{F}_{p^2}) \to \mathrm{Aut}(E[l_i])$ vanishes exactly on the subgroup $\mathrm{Gal}(\overline{K}/K_{E,l_i})$, and so induces an injection $\mathrm{Gal}(K_{E,l_i}/\mathbb{F}_{p^2}) \to \mathrm{Aut}(E[l_i])$. Whereas the order of the left-hand group gives the desired extension degree, it suffices to determine the order of its image. Yet this image is exactly that cyclic subgroup of $\mathrm{Aut}(E[l_i])$ generated by (the image of) the Frobenius endomorphism φ_{p^2}; the latter map's order is in turn that of $-p \cdot I_2 \in \mathrm{GL}_2(\mathbb{Z}/l_i\mathbb{Z})$. (For a good exposition of these ideas, we refer to Van Tuyl [VT97].)

In light of this, we choose p so that $-p$ has "low order" in *each* group $(\mathbb{Z}/l_i\mathbb{Z})^*$, where l_i varies throughout those primes necessary to establish "adequate mixing" in p's supersingular isogeny graph (as defined by [GPS20, Thm. 1] and [GPS20, Lem. 1]). (In fact, we use *two disjoint* sets of such l_i, representing the respective factors of the coprime integers S_1 and S_2 of [GPS20, §4.5].) In particular, we set p as the:

$$\arg\min_{p \in P} \left(\max_{l_i \in L_p} (\mathrm{ord}_{l_i}(-p)) \right),$$

where P is some suitable set of primes—say, with prescribed bit-length and residue class behavior—and, for each $p \in P$, $L_p = L_{1,p} \cup L_{2,p}$ gives a set of small primes l_i which establish mixing in p's isogeny graph (twice over).

5 Implementation

We now describe our implementation, available open-source at `benediamond / galbraith-petit-silva`. Our primary dependency is a fork of the number-theory library LiDIA, whose point-counting functionality (including the full Schoof–Elkies–Atkin algorithm) we have extended to the *composite* field case.

5.1 Choice of parameters

Targeting $\lambda = 8$ bits of post-quantum security, we select a 32-bit prime p (see [GPS20, §4.6]). In particular—and in light of the considerations discussed above—we set $p = 4294022113 = 2^{32} - 945183$. The sets

$$L_{1,p} = \{3, 11, 17, 23, 31, 41, 47, 59, 67, 73, 83, 97, 103, 109, 127, 137, 149, 157\}\,,$$

and independently,

$$L_{2,p} = \{5, 13, 19, 29, 37, 43, 53, 61, 71, 79, 89, 101, 107, 113, 131, 139, 151\}\,,$$

both (respectively) achieve mixing in p's supersingular isogeny graph. (The prime $q = 7$ is omitted, as it serves as the "special" value as in [EHL$^+$18, Prop. 1].) We note that $\text{ord}_{l_i}(-p)$ is unusually low for each l_i above, and in fact obtains a *maximum* value of 72 over all l_i. In fact, p minimizes this maximum, throughout those primes "near" 2^{32} which in addition satisfy $p \equiv 1 \mod 24$.

5.2 Preprocessing and torsion-basis generation

Our implementation—which in particular uses Elkies' algorithm—was able to generate torsion bases for all primes l_i above in about 990 minutes' (16.5 hours') worth of work on a single laptop CPU. The vast bulk of the work involved finding the roots of the Elkies factors over their splitting fields, as opposed to constructing these factors in the first place (which was near-instant). Generating parameters of this size would have been essentially unfeasible if division polynomials (with degrees as high as $\frac{157^2-1}{2} = 12324$) had been directly factored.

5.3 Online performance

We use the Unruh transform, as described in [GPS20, §4.6], throughout. In the measurements which follow, we take for granted and exploit the preprocessing procedure described above.

Generating a keypair—including not just an isogeny from E_0 to E_1, but also the ideal representing it (obtained using [GPS20, Alg. 3])—takes 1.8 minutes in our implementation.

Our implementation takes about 182 minutes to sign a message. We recall that the main identification protocol of [GPS20, Fig. 1] needs to be run $t = 3\lambda$—or, in our case, 24—distinct times; these individual executions are embarrassingly parallel, and could in principle be parallelized. Each individual execution takes about 7.6 minutes.

Our verification routine explicitly uses (and factors) modular polynomials, as made available by Sutherland, and constructed using the techniques of Bröker, Lauter and Sutherland [BLS12]. This routine also involves ($t = 24$) independent, parallel executions of the identification protocol [GPS20, Fig. 1], and so could also be parallelized. Its total runtime on a single thread is about 15 minutes.

5.4 Key and signature sizes

We use the isogeny path representation 3. of [GPS20, p. 140], as well a naive (i.e., one byte per character) encoding scheme. Our private and public keys are 52 and 10 bytes, respectively. Our total signature size is 3050 bytes.

References

[BLS12] Reinier Bröker, Kristin Lauter, and Andrew V. Sutherland. Modular polynomials via isogeny volcanoes. *Mathematics of Computation*, 81(278):1201–1231, 2012.

[Brö09] Reinier Bröker. Constructing supersingular elliptic curves. *J. Comb. Number Theory*, 1(3):269–273, 2009.

[BSS99] I. Blake, G. Seroussi, and N. Smart. *Elliptic Curves in Cryptography.* Cambridge University Press, Cambridge, 1999.

[CLM$^+$18] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018*, pages 395–427. Springer International Publishing, 2018.

[DFJP14] L. De Feo, D. Jao, and J. Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *Journal of Mathematical Cryptology*, 8(3):209–247, 2014.

[EHL$^+$18] Kirsten Eisenträger, Sean Hallgren, Kristin Lauter, Travis Morrison, and Christophe Petit. Supersingular isogeny graphs and endomorphism rings: Reductions and solutions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018*, pages 329–368, Cham, 2018. Springer International Publishing.

[Elk98] Noam D. Elkies. Elliptic and modular curves over finite fields and related computational issues. In D. A. Buell and J. T. Teitelbaum, editors, *Computational Perspectives on Number Theory*, Studies in Advanced Mathematics, pages 21–76. AMS, 1998.

[Gal12] Steven D. Galbraith. *Mathematics of Public Key Cryptography.* Cambridge University Press, 2012.

[GPS20] Steven D. Galbraith, Christophe Petit, and Javier Silva. Identification protocols and signature schemes based on supersingular isogeny problems. *Journal of Cryptology*, 33(1):130–175, 2020.

[MM01] Markus Maurer and Volker Müller. Finding the eigenvalue in Elkies' algorithm. *Experiment. Math.*, 10(2):275–286, 2001.

[Piz80] Arnold Pizer. An algorithm for computing modular forms on Γ0(n). *Journal of Algebra*, 64(2):340–390, 1980.

[Sch95] René Schoof. Counting points on elliptic curves over finite fields. *Journal de théorie des nombres de Bordeaux*, 7(1):219–254, 1995.

[Sil09] Joseph H. Silverman. *The Arithmetic of Elliptic Curves.* Graduate Texts in Mathematics. Springer, 2009.

[VT97] Adam Leonard Van Tuyl. The field of N-torsion points of an elliptic curve over a finite field. Masters thesis, September 1997.