

Hard Isogeny Problems over RSA Moduli and Groups with Infeasible Inversion

Salim Ali Altuğ* Yilei Chen†

May 14, 2019

Abstract

We initiate the study of computational problems on elliptic curve isogeny graphs defined over RSA moduli. We conjecture that several variants of the neighbor-search problem over these graphs are hard, and provide a comprehensive list of cryptanalytic attempts on these problems. Moreover, based on the hardness of these problems, we provide a construction of groups with infeasible inversion, where the underlying groups are the ideal class groups of imaginary quadratic orders.

Recall that in a group with infeasible inversion, computing the inverse of a group element is required to be hard, while performing the group operation is easy. Motivated by the potential cryptographic application of building a directed transitive signature scheme, the search for a group with infeasible inversion was initiated in the theses of Hohenberger and Molnar (2003). Later it was also shown to provide a broadcast encryption scheme by Irer et al. (2004). However, to date the only case of a group with infeasible inversion is implied by the much stronger primitive of self-bilinear map constructed by Yamakawa et al. (2014) based on the hardness of factoring and indistinguishability obfuscation (iO). Our construction gives a candidate without using iO.

*Boston University. saaltug@bu.edu. Research supported by the grant DMS-1702176.

†Visa Research. chenylei.ra@gmail.com. Research conducted while the author was at Boston University supported by the NSF MACS project and NSF grant CNS-1422965.

Contents

1	Introduction	1
1.1	Elliptic curve isogenies in cryptography	1
1.2	Isogeny graphs over RSA moduli	2
1.3	Constructing a trapdoor group with infeasible inversion	3
1.4	Related works	7
2	Preliminaries	7
2.1	Ideal class groups of imaginary quadratic orders	8
2.2	Elliptic curves and their isogenies	9
2.3	Isogeny volcanoes and the class groups	10
3	Isogeny graphs over composite moduli	12
3.1	Isogeny graphs over $\mathbb{Z}/N\mathbb{Z}$	13
3.2	The ℓ -isogenous neighbors problem over $\mathbb{Z}/N\mathbb{Z}$	13
3.3	The (ℓ, m) -isogenous neighbors problem over $\mathbb{Z}/N\mathbb{Z}$	15
4	Trapdoor group with infeasible inversion	17
4.1	Definitions	17
4.2	Construction - 1: Basic setting	18
4.3	Construction - 2: General case	23
5	Cryptanalysis	27
5.1	The (in)feasibility of performing computations over $\mathbb{Z}/N\mathbb{Z}$	27
5.1.1	Feasible information from a single j -invariant	28
5.1.2	Computing explicit isogenies over $\mathbb{Z}/N\mathbb{Z}$ given more than one j -invariant	28
5.2	Tackling the (ℓ, ℓ^2) -isogenous neighbor problem	29
5.2.1	Hilbert class polynomial attack	30
5.2.2	Deciding the direction of an isogeny on the volcano	30
5.2.3	More about modular curves and characteristic zero attacks	31
5.3	Cryptanalysis of the candidate group with infeasible inversion	32
5.3.1	Preventing the trivial leakage of inverses	33
5.3.2	Parallelogram attack	33
5.3.3	Hiding the class group invariants: why and how	34
5.4	Miscellaneous	37
5.5	Summary	38
6	Directed transitive signature for directed acyclic graphs	39
6.1	Definition	40
6.2	A directed transitive signature scheme for DAGs via TGII	41
7	Broadcast encryption	43
7.1	Definition	44
7.2	A private-key broadcast encryption scheme from TGII	45
8	Future directions	46

1 Introduction

Let \mathbb{G} denote a finite group written multiplicatively. The discrete-log problem asks to find the exponent a given g and $g^a \in \mathbb{G}$. In the groups traditionally used in discrete-log-based cryptosystems, such as $(\mathbb{Z}/q\mathbb{Z})^*$ [DH76], groups of points on elliptic curves [Mil85, Kob87], and class groups [BW88, McC88], computing the inverse $x^{-1} = g^{-a}$ given $x = g^a$ is easy. We say \mathbb{G} is a *group with infeasible inversion* if computing inverses of elements is hard, while performing the group operation is easy (i.e. given g, g^a, g^b , computing g^{a+b} is easy).

The search for a group with infeasible inversion was initiated in the theses of Hohenberger [Hoh03] and Molnar [Mol03], motivated with the potential cryptographic application of constructing a directed transitive signature. It was also shown by Irrer et al. [ILOP04] to provide a broadcast encryption scheme. The only existing candidate of such a group, however, is implied by the much stronger primitive of self-bilinear maps constructed by Yamakawa et al. [YYHK14], assuming the hardness of integer factorization and indistinguishability obfuscation (iO) [BGI⁺01, GGH⁺13].

In this paper we propose a candidate trapdoor group with infeasible inversion without using iO. The underlying group is isomorphic to the ideal class group of an imaginary quadratic order (henceforth abbreviated as “*the class group*”). In the standard representation¹ of the class group, computing the inverse of a group element is straightforward. The representation we propose uses the volcano-like structure of the isogeny graphs of ordinary elliptic curves. In fact, the initiation of this work was driven by the desire to explore the computational problems on the isogeny graphs defined over RSA moduli.

1.1 Elliptic curve isogenies in cryptography

An isogeny $\varphi : E_1 \rightarrow E_2$ is a morphism of elliptic curves that preserves the identity. Given two isogenous elliptic curves E_1, E_2 over a finite field, finding an explicit rational polynomial that represents an isogeny from E_1 to E_2 is traditionally called the *computational isogeny problem*.

The study of computing explicit isogenies began with the rather technical motivation of improving Schoof’s polynomial time algorithm [Sch85] to compute the number of points on an elliptic curve over a finite field (the improved algorithm is usually called Schoof-Elkies-Atkin algorithm, cf. [CM94, Sch95, E⁺98] and references therein). A more straightforward use for explicit isogenies is to transfer the elliptic curve discrete-log problem from one curve to the other [Gal99, GHS02, JMV05]. If for any two isogenous elliptic curves computing an isogeny from one to the other is efficient, then it means the discrete-log problem is equally hard among all the curves in the same isogeny class.

The best way of understanding the nature of the isogeny problem is to look at the *isogeny graphs*. Fix a finite field \mathbf{k} and a prime ℓ different than the characteristic of \mathbf{k} . Then the isogeny graph $G_\ell(\mathbf{k})$ is defined as follows: each vertex in $G_\ell(\mathbf{k})$ is a j -invariant of an isomorphism class of curves; two vertices are connected by an edge if there is an isogeny of degree ℓ over \mathbf{k} that maps one curve to another. The structure of the isogeny graph is described in the PhD thesis of Kohel [Koh96]. Roughly speaking, a connected component of an isogeny graph containing ordinary elliptic curves looks like a *volcano* (termed in [FM02]). The connected component containing supersingular elliptic curves, on the other hand, has a different structure. In this article we will focus on the ordinary case.

¹By emphasizing the “representation”, we would like to remind the readers that the hardness of group theoretical problems (like the discrete-log problem) depends on the group representation rather than the group structure. After all, most of the cryptographically interesting finite groups are chosen to be isomorphic to the innocent looking additive group $\mathbb{Z}/n\mathbb{Z}$, $n \in \mathbb{N}$. However, the isomorphism is typically hard to compute.

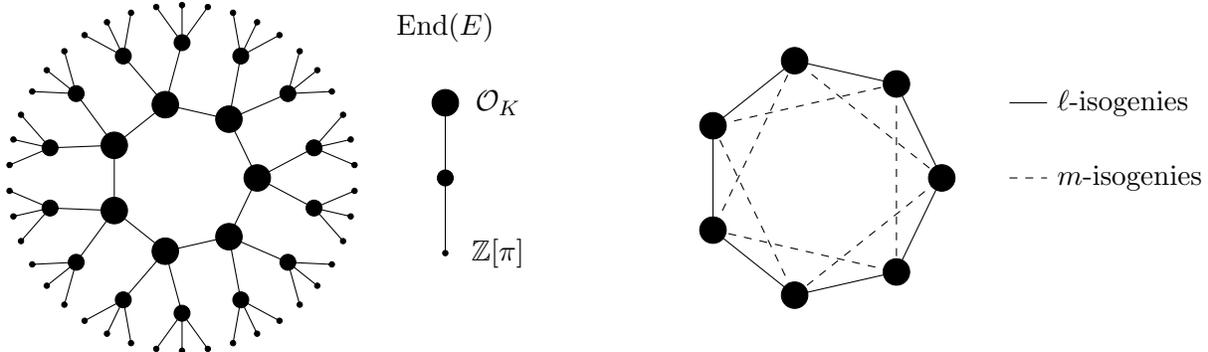


Figure 1: Examples of isogeny graphs. Left: a connected component of $G_\ell(\mathbf{k})$, and the corresponding tower of imaginary quadratic orders [Fec17]; Right: the vertex set is $\text{Ell}_{\mathcal{O}}(\mathbb{C})$ for an imaginary quadratic order \mathcal{O} , the edges represent (isomorphic classes of) isogenies of degrees ℓ, m .

A closer look at the algorithms of computing isogenies. Let \mathbf{k} be a finite field of q elements, ℓ be an integer such that $\gcd(\ell, q) = 1$. Given the j -invariant of an elliptic curve E , there are at least two different ways to find all the j -invariants of the curves that are ℓ -isogenous to E (or to a twist of E) and to find the corresponding rational polynomials that represent the isogenies:

1. Computing kernel subgroups of E of size ℓ , and then applying Vélú’s formulae to obtain explicit isogenies and the j -invariants of the image curves,
2. Calculating the j -invariants of the image curves by solving the ℓ^{th} modular polynomial Φ_ℓ over \mathbf{k} , and then constructing explicit isogenies from these j -invariants.

Both methods are able to find all the ℓ -isogenous neighbors over \mathbf{k} in time $\text{poly}(\ell, \log(q))$. In other words, *over a finite field*, one can take a stroll around the polynomial-degree isogenous neighbors of a given elliptic curve efficiently.

However, for two random isogenous curves over a sufficiently large field, finding an explicit isogeny between them seems to be hard, even for quantum computers. The conjectured hardness of computing isogenies was used in a key-exchange and a public-key cryptosystem by Couveignes [Cou06] and independently by Rostovtsev and Stolbunov [RS06]. Moreover, a hash function and a key exchange scheme were proposed based on the hardness of computing isogenies over supersingular curves [CLG09, JF11]. Isogeny-based cryptography is attracting attention partially due to its conjectured post-quantum security.

1.2 Isogeny graphs over RSA moduli

Let p, q be primes and let $N = pq$. In this work we consider computational problems related to elliptic curve isogeny graphs defined over $\mathbb{Z}/N\mathbb{Z}$, where the prime factors p, q of N are unknown. An isogeny graph over $\mathbb{Z}/N\mathbb{Z}$ is defined first by fixing the isogeny graphs over \mathbb{F}_p and \mathbb{F}_q , then taking a graph tensor product; obtaining the j -invariants in the vertices of the graph over $\mathbb{Z}/N\mathbb{Z}$ by the Chinese remainder theorem. Working over the ring $\mathbb{Z}/N\mathbb{Z}$ without the factors of N creates new sources of computational hardness from the isogeny problems. Of course, by assuming the hardness of factorization, we immediately lose the post-quantum privilege of the “traditional” isogeny problems. From now on all the discussions of hardness are with respect to the polynomial time classical algorithms.

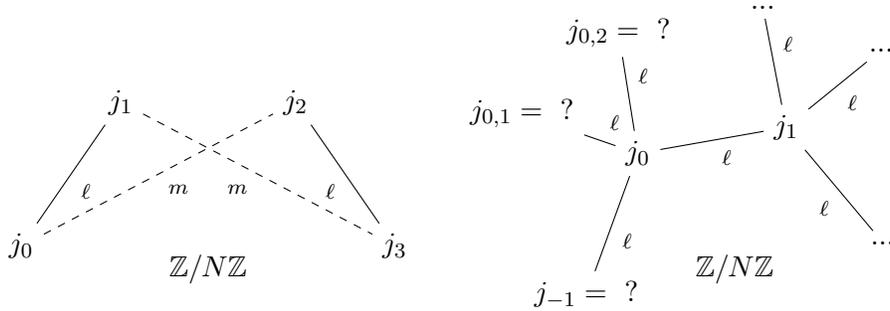


Figure 2: Left: the (ℓ, m) -isogenous neighbor problem where $\gcd(\ell, m) = 1$. Right: the (ℓ, ℓ^2) -isogenous neighbor problem.

Basic neighbor search problem over $\mathbb{Z}/N\mathbb{Z}$. When the factorization of N is unknown, it is not clear how to solve the basic problem of finding (even one of) the ℓ -isogenous neighbors of a given elliptic curve. The two algorithms over finite fields we mentioned seem to fail over $\mathbb{Z}/N\mathbb{Z}$ since both of them require solving polynomials over $\mathbb{Z}/N\mathbb{Z}$, which is hard in general when the factorization of N is unknown. In fact, we show that if it is feasible to find all the ℓ -isogenous neighbors of a given elliptic curve over $\mathbb{Z}/N\mathbb{Z}$, then it is feasible to factorize N .

Joint-neighbor search problem over $\mathbb{Z}/N\mathbb{Z}$. Suppose we are given several j -invariants over $\mathbb{Z}/N\mathbb{Z}$ that are connected by polynomial-degree isogenies, we ask whether it is feasible to compute their joint isogenous neighbors. For example, in the isogeny graph on the LHS of Figure 2, suppose we are given j_0, j_1, j_2 , and the degrees ℓ between j_0 and j_1 , and m between j_0 and j_2 such that $\gcd(\ell, m) = 1$. Then we can find j_3 which is m -isogenous to j_1 and ℓ -isogenous to j_2 , by computing the polynomial $f(x) = \gcd(\Phi_m(j_1, x), \Phi_\ell(j_2, x))$ over $\mathbb{Z}/N\mathbb{Z}$. When $\gcd(\ell, m) = 1$ the polynomial $f(x)$ turns out to be linear with its only root being j_3 , hence computing the (ℓ, m) neighbor in this case is feasible.

However, not all the joint-isogenous neighbors are easy to find. As an example, consider the following (ℓ, ℓ^2) -joint neighbor problem illustrated on the RHS of Figure 2. Suppose we are given j_0 and j_1 that are ℓ -isogenous, and asked to find another j -invariant j_{-1} which is ℓ -isogenous to j_0 and ℓ^2 -isogenous to j_1 . The natural way is to take the gcd of $\Phi_\ell(j_0, x)$ and $\Phi_{\ell^2}(j_1, x)$, but in this case the resulting polynomial is of degree $\ell > 1$ and we are left with the problem of finding a root of it over $\mathbb{Z}/N\mathbb{Z}$, which is believed to be computationally hard without knowing the factors of N .

Currently we do not know if solving this problem is as hard as factoring N . Neither do we know of an efficient algorithm of solving the (ℓ, ℓ^2) -joint neighbor problem. We will list our attempts in solving the (ℓ, ℓ^2) -joint neighbor problem in Section 5.2.

The conjectured computational hardness of the (ℓ, ℓ^2) -joint neighbor problem is fundamental to the infeasibility of inversion in the group we construct.

1.3 Constructing a trapdoor group with infeasible inversion

To explain the construction of the trapdoor group with infeasible inversion (TGII), it is necessary to recall the connection of the ideal class groups and elliptic curve isogenies. Let \mathbf{k} be a finite field as before and let E be an elliptic curve over \mathbf{k} whose endomorphism ring is isomorphic to an

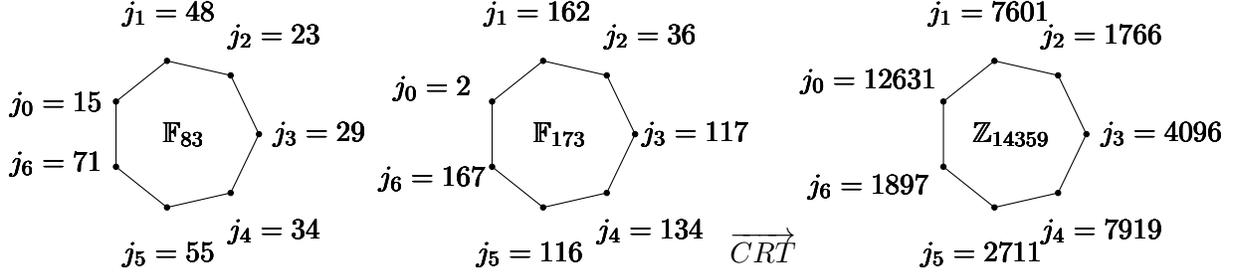


Figure 3: A representation of $\mathcal{CL}(-251)$ by a 3-isogeny volcano over \mathbb{Z}_{14359} of size $h(-251) = 7$. The \mathbb{F}_{83} part is taken from [RS06].

imaginary quadratic order \mathcal{O} . The group of invertible \mathcal{O} -ideals acts on the set of elliptic curves with endomorphism ring \mathcal{O} . The ideal class group $\mathcal{CL}(\mathcal{O})$ acts faithfully and transitively on the set

$$\text{Ell}_{\mathcal{O}}(\mathbf{k}) = \{j(E) : E \text{ with } \text{End}(E) \simeq \mathcal{O}\}.$$

In other words, there is a map

$$\mathcal{CL}(\mathcal{O}) \times \text{Ell}_{\mathcal{O}}(\mathbf{k}) \rightarrow \text{Ell}_{\mathcal{O}}(\mathbf{k}), \quad (\mathfrak{a}, j) \mapsto \mathfrak{a} * j$$

such that $\mathfrak{a} * (\mathfrak{b} * j) = (\mathfrak{a}\mathfrak{b}) * j$ for all $\mathfrak{a}, \mathfrak{b} \in \mathcal{CL}(\mathcal{O})$ and $j \in \text{Ell}_{\mathcal{O}}(\mathbf{k})$; for any $j, j' \in \text{Ell}_{\mathcal{O}}(\mathbf{k})$, there is a unique $\mathfrak{a} \in \mathcal{CL}(\mathcal{O})$ such that $j' = \mathfrak{a} * j$. The cardinality of $\text{Ell}_{\mathcal{O}}(\mathbf{k})$ equals to the class number $h(\mathcal{O})$.

We are now ready to provide an overview of the TGII construction with a toy example in Figure 3.

Parameter generation. To simplify this overview let us assume that the group $\mathcal{CL}(\mathcal{O})$ is cyclic, in which case the group \mathbb{G} with infeasible inversion is exactly $\mathcal{CL}(\mathcal{O})$ (in the detailed construction we usually choose a cyclic subgroup of $\mathcal{CL}(\mathcal{O})$). To generate the public parameter for the group $\mathcal{CL}(\mathcal{O})$, we choose two primes p, q and curves E_{0, \mathbb{F}_p} over \mathbb{F}_p and E_{0, \mathbb{F}_q} over \mathbb{F}_q such that the endomorphism rings of E_{0, \mathbb{F}_p} and E_{0, \mathbb{F}_q} are both isomorphic to \mathcal{O} . Let $N = p \cdot q$. Let E_0 be an elliptic curve over $\mathbb{Z}/N\mathbb{Z}$ as the CRT composition of E_{0, \mathbb{F}_p} and E_{0, \mathbb{F}_q} . The j -invariant of E_0 , denoted as j_0 , equals to the CRT composition of the j -invariants of E_{0, \mathbb{F}_p} and E_{0, \mathbb{F}_q} . The identity of $\mathcal{CL}(\mathcal{O})$ is represented by j_0 . The public parameter of the group is (N, j_0) .

In the example of Figure 3, we set the discriminant D of the imaginary quadratic order \mathcal{O} to be -251 . The group order is then the class number $h(\mathcal{O}) = 7$. Choose $p = 83, q = 173, N = pq = 14359$. Fix a curve E_0 so that $j(E_{0, \mathbb{F}_p}) = 15, j(E_{0, \mathbb{F}_q}) = 2$, then $j_0 = \text{CRT}(83, 173; 15, 2) = 12631$. The public parameter is $(14359, 12631)$.

The encodings. We provide two types of encodings for each group element: the canonical and composable embeddings. The *canonical encoding* of an element is uniquely determined once the public parameter is fixed and it can be directly used in the equivalence test. It, however, does not support efficient group operations. The *composable encoding* of an element, on the other hand, supports efficient group operations with the other composable encodings. Moreover, a composable encoding can be converted to a canonical encoding by an efficient, public extraction algorithm.

An element $x \in \mathcal{CL}(\mathcal{O})$ is canonically represented by the j -invariant of the elliptic curve $x * E_0$ (once again, obtained over \mathbb{F}_p and \mathbb{F}_q then composed by CRT), and we call $j(x * E_0)$ the canonical encoding of x . Note that the canonical encodings of all the elements are fixed once j_0 and N are fixed.

To make things concrete, let $a = \sqrt{-251}$ and consider the toy example above. The ideal class $x = [(3, \frac{a+1}{2})]$ acting on E_0 over \mathbb{F}_p gives $j(x * E_{0, \mathbb{F}_p}) = 48$, over \mathbb{F}_q gives $j(x * E_{0, \mathbb{F}_q}) = 162$. The

canonical encoding of x is then $j_1 = \text{CRT}(83, 173; 48, 162) = 7601$. Similarly, the canonical encodings of the ideal classes $[(7, \frac{a-1}{2})]$, $[(5, \frac{a+7}{2})]$, $[(5, \frac{a+3}{2})]$, $[(7, \frac{a+1}{2})]$, $[(3, \frac{a-1}{2})]$ are 1766, 4096, 7919, 2711, 1897.

The composable encodings and the composition law. To generate a composable encoding of $x \in \mathcal{CL}(\mathcal{O})$, we factorize x as $x = \prod_{x_i \in S} x_i^{e_i}$, where S denotes a generating set, and both the norms $N(x_i)$ and the exponents e_i being polynomial in size. The composable encoding of x then consists of the norms $N(x_i)$ and the j -invariants of $x_i^k * E_0$, for $k \in [e_i]$, for $i \in [|S|]$. The *degree* of a composable encoding is defined to be the product of the norms of the ideals $\prod_{x_i \in S} N(x_i)^{e_i}$. Note that the degree depends on the choice of S and the factorization of x , which is not unique.

As an example let us consider the simplest situation, where the composable encodings are just the canonical encodings themselves together with the norms of the ideals (i.e. the degrees of the isogenies). Set the composable encoding of $x = [(3, \frac{a+1}{2})]$ be $(3, 7601)$, the composable encoding of $y = [(7, \frac{a-1}{2})]$ be $(7, 1766)$.

Let us remark an intricacy of the construction of composable encodings. When the degrees of the composable encodings of x and y are coprime and polynomially large, the composition of x and y can be done simply by concatenating the corresponding encodings. To extract the canonical encoding of $x \circ y$, we take the gcd of the modular polynomials. In the example above, the canonical encoding of $x \circ y$ can be obtained by taking the gcd of $\Phi_7(7601, x)$ and $\Phi_3(1766, x)$ over $\mathbb{Z}/N\mathbb{Z}$. Since the degrees are coprime, the resulting polynomial is linear, with the only root being 4096, which is the canonical encoding of $[(5, \frac{a+7}{2})]$.

Note, however, that if the degrees share prime factors, then the gcd algorithm does not yield a linear polynomial, so the the above algorithm for composition does not go through. To give a concrete example to what this means let us go back to our example: if we represent $y = [(7, \frac{a-1}{2})]$ by first factorizing y as $[(3, \frac{a+1}{2})]^2$ we then get the composable encoding of y as $(3, (7601, 1766))$. In this case the gcd of $\Phi_{32}(7601, x)$ and $\Phi_3(1766, x)$ over $\mathbb{Z}/N\mathbb{Z}$ yields a degree 3 polynomial, where it is unclear how to extract the roots. Hence, in this case we cannot calculate the canonical embedding of $x \circ y$ simply by looking at the gcd.

Therefore, to facilitate the efficient compositions of the encodings of group elements, we will need to represent them as the product of *pairwise co-prime* ideals with polynomially large norms. This, in particular, means the encoding algorithm will need to keep track on the primes used in the degrees of the composable encodings in the system. In other words, the encoding algorithm is *stateful*.

The infeasibility of inversion. The infeasibility of inversion amounts to the hardness of the computation of the canonical embedding of an element $x^{-1} \in \mathbb{G}$ from a composable encoding of x , and it is based on the hardness of the (ℓ, ℓ^2) -isogenous neighbors problem for each ideal of a composable encoding.

Going back to our example, given the composable encoding $(3, 7601)$ of $x = [(3, \frac{a+1}{2})]$, the canonical encoding of $x^{-1} = [(3, \frac{a-1}{2})]$ is a root of $f(x) = \text{gcd}(\Phi_{32}(7601, x), \Phi_3(12631, x))$. The degree of f , however, is 3, so that it is not clear how to extract the root efficiently over an RSA modulus.

The difficulty of sampling the class group invariants and its implications. Let us remark that the actual instantiation of TGII is more involved. A number of challenges arise solely from working with the ideal class groups of imaginary quadratic orders. To give a simple example of the challenges we face, efficiently generating a class group with a known large prime class number is a well-known open problem. Additionally, our construction requires more than the class number (namely, a short basis of the relation lattice of the class group) to support an efficient encoding algorithm.

In our solution, we choose the discriminant D to be of size roughly $\lambda^{O(\log \lambda)}$ and polynomially smooth, so as to make the parameter generation algorithm and the encoding algorithm run in polynomial time. The discriminant D (i.e. the description of the class group $\mathcal{CL}(D)$) has to be hidden to preserve the plausible $\lambda^{O(\log \lambda)}$ -security of the TGII. Furthermore, even if D is hidden, there is an $\lambda^{O(\log \lambda)}$ attack by first guessing D or the group order, then solving the discrete-log problem given the polynomially-smooth group order. Extending the working parameters regime seems to require the solutions of several open problems concerning ideal class groups of imaginary quadratic orders.

Summary of the TGII construction. To summarize, our construction of TGII chooses two sets of j -invariants that correspond to elliptic curves with the same imaginary quadratic order \mathcal{O} over \mathbb{F}_p and \mathbb{F}_q , and glues the j -invariants via the CRT composition as the canonical encodings of the group elements in $\mathcal{CL}(\mathcal{O})$. The composable encoding of a group element x is given as several j -invariants that represent the smooth ideals in a factorization of x . The efficiency of solving the (ℓ, m) -joint-neighbor problem over $\mathbb{Z}/N\mathbb{Z}$ facilitates the efficient group operation over coprime degree encodings. The conjectured hardness of the (ℓ, ℓ^2) -joint-neighbor problem over $\mathbb{Z}/N\mathbb{Z}$ is the main reason behind the hardness of inversion, but it also stops us from composing encodings that share prime factors.

The drawbacks of our construction of TGII are as follows.

1. Composition is only feasible for coprime-degree encodings, which means in order to publish arbitrarily polynomially many encodings, the encoding algorithm has to be stateful in order to choose different polynomially large prime factors for the degrees of the encoding (we cannot choose polynomially large prime degrees and hope they are all different).
2. In the definition from [Hoh03, Mol03], the composable encodings obtained during the composition are required to be indistinguishable to a freshly sampled encoding. In our construction the encodings keep growing during compositions, until they are extracted to the canonical encoding which is a single j -invariant.
3. In addition to the (ℓ, ℓ^2) -joint-neighbor problem, the security of the TGII construction relies on several other heuristic assumptions. We will list our cryptanalytic attempts in §5.3. Moreover, even if we have not missed any attacks, the current best attack only requires $\lambda^{O(\log \lambda)}$ -time, by first guessing the discriminant or the group order.

The two applications of TGII. Let us briefly mention the impact of the limitation of our TGII on the applications of directed transitive signature (DTS) [Hoh03, Mol03] and broadcast encryption [ILOP04]. For the broadcast encryption from TGII [ILOP04], the growth of the composable encodings do not cause a problem. For DTS, in the direct instantiation of DTS from TGII [Hoh03, Mol03], the signature is a composable encoding, so the length of the signature keeps growing during the composition, which is an undesirable feature for a non-trivial DTS. So on top of the basic instantiation, we provide an additional compression technique to shrink the composed signature.

Let us also remark that in the directed transitive signature [Hoh03, Mol03], encodings are sampled by the master signer; in the broadcast encryption scheme [ILOP04], encodings are sampled by the master encrypter. At least for these two applications, having the master signer/encrypter being stateful is not ideal but acceptable.

1.4 Related works

Note that given g and g^a over the ring $(\mathbb{Z}/n\mathbb{Z})^*$, computing g^{-a} is feasible for any n . On the other hand, computing $g^{1/a}$ is infeasible for suitable subgroup \mathbb{G} of $(\mathbb{Z}/n\mathbb{Z})^*$. However, in general, it is not clear how to efficiently perform the multiplicative operation “in the exponent”.

The only existing candidate of (T)GII that supports a large number of group operations is implied by the self-bilinear maps constructed by Yamakawa et al. [YYHK14] using general purpose indistinguishability obfuscation [BGI⁺01]. The existence of iO is currently considered as a strong assumption in the cryptography community. Over the past five years many candidates (since [GGH⁺13]) and attacks (since [CHL⁺15]) were proposed for iO. Basing iO on a clearly stated hard mathematical problem is still an open research area.

Nevertheless, the self-bilinear maps construction from iO is conceptually simple. Here we sketch the idea. Given an integer N with unknown factorization, a group element $a \in (\mathbb{Z}/\frac{\phi(N)}{4}\mathbb{Z})^*$ is represented by $g^a \in QR^+(N)$ (QR^+ denotes the signed group of quadratic residues), together with an obfuscation of the circuit $C_{2a,N}$:

$$C_{2a,N} : QR^+(N) \rightarrow QR^+(N), \quad x \mapsto x^{2a}.$$

Given g^a , $\text{Obf}(C_{2a,N})$, g^b , $\text{Obf}(C_{2b,N})$, everyone is able to compute g^{2ab} . [YYHK14] proves that under the hardness of factoring and assuming that the obfuscator satisfy the security of indistinguishable obfuscation, it is infeasible for the adversary to compute g^{ab} . Such a result implies that under the same assumption, it is infeasible to compute $g^{1/x}$ given g^x and $\text{Obf}(C_{2x,N})$.

A downside of the self-bilinear maps of [YYHK14] is that the obfuscated circuits, referred to as “auxiliary inputs”, keep growing after the compositions. Self-bilinear maps without auxiliary input is recently investigated by [YYHK18] in the context of rings with infeasible inversion, but constructing such a primitive is open even assuming iO.

2 Preliminaries

Notation and terminology. Let $\mathbb{C}, \mathbb{R}, \mathbb{Q}, \mathbb{Z}, \mathbb{N}$ denote the set of complex numbers, reals, rationals, integers, and positive integers respectively. For any field K we fix an algebraic closure and denote it by \bar{K} . For $n \in \mathbb{N}$, let $[n] := \{1, \dots, n\}$. For $B \in \mathbb{R}$, an integer n is called B -smooth if all the prime factors of n are less than or equal to B . An n -dimensional vector is written as a bold lower-case letter, e.g. $\mathbf{v} := (v_1, \dots, v_n)$. For an index $k \in \mathbb{N}$, distinct prime numbers p_i for $i \in [k]$, and $c_i \in \mathbb{Z}/p_i\mathbb{Z}$ we will let $\text{CRT}(p_1, \dots, p_k; c_1, \dots, c_k)$ to denote the unique $y \in \mathbb{Z}/(\prod_i^k p_i)\mathbb{Z}$ such that $y \equiv c_i \pmod{p_i}$, for $i \in [k]$.

In cryptography, the security parameter (denoted by λ) is a variable that is used to parameterize the computational complexity of the cryptographic algorithm or protocol, and the adversary’s probability of breaking security. In theory and by default, an algorithm is called “efficient” if it runs in probabilistic polynomial time over λ . Exceptions may occur in reality and we will explicitly discuss them when they come up in our applications.

An n -dimensional lattice Λ is a discrete additive subgroup of \mathbb{R}^n that generate it as a vector space over \mathbb{R} . Given n linearly independent vectors $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n \in \mathbb{R}^n\}$, the lattice generated by \mathbf{B} is

$$\Lambda(\mathbf{B}) = \Lambda(\mathbf{b}_1, \dots, \mathbf{b}_n) = \left\{ \sum_{i=1}^n x_i \cdot \mathbf{b}_i, x_i \in \mathbb{Z} \right\}.$$

We will denote the Gram-Schmidt orthogonalization of \mathbf{B} by $\tilde{\mathbf{B}}$.

Let \mathbb{G} denote a finite abelian group. We will denote the prime factorization of its order $|\mathbb{G}|$ by $|\mathbb{G}| = \prod_{i \in [k]} p_i^{w(p_i)}$. For each p_i , we set $H(p_i) := |\mathbb{G}|/p_i^{w(p_i)}$, and $G(p_i) := \{g^{H(p_i)}, g \in \mathbb{G}\}$. Note that with this notation we have the isomorphism

$$\mathbb{G} \rightarrow \mathbb{G}(p_1) \times \dots \times \mathbb{G}(p_k), \quad g \mapsto (g^{H(p_1)}, \dots, g^{H(p_k)}).$$

For a cyclic group \mathbb{G} , the *discrete-log problem* asks to find the exponent $a \in [|\mathbb{G}|]$ given a generator g and a group element $x = g^a \in \mathbb{G}$. The Pohlig-Hellman algorithm [PH78] solves the discrete-log problem in time $O(\sum_i w(p_i)(\log |\mathbb{G}| + \sqrt{p_i}))$ if the factorization of $|\mathbb{G}|$ is known.

Over a possibly non-cyclic group \mathbb{G} , the discrete-log problem is defined as follows: given a set of elements g_1, \dots, g_k and a group element $x \in \mathbb{G}$, output a vector $\mathbf{e} \in \mathbb{Z}^k$ such that $x = \prod_{i=1}^k g_i^{e_i}$, or decide that x is not in the subgroup generated by $\{g_1, \dots, g_k\}$. A generalization of the Pohlig-Hellman algorithm works for non-cyclic groups with essentially the same cost plus an $O(\log |\mathbb{G}|)$ factor (the algorithm is folklore [PH78] and is explicitly given in [Tes99]). A further improvement removing the $\log |\mathbb{G}|$ factor is given by Sutherland [Sut11b].

2.1 Ideal class groups of imaginary quadratic orders

There are two equivalent ways of describing ideal class groups of imaginary quadratic orders: via the theory of ideals or quadratic forms. We will be using these two view points interchangeably. The main references for these are [McC88, Coh95, Cox11].

Let K be an imaginary quadratic field. An *order* \mathcal{O} in K is a subset of K such that

1. \mathcal{O} is a subring of K containing 1,
2. \mathcal{O} is a finitely generated \mathbb{Z} -module,
3. \mathcal{O} contains a \mathbb{Q} -basis of K .

The ring \mathcal{O}_K of integers of K is always an order. For any order \mathcal{O} , we have $\mathcal{O} \subseteq \mathcal{O}_K$, in other words \mathcal{O}_K is the maximal order of K with respect to inclusion.

The ideal class group (or class group) of \mathcal{O} is the quotient group $\mathcal{CL}(\mathcal{O}) = I(\mathcal{O})/P(\mathcal{O})$ where $I(\mathcal{O})$ denotes the group of proper (i.e. invertible) fractional \mathcal{O} -ideals of, and $P(\mathcal{O})$ is its subgroup of principal \mathcal{O} -ideals. Let D be the discriminant of \mathcal{O} . Note that since \mathcal{O} is quadratic imaginary we have $D < 0$. Sometimes we will denote the class group $\mathcal{CL}(\mathcal{O})$ as $\mathcal{CL}(D)$, and the class number (the group order of $\mathcal{CL}(\mathcal{O})$) as $h(\mathcal{O})$ or $h(D)$.

Let $D = D_0 \cdot f^2$, where D_0 is the *fundamental discriminant* and f is the *conductor* of \mathcal{O} (or D). The following well-known formula relates the class number of a non-maximal order to that of the maximal one:

$$\frac{h(D)}{w(D)} = \frac{h(D_0)}{w(D_0)} \cdot f \prod_{p|f} \left(1 - \frac{\left(\frac{D_0}{p}\right)}{p}\right), \quad (1)$$

where $w(D) = 6$ if $D = -3$, $w(D) = 4$ if $D = -4$, and $w(D) = 2$ if $D < -4$. Let us also remark that the Brauer-Siegel theorem implies that $\ln(h(D)) \sim \ln(\sqrt{|D|})$ as $D \rightarrow -\infty$.

Representations. An \mathcal{O} -ideal of discriminant D can be represented by its generators, or by its binary quadratic forms. A binary quadratic form of discriminant D is a polynomial $ax^2 + bxy + cy^2$ with $b^2 - 4ac = D$. We denote a binary quadratic form by (a, b, c) . The group $SL_2(\mathbb{Z})$ acts on the set of binary quadratic forms and preserves the discriminant. We shall always be assuming that our forms are positive definite, i.e. $a > 0$. Recall that a form (a, b, c) is called *primitive* if $\gcd(a, b, c) = 1$, and a primitive form is called *reduced* if $-a < b \leq a < c$ or $0 \leq b \leq a = c$. Reduced forms satisfy $a \leq \sqrt{|D|/3}$.

A fundamental fact, which goes back to Gauss, is that in each equivalence class, there is a unique reduced form (see Corollary 5.2.6 of [Coh95]). Given a form (a, b, c) , denote $[(a, b, c)]$ as its equivalence class. Note that when D is fixed, we can denote a class simply by $[(a, b, \cdot)]$. Efficient algorithms of composing forms and computing the reduced form can be found in [McC88, Page 9].

2.2 Elliptic curves and their isogenies

In this section we will recall some background on elliptic curves and isogenies. All of this material is well-known and the main references for this section are [Koh96, Sil09, Sil13, Sut13a, Feo17].

Let E be an elliptic curve defined over a finite field \mathbf{k} of characteristic $\neq 2, 3$ with q elements, given by its Weierstrass form $y^2 = x^3 + ax + b$ where $a, b \in \mathbf{k}$. By the Hasse bound we know that the order of the \mathbf{k} -rational points $E(\mathbf{k})$ satisfies

$$-2\sqrt{q} \leq \#E(\mathbf{k}) - (q + 1) \leq 2\sqrt{q}.$$

Here, $t = q + 1 - \#E(\mathbf{k})$ is the trace of Frobenius endomorphism $\pi : (x, y) \mapsto (x^q, y^q)$. Let us also recall that Schoof's algorithm [Sch85] takes as inputs E and q , and computes t , and hence $\#E(\mathbf{k})$, in time $\text{poly}(\log q)$.

The j -invariant of E is defined as

$$j(E) = 1728 \cdot \frac{4a^3}{4a^3 + 27b^2}.$$

The values $j = 0$ or 1728 are special and we will choose to avoid these two values throughout the paper. Two elliptic curves are isomorphic over the algebraic closure $\bar{\mathbf{k}}$ if and only if their j -invariants are the same. Note that this isomorphism may not be defined over the base field \mathbf{k} , in which case the curves are called twists of each other. It will be convenient for us to use j -invariants to represent isomorphism classes of elliptic curves (including their twists). In many cases, with abuse of notation, a j -invariant will be treated as the same to an elliptic curve over \mathbf{k} in the corresponding isomorphism class.

Isogenies. An *isogeny* $\varphi : E_1 \rightarrow E_2$ is a morphism of elliptic curves that preserves the identity. Every nonzero isogeny induces a surjective group homomorphism from $E_1(\bar{\mathbf{k}})$ to $E_2(\bar{\mathbf{k}})$ with a finite kernel. Elliptic curves related by a nonzero isogeny are said to be isogenous. By the Tate isogeny theorem [Tat66, pg.139] two elliptic curves E_1 and E_2 are isogenous over \mathbf{k} if and only if $\#E_1(\mathbf{k}) = \#E_2(\mathbf{k})$.

The degree of an isogeny is its degree as a rational map. An isogeny of degree ℓ is called an ℓ -isogeny. When $\text{char}(\mathbf{k}) \nmid \ell$, the kernel of an ℓ -isogeny has cardinality ℓ . Two isogenies ϕ and φ are considered equivalent if $\phi = \iota_1 \circ \varphi \circ \iota_2$ for isomorphisms ι_1 and ι_2 . Every ℓ -isogeny $\varphi : E_1 \rightarrow E_2$ has a unique dual

isogeny $\hat{\varphi} : E_2 \rightarrow E_1$ of the same degree such that $\varphi \circ \hat{\varphi} = \hat{\varphi} \circ \varphi = [\ell]$, where $[\ell]$ is the multiplication by ℓ map. The kernel of the multiplication-by- ℓ map is the ℓ -torsion subgroup

$$E[\ell] = \{P \in E(\bar{\mathbf{k}}) : \ell P = 0\}.$$

When $\ell \nmid \text{char}(\mathbf{k})$ we have $E[\ell] \simeq \mathbb{Z}/\ell\mathbb{Z} \times \mathbb{Z}/\ell\mathbb{Z}$. For a prime $\ell \neq \text{char}(\mathbf{k})$, there are $\ell + 1$ cyclic subgroups in $E[\ell]$ of order ℓ , each corresponding to the kernel of an ℓ -isogeny φ from E . An isogeny from E is defined over \mathbf{k} if and only if its kernel subgroup G is defined over \mathbf{k} (namely, for $P \in G$ and $\sigma \in \text{Gal}(\bar{\mathbf{k}}/\mathbf{k})$, $\sigma(P) \in G$; note that this does not imply $G \subseteq E(\mathbf{k})$). If $\ell \nmid \text{char}(\mathbf{k})$ and $j(E) \neq 0$ or 1728, then up to isomorphism the number of ℓ -isogenies from E defined over \mathbf{k} is 0, 1, 2, or $\ell + 1$.

Modular polynomials. Let $\ell \in \mathbb{Z}$, let \mathbb{H} denote the upper half plane $\mathbb{H} := \{\tau \in \mathbb{C} : \text{im } \tau > 0\}$ and $\mathbb{H}^* = \mathbb{H} \cup \mathbb{Q} \cup \{\infty\}$. Let $j(\tau)$ be the classical modular function defined on \mathbb{H} . For any $\tau \in \mathbb{H}$, the complex numbers $j(\tau)$ and $j(\ell\tau)$ are the j -invariants of elliptic curves defined over \mathbb{C} that are related by an isogeny whose kernel is a cyclic group of order ℓ . The minimal polynomial $\Phi_\ell(y)$ of the function $j(\ell z)$ over the field $\mathbb{C}(j(z))$ has coefficients that are polynomials in $j(z)$ with integer coefficients. Replacing $j(z)$ with a variable x gives the *modular polynomial* $\Phi_\ell(x, y) \in \mathbb{Z}[x, y]$, which is symmetric in x and y . It parameterizes pairs of elliptic curves over \mathbb{C} related by a cyclic ℓ -isogeny (an isogeny is said to be cyclic if its kernel is a cyclic group; when ℓ is a prime every ℓ -isogeny is cyclic). The modular equation $\Phi_\ell(x, y) = 0$ is a canonical equation for the modular curve $Y_0(\ell) = \mathbb{H}/\Gamma_0(\ell)$, where $\Gamma_0(\ell)$ is the congruence subgroup of $\text{SL}_2(\mathbb{Z})$ defined by

$$\Gamma_0(\ell) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \text{SL}_2(\mathbb{Z}) \mid \begin{pmatrix} a & b \\ c & d \end{pmatrix} \equiv \begin{pmatrix} * & * \\ 0 & * \end{pmatrix} \pmod{\ell} \right\}.$$

The time and space required for computing the modular polynomial Φ_ℓ are polynomial in ℓ , cf. [E⁺98, § 3] or [Coh95, Page 386]. In this article we will only use $\{\Phi_\ell \in \mathbb{Z}[x, y]\}_{\ell \in \text{poly}(\lambda)}$, so we might as well assume that the modular polynomials are computed ahead of time². In reality the coefficients of Φ_ℓ over $\mathbb{Z}[x, y]$ grow significantly with ℓ , so computing Φ_ℓ over $\mathbf{k}[x, y]$ directly is preferable using the improved algorithms of [CL05, BLS12], or even $\Phi_\ell(j_1, y)$ over $\mathbf{k}[y]$ using [Sut13b].

2.3 Isogeny volcanoes and the class groups

An isogeny from an elliptic curve E to itself is called an *endomorphism*. Over a finite field \mathbf{k} , $\text{End}(E)$ is isomorphic to an imaginary quadratic order when E is ordinary, or an order in a definite quaternion algebra when E is supersingular. In this paper we will be focusing on the ordinary case.

Isogeny graphs. These are graphs capturing the relation of being ℓ -isogenous among elliptic curves over a finite field \mathbf{k} .

Definition 2.1 (ℓ -isogeny graph). *Fix a prime ℓ and a finite field \mathbf{k} such that $\text{char}(\mathbf{k}) \neq \ell$. The ℓ -isogeny graph $G_\ell(\mathbf{k})$ has vertex set \mathbf{k} . Two vertices (j_1, j_2) have a directed edge (from j_1 to j_2) with multiplicity equal to the multiplicity of j_2 as a root of $\Phi_\ell(j_1, Y)$. The vertices of $G_\ell(\mathbf{k})$ are j -invariants and each edge corresponds to an (isomorphism class of an) ℓ -isogeny.*

²The modular polynomials Φ_ℓ for $1 \leq \ell \leq 300$ are available at <https://math.mit.edu/~drew/ClassicalModPolys.html>.

For $j_1, j_2 \notin \{0, 1728\}$, an edge (j_1, j_2) occurs with the same multiplicity as (j_2, j_1) and thus the subgraph of $G_\ell(\mathbf{k})$ on $\mathbf{k} \setminus \{0, 1728\}$ can be viewed as an undirected graph. Every curve in the isogeny class of a supersingular curve is supersingular. Accordingly, $G_\ell(\mathbf{k})$ has supersingular and ordinary components. The ordinary components of $G_\ell(\mathbf{k})$ look like ℓ -volcanoes:

Definition 2.2 (ℓ -volcano). *Fix a prime ℓ . An ℓ -volcano V is a connected undirected graph whose vertices are partitioned into one or more levels V_0, \dots, V_d such that the following hold:*

1. *The subgraph on V_0 (the surface, or the crater) is a regular graph of degree at most 2.*
2. *For $i > 0$, each vertex in V_i has exactly one neighbor in level V_{i-1} .*
3. *For $i < d$, each vertex in V_i has degree $\ell + 1$.*

Let $\phi : E_1 \rightarrow E_2$ by an ℓ -isogeny of elliptic curves with endomorphism rings $\mathcal{O}_1 = \text{End}(E_1)$ and $\mathcal{O}_2 = \text{End}(E_2)$ respectively. Then, there are three possibilities for \mathcal{O}_1 and \mathcal{O}_2 :

- If $\mathcal{O}_1 = \mathcal{O}_2$, then ϕ is called horizontal,
- If $[\mathcal{O}_1 : \mathcal{O}_2] = \ell$, then ϕ is called descending,
- If $[\mathcal{O}_2 : \mathcal{O}_1] = \ell$, then ϕ is called ascending.

Let E be an elliptic curve over \mathbf{k} whose endomorphism ring is isomorphic to an imaginary quadratic order \mathcal{O} . Then, the set

$$\text{Ell}_{\mathcal{O}}(\mathbf{k}) = \{j(E) \in \mathbf{k} \mid \text{with } \text{End}(E) \simeq \mathcal{O}\}$$

is naturally a $\mathcal{C}\mathcal{L}(\mathcal{O})$ -torsor as follows: For an invertible \mathcal{O} -ideal \mathfrak{a} the \mathfrak{a} -torsion subgroup

$$E[\mathfrak{a}] = \{P \in E(\bar{\mathbf{k}}) : \alpha(P) = 0, \forall \alpha \in \mathfrak{a}\}$$

is the kernel of a separable isogeny $\phi_{\mathfrak{a}} : E \rightarrow E'$. If the norm $N(\mathfrak{a}) = [\mathcal{O} : \mathfrak{a}]$ is not divisible by $\text{char}(\mathbf{k})$, then the degree of $\phi_{\mathfrak{a}}$ is $N(\mathfrak{a})$. Moreover, if \mathfrak{a} and \mathfrak{b} are two invertible \mathcal{O} -ideals, then $\phi_{\mathfrak{a}\mathfrak{b}} = \phi_{\mathfrak{a}}\phi_{\mathfrak{b}}$, and if \mathfrak{a} is principal then $\phi_{\mathfrak{a}}$ is an isomorphism. This gives a faithful and transitive action of $\mathcal{C}\mathcal{L}(\mathcal{O})$ on $\text{Ell}_{\mathcal{O}}(\mathbf{k})$.

Every horizontal ℓ -isogeny arises this way from the action of an invertible \mathcal{O} -ideal \mathfrak{l} of norm ℓ . Let K denote the fraction field of \mathcal{O} and \mathcal{O}_K be its ring of integers. If $\ell \mid [\mathcal{O}_K : \mathcal{O}]$ then no such ideal exists. Otherwise, \mathcal{O} is said to be maximal at ℓ and there are $1 + \left(\frac{D}{\ell}\right)$ horizontal ℓ -isogenies.

Remark 2.3 (Linking ideals and horizontal isogenies). *When ℓ splits in \mathcal{O} we have $(\ell) = \mathfrak{l} \cdot \bar{\mathfrak{l}}$. Fix an elliptic curve $E(\mathbf{k})$ with $\text{End}(E) \simeq \mathcal{O}$, the two horizontal isogenies $\phi_1 : E \rightarrow E_1$ and $\phi_2 : E \rightarrow E_2$ can be efficiently associated with the two ideals \mathfrak{l} and $\bar{\mathfrak{l}}$ when $\ell \in \text{poly}(\lambda)$ (cf. [Sch95]). To do so, factorize the characteristic polynomial of Frobenius π as $(x - \mu)(x - \nu) \pmod{\ell}$, where $\mu, \nu \in \mathbb{Z}/\ell\mathbb{Z}$. Given an ℓ -isogeny ϕ from E to E/G , the eigenvalue (say μ) corresponding to the eigenspace G can be verified by picking a point $P \in G$, then check whether $\pi(P) = [\mu]P$ modulo G . If so then μ corresponds to ϕ .*

The following fundamental result of Kohel summarizes the above discussion and more.

Lemma 2.4 ([Koh96]). *Let ℓ be a prime. Let V be an ordinary component of $G_\ell(\mathbb{F}_q)$ that does not contain 0 or 1728. Then V is an ℓ -volcano for which the following hold:*

1. *The vertices in level V_i all have the same endomorphism ring \mathcal{O}_i .*

2. The subgraph on V_0 has degree $1 + (\frac{D_0}{\ell})$, where $D_0 = \text{disc}(\mathcal{O}_0)$.
3. If $(\frac{D_0}{\ell}) \geq 0$, then $|V_0|$ is the order of $[\mathfrak{l}]$ in $\mathcal{CL}(\mathcal{O}_0)$; otherwise $|V_0| = 1$.
4. The depth of V is d , where $2d$ is the largest power of ℓ dividing $(t^2 - 4q)/D_0$, and $t^2 = \text{tr}(\pi_E)^2$ for $j(E) \in V$.
5. $\ell \nmid [\mathcal{O}_K : \mathcal{O}_0]$ and $[\mathcal{O}_i : \mathcal{O}_{i+1}] = \ell$ for $0 \leq i < d$.

Let $\mathcal{G}_{\mathcal{O},m}(\mathbf{k})$ be the regular graph whose vertices are the elements of $\text{Ell}_{\mathcal{O}}(\mathbf{k})$, and whose edges are the equivalence classes of horizontal isogenies defined over \mathbf{k} of prime degrees $\leq m$. The following result states that assuming GRH $\mathcal{G}_{\mathcal{O},m}(\mathbf{k})$ is an expander graph.

Lemma 2.5 ([JMV05, BGK⁺18]). *Let $q = \#\mathbf{k}$, \mathcal{O} be an imaginary quadratic order of discriminant D , and ϵ be a fixed constant. Let m be such that $m \geq (\log q)^{2+\epsilon}$. Assuming GRH, a random walk on $\mathcal{G}_{\mathcal{O},m}(\mathbf{k})$ will reach a subset of size S with probability at least $\frac{S}{2|\mathcal{G}_{\mathcal{O},m}(\mathbf{k})|}$ after $\text{polylog}(q)$ many steps. Furthermore, for a suitable constant C and any $\delta > 0$, assuming GRH, the distribution of a vertice obtained from a random walk on $\mathcal{G}_{\mathcal{O},m}(\mathbf{k})$ of length $C \cdot \frac{\delta + \log h(D)}{\epsilon \log \log |D|}$ is $e^{-\delta}$ -statistically close to uniform.*

More about the endomorphism ring from a computational perspective. Given an ordinary curve E over \mathbf{k} , its endomorphism ring \mathcal{O} can be determined by first computing the trace t of Frobenius endomorphism π , then computing $t^2 - 4q = v^2 D_0$, where $v^2 D_0$ is the discriminant of $\mathbb{Z}[\pi]$, $\mathbb{Z}[\pi] \subseteq \mathcal{O} \subseteq \mathcal{O}_K$, and $K = \mathbb{Q}(\sqrt{D_0})$. The discriminant of \mathcal{O} is then $u^2 D_0$ for some $u \mid v$. When v has only few small factors, determining the endomorphism ring can be done in time polynomial in $\log(q)$ [Koh96]. In general it can take up to subexponential time in $\log(q)$ under GRH [BS11, Bis11].

Let \mathcal{O} be an imaginary quadratic order of discriminant D . Let $H_D(x)$ be the Hilbert class polynomial defined by

$$H_D(x) = \prod_{j(E) \in \text{Ell}_{\mathcal{O}}(\mathbb{C})} (x - j(E)).$$

H_D has integer coefficients and is of degree $h(D)$. Furthermore, it takes $O(|D|^{1+\epsilon})$ bits of storage. Under GRH, computing $H_D \bmod q$ takes $O(|D|^{1+\epsilon})$ time and $O(|D|^{1/2+\epsilon} \log q)$ space [Sut11a]. In reality H_D is only feasible for small $|D|$ since it takes a solid amount of space to store H_D . Over $\mathbb{Z}[x]$, [Sut11a] is able to compute H_D for $|D| \approx 10^{13}$ and $h(D) \approx 10^6$. Over $\mathbb{F}_q[x]$, [Sut12b] is able to compute H_D for $|D| \approx 10^{16}$ with $q \approx 2^{256}$.

3 Isogeny graphs over composite moduli

Let p, q be distinct primes and set $N = pq$. We will be using elliptic curves over the ring $\mathbb{Z}/N\mathbb{Z}$. We will not be needing a formal treatment of elliptic curves over rings as such a discussion would take us too far afield. Instead, we will be defining objects and quantities over $\mathbb{Z}/N\mathbb{Z}$ by taking the CRT of the corresponding ones over \mathbb{F}_p and \mathbb{F}_q , which will suffice for our purposes. This follows the treatment given in [Len87].

Since the underlying rings will matter, we will denote an elliptic curve over a ring R by $E(R)$. If R is clear from the context we shall omit it from the notation. To begin, let us remark that the number of points $\#(E(\mathbb{Z}/N\mathbb{Z}))$ is equal to $\#(E(\mathbb{F}_p)) \cdot \#(E(\mathbb{F}_q))$, and the j -invariant of $E(\mathbb{Z}/N\mathbb{Z})$ is $\text{CRT}(p, q; j(E(\mathbb{F}_p)), j(E(\mathbb{F}_q)))$.

3.1 Isogeny graphs over $\mathbb{Z}/N\mathbb{Z}$

Let N be as above. For every prime $\ell \nmid N$ the isogeny graph $G_\ell(\mathbb{Z}/N\mathbb{Z})$ can be defined naturally as the graph tensor product of $G_\ell(\mathbb{F}_p)$ and $G_\ell(\mathbb{F}_q)$.

Definition 3.1 (ℓ -isogeny graph over $\mathbb{Z}/N\mathbb{Z}$). *Let ℓ , p , and q be distinct primes and let $N = pq$. The ℓ -isogeny graph $G_\ell(\mathbb{Z}/N\mathbb{Z})$ has*

- The vertex set of $G_\ell(\mathbb{Z}/N\mathbb{Z})$ is $\mathbb{Z}/N\mathbb{Z}$, identified with $\mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$ by CRT,
- Two vertices $v_1 = (v_{1,p}, v_{1,q})$ and $v_2 = (v_{2,p}, v_{2,q})$ are connected if and only if $v_{1,p}$ is connected to $v_{2,p}$ in $G_\ell(\mathbb{F}_p)$ and $v_{1,q}$ is connected to $v_{2,q}$ in $G_\ell(\mathbb{F}_q)$.

Let us make a remark for future consideration. In the construction of groups with infeasible inversion, we will be working with special subgraphs of $G_\ell(\mathbb{Z}/N\mathbb{Z})$, where the vertices over \mathbb{F}_p and \mathbb{F}_q correspond to j -invariants of curves whose endomorphism rings are the same imaginary quadratic order \mathcal{O} . Nevertheless, this is a choice we made for convenience, and it does not hurt to define the computational problems over the largest possible graph and to study them first.

3.2 The ℓ -isogenous neighbors problem over $\mathbb{Z}/N\mathbb{Z}$

Definition 3.2 (The ℓ -isogenous neighbors problem). *Let p, q be two distinct primes and let $N = pq$. Let ℓ be a polynomially large prime s.t. $\gcd(\ell, N) = 1$. The input of the ℓ -isogenous neighbor problem is N and an integer $j \in \mathbb{Z}/N\mathbb{Z}$ such that there exists (possibly more than) one integer j' that $\Phi_\ell(j, j') = 0$ over $\mathbb{Z}/N\mathbb{Z}$. The problem asks to find such integer(s) j' .*

The following theorem shows that the problem of finding *all* of the ℓ -isogenous neighbors is at least as hard as factoring N .

Theorem 3.3. *If there is a probabilistic polynomial time algorithm that finds all the ℓ -isogenous neighbors in Problem 3.2, then there is a probabilistic polynomial time algorithm that solves the integer factorization problem.*

The idea behind the reduction is as follows. Suppose it is efficient to pick an integer j over $\mathbb{Z}/N\mathbb{Z}$, let $j_p = j \pmod{p}$ and $j_q = j \pmod{q}$, such that j_p has at least two distinct neighbors in $G_\ell(\mathbb{F}_p)$, and j_q has at least two distinct neighbors in $G_\ell(\mathbb{F}_q)$. In this case if we are able to find *all* the integer solutions $j' \in \mathbb{Z}/N\mathbb{Z}$ such that $\Phi_\ell(j, j') = 0$ over $\mathbb{Z}/N\mathbb{Z}$, then there exist two distinct integers j'_1 and j'_2 among the solutions such that $N > \gcd(j'_1 - j'_2, N) > 1$. One can also show that finding *one* of the integer solutions is hard using a probabilistic argument, assuming the underlying algorithm outputs a random solution when there are multiple ones.

In the reduction we pick the elliptic curve E randomly, so we have to make sure that for a non-negligible fraction of the elliptic curves E over \mathbb{F}_p , $j(E) \in G_\ell(\mathbb{F}_p)$ has at least two neighbors. The estimate for this relies on the following lemma:

Lemma 3.4 ([Len87] (1.9)). *There exists an efficiently computable positive constant c such that for each prime number $p > 3$, for a set of integers $S \subseteq \{s \in \mathbb{Z} \mid |p + 1 - s| < \sqrt{p}\}$, we have*

$$\#\{E \mid E \text{ is an elliptic curve over } \mathbb{F}_p, \#E(\mathbb{F}_p) \in S\}_{/\simeq_{\mathbb{F}_p}} \geq c(\#S - 2) \frac{\sqrt{p}}{\log p}.$$

where $\#\{E\}_{/\simeq_{\mathbb{F}_p}}$ denotes the number of isomorphism classes of elliptic curves over \mathbb{F}_p , each counted with weight $(\#\text{Aut}E)^{-1}$.

Theorem 3.5. *Let p, ℓ be primes such that $6\ell < \sqrt{p}$. The probability that for a random elliptic curve E over \mathbb{F}_p (i.e. a random pair $(a, b) \in \mathbb{F}_p \times \mathbb{F}_p$ such that $4a^3 + 27b^2 \neq 0$) $j(E) \in G_\ell(\mathbb{F}_p)$ having at least two neighbors is $\Omega(\frac{1}{\log p})$.*

Proof of Theorem 3.5. We first give a lower bound on the number of ordinary elliptic curves over \mathbb{F}_p whose endomorphism ring has discriminant D such that $(\frac{D}{\ell}) = 1$. If for some pair of (ℓ, p) there are not enough elliptic curves over \mathbb{F}_p with two horizontal ℓ -isogenies then we count the elliptic curves with vertical ℓ -isogenies.

We start with estimating the portion of $t \in [\ell]$ that satisfies $(\frac{t^2-4p}{\ell}) = 1$:

$$\Pr_{t \in [\ell]} \left[\left(\frac{t^2 - 4p}{\ell} \right) = 1 \right] = \begin{cases} 0 & \ell = 2 \\ \frac{1}{2} - \frac{3}{2\ell} & \ell > 2 \text{ and } (\frac{4p}{\ell}) = 1 \\ \frac{1}{2} - \frac{1}{2\ell} & \ell > 2 \text{ and } (\frac{4p}{\ell}) = -1 \end{cases} \quad (2)$$

where the last two equations follows the identity³ $\sum_{t=1}^{\ell} (\frac{t^2-4p}{\ell}) = -1$. Hence for $\ell \geq 5$ or $\ell = 3$ and $(\frac{4p}{3}) = (\frac{p}{3}) = -1$, no less than $\frac{1}{2} - \frac{3}{2\ell}$ of the $t \in [\ell]$ satisfy $(\frac{t^2-4p}{\ell}) = 1$.

We now estimate the number of elliptic curves over \mathbb{F}_p whose discriminant of the endomorphism ring D satisfies $(\frac{D}{\ell}) = 1$. To do so we set $r = \lfloor \sqrt{p}/\ell \rfloor$, and use Lemma 3.4 by choosing the set S as

$$S = \left\{ s \mid \left(\frac{(p+1-s)^2 - 4p}{\ell} \right) = 1, s \in \{(p+1) - r \cdot \ell, \dots, (p+1) + r \cdot \ell\} \setminus \{p+1\} \right\}.$$

Note that $\#S \geq r(\ell - 3)$. Therefore, Lemma 3.4, there exists an effectively computable constant c such that the number of isomorphism classes of elliptic curves over \mathbb{F}_p with the number of points in the set S is greater or equal to

$$c \cdot (r(\ell - 3) - 2) \cdot \frac{\sqrt{p}}{\log p} \geq c \cdot ((\frac{\sqrt{p}}{\ell} - 1)(\ell - 3) - 2) \cdot \frac{\sqrt{p}}{\log p} > c \cdot (\frac{\sqrt{p}}{2\ell} \cdot \frac{\ell}{3} - 2) \cdot \frac{\sqrt{p}}{\log p} \geq c \cdot \frac{p}{18 \log p}. \quad (3)$$

Since the total number of elliptic curves over \mathbb{F}_p is $p^2 - p$; the number of elliptic curves isomorphic to a given elliptic curve E is $\frac{(p-1)}{\#\text{Aut } E}$ [Len87, (1.4)]. So for $\ell \geq 5$ or $\ell = 3$ and $(\frac{4p}{3}) = -1$, the ratio of elliptic curves over \mathbb{F}_p with discriminant D such that $(\frac{D}{\ell}) = 1$ is $\Omega(\frac{1}{\log p})$.

To finish the treatment of the case, where $\ell \geq 5$, or $\ell = 3$ and $(\frac{4p}{3}) = (\frac{p}{3}) = -1$, we will show that among such curves the proportion of the j -invariants $j(E)$ on the crater of the volcano having one or two neighbors is $o(\frac{1}{\log p})$. Recall that we are in the case $\ell \nmid D$ and $\ell = \mathfrak{l}_1 \mathfrak{l}_2$ in $\mathbb{Q}(\sqrt{D})$, and that the crater has size equals to the order of \mathfrak{l}_1 (which is the same as the order of \mathfrak{l}_2) in $\mathcal{CL}(\mathcal{O})$.

If the crater has 1 or 2 vertices, \mathfrak{l}_1 must have order dividing 2 in $\mathcal{CL}(\mathcal{O})$. If \mathfrak{l}_1 has order 1 in $\mathcal{CL}(\mathcal{O})$ then we have $x^2 - Dy^2 = \ell$ for some $x, y \in \mathbb{Z}$. Since ℓ is prime we necessarily have $y \neq 0$. Moreover, since $6\ell < \sqrt{p}$ we have $-D < \sqrt{p}$ and therefore $4p - \sqrt{p} < t^2$. On the other hand, by the Hasse bound we have $t^2 \leq 4p$, hence $4p - \sqrt{p} < t^2 \leq 4p$. Therefore, there are at most $O(p^{\frac{1}{4}})$ j -invariants for which the the top of the volcano consists of a single vertex. This handles the case of \mathfrak{l}_1 having order 1.

The remaining case of \mathfrak{l}_1 having order 2, on the other hand, cannot happen because of genus theory. More precisely, let D_0 be the discriminant of $\mathbb{Q}(\sqrt{D})$. Since \mathfrak{l}_1 has order 2 in $\mathcal{CL}(D)$ and $\mathfrak{l}_1 \nmid D$ it has order 2 in the class group $\mathcal{CL}(D_0)$ of $\mathbb{Q}(\sqrt{D})$. Now, recall that, by genus theory, the 2-torsion in

³Derived from Theorem 19 in <http://www.imomath.com/index.php?options=328&lmm=0>.

$\mathcal{CL}(D_0)$ is generated by primes dividing D_0 . Therefore, if ι_1 has order 2, then $\iota_1 \mid D_0$, which gives a contradiction.

Therefore, in the case of $\left(\frac{D}{\ell}\right) = 1$ the probability that a random $j(E) \in \mathbb{F}_p$ having ≤ 2 neighbors on the crater of the volcano is $O(p^{-\frac{3}{4}}) = o(\frac{1}{\log p})$, which finishes the treatment of the case $\ell \geq 5$ or $\ell = 3$ and $\left(\frac{4p}{3}\right) = \left(\frac{p}{3}\right) = -1$.

For the remaining cases, where $\ell = 2$, or $\ell = 3$ and $\left(\frac{p}{3}\right) = 1$, we count the number of vertical isogenies. Following the formula for the depth of an isogeny volcano in Lemma 2.4, for an elliptic curve $E(\mathbb{F}_p)$ of trace t with $\ell^2 \mid t^2 - 4p$, the curve lives on the part of the volcano of depth ≥ 1 . In this case we only need to make sure that the curve does not live at the bottom of the volcano because otherwise it will necessarily have at least ℓ -neighbors (if it is at the bottom it has only one neighbor).

When $\ell = 2$, every $t = 2t_1$ satisfies $4 \mid t^2 - 4p$. So every $t \in [-2\sqrt{p}, 2\sqrt{p}] \cap 2\mathbb{Z}$ corresponds to trace of an elliptic curve over \mathbb{F}_p with at least two neighbors.

When $\ell = 3$ and $\left(\frac{p}{3}\right) = 1$, Hensel's lemma implies that $\frac{2}{9}$ of the $t \in [-2\sqrt{p}, 2\sqrt{p}] \cap \mathbb{Z}$ satisfy $t^2 \equiv 4p \pmod{9}$. These all correspond to traces of an elliptic curves over \mathbb{F}_p with at least two neighbors.

Finally, in both cases using Lemma 3.4 with a set S that takes a constant fraction from $[p+1 - \sqrt{p}, p+1 + \sqrt{p}] \cap \mathbb{Z}$, we see that the $O(\frac{1}{\log p})$ lower bound also applies for $\ell = 2$ or $\ell = 3$ and $\left(\frac{p}{3}\right) = 1$. \square

Proof of Theorem 3.3. Suppose that there is a probabilistic polynomial time algorithm A that finds all the ℓ -isogenous neighbors in Problem 3.2 with non-negligible probability η . We will build a probabilistic polynomial time algorithm A' that solves factoring. Given an integer N , A' samples two random integers $a, b \in \mathbb{Z}/N\mathbb{Z}$ such that $4a^3 + 27b^2 \neq 0$, and computes $j = 1728 \cdot \frac{4a^3}{4a^3 + 27b^2}$. With all but negligible probability $\gcd(j, N) = 1$ and $j \neq 0, 1728$; if j happens to satisfy $1 < \gcd(j, N) < N$, then A' outputs $\gcd(j, N)$.

A' then sends N, j_0 to the solver A for Problem 3.2 for a fixed polynomially large prime ℓ , gets back a set of solutions $\mathcal{J} = \{j_i\}_{i \in [k]}$, where $0 \leq k \leq (\ell + 1)^2$ denotes the number of solutions. With probability $\Omega(\frac{1}{\log^2 N})$, the curve $E : y^2 = x^3 + ax + b$ has at least two ℓ -isogenies over both \mathbb{F}_p and \mathbb{F}_q due to Theorem 3.5. In that case there exists $j, j' \in \mathcal{J}$ such that $1 < \gcd(j - j', N) < N$, which gives a prime factor of N . \square

3.3 The (ℓ, m) -isogenous neighbors problem over $\mathbb{Z}/N\mathbb{Z}$

Definition 3.6 (The (ℓ, m) -isogenous neighbors problem). *Let p and q be two distinct primes. Let $N := p \cdot q$. Let ℓ, m be two polynomially large integers s.t. $\gcd(\ell m, N) = 1$. The input of the (ℓ, m) -isogenous neighbor problem is the j -invariants j_1, j_2 of two elliptic curves E_1, E_2 defined over $\mathbb{Z}/N\mathbb{Z}$. The problem asks to find all the integers j' such that $\Phi_\ell(j(E_1), j') = 0$, and $\Phi_m(j(E_2), j') = 0$ over $\mathbb{Z}/N\mathbb{Z}$.*

When $\gcd(\ell, m) = 1$, applying the Euclidean algorithm on $\Phi_\ell(j_1, x)$ and $\Phi_m(j_2, x)$ gives a linear polynomial over x .

Lemma 3.7 ([ES10]). *Let $j_1, j_2 \in \text{Ell}_{\mathcal{O}}(\mathbb{F}_p)$, and let $\ell, m \neq p$ be distinct primes with $4\ell^2 m^2 < |D|$. Then the degree of $f(x) := \gcd(\Phi_\ell(j_1, x), \Phi_m(j_2, x))$ is less than or equal to 1.*

When $\gcd(\ell, m) = d > 1$, applying the Euclidean algorithm on $\Phi_\ell(j_1, x)$ and $\Phi_m(j_2, x)$ gives a polynomial of degree at least d . We present a proof in the case where $m = \ell^2$, which has the general idea.

Lemma 3.8. *Let $p \neq 2, 3$ and $\ell \neq p$ be primes, and let j_0, j_1 be such that $\Phi_\ell(j_0, j_1) = 0 \pmod p$. Let $\Phi_\ell(X, j_0)$ and $\Phi_{\ell^2}(X, j_1)$ be the modular polynomials of levels ℓ and ℓ^2 respectively. Then,*

$$(X - j_1) \cdot \gcd(\Phi_\ell(X, j_0), \Phi_{\ell^2}(X, j_1)) = \Phi_\ell(X, j_0)$$

in $\mathbb{F}_p[X]$. In particular,

$$\deg(\gcd(\Phi_\ell(X, j_0), \Phi_{\ell^2}(X, j_1))) = \ell$$

Proof. Without loss of generality we can, and we do, assume that $\Phi_\ell(X, j_0)$, $\Phi_\ell(X, j_1)$, and $\Phi_{\ell^2}(X, j_1)$ split over \mathbb{F}_p (otherwise we can base change to an extension k'/\mathbb{F}_p , where the full ℓ^2 -torsion is defined, this does not affect the degree of the gcd).

Assume that the degree of the gcd is N_{\gcd} . We have,

$$\deg(\Phi_\ell(X, j_0)) = \ell + 1, \quad \deg(\Phi_{\ell^2}(X, j_1)) = \ell(\ell + 1). \quad (4)$$

Let E_0, E_1 denote the (isomorphism classes of) elliptic curves with j -invariants j_0 and j_1 respectively, and $\varphi_\ell : E_0 \rightarrow E_1$ be the corresponding isogeny. We count the number N_{ℓ^2} of cyclic ℓ^2 -isogenies from E_1 two ways. First, N_{ℓ^2} is the number of roots of $\Phi_{\ell^2}(X, j_1)$, which, by (4) and the assumption that $\ell^2 + \ell < p$, is $\ell^2 + \ell$.

Next, recall (cf. Corollary 6.11 of [Sut]) that every isogeny of degree ℓ^2 can be decomposed as a composition of two degree ℓ isogenies (which are necessarily cyclic). Using this N_{ℓ^2} is bounded above by $N_{\gcd} + \ell^2$, where the first factor counts the number of ℓ^2 -isogenies $E_1 \rightarrow E$ that are compositions $E_1 \xrightarrow{\hat{\varphi}_\ell} E_0 \rightarrow E$, and the second factor counts the isogenies that are compositions $E_1 \rightarrow E' \rightarrow E$, where $E' \not\cong E_1$. Note that we are not counting compositions $E_1 \xrightarrow{\phi} \tilde{E} \xrightarrow{\hat{\phi}} E_1$ since these do not give rise to cyclic isogenies.

This shows that $\ell^2 + \ell \leq \ell^2 + N_{\ell^2} \Rightarrow N_{\gcd} \geq \ell$. On the other hand, by (4) $N_{\gcd} \leq \ell$ since $\Phi_\ell(X, j_0)/(X - j_0)$ has degree ℓ and each root except for j_1 gives a (possibly cyclic) ℓ^2 -isogeny by composition with $\hat{\varphi}_\ell$. This implies that $N_{\gcd} = \ell$ and that all the ℓ^2 -isogenies obtained this way are cyclic. In particular, we get that the gcd is $\Phi_\ell(X, j_0)/(X - j_1)$. \square

Discussions. Let us remark that we do not know if solving the (ℓ, ℓ^2) -isogenous neighbors problem is as hard as factoring. To adapt the same reduction in the proof of Theorem 3.3, we need the feasibility of sampling two integers j_1, j_2 such that $\Phi_\ell(j_1, j_2) = 0 \pmod N$, and j_1 or j_2 has to have another isogenous neighbor over \mathbb{F}_p or \mathbb{F}_q . However the feasibility is unclear to us in general.

From the cryptanalytic point of view, a significant difference of the (ℓ, ℓ^2) -isogenous neighbors problem and the ℓ -isogenous neighbors problem is the following. Let ℓ be an odd prime. Recall that an isogeny $\phi : E_1 \rightarrow E_2$ of degree ℓ can be represented by a rational polynomial

$$\phi : E_1 \rightarrow E_2, \quad (x, y) \mapsto \left(\frac{f(x)}{h(x)^2}, \frac{g(x, y)}{h(x)^3} \right),$$

where $h(x)$ is its *kernel polynomial* of degree $\frac{\ell-1}{2}$. The roots of $h(x)$ are the x -coordinates of the kernel subgroup $G \subset E_1[\ell]$ such that $\phi : E_1 \rightarrow E_1/G$.

Given a single j -invariant j' over $\mathbb{Z}/N\mathbb{Z}$, it is infeasible to find a rational polynomial ϕ of degree ℓ that maps from a curve E with j -invariant j' to another curve j'' , since otherwise j'' is a solution to the ℓ -isogenous neighbors problem. However, if we are given two j -invariants $j_1, j_2 \in \mathbb{Z}/N\mathbb{Z}$ such that $\Phi_\ell(j_1, j_2) = 0 \pmod{N}$, as in the (ℓ, ℓ^2) -isogenous neighbors problem; then it is feasible to compute a pair of curves E_1, E_2 such that $j(E_1) = j_1, j(E_2) = j_2$, together with an explicit rational polynomial of an ℓ -isogeny from E_1 to E_2 . This is because the arithmetic operations involved in computing the kernel polynomial $h(x)$ mentioned in [CM94, Sch95, E⁺98] works over $\mathbb{Z}/N\mathbb{Z}$ by reduction mod N , and does not require the factorization of N .

Proposition 3.9. *Given $\ell, N \in \mathbb{Z}$ such that $\gcd(\ell, N) = 1$, and two integers $j_1, j_2 \in \mathbb{Z}/N\mathbb{Z}$ such that $\Phi_\ell(j_1, j_2) = 0$ over $\mathbb{Z}/N\mathbb{Z}$, the elliptic curves E_1, E_2 , and the kernel polynomial $h(x)$ of an isogeny ϕ from E_1, E_2 can be computed in time polynomial in $\ell, \log(N)$. From the kernel polynomial $h(x)$ of an isogeny ϕ , computing $f(x), g(x, y)$, hence the entire rational polynomial of ϕ , is feasible over $\mathbb{Z}/N\mathbb{Z}$ via Vélu's formulae [Vél71].*

However, it is unclear how to utilize the rational polynomial to solve the (ℓ, ℓ^2) -joint neighbors problem. We postpone further discussions on the hardness and cryptanalysis to Section 5.

4 Trapdoor group with infeasible inversion

In this section we present the construction of the trapdoor group with infeasible inversion. As the general construction is somewhat technical we will present it in two steps: first we will go over the basic algorithms that feature a simple encoding and composition rule, which suffices for the instantiations of the applications; we will then move to the general algorithms that offer potential optimization and flexibility.

4.1 Definitions

Let us first provide the definition of a TGII, adapted from the original definition in [Hoh03, Mol03] to match our construction. The main differences are:

1. The trapdoor in the definition of [Hoh03, Mol03] is only used to invert an encoded group element, whereas we assume the trapdoor can be used to encode and decode (which implies the ability of inverting).
2. We classify the encodings of the group elements as *canonical encodings* and *composable encodings*, whereas the definition from [Hoh03, Mol03] does not. In our definition, the canonical encoding of an element is uniquely determined once the public parameter is fixed. It can be directly used in the equivalence test, but it does not support efficient group operations. Composable encodings of group elements support efficient group operations. A composable encoding, moreover, can be converted into a canonical encoding by an efficient, public extraction algorithm.

Definition 4.1. *Let $\mathbb{G} = (\circ, 1_{\mathbb{G}})$ be a finite multiplicative group where \circ denotes the group operator, and $1_{\mathbb{G}}$ denotes the identity. For $x \in \mathbb{G}$, denote its inverse by x^{-1} . \mathbb{G} is associated with the following efficient algorithms:*

Parameter generation. $\text{Gen}(1^\lambda)$ takes as input the security parameter 1^λ , outputs the public parameter PP and the trapdoor τ .

Private sampling. $\text{TrapSam}(\text{PP}, \tau, x)$ takes as inputs the public parameter PP , the trapdoor τ , and a plaintext group element $x \in \mathbb{G}$, outputs a composable encoding $\text{enc}(x)$.

Composition. $\text{Compose}(\text{PP}, \text{enc}(x), \text{enc}(y))$ takes as inputs the public parameter PP , two composable encodings $\text{enc}(x), \text{enc}(y)$, outputs $\text{enc}(x \circ y)$. We often use the notation $\text{enc}(x) \circ \text{enc}(y)$ for $\text{Compose}(\text{PP}, \text{enc}(x), \text{enc}(y))$.

Extraction. $\text{Ext}(\text{PP}, \text{enc}(x))$ takes as inputs the public parameter PP , a composable encoding $\text{enc}(x)$ of x , outputs the canonical encoding of x as $\text{enc}^*(x)$.

The hardness of inversion requires that it is infeasible for any efficient algorithm to produce the canonical encoding of x^{-1} given a composable encoding of $x \in \mathbb{G}$.

Hardness of inversion. For any p.p.t. algorithm A ,

$$\Pr[z = \text{enc}^*(x^{-1}) \mid z \leftarrow A(\text{PP}, \text{enc}(x))] < \text{negl}(\lambda),$$

where the probability is taken over the randomness in the generation of PP , x , $\text{enc}(x)$, and the adversary A .

4.2 Construction - 1: Basic setting

In this section we provide the formal construction of the TGII with the basic setting of algorithms. The basic setting assumes that in the application of TGII, the encoding sampling algorithm can be stateful, and it is easy to determine which encodings have to be pairwise composable, and which are not. Under these assumptions, we show that we can always sample composable encodings so that the composition always succeeds. That is, the degrees of the any two encodings are chosen to be coprime if they will be composed in the application, and not coprime if they will not be composed. The reader may be wondering why we are distinguishing pairs that are composable and those that are not, as opposed to simply assuming that every pairs of encoding are composable. The reason is for security, mainly due to the parallelogram attack in §5.3.2.

The basic setting suffices for instantiating the directed transitive signature [Hoh03, Mol03] and the broadcast encryption schemes [ILOP04], where the master signer and the master encrypter are stateful. We will explain how to determine which encodings are pairwise composable in these two applications, so as to determine the prime degrees of the encodings (the rest of the parameters are not application-specific and follow the universal solution from this section).

For convenience of the reader and for further reference, we provide in Figure 4 a summary of the parameters, with the basic constraints they should satisfy, and whether they are public or hidden. The correctness and efficiency reasons behind these constraints will be detailed in the coming paragraphs, whereas the security reasons will be explained in §5.

Parameter generation. The parameter generation algorithm $\text{Gen}(1^\lambda)$ takes the security parameter 1^λ as input, first chooses a non-maximal order \mathcal{O} of an imaginary quadratic field as follows:

1. Select a square-free negative integer $D_0 \equiv 1 \pmod{4}$ as the fundamental discriminant, such that D_0 is polynomially large and $h(D_0)$ is a prime.

Parameters	Basic constraints	Public?
The modulus N	$N = pq$, p, q are primes, $ p , q \in \text{poly}(\lambda)$	Yes
The identity $j(E_0)$	$\text{End}(E_0(\mathbb{F}_p)) \simeq \text{End}(E_0(\mathbb{F}_q)) \simeq \mathcal{O}$	Yes
$\#(E_0(\mathbb{F}_p))$, $\#(E_0(\mathbb{F}_q))$	not polynomially smooth	No
The discriminant D of \mathcal{O}	$D = D_0 \cdot f^2$, $D \approx \lambda^{O(\log \lambda)}$, D is polynomially smooth	No
The class number $h(D)$	follows the choice of D	No
A set S in an encoding:	$S = \{C_i = [(p_i, b_i, \cdot)]\}_{i \in [w]}$ generates $\mathcal{CL}(D)_{\text{odd}}$	See below
The number w of ideals	$w \in O(\log \lambda)$	Yes
The degree p_i of isogenies	$p_i \in \text{poly}(\lambda)$	Yes
The basis \mathbf{B} of Λ_S	$\ \tilde{\mathbf{B}}\ \in \text{poly}(\lambda)$	No

Figure 4: Summary of the choices of parameters in the basic setting.

2. Choose $k = O(\log(\lambda))$, and a set of distinct polynomially large prime numbers $\{f_i\}_{i \in [k]}$ such that the odd-part of $\left(f_i - \left(\frac{D_0}{f_i}\right)\right)$ is square-free and not divisible by $h(D_0)$. Let $f = \prod_{i \in [k]} f_i$.
3. Set $D = f^2 D_0$. Recall from Eqn. (1) that

$$h(D) = 2 \cdot \frac{h(D_0)}{w(D_0)} \prod_{i \in [k]} \left(f_i - \left(\frac{D_0}{f_i}\right)\right) \quad (5)$$

Let $\mathcal{CL}(\mathcal{O})_{\text{odd}}$ be the odd part of $\mathcal{CL}(\mathcal{O})$, $h(D)_{\text{odd}}$ be largest odd factor of $h(D)$. Note that due to the choices of D_0 and $\{f_i\}$, $\mathcal{CL}(\mathcal{O})_{\text{odd}}$ is cyclic, and we have $|D|, h(D)_{\text{odd}} \in \lambda^{O(\log \lambda)}$. The group with infeasible inversion \mathbb{G} is then $\mathcal{CL}(\mathcal{O})_{\text{odd}}$ with group order $h(D)_{\text{odd}}$.

We then sample the public parameters as follows:

1. Choose two primes p, q , and elliptic curves $E_{0, \mathbb{F}_p}, E_{0, \mathbb{F}_q}$ with discriminant D , using the CM method (cf. [LZ94] and more).
2. Check whether p and q are safe RSA primes (if not, then back to the previous step and restart). Also, check whether the number of points $\#(E_0(\mathbb{F}_p)), \#(E_0(\mathbb{F}_q)), \#(\tilde{E}_0(\mathbb{F}_p)), \#(\tilde{E}_0(\mathbb{F}_q))$ (where \tilde{E} denotes the quadratic twist of E) are polynomially smooth (if yes, then back to the previous step and restart). p, q and the number of points should be hidden for security.
3. Set the modulus N as $N := p \cdot q$ and let $j_0 = \text{CRT}(p, q; j(E_{0, \mathbb{F}_p}), j(E_{0, \mathbb{F}_q}))$. Let j_0 represent the identity of \mathbb{G} .

Output (N, j_0) as the public parameter PP. Keep (D, p, q) as the trapdoor τ (D and the group order of \mathbb{G} should be hidden for security).

The sampling algorithm and the group operation of the composable encodings. Next we provide the definitions and the algorithms for the composable encoding.

Definition 4.2 (Composable encoding). *Given a factorization of x as $\prod_{i=1}^w C_i^{e_i}$, where $w \in O(\log \lambda)$; $C_i = [(p_i, b_i, \cdot)] \in \mathbb{G}$, $e_i \in \mathbb{N}$, for $i \in [w]$. A composable encoding of $x \in \mathbb{G}$ is represented by*

$$\text{enc}(x) = (L; T_1, \dots, T_w) = ((p_1, \dots, p_w); (j_{1,1}, \dots, j_{1,e_1}), \dots, (j_{w,1}, \dots, j_{w,e_w})),$$

where all the primes in the list $L = (p_1, \dots, p_w)$ are distinct; for each $i \in [w]$, $T_i \in (\mathbb{Z}/N\mathbb{Z})^{e_i}$ is a list of the j -invariants such that $j_{i,k} = C_i^k * j_0$, for $k \in [e_i]$.

The degree of an encoding $\text{enc}(x)$ is defined to be $d(\text{enc}(x)) := \prod_{i=1}^w p_i^{e_i}$.

Notice that the factorization of $x = \prod_{i=1}^w C_i^{e_i}$ has to satisfy $e_i \in \text{poly}(\lambda)$, for all $i \in [w]$, so as to ensure the length of $\text{enc}(x)$ is polynomial. Looking ahead, we also require each p_i , the degree of the isogeny that represents the C_i -action, to be polynomially large so as to ensure Algorithm 4.3 in the encoding sampling algorithm and Algorithm 4.9 in the extraction algorithm run in polynomial time.

The composable encoding sampling algorithm requires the following subroutine:

Algorithm 4.3. $\text{act}(\tau, j, C)$ takes as input the trapdoor $\tau = (D, p, q)$, a j -invariant $j \in \mathbb{Z}/N\mathbb{Z}$, and an ideal class $C \in \mathcal{CL}(\mathcal{O})$, proceeds as follows:

1. Let $j_p = j \pmod p$, $j_q = j \pmod q$.
2. Compute $j'_p := C * j_p \in \mathbb{F}_p$, $j'_q := C * j_q \in \mathbb{F}_q$.
3. Output $j' := \text{CRT}(p, q; j'_p, j'_q)$.

Algorithm 4.4 (Sample a composable encoding). Given as input the public parameter $\text{PP} = (N, j_0)$, the trapdoor $\tau = (D, p, q)$, and $x \in \mathbb{G}$, $\text{TrapSam}(\text{PP}, \tau, x)$ produces a composable encoding of x is sampled as follows:

1. Choose $w \in O(\log \lambda)$ and a generation set $S = \{C_i = [(p_i, b_i, \cdot)]\}_{i \in [w]} \subset \mathbb{G}$.
2. Sample a short basis \mathbf{B} (in the sense that $\|\tilde{\mathbf{B}}\| \in \text{poly}(\lambda)$) for the relation lattice Λ_S :

$$\Lambda_S := \left\{ \mathbf{y} \mid \mathbf{y} \in \mathbb{Z}^w, \prod_{i \in [w]} C_i^{y_i} = 1_{\mathbb{G}} \right\}. \quad (6)$$

3. Given x , S , \mathbf{B} , sample a short vector $\mathbf{e} \in \{\text{poly}(\lambda) \cap \mathbb{N}\}^w$ such that $x = \prod_{i \in [w]} C_i^{e_i}$.
4. For all $i \in [w]$:
 - (a) Let $j_{i,0} := j_0$.
 - (b) For $k = 1$ to e_i : compute $j_{i,k} := \text{act}(\tau, j_{i,k-1}, C_i)$.
 - (c) Let $T_i := (j_{i,1}, \dots, j_{i,e_i})$.
5. Let $L \in \mathbb{N}^w$ be a list where the i^{th} entry of L is p_i .

6. Output the composable encoding of x as

$$\text{enc}(x) = (L; T_1, \dots, T_w) = ((p_1, \dots, p_w); (j_{1,1}, \dots, j_{1,e_1}), \dots, (j_{w,1}, \dots, j_{w,e_w})).$$

Remark 4.5 (Thinking of each adjacent pair of j -invariants as an isogeny). In each T_i , each adjacent pair of the j -invariants can be thought of representing an isogeny ϕ that corresponds to the ideal class $C_i = [(p_i, b_i, \cdot)]$. Over the finite field, C_i can be explicitly recovered from an adjacent pair of the j -invariants and p_i (cf. Remark 2.3). Over $\mathbb{Z}/N\mathbb{Z}$, the rational polynomial of the isogeny ϕ can be recovered from the adjacent pair of the j -invariants and p_i (cf. Proposition 3.9), but it is not clear how to recover b_i in the binary quadratic form representation of C_i .

Remark 4.6 (The only stateful step in the sampling algorithm). Recall that the basic setting assumes the encoding algorithm is stateful, where the state records the prime factors of the degrees used in the existing composable encodings. The state is only used in the first step to choose the $\{p_i\}$ of the ideals in the generation set $S = \{C_i = [(p_i, b_i, \cdot)]\}_{i \in [w]}$.

Group operations. Given two composable encodings, the group operation is done by simply concatenating the encodings if their degrees are coprime, or otherwise outputting “failure”.

Algorithm 4.7. *The encoding composition algorithm $\text{Compose}(\text{PP}, \text{enc}(x), \text{enc}(y))$ parses $\text{enc}(x) = (L_x; T_{x,1}, \dots, T_{x,w_x})$, $\text{enc}(y) = (L_y; T_{y,1}, \dots, T_{y,w_y})$, produces the composable encoding of $x \circ y$ as follows:*

- If $\gcd(d(\text{enc}(x)), d(\text{enc}(y))) = 1$, then output the composable encoding of $x \circ y$ as

$$\text{enc}(x \circ y) = (L_x || L_y; T_{x,1}, \dots, T_{x,w_x}, T_{y,1}, \dots, T_{y,w_y}).$$

- If $\gcd(d(\text{enc}(x)), d(\text{enc}(y))) > 1$, output “failure”.

The canonical encoding and the extraction algorithm.

Definition 4.8 (Canonical encoding). *The canonical encoding of $x \in \mathbb{G}$ is $x * j_0 \in \mathbb{Z}/N\mathbb{Z}$.*

The canonical encoding of x can be computed by first obtaining a composable encoding of x , and then converting the composable encoding into the canonical encoding using the extraction algorithm. The extraction algorithm requires the following subroutine.

Algorithm 4.9 (The “gcd” operation). *The algorithm $\text{gcd.op}(\text{PP}, \ell_1, \ell_2; j_1, j_2)$ takes as input the public parameter PP , two degrees ℓ_1, ℓ_2 and two j -invariants j_1, j_2 , proceeds as follows:*

- If $\gcd(\ell_1, \ell_2) = 1$, then it computes the linear function $f(x) = \gcd(\Phi_{\ell_2}(j_1, x), \Phi_{\ell_1}(j_2, x))$ over $\mathbb{Z}/N\mathbb{Z}$, and outputs the only root of $f(x)$;
- If $\gcd(\ell_1, \ell_2) > 1$, it outputs \perp .

Algorithm 4.10. $\text{Ext}(\text{PP}, \text{enc}(x))$ converts the composable encoding $\text{enc}(x)$ into the canonical encoding $\text{enc}^*(x)$. The algorithm maintains a pair of lists (U, V) , where U stores a list of j -invariants $(j_1, \dots, j_{|U|})$, V stores a list of degrees where the i^{th} entry of V is the degree of isogeny between j_i and j_{i-1} (when $i = 1$, j_{i-1} is the j_0 in the public parameter). The lengths of U and V are always equal during the execution of the algorithm.

The algorithm parses $\text{enc}(x) = (L; T_1, \dots, T_w)$, proceeds as follows:

1. Initialization: Let $U := T_1$, $V := (L_1, \dots, L_1)$ of length $|T_1|$ (i.e. copy L_1 for $|T_1|$ times).
2. For $i = 2$ to w :
 - (a) Set $u_{\text{temp}} := |U|$.
 - (b) For $k = 1$ to $|T_i|$:
 - i. Let $t_{i,k,0}$ be the k^{th} j -invariant in T_i , i.e. $j_{i,k}$;
 - ii. For $h = 1$ to u_{temp} :
 - If $k = 1$, compute $t_{i,k,h} := \text{gcd.op}(\text{PP}, L_i, V_h; t_{i,k,h-1}, U_h)$;
 - If $k > 1$, compute $t_{i,k,h} := \text{gcd.op}(\text{PP}, L_i, V_h; t_{i,k,h-1}, t_{i,k-1,h})$;
 - iii. Append $t_{i,k,u_{\text{temp}}}$ to the list U , append L_i to the list V .
3. Return the last entry of U .

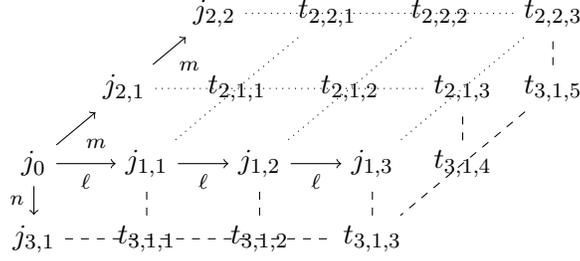


Figure 5: An example for the composable encoding and the extraction algorithm.

Example 4.11. Let us give a simple example for the composition and the extraction algorithms. Let ℓ, m, n be three distinct polynomially large primes. Let the composable encoding of an element y be $\text{enc}(y) = ((\ell); (j_{1,1}, j_{1,2}, j_{1,3}))$, based on the factorization of $y = C_1^{e_1} = [(\ell, b_\ell, \cdot)]^3$. Let the composable encoding of an element z be $\text{enc}(z) = ((m, n); (j_{2,1}, j_{2,2}), (j_{3,1}))$, based on the factorization of $z = C_2^{e_2} \cdot C_3^{e_3} = [(m, b_m, \cdot)]^2 \cdot [(n, b_n, \cdot)]^1$. Then the composable encoding of $x = y \circ z$ obtained from Algorithm 4.7 is $\text{enc}(x) = ((\ell, m, n); (j_{1,1}, j_{1,2}, j_{1,3}), (j_{2,1}, j_{2,2}), (j_{3,1}))$.

Next we explain how to extract the canonical encoding of x from $\text{enc}(x)$. In Figure 5, the j -invariants in $\text{enc}(x)$ are placed on the solid arrows (their positions do not follow the relative positions on the volcano). We can think of each gcd operation in Algorithm 4.9 as fulfilling a missing vertex of a parallelogram defined by three existing vertices.

When running $\text{Ext}(\text{PP}, \text{enc}(x))$, the list U is initialized as $(j_{1,1}, j_{1,2}, j_{1,3})$, the list V is initialized as (ℓ, ℓ, ℓ) . Let us go through the algorithm for $i = 2$ and $i = 3$ in the second step.

- When $i = 2$, u_{temp} equals to $|U| = 3$. The intermediate j -invariants $\{t_{2,k,h}\}_{k \in [|T_2|], h \in [u_{\text{temp}}]}$ are placed on the dotted lines, computed in the order of $t_{2,1,1}, t_{2,1,2}, t_{2,1,3}, t_{2,2,1}, t_{2,2,2}, t_{2,2,3}$. The list U is updated to $(j_{1,1}, j_{1,2}, j_{1,3}, t_{2,1,3}, t_{2,2,3})$, the list V is updated to (ℓ, ℓ, ℓ, m, m)
- When $i = 3$, u_{temp} equals to $|U| = 5$. The intermediate j -invariants $\{t_{3,1,h}\}_{h \in [u_{\text{temp}}]}$ are placed on the dashed lines, computed in the order of $t_{3,1,1}, \dots, t_{3,1,5}$. In the end, $t_{3,1,5}$ is appended to U , n is appended to V .

The canonical encoding of x is then $t_{3,1,5}$.

On correctness and efficiency. We now verify the correctness and efficiency of the parameter generation, encoding sampling, composition, and the extraction algorithms.

To begin with, we verify that the canonical encoding correctly and uniquely determines the group element in $\mathcal{CL}(\mathcal{O})$. It follows from the choices of the elliptic curves $E_0(\mathbb{F}_p)$ and $E_0(\mathbb{F}_q)$ with $\text{End}(E_0(\mathbb{F}_p)) \simeq \text{End}(E_0(\mathbb{F}_q)) \simeq \mathcal{O}$, and the following bijection once we fix E_0 :

$$\mathcal{CL}(\mathcal{O}) \rightarrow \text{Ell}_{\mathcal{O}}(\mathbf{k}), \quad x \mapsto x * j(E_0(\mathbf{k})), \quad \text{for } \mathbf{k} \in \{\mathbb{F}_p, \mathbb{F}_q\}$$

Next, we will show that generating the parameters, i.e. the curves $E_{0,\mathbb{F}_p}, E_{0,\mathbb{F}_q}$ with a given fundamental discriminant D_0 and a conductor $f = \prod_i^k f_i$, is efficient when $|D_0|$ and all the factors of f are of polynomial size. Let u be an integer such that $f \mid u$. Choose a p and t_p such that $t_p^2 - 4p = u^2 D_0$. Then, compute the Hilbert class polynomial H_{D_0} over \mathbb{F}_p and find one of its roots

j . From j , descending on the volcanoes $G_{f_i}(\mathbb{F}_p)$ for every f_i gives the j -invariant for the curve with desired discriminant. The same construction works verbatim for q .

We then show that sampling the composable encodings can be done in heuristic polynomial time:

1. Given a logarithmically large generation set $S = \{C_i = [(\ell_i, b_i, \cdot)] \in \mathcal{CL}(\mathcal{O})\}_{i \in [w]}$, a possibly big basis of the relation lattice Λ_S can be obtained by solving the discrete-log problem over $\mathcal{CL}(\mathcal{O})$, which can be done in polynomial time since the group order is polynomially smooth.
2. Suppose that the lattice Λ_S satisfies the Gaussian heuristic (this is the only heuristic we assume). That is, for all $1 \leq i \leq w$, the i^{th} successive minimum of Λ_S , denoted as λ_i , satisfies $\lambda_i \approx \sqrt{w} \cdot h(\mathcal{O})^{1/w} \in \text{poly}(\lambda)$. Since $w = O(\log(\lambda))$, the short basis \mathbf{B} of Λ_S , produced by the LLL algorithm, satisfies $\|\mathbf{B}\| \leq 2^{\frac{w}{2}} \cdot \lambda_w \in \text{poly}(\lambda)$.
3. Given a target group element $x \in \mathcal{CL}(\mathcal{O})$, the polynomially short basis \mathbf{B} , we can sample a vector $\mathbf{e} \in \mathbb{N}^w$ such that $\prod_{i=1}^m C_i^{e_i} = x$ and $\|\mathbf{e}\|_1 \in \text{poly}(\lambda)$ in polynomial time using e.g. Babai's algorithm [Bab86]. (In §5.3.3, we will explain that the GPV sampler [GPV08] is preferred for the security purpose.)
4. The unit operation $\text{act}(\tau, j, C)$ is efficient when the ideal class C corresponds to a polynomial degree isogeny, since it is efficient to compute polynomial degree isogenies over the finite fields.
5. The length of the final output $\text{enc}(x)$ is $(w + \|\mathbf{e}\|_1) \cdot \text{poly}(\lambda) \in \text{poly}(\lambda)$.

The algorithm $\text{Compose}(\text{PP}, \text{enc}(x), \text{enc}(y))$ simply concatenates $\text{enc}(x)$, $\text{enc}(y)$, so it is efficient as long as $\text{enc}(x)$, $\text{enc}(y)$ are of polynomial size.

The correctness of the unit operation gcd.op follows the commutativity of the endomorphism ring \mathcal{O} . The operation $\text{gcd.op}(\text{PP}, \ell_1, \ell_2; j_1, j_2)$ is efficient when $\text{gcd}(\ell_1, \ell_2) = 1$, $\ell_1, \ell_2 \in \text{poly}(\lambda)$, given that solving the (ℓ_1, ℓ_2) isogenous neighbor problem over $\mathbb{Z}/N\mathbb{Z}$ is efficient under these conditions.

When applying $\text{Ext}()$ (Algorithm 4.10) on a composable encoding $\text{enc}(x) = (L_x; T_{x,1}, \dots, T_{x,w_x})$, it runs gcd.op for $\max_{i=1}^{w_x} |T_{x,i}| \cdot (\sum_{i=1}^{w_x} |T_{x,i}|)$ times. So obtaining the canonical encoding is efficient as long as all the primes in L_x are polynomially large, and $|T_{x,i}| \in \text{poly}(\lambda)$ for all $i \in [w_x]$.

Remark 4.12 (The parameters in practice). *Let us mention the two extreme sides on the deviations of the parameters between theory and practice that we expect. The discriminant $D = D_0 \cdot f^2$ is bounded by $\lambda^{O(\log \lambda)}$ asymptotically, which leads to a $\lambda^{O(\log \lambda)}$ -time attack by first guessing D , then solving the discrete-log problem over $\mathcal{CL}(D)$. The bottleneck of the bound of D is at the dimension w of the lattice Λ_S , which is set to be $O(\log \lambda)$ so that the lattice reduction algorithm runs in polynomial time. In practice the lattice reduction algorithms are known to outperform the asymptotic bound. So we expect the discriminant D can be set larger than a direct estimation from $\lambda^{O(\log \lambda)}$.*

On the other hand, the modular polynomials take a solid amount of space to store in practice, so we expect the gcd operation in Algorithm 4.9 to be slow for large polynomial degrees of ℓ_1 and ℓ_2 .

4.3 Construction - 2: General case

We describe the generalizations of the basic algorithms that offer potential optimization and flexibility.

Composing non-coprime degree encodings using “ladders”. In the basic setting of the composable encodings, the group operation is feasible only for encodings with relatively prime degrees. To support the compositions of non-relatively prime degree encodings, say the shared prime degree is ℓ , we can include in the public parameter the encodings of $\{x^i\}_{i \in [k]}$, where $x = [(\ell, b, \cdot)]$, k is a polynomial. This then supports the composition of two encodings whose sum of exponents on the degree ℓ is $\leq k$.

Sampling ladders has the benefit of supporting bounded number of shared-degree compositions. In some settings of the applications (e.g. in the broadcast encryption where the number of users is bounded a priori), ladders enable stateless encoding sampling algorithms.

Algorithm 4.13 (Sampling a ladder). *Given a polynomially large prime ℓ , a polynomial k , and an element $x = [(\ell, b, \cdot)] \in \mathcal{CL}(D)$, sample a ladder for degree ℓ of length k as follows:*

1. For $i = 1$ to k : compute $j_i := \text{act}(\tau, j_{i-1}, x)$. (Recall j_0 is the identity from the public parameter.)
2. Let $\text{ladder}(\ell) := (j_1, \dots, j_k)$. Include $\text{ladder}(\ell)$ in the public parameter.

For the convenience of the description of the group operation with ladders, let $\iota(\ell, L)$ take as input an integer ℓ , a list L , and output the index of ℓ in L .

Algorithm 4.14 (Composition with ladders). *The algorithm $\text{Compose}(\text{PP}, \text{enc}(x), \text{enc}(y))$ parses $\text{enc}(x) = (L_x; T_{x,1}, \dots, T_{x,w_x})$, $\text{enc}(y) = (L_y; T_{y,1}, \dots, T_{y,w_y})$, produces the composable encoding of $z = x \circ y$ as follows:*

1. Let $L_z = L_x \cup L_y$.
2. For all $\ell \in L_x \setminus L_x \cap L_y$, let $T_{z,\iota(\ell, L_z)} = T_{x,\iota(\ell, L_x)}$; for all $\ell \in L_y \setminus L_x \cap L_y$, let $T_{z,\iota(\ell, L_z)} = T_{y,\iota(\ell, L_y)}$.
3. For all $\ell \in L_x \cap L_y$:
 - If $|\text{ladder}(\ell)| \geq |T_{x,\iota(\ell, L_x)}| + |T_{y,\iota(\ell, L_y)}|$, then let $T_{z,\iota(\ell, L_z)}$ be the list of the first $|T_{x,\iota(\ell, L_x)}| + |T_{y,\iota(\ell, L_y)}|$ elements in $\text{ladder}(\ell)$.
 - If $|\text{ladder}(\ell)| < |T_{x,\iota(\ell, L_x)}| + |T_{y,\iota(\ell, L_y)}|$, then the composition is infeasible. Return “failure”.
4. Output the composable encoding of z as $\text{enc}(z) = (L_z; T_{z,1}, \dots, T_{z,|L_z|})$.

Sample the composable encoding of a random element. In the applications we are often required to sample (using the trapdoor) the composable encoding of a random element in \mathbb{G} under the generation set $S = \{C_i := [(\ell_i, b_i, \cdot)]\}_{i \in [w]}$ and. Let us call such an algorithm $\text{RandomSam}(\tau, S)$.

An obvious instantiation of $\text{RandomSam}(\tau, S)$ is to simply choose a random element x from \mathbb{G} first, then run $\text{TrapSam}(\text{PP}, \tau, x)$ in Algorithm 4.4.

Alternatively, we can pick a random exponent vector $\mathbf{e} \in [-B, B]^w$, and let $x = \prod_{i \in [w]} C_i^{e_i}$. For proper choices of B , w , and the set of ideals, x is with high min-entropy heuristically, but figuring out the exact distribution of x is difficult in general. If B , w , and the set of ideals are chosen according to Lemma 2.5, then under GRH, x is statistically close to uniform over $\mathcal{CL}(\mathcal{O})$.

Compressing the composable encoding using the partial extraction algorithm. In the basic setting, the composition of composable encodings keeps growing. In some application, it is tempting to compress the composable encodings as much as possible.

We provide a partial extraction algorithm, which takes two composable encodings $\text{enc}(x), \text{enc}(y)$ and the public parameter as inputs. It outputs a j -invariant j_x as the canonical encoding of x , and a sequences of isogenies $\phi_y = (\phi_1, \dots, \phi_n)$ such that $\phi_n \circ \dots \circ \phi_1(j_x) = j_{x \circ y}$ (we abuse the notation by using the j -invariant to represent an elliptic curve with such a j -invariant), where $j_{x \circ y}$ is the canonical encoding of $x \circ y$, n is the number of the j -invariants in $\text{enc}(y)$. In other words, ϕ_y represents the ideal class $C_y \in \mathbb{G}$ defined by $\text{enc}(y)$. The one-side partial extraction algorithm is useful in the situation where the encoding $\text{enc}(x)$ is relatively long and needs to be compressed, the encoding $\text{enc}(y)$ is relatively short and doesn't have to be compressed.

Now we present the partial extraction algorithm. The algorithm first runs the basic extraction algorithm $\text{Ext}(\text{PP}, \cdot)$ on $\text{enc}(x)$ (cf. Algorithm 4.10), then produces ϕ_y using $\text{enc}(y)$ and the intermediate information from $\text{Ext}(\text{PP}, \text{enc}(x))$.

Algorithm 4.15. *Partial.Ext(PP, enc(x), enc(y)) maintains a pair of lists (U, V) , where U stores a list of j -invariants $(j_1, \dots, j_{|U|})$, V stores a list of degrees where the i^{th} entry of V is the degree of isogeny between j_i and j_{i-1} (when $i = 1$, j_{i-1} is the j_0 in the public parameter). The lengths of U and V are always equal during the execution of the algorithm.*

The algorithm parses $\text{enc}(x) = (L_x; T_{x,1}, \dots, T_{x,w_x})$, $\text{enc}(y) = (L_y; T_{y,1}, \dots, T_{y,w_y})$, proceeds as follows:

1. *Initialization: Let $U := T_{x,1}$, $V := (L_{x,1}, \dots, L_{x,1})$ of length $|T_{x,1}|$.*
2. *For $i = 2$ to w_x :*
 - (a) *Set $u_{\text{temp}} := |U|$.*
 - (b) *For $k = 1$ to $|T_{x,i}|$:*
 - i. *Let $t_{x,i,k,0}$ be the k^{th} j -invariant in $T_{x,i}$, i.e. $j_{x,i,k}$;*
 - ii. *For $h = 1$ to u_{temp} :*
 - *If $k = 1$, compute $t_{x,i,k,h} := \text{gcd.op}(\text{PP}, L_{x,i}, V_h; t_{x,i,k,h-1}, U_h)$;*
 - *If $k > 1$, compute $t_{x,i,k,h} := \text{gcd.op}(\text{PP}, L_{x,i}, V_h; t_{x,i,k,h-1}, t_{x,i,k-1,h})$;*
 - iii. *Append $t_{x,i,k,u_{\text{temp}}}$ to the list U , append $L_{x,i}$ to the list V .*
3. *Let j_x be the last entry of U .*
4. *Initialize a counter $z := 1$.*
5. *For $i = 1$ to w_y :*
 - (a) *Set $u_{\text{temp}} := |U|$.*
 - (b) *For $k = 1$ to $|T_{y,i}|$:*
 - i. *Let $t_{y,i,k,0}$ be the k^{th} j -invariant in $T_{y,i}$, i.e. $j_{y,i,k}$;*
 - ii. *For $h = 1$ to u_{temp} :*
 - *If $k = 1$, compute $t_{y,i,k,h} := \text{gcd.op}(\text{PP}, L_{y,i}, V_h; t_{y,i,k,h-1}, U_h)$;*
 - *If $k > 1$, compute $t_{y,i,k,h} := \text{gcd.op}(\text{PP}, L_{y,i}, V_h; t_{y,i,k,h-1}, t_{y,i,k-1,h})$;*
 - iii. *Let j_{temp} be the last entry in U .*
 - iv. *(*) Produce an isogeny ϕ_z of degree $L_{y,i}$ such that $\phi_z(j_{\text{temp}}) = t_{y,i,k,u_{\text{temp}}}$.*

- v. Increase the counter $z := z + 1$.
 - vi. Append $t_{y,i,k,u_{\text{temp}}}$ to the list U , append $L_{y,i}$ to the list V .
6. Let $\phi_y := (\phi_1, \dots, \phi_n)$, where $n = z - 1$.
7. Output j_x and ϕ_y .

Note that the operation $(*)$ can be performed efficiently due to Proposition 3.9.

The partial extraction algorithm is used in the directed transitive signature scheme to compress a composed signature. In fact, in the DTS scheme, we need to efficiently verify that ϕ_y does represent the ideal class $C_y \in \mathbb{G}$ defined by $\text{enc}(y)$. We don't know how to verify that solely from $\text{enc}(y)$ and the rational polynomial of ϕ_y over $\mathbb{Z}/N\mathbb{Z}$. (As mentioned in Remark 4.5, it is not clear how to recover the explicit ideal class C given an isogeny ϕ over $\mathbb{Z}/N\mathbb{Z}$. Of course we can recover the norm of the ideal from the degree of ϕ , but normally there are two ideals of the same prime norm.) Instead we provide a succinct proof on the correctness of the execution of Algorithm 4.15.

Let us recall the definition of a succinct non-interactive argument (SNARG).

Definition 4.16. Let $\mathcal{C} = \{C_\lambda : \{0, 1\}^{g(\lambda)} \times \{0, 1\}^{h(\lambda)} \rightarrow \{0, 1\}\}_{\lambda \in \mathbb{N}}$ be a family of Boolean circuits. A SNARG for the instance-witness relation defined by \mathcal{C} is a tuple of efficient algorithms $(\text{Gen}, \text{Prove}, \text{Verify})$ defined as:

- $\text{Gen}(1^\lambda)$ take the security parameter λ , outputs a common reference string CRS.
- $\text{Prove}(\text{CRS}, x, w)$ takes the CRS, the instance $x \in \{0, 1\}^{g(\lambda)}$, and a witness $w \in \{0, 1\}^{h(\lambda)}$, outputs a proof π .
- $\text{Verify}(\text{CRS}, x, \pi)$ takes as input an instance x , and a proof π , outputs 1 if it accepts the proof, and 0 otherwise.

It satisfies the following properties:

- *Completeness:* For all $x \in \{0, 1\}^{g(\lambda)}$, $w \in \{0, 1\}^{h(\lambda)}$ such that $C(x, w) = 1$:

$$\Pr[\text{Verify}(\text{CRS}, x, \text{Prove}(\text{CRS}, x, w)) = 1] = 1.$$
- *Soundness:* For all $x \in \{0, 1\}^{g(\lambda)}$ such that $C(x, w) = 0$ for all $w \in \{0, 1\}^{h(\lambda)}$, for all polynomially bounded cheating prover P^* :

$$\Pr[\text{Verify}(\text{CRS}, x, P^*(\text{CRS}, x)) = 1] \leq \text{negl}(\lambda).$$
- *Succinctness:* There exists a universal polynomial Q (independent of \mathcal{C}) such that Gen runs in time $Q(\lambda + \log |\mathcal{C}_\lambda|)$, the length of the proof output by Prove is bounded by $Q(\lambda + \log |\mathcal{C}_\lambda|)$, and Verify runs in time $Q(\lambda + g(\lambda) + \log |\mathcal{C}_\lambda|)$.

A construction of SNARG in the random oracle model is given by [Mic00].

With a SNARG in hand, we can add in Algorithm 4.15 a proof π for the instance $(\text{enc}(y), \phi_y)$ and statement “there exists an encoding $\text{enc}(x)$ such that ϕ_y is computed from running a circuit that instantiates Algorithm 4.15 on inputs $\text{enc}(x), \text{enc}(y)$ ”. Here the public parameter PP is hardcoded in the circuit. The encoding $\text{enc}(x)$ is the witness in the relation. The soundness of SNARG guarantees that ϕ_y is an isogeny that corresponds to an \mathcal{O} -ideal represented by $\text{enc}(y)$. The succinctness of SNARG guarantees that the length of the proof π is $\text{poly} \log(|\text{enc}(x)|)$ and the time to verify the proof is $\text{poly}(|\text{enc}(y)|, \log(|\text{enc}(x)|))$.

Alternative choices for the class group. Ideally we would like to efficiently sample an imaginary quadratic order \mathcal{O} of large discriminant D , together with the class number $h(D)$, a generation set $\{C_i\}$ and a short basis \mathbf{B} for $\Lambda_{\mathcal{O}}$, and two large primes p, q as well as curves $E_{0, \mathbb{F}_p}, E_{0, \mathbb{F}_q}$ such that $\text{End}(E_{0, \mathbb{F}_p}) \simeq \text{End}(E_{0, \mathbb{F}_q}) \simeq \mathcal{O}$. In Section 5 we will explain that if $|D|$ is polynomial then computing the group inversion takes polynomial time, so we are forced to choose a super-polynomially large $|D|$. On the other hand, there is no polynomial time solution for the task of choosing a square-free discriminant D with a super-polynomially large $h(D)$ (see, for instance, [HM00]).

In the basic setting of the parameter, we have described a solution where the resulting D is not square-free, of size $\approx \lambda^{O(\log \lambda)}$, and polynomially smooth. Here we provide the background if one would like to work with a square-free discriminant D . In this case, \mathcal{O} , the ring of integers of $\mathbb{Q}(\sqrt{D})$, is the maximal order of an imaginary quadratic field K .

According to the Cohen-Lenstra heuristics [CL84], about 97.7575% of the imaginary quadratic fields K have the odd part of $\mathcal{CL}(\mathcal{O}_K)$ cyclic. If we choose D such that $|D| \equiv 3 \pmod{4}$ and is a prime, then $h(D)$ is odd (by genus theory). So we might as well assume that $\mathcal{CL}(\mathcal{O}_K)$ is cyclic with odd order.

For a fixed discriminant D , heuristically about half of the primes ℓ satisfies $\left(\frac{D}{\ell}\right) = 1$, so there are polynomially many ideals of polynomially large norm that can be used in the generation set.

However, it is not clear how to efficiently choose p, q and curves $E_{0, \mathbb{F}_p}, E_{0, \mathbb{F}_q}$ such that the endomorphism rings of E_{0, \mathbb{F}_p} and E_{0, \mathbb{F}_q} have the given discriminant D of super-polynomial size. The classical CM method (cf. [LZ94] and more) requires computing the Hilbert class polynomial H_D , whose cost grows proportional to $|D|$. Let us remark that the CM method might be an overkill, since we do not need to specify the number of points $\#(E_{0, \mathbb{F}_p}(\mathbb{F}_p))$ and $\#(E_{0, \mathbb{F}_q}(\mathbb{F}_q))$. However, we do not know any other better methods.

5 Cryptanalysis

In this section we will discuss our cryptanalysis attempts, and the countermeasures.

Central to the security of our cryptosystem is the conjectured hardness of solving various problems over $\mathbb{Z}/N\mathbb{Z}$ without knowing the factors of N . So we start from the feasibility of performing several individual computational tasks over $\mathbb{Z}/N\mathbb{Z}$; then focus on the (ℓ, ℓ^2) -isogenous neighbor problem over $\mathbb{Z}/N\mathbb{Z}$, whose hardness is necessary for the security of our candidate TGII; finally address all the other attacks in the TGII construction.

5.1 The (in)feasibility of performing computations over $\mathbb{Z}/N\mathbb{Z}$

The task of finding roots of polynomials of degree $d \geq 2$ over $\mathbb{Z}/N\mathbb{Z}$ sits in the subroutines of many potential algorithms we need to consider, so let us begin with a clarification on the status of this problem. Currently, no polynomial time algorithm is known for solving this problem in general. The hardness of a few special instances have been extensively studied. They fall into the following three categories:

1. For certain families of polynomials, it is known that finding a root of them over $\mathbb{Z}/N\mathbb{Z}$ is as hard as factorizing N . For example, the family of polynomials $\{f_a(x) = x^2 - a\}_{a \in (\mathbb{Z}/N\mathbb{Z})^\times}$ whose potential roots are the solutions for the quadratic residue problem [Rab79].

2. There are families of polynomials, where finding at least one root is feasible. For example, if a root of a polynomial over $\mathbb{Z}/N\mathbb{Z}$ is known to be the same as the root over \mathbb{Q} , then we can use LLL [LLL82]; or if a root is known to be smaller than roughly $O(N^{1/d})$, then Coppersmith-type algorithms can be used to find such a root [Cop97]. But these families of polynomials only form a negligible portion of all the polynomials with polynomially bounded degrees.
3. The majority of the polynomials seem to live in the third case, where finding a root is conjectured to be hard, however the hardness is not known to be based on integer factorization. Among them, some specific families are even conjectured to be unlikely to have a reduction from integer factorization. For example, when $\gcd(3, \phi(N)) = 1$, the family of polynomials $\{f_a(x) = x^3 - a\}_{a \in (\mathbb{Z}/N\mathbb{Z})^\times}$ is conjectured to be hard to solve, and unlikely to be as hard as integer factorization [BV98].

5.1.1 Feasible information from a single j -invariant

Let $N = pq$, as before, where p and q are large primes. From any $j \in \mathbb{Z}/N\mathbb{Z}, j \neq 0, 1728$, we can easily find the coefficients a and b of the Weierstrass form of an elliptic curve $E(\mathbb{Z}/N\mathbb{Z})$ with $j(E) = j$ by computing $a = 3j(1728 - j), b = 2j(1728 - j)^2$. However, this method does not guarantee that the curve belongs to a specific isomorphism class (there are four of them). By choosing a value $u \in (\mathbb{Z}/N\mathbb{Z})^\times$ and let $a^* = u^4a, b^* = u^6b$ one gets the coefficient of another curve with the same j -invariant, each belonging to one of the four isomorphism classes.

On the other hand, choosing a curve over $\mathbb{Z}/N\mathbb{Z}$ with a given j -invariant together with a point on the curve seems tricky. Note that there are methods of sampling a curve with a known ℓ -torsion point over $\mathbb{Z}/N\mathbb{Z}$ (cf. [Kub76, Sut12a]). But with the additional constraint on the j -invariant, these methods seem to require solving a non-linear polynomial over $\mathbb{Z}/N\mathbb{Z}$.

Nevertheless, it is always feasible to choose a curve together with the x -coordinate of a point on it, since a random $x \in \mathbb{Z}/N\mathbb{Z}$ is the x -coordinate of some point on the curve with probability roughly $\frac{1}{2}$. It is also known that computing the multiples of a point P over $E(\mathbb{Z}/N\mathbb{Z})$ is feasible solely using the x -coordinate of P (cf. [Dem93]). The implication of this is that we should at the very least not give out the group orders of the curves involved in the scheme. More precisely, we should avoid the j -invariants corresponding to curves (or their twists) with polynomially smooth cardinalities over either \mathbb{F}_p or \mathbb{F}_q . Otherwise Lenstra's algorithm [Len87] can be used to factorize N .

In our application we also assume that the endomorphism rings of $E(\mathbb{F}_p)$ and $E(\mathbb{F}_q)$ are isomorphic and not given out to begin with. Computing the discriminant of $\mathcal{O} \simeq \text{End}(E(\mathbb{F}_p)) \simeq \text{End}(E(\mathbb{F}_q))$ or the number of points of E over $\mathbb{Z}/N\mathbb{Z}$ seems to be hard given only N and a j -invariant. In fact Kunihiro and Koyama (and others) have reduced factorizing N to computing the number of points of general elliptic curves over $\mathbb{Z}/N\mathbb{Z}$ [KK98]. However, these reductions are not efficient in the special case, where the endomorphism rings of $E(\mathbb{F}_p)$ and $E(\mathbb{F}_q)$ are required to be isomorphic. So, the result of [KK98] can be viewed as evidence that the polynomial time algorithms for counting points on elliptic curves over finite fields may fail over $\mathbb{Z}/N\mathbb{Z}$ without making use of the fact that the endomorphism rings of $E(\mathbb{F}_p)$ and $E(\mathbb{F}_q)$ are isomorphic.

5.1.2 Computing explicit isogenies over $\mathbb{Z}/N\mathbb{Z}$ given more than one j -invariant

Let ℓ be a prime. We will be concerned with degree ℓ isogenies. If we are only given a single j -invariant $j_1 \in \mathbb{Z}/N\mathbb{Z}$, then finding an integer j_2 such that $\Phi_\ell(j_1, j_2) = 0 \pmod{N}$ seems hard.

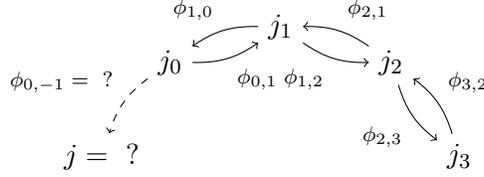


Figure 6: A pictorial description of the (ℓ, ℓ^2) -isogenous neighbor problem.

We remark that Theorem 3.3 does not guarantee that finding j_2 is as hard as factoring when the endomorphism rings of $E(\mathbb{F}_p)$ and $E(\mathbb{F}_q)$ are isomorphic. However, as of now, we do not know how to make use of the condition that the endomorphism rings are isomorphic to mount an attack on the problem.

Of course, in the construction of a TGII, we are not only given a single j -invariant, but many j -invariants with each neighboring pair of them satisfying the ℓ^{th} modular polynomial, a polynomial degree $\ell + 1$. We will study what other information can be extracted from these neighboring j -invariants.

In Proposition 3.9, we have explained that given two integers $j_1, j_2 \in \mathbb{Z}/N\mathbb{Z}$ such that $\Phi_\ell(j_1, j_2) = 0$ over $\mathbb{Z}/N\mathbb{Z}$, the elliptic curves E_1, E_2 , and the kernel polynomial $h(x)$ of an isogeny ϕ from E_1, E_2 can be computed in time polynomial in $\ell, \log(N)$. However, it is not clear how to use the explicit expression of ϕ to break factoring or solve the inversion problem.

A natural next step is to recover a point in the kernel of ϕ , but it is also not clear how to recover even the x -coordinate of a point in the kernel when $\ell \geq 5$. For $\ell = 3$, on the other hand, the kernel polynomial does reveal the x -coordinate of a point P in the kernel $G \subset E_1[3]$ (note that $h(\cdot)$ is of degree 1 in this particular case). But revealing the x -coordinate of a point $P \in E_1[3]$ does not immediately break factoring, since $3P$ is O over both \mathbb{F}_p and \mathbb{F}_q . At this moment we do not know of a full attack from a point in $\ker(\phi)$. Nevertheless, we still choose to take an additional safeguard by avoiding the use of 3-isogenies since it reveals the x -coordinate of a point in $E_1[3]$, and many operations on elliptic curves are feasible given the x -coordinate of a point.

5.2 Tackling the (ℓ, ℓ^2) -isogenous neighbor problem

The hardness of infeasible inversion of our group representation in fact relies on the hardness of the following generalization of the (ℓ, ℓ^2) -isogenous neighbor problem (cf. Definition 3.6). Let the modulus $N = pq$, where the large prime factors p and q are hidden. Let ℓ be the degree of the isogenies. For a polynomially large $k \in \mathbb{N}$ and a given a sequence of integers $j_0, j_1, j_2, \dots, j_k$ such that $\Phi_\ell(j_{i-1}, j_i) = 0 \pmod{N}$ for all $i \in [k]$, the problem asks to find an integer j_{-1} such that $\Phi_{\ell^{i+1}}(j_{-1}, j_i) = 0 \pmod{N}$ for all $i \in [k]$. In addition, for all $i \in \{0, 1, \dots, k\}$ the endomorphism rings of $E_{i, \mathbb{F}_p}, E_{i, \mathbb{F}_q}$ are isomorphic to an imaginary quadratic order \mathcal{O} (\mathcal{O} is supposed to be hidden for reasons to be explained later). See Figure 6 for the pictorial description of the problem.

From the discussion in § 5.1.2 we know that it is feasible to compute the kernel polynomials of the isogenies $\phi_{i, i+1}$ and their duals $\phi_{i+1, i}$, for $i = 0, \dots, k - 1$. Denote these kernel polynomials by $h_{i, i+1}$ and $h_{i+1, i}$ respectively. If one can compute the kernel polynomial $h_{0, -1}$ of the isogeny $\phi_{0, -1}$ that maps from j_0 to j one can then solve the (ℓ, ℓ^2) -problem. Since $h_{0, 1}$ can be recovered, there is a chance of obtaining $h_{0, -1}$ from $h_{0, 1}$. As mentioned in Remark 2.3, over a finite field, the two kernel polynomials $h_{0, -1}$ and $h_{0, 1}$ of the two horizontal isogenies can be explicitly related via the Frobenius endomorphism. However it is not clear how to use the relation over $\mathbb{Z}/N\mathbb{Z}$.

Another attempt is to see if $\phi_{1,0}(\ker(\phi_{1,2}))$ gives $\ker(\phi_{0,-1})$ (the map $\phi_{1,0}(\ker(\phi_{1,2}))$ can be computed using Newton's identities). However, applying $\phi_{1,0}$ on the kernel subgroup $G_{1,2}$ of $\phi_{1,2}$ gives $\ker(\phi_{0,1})$, since $\phi_{0,1} \circ \phi_{1,0}(G_{1,2}) = [\ell]G_{1,2} = O$, so it does not give the desired kernel subgroup.

5.2.1 Hilbert class polynomial attack

Let D be the discriminant of the imaginary quadratic order \mathcal{O} that we are working with. If computing the Hilbert class polynomial H_D is feasible, then we can solve the (ℓ, ℓ^2) -isogenous neighbor problem. Given j_0, j_1 such that $\Phi_\ell(j_0, j_1) = 0$, compute the polynomial $\gamma(x)$,

$$\gamma(x) := \gcd(\Phi_\ell(j_0, x), \Phi_{\ell^2}(j_1, x), H_D(x)) \in (\mathbb{Z}/N\mathbb{Z})[x].$$

The gcd of $\Phi_\ell(j_0, x)$ and $\Phi_{\ell^2}(j_1, x)$ gives a polynomial of degree ℓ . The potential root they share with $H_D(x)$ is the only one with the same endomorphism ring with j_0 and j_1 , which is j_{-1} . So $\gamma(x)$ is a linear function.

This attack is ineffective when D is super-polynomial.

5.2.2 Deciding the direction of an isogeny on the volcano

Fix an isogeny volcano $G_\ell(\mathbb{F}_q)$ over a finite field \mathbb{F}_q . As we mentioned, given a j -invariant j on $G_\ell(\mathbb{F}_q)$ there are efficient algorithms that output all the ℓ -isogenous neighbors of j . When there are 1 or 2 neighbors, we know j is at the bottom of the volcano (the case of 2 neighbors corresponding to when the volcano consists just of the surface). When there are more than 2 (i.e. $\ell + 1$) neighbors, can we decide which isogeny is ascending, horizontal, or descending? The method mentioned in [Koh96, FM02, Sut13a] takes a trial and error approach. It picks a random neighbor and goes forward, until the path reaches one of the terminating conditions. For instance, if it reaches a point where there is only one neighbor, then that means the path is descending; if the path forms a loop, or takes longer than the estimated maximum depth of the volcano, then the initial step is ascending or horizontal.

The only algorithm that is able to produce an isogeny with a designated direction is given by Ionica and Joux [IJ13]. They recognize an invariant related to the group structure of the curves lying on the same level of the volcano. The highlevel structure of the algorithm is as follows:

1. First decide the group structure via a pairing ([IJ13] uses reduced Tate pairing).
2. Then find the kernel subgroup with a property that depends on whether the desired isogeny is ascending, descending, or horizontal.

Either steps seem to carry out over $\mathbb{Z}/N\mathbb{Z}$. One of the main barriers is to compute (even the x -coordinate of) a point that sits in the specific subgroup of the curve.

The precise descriptions of the invariant and the algorithm are rather technical. So we only sketch the main theorem from [IJ13], skipping many technical details. Let $n \geq 0$, let $E[\ell^n](\mathbb{F}_{q^k})$ be the subgroup of points of order ℓ^n defined over an extension field over \mathbb{F}_q . Let $E[\ell^\infty](\mathbb{F}_q)$ be the ℓ -Sylow subgroup of $E(\mathbb{F}_q)$.

Let m be an integer such that $m \nmid \#E(\mathbb{F}_q)$. Let k be the embedding degree, i.e. the smallest integer that $m \mid q^k - 1$. Let $T_m : E[m](\mathbb{F}_{q^k}) \times E(\mathbb{F}_{q^k})/mE(\mathbb{F}_{q^k}) \rightarrow \mu_m$ be the reduced Tate pairing, where μ_m denotes the m^{th} roots of unity. Furthermore, define the symmetric pairing

$$S(P, Q) = (T_{\ell^n}(P, Q) T_{\ell^n}(Q, P))^{1/2},$$

and call $S(P, P) = T_{\ell^n}(P, P)$ the self-pairing of P .

Theorem 5.1 (The main theorem of [IJ13], informally). *Let E be an elliptic curve defined a finite field \mathbb{F}_q and let $E[\ell^\infty](\mathbb{F}_q)$ be isomorphic to $\mathbb{Z}/\ell^{n_1}\mathbb{Z} \times \mathbb{Z}/\ell^{n_2}\mathbb{Z}$ with $n_1 \geq n_2 \geq 1$. Then*

- *Suppose P is a ℓ^{n_2} -torsion point such that $T_{\ell^{n_2}}(P, P)$ is a primitive ℓ^{n_2} th root of unity. Then the ℓ -isogeny whose kernel is generated by $\ell^{n_2-1}P$ is descending.*
- *Suppose certain condition holds, and let P be a ℓ^{n_2} -torsion point with degenerate self-pairing. Then the ℓ -isogeny whose kernel is generated by $\ell^{n_2-1}P$ is either ascending or horizontal. Moreover, for any ℓ^{n_2} -torsion point Q whose self-pairing is non-degenerate, the isogeny with kernel spanned by $\ell^{n_2-1}Q$ is descending.*

Carrying out the algorithm of finding an isogeny with a specific direction (say horizontal) requires finding (even merely the x -coordinate of) a point in the designated subgroup. Over a finite field, such points can be found efficiently by picking a random point R and compute $P = \frac{\#(E(\mathbb{F}_q))}{r}R$, where r is the order of the designated subgroup; then test if the candidate point P lies in the correct subgroup by taking pairing. Over $\mathbb{Z}/N\mathbb{Z}$ it seems hard even to find a point with a specific order since we do not know $\#E(\mathbb{Z}/N\mathbb{Z})$. Even if we are able to find points in desired subgroups, there are additional technicalities such as computing pairing over $\mathbb{Z}/N\mathbb{Z}$ (cf. [GM05]).

In our application we are mostly interested in finding the unvisited neighbor over $\mathbb{Z}/N\mathbb{Z}$ that lies on the same level of the volcano, which equals to finding the horizontal isogeny of a given curve. As we conjectured, the algorithm in [IJ13] do not extend to $\mathbb{Z}/N\mathbb{Z}$. So if the task of finding an isogeny with a designated direction is feasible over $\mathbb{Z}/N\mathbb{Z}$, then it is likely to imply a new algorithm of the same task over the finite field, which seems to be challenging on its own.

5.2.3 More about modular curves and characteristic zero attacks

Given j , solving $\Phi_\ell(j, x)$ is not the only way to find the j -invariants of the ℓ -isogenous curves. Alternative complex analytic (i.e. characteristic zero) methods have been discussed, for instance, in [E+98, Section 3]. However, these methods all involve solving polynomials of degree ≥ 2 to get started.

As mentioned in Section 2.2, the curve $\mathbb{H}/\Gamma_0(\ell)$ parameterizes pairs of elliptic curves over \mathbb{C} related by a cyclic ℓ -isogeny. The (ℓ, ℓ^2) -isogenous neighbor problem, on the other hand, concerns curves that are horizontally ℓ -isogenous, i.e. ℓ -isogenous and have the same endomorphism ring. To avoid an attack through characteristic zero techniques, we make sure that there is no immediate quotient of \mathbb{H} that parametrizes curves which are related with an ℓ -isogeny and have the same endomorphism ring. Below, we first go over the well-known moduli description of modular curves⁴ to make sure that they don't lead to an immediate attack, and then show that there is indeed no quotient of \mathbb{H} between $\mathbb{H}/\mathrm{SL}_2(\mathbb{Z})$ and $\mathbb{H}/\Gamma_0(\ell)$, so we don't have to worry about possible attacks on that end.

Let $\Gamma := \mathrm{SL}_2(\mathbb{Z})$, and let $\Gamma(\ell)$ and $\Gamma_1(\ell)$ denote the congruence subgroups,

$$\Gamma(\ell) := \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{SL}_2(\mathbb{Z}) \mid \begin{pmatrix} a & b \\ c & d \end{pmatrix} \equiv \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \pmod{\ell} \right\},$$

⁴Recall that a modular curve is a quotient of the extended upper half plane by a congruence subgroup, and a congruence subgroup is a subgroup of $\mathrm{SL}_2(\mathbb{Z})$, which contains $\Gamma(\ell)$.

$$\Gamma_1(\ell) := \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{SL}_2(\mathbb{Z}) \mid \begin{pmatrix} a & b \\ c & d \end{pmatrix} \equiv \begin{pmatrix} 1 & * \\ 0 & 1 \end{pmatrix} \pmod{\ell} \right\}.$$

It is well-known that the curves $\mathbb{H}/\Gamma_1(\ell)$ and $\mathbb{H}/\Gamma(\ell)$ parametrize elliptic curves with extra data on their ℓ -torsion (cf. [Koh96]). $\mathbb{H}/\Gamma_1(\ell)$ parametrizes (E, P) , where P is a point on E having order exactly ℓ , and $\mathbb{H}/\Gamma(\ell)$ parametrizes triples (E, P, Q) , where $E[\ell] = \langle P, Q \rangle$ and they have a fixed Weil pairing. These curves carry more information than the ℓ -isogenous relation and they are not immediately helpful for solving the (ℓ, ℓ^2) -isogenous neighbor problem.

As for the quotients between $\mathbb{H}/\mathrm{SL}_2(\mathbb{Z})$ and $\mathbb{H}/\Gamma_0(\ell)$, the following lemma shows that there are indeed none.

Lemma 5.2. *Let ℓ be a prime. If $H \leq \Gamma$ is such that $\Gamma_0(\ell) \leq H \leq \Gamma$, then either $H = \Gamma_0(\ell)$ or $H = \Gamma$.*

Proof. Let $\sigma_1 = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$, $\sigma_2 = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$, $\sigma_3 = \sigma_1\sigma_2^{-1}$, and recall that $\mathrm{SL}_2(\mathbb{Z}/\ell\mathbb{Z}) = \langle \sigma_1, \sigma_2 \rangle = \langle \sigma_1, \sigma_3 \rangle$. Recall that the natural projection $\pi : \Gamma \rightarrow \mathrm{SL}_2(\mathbb{Z}/\ell\mathbb{Z})$ is surjective. Assume that $H \neq \Gamma_0(\ell)$. This implies that $\pi(H) = \mathrm{SL}_2(\mathbb{Z}/\ell\mathbb{Z})$ (we shall give a proof below). Assuming this claim for the moment let $g \in \Gamma \setminus H$. Since $\pi(\Gamma) = \pi(H)$ there exists $h \in H$ such that $\pi(g) = \pi(h)$. Therefore, $gh^{-1} \in \ker(\pi) = \Gamma(\ell) \subset H$. Therefore, $g \in H$ and $\Gamma = H$.

To see that $\pi(\Gamma) = \pi(H)$, first note that since $\Gamma_0(\ell) \subset H$ we have all the upper triangular matrices in $\pi(H)$. Next, let $h = \begin{pmatrix} h_1 & h_2 \\ h_3 & h_4 \end{pmatrix} \in H \setminus \Gamma_0(\ell)$ such that $\pi(h) = \begin{pmatrix} \bar{h}_1 & \bar{h}_2 \\ \bar{h}_3 & \bar{h}_4 \end{pmatrix} \in \pi(H) \setminus \pi(\Gamma_0(\ell))$ (note that this difference is non-empty since otherwise $\Gamma_0(\ell) = H$).

We have two cases depending on $\bar{h}_1 = 0$ or not. If $\bar{h}_1 = 0$ then $\bar{h}_3 \neq 0$ and $\sigma_3 = \begin{pmatrix} \bar{h}_3^{-1} & \bar{h}_4 \\ 0 & \bar{h}_3 \end{pmatrix} \bar{h}^{-1} \in \pi(H)$. On the other hand, if $\bar{h}_1 \neq 0$ multiplying on the right by $\begin{pmatrix} \bar{h}_1^{-1} & -\bar{h}_2 \\ 0 & \bar{h}_1 \end{pmatrix} \in \pi(H)$ we see that $\begin{pmatrix} 1 & 0 \\ \bar{h}_3\bar{h}_1^{-1} & 1 \end{pmatrix} \in \pi(H)$. For any integer m , the m 'th power of this matrix is $\begin{pmatrix} 1 & 0 \\ m\bar{h}_3\bar{h}_1^{-1} & 1 \end{pmatrix} \in \pi(H)$. Taking $m \equiv \bar{h}_1\bar{h}_3^{-1}$ shows that $\sigma_2 \in \pi(H)$. This shows that $\pi(H) = \mathrm{SL}_2(\mathbb{Z}/\ell\mathbb{Z})$. \square

Let us also remark that for special values of ℓ the (ℓ, ℓ^2) -isogenous neighbor problem may be more prone to characteristic zero attacks (although we do not know of such an attack). For instance, for those ℓ for which $\mathbb{H}/\Gamma_0(\ell)$ (more precisely, its compactification $X_0(\ell)$) is genus 0 or 1, it (may) have many rational points. These points, in turn, can be used to get points over $\mathbb{Z}/N\mathbb{Z}$ without knowing the factorization of N . Nevertheless, we may just avoid these values in the system. As mentioned above, we currently do not see an attack based on this. This point is brought up just as an extra precautionary measure.

5.3 Cryptanalysis of the candidate group with infeasible inversion

We now cryptanalyze the concrete candidate TGII. Recall the format of an encoding of a group element x from Definition 4.2:

$$\mathrm{enc}(x) = (L_x; T_{x,1}, \dots, T_{x,w_x}) = ((p_{x,1}, \dots, p_{x,w_x}); (j_{x,1,1}, \dots, j_{x,1,e_{x,1}}), \dots, (j_{x,w_x,1}, \dots, j_{x,w_x,e_{x,w_x}})).$$

The ‘‘exponent vector’’ $\mathbf{e}_x \in \mathbb{Z}^{w_x}$ can be read from the encoding as $\mathbf{e}_x = (|T_{x,1}|, \dots, |T_{x,w_x}|)$.

We assume polynomially many composable encodings are published in the applications of a TGII. In down-to-earth terms it means the adversary is presented with polynomially many j -invariants on

the crater of a volcano, and the explicit isogenies between each pair of the neighboring j -invariants (due to Proposition 3.9).

We will be considering the following model on the adversary’s attacking strategy:

Definition 5.3 (The GCD attack model). *In the GCD attack model, the adversary is allowed to try to find the inverse of a target group element only by executing the unit gcd operation $\text{gcd.op}(\text{PP}, \ell_1, \ell_2; j_1, j_2)$ given in Algorithm 4.9 for polynomially many steps, where $\ell_1, \ell_2; j_1, j_2$ are from the published encodings or obtained from the previous executions of the gcd evaluations.*

We do not know how to prove the construction of TGII is secure even if the adversary is restricted to attack in the GCD model. Our cryptanalysis attempts can be classified as showing (1) how to prevent the attacks that obey the GCD evaluation law (mainly in §5.3.1 and §5.3.2); (2) how to prevent the other attack approaches.

5.3.1 Preventing the trivial leakage of inverses

In applications we are often required to publish the encodings of elements that are related in some way. A typical case is the following: for $x, y \in \mathcal{CL}(\mathcal{O})$, the scheme may require publishing the encodings of x and $z = y \circ x^{-1}$ without revealing a valid encoding of x^{-1} . As a toy example, let $x = [(p_x, b_x, \cdot)]$, $y = [(p_y, b_y, \cdot)]$, where p_x and p_y are distinct primes. Let j_0 , the j -invariant of a curve E_0 , represent the identity element in the public parameter. Let $((p_x); (j_x))$ be a composable encoding of x and $((p_y); (j_y))$ be a composable encoding of y .

Naively, a composable encoding of $z = y \circ x^{-1}$ could then be $((p_x, p_y); (j_{x^{-1}}, j_y))$, where $j_{x^{-1}}$ is the j -invariant of $E_{x^{-1}} = x^{-1}E_0$. Note, however, that $((p_x); (j_{x^{-1}}))$ is a valid encoding of x^{-1} . In other words such an encoding of $y \circ x^{-1}$ trivially reveals the encoding of x^{-1} .

One way of generating an encoding of $z = y \circ x^{-1}$ without trivially revealing $j_{x^{-1}}$ is to first pick a generator set of ideals where the norms of the ideals are coprime to p_x and p_y , then solve the discrete-log of z over these generators to compute the composable encoding. This is the approach we take in this paper.

5.3.2 Parallelogram attack

In the applications we are often required to publish the composable encodings of group elements a, b, c such that $a \circ b = c$. If the degrees of the three encodings are coprime, or in general, when $\text{enc}(a)$, $\text{enc}(b)$ and $\text{enc}(c)$ are pairwise composable, then we can recover the encodings of a^{-1} , b^{-1} , and c^{-1} using the following “parallelogram attack”. This is a non-trivial attack which obeys the gcd evaluation law in Definition 5.3.

Let us illustrate the attack via the examples in Figure 7, where the solid arrows represent the isogenies that are given as the inputs (the j -invariants of the target curves are written at the head of the arrows, their positions do not follow the relative positions on the volcano; the degree of the isogeny is written on the arrow); the dashed lines and the j -invariants on those lines are obtained from the gcd evaluation law.

For simplicity let us first look at the example in the left figure. Let composable encodings of a, b, c be simply given by $(\ell_1; (j_1))$, $(\ell_2; (j_2))$, $(\ell_3; (j_3))$, where ℓ_1, ℓ_2, ℓ_3 are polynomial and pairwise coprime. A composable encoding of b^{-1} then can be written as $(\ell_2; (j_4))$, where j_4 is the root of the

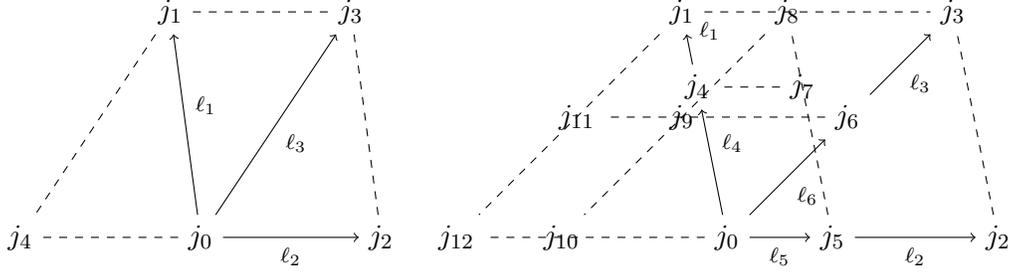


Figure 7: The parallelogram attack.

linear equation $f(x) = \gcd(\Phi_{\ell_3}(j_1, x), \Phi_{\ell_2}(j_0, x))$. This is due to the relation $c \circ b^{-1} = a$, which, in particular implies that j_1 and j_4 are connected by an isogeny of degree ℓ_3 .

The simple attack above uses the fact that the degrees of the entire encodings of a , b and c are polynomial. Let us use the example in the right figure to illustrate that even if the encodings are composed of many polynomial degree isogenies (so that the total degrees may be super-polynomial), the attack is still effective. The idea is to view the composition law as filling the missing edges of a parallelogram given the j -invariants on a pair of adjacent edges. The final goal is to find the missing corner j_{12} in the parallelogram $j_0 - j_3 - j_1 - j_{12}$. To arrive there we need the j -invariants on a pair of adjacent edges to begin with, so we first have to fill the j -invariants on, for instance, the edge $j_1 - j_3$. Therefore, as the first step, we consider the parallelogram $j_0 - j_2 - j_3 - j_1$. To fill the j -invariants on the edge $j_1 - j_3$, we first compute j_7 as the root of $f_7(x) = \gcd(\Phi_{\ell_4}(j_5, x), \Phi_{\ell_5}(j_4, x))$, then compute j_8 as the root of $f_8(x) = \gcd(\Phi_{\ell_1}(j_7, x), \Phi_{\ell_5}(j_1, x))$ (the polynomials f_7, f_8 are linear since the degrees of $\text{enc}(a)$ and $\text{enc}(b)$ are coprime). In the second step, we consider the parallelogram $j_0 - j_3 - j_1 - j_{12}$. To find j_{12} we use the gcd evaluation law to find $j_9, j_{10}, j_{11}, j_{12}$ one-by-one (using the condition that the degrees of $\text{enc}(c)$ and $\text{enc}(b)$ are coprime).

In the previous examples we illustrated that the attack is applicable when the degrees of $\text{enc}(a)$, $\text{enc}(b)$, $\text{enc}(c)$ are coprime. Let us also remark that the attack applies as long as $\text{enc}(a)$, $\text{enc}(b)$, $\text{enc}(c)$ are pairwise composable, i.e. they don't have to be of coprime degrees, which happens when the ladders in the general algorithms are applied (cf. Algorithm 4.13 in §4.3).

The parallelogram attack is very powerful, in the sense that it is not preventable when application requires to publish the composable encodings of a, b, c such that $a \circ b = c$, and $\text{enc}(a), \text{enc}(b), \text{enc}(c)$ to be pairwise composable. However, the parallelogram attack does not seem to work when 2 out of the 3 pairs of $\text{enc}(a), \text{enc}(b)$ and $\text{enc}(c)$ are not composable. In the applications of directed transitive signature and broadcast encryption, there are encodings of a, b, c such that $a \circ b = c$. Luckily, only one pair of the encodings among the three has to be composable to provide the necessary functionalities of these applications. We will explicitly mention how to choose the prime factors in the degrees of these encodings to prevent the parallelogram attack.

5.3.3 Hiding the class group invariants: why and how

The hardness of discrete-log problem over $\mathcal{CL}(D)$ is necessary for the infeasibility of inversion⁵. Recall from Lemma 2.5 that, under GRH, the graph $\mathcal{G}_{\mathcal{O}, m}(\mathbf{k})$ for a sufficiently large m is an expander, which means it is reasonable to assume that the closure of the composition of polynomially many j -invariants covers all the $h(D)$ j -invariants. So finding a composition of the published j -invariants

⁵The actual group with infeasible inversion is $\mathcal{CL}(D)_{\text{odd}}$; here we discuss over $\mathcal{CL}(D)$ for notational simplicity.

that reaches a specific vertex (e.g. the vertex that represents the inverse of some encoded element) in the isogeny graph is feasible by solving the discrete-log problem over $\mathcal{CL}(D)$.

However, in the basic setting of the parameters, the class number $h(D)$ is polynomially smooth, so the discrete-log problem over $\mathcal{CL}(D)$ can be solved in polynomial time once $h(D)$ is given, and $h(D)$ can be recovered from D or any basis of a lattice Λ of dimension w such that $\mathbb{Z}^w/\Lambda \simeq \mathcal{CL}(D)$. So we do need to hide the discriminant D , the class number $h(D)$, and any lattice Λ defined above.

If self-composition of an encoding is feasible, then one can efficiently guess all the polynomially smooth factors of $h(D)$. However for our construction self-composition is infeasible, due to the hardness of the (ℓ, ℓ^2) -isogenous neighbor problem. Nevertheless, one can still attack by first guessing D or $h(D)$, which takes $\lambda^{O(\lambda)}$ time according to the current setting of parameter.

We remark that if it is feasible to choose a super-polynomially large square-free discriminant D with a trapdoor τ_D so that solving discrete-log over $\mathcal{CL}(D)$ given D is hard but feasible given the trapdoor τ_D , then we might be able to maintain the security of the system even if D and $h(D)$ are public. This would avoid all the complications in hiding the class group invariants. However currently we do not know of such a method, so we need to hide the class group invariants.

The possibility of recovering the discriminant D . Recall that for an elliptic curve over a finite field \mathbb{F}_q , the discriminant D can be obtained by first computing the trace t of Frobenius, and then computing the integers v and D_0 such that $v^2 D_0 = t^2 - 4q$, and D_0 is square-free. Then $D = u^2 D_0$, where $u \mid v$. When v is smooth D can be recovered efficiently.

Over $\mathbb{Z}/N\mathbb{Z}$, on the other hand, since we lose the notion of the Frobenius automorphism, we do not know how to apply the previous algorithm. Also, just a set of j -invariants over $\mathbb{Z}/N\mathbb{Z}$, corresponding to curves with the same endomorphism rings over \mathbb{F}_p and \mathbb{F}_q , does not seem to allow us to recover the discriminant.

However, we do know that the range of D is bounded by $|D| \leq 4p$ and $|D| \leq 4q$, so that $|D| < \sqrt{N}$. From the encodings we also learn the set $\mathcal{P}_B = \left\{ p \leq B \mid p \text{ is a prime, } \left(\frac{D}{p}\right) = 1 \right\}$, where $B \in \text{poly}(\lambda)$. This brings us to the following problem of independent interest:

Definition 5.4. *Given an integer N and a set of primes $\mathcal{P} = \{p_1, \dots, p_m\}$, find a negative value D such that $|D| < \sqrt{N}$ and $\left(\frac{D}{p}\right) = 1$ for all $p \in \mathcal{P}$.*

Note that each condition $\left(\frac{D}{p}\right) = \pm 1$ cuts the possible D 's roughly by half. Hence, for a sequence $\epsilon_1, \epsilon_2, \dots, \epsilon_{\omega(\log(N))}$, $\epsilon_j \in \{\pm 1\}$, with high probability there is at most one D with $|D| < \sqrt{N}$ that satisfies $\epsilon_j = \left(\frac{D}{p_j}\right)$. To find such a D , on the other hand, is a separate problem and the only known methods require $\sqrt{|D|}$ of the values of the sequence $\{\epsilon_j\}$ (cf. pg. 6 of [Hof14], also see pg. 14 of loc. cit. and [GH93] for the analogue of the same problem in the context of modular forms)

We also remark that the following similar problem, first mentioned by Damgård [Dam88], is conjectured to be hard:

Definition 5.5 (Problem P1 in [Dam88]). *The Legendre sequence with length ℓ and starting point a is the ± 1 sequence*

$$L = \left(\frac{a}{p}\right), \left(\frac{a+1}{p}\right), \dots, \left(\frac{a+\ell}{p}\right).$$

Given L (with a polynomial ℓ) but not a and p , the problem asks to determine $\left(\frac{a+\ell+1}{p}\right)$.

Even if Problem 5.4 can be solved efficiently, an obvious safeguard is to choose less than $\log(|D|)$ many primes for the degrees of the isogenies involved in the composable encodings. However, it is not always possible to restrict the number of distinct prime degrees in applications. In that case, we can apply another safeguard by first choosing an enlarged set of k polynomially large primes $R = \{p_1, \dots, p_k\}$, and letting R' be a random subset of R . We can then set $D = (\prod_{p \in R'} p)^2 D_0$, where D_0 is a polynomial size fundamental discriminant. Then in the composable encodings, we avoid the primes from the entire set R as the degrees of the isogenies.

The possibility of revealing Λ . Revealing any full-rank (not necessarily short) basis \mathbf{B} of a relation lattice Λ of dimension w such that $\mathbb{Z}^w / \Lambda \simeq \mathcal{CL}(D)$ implies the disclosure of the class number $h(\mathcal{O}) = |\det(\mathbf{B})|$. So we should prevent leaking any full-rank bases of such lattices.

An immediate consequence is that we cannot give out many non-trivial encodings of $1_{\mathbb{G}}$ (where $\mathbb{G} = \mathcal{CL}(\mathcal{O})$) that form a full-rank basis of a potential relation lattice Λ . Recall that the trivial (i.e. canonical) encoding of $1_{\mathbb{G}}$ is j_0 . A non-trivial encoding of $1_{\mathbb{G}}$ can be obtained by composing encodings which lead to a non-zero exponent vector. As an example, suppose that the encodings of $x, y, z \in \mathcal{CL}(\mathcal{O})$ share the same prime generation set of norm from L , and $x \circ y \circ z = 1_{\mathbb{G}}$, and denote them by $\text{enc}(v) = (L; T_{v,1}, \dots, T_{v,|L|})$, $v \in \{x, y, z\}$. Then, the following vector is in Λ

$$\mathbf{e}_{x \circ y \circ z} = (|T_{x,1}| + |T_{y,1}| + |T_{z,1}|, \dots, |T_{x,|L|}| + |T_{y,|L|}| + |T_{z,|L|}|).$$

Note that in the example we are not required to compute the composable encoding of $x \circ y \circ z$. We only need to read off the exponents from the lengths of $T_{*,*}$.

In the applications we do face the situation where the general security setting does not restrict the number of non-trivial encodings of $1_{\mathbb{G}}$. A countermeasure is to enforce each distinct non-zero encoding of $1_{\mathbb{G}}$ to have an new ideal with distinct prime norm, so that the dimension of the potential basis is larger than the possible number of vectors (relations) to be collected. We will illustrate how to apply this countermeasure in the applications.

Even for the encodings of non-identity elements, sampling the vector \mathbf{e} from an arbitrary distribution over the cosets of Λ might leak a basis. For example, [NR06] shows that if there are enough short vectors from the parallelepiped of a short basis, then one can find the parallelepiped and, therefore, recover the basis.

The countermeasure is to sample the vector $\mathbf{e}_x \in \mathbb{Z}^{w_x}$ from the discrete-Gaussian distribution [GPV08]. The sampler is known of being basis-independent.

Formally, for any $\sigma \in \mathbb{R}^+$, $\mathbf{c} \in \mathbb{R}^n$, define the (non-normalized) Gaussian function with center \mathbf{c} and standard deviation σ for any $\mathbf{x} \in \mathbb{R}^n$ as $\rho_{\sigma, \mathbf{c}}(\mathbf{x}) = e^{-\pi \|\mathbf{x} - \mathbf{c}\|^2 / \sigma^2}$. For any n -dimensional lattice Λ , define the discrete Gaussian distribution over Λ as:

$$\forall \mathbf{x} \in \Lambda, D_{\Lambda, \sigma, \mathbf{c}}(\mathbf{x}) = \frac{\rho_{\sigma, \mathbf{c}}(\mathbf{x})}{\rho_{\sigma, \mathbf{c}}(\Lambda)},$$

where $\rho_{\sigma, \mathbf{c}}(\Lambda) := \sum_{\mathbf{y} \in \Lambda} \rho_{\sigma, \mathbf{c}}(\mathbf{y})$ is the normalization factor.

Lemma 5.6 ([MR07]). *Let \mathbf{B} be a basis of an n -dimensional lattice Λ , and let $\sigma \geq \|\tilde{\mathbf{B}}\| \cdot \omega(\log n)$, then $\Pr_{\mathbf{x} \leftarrow D_{\Lambda, \sigma, \mathbf{0}}}[\|\mathbf{x}\| \geq \sigma \cdot \sqrt{n} \vee \mathbf{x} = \mathbf{0}] \leq \text{negl}(n)$.*

Lemma 5.7 ([GPV08, BLP⁺13]). *There is a p.p.t. algorithm that, given a basis \mathbf{B} of an n -dimensional lattice $\Lambda(\mathbf{B})$, $\mathbf{c} \in \mathbb{R}^n$, and $\sigma \geq \|\tilde{\mathbf{B}}\| \sqrt{\ln(2n+4)}/\pi$, outputs a sample from $D_{\Lambda, \sigma, \mathbf{c}}$.*

Note, however, that sampling from discrete-Gaussian distribution could lead to a vector with negative entries. Having negative entries as the exponent vector might trivially leak the inverse of other encodings (as mentioned in § 5.3.1). There are at least two solutions to this problem. One is to sample a vector \mathbf{v}_0 with positive large entries in Λ , then add \mathbf{v}_0 on any other vectors \mathbf{v}' sampled from discrete-Gaussian so that $\mathbf{v}_0 + \mathbf{v}'$ have all positive entries (this solution does leak one vector \mathbf{v}_0 in Λ). The other solution is to pick at least one new prime ideal in the generation set of each composable encoding, so that the inverse won't be trivially obtained from the other encodings.

The possibility of leaking $h(D)$ from other sources. We have discussed the possibilities of leaking D and a basis of $\Lambda_{\mathcal{O}}$ from the encodings and the corresponding countermeasures. It remains to check whether there are other possibilities of leaking $h(D)$.

Again recall that in the basic parameter setting, $h(D)$ is set to be polynomially smooth. The immediate consequence is that in the encodings, we shall not use prime degrees ℓ such that the order of the ideal class $[(\ell, b, \cdot)]$ is polynomially large in $\mathcal{CL}(D)$, to avoid the risk of unnecessarily leaking any factors of $h(D)$.

Additionally, we ask:

1. Given an encoding, is it feasible to recognize that it encodes an element of polynomial order in the group $\mathcal{CL}(\mathcal{O})$?
2. Given an encoding of an element that is known to be of polynomial order, is the order explicitly revealed (instead of having an $1/\text{poly}(\lambda)$ chance of being guessed correctly)?

If self-composition is feasible, then the answers to both questions are yes. But for our construction, self-composition of the encoded group elements is infeasible.

Denote the degree of an encoding $\text{enc}(x)$ by d and the order of x in $\mathcal{CL}(\mathcal{O})$ by r . Let the canonical encoding of $\text{enc}(x)$ be j_x . In the special case where d^r is polynomially large, then we can efficiently recognize that $\text{enc}(x)$ has order r by first guessing d^r and then testing whether $\Phi_d(j_0, j_x) = 0 \pmod{N}$, and $\Phi_{d^{r-1}}(j_0, j_x) = 0 \pmod{N}$. The presence of such an encoding then leaks r as a factor of the group order.

One way of minimizing the possibility of having an element with small order is to choose the prime factors of $h(\mathcal{O})$ to be as large as possible. For example, we can choose D_0 and the odd prime factors of $f_i - 1$ (where f_i is a prime factor of the conductor) to be larger than $O(\lambda^3)$.

5.4 Miscellaneous

On the decisional version of the inversion problem. Like the typical hard problems in cryptography, a search problem usually comes with a decisional variant. For the hardness of inversion, the natural way of defining the decision problem is to say that given a group element x , it is hard to decide whether a string s represents x^{-1} or a random group element. While in the ideal interface of (T)GII the decisional variant is easy, simply due to the fact that one can compose x and s , then check if the result is equal to $1_{\mathbb{G}}$ or not. In our concrete instantiation, the following variant of the decisional problem has a chance to be hard:

Definition 5.8 (Decisional inversion problem). *Given the public parameter PP and a (composable) encoding $\text{enc}(x)$, decide whether a string s represents the canonical encoding of x^{-1} (namely $j(x^{-1} * E_0)$) or a random value in $(\mathbb{Z}/N\mathbb{Z})^\times$.*

Given that the canonical encoding is not composable, there is a chance that the decisional problem is hard. However, when the degree of $\text{enc}(x)$ is a polynomial d , Problem 5.8 is easy, since we can test whether $\Phi_d(j_0, s) = 0 \pmod{N}$ or not (in contrast, we conjecture the search problem is hard even for polynomial degree encodings). If the encoding is of super-polynomial degree, then the decisional problem seems hard.

We remark that the applications of this paper do not necessarily rely on the hardness of the decisional version of the inversion problem, so the cryptanalysis effort we spend on the decisional problem is not as much as the search problem.

Rational points on $X_0(\ell)$. Rational points on $X_0(\ell)$ give solutions to $\Phi_\ell(x, y) = 0$ over $\mathbb{Z}/n\mathbb{Z}$ for any $n \in \mathbb{N}$, unless the denominators of x or y are not in $(\mathbb{Z}/n\mathbb{Z})^\times$. Recall that the genus of the modular curve $X_0(\ell)$ grows linearly with ℓ , hence for large ℓ , by Faltings' theorem, there are only finitely many rational points on $X_0(\ell)$. So for large ℓ , one cannot hope to find ℓ -isogenous neighbors over $\mathbb{Z}/N\mathbb{Z}$ by first finding points over \mathbb{Q} and then reducing them mod N . For small ℓ , on the other hand, there are rational points that seem to be easy to find.

Let $P = (x, y)$ be such a (\mathbb{Q} -rational) point on $X_0(\ell)$, where neither x nor y is equal to 0 or 1728. The presence of these points, on one hand, allows one to easily find one j -isogenous neighbor of x mod N (if the denominator of x is invertible in $\mathbb{Z}/N\mathbb{Z}$, that is y mod N).

On the other hand, for an attack towards the (ℓ, ℓ^2) -neighbor problem one would like to find two \mathbb{Q} -rational points P, Q sharing a common coordinate (i.e. either $x_P = x_Q$ or $y_P = y_Q$). We remark that having such \mathbb{Q} -rational points is highly unlikely for large ℓ . More precisely, wlog assume that $x_P = x_Q$. Then y_P and y_Q correspond to the j -invariants of ℓ^2 -isogenous curves. They, moreover, are both in \mathbb{Q} , and hence the point (y_P, y_Q) defines a \mathbb{Q} -rational point on $X_0(\ell^2)$. If $\ell > 7$, $X_0(\ell^2)$ has genus strictly great than 1 and therefore has finitely many rational points. Therefore, for $\ell > 7$ even if one can find a pair of \mathbb{Q} -rational points P and Q on $X_0(\ell)$, it is highly unlikely that they would share a common coordinate.

Although we do not see an immediate attack through rational points on $X_0(\ell)$, we can always take $\ell > 7$ for extra precaution.

5.5 Summary

We summarize the potential attacks and the countermeasures.

The adversary is given the public parameters N , j_0 , and polynomially many composable encodings, each containing several j -invariants connected by isogenies of polynomial degrees. Following Proposition 3.9 we can recover all the explicit isogenies between neighboring j -invariants. Given that we choose all the prime degrees ℓ of the isogenies to be ≥ 5 (or > 7 for extra precaution), the explicit isogenies do not trivially leak the x -coordinates of the points in the kernel of the isogeny.

We should avoid leaking the number of points on those elliptic curves over \mathbb{F}_p and \mathbb{F}_q with endomorphism ring \mathcal{O} . Otherwise the adversary can pick a random x as the x -coordinate of a random point on the curve, and run Lenstra's factoring algorithm to factorize N .

Central to the hardness of inversion is the (ℓ, ℓ^2) -isogenous neighbor problem over a composite modulus N with unknown factorization. Among the potential solutions to the (ℓ, ℓ^2) -isogenous neighbor problem, finding the one corresponding to the image of a horizontal isogeny would break our candidate group with infeasible inversion, so it is worth investigating algorithms which find isogenies with

specific directions. However, the only known such algorithm, that of [IJ13], does not seem to work over $\mathbb{Z}/N\mathbb{Z}$.

In the concrete instantiation of a group with infeasible inversion, the adversary can always try to reach new j -invariants by performing the legal GCD operations. Such attacks are captured under the model given in Definition 5.3. The parallelogram attack is a powerful attack under this model. It is effective when given the composable encodings of group elements a, b, c such that $a \circ b = c$, and the degrees of the three encodings are coprime. This attack can be prevented when 2 out of the 3 pairs of the degrees of $\text{enc}(a)$, $\text{enc}(b)$ and $\text{enc}(c)$ are not composable. In the applications of the directed transitive signature and the broadcast encryption, we are able to set the parameters to prevent the attack.

Under GRH the isogeny graph is an expander, which means given polynomially many composable encodings, it is reasonable to assume that the closure of the composition covers all the $h(D)$ j -invariants. However, finding a path of composition to reach a specific point (e.g. the inverse of some encoding) may still require solving the discrete-log problem over $\mathcal{CL}(D)$. Under the current choice of parameters, the discrete-log problem over $\mathcal{CL}(D)$ can be solved efficiently once $h(D)$ is given, and $h(D)$ can be recovered efficiently given D or any relation lattice Λ of dimension w such that $\mathbb{Z}^w/\Lambda \simeq \mathcal{CL}(D)$. So we should prevent leaking any of Λ , D , or $h(D)$.

The discriminant D cannot be polynomially large for two reasons: First, if $D \in \text{poly}(\lambda)$, then we can guess D and compute the class number in polynomial time, which enables us to efficiently solve the discrete-log problem over $\mathcal{CL}(D)$. Second, we can also compute the Hilbert class polynomial H_D in polynomial time and therefore solve the (ℓ, ℓ^2) -isogenous neighbor problem.

Even if D is super-polynomial, it should be kept hidden. Since D has to be hidden, we cannot give out a plaintext group element of $\mathcal{CL}(D)$ that allows efficient group operation, e.g. the quadratic form representation of an ideal class $[(a, b, c)]$ (cf. Section 2.1), since the discriminant D can be read off from $b^2 - 4ac$.

Finally, we also remark that if we were able to choose a super-polynomially large square-free discriminant D with a trapdoor τ_D , so that solving discrete-log over $\mathcal{CL}(D)$ is hard if one is given only D , but feasible if one is given the trapdoor τ_D , then we can construct a system with D and $h(D)$ being public, which would avoid all the complications in hiding D and $h(D)$.

6 Directed transitive signature for directed acyclic graphs

The concept of transitive signature (DTS) was introduced by Rivest and Micali [MR02]. In a transitive signature scheme the master signer is able to sign on the edges of a graph G using the master signing key. Given the signatures on a specific set S of edges, say $S = \{(u, v), (v, w)\}$, everyone can compute the signature on the edge (u, w) , and in general, any edge in the transitive closure of S , but not for any edge beyond the transitive closure of S . Transitive signatures are useful in the scenario where the graph represents some authorization relationship. The master signer has limited availability and has the intention of signing a limited number of edges ahead of time. New users can then dynamically join the graph, build edges, and obtain the composed signatures if the edges live in the transitive closure of the existing ones.

Transitive signatures for undirected graphs are constructed in [MR02, BN02] and many others. But for directed graphs, only the special case of directed trees was achieved by Yi [Yi07] from the RSA assumption. However, Neven later gave a construction of DTS for directed trees from any standard digital signature, which shows that directed trees are indeed simpler to achieve [Nev08].

We instantiate DTS from TGII for any directed acyclic graph (DAG). The basic instantiation directly follows the construction from [Hoh03, Mol03], where the signatures are the composable encodings of some group elements. Since the lengths of the composable encodings in our TGII keep growing during composition, the lengths of the composed signatures grow proportionally to the number of individual signatures. However, a DTS where the lengths of composed signatures grow proportionally to the number of individual signatures is trivially achievable – simply use any signature scheme and interpret the concatenation of a signature on the edge \vec{ij} and a signature on the edge \vec{jk} as the signature on the edge \vec{ik} – as was explicitly mentioned in [Mol03, Page 14].

To provide a DTS scheme with short composed signatures, we add a signature compressing step using the partial extraction algorithm (Algorithm 4.15) described in the general version of TGII in Section 4.3. However, the compressed signature cannot be further composed with other signatures, so the scheme has not yet met the ideal definition of DTS provided by [Hoh03, Mol03]. Still, being able to compress the final composed signature is desirable for a DTS, and it does not seem to be trivially achievable.

6.1 Definition

We adopt the definition of a directed transitive signature from [Hoh03, Mol03].

Definition 6.1. *A directed transitive signature scheme $\text{DTS} = (\text{Gen}, \text{Cert}, \text{Sign}, \text{Compose}, \text{Ver})$ consists of the following tuple of efficient algorithms:*

- **Gen:** *The key generation algorithm Gen takes as input the security parameter 1^λ , returns the master public-key secret-key pair (MPK, MSK).*
- **Cert:** *The node certification algorithm Cert takes as input the master secret key MSK and a node $i \in \mathbb{N}$, returns $(\text{PK}(i), \text{SK}(i))$, a public value and a secret value for node i .*
- **Sign:** *The edge signing algorithm Sign takes as input the master secret key MSK, the source node i and destination node k , and the associated $(\text{PK}(i), \text{SK}(i))$, $(\text{PK}(k), \text{SK}(k))$, outputs a signature $\sigma_{i,k}$ of the edge \vec{ik} .*
- **Compose:** *The composition algorithm Compose takes as input MPK, two consecutive edges \vec{ij} , \vec{jk} , and the signatures $\sigma_{i,j}$, $\sigma_{j,k}$, outputs a signature $\sigma_{i,k}$ for the edge \vec{ik} .*
- **Ver:** *The verification algorithm Ver takes as input the edge \vec{ik} and its public values $\text{PK}(i), \text{PK}(k)$, and a potential signature σ' , returns 1 iff σ' is a valid signature of the edge \vec{ik} , 0 otherwise.*

In this paper we will be considering the simplest definitions of correctness and security (we refer the readers to [Hoh03, Mol03] for the more formal definitions). For correctness we require that the verification algorithm outputs 1 on all signatures obtained from compositions in the transitive closure of edges signed by the master signing key. For security we require that it is infeasible for any p.p.t. adversary to forge a signature beyond the transitive closure of the signed edges. The adversary is allowed to dynamically add nodes and edges in the graph, and is allowed to request the master signer to sign as long as the target edge for forgery does not trivially fall in the transitive closure of the signed edges.

6.2 A directed transitive signature scheme for DAGs via TGII

Assume the DAG in the protocol has n vertices $[1, 2, \dots, n]$. All the directed edges \vec{ik} are heading from a lower index to a higher index, i.e. $i < k$ (any DAG has at least one topological ordering).

Construction 6.2 (DTS via TGII). *Given a trapdoor group with infeasible inversion TGII, a standard digital signature scheme DS, construct a directed transitive signature DTS as follows:*

- **Gen:** The key generation algorithm **Gen** takes as input the security parameter 1^λ and runs the TGII parameter generation algorithm to produce the public parameters **TGII.PP** and the trapdoor τ of a group $\mathbb{G} = (\circ, 1_{\mathbb{G}})$. It also generates the signing and verification keys for the regular signature scheme $\text{DS.Gen}(1^\lambda) \rightarrow \text{DS.SK}, \text{DS.VK}$. It returns the master public-key $\text{DTS.MPK} = (\text{TGII.PP}, \text{DS.VK})$ and the master secret-key $\text{DTS.MSK} = (\tau, \text{DS.SK})$.
- **Cert:** The node certification algorithm **Cert** takes as input the master secret key $\text{DTS.MSK} = (\tau, \text{DS.SK})$ and a node $i \in \mathbb{N}$, samples a random element $x_i \in \mathbb{G}$ and a composable encoding $\text{enc}(x_i)$, and sets the encoding to be the public information for i :

$$\text{PK}(i) := \text{enc}(x_i) = \text{TGII.TrapSam}(\text{TGII.PP}, \tau, x_i).$$

The secret information on node i can be set as $\text{SK}(i) = x_i^{-1}$, or simply be left as \perp since the master trapdoor holder can invert $\text{PK}(i) = \text{enc}(x_i)$ to get x_i^{-1} . **Cert** then produces the signature $\Sigma(i) = \text{DS.Sign}(\text{DS.SK}, i || \text{PK}(i))$ and takes $\Sigma(i)$ as the certificate of node i .

- **Sign:** The edge signing algorithm **Sign** takes as input the master secret key $\text{DTS.MSK} = (\tau, \text{DS.SK})$, a source node i , a destination node k , and the associated $(\text{PK}(i), \text{SK}(i)), (\text{PK}(k), \text{SK}(k))$. It first verifies the node certificates, then recovers x_i^{-1} from $\text{SK}(i)$ and x_k from $\text{PK}(k)$. It then outputs

$$\sigma_{i,k} := \text{enc}(x_i^{-1} \circ x_k) = \text{TGII.TrapSam}(\text{TGII.PP}, \tau, x_i^{-1} \circ x_k).$$

- **Compose:** The composition algorithm **Compose** takes as input **MPK**, two consecutive edges \vec{ij} , \vec{jk} , and the signatures $\sigma_{i,j}$, $\sigma_{j,k}$, outputs $\sigma_{i,j} \circ \sigma_{j,k}$ as the composed signature for the edge \vec{ik} .
- **Ver:** The verification algorithm **Ver** takes as input the edge \vec{ik} and its public values $\text{PK}(i), \text{PK}(k)$, and a potential signature σ' . It parses the public information as $(i, \text{enc}(i), \Sigma(i)), (k, \text{enc}(k), \Sigma(k))$. If any of the certificates on the nodes is invalid, it returns 0. Otherwise, it checks whether

$$\text{Ext}(\text{TGII.PP}, \text{enc}(i) \circ \sigma') = \text{Ext}(\text{TGII.PP}, \text{enc}(k)).$$

If so it returns 1, otherwise it returns 0.

Choosing the generation sets S . Now we provide a detailed instantiation of the DTS using our basic version of the TGII (cf. § 4.2). The only parameters left to be specified are the primes $\{p_i\}$ of the generation set $S = \{C_i = [(p_i, b_i, \cdot)]\}_{i \in [w]} \subset \mathbb{G} = \mathcal{CL}(\mathcal{O})_{\text{odd}}$ in the first step of the (stateful) encoding sampling algorithm (cf. Algorithm 4.4). To do so we need to clarify which encodings are composable in the system and which are not (for functionality); whether there are pairwise composable encodings of a, b, c that satisfy $a \circ b = c$ (to prevent the parallelogram attack detailed in § 5.3.2); and whether we are able to hide the bases of the relation lattices (to prevent the attack detailed in § 5.3.3).

One way of choosing the generation sets is as follows: Let the following be sets of \mathcal{O} -ideals of distinct prime norms, each set is of size $O(\log(\lambda))$: $S_{i,src}$, $S_{i,dst}$, for all $i \in [n]$, and S_{common} .

- Let the generation set for $\text{PK}(i) = \text{enc}(x_i)$ be $S_{\text{common}} \cup S_{i,\text{dst}}$, plus another \mathcal{O} -ideal of prime norm ℓ_i that has never been used.
- Let the generation set for $\sigma_{i,k} = \text{enc}(x_i^{-1} \circ x_k)$ be $S_{i,\text{src}} \cup S_{k,\text{dst}}$, plus another \mathcal{O} -ideal of prime norm $\ell_{i,k}$ that has never been used.

Let us first check correctness of our construction: For each index $i \in [n]$, the public key on node i , $\text{PK}(i) = \text{enc}(x_i)$, should be composable with all the edges \vec{jk} with $j \geq i$ (for correctness it is not important if $\text{PK}(i)$ is composable with any other edges or vertices). A signature $\sigma_{i,k} = \text{enc}(x_i^{-1} \circ x_k)$ should be composable with any signature $\sigma_{i',k'}$ such that $i' \geq k$, or $k' \leq i$, and with any public value $\text{PK}(h)$ such that $h \leq i$, but not with any other vertices or edges. So our assignment of the prime ideals supports the necessary composition functionalities.

We now verify that the parallelogram attack from § 5.3.2 does not apply to our construction: In the DTS scheme there are two ways of obtaining composable encodings of a, b, c such that $a \circ b = c$.

1. Through the compositions of the signatures on the edges $\vec{ij}, \vec{jk}, \vec{ik}$, which gives

$$\sigma_{i,j} \circ \sigma_{j,k} = \sigma_{i,k} \Rightarrow \text{Ext}(\text{TGII.PP}, \text{enc}(x_i^{-1} \cdot x_j) \circ \text{enc}(x_j^{-1} \cdot x_k)) = \text{Ext}(\text{TGII.PP}, \text{enc}(x_i^{-1} \cdot x_k)). \quad (7)$$

Our choices of the generation sets make sure that $\sigma_{j,k}$ and $\sigma_{i,k}$ are not composable due to the common $S_{k,\text{dst}}$; $\sigma_{i,j}$ and $\sigma_{i,k}$ are not composable due to the common $S_{i,\text{src}}$.

2. Through the verification algorithm on the public values $\text{PK}(i)$, $\text{PK}(k)$, and the signature on \vec{ik} , which gives

$$\text{PK}(i) \circ \sigma_{i,k} = \text{PK}(k) \Rightarrow \text{Ext}(\text{TGII.PP}, \text{enc}(x_i) \circ \text{enc}(x_i^{-1} \cdot x_k)) = \text{Ext}(\text{TGII.PP}, \text{enc}(x_k)). \quad (8)$$

Our choices of the generation sets make sure that $\text{PK}(i)$ and $\text{PK}(k)$ are not composable due to the common S_{common} ; $\sigma_{i,k}$ and $\text{PK}(k)$ are not composable due to the common $S_{k,\text{dst}}$.

Therefore, in both cases, two of the three pairs of the degrees share prime factors. Therefore the parallelogram attack does not apply.

Finally, we verify that it is unlikely to leak a full-rank basis of any relation lattice of $\mathcal{CL}(\mathcal{O})$. Note that the only two ways of obtaining non-trivial encodings of the identity are mentioned in Eqn. (7) and Eqn. (8). Due to the fresh prime ideal inserted in the generation set of each encoding, there are always more dimensions than the number of linearly independent relations in a potential relation lattice, which means it is unlikely to obtain any full-rank basis of such a lattice.

Remark 6.3 (Why not supporting waiting signatures). *The definition of DTS from [Hoh03] additionally allows the signatures to be signed, composed, or verified over non-consecutive edges. Such signatures are called waiting signatures. The DTS construction from ideal TGII in [Hoh03, Mol03] naturally supports waiting signatures. To do so, the verification algorithm Ver takes as input two multisets of sources and destinations $P_{\text{src}} = \{a_1, \dots, a_z\}$, $P_{\text{dst}} = \{b_1, \dots, b_z\}$, and a potential signature σ' . It checks whether*

$$\text{Ext}(\text{TGII.PP}, \sigma' \circ \text{enc}(a_1) \circ \dots \circ \text{enc}(a_z)) = \text{Ext}(\text{TGII.PP}, \text{enc}(b_1) \circ \dots \circ \text{enc}(b_z)).$$

If so returns 1, otherwise returns 0.

Our TGII construction is able to support waiting signatures in terms of functionality, but not security. The reason is: to verify waiting signatures, we would have to make sure that all the public values of the vertices are composable. But then the parallelogram attack would apply through Eqn. (8). The attacker would then be able to recover the encoding of x_i^{-1} , which is the secret-key on node i .

Compressing the composed signature. The definition of [Hoh03, Mol03] additionally requires that for any consecutive edge in the graph, the composed signature should be indistinguishable from a signature produced by the master signer. We remark that the DTS from our basic version of the TGII (cf. §4.2) does not achieve this property, since the signatures grow as they are composed⁶.

Instead, we will provide a signature compression technique using the partial extraction algorithm from the general version of TGII in §4.3. The compression can only be applied to a composed signature $\sigma_{i,k}$ which can be verified (given $\text{PK}(i)$ and $\text{PK}(k)$) but can no longer be composed with other signatures. Our construction, therefore, does not fully achieve the indistinguishability definition of [Hoh03, Mol03], in the sense that a compressed signature cannot be composed with others later.

To obtain the compressed signature, we apply $\text{Partial.Ext}(\text{TGII.PP}, \sigma_{i,k}, \text{PK}(i))$ to obtain the j -invariant $j_{\vec{ik}} = \text{Ext}(\text{TGII.PP}, \sigma_{i,k})$, the isogeny $\phi_i : j_{\vec{ik}} \rightarrow j_{x_k}$ that represents $\text{PK}(i)$, and a SNARG π for the instance $(\text{PK}(i), \phi_i)$ and statement “there exists a string str such that ϕ_i is computed from running a circuit that instantiates Algorithm 4.15 on inputs str, ϕ_i ”. Let $(j_{\vec{ik}}, \phi_i, \pi)$ be the compressed signature. Note that its size depends only on the security parameter.

We change the verification algorithm as follows: Ver takes as input a source node a , a destination node b , and a potential compressed signature (j, ϕ, π) for an edge from a to b . It parses the public information as $(a, \text{enc}(a), \Sigma(a))$, $(b, \text{enc}(b), \Sigma(b))$ and if any of the certificates on the nodes is invalid it returns 0. Otherwise, it checks:

- Whether $\phi(j) = \text{Ext}(\text{TGII.PP}, \text{enc}(b))$,
- Whether π is a valid proof for the following statement: there is a string str such that ϕ is obtained by applying Algorithm 4.15 over the input $\text{str}, \text{enc}(a)$.

If both of the checks pass, returns 1, otherwise returns 0.

The correctness of the verification algorithm follows from the correctness of the encoding algorithm, the algorithm Partial.Ext , and the completeness of the SNARG. The soundness of SNARG guarantees that the isogeny ϕ is obtained honestly from applying Partial.Ext on some string str and $\text{enc}(a)$, which means ϕ correctly represents $\text{PK}(a)$ and therefore j must be $\text{Ext}(\text{TGII.PP}, \text{enc}(a^{-1} \circ b))$.

Furthermore, we can use a zero-knowledge SNARG (from e.g. [Gro16]), so that the compressed signature is indistinguishable from a compressed signature produced by the master signer.

7 Broadcast encryption

A broadcast encryption scheme allows the encrypter to generate ciphertexts that are decryptable by a designated subset of the receivers. A trivial solution for broadcast encryption is to simply encrypt the message many times for each user’s decryption key. Although this does provide a solution, it clearly is not effective as the number of receivers grow. A meaningful broadcast encryption scheme needs to be more efficient on at least one of: the cost of encryption, size of the public parameters, or the sizes of users’ decryption keys (cf. [FN93]).

In this section we give a concrete instantiation of the broadcast encryption scheme of Irer et al. [ILOP04], which was designed under the ideal interface of GII. Our instantiation supports private-key encryption and allows any numbers of users to collude. The encryption overhead and the users’

⁶The same happens in the DTS from the TGII implied by the self-bilinear maps with auxiliary inputs [YYHK14], where the auxiliary input in the composed encoding keeps growing.

secret keys are independent of the total number of the users n , however the public parameter size is linear in n . Let us remark that our aim here is to give a concrete application of our TGII construction and hence we did not try to optimize the public parameter overhead. Let us also mention that the asymptotic efficiency of our scheme has been achieved by the construction in [BGW05] using bilinear maps. There are other schemes in literature that achieve $\log(n)$ public parameter overhead under the assumption of the security of multilinear maps [BWZ14] or iO [Zha14]. We leave the challenge of achieving smaller public parameter blowup from GII or TGII to the interested reader.

7.1 Definition

A private-key broadcast encryption scheme consists of a tuple of efficient algorithms:

- **Setup**(1^λ): The setup algorithm takes as input the security parameter 1^λ , then generates the public parameters PP and the master secret key MSK .
- **Gen**(PP, MSK, u): The user key generation algorithm takes as input the user id, u , in the list of users \mathcal{U} . It generates a secret key SK_u and (possibly) a public key PK_u for user u . PK_u is included in the public parameters.
- **Enc**($\text{PP}, \text{MSK}, \Gamma, m$): The encryption algorithm takes as input a polynomial size set $\Gamma \subseteq \mathcal{U}$ of recipients. It applies a deterministic algorithm over $\text{PP}, \text{MSK}, \Gamma$ to derive a key K of a symmetric-key encryption scheme. Denoting the encryption of the message m under K by $\text{CT}_{K,m}$, it outputs $\text{CT} = (\Gamma, \text{CT}_{K,m})$.
- **Dec**($\text{PP}, u, \text{SK}_u, \text{CT}$): The decryption algorithm first parses CT as $(\Gamma, \text{CT}_{K,m})$ and derives the symmetric-key K from PP, SK_u , and Γ . It then uses K to decrypt $\text{CT}_{K,m}$.

The scheme is said to be correct if for all subsets of the users $\Gamma \subseteq \mathcal{U}$ and for all messages m , any user u from the set Γ decrypts the correct message, i.e. $\text{Dec}(\text{PP}, u, \text{SK}_u, \Gamma, \text{Enc}(\text{PP}, \text{MSK}, \Gamma, m)) = m$.

For security we consider the simplest form of the key-recovery attack, defined by the following “key-recovery” game between an adversary and a challenger.

1. The challenger runs the setup algorithm to generate a master secret key MSK and the public parameters. The challenger then picks a set of users \mathcal{U} and generates the public keys and the secret keys of the users. The adversary is given \mathcal{U} and all the public parameters.
2. The adversary picks a subset $\Gamma \subseteq \mathcal{U}$ of users where it wants to attack. The challenger gives the adversary all the secret keys for the users not in Γ , i.e. SK_u for $u \notin \Gamma$.
3. The adversary can make encryption queries on any message m and any subset $\Gamma' \subseteq \mathcal{U}$ (including $\Gamma' = \Gamma$). The challenger runs the Encrypt algorithm to obtain $\text{CT} = \text{Enc}(\text{PP}, \text{MSK}, \Gamma', m)$ and sends to the adversary.
4. The adversary outputs a symmetric key K^* and wins the game if K^* is equal to the symmetric key K derived from PP, MSK and Γ , loses otherwise.

A broadcast encryption scheme is said to be secure if for any polynomial time adversary the odds of winning the key-recover game is negligible.

Construction	param	user key	CT	Based on	PK	CR
[FN93]	$O(t^2 n \log n)$	$O(t \log^2 t \log n)$	$O(t^2 \log^2 t \log n)$	RSA assumption	No	$\leq t$ users
[ILOP04]	$O(n)$	$O(1)$	$O(1)$	Ideal GII	No	Arbitrary
[BGW05] I	$O(n)$	$O(1)$	$O(1)$	Bilinear maps	Yes	Arbitrary
[BGW05] II	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(1)$	Bilinear maps	Yes	Arbitrary
[BWZ14]	$O(\log n)$	$O(\log n)$	$O(\log n)$	Mmaps	Yes	Arbitrary
[Zha14]	$O(\log n)$	$O(\log n)$	$O(\log n)$	iO	Yes	Arbitrary
This work	$O(n)$	$O(1)$	$O(1)$	A concrete TGII	No	Arbitrary

Figure 8: A brief summary of the existing collusion resistant broadcast encryption schemes. n represents the number of users; “PK” stands for supporting public key encryption; “CR” stands for being collusion resistant. All the parameters ignore the (possibly multiplicative) $\text{poly}(\lambda)$ factors.

7.2 A private-key broadcast encryption scheme from TGII

We first present the construction from an abstract TGII, then specify the parameters in detail.

Construction 7.1 (Broadcast encryption under an abstract TGII). *Given a trapdoor group with infeasible inversion TGII and a symmetric-key encryption scheme Sym, construct a broadcast encryption scheme BE as follows:*

- **BE.Setup(1^λ)**: *The setup algorithm takes as input the security parameter 1^λ , runs the TGII parameter generation algorithm to produce the public parameters TGII.PP and the trapdoor τ of a group $\mathbb{G} = (\circ, 1_{\mathbb{G}})$. Then sample a random element $s \in \mathbb{G}$. The public parameter BE.PP is set to be TGII.PP. The master secret key BE.MSK includes τ and s .*
- **BE.Gen(BE.PP, BE.MSK, u)**: *The user secret key generation algorithm parses τ, s from BE.MSK. It samples a random element $x \in \mathbb{G}$, computes $\text{enc}(x) \leftarrow \text{TGII.TrapSam}(\text{TGII.PP}, \tau, x)$ as the user’s public key PK_u ; computes $\text{enc}(s \circ x) \leftarrow \text{TGII.TrapSam}(\text{TGII.PP}, \tau, s \circ x)$ and treats it as the user’s secret key SK_u .*
- **BE.Enc(BE.PP, BE.MSK, Γ, m)**: *The encryption algorithm takes as input a polynomial size set $\Gamma \subseteq \mathcal{U}$ of recipients and a message m . It first computes the message encryption key $K = (\prod_{i \in \Gamma} x_i) \circ s$, then computes $\text{Sym.CT}_{K,m} := \text{Sym.Enc}(K, m)$ and outputs $\text{BE.CT} = (\Gamma, \text{Sym.CT}_{K,m})$.*
- **BE.Dec(BE.PP, $u, \text{SK}_u, \text{CT}$)**: *The decryption algorithm extracts the set Γ from BE.CT, takes the public keys PK_i for users $i \in \Gamma \setminus u$, the secret key SK_u for user u , and computes $K' = \text{Ext}(\text{TGII.PP}, \prod_{i \in \Gamma \setminus u} (\text{PK}_i) \circ \text{SK}_u)$. It then decrypts $\text{Sym.CT}_{K,m}$ using K' .*

To be more cautious, we can also apply a randomness extractor on K to derive a key that is statistically close to the uniform in distribution.

Choosing the generation sets. We now provide the detailed instantiation of the broadcast encryption scheme using our basic version of the TGII (cf. Section 4.2). Similar to the situation of the DTS, the only parameters left to be specified are the primes $\{p_i\}$ in the generation set $S = \{C_i = [(p_i, b_i, \cdot)]\}_{i \in [w]} \subset \mathbb{G} = \mathcal{CL}(\mathcal{O})_{\text{odd}}$ in the first step of the (stateful) encoding sampling algorithm (cf. Algorithm 4.4).

One way of choosing the primes for the generation sets is as follows. Choose sets of \mathcal{O} -ideals S_i for $i \in \mathcal{U}$ and S_{msk} such that they consist of ideals of distinct prime norms and each one of them is of size $O(\log(\lambda))$.

- Let the generation set of $\text{PK}_i = \text{enc}(x_i)$ be S_i , plus another \mathcal{O} -ideal of prime norm ℓ_i that has never been used.
- Let the generation set of $\text{SK}_i = \text{enc}(x_i \circ s)$ be $S_i \cup S_{msk}$, plus another \mathcal{O} -ideal of prime norm ℓ'_i that has never been used.

Let us first check the correctness. The secret key SK_i does not have to be composable with any other secret keys, it has to be composable with all the public keys except PK_i . The PK_i has to be composable with everything other than SK_i . So our assignment of the prime ideals supports the necessary composition functionalities.

We will now consider the parallelogram attack from Section 5.3.2. Since the only parameters in the encryption scheme are SK_u and PK_v for $u, v \in \mathcal{U}$, the only way we can obtain an identity of the type $a \circ b = c$ is through the relations

$$\text{PK}_i \circ \text{SK}_k = \text{PK}_k \circ \text{SK}_i, \quad (9)$$

where the public-key secret-key pairs are $\text{PK}_i = \text{enc}(x_i)$, $\text{SK}_i = \text{enc}(x_i \circ s)$, $\text{PK}_k = \text{enc}(x_k)$, $\text{SK}_k = \text{enc}(x_k \circ s)$. The attack scenario in Eqn. (9) may happen when Users i and j collude to find the secret keys of other users.

Our choices of the generation sets make sure that SK_i and SK_k are not composable for any i, k due to the common S_{msk} . Similarly, SK_i and PK_i are not composable for any i due to the common S_i . So there are two out of three pairs of the degrees that share prime factors. Therefore, our instantiation is secure to the parallelogram attack.

Finally we check whether there is a chance of leaking a full-rank basis of any relation lattice of $\mathcal{CL}(\mathcal{O})$. Note that the only two ways of obtaining non-trivial encodings of the identity are mentioned in Eqn. (9). Due to the new prime ideals inserted in the generation set of each encoding, there are always more dimensions than the number of linearly independent relations in a potential relation lattice, which means it is unlikely to obtain any full-rank basis of such a lattice.

Remark 7.2. *The scheme we presented above makes a slight change over the original scheme of Irer et al. [ILOP04]. In [ILOP04] the user's secret key is x and the user's public key is $s \circ x$. The encryption key with respect to the set Γ is $K = (\prod_{i \in \Gamma} x_i) \circ s^{|\Gamma|-1}$. Whereas in our scheme we flip the public key and secret key, and change the encryption key accordingly. The purpose of the change is to provide a candidate instantiation where only the short-basis of the relation lattice for S_{msk} has to be computed.*

Note that this change does not affect the security analysis from [ILOP04], which shows that a key recovery attack to the broadcast encryption scheme implies the ability of computing inverses.

Remark 7.3. *The CPA-style security definition in [BGW05] requires that it is hard to distinguish a correct decryption key from a random key. Our scheme is also a candidate that satisfies the CPA definition when the decisional inversion problem is hard, which is plausible under the current setting of parameters.*

8 Future directions

We conclude the paper with several future directions.

Further investigation of the (ℓ, ℓ^2) -isogenous neighbors problem. The hardness of the (ℓ, ℓ^2) -isogenous neighbor problem over $\mathbb{Z}/N\mathbb{Z}$ is necessary for the security of our candidate trapdoor group with infeasible inversion. Let us remark that once the adversary is given two j -invariants that are ℓ -isogenous, she can recover the rational polynomial of the isogeny, which is not available in the ℓ -isogenous neighbor problem where the adversary is given a single j -invariant as input. Although it is not clear how to use the explicit rational polynomial of the isogeny to mount an attack, it should serve as a warning sign that the (ℓ, ℓ^2) -isogenous neighbors problem might be easier than the ℓ -isogenous neighbors problem.

Proving security in the GCD evaluation model. The GCD evaluation model (cf. Definition 5.3) provides a simplified interface for the attacker. It will be interesting to prove (or disprove) that the TGII is secure if the adversary is restricted to follow the GCD evaluation model.

Looking for alternative constructions of GII or TGII. Given the complications and the limitations of our construction of TGII, one might want to look for a simpler or a different construction of GII or TGII. Some concrete directions to study further are:

1. A construction, where the encoding is stateless.
2. A construction, where the size of the encoding does not grow with composition.
3. A (T)GII candidate for a non-commutative group.

Acknowledgments

We would like to thank Andrew V. Sutherland, and the anonymous reviewers for their helpful comments.

References

- [Bab86] László Babai. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.
- [BGI⁺01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *LNCS*, pages 1–18. Springer, 2001.
- [BGK⁺18] Dan Boneh, Darren Glass, Daniel Krashen, Kristin E. Lauter, Shahed Sharif, Alice Silverberg, Mehdi Tibouchi, and Mark Zhandry. Multiparty non-interactive key exchange and more from isogenies on elliptic curves. *IACR Cryptology ePrint Archive*, 2018:665, 2018.
- [BGW05] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 258–275. Springer, 2005.

- [Bis11] Gaetan Bisson. Computing endomorphism rings of elliptic curves under the grh. *Journal of Mathematical Cryptology*, 5(2):101–113, 2011.
- [BLP⁺13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 575–584. ACM, 2013.
- [BLS12] Reinier Bröker, Kristin Lauter, and Andrew V. Sutherland. Modular polynomials via isogeny volcanoes. *Mathematics of Computation*, 81(278):1201–1231, 2012.
- [BN02] Mihir Bellare and Gregory Neven. Transitive signatures based on factoring and rsa. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 397–414. Springer, 2002.
- [BS11] Gaetan Bisson and Andrew V. Sutherland. Computing the endomorphism ring of an ordinary elliptic curve over a finite field. *Journal of Number Theory*, 131(5):815–831, 2011.
- [BV98] Dan Boneh and Ramarathnam Venkatesan. Breaking RSA may not be equivalent to factoring. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 59–71. Springer, 1998.
- [BW88] Johannes A. Buchmann and Hugh C. Williams. A key-exchange system based on imaginary quadratic fields. *J. Cryptology*, 1(2):107–118, 1988.
- [BWZ14] Dan Boneh, Brent Waters, and Mark Zhandry. Low overhead broadcast encryption from multilinear maps. In *CRYPTO (1)*, volume 8616 of *Lecture Notes in Computer Science*, pages 206–223. Springer, 2014.
- [CHL⁺15] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. In *EUROCRYPT (1)*, volume 9056 of *Lecture Notes in Computer Science*, pages 3–12. Springer, 2015.
- [CL84] Henri Cohen and Hendrik Willem Lenstra. Heuristics on class groups of number fields. 1984.
- [CL05] Denis Charles and Kristin Lauter. Computing modular polynomials. *LMS Journal of Computation and Mathematics*, 8:195–204, 2005.
- [CLG09] Denis Xavier Charles, Kristin E. Lauter, and Eyal Z. Goren. Cryptographic hash functions from expander graphs. *J. Cryptology*, 22(1):93–113, 2009.
- [CM94] Jean-Marc Couveignes and François Morain. Schoof’s algorithm and isogeny cycles. In *International Algorithmic Number Theory Symposium*, pages 43–58. Springer, 1994.
- [Coh95] Henri Cohen. A course in computational algebraic number theory. 1995.
- [Cop97] Don Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *J. Cryptology*, 10(4):233–260, 1997.
- [Cou06] Jean-Marc Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291, 2006.

- [Cox11] David A Cox. *Primes of the form $x^2 + ny^2$: Fermat, class field theory, and complex multiplication*, volume 34. John Wiley & Sons, 2011.
- [Dam88] Ivan Damgård. On the randomness of legendre and jacobi sequences. In *CRYPTO*, volume 403 of *Lecture Notes in Computer Science*, pages 163–172. Springer, 1988.
- [Dem93] N. Demytko. A new elliptic curve based analogue of RSA. In *EUROCRYPT*, volume 765 of *Lecture Notes in Computer Science*, pages 40–49. Springer, 1993.
- [DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, 22(6):644–654, 1976.
- [E⁺98] Noam D Elkies et al. Elliptic and modular curves over finite fields and related computational issues. *AMS IP STUDIES IN ADVANCED MATHEMATICS*, 7:21–76, 1998.
- [ES10] Andreas Enge and Andrew V. Sutherland. Class invariants by the crt method. In *International Algorithmic Number Theory Symposium*, pages 142–156. Springer, 2010.
- [Feo17] Luca De Feo. Mathematics of isogeny based cryptography. *arXiv preprint arXiv:1711.04062*, 2017.
- [FM02] Mireille Fouquet and François Morain. Isogeny volcanoes and the SEA algorithm. In *International Algorithmic Number Theory Symposium*, pages 276–291. Springer, 2002.
- [FN93] Amos Fiat and Moni Naor. Broadcast encryption. In *CRYPTO*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491. Springer, 1993.
- [Gal99] Steven D Galbraith. Constructing isogenies between elliptic curves over finite fields. *LMS Journal of Computation and Mathematics*, 2:118–138, 1999.
- [GGH⁺13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, pages 40–49. IEEE Computer Society, 2013.
- [GH93] Dorian Goldfeld and Jeffrey Hoffstein. On the number of Fourier coefficients that determine a modular form. In Marvin Knopp and Mark Sheingorn, editors, *A Tribute to Emil Grosswald: Number Theory and Related Analysis*, volume 143 of *Contemporary Mathematics*, pages 385–393, Providence, Rhode Island, 1993. AMS.
- [GHS02] Steven D. Galbraith, Florian Hess, and Nigel P. Smart. Extending the GHS weil descent attack. In *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 29–44. Springer, 2002.
- [GM05] Steven D Galbraith and James F McKee. Pairings on elliptic curves over finite commutative rings. In *IMA International Conference on Cryptography and Coding*, pages 392–409. Springer, 2005.
- [GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.
- [Gro16] Jens Groth. On the size of pairing-based non-interactive arguments. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 305–326. Springer, 2016.

- [HM00] Safuat Hamdy and Bodo Möller. Security of cryptosystems based on class groups of imaginary quadratic orders. In *ASIACRYPT*, volume 1976 of *Lecture Notes in Computer Science*, pages 234–247. Springer, 2000.
- [Hof14] Jeffrey Hoffstein. A History of the Development of NTRU. Eurocrypt talk, 2014.
- [Hoh03] Susan Rae Hohenberger. The cryptographic impact of groups with infeasible inversion. Master’s thesis, Massachusetts Institute of Technology, 2003.
- [IJ13] Sorina Ionica and Antoine Joux. Pairing the volcano. *Math. Comput.*, 82(281):581–603, 2013.
- [ILOP04] Jim Irrer, Satyanarayana Lokam, Lukasz Opyrchal, and Atul Prakash. Infeasible group inversion and broadcast encryption. *University of Michigan Electrical Engineering and Computer Science Tech Note CSE-TR-485-04*, 2004.
- [JF11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *PQCrypto*, volume 7071 of *Lecture Notes in Computer Science*, pages 19–34. Springer, 2011.
- [JMV05] David Jao, Stephen D. Miller, and Ramarathnam Venkatesan. Do all elliptic curves of the same order have the same difficulty of discrete log? In *ASIACRYPT*, volume 3788 of *Lecture Notes in Computer Science*, pages 21–40. Springer, 2005.
- [KK98] Noboru Kunihiro and Kenji Koyama. Equivalence of counting the number of points on elliptic curve over the ring \mathbb{Z}_n and factoring n . In *EUROCRYPT*, volume 1403 of *Lecture Notes in Computer Science*, pages 47–58. Springer, 1998.
- [Kob87] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of computation*, 48(177):203–209, 1987.
- [Koh96] David Russell Kohel. *Endomorphism rings of elliptic curves over finite fields*. PhD thesis, University of California, Berkeley, 1996.
- [Kub76] Daniel Sion Kubert. Universal bounds on the torsion of elliptic curves. *Proceedings of the London Mathematical Society*, 3(2):193–237, 1976.
- [Len87] Hendrik Willem Lenstra. Factoring integers with elliptic curves. *Annals of mathematics*, pages 649–673, 1987.
- [LLL82] Arjen Klaas Lenstra, Hendrik Willem Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.
- [LZ94] Georg-Johann Lay and Horst G Zimmer. Constructing elliptic curves with given group order over large finite fields. In *International Algorithmic Number Theory Symposium*, pages 250–263. Springer, 1994.
- [McC88] Kevin S McCurley. *Cryptographic key distribution and computation in class groups*. IBM Thomas J. Watson Research Division, 1988.
- [Mic00] Silvio Micali. Computationally sound proofs. *SIAM J. Comput.*, 30(4):1253–1298, 2000.
- [Mil85] Victor S Miller. Use of elliptic curves in cryptography. In *Conference on the theory and application of cryptographic techniques*, pages 417–426. Springer, 1985.

- [Mol03] David Molnar. Homomorphic signature schemes. B.s. thesis, Harvard College, 2003.
- [MR02] Silvio Micali and Ronald L. Rivest. Transitive signature schemes. In *CT-RSA*, volume 2271 of *Lecture Notes in Computer Science*, pages 236–243. Springer, 2002.
- [MR07] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measure. *SIAM Journal on Computing*, 37(1):267–302, 2007.
- [Nev08] Gregory Neven. A simple transitive signature scheme for directed trees. *Theoretical Computer Science*, 396(1-3):277–282, 2008.
- [NR06] Phong Q. Nguyen and Oded Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. In *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 271–288. Springer, 2006.
- [PH78] Stephen C. Pohlig and Martin E. Hellman. An improved algorithm for computing logarithms over $\text{gf}(p)$ and its cryptographic significance (corresp.). *IEEE Transactions on Information Theory*, 24(1):106–110, 1978.
- [Rab79] Michael O Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE LAB FOR COMPUTER SCIENCE, 1979.
- [RS06] Alexander Rostovtsev and Anton Stolbunov. Public-key cryptosystem based on isogenies. Cryptology ePrint Archive, Report 2006/145, 2006.
- [Sch85] René Schoof. Elliptic curves over finite fields and the computation of square roots mod p . *Mathematics of computation*, 44(170):483–494, 1985.
- [Sch95] René Schoof. Counting points on elliptic curves over finite fields. *J. Théor. Nombres Bordeaux*, 7(1):219–254, 1995.
- [Sil09] Joseph H Silverman. *The arithmetic of elliptic curves*, volume 106. Springer Science & Business Media, 2009.
- [Sil13] Joseph H Silverman. *Advanced topics in the arithmetic of elliptic curves*, volume 151. Springer Science & Business Media, 2013.
- [Sut] Andrew V. Sutherland. Isogeny kernels and division polynomials. https://ocw.mit.edu/courses/mathematics/18-783-elliptic-curves-spring-2017/lecture-notes/MIT18_783S17_lec6.pdf. Accessed: 2018-09-03.
- [Sut11a] Andrew V. Sutherland. Computing Hilbert class polynomials with the Chinese remainder theorem. *Mathematics of Computation*, 80(273):501–538, 2011.
- [Sut11b] Andrew V. Sutherland. Structure computation and discrete logarithms in finite abelian p -groups. *Mathematics of Computation*, 80(273):477–500, 2011.
- [Sut12a] Andrew Sutherland. Constructing elliptic curves over finite fields with prescribed torsion. *Mathematics of Computation*, 81(278):1131–1147, 2012.
- [Sut12b] Andrew V. Sutherland. Accelerating the CM method. *LMS Journal of Computation and Mathematics*, 15:172–204, 2012.

- [Sut13a] Andrew V. Sutherland. Isogeny volcanoes. *The Open Book Series*, 1(1):507–530, 2013.
- [Sut13b] Andrew V. Sutherland. On the evaluation of modular polynomials. *The Open Book Series*, 1(1):531–555, 2013.
- [Tat66] John Tate. Endomorphisms of abelian varieties over finite fields. *Inventiones mathematicae*, 2(2):134–144, 1966.
- [Tes99] Edlyn Teske. The Pohlig–Hellman method generalized for group structure computation. *Journal of Symbolic Computation*, 27(6):521–534, 1999.
- [Vél71] Jacques Vélú. Isogénies entre courbes elliptiques. *Comptes Rendus de l’Académie des Sciences de Paris*, 273:238–241, 1971.
- [Yi07] Xun Yi. Directed transitive signature scheme. In *CT-RSA*, volume 4377 of *Lecture Notes in Computer Science*, pages 129–144. Springer, 2007.
- [YYHK14] Takashi Yamakawa, Shota Yamada, Goichiro Hanaoka, and Noboru Kunihiro. Self-bilinear map on unknown order groups from indistinguishability obfuscation and its applications. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II*, pages 90–107, 2014.
- [YYHK18] Takashi Yamakawa, Shota Yamada, Goichiro Hanaoka, and Noboru Kunihiro. Generic hardness of inversion on ring and its relation to self-bilinear map. *IACR Cryptology ePrint Archive*, 2018:463, 2018.
- [Zha14] Mark Zhandry. Adaptively secure broadcast encryption with small system parameters. *IACR Cryptology ePrint Archive*, 2014:757, 2014.