Efficient Computation of $(3^n, 3^n)$-Isogenies

Thomas Decru, Sabrina Kunzweiler

Efficient Computation of $(3^n, 3^n)$-Isogenies

Thomas Decru[1] and Sabrina Kunzweiler[2]

[1] imec-COSIC, KU Leuven, Belgium
thomas.decru@kuleuven.be
[2] Univ. Bordeaux, CNRS, Bordeaux INP, Inria, France
sabrina.kunzweiler@math.u-bordeaux.fr

Abstract. The parametrization of $(3,3)$-isogenies by Bruin, Flynn and Testa requires over 37.500 multiplications if one wants to evaluate a single isogeny in a point. We simplify their formulae and reduce the amount of required multiplications by 94%. Further we deduce explicit formulae for evaluating $(3,3)$-splitting and gluing maps in the framework of the parametrization by Bröker, Howe, Lauter and Stevenhagen. We provide implementations to compute $(3^n, 3^n)$-isogenies between principally polarized abelian surfaces with a focus on cryptographic application. Our implementation can retrieve Alice's secret isogeny in 11 seconds for the SIKEp751 parameters, which were aimed at NIST level 5 security.

1 Introduction

Elliptic curves have a rich history of being used for cryptographic purposes. Their higher-dimensional variants have also been studied in the context of the discrete logarithm problem, but were deemed not practical or safe enough to be used (see for example [16,27]). With the advent of quantum computers in mind, a lot of this research has shifted towards using isogenies between elliptic curves.

In 2009, Charles, Goren and Lauter (CGL) used isogenies between supersingular elliptic curves over $\mathbb{F}_{p^2}$ to construct a hash function based on their expander graph properties [10]. In 2018, Takashima generalized this construction to supersingular Jacobians of hyperelliptic curves of genus two [28], but this hash function was quickly found to allow many collisions by Flynn and Ti [14]. These collisions were fixed by Castryck, Decru and Smith, and they also argued for the correct generalization to superspecial abelian varieties [8].

In 2011, Jao and De Feo constructed a Diffie–Hellman style key exchange, called Supersingular Isogeny Diffe–Hellman (SIDH), based on the isogeny graph underlying the CGL hash function [17]. The protocol can also be generalized to allow a key exchange when using abelian surfaces instead of elliptic curves, as shown by Flynn and Ti [14]. This higher-dimensional variant was further improved in a follow-up work by Kunzweiler, Ti and Weitkämper [20].

The SIDH protocol was used as the basis for the Supersingular Isogeny Key Encapsulation (SIKE) which was submitted to NIST as a candidate for their post-quantum standardization process. Early July 2022, NIST announced SIKE to be one of only four candidates to advance to round 4 of the post-quantum

standardization process for public key exchanges [23]. That same month however, Castryck and Decru published a devastating attack on SIKE, retrieving Bob's private key in minutes to hours depending on the security level [7]. Their attack relied on embedding elliptic curves into abelian surfaces, and used Kani's reducibility criterion [18] as part of their decisional oracle. The attack got improved by a quick series of follow-up works using a direct computational approach [22,24], and finally Robert managed to prove that even if the endomorphism ring of the starting curve in SIDH is unknown, there is always a polynomial-time attack by using abelian eightfolds [25].

Despite these generalizations and the increasing interest in higher-dimensional cryptographic applications, most of the aforementioned implementations restrict themselves to isogenies of very low prime degree. The genus-2 version of the CGL hash function in [8] used $(2,2)$-isogenies only, since they are by far easiest to compute. Kunzweiler improved further on these $(2,2)$-isogeny formulae in [19], and Castyck and Decru provided a $(3,3)$-version based on their multiradical isogeny setting [6]. The (now also broken) genus-2 variant of SIDH in [14] used $(2,2)$- and $(3,3)$-isogenies to obtain a five-minute key exchange on the basic security level. The implementation of the attacks on SIKE in [7] and [22] only target Bob's private key, since this requires only using $(2,2)$-isogenies. In [26], Santos, Costello and Frengley do manage to use up to $(11,11)$-isogenies, but only as a decisional tool to detect (N,N)-split Jacobians.

The reason for these restrictions is that computing isogenies between abelian surfaces is typically a lot harder than isogenies between elliptic curves. The general (ℓ,ℓ)-isogeny formulae by Cosset and Robert [12] have polynomial time complexity $\mathcal{O}(\ell^2)$ or $\mathcal{O}(\ell^4)$, depending on $\ell \bmod 4$, but arithmetic has to be performed in the field extension where the theta coordinates are defined, which can turn expensive quickly for cryptographic purposes. The only other known general parametrization are the $(3,3)$-isogeny formulae by Bruin, Flynn and Testa (BFT) [4], which were used as a basis for both the multiradical $(3,3)$-hash function and the genus-2 variant of SIDH. The parametrization is complete, but the formulae require over 37.500 multiplications if one wants to also evaluate points and not just compute the codomain curve.

Our contribution. We optimize the BFT-formulae from [4] and reduce the amount of required multiplications by 94%. We also develop concrete and efficient gluing and splitting formulae for $(3,3)$-isogenies, which allow us to evaluate them on points. All of these operations are furthermore done over the ground field. Our implementations and formulae are with cryptographic applications in mind and may not work for some small field characteristics. Additionally, certain exceptional cases occur with probability $\mathcal{O}(p^{-1})$ or less, in which case we do not implement them in generic applications to reduce the overhead. Exceptions to this which are useful for cryptographic purposes - such as the gluing and splitting - are of course exempt from this exclusion. We provide a $(3,3)$-variant of the CGL hash function similar to the one from [6], and implement an attack targeting Alice's secret isogeny in the SIKE protocol. The latter can be done in

11 seconds for the SIKEp751 parameters, aimed at NIST level 5 security, down from the 1 hour computation for Bob's secret isogeny in [24].

Outline. We will provide necessary mathematical preliminaries in Section 2. In Section 3, we will recap the BFT parametrization and discuss our improvements to it. In Section 4 we will discuss isogenies between abelian surfaces of which at least one of the domain or codomain is a product of elliptic curves, followed by the necessary coordinate transformations between these parametrizations in Section 5. Our version of the $(3,3)$-hash function and attack on Alice's private key in the SIKE protocol will be discussed in Section 6, respectively 7. Finally, we will provide an overview of the auxiliary Magma [2] and SageMath [30] code in Section 8, which can be found at `https://github.com/KULeuven-COSIC/3_3_isogenies`.

Acknowledgements. This work is a result of a workshop during Leuven Isogeny Days 2022, supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement ISOCRYPT - No. 101020788). This work was supported in part by CyberSecurity Research Flanders with reference number VR20192203, the DFG under Germany's Excellence Strategy - EXC 2092 CASA - 390781972 and the Agence Nationale de la Recherche under the grant ANR CIAO (ANR-19-CE48-0008). We thank Anna Somoza and Eda Kırımlı for helpful discussions.

2 Preliminaries

Below are some notes on the definitions that we will need later. In general, we assume to work over a field k with nonnegative characteristic $p > 5$, though some of the results generalize beyond this restriction.

2.1 Genus-2 Curves and their Jacobians

Let $\mathcal{C}$ be an algebraic curve of genus 2. Any such curve is hyperelliptic and admits an affine equation of the form $\mathcal{C} : y^2 = f(x)$, where $f \in k[x]$ is a square-free polynomial of degree 5 or 6. If f is a degree-5 polynomial, then the corresponding genus-2 curve has precisely one point at infinity which we denote by ∞. On the other hand, if the degree of f is 6, then there are two points at infinity and we denote them by ∞_+ and ∞_-. Note that ∞_+ and ∞_- get swapped by the hyperelliptic involution $\tau : \mathcal{C} \to \mathcal{C}, (x, y) \mapsto (x, -y)$, whereas in the degree-5 case, ∞ is a fixed point. In general, points fixed by the involution are referred to as the *Weierstrass points* of $\mathcal{C}$.

While the points on a genus-2 curve do *not* form a group, we will work with Jacobians of such curves. The Jacobian $\mathrm{Jac}(\mathcal{C})$ of a genus-2 curve $\mathcal{C}$ is an abelian surface, i.e. an abelian variety of dimension 2. Moreover it comes equipped with a principal polarization, i.e. an isomorphism to its dual and is therefore considered

a *principally polarized abelian surface (p.p.a.s.)*. In general, p.p.a.s. come in two flavours. They are either irreducible and hence the Jacobian of a genus-2 curve, or they are reducible in which case they are the product of two elliptic curves.

To work with elements of the Jacobian $\mathrm{Jac}(\mathcal{C})$, one usually exploits its link to the Picard group of $\mathcal{C}$. Recall that for any field extension k'/k, the group of k'-rational points $\mathrm{Jac}(\mathcal{C})(k')$ is isomorphic to the Picard group $Pic^0_{\mathcal{C}}(k')$. This allows us to represent elements of $\mathrm{Jac}(\mathcal{C})$ as equivalence classes of degree-0 divisors on $\mathcal{C}$. Moreover, any element $[D] \in \mathrm{Jac}(\mathcal{C})$ has a unique representative of the form $[P_1 + P_2 - D_\infty]$, where

$$D_\infty = \begin{cases} 2 \cdot \infty & \text{if } \deg(f) = 5, \\ \infty_+ + \infty_- & \text{if } \deg(f) = 6, \end{cases}$$

and $P_1 + P_2$ is an effective divisor with affine part in general position, i.e. $P_1 \neq \tau(P_2)$, see [15, Proposition 1]. This facilitates a compact representation in terms of Mumford coordinates. For simplicity, assume that $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ are both affine, then the Mumford presentation of $[D]$ is defined as the pair of polynomials $[a, b] \in k[x]^2$ with $a = (x - x_1)(x - x_2)$ and $y = b(x)$ is the line connecting P_1 and P_2. For the general definition, we refer to [11].

2.2 Torsion Subgroups and Isogenies of p.p.a.s.

For an integer $N \in \mathbb{N}$, the N-torsion subgroup of a p.p.a.s. is defined as $\mathcal{A}[N] = \{P \in \mathcal{A} \mid N \cdot P = 0\}$. If N is not divisible by p, then this is a free $\mathbb{Z}/N\mathbb{Z}$-module of rank 4. To describe subgroups defining isogenies between p.p.a.s., it is necessary to take into account the *Weil pairing* which is an alternating, bilinear pairing $e_N : \mathcal{A}[N] \times \mathcal{A}[N] \to \mu_N$, where μ_N denotes the group of N-th roots of unity. A subgroup $G \subset \mathcal{A}[N]$ is called *maximal N-isotropic* if the following two properties are satisfied.

1. The Weil pairing restricts trivially onto G (isotropy).
2. There is no proper subgroup $H \subset \mathcal{A}[N]$ properly containing G (maximality).

Let $G \subset \mathcal{A}[N]$ be a maximal N-isotropic subgroup, then up to isomorphism there exists a unique p.p.a.s. $\mathcal{A}'$ together with an isogeny $\Phi : \mathcal{A} \to \mathcal{A}'$ with kernel $\ker(\Phi) = G$. In our paper, we always consider kernel groups of rank 2, i.e. $G \cong \mathbb{Z}/N\mathbb{Z} \times \mathbb{Z}/N\mathbb{Z}$. In this case, we simply refer to the kernel as an (N, N)-*subgroup* and call the corresponding isogeny an (N, N)-*isogeny*. Indeed, we will only be interested in the case where $N = 3$ or more generally $N = 3^n$.

Since there are two types of p.p.a.s., Jacobians of genus-2 curves and products of elliptic curves, there exist four different types of isogenies of p.p.a.s depending on the nature of the domain and codomain. We distinguish the following cases.

- Generic case: $\Phi : \mathrm{Jac}(\mathcal{C}) \to \mathrm{Jac}(\mathcal{C}')$.
- Splitting case: $\Phi : \mathrm{Jac}(\mathcal{C}) \to E_1' \times E_2'$.
- Gluing case: $\Phi : E_1 \times E_2 \to \mathrm{Jac}(\mathcal{C}')$.
- Product case: $\Phi : E_1 \times E_2 \to E_1' \times E_2'$.

In higher-dimensional isogeny-based cryptography, one almost always works with superspecial abelian varieties, see for example [8]. Given that the superspecial products of elliptic curves only constitute a proportion of $\mathcal{O}(p^{-1})$ of all superspecial p.p.a.s., the generic case occurs most often in cryptographic contexts.

2.3 The Quartic Model of the Kummer Surface

Instead of working with the Jacobian of a genus-2 curve, it is sometimes favourable to work with the associated Kummer surface obtained by taking the quotient by the action of $[-1]$ on $\mathrm{Jac}(\mathcal{C})$. While this results in losing the full picture of the group, it has the geometric advantage that the Kummer surface can be compactly defined as a variety in $\mathbb{P}^3$. In general, the Kummer surface can be seen as the natural analogue to x-only arithmetic often used in elliptic curve cryptography. We now provide some more details on the definition.

Let $\mathcal{C} : y^2 = F(x)$ with $F = \sum_{i=0}^{6} f_i x^i$ be a curve of genus 2 and let $\mathrm{Jac}(\mathcal{C})$ be its Jacobian. Consider the map $\xi : \mathrm{Jac}(\mathcal{C}) \to \mathbb{P}^3$, generically defined as

$$[(x_1, y_1) + (x_2, y_2) - D_\infty] \mapsto (\xi_0 : \xi_1 : \xi_2 : \xi_3),$$

where

$$\xi_0 = 1, \quad \xi_1 = x_1 + x_2, \quad \xi_2 = x_1 x_2, \quad \xi_3 = \frac{\varphi(\xi_0, \xi_1, \xi_2) - 2y_1 y_2}{\xi_1^2 - 4\xi_0 \xi_2}$$

and

$$\varphi = 2f_0\xi_0^3 + f_1\xi_0^2\xi_1 + 2f_2\xi_0^2\xi_2 + f_3\xi_0\xi_1\xi_2 + 2f_4\xi_0\xi_2^2 + f_5\xi_1\xi_2^2 + 2f_6\xi_2^3.$$

The image of the map ξ in $\mathbb{P}^3$ is a quartic surface defined by the equation

$$K(\xi_0, \xi_1, \xi_2, \xi_3) = (\xi_1^2 - 4\xi_0\xi_2)\xi_3^2 + \varphi(\xi_0, \xi_1, \xi_2)\xi_3 + \eta(\xi_0, \xi_1, \xi_2) = 0,$$

where $\eta \in k[\xi_0, \xi_1, \xi_2]$ is a homogeneous degree-4 polynomial. The image of ξ is called the *Kummer surface* of $\mathcal{C}$ and we denote it as $\mathcal{K}(\mathcal{C})$. While we omit the formulae for η here, we remark that its coefficients lie in $\mathbb{Z}[f_0, \ldots, f_6]$. For an explicit description of η and more details on the definition of the Kummer surface, we refer to [5, Chapter 3].

Remnants of the group structure Applying the map $\xi : \mathrm{Jac}(\mathcal{C}) \to \mathcal{K}(\mathcal{C})$, the group structure gets lost, in particular the Kummer surface is *not* an abelian variety. For instance, consider two elements $T, T' \in \mathrm{Jac}(\mathcal{C})$. Given only $\xi(T)$ and $\xi(T')$, it is not possible to determine $\xi(T + T')$. However, there are some remnants of the group structure of the Jacobian. In particular, multiplication by an integer $n \in \mathbb{Z}$ remains meaningful on the Kummer surface. That is, given $\xi(T)$ for some element $T \in \mathrm{Jac}(\mathcal{C})$, one can compute $n \cdot \xi(T) \coloneqq \xi(nT)$ (see [5, Chapter 3.6]). Furthermore, one can use differential additions to compute images of points on the Kummer surface that lift to a specific element of the Jacobian. Since our applications do not require this, we shall not elaborate.

Isogenies An isogeny $\Phi : \mathrm{Jac}(\mathcal{C}) \to \mathrm{Jac}(\mathcal{C}')$ descends to a rational map $\Phi_{\mathcal{K}} : \mathcal{K}(\mathcal{C}) \to \mathcal{K}(\mathcal{C}')$ which makes the following square commute.

$$\begin{array}{ccc} \mathrm{Jac}(\mathcal{C}) & \xrightarrow{\Phi} & \mathrm{Jac}(\mathcal{C}') \\ \downarrow{\scriptstyle \xi} & & \downarrow{\scriptstyle \xi'} \\ \mathcal{K}(\mathcal{C}) & \overset{\Phi_{\mathcal{K}}}{\dashrightarrow} & \mathcal{K}(\mathcal{C}'). \end{array}$$

Being a map of Kummer surfaces, $\Phi_{\mathcal{K}}$ is strictly speaking not an isogeny. However, we slightly abuse notation and refer to $\Phi_{\mathcal{K}}$ as the isogeny on the level of Kummer surfaces.

3 (3, 3)-Isogenies between Jacobians

In the first part of this section, we summarize the parametrization of genus-2 curves whose Jacobians have a $(3,3)$-torsion subgroup with rational generators, as well as the corresponding isogeny formulae by Bruin, Flynn and Testa from [4]. In the second part, we explain optimizations for the evaluation of these formulae.

3.1 BFT Approach

Consider a genus-2 curve $\mathcal{C}$ and a maximal isotropic group $G = \langle T, T' \rangle \subset \mathrm{Jac}(\mathcal{C})[3](k)$. In [4], the authors show that if the data $(\mathcal{C}, G)$ is sufficiently general, then there exist $r, s, t \in k$ such that $\mathcal{C}$ is isomorphic to the curve

$$\mathcal{C}_{r,s,t} : y^2 = G_{r,s,t}(x)^2 + \lambda H_{r,s,t}(x)^3 = G'_{r,s,t}(x)^2 + \lambda' H'_{r,s,t}(x)^3,$$

where

$$\begin{aligned} H_{r,s,t}(x) &= x^2 + rx + t, \qquad \lambda = 4s, \\ G_{r,s,t}(x) &= (s - st - 1)x^3 + 3s(r-t)x^2 + 3sr(r-t)x - st^2 + sr^3 + t, \\ H'_{r,s,t}(x) &= x^2 + x + r, \qquad \lambda' = 4st, \\ G'_{r,s,t}(x) &= (s - st + 1)x^3 + 3s(r-t)x^2 + 3sr(r-t)x - st^2 + sr^3 - t. \end{aligned}$$

The 3-torsion elements T and T' are given by $[H_{r,s,t}(x), G_{r,s,t}(x)]$ respectively $[H'_{r,s,t}(x), G'_{r,s,t}(x)]$.[1] For an explicit description of the full torsion subgroup $\langle T, T' \rangle$, see [4, Theorem 6].

Remark 1. The generality stems from the fact that both T and T' require non-degenerate support. However, due to [4, Lemma 3], it is always possible to choose two generators of $\langle T, T' \rangle$ that satisfy this. The chance of two random generators having degenerate support is $\mathcal{O}(p^{-1})$, so we will not elaborate on those cases.

[1] Remark the slight abuse of notation: the polynomials $G_{r,s,t}(x)$ and $G'_{r,s,t}(x)$ are cubic and hence this is not a Mumford representation. However, they reduce to the correct linear expression modulo $H_{r,s,t}(x)$, respectively $H'_{r,s,t}(x)$.

Remark 2. When considering the isomorphism $(x, y) \mapsto (x, y + G_{r,s,t}(x))$, the parametrization gets a cleaner form which is similar to the well-known parametrization of $X_1(3)$ in the elliptic-curve case. Indeed, the curve is then given by

$$\mathcal{C} : y^2 + G_{r,s,t}(x)y = sH_{r,s,t}(x)^3.$$

The $(3, 3)$-subgroup is then generated by $[H_{r,s,t}(x), 0]$ and $[H'_{r,s,t}(x), x^3 - t]$.

We consider the $(3, 3)$-isogeny $\Phi : \mathrm{Jac}(\mathcal{C}_{r,s,t}) \to \mathrm{Jac}(\mathcal{C}_{r,s,t})/\langle T, T' \rangle$, where we assume $\mathrm{Jac}(\mathcal{C}_{r,s,t})/\langle T, T' \rangle = \mathrm{Jac}(\widetilde{\mathcal{C}})$ is again the Jacobian of a genus-2 curve. Along with the formulae for the codomain curve, the authors of [4] also provide explicit formulae for the induced map $\Phi_{\mathcal{K}} : \mathcal{K}(\mathcal{C}_{r,s,t}) \to \mathcal{K}(\widetilde{\mathcal{C}})$ on the corresponding Kummer surface. [2] Naturally, the map on the Kummer surfaces is of degree three. More precisely, it is of the form

$$\begin{aligned} \Phi_{\mathcal{K}} : \mathcal{K}(\mathcal{C}_{r,s,t}) &\to \mathcal{K}(\widetilde{\mathcal{C}}) \\ (\xi_0 : \xi_1 : \xi_2 : \xi_3) &\mapsto (\widetilde{\xi_0} : \widetilde{\xi_1} : \widetilde{\xi_2} : \widetilde{\xi_3}), \end{aligned}$$

with

$$\begin{aligned} \widetilde{\xi_0} &= \sum_{0 \le i \le j \le k \le 3} a_{i,j,k}\xi_i\xi_j\xi_k, \qquad \widetilde{\xi_1} = \sum_{0 \le i \le j \le k \le 3} b_{i,j,k}\xi_i\xi_j\xi_k, \\ \widetilde{\xi_2} &= \sum_{0 \le i \le j \le k \le 3} c_{i,j,k}\xi_i\xi_j\xi_k, \qquad \widetilde{\xi_3} = \sum_{0 \le i \le j \le k \le 3} d_{i,j,k}\xi_i\xi_j\xi_k. \end{aligned} \tag{1}$$

Note that there are exactly 20 monomials of degree 3 in four variables. There exist expressions $a_{0,0,0}, \ldots, d_{3,3,3} \in \mathbb{Z}[r, s, t]$ for the 80 coefficients. Unfortunately, these expressions are not very compact and their evaluation requires over 37.500 multiplications using a multivariate Horner scheme.

3.2 Improvements

To reduce the number of multiplications in the evaluation of the $(3, 3)$-isogeny, we find more compact representations for the coefficients $a_{i,j,k}, b_{i,j,k}, c_{i,j,k}, d_{i,j,k}$ introduced in Equation 1.

Relations among the coefficients As a first step, we observe that there exist various relations among the $a_{i,j,k}, b_{i,j,k}, c_{i,j,k}, d_{i,j,k}$ from above and the coefficients of the curve equations from $\mathcal{C}_{r,s,t}$ and $\widetilde{\mathcal{C}}$. To make this more explicit, denote $\mathcal{C}_{r,s,t} : y^2 = \sum f_i x^i$ and $\widetilde{\mathcal{C}} : y^2 = \sum g_i x^i$ keeping in mind that we know explicit descriptions of the coefficients from Section 3.1. The coefficients of the Kummer surface equations $K_{r,s,t}$ of $\mathcal{K}(\mathcal{C}_{r,s,t})$ can be expressed in terms of $f_0, \ldots, f_6$, and in the same way the coefficients of the equation $\widetilde{K}$ of $K(\widetilde{\mathcal{C}})$ can be expressed in terms of $g_0, \ldots, g_6$. In the following, we interpret the polynomial $\widetilde{K} \in k[\widetilde{\xi_0}, \widetilde{\xi_1}, \widetilde{\xi_2}, \widetilde{\xi_3}]$ as a polynomial in $k[\xi_0, \xi_1, \xi_2, \xi_3]$ via the identities for $\widetilde{\xi_i}$ from

[2] They can be found online at `http://www.cecm.sfu.ca/~nbruin/c3xc3/`

Equation 1. In that setting, $\widetilde{K}$ is a degree-12 polynomial. Further note that $\widetilde{K}$ vanishes at all points $(\xi_0 : \xi_1 : \xi_2 : \xi_3)$ of $\mathcal{K}(\mathcal{C}_{r,s,t})$, hence $\widetilde{K}$ is divisible by $K_{r,s,t}$ and we can write

$$\widetilde{K} = Q_{\text{aux}} \cdot K_{r,s,t} \in k[\xi_0, \xi_1, \xi_2, \xi_3] \tag{2}$$

for a degree-8 polynomial Q_{aux} in $k[\xi_0, \xi_1, \xi_2, \xi_3]$.

While there exist known expressions for $a_{i,j,k}, b_{i,j,k}, c_{i,j,k}, d_{i,j,k}$ we treat them as variables. The only exceptions are $a_{i,3,3}, b_{i,3,3}, c_{i,3,3}, d_{i,3,3}$ with $i \in \{0, 1, 2, 3\}$, for which we insert the already known (and compact) expressions. In fact, these are either 0 or Δ, where the latter is a factor of the discriminant of $\widetilde{\mathcal{C}}$. By comparing coefficients of the identity in Equation 2, we obtain in total 447 relations among the $a_{i,j,k}, b_{i,j,k}, c_{i,j,k}, d_{i,j,k}$ and the f_i, g_i. Note that these relations also include the 165 coefficients $q_{i_1,\ldots,i_8}$ of the degree-8 polynomial Q_{aux}. However, it is easy to eliminate these unknowns from the system which reduces the number of relations by 165 and leaves us with 282 relations purely between $a_{i,j,k}, b_{i,j,k}, c_{i,j,k}, d_{i,j,k}$ and f_i, g_i. If $K_{r,s,t}$ and $\widetilde{K}$ were general degree-4 equations, (almost) all the obtained relations would be quartic in $a_{i,j,k}, b_{i,j,k}, c_{i,j,k}, d_{i,j,k}$. However, given the special form of the Kummer surface equations, we obtain several quadratic and even linear relations. For instance, one immediately obtains

$$a_{2,2,3} = 4(f_6\Delta - g_6).$$

This requires a total of one full and one small-scalar multiplication, compared to 96 multiplications to compute the same coefficient in the original formulae by means of a Horner scheme.

Expressing all coefficients from these relations Despite the relations between the coefficients only being quartic, we still have a system of 282 equations in 80 unknowns. A direct Gröbner basis computation is hence completely out of reach. After clearing the easiest (linear) relations, it becomes evident that one can not simply backsubstitute to obtain easy expressions for all $a_{i,j,k}, b_{i,j,k}, c_{i,j,k}, d_{i,j,k}$, since more of these always show up when trying to introduce new relations for a partial Gröbner basis computation. Nonetheless, looking at the lowest degrees in which the coefficients occur, we can distinguish four sets as follows:

$$\begin{aligned}
\mathcal{S}_1 &= \{a_{i,3,3}, b_{i,3,3}, c_{i,3,3} \mid 0 \le i \le 3\},\\
\mathcal{S}_2 &= \{a_{i,j,3}, b_{i,j,3}, c_{i,j,3}, d_{i,3,3} \mid 0 \le i, j \le 2\},\\
\mathcal{S}_3 &= \{a_{i,j,k}, b_{i,j,k}, c_{i,j,k}, d_{i,j,3} \mid 0 \le i, j, k \le 2\},\\
\mathcal{S}_4 &= \{d_{i,j,k} \mid 0 \le i, j, k \le 2\}.
\end{aligned}$$

The coefficients in $\mathcal{S}_1$ are easiest to express since they are all either zero or (a small power of) Δ. All coefficients in $\mathcal{S}_2$ satisfy at least one linear relation only involving terms in $\mathcal{S}_2 \cup \mathfrak{G}_2$, where $\mathfrak{G}_2 = \{f_i\Delta \mid 0 \le i \le 6\} \cup \{g_i \mid 0 \le i \le 6\}$. As evident by the example of $a_{2,2,3}$ above, some of these can be expressed directly in terms of $\mathfrak{G}_2$. There are however fewer such linear relations than variables,

hence we can not expect this to always be the case. Nonetheless, it seems like $\mathfrak{G}_2$ is a good candidate set to express the elements of $\mathcal{S}_2$ in the following sense.

Let $\mathcal{S}'_2 \subset \mathcal{S}_2$ be the subset of coefficients for which we have already found an (easy) expression. Fix an (arbitrary) ordering for all monomials $r^i s^j t^k$ occurring in $\mathcal{S}'_2 \cup \mathfrak{G}_2$. Define the matrix A as the one where each row represents an element from $\mathcal{S}'_2 \cup \mathfrak{G}_2$, and where the column entries correspond to the coefficient at the (fixed ordering) monomial $r^i s^j t^k$ (including a lot of zeros for missing terms). Choose an element $s \in \mathcal{S}_2$ which is not in $\mathcal{S}'_2$, and express it as a column vector $\vec{s}$ based on that same monomial ordering. Finding an expression for s in terms of $\mathcal{S}'_2 \cup \mathfrak{G}_2$ now boils down to finding a solution for the linear system

$$A\vec{x} = \vec{s}.$$

For all but three of the elements of $\mathcal{S}_2$, we could find such solutions $\vec{x}$.[3] However, we are not looking for just *any* solution $\vec{x}$, we are looking for one such that it results in an easy to evaluate expression for the respective s. To find the actual fastest evaluation of s, we would need to know the concrete performance of our hard- and software related to finite field arithmetic. To avoid this being too platform-dependent, we content ourselves by trying to find the $\vec{x}$ such that $||\vec{x}||_0$ is smallest; i.e. we are looking for the sparsest solution $\vec{x}$ out of all options.

If an extremely sparse solution exists (such as in the case of $a_{2,3,3}$), an algebra software package such as Magma may return it directly when asked to solve the system $A\vec{x} = \vec{s}$. Unfortunately, this stops being the case rather quickly. Denoting $\mathcal{L}$ as the lattice of the kernel of A, we can also compute the closest vectors $\vec{v_i}$ to $\mathcal{L}$ for one already-found $\vec{x}$. The difference between $\vec{x}$ and these close vectors yield solutions to the linear system of equations with minimized L_2-norm. Even though a minimal $||\vec{x}-\vec{v_i}||_2$ will not result in a minimal $||\vec{x}-\vec{v_i}||_0$, generically we can still expect the latter to be small as well. Assuming we can enumerate enough close vectors $\vec{v_i}$, we can simply compute all $||\vec{x} - \vec{v_i}||_0$ and choose the one which results in the sparsest solution. Even though this strategy provided some solid results, it - again unfortunately - stops being convenient rather quickly since the close vectors with regard to the L_2-norm stopped yielding sparse solutions. Furthermore, for $\mathcal{S}_3$ and $\mathcal{S}_4$ up ahead, this approach resulted in too large dimensions for $\mathcal{L}$ so we had to resort to other methods.

We are trying to find solutions to $A\vec{x} = \vec{s}$ under the condition that $||\vec{x}||_0$ is minimal. This setting can be translated to the following system of conditions:

$$\begin{aligned} A\vec{x} = \vec{s}, \qquad & y_i \in \{0, 1\}, \\ \sum_i y_i \leq m, \qquad & |x_i| \leq M y_i, \end{aligned}$$

where $\vec{y}$ is a vector of boolean predicates with coefficients y_i, the integer m is an upper bound for the amount of nonzero entries in $\vec{x}$, and M is an upper bound for the coefficients in $\vec{x}$. This is a problem that can be solved using

[3] These three missing expressions seem independent on the order in which we add elements to $\mathcal{S}'_2$. They are still relatively compactly representable however.

Mixed Integer Linear Programming and is used often in machine learning. This problem is NP-hard, but for smallish m, M, and somewhat restricted dimensions of A, this can be solved in reasonable time using Python's built-in function `scipy.optimize.milp`.

Using these methods, we managed to find sparse solutions $\vec{x}$ for all $s \in \mathcal{S}_2$. For $\mathcal{S}_3$ we remark that they stem from quadratic and cubic equations which always involve a factor Δ, hence we must consider $\Delta\mathcal{S}_3 = \{\Delta s \mid s \in \mathcal{S}_3\}$ instead. The other terms in the obtained relations are in

$$\mathfrak{G}_3 = \mathcal{S}_2^2 \cup \{f_i \Delta s \mid 0 \leq i \leq 6, s \in \mathcal{S}_2\} \cup \{g_i s \mid 0 \leq i \leq 6, s \in \mathcal{S}_2\},$$

where $\mathcal{S}_2^2 = \{s_i s_j | s_i, s_j \in \mathcal{S}_2\}$. For $\mathcal{S}_4$ the trend continues in a similar fashion, and the other terms in the relation come from

$$\mathfrak{G}_4 = \mathcal{S}_2^3 \cup \Delta\mathcal{S}_3 \cup \{f_i \Delta s \mid 0 \leq i \leq 6, s \in \mathcal{S}_2^2\} \cup \{g_i s \mid 0 \leq i \leq 6, s \in \mathcal{S}_2^2\},$$

where $\mathcal{S}_2^3 = \{s_i s_j s_k | s_i, s_j, s_k \in \mathcal{S}_2\}$.

Final results Using the methods outlined above, we managed to find expressions for all $a_{i,j,k}, b_{i,j,k}, c_{i,j,k}, d_{i,j,k}$. Replacing a squaring and cubing with one respectively two multiplications, evaluating all of these coefficients now takes 2.234 multiplications, which is a 94% reduction compared to the formulae from [4]. These coefficients can be recycled for each point on the Kummer surface, such that pushing multiple points through at once comes with little overhead compared to pushing through just one point. We emphasize that the obtained expressions do not work for certain small field characteristics, but this imposes no restrictions for cryptographic applications.

4 Non-generic $(3, 3)$-Isogeny Formulae

In this section we will describe formulae for $(3, 3)$-isogenies where the domain or the codomain are products of elliptic curves. For the splitting and the gluing case our formulae are based on the parametrization by Bröker, Howe, Lauter and Stevenhagen [3].

4.1 $(3, 3)$-Isogenies between Elliptic Products

Let $\Phi : E_1 \times E_2 \to E_1' \times E_2'$ be a $(3, 3)$-isogeny such that $\ker \Phi$ is diagonal; i.e. it is of the form$\langle (P_1, \infty_{E_2}), (\infty_{E_1}, P_2) \rangle$. Then Φ decomposes as a direct product $\phi_1 \times \phi_2$, where $\phi_i : E_i \to E_i'$ are 3-isogenies for $i \in \{1, 2\}$. Formulae for Φ can hence be obtained from elliptic curves with rational 3-torsion.

Proposition 1 *Let Φ be a $(3, 3)$-isogeny between products of elliptic curves, with diagonal kernel which is generated by rational points. Then up to isomorphism, this isogeny is given by*

$$\begin{aligned} \Phi : E_1 \times E_2 &\to E_1' \times E_2' \\ (P_1, P_2) &\mapsto (\phi_1(P_1), \phi_2(P_2)), \end{aligned}$$

where for $i \in \{1, 2\}$ *we have*

$$E_i : y^2 + a_i xy + b_i y = x^3$$
$$E_i' : y^2 + a_i xy + b_i y = x^3 - 5a_i b_i x - a_i^3 b_i - 7b_i^2$$

and

$$\phi_i(x, y) = \left(\frac{x^3 + a_i b_i x + b_i^2}{x^2}, \frac{x^3 y - a_i^2 b_i x^2 - a_i b_i xy - 2a_i b_i^2 x - 2b_i^2 y - b_i^3}{x^3} \right).$$

Proof. The parametrization of 3-isogenies between elliptic curves is well-known, see for example [9, Section 4]. The evaluation on points is a straightforward computation by using $(0, 0)$ as kernel generator in Vélu-style formulae. □

If the kernel of Φ is nondiagonal, then E_1 and E_2 are necessarily connected by a 2-isogeny, which follows from Kani's reducibility criterion [18]. Its parametrization is independent of the rationality of the 3-torsion.

Proposition 2 *Let* Φ *be a* $(3, 3)$*-isogeny between products of elliptic curves, with nondiagonal kernel. Then this isogeny is an endomorphism given by*

$$\Phi : E_1 \times E_2 \to E_1 \times E_2$$
$$(P_1, P_2) \mapsto (P_1 + \hat{\phi}(P_2), P_2 - \phi(P_1)),$$

where $\phi : E_1 \to E_2$ *is a* 2*-isogeny such that* $\ker \Phi = \{(P, \phi(P)) \mid P \in E_1[3]\}$*. Up to isomorphism*[4]*,* ϕ *is given by*

$$\phi : E_1 \to E_2$$
$$(x, y) \mapsto \left(\frac{x^2 + b}{x}, y \cdot \frac{x^2 - b}{x^2} \right),$$
$$\hat{\phi} : E_2 \to E_1$$
$$(x, y) \mapsto \left(\frac{x^2 - 4b}{4(x + a)}, y \cdot \frac{x^2 + 2ax + 4b}{8(x + a)^2} \right),$$

where

$$E_1 : y^2 = x^3 + ax^2 + bx, \qquad E_2 : y^2 = x^3 + ax^2 - 4bx - 4ab.$$

Proof. Due to [18, Theorem 2.6], it must hold that E_1 and E_2 are connected by means of a 2-isogeny. Consider the following commutative diagram as in [22, Theorem 1], where ϕ and ϕ' are 2-isogenies:

$$\begin{array}{ccc} E_1 & \xrightarrow{\phi} & E_2 \\ \downarrow{\scriptstyle \gamma} & & \downarrow{\scriptstyle \gamma'} \\ E_1' & \xrightarrow{\phi'} & E_2' \end{array}$$

[4] If the relevant 2-torsion is not rational, this isomorphism may be defined over a (small) field extension, regardless of the rationality of the 3-torsion.

Then

$$\begin{aligned} \Phi : E_1' \times E_2 &\to E_1 \times E_2' \\ (P_1, P_2) &\mapsto (\hat{\gamma}(P_1) + \hat{\phi}(P_2), \gamma'(P_2) - \phi'(P_1)) \end{aligned}$$

is a $(\deg\gamma + \deg\phi, \deg\gamma + \deg\phi)$-isogeny which preserves product polarizations. The degree of γ and γ' is then necessarily one if we want to construct a $(3,3)$-isogeny, such that up to isomorphism we can choose $E_i = E_i'$ for $i \in \{1,2\}$, as well as $\phi = \phi'$. The isogeny Φ can then be evaluated as $\Phi(P_1, P_2) = (P_1 + \hat{\phi}(P_2), P_2 - \phi(P_1))$. The parametrization of the 2-isogeny ϕ is such that $\ker\phi = \langle(0,0)\rangle$. □

Remark 3. For a fixed $\mathbb{F}_{p^2}$, there are $\mathcal{O}(p)$ supersingular elliptic curves and all have exactly three outgoing 2-isogenies, yet there are $\mathcal{O}(p^3)$ superspecial p.p.a.s. The ratio of products of 2-isogenous supersingular elliptic curves to superspecial p.p.a.s. is hence $\mathcal{O}(p^{-2})$. Therefore, it is to be expected that in cryptographic applications, isogenies as in Proposition 2 will never be used, unless the protocol is constructed in a particular way to encounter this type.

4.2 Splitting

Here, we consider $(3,3)$-isogenies where the domain is the Jacobian of a genus-2 curve and the codomain is a product of elliptic curves. Such an isogeny is called a $(3,3)$-splitting and it arises from degree-3 covers $\psi_1 : \mathcal{C} \to E_1$ and $\psi_2 : \mathcal{C} \to E_2$. More precisely, it is the product of the push forwards of these maps, i.e. $\Phi = \psi_{1,*} \times \psi_{2,*}$ To make these more explicit, assume that the maps ψ_1 and ψ_2 are compatible with the hyperelliptic involution $\iota : \mathcal{C} \to \mathcal{C}$, i.e. we have $\psi_i(\iota(P)) = -\psi_i(P)$ for all $P \in \mathcal{C}(K)$. In that case, the isogeny is given by

$$\begin{aligned} \Phi : \mathrm{Jac}(\mathcal{C}) &\to E_1 \times E_2 \\ [P + Q - D_\infty] &\mapsto (\psi_1(P) + \psi_1(Q),\ \psi_2(P) + \psi_2(Q)) \,. \end{aligned} \tag{3}$$

There exists a complete parametrization for this case and an explicit description of the maps ψ_1 and ψ_2 by Bröker, Howe, Lauter and Stevenhagen [3].

Proposition 3 (Proposition A.2 in [3]) *Let $a, b, c, d, t \in k$ satisfy*

$$12ac + 16bd = 1, \quad \Delta_1 = a^3 + b^2 \neq 0, \quad \Delta_2 = c^3 + d^2 \neq 0, \quad t \neq 0.$$

Define polynomials $F_{a,b,c,d,t}, f_1, f_2$ by

$$\begin{aligned} F_{a,b,c,d,t} &= (x^3 + 3ax + 2b)(2dx^3 + 3cx^2 + 1), \\ f_1 &= x^3 + 12(2a^2d - bc)x^2 + 12(16ad^2 + 3c^2)\Delta_1 x + 512\Delta_1^2 d^3, \\ f_2 &= x^3 + 12(2bc^2 - ad)x^2 + 12(16b^2c + 3a^2)\Delta_2 x + 512\Delta_2^2 b^3, \end{aligned}$$

and consider the curves $C_{a,b,c,d,t} : ty^2 = f(x)$ and $E_{a,b,c,d,t,i} : ty^2 = f_i(x)$ for $i \in \{1,2\}$. Then there are degree-3 morphisms given by

$$\begin{aligned} \psi_{a,b,c,d,t,i} : \mathcal{C}_{a,b,c,d,t} &\to E_{a,b,c,d,t,i} \\ (x, y) &\mapsto (r_i(x), s_i(x) \cdot y) \end{aligned}$$

with

$$r_1(x) = 12\Delta_1 \frac{-dx+e}{x^3+3ax+2b}, \qquad s_1 = \Delta_1 \frac{16dx^3-12cx^2-1}{(x^3+3ax+2b)^2},$$
$$r_2(x) = 12\Delta_2 \frac{x^2(ax-2b)}{2dx^3+3cx^2+1}, \qquad s_2 = \Delta_2 \frac{x^3+12ax-16b}{(2dx^3+3cx^2+1)^2}.$$

On the other hand, if $\mathcal{C}$ is a genus-2 curve whose Jacobian is $(3,3)$-isogenous to a product of elliptic curves $E_1 \times E_2$, then there exists a quintuple (a,b,c,d,t) as above and isomorphisms $\mathcal{C} \to \mathcal{C}_{a,b,c,d,t}$ and $E_i \to E_{a,b,c,d,t,i}$ for $i \in \{1,2\}$.

From now on, we assume that we are in the setting of the above proposition. The determination of the necessary coordinate transformation from $\mathcal{C}$ to a curve $\mathcal{C}_{a,b,c,d,t}$ will be explained in Section 5.2. Using the explicit description of the maps $\psi_{a,b,c,d,t,1}$, $\psi_{a,b,c,d,t,2}$ one can derive a formula for the map $\Phi : \mathrm{Jac}(\mathcal{C}_{a,b,c,d,t}) \to E_{a,b,c,d,t,1} \times E_{a,b,c,d,t,2}$ using the description in Equation 3. That said, here we focus on the induced map on the level of Kummer varieties.

Proposition 4 *Let $\mathcal{C}_{a,b,c,d,t}$ and $E_{a,b,c,d,t,i}$ for $i \in \{1,2\}$ be as in Proposition 3. Define*

$$\Phi_{\mathcal{K}} : \mathcal{K}(\mathcal{C}_{a,b,c,d,t}) \to \mathbb{P}^1 \times \mathbb{P}^1$$

as the induced $(3,3)$-isogeny on the level of Kummer surfaces. Then the kernel of $\Phi_{\mathcal{K}}$ is given by the subvariety of $\mathcal{K}(\mathcal{C})$ defined by

$$\left\{ \begin{array}{l} 0 = \xi_0^2 + 4c(\xi_1^2 - \xi_0\xi_2) - 8d\xi_1\xi_2 \\ 0 = \xi_2^2 + 4a(\xi_1^2 - \xi_0\xi_2) - 8b\xi_0\xi_1 \end{array} \right\}$$

Moreover, there exist polynomials $g_{x,1}, g_{z,1}, g_{x,2}, g_{z,2} \in \mathbb{Z}[\xi_0,\xi_1,\xi_2,\xi_3]$ such that

$$\Phi_{\mathcal{K}} : (\xi_0 : \xi_1 : \xi_2 : \xi_3) \mapsto ((g_{x,1}, g_{z,1}), (g_{x,2}, g_{z,2}))\,.$$

The explicit formulae for the map are provided in the auxiliary material, see Section 8.

Proof. The derivation of the explicit formulae for $\Phi_{\mathcal{K}}$ can be done using a computer algebra package. Here, we sketch the main steps.

Let $\xi = (\xi_0 : \xi_1 : \xi_2 : \xi_3) \in \mathcal{K}(\mathcal{C})$. We write $[P + Q - D_\infty] \in \mathrm{Jac}(\mathcal{C})$ with $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ for a point lying above ξ. To find explicit formulae, we will be working in the ring $\mathbb{Z}[\xi_0,\xi_1,\xi_2,\xi_3,x_1,y_1,x_2,y_2]$, but at least at the final step, the variables x_1, y_1, x_2, y_2 need to be eliminated. While we cannot deduce the explicit coordinates of P and Q from ξ alone, we can use the following identities for this elimination.

$$\begin{aligned} x_1 + x_2 &= \xi_1/\xi_0, \\ x_1 x_2 &= \xi_2/\xi_0, \\ 2y_1 y_2 &= \left(\phi(\xi_0,\xi_1,\xi_2) - \xi_3(\xi_1^2 - 4\xi_0\xi_2)\right)/\xi_0^3. \end{aligned} \tag{4}$$

Denote $\psi_i = \psi_{a,b,c,d,t,i}$ for $i \in \{1,2\}$. The isogeny Φ is equal to the product $\psi_{1,*} \times \psi_{2,*}$, cf. Equation 3. Using the explicit description of $\psi_i : \mathcal{C}_{a,b,c,d,t} \to E_{a,b,c,d,t,i}$ from Proposition 3, we symbolically compute $P_i = \psi_i(P)$ and $Q_i = \psi_i(Q)$ on E_i for $i \in \{1,2\}$. Applying the standard formulae for elliptic curve addition, we then compute $R_i = P_i + Q_i$ for $i \in \{1,2\}$. By construction, the obtained expressions for the coordinates of R_i are invariant under swapping P and Q, hence symmetric in x_1, x_2 and in y_1, y_2. Using the Kummer surface coordinates from Equation 4, it is therefore possible to completely eliminate x_1 and x_2 from the coordinates. The elimination of y_1 and y_2 is more subtle. While we can express $y_1 y_2$ in Kummer surface coordinates, this is not possible for the symmetric expression $y_1 + y_2$. However, it is still possible to eliminate the variables y_1 and y_2 from the formula for the x-coordinate of R_i, but this is not the case for its y-coordinate.[5]

As a result, we obtain quartic polynomials $g_{x,1}, g_{z,1}, g_{x,2}, g_{z,2} \in \mathbb{Z}[\xi_0, \xi_1, \xi_2, \xi_3]$ defining the map $\Phi_{\mathcal{K}} \to \mathbb{P}^1 \times \mathbb{P}^1$. The polynomials

$$g_{z,1} = \left(\xi_0^2 + 4c(\xi_1^2 - \xi_0\xi_2) - 8d\xi_1\xi_2\right)^2, \quad g_{z,2} = \left(\xi_2^2 + 4a(\xi_1^2 - \xi_0\xi_2) - 8b\xi_0\xi_1\right)^2$$

define the kernel of the isogeny. Explicit descriptions for $g_{x,1}, g_{x,2}$ and a formal verification can be found in the auxiliary material, see Section 8. □

4.3 Gluing

Let us now consider the case where the domain of the $(3,3)$-isogeny is a product of elliptic curves and its codomain the Jacobian of a genus-2 curve. Such an isogeny is called a $(3,3)$-*gluing*, and it is the dual of the $(3,3)$-split isogeny described above. Similar as in that case, a $(3,3)$-gluing Φ arises from degree-3 covers $\psi_1 : \mathcal{C} \to E_1$ and $\psi_2 : \mathcal{C} \to E_2$. And in this case it is the product of the pull-backs, i.e. $\Phi = \psi_1^* \times \psi_2^* : E_1 \times E_2 \to \mathrm{Jac}(\mathcal{C})$. To derive explicit formulae for this isogeny, we again use the parametrization from [3], cf. Proposition 3.

Proposition 5 *Let $\mathcal{C}_{a,b,c,d,t}$ and $E_{a,b,c,d,t,i}$ for $i \in \{1,2\}$ be as in Proposition 3. Denote*

$$\Phi : E_{a,b,c,d,t,1} \times E_{a,b,c,d,t,2} \to \mathit{Jac}(\mathcal{C}_{a,b,c,d,t})$$

the $(3,3)$-isogeny induced by the maps $\psi_{a,b,c,d,t,1}$ and $\psi_{a,b,c,d,t,2}$. Writing

$$\begin{aligned} \alpha_1 &= x_1 + 8a^2d - 6bc, & \beta_1 &= ax_1 + 8d\Delta_1, & \gamma_1 &= 48c\Delta_1 - 8bx_1, \\ \alpha_2 &= x_2 + 8bc^2 - 6ad, & \beta_2 &= cx_2 + 8b\Delta_2, & \gamma_2 &= 48a\Delta_2 - 8dx_2, \end{aligned}$$

we have

$$\begin{aligned} \psi^*_{a,b,c,d,t,i} : E_{a,b,c,d,t,i} &\to \mathit{Jac}(\mathcal{C}_{a,b,c,d,t}) \\ (x_i, y_i) &\mapsto [x^2 + \lambda_{i,1}x + \lambda_{i,0}, -4y_i(\mu_{i,1}x + \mu_{i,0})] \end{aligned}$$

[5] The element $y_1 + y_2$ can of course be represented in terms of the Mumford coordinates of $[P + Q - D_\infty]$ allowing to deduce a formula for the y-coordinates in that setting as well.

with

$$\lambda_{1,1} = \frac{-4\beta_1}{4\alpha_1^2 + a}, \qquad \lambda_{1,0} = \frac{\gamma_1}{4\alpha_1^2 + a}, \qquad \lambda_{2,1} = \frac{-4\beta_2}{\gamma_2}, \qquad \lambda_{2,0} = \frac{4\alpha_2^2 + c}{\gamma_2}$$

and

$$\mu_{1,1} = \frac{4\alpha_1^3 + 3a\alpha_1 + b}{(4\alpha_1^2 + a)^2}, \qquad \mu_{1,0} = \frac{4\beta_1^2 - \Delta_1}{a(4\alpha_1^2 + a)^2}$$
$$\mu_{2,1} = \frac{-4\Delta_2}{\gamma_2^2}, \qquad \mu_{2,0} = \frac{16\Delta_2(2bc + 3a(x_2 + \alpha_2)) - 8dx_2^2}{\gamma_2^2}.$$

Proof. Denote $\psi_i = \psi_{a,b,c,d,t,i} : \mathcal{C}_{a,b,c,d,t} \to E_{a,b,c,d,t,i}$. Counted with multiplicities, each point $P_i = (x_i, y_i) \in E_{a,b,c,d,t,i}$ has precisely 3 preimages in $\mathcal{C}_{a,b,c,d,t}$. From the explicit description of the maps ψ_i, one can directly read off the preimages of the neutral element on the two elliptic curves. Naturally, these consist of two disjoint sets of the Weierstrass point of the hyperelliptic curve. More precisely,

$$\psi_i^{-1}(\infty) = \{(\alpha, 0) \in \mathcal{C}_{a,b,c,d,t}(\bar{k}) \mid F_i(\alpha) = 0\},$$

where $F_1 = x^3 + 3ax + 2b$ and $F_2 = 2dx^3 + 3cx^2 + 1$ are the two factors of the defining polynomial of $\mathcal{C}_{a,b,c,d,t}$. This means that under the pull-back $\psi_i^* : E_{a,b,c,d,t,i} \to \mathrm{Jac}(\mathcal{C}_{a,b,c,d,t})$, the element $[P_i - \infty]$ gets mapped to

$$\left[\sum_{P' \in \psi_i^{-1}(P_i)} P' - \sum_{F_i(\alpha)=0} (\alpha, 0) \right] = \left[\sum_{P' \in \psi_i^{-1}(P_i)} P' + \sum_{F_i(\alpha)=0} (\alpha, 0) - 3 \cdot D_\infty \right].$$

Our strategy to find explicit formulae of this map is very similar to that in the proof of Proposition 4. That is, we first express the preimage of a general point $P_i = (x_i, y_i) \in E_i$ symbolically. This requires working in the ring $R = \mathbb{Q}[x_i, y_i, u_1, u_2, u_3, v_1, v_2, v_3]$, where (u_j, v_j) represent the coordinates of the points $\{P'_j\}$ in the preimage of P_i. In the final formula, the variables u_j, v_j need to be eliminated. To this end, one uses the explicit description of the maps ψ_i (Proposition 3), more precisely we use the identities

$$r_i(u_j) = x_i \quad \text{and} \quad s_i(u_j)v_j = y_i \tag{5}$$

for all $j \in \{1, 2, 3\}$. In particular, u_1, u_2, u_3 are the roots of a cubic polynomial in $\mathbb{Q}[x_i] \subset R$, hence their elementary symmetric polynomials are in $\mathbb{Q}[x_i]$. The next step is to find the unreduced Mumford coordinates of the divisor

$$D_P = \sum_{P' \in \psi_i^{-1}(P)} P' + \sum_{F_i(\alpha)=0} (\alpha, 0) - 3 \cdot D_\infty.$$

These consist of a pair of polynomials $[A, B]$, where $A \in R[x]$ is a degree-6 polynomial with roots the x-coordinates of the affine points in the support of D_P and $B \in R[x]$ interpolates the points in the support of D_P. For A, one

immediately finds an expression with coefficients in $\mathbb{Q}[x_i] \subset R$. It is the product of F_1 and the cubic polynomial with roots u_1, u_2, u_3. The computation of B requires more work. In our approach, we used the standard Newton interpolation method to represent the polynomial in $R[x]$, after which we used the above relations obtained from Equation 5 to eliminate the variables u_j, v_j.

It remains to compute the reduced Mumford coordinates for $[D_P]$. This can be done in two reduction steps following Cantor's algorithm. The resulting Mumford coordinates are $[x^2 + \lambda_{i,1}x + \lambda_{i,0}, -4y_i(\mu_{i,1}x + \mu_{i,0})]$ as in the statement of the proposition. A formal verification of the correctness of the formulae is provided in the auxiliary material, see Section 8. □

5 Coordinate Transformations

In the two previous sections, we discussed explicit formulae for all 4 types of $(3,3)$-isogenies. In order to apply these formulae to compute a specific $(3,3)$-isogeny, it is necessary to compute a coordinate transformation to a given parametrization. Here, we first discuss coordinate transformations in general and provide explicit formulae for the induced transformation on the Kummer surface. In the second part, we explain the determination of a suitable transformation, given as input a description of the $(3,3)$-kernel.

5.1 Explicit Formulae for Transformations

To fix some notation, we first recall a standard result about the coordinate transformations for hyperelliptic curves, see for example [21, Corollary 7.4.33].

Proposition 6 *Let $\mathcal{C} : y^2 = f(x), \mathcal{C}' : y'^2 = g(x')$ be two curves of genus 2 defined over k. Then $\mathcal{C}$ and $\mathcal{C}'$ are k-isomorphic iff there exist $\begin{pmatrix}\alpha & \beta \\ \gamma & \delta\end{pmatrix} \in GL_2(k)$ and $\epsilon \in k^\times$ such that*

$$x' = \frac{\alpha x + \beta}{\gamma x + \delta}, \quad y' = \frac{\epsilon y}{(\gamma x + \delta)^3}.$$

The above proposition describes the effect of coordinate transformations on the points of a hyperelliptic curve $\mathcal{C}$. One can also consider the induced maps on the Mumford coefficients of elements in the Jacobian $\mathrm{Jac}(\mathcal{C})$ or on the Kummer surface $\mathcal{K}(\mathcal{C})$. In our application, the latter type of transformation will be important. The derivation of the induced maps on the Kummer surface can be done by simple algebra. Since we could not find any explicit formulae in the literature, we provide the resulting formulae here.

Proposition 7 *Let $\mathcal{C} : y^2 = F(x), \mathcal{C}' : y'^2 = G(x')$ be two k-isomorphic genus-2 curves. Assume that the coordinate transformation between the curves is defined*

by $\begin{pmatrix}\alpha & \beta\\ \gamma & \delta\end{pmatrix} \in GL_2(k)$ *and* $\epsilon \in k^\times$ *as described in Proposition 6 and set* $\vartheta = \alpha\delta + \beta\gamma$, $\varrho = \alpha\beta\gamma\delta$. *Then*

$$\begin{aligned}
\xi_0' &= \delta^2\xi_0 + \gamma\delta\xi_1 + \gamma^2\xi_2,\\
\xi_1' &= 2\beta\delta\xi_0 + \vartheta\xi_1 + 2\alpha\gamma\xi_2,\\
\xi_2' &= \beta^2\xi_0 + \alpha\beta\xi_1 + \alpha^2\xi_2,\\
\xi_3' &= \left(\lambda_0\xi_0 + \lambda_1\xi_1 + \lambda_2\xi_2 + (\alpha\delta - \beta\gamma)^4\xi_3\right)/\epsilon^2,
\end{aligned}$$

with

$$\begin{aligned}
\lambda_0 =& -2\alpha^2\gamma^2(3\vartheta^2 - 8\varrho)f_0 + 2\alpha\gamma\vartheta(\vartheta^2 - 2\varrho)f_1 - 4\varrho(\vartheta^2 - 2\varrho)f_2\\
&+ \beta\delta\vartheta^3 f_3 - 2\beta^2\delta^2\vartheta^2 f_4 + 4\beta^3\delta^3\vartheta f_5 - 8\beta^4\delta^4 f_6,\\
\lambda_1 =& -4\alpha^3\gamma^3\vartheta f_0 + \alpha^2\gamma^2(\vartheta^2 + 4\varrho)f_1 - 4\alpha\gamma\varrho\vartheta f_2 + \varrho(\vartheta^2 + 4\varrho)f_3\\
&- 4\beta\delta\varrho\vartheta f_4 + \beta^2\delta^2(\vartheta^2 + 4\varrho)f_5 - 4\beta^3\delta^3\vartheta f_6,\\
\lambda_2 =& -8\alpha^4\gamma^4 f_0 + 4\alpha^3\gamma^3\vartheta f_1 - 2\alpha^2\gamma^2\vartheta^2 f_2 + \alpha\gamma\vartheta^3 f_3 - 4\varrho(\vartheta^2 - 2\varrho)f_4\\
&+ 2\beta\delta\vartheta(\vartheta^2 - 2\varrho)f_5 - 2\beta^2\delta^2(3\vartheta^2 - 8\varrho)f_6,
\end{aligned}$$

defines the corresponding coordinate transformations between the Kummer surfaces $\mathcal{K}(\mathcal{C})$ *and* $\mathcal{K}(\mathcal{C}')$.

Proof. This can be verified by a direct computation. □

5.2 Finding the Correct Transformation

Given two generators T_1, T_2 of a $(3,3)$-subgroup of a p.p.a.s., we explain how to compute the coordinate transformation that allows us to apply one of the isogeny formulae from Section 3 and Subsection 4.2. While the exact procedure depends on the type of $(3,3)$-isogeny, we use a Gröbner basis approach as the underlying method in all cases.

Generic case (Section 3) We first consider the generic case, that is we are given two kernel generators $T_1, T_2 \in \mathrm{Jac}(\mathcal{C})$ which define a $(3,3)$-isogeny to the Jacobian of another genus-2 curve. The goal is to find parameters $(\alpha, \beta, \gamma, \delta, \epsilon)$ defining a coordinate transformation as in Proposition 6 and parameters (r, s, t) such that the image curve is in the form of $\mathcal{C}_{r,s,t}$ and the images of the kernel generators are given by T and T' as described in Subsection 3.1. An efficient method for this has already been developed in the context of a $(3,3)$-based hash function in [6]. We use their implementation for our algorithms as well.

Product case (Subsection 4.1) Transformations between products of elliptic curves are products of isomorphisms between elliptic curves. These are well-known and of the form $(x, y) \mapsto (u'^2x + r', u'^3y + s'u'^2x + t')$ for elliptic curves in Weierstraß form, where u' is necessarily a unit (see for example [11, Subsection 4.4.2]).

Splitting case (Subsection 4.2) Now assume we are given two kernel generators $T_1, T_2 \in \mathrm{Jac}(\mathcal{C})$ that define a $(3,3)$-isogeny to a product of elliptic curves. In this case, we need to find parameters $(\alpha, \beta, \gamma, \delta, \epsilon)$ defining the transformation together with parameters (a, b, c, d, t) such that the image of the curve $\mathcal{C} : y^2 = f(x)$ under the transformation is equal to $\mathcal{C}_{a,b,c,d,t} : ty^2 = F_{a,b,c,d,t}(x)$ as in Proposition 3. Note that such a transformation only exists if $\mathrm{Jac}(\mathcal{C})$ is indeed $(3,3)$-isogenous to a product of elliptic curves. In this case the splitting is unique with overwhelming probability. One method to (probabilistically) find the correct parameters is to symbolically equate the coefficients of

$$ty^2 = F_{a,b,c,d,t}(x) \quad \text{and} \quad y'^2 = f(x')$$

with

$$x' = \frac{\alpha x + \beta}{\gamma x + \delta}, \quad y' = \frac{\epsilon y}{(\gamma x + \delta)^3}.$$

Together with the conditions on the parameters

$$12ac + 16bd = 1, \quad \Delta_1 = a^3 + b^2 \neq 0, \quad \Delta_2 = c^3 + d^2 \neq 0, \quad t \neq 0,$$

this yields a system of equations in $k[a, b, c, d, t, \alpha, \beta, \gamma, \delta, \epsilon]$ which can be solved by a Gröbner basis computation. As mentioned before, the defined splitting is unique, however there will be in general multiple solutions to the system stemming from an equivalence relation on the set of parameters $\{a, b, c, d, t\}$, as well as on the parameters defining the transformation.

In practice, the Gröbner basis computation can be sped up by providing additional information on the parameters. Here, such information is available by means of the kernel generators T_1, T_2. While we are not aware of explicit formulae for the kernel generators defining the $(3,3)$-isogeny from Proposition 3, we did derive relations for the Kummer coordinates of the kernel generators for exactly the isogeny from Proposition 4. Hence, we add the conditions

$$\begin{aligned} 0 &= \left(\xi_0'^2 + 4c(\xi_1'^2 - \xi_0'\xi_2') - 8d\xi_1'\xi_2'\right)^2, \\ 0 &= \left(\xi_2'^2 + 4a(\xi_1'^2 - \xi_0'\xi_2') - 8b\xi_0'\xi_1'\right)^2 \end{aligned}$$

with $(\xi_0', \xi_1', \xi_2', \xi_3')$ the image of $(\xi_0, \xi_1, \xi_2, \xi_3) \in \{T_1, T_2\}$ computed as in Proposition 7. This not only speeds up the Gröbner basis computation to solve the system by several orders of magnitude, but it also guarantees that we use the correct kernel in the (unlikely) case that there exist two $(3,3)$-splittings.

Gluing case (Subsection 4.3) Despite being the dual of the splitting case, finding the coordinate transformation for the gluing is more intricate. Let $E_1 \times E_2$ be a product of elliptic curves. Over the algebraic closure $\bar{k}$, there exist 24 different ways to glue elliptic curves along their 3-torsion.[6] For instance, [3, Algorithm 5.4] explains how to compute all $(3,3)$-isogenous Jacobians of genus-2 curves for

[6] In light of Proposition 2, there may be slightly fewer if E_1 and E_2 are 2-isogenous.

a given product of elliptic curves. Here, we are only interested in one specific isogeny defined by a specific $(3,3)$-subgroup $\langle T_1, T_2\rangle \subset E_1 \times E_2$.

If the elliptic curves are given in general Weierstrass form $y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$, then we first apply the transformation $y' = (y + a_1x + a_3)/2$ to obtain an equation of the form $y'^2 = f(x)$. Now it suffices to consider coordinate transformations of the form $x = \alpha x + \beta$, $y' = \epsilon y$ preserving this form. In that setting, our goal is to find a pair of transformations $(\alpha_i, \beta_i, \epsilon_i)$ for E_i with $i \in \{1, 2\}$ and parameters (a, b, c, d, t) such that after applying the transformations, the elliptic curves are as in Proposition 3. Moreover, we require that (the transformations of) T_1, T_2 are in the kernel of the resulting $(3,3)$-isogeny.

The first conditions can be imposed by a standard coefficient comparison similar to the splitting case. For the second condition, we use our explicit description of the gluing map from Proposition 5 which yields four additional conditions per kernel generator. More precisely, we require that

$$\psi^*_{a,b,c,d,t,1}(T'_i) = -\psi^*_{a,b,c,d,t,2}(T'_i) \quad \text{for } i \in \{1, 2\}$$

which provides one condition per Mumford coordinate. The resulting system has a unique solution up to the equivalence relations described in the previous part, and it can be solved efficiently by a Gröbner basis computation.

6 A $(3,3)$-Variant of the CGL Hash Function

In this section, we will discuss an implementation of a $(3,3)$-variant of the CGL hash function similar to the one from [6].

6.1 Starting p.p.a.s.

In the elliptic-curve case, knowledge of the starting curve's endomorphism ring reveals cycles in the isogeny graph, which can be used to create collisions in the corresponding hash function (see for example [13]). Even though this has not been written down explicitly yet, there is no reason to assume that the same does not hold in higher dimensions. Ideally, our starting p.p.a.s. would thus be sampled randomly from the set of superspecial p.p.a.s. such that its endomorphism ring is unknown. Unfortunately, there is no known way to do this without revealing an isogeny path to a superspecial p.p.a.s. of which the endomorphism ring is known. This connecting isogeny can then be used to reveal the endomorphism ring of the resulting superspecial p.p.a.s. as well. In fact, hashing into the set of supersingular elliptic curves is still an open problem in cryptography.[7]

In our setting, the field characteristic will always be of the form $p = f \cdot 3^n - 1$ for some (necessarily even) cofactor f. This results in the Jacobians of the well-known genus-2 curve $\mathcal{C} : y^2 = x^6 - 1$ to be superspecial, and it is isomorphic to the

[7] At least in a trustless set-up; for a trusted set-up variant, see [1].

curve $\mathcal{C}' : y^2 = (x^2-1)(x^2-2x)(x-1/2)$ used in [8]. Since we can not randomly sample a superspecial p.p.a.s., we may as well start from $\mathrm{Jac}(\mathcal{C})$. Unfortunately this is a Jacobian which has degenerate 3-torsion elements in the sense of Remark 1. Indeed, one can readily verify that the divisor $D = (0, \sqrt{-1}) - \infty$ is an element in $\mathrm{Jac}(\mathcal{C})[3]$. Given how this would immediately put us in a setting where we would have to deal with overhead occurring with probability $\mathcal{O}(p^{-1})$, we thus choose a different starting p.p.a.s. by taking a random walk in the superspecial $(3,3)$-isogeny graph starting from $\mathrm{Jac}(\mathcal{C})$. The resulting superspecial p.p.a.s. are hardcoded in the accompanying code, one for each level of security. The code also provides a symplectic basis of $\mathrm{Jac}(\mathcal{C})[3^n]$ for each security level such that $(3^n, 3^n)$-groups can be sampled uniformly at random using the methods from [20, Section 2.3].

6.2 Optimal Strategies for $(3^n, 3^n)$-Isogeny Computations

Given a maximal isotropic group $\langle T, T' \rangle \subset \mathrm{Jac}(\mathcal{C})[3^n]$, the following are two possible ways of computing the isogeny with kernel $\langle T, T' \rangle$ as chain of n $(3,3)$-isogenies.

- One can compute $(3^{n-1}T, 3^{n-1}T')$, quotient out the $(3,3)$-subgroup generated by these two elements, and compute the images of T, T' under this initial isogeny Φ_1. In the next step, we can repeat this process, but then with $(3^{n-2}\Phi_1(T), 3^{n-2}\Phi_1(T'))$ and pushing through $(\Phi_1(T), \Phi_1(T'))$. Iteratively repeating this will compute the entire $(3^n, 3^n)$-isogeny.
- One can compute the set of pairs $\{(3^iT, 3^iT') : 0 \leq i < n\}$ inductively on the starting Jacobian by iterated multiplication by 3. Starting from the pair $(3^{n-1}T, 3^{n-1}T')$ one can then quotient out the $(3,3)$-subgroup generated by this pair, and push all other pairs of divisors through this initial isogeny Φ_1. This results in immediate access to the pair $(3^{n-2}\Phi_1(T), 3^{n-2}\Phi_1(T'))$ on the codomain Jacobian and we can continue inductively.

The above methods are extreme in the sense that the former computes a maximal amount of multiplications by 3, whereas the latter computes a maximal amount of images under each isogeny. In [17, Section 4], they discuss this in-depth in terms of optimal strategies, where trade-offs can be made to obtain a method in-between the extreme versions described here. They describe optimal strategies in terms of elemental $\mathbb{F}_{p^2}$-operations, which is possible since 2- and 3-isogenies between elliptic curves require a limited amount of such operations and are hence almost surely optimized already.

Unfortunately, this is not the case when working with elements of Jacobians. A first subtlety is that Magma has optimized internal arithmetic for computing a scalar multiple kT for $T \in \mathrm{Jac}(\mathcal{C})$. This internal structure outperforms reimplementing the arithmetic ourselves, which makes it impossible to express things in elemental $\mathbb{F}_{p^2}$-operations. Furthermore, the implemented scalar multiplication on Jacobians is noticeably faster if we compute the scalar $k = 3^{n-i}$ first, compared to repeatedly multiplying T by 3.

Another subtlety is that we push points through on the Kummer surface, and not on the Jacobian. This means that after the first step we do not have access to $(\Phi_1(T), \Phi_1(T'))$ but to their images on the associated Kummer surface. This is not a problem since we are only interested in quotienting out the subgroup generated by these elements, and one can perform scalar multiplications on the Kummer surface as well. The scalar multiplication formulae on our model of the Kummer surface are a lot more involved compared to their Jacobian counterparts however. In practice, one notices that it is often more efficient to lift points on the Kummer surface to an arbitrary preimage on the Jacobian, perform the scalar multiplication there, and then project back to the Kummer surface.

Due to these limitations, it is impossible to reuse the mathematical optimal strategies obtained in [17, Section 4]. Instead, we heuristically try out their well-formed balanced strategies for various pairs of weights. In any case, this leads to an approach that only requires $\mathcal{O}(n \log n)$ computations compared to the $\mathcal{O}(n^2)$ computations in the extreme methods outlined at the start of this subsection. In practice, putting the weights equal seemed to perform extremely well for all fields of cryptographic characteristic. The equal-weight strategy boils down to pushing $(3^{\lfloor n/2 \rfloor}T, 3^{\lfloor n/2 \rfloor}T')$ through the initial isogeny Φ_1, then $(3^{\lfloor 3n/4 \rfloor - 1}\Phi_1(T), 3^{\lfloor 3n/4 \rfloor - 1}\Phi_1(T'))$ through the second isogeny Φ_2, and so on. This resulted in a speed-up of a factor at least two compared to naively using either of the extreme methods.

6.3 Implementation

We ran our $(3,3)$-variant of the CGL hash function on an Intel Xeon Gold 6248R CPU at 3.00GHz in Magma V2.27-7. For a fair comparison to the $(3,3)$-variant of [6], we reran their code using this same set-up. The results can be found below.

	$p \approx 2^{86}$	$p \approx 2^{128}$	$p \approx 2^{171}$	$p \approx 2^{256}$
bits of classical security	128	192	256	384
bits of quantum security	86	128	170	256
time per bit processed [6] (reran)	3.23ms	3.30ms	3.56ms	4.09ms
time per bit processed (this work)	6.81ms	7.47ms	7.53ms	7.96ms

Even though these results seem dissatisfying, they are not completely unexpected. The difference in approach is that the $(3,3)$-variant of [6] does not require pushing points through the isogeny, nor does it require computing scalar multiplications on the Jacobians (or lifts from the Kummer to the Jacobian). Our version on the other hand has the benefit of not having to compute three cubic roots at each step in the isogeny chain. Over a field $\mathbb{F}_{p^2}$ with $p = f \cdot 3^n - 1$, the computation of a cubic root requires $O(n)$ multiplications. In contrast to that, our algorithm requires $O(\log(n))$ multiplications at each step. Note that we cannot provide a precise count of operations due to the small Gröbner basis computation involved at each step. This implies that our algorithm scales better with increasing bitsize. From a certain point onwards, it should outperform the one from [6], though only at very high security levels.

We stress that our implementation does not lend itself to an actual practical hash function. Recall that we work over a field with $p = f \cdot 3^n - 1$. With a given symplectic 3^n-torsion basis of the starting Jacobian, we can thus hash $\lfloor \log_2 3^{3n} \rfloor$ bits before having "depleted" the available 3^n-torsion. If we want to hash inputs that are longer, we then have to resample a new symplectic basis, which is a nontrivial operation compared to the hash function itself. The main goal of this implementation is to show how fast we can manoeuvre in the $(3,3)$-isogeny graph with a *given* $(3^n, 3^n)$-subgroup, *whilst* also being able to push points through the isogeny chain. Both of these requirements are needed in certain other applications, such as attacking Alice's private key in SIKE, which is the topic of the next section.

7 Recovering Alice's Private Key in SIKE

In essence, we follow the attack strategy described in [22]. The main difference is that we target the recovery of Alice's secret key. Let $p = 2^a 3^b - 1$ be a SIKE prime and define

$$E_0/\mathbb{F}_{p^2} : y^2 = x^3 + 6x^2 + x$$

to be the starting curve, as is the case in all SIKE instantiations, including fixed bases for its 2^a- and 3^b-torsion; i.e. $E_0[2^a] = \langle P_A, Q_A \rangle$ and $E_0[3^b] = \langle P_B, Q_B \rangle$. Define the curve $E_0'/\mathbb{F}_{p^2} : y^2 = x^3 + x$ with its well-known endomorphism $\iota : E_0' \to E_0',\ (x, y) \mapsto (-x, iy)$ and define $\rho : E_0 \to E_0'$ as the 2-isogeny connecting these curves.

Let k_A be the private key of Alice, which is used to construct the isogeny $\phi_A : E_0 \to E_A = E_0/\langle P_A + k_A Q_A \rangle$. In the SIDH protocol, Alice then sends the information $(E_A, \phi_A(P_B), \phi_A(Q_B))$ to Bob. To retrieve her private key, we consider the following commutative diagram:

$$\begin{array}{ccc} E_0 & \xrightarrow{\phi_A} & E_A \\ \downarrow{\scriptstyle \gamma} & & \downarrow{\scriptstyle \gamma'} \\ E_0 & \xrightarrow{\phi_A'} & X \end{array}$$

with the endomorphism

$$\begin{aligned} \gamma : E_0 &\to E_0 \\ P &\mapsto [u]P + \hat{\rho} \circ \rho \circ [v]P \end{aligned}$$

for certain integers u and v. It follows that the endomorphism γ is of degree $c = u^2 + 4v^2$. By [22, Theorem 1], the isogeny

$$\begin{aligned} \Phi : E_0 \times E_A &\to E_0 \times X \\ (P, Q) &\mapsto (\hat{\gamma}(P) + \hat{\phi}_A(Q), \gamma'(Q) - \phi_A'(P)) \end{aligned}$$

is a $(2^a + c, 2^a + c)$-isogeny preserving product polarizations. Furthermore, $\ker \Phi = \{([2^a]P, -\phi_A \circ \hat{\gamma}(P) \mid P \in E_0[2^a + c]\}$. We can compute Alice's private key k_A

from the kernel of ϕ_A, which in turn can be computed from $\hat{\phi}_a$. This dual isogeny can be retrieved, since

$$\hat{\phi}_A = E_A \to E_0 \times E_A \xrightarrow{\Phi} E_0 \times X \to E_0,$$

where the first map is inclusion and the last map is projection. To efficiently evaluate the $(2^a + c, 2^a + c)$-isogeny Φ using $(3,3)$-isogeny formulae, ideally we have that $c = 3^b - 2^a$. However, we run into two issues:

- the integer $3^b - 2^a$ may be negative;
- if $3^b - 2^a$ has a prime factor ℓ with odd multiplicity such that $\ell \equiv 3 \bmod 4$ then there are no u, v to compute γ.

For all the SIKE parameters, at least one of these is true. To combat this, we can instead look for integers $c' = 3^{b'} - d'2^{a'}$, where

1. if $b' > b$ then we need to guess $b' - b$ additional steps after the $(3^b, 3^b)$-isogeny;
2. if $b' < b$ then we we only need to compute a $(3^{b-b'}, 3^{b-b'})$-isogeny;
3. if $a' > a$ then we need to extend Alice's isogeny with a $2^{a'-a}$-isogeny;[8]
4. if $a' < a$ then we need to guess the first $2^{a-a'}$-isogeny component of ϕ_A;
5. d' gives us leeway to extend ϕ_A with an isogeny of degree d'.

Options 2, 3 and 5 seem most interesting because they involve no guessing, but they all come with the side effect that c' is less likely to be positive. Remark that d' needs to be coprime with 3. If we allow d' to be even, then we can always choose $a' \leq a$. Choosing $d' > 1$ will likely result in computing an isogeny with nonrational kernel generator, unless it factors as $2m^2$, in which case one can compute a rational 2-isogeny followed by $[m]$. We suggest the following choices, where we allow d' to be even and thus always have $a' \leq a$.

	b	a	b'	a'	d'
SIKEp434	137	216	138	215	1
SIKEp503	159	250	160	250	4
SIKEp610	192	305	192	301	1
			194	303	1
SIKEp751	239	372	238	372	10

For SIKEp434 this means we need to guess the first step of Alice's chain of 2-isogenies, and then after computing a $(3^{137}, 3^{137})$-isogeny determine the (almost always unique) $(3,3)$-splitting. For SIKEp503 we need to also guess the final $(3,3)$-splitting, but can just extend ϕ_A with [2] instead of having to guess. The options for SIKEp610 are a trade-off: either guess the first 2^4-isogeny component of ϕ_A, or only guess the 2^2-isogeny component but then also guess which $(3^2, 3^2)$-isogeny splits after computing a $(3^{192}, 3^{192})$-isogeny.

[8] Remark that we may extend by a 2^ε-isogeny which is a part of $\hat{\phi}_A$ for some $\epsilon \leq a' - a$. We would then need to guess the final 2^ε-isogeny of ϕ_A but this is easy for small ε.

For the SIKEp751 parameters we can use an endomorphism of degree $3^{238} - 10 \cdot 2^{372}$, which was factored using the number sieve from [29]. This factorisation allows us to avoid any need for guessing, and only requires us to extend Alice's secret isogeny by an (arbitrary) isogeny of degree ten. The 2-torsion is of course rational, and the twist contains rational 5-torsion, which allows for a swift auxiliary 10-isogeny using x-only arithmetic over $\mathbb{F}_{p^2}$. Using the techniques discussed earlier, we manage to retrieve Alice's secret isogeny ϕ_A in 11 seconds. To the best of our knowledge, the fastest recovery of Bob's secret isogeny is 1 hour for this security level [24]. Our computation consists of three nontrivial parts and is pretty consistent with regard to timing: 0.5 seconds for the Gröbner basis computation to glue the elliptic curves together, 10 seconds to compute the $(3^{236}, 3^{236})$-isogeny between Jacobians, and 0.5 second for the Gröbner basis computation for the final $(3, 3)$-splitting. All timings are based on using an Intel Xeon Gold 6248R CPU at 3.00GHz in Magma V2.27-7.

8 Auxiliary Code

The auxiliary material contains the following code and can be found online at `https://github.com/KULeuven-COSIC/3_3_isogenies`

- `33_hash_BFT.m` contains an implementation of the CGL hash function described in Section 6.
- `BFT_verification.m` contains the symbolic formulae verification of the results from Subsection 3.2.
- `symplectic_basis.m` contains a generating function for symplectic bases, which was used to hardcode the starting torsion of `33hashBFT.m`.
- `SIKEp751_attack.m` includes the attack on Alice's secret isogeny as described in Section 7.
- `uv_list.m` contains factorizations relevant to the SIKE attack parameters from Section 7.
- `verification_split.sage` contains a script to verify the splitting formulae from Subsection 4.2.
- `verification_glue.sage` contains a script to verify the gluing formulae from Subsection 4.3.

References

1. Basso, A., Codogni, G., Connolly, D., De Feo, L., Fouotsa, T.B., Lido, G.M., Morrison, T., Panny, L., Patranabis, S., Wesolowski, B.: Supersingular curves you can trust. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology - EUROCRYPT 2023. Lecture Notes in Computer Science, vol. 14005, pp. 405–437. Springer (2023). `https://doi.org/10.1007/978-3-031-30617-4_14`
2. Bosma, W., Cannon, J., Playoust, C.: The Magma algebra system. I. The user language. Journal of Symbolic Computation **24**(3-4), 235–265 (1997). `https://doi.org/10.1006/jsco.1996.0125`

3. Bröker, R., Howe, E.W., Lauter, K.E., Stevenhagen, P.: Genus-2 curves and Jacobians with a given number of points. LMS Journal of Computation and Mathematics **18**(1), 170–197 (2015). https://doi.org/doi:10.1112/S1461157014000461
4. Bruin, N., Flynn, E.V., Testa, D.: Descent via (3, 3)-isogeny on Jacobians of genus 2 curves. Acta Arithmetica **165**(3), 201–223 (2014), http://eudml.org/doc/279018
5. Cassels, J.W.S., Flynn, E.V.: Prolegomena to a middlebrow arithmetic of curves of genus 2, vol. 230. Cambridge University Press (1996). https://doi.org/10.1017/CBO9780511526084
6. Castryck, W., Decru, T.: Multiradical isogenies. In: 18th International Conference Arithmetic, Geometry, Cryptography, and Coding Theory, Contemporary mathematics, vol. 779, pp. 57–89. American Mathematical Society (2022). https://doi.org/10.1090/conm/779
7. Castryck, W., Decru, T.: An efficient key recovery attack on SIDH. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology - EUROCRYPT 2023. Lecture Notes in Computer Science, vol. 14008, pp. 423–447. Springer (2023). https://doi.org/10.1007/978-3-031-30589-4_15
8. Castryck, W., Decru, T., Smith, B.: Hash functions from superspecial genus-2 curves using Richelot isogenies. Journal of Mathematical Cryptology **14**(1), 268–292 (2020). https://doi.org/doi:10.1515/jmc-2019-0021
9. Castryck, W., Decru, T., Vercauteren, F.: Radical isogenies. In: Advances in Cryptology – ASIACRYPT 2020. vol. 2, pp. 493–519. Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-64834-3_17
10. Charles, D.X., Lauter, K.E., Goren, E.Z.: Cryptographic hash functions from expander graphs. J. Cryptol. **22**(1), 93–113 (2009). https://doi.org/10.1007/s00145-007-9002-x
11. Cohen, H., Frey, G., Avanzi, R., Doche, C., Lange, T., Nguyen, K., Vercauteren, F.: Handbook of elliptic and hyperelliptic curve cryptography. CRC press (2005). https://doi.org/10.1201/9781420034981
12. Cosset, R., Robert, D.: Computing (ℓ, ℓ)–isogenies in polynomial time on Jacobians of genus 2 curves. Mathematics of Computation **84**(294), 1953–1975 (2015), http://www.jstor.org/stable/24489183
13. Eisenträger, K., Hallgren, S., Lauter, K., Morrison, T., Petit, C.: Supersingular isogeny graphs and endomorphism rings: Reductions and solutions. In: Advances in Cryptology – EUROCRYPT 2018. vol. 3, pp. 329–368. Springer International Publishing (2018). https://doi.org/10.1007/978-3-319-78372-7_11
14. Flynn, E.V., Ti, Y.B.: Genus two isogeny cryptography. In: Post-Quantum Cryptography. pp. 286–306. Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-25510-7_16
15. Galbraith, S.D., Harrison, M., Mireles Morales, D.J.: Efficient hyperelliptic arithmetic using balanced representation for divisors. In: International Algorithmic Number Theory Symposium. pp. 342–356. Springer (2008). https://doi.org/10.1007/978-3-540-79456-1_23
16. Gaudry, P.: An algorithm for solving the discrete log problem on hyperelliptic curves. In: Advances in Cryptology — EUROCRYPT 2000. pp. 19–34. Springer Berlin Heidelberg (2000). https://doi.org/10.1007/3-540-45539-6_2
17. Jao, D., De Feo, L.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In: Post-Quantum Cryptography. pp. 19–34. Springer Berlin Heidelberg (2011). https://doi.org/10.1007/978-3-642-25405-5_2
18. Kani, E.: The number of curves of genus two with elliptic differentials. Journal für die reine und angewandte Mathematik **1997**(485), 93–122 (1997). https://doi.org/doi:10.1515/crll.1997.485.93

19. Kunzweiler, S.: Efficient computation of $(2^n, 2^n)$-isogenies. Cryptology ePrint Archive, Paper 2022/990 (2022), https://eprint.iacr.org/2022/990
20. Kunzweiler, S., Ti, Y.B., Weitkämper, C.: Secret keys in genus-2 SIDH. In: Selected Areas in Cryptography - 28th International Conference, SAC 2021, Revised Selected Papers. Lecture Notes in Computer Science, vol. 13203, pp. 483–507. Springer (2021). https://doi.org/10.1007/978-3-030-99277-4_23
21. Liu, Q.: Algebraic geometry and arithmetic curves, vol. 6. Oxford University Press (2002)
22. Maino, L., Martindale, C., Panny, L., Pope, G., Wesolowski, B.: A direct key recovery attack on SIDH. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology - EUROCRYPT 2023. Lecture Notes in Computer Science, vol. 14008, pp. 448–471. Springer (2023). https://doi.org/10.1007/978-3-031-30589-4_16
23. National Institute of Standards and Technology (NIST): Post-quantum cryptography standardization process, https://csrc.nist.gov/projects/post-quantum-cryptography
24. Oudompheng, R., Pope, G.: A note on reimplementing the Castryck–Decru attack and lessons learned for SageMath. Cryptology ePrint Archive, Paper 2022/1283 (2022), https://eprint.iacr.org/2022/1283
25. Robert, D.: Breaking SIDH in polynomial time. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology - EUROCRYPT 2023. Lecture Notes in Computer Science, vol. 14008, pp. 472–503. Springer (2023). https://doi.org/10.1007/978-3-031-30589-4_17
26. Santos, M.C.R., Costello, C., Frengley, S.: An algorithm for efficient detection of (N, N)-splittings and its application to the isogeny problem in dimension 2. Cryptology ePrint Archive, Paper 2022/1736 (2022), https://eprint.iacr.org/2022/1736
27. Smith, B.: Isogenies and the discrete logarithm problem in Jacobians of genus 3 hyperelliptic curves. In: Advances in Cryptology – EUROCRYPT 2008. pp. 163–180. Springer Berlin Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_10
28. Takashima, K.: Efficient algorithms for isogeny sequences and their cryptographic applications. In: Mathematical Modelling for Next-Generation Cryptography: CREST Crypto-Math Project. pp. 97–114. Springer Singapore (2018). https://doi.org/10.1007/978-981-10-5065-7_6
29. The CADO-NFS Development Team: CADO-NFS, an implementation of the number field sieve algorithm (2017), http://cado-nfs.inria.fr/, release 2.3.0
30. The Sage Developers: SageMath, the Sage Mathematics Software System (Version 9.0) (2023), https://www.sagemath.org