# Faster Algorithms for Isogeny Problems using Torsion Point Images

Christophe Petit

School of Computer Science, University of Birmingham

**Abstract.** [1] There is a recent trend in cryptography to construct protocols based on the hardness of computing isogenies between supersingular elliptic curves. Two prominent examples are Jao-De Feo's key exchange protocol and the resulting encryption scheme by De Feo-Jao-Plût. One particularity of the isogeny problems underlying these protocols is that some additional information is given in input, namely the image of some torsion points with order coprime to the isogeny. This additional information was used in several active attacks against the protocols but the current best passive attacks on the protocols make no use of it at all.

In this paper, we provide new algorithms that exploit the additional information provided in isogeny protocols to speed up the resolution of the underlying problems. Our techniques lead to a heuristic polynomial-time key recovery on a non-standard variant of De Feo-Jao-Plût's protocols in a plausible attack model. This shows that at least some isogeny problems are easier to solve when additional information is leaked.

## 1 Introduction

Following calls from major national security and standardization agencies, the next cryptographic standards will have to be "post-quantum secure", namely they will have to rely on computational problems that will (at least to the best of knowledge) remain hard for quantum computers. Several directions are currently explored for post-quantum cryptography, including lattice-based cryptography, code-based cryptography, multivariate cryptography, hash-based cryptography and most recently cryptography based on isogeny problems. The latter are appealing for their mathematical elegance but also for the relatively small key sizes compared to other post-quantum candidates.

The interest in isogeny problems as potential cryptographic building blocks is relatively new, and there has therefore not been been much cryptanalytic work on them. The most established isogeny problem is the endomorphism ring computation problem, which was already considered by Kohel in his PhD thesis [10]. In the supersingular case this problem is (heuristically at least) equivalent to the problem of computing an isogeny between two randomly chosen curves, and it remains exponential time even for quantum algorithms today.

The supersingular key exchange protocol of Jao-De Feo [9] and the encryption scheme and signature schemes that are derived from it [5, 7, 20] rely on variants of

[1] This work was developed while the author was at the Mathematical Institute of the University of Oxford, funded by a research grant from the UK government.

these problems, where special primes and relatively small degree isogenies are used. More importantly for this paper, the attacker is provided with the image by the isogeny of a large torsion group, in addition to the origin and image curves. Although it was observed that this additional information could a priori make the problems easier, all security evaluations against passive attacks were based on a meet-in-the-middle strategy that makes no use at all of it.

### 1.1 Contributions

In this paper, we study the impact of revealing the images of torsion points on the hardness of isogeny problems. We provide new techniques to successively exploit this additional information and improve on the best previous attacks, namely meet-in-the-middle attacks (see Section 2). Among other results, these techniques lead to polynomial-time algorithms to compute isogenies between two curves $E_0$ and $E_1$ assuming

1. Some non scalar endomorphisms of $E_0$ are known and/or are of small degree.
2. The images of $N_2$ torsion points are revealed, where $N_2$ is significantly larger than the degree of the isogeny $N_1$.

So far our techniques do not invalidate the parameters proposed in the original protocol (where $N_1 \approx N_2$). However, we describe two natural variants, which we call unbalanced variant and optimal degree variant, which are affected in plausible attack scenarii. We believe these generalizations are of independent interest, as they have some advantages over the original protocol when appropriate parameters are chosen.

Our main contribution in this paper is our new attack techniques. We illustrate their potential with the following results:

1. (Section 3.) A near to square root speedup on the problem of computing an endomorphism of a supersingular elliptic curve of a certain degree, when provided with some torsion point images through this endomorphism.
2. (Section 4.3.) A polynomial time key recovery attack on our optimal length variant, provided $N_2 > N_1^4$ and $E_0$ is "special" (such special curves were suggested in previous implementations [2, 5] for efficiency reasons).
3. (Section 4.4.) A polynomial time key recovery attack on both variants, provided $\log N_2 = O(\log^2 N_1)$ and $E_0$ has a small degree endomorphism.

These attacks show that (at least some) isogeny problems are easier to solve when the image of torsion points through the isogeny is revealed. Some of these attacks require further assumptions on $N_2$; we refer to the next sections for details. We provide a heuristic analysis for all these attacks. The heuristics used involve factorization patterns and other properties of particular numbers appearing in our algorithms, which we treat as random numbers of the same size. For the first two attacks these heuristics are very plausible and it may be possible to remove them. For the third attack they are still a priori plausible, but they may be very hard to remove. Indeed this this attack involves a recursive step, and a rigorous result would have to take into account correlations between successive steps. For this reason we additionally provide some experimental support for our third attack.

We believe the three attacks we develop here are only some examples of what our new techniques can achieve, and we leave further developments to further work.

### 1.2 Background Reading

We refer to the books of Silverman [14] and Vignéras [17] for background results on elliptic curves and quaternion algebras. Recent cryptographic constructions based on isogeny problems include [1,5,7,9,13,19]. Computational aspects related to isogenies are covered in David Kohel's PhD thesis [10] and more recently in [6,7].

### 1.3 Outline

In Section 2 we first describe the supersingular key exchange protocol of Jao-De Feo [9], then we propose a generalization of this protocol, and finally we recall the most relevant cryptanalysis results on this protocol. In Section 3 we describe faster algorithms to compute an endomorphism of a given supersingular elliptic curve, given the image of torsion points by this endomorphism. In Section 4 we turn to the problem of computing an isogeny between two supersingular elliptic curves given the images of torsion points by this isogeny, and we describe two attacks faster than the state-of-the-art meet-in-the-middle algorithm in this context. Finally, we summarize the impact of our techniques and results in Section 5, and we give perspectives for further work.

## 2 Supersingular Isogeny Key Exchange

### 2.1 Jao-De Feo's Key Exchange

We recall the supersingular key exchange protocol of Jao-De Feo [9].

*Setup* Let $\ell_1, \ell_2$ be two small primes. Given a security parameter $\lambda$, let $e_1, e_2$ be the smallest integers such that $\ell_1^{e_1}, \ell_2^{e_2} \geq 2^{2\lambda}$ (or $2^{3\lambda}$ for post-quantum security). Let $f$ be the smallest integer such that $p = \ell_1^{e_1}\ell_2^{e_2}f - 1$ is prime. Let $E_0$ be a supersingular elliptic curve over $\mathbb{F}_{p^2}$. Let $P_1, Q_1$ and $P_2, Q_2$ be respectively bases of the $\ell_1^{e_1}$ and $\ell_2^{e_2}$ torsions on $E_0$.

*First round* Alice chooses a random cyclic subgroup of order $\ell_1^{e_1}$, say $G_1 = \langle \alpha_1 P_1 + \beta_1 Q_1 \rangle$ with at least one of $\alpha_1, \beta_1$ coprime to $\ell_1$. She computes the corresponding isogeny $\phi_1$ and image curve $E_1$, as well as $\phi_1(P_2)$ and $\phi_1(Q_2)$. She sends $E_1$, $\phi_1(P_2)$ and $\phi_1(Q_2)$ to Bob. Bob proceeds similarly, permuting the roles of $\ell_1$ and $\ell_2$.

*Second round* Upon receiving $E_2$, $\phi_2(P_1)$ and $\phi_2(Q_1)$, Alice computes $G_1' = \langle \alpha_1\phi_2(P_1) + \beta_1\phi_2(Q_1)\rangle$, the corresponding isogeny $\phi_1'$, the image curve $E_{12} = E/\langle G_1, G_2\rangle$ and its $j$-invariant $j_{12}$. Bob computes $j_{21} = j_{12}$ similarly with the information sent by Alice. The shared secret is the value $j_{12} = j_{21}$, or the result of applying some key derivation function to this value.

The protocol is summarized in the following commutative diagram:

$$\begin{tikzcd}
 & E_1 = E_0/G_1 \arrow[rd] & \\
E_0 \arrow[ru, "\phi_1"] \arrow[rd, "\phi_2"'] & & E_{12} = E_0/\langle G_1, G_2 \rangle \\
 & E_2 = E_0/G_2 \arrow[ru] &
\end{tikzcd}$$

This protocol can be broken if one can compute isogenies between two given curves. However we stress that the curves appearing in this protocol are closer to each other in the isogeny graphs than random curves would be: indeed for any fixed $E_0$ there are only $(\ell_i + 1)\ell_i^{e_i - 1} \approx \sqrt{p}$ possible curves for $E_1$, while there are roughly $p/12$ supersingular $j$-invariants over $\mathbb{F}_{p^2}$. This allows more efficient meet-in-the-middle attacks in complexity $O(\sqrt[4]{p})$ instead of $O(\sqrt{p})$ for a generic curve pair. More importantly for this paper, some information on the isogenies is leaked by the protocol, as the image of a full torsion coprime with the isogeny degree is revealed. Finally, special primes are used to ensure that the $\ell_i^{e_i}$ torsions are defined over $\mathbb{F}_{p^2}$. For arbitrary $p$ these torsions subgroups would be defined over large field extensions, resulting in an inefficient protocol.

*Remark* Let $N_1 = \ell_1^{e_1}$. If the image of the $N_1$ torsion by a degree $N_1$ isogeny was revealed it would be straightforward to recompute the isogeny, as this image would be the kernel of the dual isogeny. More generally if $N_1$ is not coprime with the degree then part of the isogeny can be recovered efficiently.

### 2.2 Unbalanced and Optimal Degree Variants

We now present two variants of the protocol, which we call unbalanced and optimal degree variants.

*Unbalanced variant.* In their paper Jao and De Feo suggested parameters such that $\ell_1^{e_1} \approx \ell_2^{e_2}$. We suggest to generalize the setup to allow for unbalanced parameters $\ell_2^{e_2} \gg \ell_1^{e_1}$ in some contexts. The size of $\ell_i^{e_i}$ determines the security of the corresponding secret key $G_i$ with respect to all previous attacks (see next subsection), while the size of $p$ would influence efficiency. Jao and De Feo therefore chose $\ell_1^{e_1} \approx \ell_2^{e_2}$ to provide the same security level on both Alice and Bob's ephemeral keys. However in some contexts as in the public key encryption scheme [5] one secret key is static and it may therefore make sense to protect it more strongly. This is achieved by our unbalanced variant.

In the unbalanced variant, the setup procedure takes two security parameters $\lambda_1$ and $\lambda_2$ in entry. For $i = 1, 2$ it computes the smallest integer $e_i$ such that $\ell_i^{e_i}, \geq 2^{2\lambda_i}$ (or $2^{3\lambda_i}$ for post-quantum security), and then the smallest integer $f$ such that $p = \ell_1^{e_1}\ell_2^{e_2}f - 1$ is prime. The rest of the protocol is like in Jao and De Feo.

*Optimal degree variant.* We now generalize the parameters such that the isogeny degrees are large enough to ensure uniform distribution of $E_i$ among all curves on the isogeny graphs, and moreover such that arbitrary primes $p$ can be used.

We recall that a number $N = \prod p_i^{e_i}$ is $B$-powersmooth if for all $i$ we have $p_i^{e_i} < B$. In this paper we say that a number is powersmooth if it is $B$-powersmooth for some bound $B$ that is polynomial in the security parameter. For an arbitrary prime $p$, we replace $\ell_1^{e_1}$ and $\ell_2^{e_2}$ in the protocol by any powersmooth numbers $N_1$ and $N_2$ that are coprime to each other and of size about $p^2$. Note that the $N_1$ and $N_2$ torsions are a priori not defined over $\mathbb{F}_{p^2}$; however the powersmooth requirement ensures that they can be efficiently represented in a Chinese remainder manner (see [7]). On the other hand, the coprimality requirement ensures that the isogeny diagram commutes as in the original protocol. Finally, the condition $N_i \approx p^2$ on the isogeny degrees guarantees that $E_1$ and $E_2$ are close to uniformly distributed [7], while for the original parameters and the unbalanced variant above we have $N_1 N_2 \approx p$.

In the optimal degree variant, the setup procedure takes a security parameter $\lambda$. It chooses a random prime $p$ with $2\lambda$ bits (or $3\lambda$ bits for post-quantum security). Then $N_1$ and $N_2$ are chosen coprime to each other, such that both of them are powersmooth and have at least $2\log p$ bits. Then for each maximal prime power $\ell_j^{e_j}$ dividing either $N_1$ or $N_2$ we fix a basis for the $\ell_j^{e_j}$ torsion. Note that this is defined over an extension field of degree at most $2\ell_j^{e_j}$, which is polynomial in $\lambda$.

If $N_1 = \prod p_j^{e_j}$ then in the first round Alice chooses for each $j$ one cyclic subgroup $G_{1j} = \langle \alpha_j P_j + \beta_j Q_j \rangle$ with at least one of $\alpha_j, \beta_j$ coprime to $p_j$. This implicitly defines a cyclic subgroup $G_1$ of order $N_1$ such that $G_i = G_{ij} \mod E_0[\ell_j^{e_j}]$. She computes the corresponding isogeny $\phi_1$ as a composition of isogenies of prime degrees, the image curve $E_1 = E_0/G_1$, and the image by $\phi_1$ of the $\ell_j^{e_j}$ torsion basis points, for each $\ell_j^{e_j}$ dividing $N_2$. Alice sends $E_1$ and all torsion point images to Bob. Note that although the torsion points and their images are defined over some field extensions, all isogenies computed are defined over $\mathbb{F}_{p^2}$. Moreover the degree of any extension field involved is bounded by $2\ell_j^{e_j}$ which is polynomial in the security parameter, so all elements can be efficiently represented and the computation runs in polynomial time. Bob proceeds similarly.

In the second round, Alice computes $\phi_2(G_{1j})$ using the information sent by the other party (as in the original protocol), then she computes $E_2/\phi_2(G_1)$ as above, and finally the $j$-invariant of this curve. Bob proceeds similarly.

Because it allows both for arbitrary primes and for "large enough" degree isogenies, the optimal degree variant can a priori be more secure than the original protocol. On the other hand, working over field extensions, even of moderate degrees, may have a significant efficiency cost in practice. We leave a precise complexity estimation and a thorough comparison of this variant with the original protocol to further work.

*Remark.* Of course, one could also allow intermediate parameters where $\gcd(N_1 N_2, p^2 - 1)$ is a medium size factor of $p^2 - 1$ to ensure that the primes are not too special and at the same limit the size of the extension fields needed.

### 2.3 State-of-the Art on Cryptanalysis

We refer to [7] for a thorough discussion of existing cryptanalysis results, and only focus on the most relevant work here. With the exception of an active attack in [6], previous cryptanalysis results have ignored the additional information revealed in De Feo-Jao-Plût's protocols. They therefore considered the following problem:

**Problem 1** *Let $N$ be a positive integer, let $p$ be a prime and let $E_1, E_2$ be two supersingular elliptic curves defined over $\mathbb{F}_{p^2}$, such that there exists an isogeny $\phi$ of degree $N$ such that $E_2 = E_1/\ker\phi$. Compute $\phi$.*

*Remark.* The most natural representation of $\phi$ is some canonical representation as two elements of the function field $E_1(x, y)$. In cryptographic contexts the degree of $\phi$ is of exponential size so this representation is not efficient. However in these contexts the degree is often a smooth number so that the isogeny can be efficiently returned as a composition of rational maps.

When $N$ is large enough any pair of elliptic curves are connected by an isogeny of degree $N$, and this problem is heuristically equivalent to the endomorphism ring computation problem (see [7]). In De Feo-Jao-Plût's protocols, however, $N = O(\sqrt{p})$ is too small to ensure this, and as $N$ is moreover smooth one can do a meet-in-the-middle attack with complexity $O(\sqrt[4]{p})$ (respectively $O(\sqrt[6]{p})$ with a quantum computer) even if the endomorphism ring computation problem remains of complexity $O(\sqrt{p})$ (respectively $O(\sqrt[3]{p})$ with a quantum computer). We stress that the optimal degree variant we introduced above does not suffer from this problem, as isogeny degrees are chosen large enough to ensure a uniform distribution of $E_2$.

The following lemma generalizes the meet-in-the-middle strategy when the smooth bound on $N$ is not polynomial in $\log p$.

**Lemma 1.** *Assume $N = N_1 \cdot N_2$ where both $N_1$ and $N_2$ are $B$-smooth. Then the meet-in-the-middle strategy has a time and memory complexity $O(B \max(N_1, N_2))$, neglecting log factors.*

PROOF: The factorization of $N$ can be obtained in subexponential time, which is negligible with respect to $\max(N_1, N_2)$. Isogenies of prime degree can be computed in quasilinear time in the degree. The meet-in-the-middle strategy computes $O(N_1)$ isogenies of degree $N_1$ and $O(N_2)$ isogenies of degree $N_2$, each of them as a composition of isogenies of degree at most $B$. □

The active attack presented in [6] runs $O(\log p)$ executions of the key exchange protocol with the same party. Assuming this party uses a static secret key $G_1$, the attacker provides them with incorrect values for $\phi_2(P_1), \phi_2(Q_1)$, observes variations in the resulting shared key $j(E_{12})$, and progressively deduces the key $G_1$. The loop-abort fault attack developed in [8] is similar to this attack. A fault attack is also used in [16] to replace $\phi_2(P_1)$ and $\phi_2(Q_1)$ by points whose order is not coprime with the isogeny degree. Our goal in this paper is to show how to exploit the "torsion image" information revealed in De Feo-Jao-Plût's protocols, but in passive attacks.

# 3 Computing an Endomorphism from Additional Information

From a computational number theory point of view, computing endomorphisms of a curve is a somewhat more natural task than computing isogenies between two curves. At the same time, there are strong relations between the two problems (see [7]). In this section we define an "endomorphism computation" counterpart to De Feo-Jao-Plût's isogeny problem, and we show how leaking the image of torsion points helps in solving this problem.

## 3.1 Endomorphism Computation Problem with Additional Information

We consider the following problem:

*Problem 1.* Let $p$ be a prime and let $E$ be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$. Let $\phi$ be a non scalar endomorphism of $E$ with smooth order $N_1$. Let $N_2$ be a smooth integer with $\gcd(N_1, N_2) = 1$, and let $P, Q$ be a basis of $E[N_2]$. Let $R$ be a subring of $\mathrm{End}(E)$ that is either easy to compute, or given. Given $E$, $P$, $Q$, $\phi(P)$, $\phi(Q)$, $\deg\phi$, $R$, compute $\phi$.

*Remark.* This problem is similar to the problem appearing in De Feo-Jao-Plût protocols, with the additional requirement $E_1 = E_2$.

*Remark.* When no endomorphism subring is explicitly given one can take $R = \mathbb{Z}$, namely the scalar multiplications.

*Remark.* If we do not use the additional information the best algorithm for this problem will be a meet-in-the-middle approach: compute all isogenies of degree $\approx \sqrt{N_1}$ from $E$ and search for a collision. As $N_1$ is smooth the cost for each isogeny is polynomial, resulting in an algorithm with roughly $O(\sqrt{N_1})$ complexity.

## 3.2 General Strategy

Our general strategy is summarized in Algorithm 1.

**Algorithm 1** Computing an Endomorphism from Additional Information

```
Require: As in Problem 1, plus parameter $B$.
Ensure: A description of $\phi$ as a composition of low degree maps.
Find $N_1' \in \mathbb{N}$ and $\theta_1, \theta_2 \in R$ such that $n(\theta_1\phi + \theta_2) = N_1' N_2$ and $\gcd(n(\theta_1), N_1) = 1$, and such that $N_1'$ is $B$ smooth and as small as possible.
Let $\theta_1\phi + \theta_2 = \psi_{N_1'}\psi_{N_2}$
Compute $\ker\psi_{N_2}$ using the additional information.
Compute $\psi_{N_1'}$ using a meet-in-the-middle approach.
Compute $\ker\phi = \ker(\theta_1^{-1}(\psi_{N_1'}\psi_{N_2} - \theta_2))$ by evaluating all maps on the $N_1$ torsion.
Compute $\phi$ from $\ker\phi$
```

From what is given in the problem we can compute the image of $\phi$ on any point in $E[N_2]$. Let $\theta_1, \theta_2 \in R$ be known endomorphisms of $E$, to which we associate another endomorphism

$$\psi := \theta_1\phi + \theta_2.$$

Of course we do not know $\phi$ so far, but since we know $\theta_1, \theta_2$, and the action of $\phi$ on $E[N_2]$ we can nevertheless evaluate $\psi$ on any point of $E[N_2]$.

Let us now assume that the maps $\theta_1, \theta_2$ are chosen such that $\deg\psi = N_1'N_2$ for some $N_1' \in \mathbb{Z}$. Algorithms to achieve that together with some additional conditions will be described in the next subsections. The endomorphism $\psi$ can be written as a composition of two isogenies

$$\psi = \psi_{N_1'}\psi_{N_2}$$

with $\psi_{N_1'}$ and $\psi_{N_2}$ respectively of degrees $N_1'$ and $N_2$.

By computing $\psi$ on a basis of $E[N_2]$ and solving some discrete logarithm problem in $E[N_2]$ we deduce the kernel of $\psi_{N_2}$ and then deduce $\psi_{N_2}$ itself. This is efficient since $N_2$ is smooth by assumption.

The map $\psi_{N_1'}$ can then be recovered using a meet-in-the-middle approach. The efficiency of this step depends on the factorization of $N_1'$ and is given by Lemma 1.

At this point, we have computed the map $\psi$ as a composition $\psi = \psi_{N_1'}\psi_{N_2}$. We deduce an expression for $\phi$, namely $\theta_1^{-1}(\psi_{N_1'}\psi_{N_2} - \theta_2)$, and assuming $\gcd(n(\theta_1), N_1) = 1$ we evaluate this map on the $N_1$ torsion to identify $\ker\phi$, from which we recompute a more canonical description of $\phi$. This is efficient as $N_1$ is smooth.

*Remark.* We do not use the additional information to compute $\phi_{N_1'}$. Note that part of the $N_2$ torsion is annihilated by $\phi_{N_2}$ so we only know $\phi_{N_1'}$ and its dual on one cyclic subgroup of the respective $N_2$ torsions.

*Remark.* There is no gain in generality in considering maps of the form $\psi := \theta_1\phi\theta_3 + \theta_2$ for $\theta_1, \theta_2, \theta_3 \in R$. Indeed we have $\theta_1\phi\theta_3 + \theta_2 = \hat{\phi}\hat{\theta}_1\theta_3 + \theta_2 + \mathrm{Tr}(\theta_1\phi)\theta_3$. Taking conjugates we obtain an element $\hat{\psi} = \hat{\theta}_3\theta_1\phi + \hat{\theta}_2 + \mathrm{Tr}(\theta_1\phi)\hat{\theta}_3 \in R\phi + R$ with the same norm. Similarly, there is no gain in generality in using powers of $\phi$ since $\phi^2 = -(\mathrm{Tr}\,\phi)\phi - \deg\phi$.

### 3.3 Attack when $R = \mathbb{Z}$

We first consider the most generic case where the only known endomorphisms of $E$ are scalar multiplications. We define

$$\psi = \psi_{a,b} = a\phi + b$$

for $a, b \in \mathbb{Z}$, which has degree

$$\deg\psi_{a,b} = a^2\deg\phi + b^2 + ab\,\mathrm{Tr}\,\phi = \left(b + a\frac{\mathrm{Tr}\,\phi}{2}\right)^2 + a^2\left(\deg\phi - \left(\frac{\mathrm{Tr}\,\phi}{2}\right)^2\right).$$

Our goal is to find $a, b$ such that $\deg\psi_{a,b} = N_1'N_2$, where $N_1'$ is as small and as smooth as possible.

*Parameter Restriction.* The attack we describe below requires two additional assumptions. First, we require that $N_2 > 2\sqrt{N_1}$. Note that in Jao-De Feo key exchange protocol we have $N_2 \approx N_1$ so the assumption does not look too strong. Second, we require that $-D$ is a square modulo $N_2$, where $D = \deg\phi - \left(\frac{\operatorname{Tr}\phi}{2}\right)^2$. This is a stronger requirement. By Hensel's lifting lemma the requirement is equivalent to $-D$ being a square modulo every odd prime factor of $N_2$ and congruent to 1 modulo 8 when 2 divides $N_2$. The requirement will be satisfied for half of the prime powers. Powersmooth $N_2$ values will have $O(\log p)$ distinct prime factors hence they will need to be specially crafted to satisfy the requirement. However, for random such $N_2$ we can expect to find a factor $N_2'|N_2$ with $N_2' \approx \sqrt{N}_2$ satisfying the requirement. Moreover if $N_2 > N_1$ or $N_2 \approx N_1$ we have a good chance to find $N_2' > 2\sqrt{N_1}$ satisfying the requirement, which can then be used in the attack instead of $N_2$.

*Algorithm* Remember that from the additional information given in the problem we can compute the image of $\phi$ on any point in $E[N_2]$. Note that as $N_1$ and $N_2$ are coprime, $\phi$ is a one-to-one map on $E[N_2]$. From the relation $\phi\hat{\phi} = [\deg\phi]$ we can also compute the image of any point in $E[N_2]$ by the dual map $\hat{\phi}$. We can therefore also evaluate $\operatorname{Tr}\phi$ on $E[N_2]$. By solving a discrete logarithm problem in $E[N_2]$ we deduce $\operatorname{Tr}\phi \bmod N_2$. By the Gram-Schmidt inequality we also have $\operatorname{Tr}\phi < 2\sqrt{\deg\phi}$ so under our first parameter restriction that $N_2 > 2\sqrt{N_1}$ we actually recover $\operatorname{Tr}\phi$ exactly.

Let $D = \deg\phi - \frac{1}{4}\operatorname{Tr}^2\phi$ and let $\tau$ such that $\tau^2 = -D \bmod N_2$. Such a $\tau$ exists under our second parameter restriction, and can be efficiently computed using Tonelli-Shanks algorithm and Hensel's lifting lemma. Points $(x, y)$ in the lattice generated by the two vectors $(N_2, 0)$ and $(\tau, 1)$ correspond to solutions of the equation $x^2 + Dy^2 = 0 \bmod N_2$. We compute a reduced basis for the lattice, with respect to a weighted inner product norm where the second component is weighted by $\sqrt{D}$. This can be done in polynomial time. Finally we let $a := y_0$, $b = x_0 - \frac{\operatorname{Tr}\phi}{2}y_0$ and $N_1' = \frac{x_0^2+Dy_0^2}{N_2}$, where $(x_0, y_0)$ is a well-chosen short vector in the lattice.

To choose $(x_0, y_0)$ we proceed as follows. Using the short basis computed above we generate short vectors and compute the corresponding $N_1'$ values, until we obtain $N_1'$ such that the meet-in-the-middle strategy is efficient enough (see Lemma 1).

*Complexity analysis* We start by analyzing the expected size of $N_1'$. Heuristically, a proportion about $1/N_2$ pairs $(a, b)$ will satisfy the congruence mod $N_2$ so we expect $ab \approx N_2$. If $D < N_2^2$, the mininal vector will have $a \neq 0$ and balance the two terms so $a^2 D \approx b^2$. The two approximations together give $a^2 \approx \frac{N_2}{b}\frac{b}{\sqrt{D}} \approx \frac{N_2}{\sqrt{D}}$ and $c \approx \frac{a^2 D}{N_2} \approx \sqrt{D}$. In general we should expect $D \approx N_1$ so $c \approx \sqrt{N_1}$.

We allow for bigger $N_1'$ values to make sure they are nicely composite. Let heuristically assume that $N_1'$ values obtained with random linear combinations of the two short vectors will factor like random numbers of the same size. The probability that $N_1'$ is $N_1^{1/2n}$ smooth is then equal to $\rho(n)$ where $\rho$ is the Dickman function.

We have $N_1' \approx \sqrt{N_1}$. If $N_1'$ is $N_1^{1/2n}$ smooth then $N_1'$ can be decomposed into two factors $c_1, c_2$ both of them bounded by $N_1^{(n/2+1)/2n}$, so by Lemma 1 the cost for the meet-in-the-middle strategy is $O(N_1^{1/4+1/n})$. So as long as $n > 4$ our approach will beat the straightforward meet-in-the-middle strategy. According to the previous paragraph we expect this to happen with constant probability $\rho(4) \approx 1/200$.

To decrease the cost further, we can choose random values of $N_1'$ until we achieve a subexponential smoothness bound, with a resulting complexity $O(N_1^{1/4+\epsilon})$. This results in a speedup close to square root over the best algorithm when no torsion image is given.

*Improvement when* $\gcd(D, N_2) \neq 1$. For any $r | \gcd(D, N_2)$ there exist $a, b \in \mathbb{Z}$ with $(a\phi + b)/r \in \mathrm{End}(E)$ and $r \nmid \gcd(a, b)$. Moreover we can normalize pairs of this form such that $a = 1$. We can identify the corresponding correct $b$ by trying every possibility until $\phi + b$ annihilates the $r$ torsion. This has a cost $O(r)$. Alternatively we can solve some discrete logarithm problem to find $b$ in at most $O(\sqrt{r})$ operations. In any case since $N_2$ is smooth we can process small factors one at the time, and efficiently deduce $\phi' \in \frac{1}{r}\mathbb{Z}[1, \phi] \cap \mathrm{End}(E)$, with a new $D$ value $D' = D/\gcd(D, N_2)$. Moreover we can evaluate $\phi'$ on the $N_2/\gcd(D, N_2)$ torsion. Following the analysis above, we expect that this will reduce the complexity by a factor $\sqrt{\gcd(D, N_2)}$.

### 3.4 Potential Improvement for Subfield Curves

In this section we take the additional assumption that $E$ is defined over $\mathbb{F}_p$. In this case we can use $R = \mathbb{Z} + \pi_p\mathbb{Z}$ where $\pi_p : (x, y) \to (x^p, y^p)$.

*Remark.* When $E$ is defined over $\mathbb{F}_p$ one can compute the full endomorphism ring in expected time $O(p^{1/4})$ using the techniques of Delfs and Galbraith [4].

*Remark.* Any supersingular elliptic curve can be defined over $\mathbb{F}_{p^2}$, and then it admits the Frobenius endomorphism $\pi_{p^2} : (x, y) \to (x^{p^2}, y^{p^2})$. However this is actually a scalar multiplication so it does not generalize the situation studied in the previous section.

Under the (reasonable) parameter restriction that $N_2 > 2\sqrt{N_1}$ we can compute $\mathrm{Tr}\,\phi$ as above, and substitute $\phi$ by $\phi' = \phi - \frac{\mathrm{Tr}\,\phi}{2}$ in the problem so that $\mathrm{Tr}\,\phi' = 0$. Let $\Delta := \deg \phi' = N_1 - \frac{1}{4}(\mathrm{Tr}\,\phi)^2$. We can consider an endomorphism of the form

$$\psi = (a\phi' + b)\pi_p + c\phi' + d,$$

with degree

$$\begin{aligned}\deg \psi &= (a^2\Delta + b^2)p + (c^2\Delta + d^2) + (a\phi' + b)\pi_p(-c\phi' + d) - (c\phi' + d)\pi_p(-a\phi' + b)\\ &= (a^2\Delta + b^2)p + (c^2\Delta + d^2) + ad(\phi'\pi_p + \pi_p\phi') + bc(-\pi_p\phi' - \phi'\pi_p)\\ &= (a^2\Delta + b^2)p + (c^2\Delta + d^2) + (ad - bc)\,\mathrm{Tr}(\phi'\pi_p).\end{aligned}$$

If $N_2 > 2\sqrt{N_1 p}$ we can evaluate $\mathrm{Tr}(\pi_p\phi')$. We are then left with finding $a, b, c, d, N_1' \in \mathbb{Z}$ such that $\deg \psi = N_1' N_2$ and moreover $N_1'$ is both small and smooth such that the meet-in-the-middle strategy (Lemma 1) is efficient.

Note that for the minimal solution we expect $a^2pN_1 \approx b^2N_1 \approx c^2p \approx d^2 \approx N_1'N_2$ and $abcd \approx N_2$, hence $d^4 \approx N_2N_1p$ and $N_1' \approx N_1^{1/2}p^{1/2}N_2^{-1/2}$. This means that if $N_2 \approx N_1p$ we can expect a solution with $N_1' = O(1)$.

*Remark.* The discussion in this section provides a reduction from an isogeny problem to a Diophantine equation problem, arguably a step forward in the cryptanalysis. We leave the construction of an efficient (classical or quantum) algorithm to solve this Diophantine equation to further work.

*Remark.* Efficient solutions for quaternary quadratic form equations exist over the rationals [3, 15]; however we are not aware of any efficient algorithm that would return integer solutions.

### 3.5 Further Extensions

We could consider variants of Problem 1 where information is given on several endomorphism of a single curve, and develop similar attacks.

## 4 Attacks on the (Generalized) Key Exchange Protocol

We now turn to isogeny problems with additional information, as in De Feo-Jao-Plût's protocols.

### 4.1 Problem Statement

In this section we consider the following problem.

*Problem 2.* Let $p$ be a prime. Let $N_1, N_2 \in \mathbb{Z}$ coprime. Let $E_0$ be a supersingular elliptic curve over $\mathbb{F}_{p^2}$. Let $\phi_1 : E_0 \to E_1$ be an isogeny of degree $N_1$. Let $R_0, R_1$ be subrings of $\mathrm{End}(E_0)$, $\mathrm{End}(E_1)$ respectively. Given $N_1$, $E_1$, $R_0$, $R_1$ and the image of $\phi_1$ on the whole $N_2$ torsion, compute $\phi_1$.

*Remark.* The most generic case for this problem is $R_0 = R_1 = \mathbb{Z}$, namely only the scalar multiplications are known (and do not need to be explicitly given). If $E_0$ is defined over $\mathbb{F}_p$ we can take $R_0 = \mathbb{Z}[\pi_p]$ where $\pi_p$ is the Frobenius. In some previous implementation works [2, 5] it was suggested for efficiency reasons to use special curves in the key exchange protocol, such as a curve with $j$-invariant $j = 1728$. In this case we have $R_0 = \mathrm{End}(E_0)$, and moreover $R_0$ contains some non scalar elements of small degrees.

### 4.2 Attack Model and General Strategy

We provide algorithms that use the additional information provided by the image of torsion points to solve Problem 2 with dramatic speedups compared to the basic meet-in-the-middle strategy.

All our attacks assume that the subring of endomorphisms $R_0$ contains more than the scalar multiplications. They are particularly efficient when special curves $E_0$ are used, such as in [2, 5].

Another current limitation of our attacks is that they require $N_2$ significantly larger than $N_1$. This condition could have plausibly be met in practice (should this paper not have warned against them!) in the following scenarii:

- In the unbalanced variant of the original protocol. We recall that this variant could a priori have been used when one party uses a static key and the other party uses an ephemeral key, as is the case for example in the public key encryption scheme.
- In the optimal degree variant of the protocol, a server may use a static key and publish the images of a very large torsion group $E_0[N_2]$, for example to allow connections with a wide range of clients using different sets of parameters.

Our basic strategy is as follows. For any known endomorphism $\theta \in \mathrm{End}(E_0)$ we can consider the endomorphism $\phi = \phi_1\theta\hat{\phi}_1 \in \mathrm{End}(E_1)$. Moreover if $\theta$ is non scalar then $\phi$ is also non scalar. Using our knowledge of how $\phi_1$ acts on the $N_2$ torsion we can also evaluate $\phi$ on the $N_2$ torsion, and hence apply the techniques from the previous section. Once we have an expression for $\phi$ we can use it to evaluate $\phi_1\theta_0\hat{\phi}_1$ on the $N_1$ torsion. This gives us $\ker\phi_1$ hence $\phi_1$ itself.

*Remark.* This basic strategy requires that $R_0$ contains more than the scalar multiplications, as otherwise $\phi$ is just a scalar multiplication.

Below we give two examples of attacks that can be developed using our techniques.

- The first attack assumes $E_0$ is defined over $\mathbb{F}_p$, and moreover that $E_0$ has a small degree endomorphism $\iota$ such that $\mathrm{Tr}(\iota) = \mathrm{Tr}(\iota\pi_p) = 0$. This is the case for example if $j(E_0) = 1728$. Currently the attack applies only to our optimal degree variant. For well-chosen values of $N_2$ larger than $N_1^4$ the attack recovers the secret key $G_1$ in polynomial time.
- The second attack only requires that $E_0$ has a small degree endomorphism, but on the other hand it needs $\log N_2 = O(\log^2 N_1)$ to recover the secret key $G_1$ in polynomial time. This attack deviates from the basic strategy explained above and it uses some recursive step. We provide a heuristic analysis and some experimental support for this attack for both the unbalanced and the optimal degree variants.

Both attacks are heuristic, as their analysis makes unproven assumptions on factorization properties of certain numbers. We leave a better analysis, further variants and improvements to further work.

### 4.3 Attack when $E_0$ is special

In this section we focus on the optimal degree variant of the protocol. We assume $E_0$ is defined over $\mathbb{F}_p$, so that $\mathrm{End}(E_0)$ contains the Frobenius endomorphism $\pi_p : (x, y) \to (x^p, y^p)$. Moreover we assume $\mathrm{End}(E_0)$ contains some small degree element $\iota$ such that $\mathrm{Tr}(\iota) = \mathrm{Tr}(\iota\pi_p) = 0$. (Maximal orders with minimal such $\iota$ were called special in [11].) Then clearly the attacker knows $\pi_p$ and they can efficiently compute $\iota$ by testing all isogenies of small degree. We consider the endomorphism

$$\psi = \phi_1(a\iota\pi_p + b\pi_p + c\iota)\hat{\phi}_1 + d$$

with norm

$$\deg\psi = N_1^2pqa^2 + N_1^2pb^2 + N_1^2qc^2 + d^2.$$

*Remark.* There is no gain of generality in allowing scalar components in $R_0$: indeed $\phi_1\mathbb{Z}\hat{\phi}_1 = N_1\mathbb{Z} \subset R_1$.

Similarly as before, our goal is now to find tuples of integers $(a, b, c, d)$ such that $\deg\psi = N_1'N_2$ and $N_1'$ is small.

*Choice of* $N_1'$ *and* $N_2$. Heuristically we expect that a proportion $N_2^{-1}$ of the tuples $(a, b, c, d)$ will give a solution modulo $N_2$, so we a priori need $abcd \approx N_2$ or bigger. Moreover, we cannot expect that $N_1'N_2$ is significantly smaller than $N_1^4$, as $d$ is integer and $d \bmod N_1^2$ is determined up to the sign by $N_1'N_2$. Finally, it seems reasonable to expect most solutions $(a, b, c, d)$ to satisfy $N_1^2pqa^2 \approx N_1^2pb^2 \approx N_1^2qc^2 \approx d^2 \approx N_1'N_2$ for reasonably random parameters $(N_1', N_2, N_1)$. In our variant of the protocol one would choose $N_1 \approx p^2$ to ensure that $j(E_1)$ is uniformly distributed in the set of supersingular invariants. In this case taking $N_2 \approx N_1^4$ will a priori ensure a solution $(a, b, c, d)$ with $N_1'$ small.

Clearly the equation $\deg\psi = N_1'N_2$ has integer solutions only if $N_1'N_2$ is a square modulo $N_1^2$. If $N_2$ is a square modulo $N_1^2$ (for example if $N_2$ is a square) then one can choose any small square for $N_1'$ in order to satisfy this condition.

*Remark.* It might be possible to choose $N_2 \approx N_1^3pq$ and $N_1$ small, as long as there exists $d < \sqrt{N_1^3pq}$ with $d^2 = N_1'N_2 \bmod N_1^2$. Note, however, that this would at best only work for very special parameters.

*Algorithm.* Recall that in the optimal degree variant we have $N_1 \approx p^2$. From now on we assume $N_2 \approx N_1^4$ is a square modulo $N_1^2$. Algorithm 2 then computes a tuple $(a, b, c, d)$ to be used in our attack.

*Correctness and analysis.* As $N_2 \approx N_1^4$ is a square modulo $N_1^2$, every small value square of $N_1'$ nearly uniquely determines $d < \sqrt{N_1'N_2}$ with $d \bmod N_1^2$ fixed. The algorithm progressively increases $N_1'$ (taking only square values) and tests all possible corresponding values for $d$ until $(N_1', d)$ are found such that $mq = \frac{N_1'N_2 - d^2}{N_1^2}q$ is a square modulo $p$. We heuristically expect the quadratic residuosity condition to hold with a probability 1/2, so we expect the algorithm to arrive at Step 12 with a very small value of $N_1'$. Steps 13 and 14 will then select a random element in an arithmetic progression in an interval, and asymptotically (and heuristically, for small parameters as well), this will lead to a prime $n$ value with a probability $1/\log n \approx 1/\log(N_1^2/p) \approx 1/3\log(p)$. Moreover the probability that this $n$ can be represented by $a^2 + qb^2$ can be approximated by $1/q$, which expect to be $1/\log p$ (the last point is true under the generalized Riemann hypothesis). Since all the loops are only executed a polynomial number of times, and as all the steps and subroutines are polynomial time, the whole algorithm is polynomial time. Moreover as $N_1'$ will be small the resulting attack on Problem 2 will be polynomial time as well.

*Remark.* In the original and unbalanced variants of the protocol we have $N_1N_2 < p$ so $N_1' > N_1^2p/N_2 > N_1$, unless $a = b = 0$. In the next section we provide an attack that works in this setting.

**Algorithm 2** Finding attack parameters when $E_0$ is special

```
Require: $N_1, N_2, q$ as above.
Ensure: Parameters $(a, b, c, d)$ and $N_1'$ for an attack.
 1: $i \leftarrow 1$.
 2: $N_1' \leftarrow i^2$.
 3: Let $d$ such that $0 \leq d \leq N_1^2$ and $d^2 = N_1' N_2 \bmod N_1^2$.
 4: $m \leftarrow \frac{N_1' N_2 - d^2}{N_1^2}$.
 5: if $mq$ is not a square modulo $p$ then
 6:     if $d < N_1' N_2 - N_1^2$ then
 7:         $d \leftarrow d + N_1^2$.
 8:         go to Step 4.
 9:     else
10:         $i \leftarrow i + 1$.
11:         go to Step 2.
12: Let $\hat{c}$ such that $0 \leq \hat{c} < p$ and $q\hat{c}^2 = m \bmod p$.
13: Let $r$ be a random integer in $[0, m/p]$.
14: $c \leftarrow \hat{c} + rp$.
15: $n \leftarrow \frac{N_1' N_2 - d^2 - c^2 N_1^2 q}{N_1^2 p}$.
16: if $n$ has an easy factorization (for example if $n$ is prime) then
17:     Solve equation $a^2 q + b^2 = n$ with Cornacchia's algorithm
18:     if there is no solution then
19:         go to Step 13.
20: return $(a, b, c, d, N_1')$.
```

### 4.4 Attack when $R_0 = \mathbb{Z} + \theta\mathbb{Z}$ (with $\deg \theta$ small) and $R_1 = \mathbb{Z}$

An algorithm to recover $\psi$ using only the scalar multiplications of $E_1$ and the image of $\psi$ on the $N_2$ torsion was described in Section 3.3. However this in combination with our basic strategy above does not a priori provide any speedup on the straighforward meet-in-the-middle approach. Indeed we have $\deg \psi = N_1^2 \deg \theta \approx N_1^2$ in the most favorable case (when $\deg \theta = 1$) so by the analysis of Section 3.3 we expect to have at best $N_1' \approx \sqrt{D} \approx \sqrt{\deg \psi} \approx N_1$. We therefore modify the basic strategy.

*Modified Strategy.* We adapt the techniques of Section 3.3 to reduce Problem 2 to another instance of itself with smaller parameters $N_1' < N_1/2$ and $N_2'$ some factor of $N_2$. After repeating this reduction step $O(\log N_1)$ times we end up with an instance of Problem 2 where $N_1$ is sufficiently small that it can be solved in polynomial time with a meet-in-the-middle approach.

*Parameter Restriction.* We will require that $\operatorname{End}(E_0)$ has some non scalar element $\theta$ of small degree (which does not need to be explicitly given, as it can then be computed efficiently by trying all isogenies of this degree). This is for example the case in Costello et al.'s implementation [2] where $j = 1728$. In our reduction we will also require $N_2/N_2' > 2N_1 \Delta_\theta$ where $\Delta_\theta = \deg \theta - \frac{1}{4} \operatorname{Tr}^2 \theta$. This implies that we will need to start with parameters such that $\log N_2$ is at least $O(\log^2 N_1)$. Note that in the original De Feo-Jao-Plût protocols we had $N_1 \approx N_2$.

*Reduction Step.* We fix some $\theta \in \operatorname{End}(E_0)$ with small norm $q$, and let $\Delta_\theta := \deg\theta - \frac{1}{4}\operatorname{Tr}^2\theta$. Then we choose some factor $\tilde{N}_2$ of $N_2$ such that $\tilde{N}_2 > KN_1q$ for some $K > 1$, and $-\Delta_\theta$ is a square modulo $\tilde{N}_2$. We proceed as in Section 3.3 to compute $a$, $b$ and $N_1'$ such that $\deg(a\phi_1\theta\hat{\phi}_1 + b) = N_1'\tilde{N}_2$ and $N_1'$ is as small as possible. Namely, we choose $\tau$ such that $\tau^2 = -d \bmod \tilde{N}_2$, then we compute a short vector in a two-dimensional lattice generated by two vectors $(\tilde{N}_2, 0)$ and $(\tau, 1)$ with a weighted norm $||(x,y)|| = (x^2 + Dy^2)^{1/2}$, and we deduce $a$, $b$ and $N_1'$. If $N_1' > N_1/2$ we start again with a new square root of $-D$ modulo $\tilde{N}_2$, or with a new $\tilde{N}_2$ value. If $N_1' < N_1/2$ we define $\phi_{N_1'}, \phi_{\tilde{N}_2}$ two (still unknown) isogenies of degrees $N_1'$ and $\tilde{N}_2$ such that $a\phi_1\theta\hat{\phi}_1 + b = \phi_{N_1'}\phi_{\tilde{N}_2}$. We evaluate $a\phi_1\theta\hat{\phi}_1 + b$ on the $\tilde{N}_2$ torsion to identify the $\tilde{N}_2$ part of the kernel of $a\phi_1\theta\hat{\phi}_1 + b$, then the corresponding isogeny. We evaluate this isogeny on the $N_2' = N_2/\tilde{N}_2$ torsion, and deduce the action of $\phi_{N_1'}$ on the $N_2'$ torsion. We then apply the reduction step recursively to compute some representation of $\phi_{N_1'}$. Finally, we evaluate $(\phi_{N_1'}\phi_{\tilde{N}_2} - b)/a$ on the $N_1$ torsion to compute $\ker\hat{\phi}_1 = \ker\phi_1\theta\hat{\phi}_1 \cup E_1[N_1]$, and from there we compute a more canonical expression for $\phi$.

*Complexity analysis.* Following the analysis of Section 3.3 we expect $N_1' \approx N_1\sqrt{D_\theta}$. However, with some probability on the choice of $\tilde{N}_2$, $N_1'$ will be smaller than $N_1\sqrt{D_\theta}$ by a factor $2\sqrt{D_\theta}$; in fact it seems reasonable to expect this to occur with a probability $(2\sqrt{D_\theta})^{-1}$. All the steps are polynomial time if $N_1$ and $N_2$ are powersmooth and $D_\theta$ is polynomial.

*Remark.* Suppose $p = 3 \bmod 4$ and suppose $j_0 = 1728$ as in Costello et al.'s implementation [2]. In this case there exists a non scalar endomorphism $\iota \in \operatorname{End}(E_0)$ with norm 1 and trace 0. Any $\theta \in \operatorname{End}(E_0)$ must either have a large norm or be of the form $\theta = a\iota + b$ for two small $a, b \in \mathbb{Z}$. In the last case we then have $\Delta_\theta = a^2$, so $-\Delta_\theta$ is a square modulo some prime $r$ if and only if $-1$ is a square modulo $r$. This implies that no prime factor $r$ of $N_2$ with $r = 3 \bmod 4$ can be used in our attack. On the other hand, any prime factors with $r = 1 \bmod 4$ can be used in the attack.

*Experiments for the optimal degree variant.* We wrote a small Magma program [18] to compute the successive pairs of parameters $(a, b)$ to use in our attack, and test the heuristic assumptions involved in our analysis (the code is available in Appendix A). In our experiments we generate random $p$ values, choose $N_1$ powersmooth and then search for a coprime $\tilde{N}_2 > 2qN_1$ leading to $N_1' < N_1/2q$. We repeat this recursively until $N_1'$ is small enough (smaller than some polylog bound in $p$). We used $K = 2$ in these experiments. For 80-bit security parameters our program gives the parameters of an attack in a few seconds. The full attack requires isogenies of degree at most about 36000.

*Experiments for the unbalanced variant.* We also ran attack experiments for the unbalanced protocol variant. In all experiments we took $\ell_1 = 2$ and $\ell_2 = 5$. We considered values of $e_1$ between 20 and 100, and we searched for the minimal value of $e_2$ such that the attack could reduce $N_1$ to a value smaller than 100. Table 1 provides some successful attack parameters. In addition to $e_1$ and $e_2$ it shows the value $\left\lceil \frac{e_2 \log_2 5}{e_1^2} \right\rceil$ (which seems close to a constant 1/2, as expected), the value $K$ used for these parameters, and the number of reduction steps used. Our Magma code is provided in Appendix B.

**Table 1.** Some successful attack parameters against the unbalanced variant ($\ell_1 = 2$ and $\ell_2 = 5$)

| $e_1$ | $e_2$ | $\left\lceil \frac{e_2 \log_2 5}{e_1^2} \right\rfloor$ | $K$ | # steps |
|---|---|---|---|---|
| 20 | 102 | 0.59 | 50 | 11 |
| 30 | 194 | 0.50 | 50 | 17 |
| 40 | 330 | 0.48 | 50 | 22 |
| 50 | 405 | 0.38 | 10 | 30 |
| 60 | 610 | 0.39 | 10 | 38 |
| 70 | 1047 | 0.50 | 2 | 61 |
| 80 | 1473 | 0.53 | 2 | 72 |
| 90 | 1775 | 0.51 | 2 | 80 |
| 100 | 2180 | 0.51 | 2 | 90 |

*Remark.* The parameter $K$ must be larger than 1 in our attack as $N_1' > N_1^2 q/\tilde{N}_2$ for any $a \neq 0$. We experimentally observed that $K = 2$ was sufficient in the optimal degree variant to make $N_1$ decrease by a factor 2 at each reduction step. The unbalanced variant leaves less flexibility in the parameter choice, so we did not impose a factor 2 decrease on $N_1$ (and in fact we even allowed it to increase in some reduction steps). We observed that lower values of $K$ were then sufficient. We have also observed experimentally that the value of $K$ has some moderate impact on the overall performances of the attack (required size for $N_2$, number of reduction steps). We leave a thorough investigation of optimal parameter choices for our attack to further work.

*Remark.* When $N_2$ is too small to execute $O(\log N_1)$ reduction steps, then we may replace the missing last reduction steps by a final meet-in-the-middle strategy. Depending on the final size of $N_1'$ and on its factorization this may still provide some exponential speedup over the basic meet-in-the-middle strategy. We note, however, that for the original parameters proposed by Jao and De Feo, at most one recursive step can be performed. In this case it might be possible to find some (exceptional) set of parameters that would improve the best attack by a few bits, but for most parameters we do not expect any savings.

### 4.5 Possible Extensions

One can vary $R_0$ and $R_1$ depending on the attack model, or consider variants of Problem 2 involving several isogenies, and derive similar attacks. We leave details to the reader and further work.

## 5 Impact and Perspectives

The techniques developed in this paper solve some isogeny problems using the images of certain torsion points by the isogenies. Such images are revealed in De Feo-Jao supersingular key exchange protocol as well as the public key encryption and signature scheme that derive from it (see [5,20] and the first signature scheme of [7]). Until now all existing attacks against these protocols made no use at all of this auxiliary information.

At the moment our techniques do not apply to the parameters originally proposed in these protocols. However they apply on some natural generalizations of them, and they issue a warning that the auxiliary information might weaken isogeny problems. One could also fear that further developments of our techniques and particular attack models will be able to threaten the original protocol itself.

In anticipation of potential future improvements of our attacks, we recommend to avoid the use of special $E_0$ in the protocols, as any (partial) knowledge of the endomorphism ring of $E_0$ may a priori be useful to the attacker with our techniques. We stress, however, that the only known algorithm to avoid special curves for $E_0$ consists in generating a special curve and then performing a random walk from there to obtain a truly random curve; depending on the context this procedure might still allow some form of backdoor attacks. An algorithm that could generate a random supersingular $j$-invariant without performing a random walk from a curve with known endomorphism ring would be a handy tool for designing cryptosystems based on supersingular isogeny problems. Of course, the algorithm may come with additional insight on the underlying Mathematics, which might also help further cryptanalysis. We would like to encourage research in this direction.

We note that the hash function proposed by Charles-Goren-Lauter [1] can also be attacked when starting from a curve with known endomorphism ring. There is also a corresponding "backdoor collision attack"; however the attack is less powerful than above as it can be detected and any use of the backdoor will leak it. These attacks follow from the techniques developed in [11, 12]. On the other hand, the second signature scheme of [7] relies on the endomorphism ring computation problem for random curves, with no extra information leaked, and is not affected by our techniques.

In contrast to the isogeny problem variants considered in this paper, we are not aware of any cryptanalysis result that affects the endomorphism ring computation problem, and we believe that cryptosystems based on this problem offer the strongest security guarantees in the area of isogeny-based cryptography. Of course, cryptanalysis research in this direction is also fairly scarce despite some early work by Kohel [10], and more cryptanalysis will be needed to gain confidence on their security.

## Acknowledgments.

We thank Bryan Birch, Jonathan Bootle, Luca De Feo, Steven Galbraith and Yan Bo Ti for their useful comments on a preliminary version of this paper.

## References

1. Denis Xavier Charles, Kristin E. Lauter, and Eyal Z. Goren. Cryptographic hash functions from expander graphs. *J. Cryptology*, 22(1):93–113, 2009.

2. Craig Costello, Patrick Longa, and Michael Naehrig. Efficient algorithms for supersingular isogeny diffie-hellman. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, pages 572–601, 2016.
3. J. E. Cremona and D. Rusin. Efficient solution of rational conics. *Math. Comput.*, 72(243):1417–1441, 2003.
4. Christina Delfs and Steven D. Galbraith. Computing isogenies between supersingular elliptic curves over Ʊp. *Des. Codes Cryptography*, 78(2):425–440, 2016.
5. Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *J. Mathematical Cryptology*, 8(3):209–247, 2014.
6. Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. Cryptology ePrint Archive, Report 2016/859, 2016. `http://eprint.iacr.org/2016/859`.
7. Steven D. Galbraith, Christophe Petit, and Javier Silva. Signature schemes based on supersingular isogeny problems. Cryptology ePrint Archive, Report 2016/1154, 2016. `http://eprint.iacr.org/2016/1154`.
8. Alexandre Gélin and Benjamin Wesolowski. Loop-abort faults on supersingular isogeny cryptosystems. Cryptology ePrint Archive, Report 2017/374, 2017. `http://eprint.iacr.org/2017/374`.
9. David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *PQCrypto*, pages 19–34, 2011.
10. David Kohel. *Endomorphism rings of elliptic curves over finite fields*. PhD thesis, University of California, Berkeley, 1996.
11. David Kohel, Kristin Lauter, Christophe Petit, and Jean-Pierre Tignol. On the quaternion $\ell$-isogeny path problem. *LMS Journal of Computation and Mathematics*, 17A:418–432, 2014.
12. Christophe Petit. On the quaternion $\ell$-isogeny problem. Presentation slides from a talk at the University of Neuchâtel, March 2015.
13. Alexander Rostovtsev and Anton Stolbunov. Public-key cryptosystem based on isogenies. Cryptology ePrint Archive, Report 2006/145, 2006. `http://eprint.iacr.org/`.
14. Joseph Silverman. *The Arithmetic of Elliptic Curves*. Springer Verlag, 1986.
15. Denis Simon. Quadratic equations in dimensions 4, 5 and more. Preprint, `http://www.math.unicaen.fr/~simon/`, 2005.
16. Yan Bo Ti. Fault attack on supersingular isogeny cryptosystems. Cryptology ePrint Archive, Report 2017/379, 2017. `http://eprint.iacr.org/2017/379`.
17. Marie-France Vignéras. *The arithmetic of quaternion Algebra*. 2006.
18. C. Fieker A. Steel (eds.) W. Bosma, J. J. Cannon. Handbook of Magma functions, edition 2.20. `http://http://magma.maths.usyd.edu.au/magma/`, 2013.
19. Sun Xi, Haibo Tian, and Yumin Wang. Toward quantum-resistant strong designated verifier signature from isogenies. *International Journal of Grid and Utility Computing*, 5(2):292–296, September 2012.
20. Youngho Yoo, Reza Azarderakhsh, Amir Jalali, David Jao, and Vladimir Soukharev. A post-quantum digital signature scheme based on supersingular isogenies. Financial Crypto, 2017.

## A Magma code for the Experiments of Section 4.4 against the optimal degree variant

```
// security parameter
lambda:=80;
"lambda=",lambda;
```

```
// arbitrary constant such that all prime powers of Ni are smaller than (\log p)
PowBound:=3;

// impose parameters such that q=1
QEQ1:=false;

// N2t> K*2*N1*q, by default K=1
K:=1;


// choose p and deduce q
p:=NextPrime(Random(2^(2*lambda))) ;

if QEQ1 then
while (p mod 4) ne 3 do
p:=NextPrime(Random(2^(2*lambda))) ;
end while;
end if;


Fp:=FiniteField(p);

if (p mod 4) eq 3 then
q:=1;
else // find q=3 mod 4 and with -q not square mod p
q:=3;
while IsSquare(-Fp!q) or (not IsPrime(q)) do
q:=q+4;
end while;
end if;

"p=",p;
"q=",q;

B:=Floor((Log(2,p))^PowBound);
"bound on prime power is",B;


// choose N1
N1:=1;
qi:=2;
while N1 lt p^2 do
qi:=NextPrime(qi);
ei:=Min(Floor(Log(qi,B)),Ceiling(Log(qi,p^2/N1)));
```

```
N1:=N1*qi^ei;
end while;
"N1=",N1;
"N1=",Factorization(N1);
"N1>p^2?", N1 gt p^2;
```

```
// recursive attack
// p and q define quaternion algebra
// qi is largest prime previously used
// N1 is current value of N1
// N2 is torsion needed so far
function RecursiveAttack(p,q,qi,N1,N2)
N1d:=N1;
D:=(N1^2)*q;
N1dSmall:=false;

while not N1dSmall do
// generate some tilde N2
// CHANGE: suboptimal method for now: if one value of tilde N2 does not work, it
"\nconstructing a new N2tilde value";
N2t:=1;
N2tfac:=[];
while N2t lt K*2*N1*q do
qi:=NextPrime(qi);

// check that -q is a square mod qi
while not IsSquare(-FiniteField(qi)!q) do
qi:=NextPrime(qi);
end while;

ei:=Min(Floor(Log(qi,B)),Ceiling(Log(qi,K*2*N1*q/N2t)));
N2t:=N2t*qi^ei;
N2tfac:=N2tfac cat [qi^ei] ;
end while;
"N2 tilde factors are",Factorization(N2t);

// compute sqrt of -q modulo all prime power divisor of N2t
vecTau:=[Integers()!(Sqrt(-IntegerRing(fac)!D)) : fac in N2tfac] ;

// build lattice for each square root of -qN1^2
indexsqrt:=0;
```

```
while (not N1dSmall) and (indexsqrt lt 2^#N2tfac) do
// find a square root of -D mod tilde N2
indexsqrt:=indexsqrt+1;
//signs:=Eltseq([f:f in FiniteField(2^#N2tfac)][indexsqrt]);
signs:=Eltseq(Random(FiniteField(2^#N2tfac)));
tau:=CRT([(-1)^(Integers()!signs[ind])*vecTau[ind]: ind in [1..#N2tfac]] , N2tfa
//"tau=",tau;
//"tau^2+D mod N2", (tau^2+D) mod N2;

// build lattice
L:=Lattice(Matrix(Rationals(),2,2,[N2t,0,tau,1]), Matrix(Rationals(),2,2,[1,0,0,

// compute short vectors
shortest:=LLLBasisMatrix(L);
if shortest[1][2] eq 0 then
"shortest vector scalar";
x:=shortest[2][1];
y:=shortest[2][2];
else
x:=shortest[1][1];
y:=shortest[1][2];
end if;

// compute N1d
N1d:=Integers()!( (x^2+D*y^2) / N2t );
"N1d/N1=",RealField()!(N1d/N1);

// test condition on N1d
N1dSmall:=(N1d lt N1/2);
//N1dSmall:=(N1d lt N1/(2*q));
//"N1d is small?",N1dSmall;

end while;
end while;



// recursive call
listN1:=[];
if N1d gt B then
"\n\nCalling the attack recursively";
listN1, N2:=RecursiveAttack(p,q,qi,N1d,N2*N2t);
end if;

// return
```

```
return [N1] cat listN1 , N2 ;
end function;



// call attack
N2:=1;
listN1,N2 := RecursiveAttack(p,q,qi,N1,N2) ;
"listN1 values",listN1;
"factorization of N2",Factorization(N2);
"bitsize N1 is", Log(2,N1);
"bitsize N2 is", Log(2,N2);
```

## B Magma code for the Experiments of Section 4.4 against the unbalanced variant

```
// small primes
p1:=2;
p2:=5;

// exponents
e1:=Ceiling(100*Log(p1,2));
e2:=Ceiling(5060*Log(p2,2));
"e2=",e2;
N1:=p1^e1;
N2:=p2^e2;
"log(N2)/log(N1)=", Log(2,N2)/ Log(2,N1) ;

"bitsize N1 at the beginning is", Ceiling(Log(2,N1));



// N2t> K*N1*q, by default K=1
K:=2;

// value of N1 where we stop recursion
N1_MIN:=100;

"bitsize N2 is",Ceiling(Log(2,N2));
"K=",K;
"N1_MIN=",N1_MIN;
"\n";
```

```
// find p
f:=1;
p:=p1^e1*p2^e2-1;
while(not IsPrime(p)) do
f:=f+1;
p:=p1^e1*p2^e2*f-1;
end while;

Fp:=FiniteField(p);


//find q
if (p mod 4) eq 3 then
q:=1;
else // find q=3 mod 4 and with -q not square mod p
q:=3;
while IsSquare(-Fp!q) or (not IsPrime(q)) do
q:=q+4;
end while;
end if;

"p=",p;
"q=",q;




N1d:=N1;
D:=(N1^2)*q;
e2left:=e2;
nbrec:=0;


while( (Ceiling(Log(p2,K*N1*q)) le e2left) and (N1 gt N1_MIN)) do
nbrec:=nbrec+1;

// choose N2t and update e2left
e2t:=Ceiling(Log(p2,K*N1*q));
N2t:=p2^e2t;
"N2t=",N2t;
e2left:=e2left-e2t;

// compute sqrt of -q modulo N2t
```

```
tau:=Integers()!(Sqrt(-IntegerRing(N2t)!D));

// build lattice
L:=Lattice(Matrix(Rationals(),2,2,[N2t,0,tau,1]), Matrix(Rationals(),2,2,[1,0,0,

// compute shortest vector
shortest:=LLLBasisMatrix(L);
if shortest[1][2] eq 0 then
"shortest vector scalar";
x:=shortest[2][1];
y:=shortest[2][2];
else
x:=shortest[1][1];
y:=shortest[1][2];
end if;

// compute N1d
N1d:=Integers()!( (x^2+D*y^2) / N2t );
"N1d/N1=",RealField()!(N1d/N1);

// update N1 and D
N1:=N1d;
D:=(N1d^2)*q;
end while;

"\n";
if N1 le N1_MIN then "Attack succesful"; end if;
"bitsize N1 is", Log(2,N1);
"bitsize N2 used is", Ceiling((e2-e2left)*Log(2,5));
"number of recursion steps is",nbrec;
"factorization(N1)=",Factorization(N1);
```