Designs, Codes and Cryptography
https://doi.org/10.1007/s10623-024-01441-7

CSI-Otter: isogeny-based (partially) blind signatures from the class group action with a twist

Shuichi Katsumata[1,2] · **Yi-Fu Lai**[3,4] · **Jason T. LeGrow**[5] · **Ling Qin**[3]

Received: 16 August 2023 / Revised: 17 May 2024 / Accepted: 3 June 2024
© The Author(s) 2024

Abstract
In this paper, we construct the first provably-secure isogeny-based (partially) blind signature scheme. While at a high level the scheme resembles the Schnorr blind signature, our work does not directly follow from that construction, since isogenies do not offer as rich an algebraic structure. Specifically, our protocol does not fit into the *linear identification protocol* abstraction introduced by Hauck, Kiltz, and Loss (EUROCYRPT' 19), which was used to generically construct Schnorr-like blind signatures based on modules such as classical groups and lattices. Consequently, our scheme is provably secure in the random oracle model (ROM) against poly-logarithmically-many concurrent sessions assuming the subexponential hardness of the group action inverse problem. In more detail, our blind signature exploits the *quadratic twist* of an elliptic curve in an essential way to endow isogenies with a strictly richer structure than abstract group actions (but still more restrictive than modules). The basic scheme has public key size 128 B and signature size 8 KB under the CSIDH-512 parameter sets—these are the smallest among all provably secure post-quantum secure blind signatures. Relying on a new *ring* variant of the group action inverse problem (rGAIP), we can halve the signature size to 4 KB while increasing the public key size to 512 B. We provide preliminary cryptanalysis of rGAIP and show that for certain parameter settings, it is essentially as secure as the standard GAIP. Finally, we show a novel way to turn our blind signature into a partially blind signature, where we deviate from prior methods since they require hashing into the set of public keys while hiding the corresponding secret key—constructing such a hash function in the isogeny setting remains an open problem.

Keywords Isogeny-based cryptography · Blind signatures · Group action-based cryptography · Post-quantum cryptography

Mathematics Subject Classification 11T71 · 13P25 · 14K02 · 94A60 · 94A62

Communicated by A. Enge.

Author list in alphabetical order.

Extended author information available on the last page of the article

1 About

An extended abstract of this work was published in CRYPTO 2023 [56]. This is a full version of that paper. In particular, in this work we present additional explanation of the framework of Kastner et al. [55] in the context of our work; provide complete security proofs for the blind signature scheme of Sect. 5; present proofs of correctness and blindness for the partially blind signature scheme of Sect. 6; provide proofs of correctness and the honest verifier zero-knowledge property of the sigma protocol of Sect. 7, and; provide a proof of blindness of the blind and partially blind signatures of Sect. 7 with special attention to the setting of adversarially-generated public keys, which may be malformed. In Sect. 5.1 we also provide additional discussion of the sigma protocol which underlies the blind and partially blind signatures of Sects. 5 and 6, while in Sects. 8.2 and 8.3 we provide additional analysis the problem of sampling appropriate primitive roots of unity in $\mathbb{Z}_N^\times$, which is required for our optimizations in Sect. 7.

2 Introduction

Blind signatures, introduced by Chaum [27], allow a user to obtain a signature on a message from a signer, while the signer is *blind* to the message it signed. One can think of the physical analogy where a user puts a letter—acting as the message—to be signed into a special carbon paper envelope. The signer can sign the envelope without opening it; his signature is transferred to the letter by the carbon paper, and the letter is never visible to the signer. In practice, it is sometimes necessary to consider the extension of *partially* blind signatures, introduced by Abe and Fujisaki [3], that further allow embedding a message agreed on by both the signer and the user into the signature. The messages can now be divided into public and private parts, where the public part can include, for instance, the expiration date of the signature. While (partially) blind signatures[1] were originally used to construct e-cash [27, 30, 66], anonymous credentials [20, 22], and e-voting [28, 44], the notion has recently seen renewed interest due to applications in blockchains [21, 85] and privacy-preserving authentication tokens [51, 84].

Currently, the most promising class of efficient blind signatures known to withstand quantum attacks is those based on lattices. We have recently encountered significant progress in lattice-based blind signatures, such as [5, 37, 50, 64], where the signature size currently sits around 50 KB to 10 MB. However, this is still an order of magnitude larger than their classical counterparts, with a signature size ranging from a few hundred bytes to 1 KB. As we see a continuous surge of interest in post-quantum security and better user privacy, we aim to investigate a post-quantum blind signature with a smaller signature size.

One potentially promising path to a post-quantum blind signature with a short signature is to rely on *isogeny*-based constructions. This is because while their signing and verification times are less efficient, standard isogeny-based signature schemes [12, 35, 36] are known to produce comparable or even smaller signatures compared to lattices. In fact, for a more advanced form of signature schemes such as ring signatures and group signatures, isogenies can produce much shorter signatures compared to their lattice counterparts [13, 14].

Unfortunately, at first glance this path seems difficult to follow. Very roughly, there are two approaches to constructing a blind signature. The first approach is based on the Schnorr

[1] For readability, we focus on blind signatures below when the distinction between the partial and non-partial is insignificant.

blind signature [29]. This approach builds on a sigma (or an identification) protocol with a "nice" algebraic property and boosts it into a blind signature by appropriately randomizing the interaction. This nice algebraic property has recently been stated informally to be *modules* [49, 50], where isogenies are not known to be endowed with: isogenies are only *group actions* that are strictly less structured than modules (see Sect. 2.2 for more details). The second approach is based on the generic construction proposed by Fischlin [41] that requires proving, at the minimum, possession of a valid signature of a standard signature scheme using a non-interactive zero-knowledge proof (NIZK). While del Pino and Katsumata [37] and Agrawal et al. [5] recently used this approach to construct more efficient lattice-based blind signatures than were previously known, this seems impractical to translate to the isogeny setting due to the lack of efficient NIZKs for such complex languages.

In summary, while isogenies have the potential to produce the shortest post-quantum blind signatures, it is unclear how we can leverage known approaches to build them. This brings us to the main question of this work:

Can we construct an efficient post-quantum (partially) blind signature scheme from isogenies?

2.1 Our contribution

In this work, we answer the above question in the affirmative through four contributions. Our first contribution is to construct the first post-quantum blind signature based on isogenies (or CSIDH group actions to be more specific) called CSI-Otter, short for CSI-fish with Or-proof Twisted ThreE-Round protocol. The construction is akin to the Schnorr blind signature [29] but follows a slightly different approach. Unlike previous constructions that required the underlying mathematical tool to be a module [49, 50], we bypass this requirement. The crux of our construction is to effectively use the *quadratic twist* of an elliptic curve, or in layman's terms, we use the fact that isogenies are *slightly* more expressive than a group action. We build a basic blind signature with public key size 128 B and signature size 8 KB based on the standard group action inverse problem (GAIP) over the CSIDH-512 parameter sets. We formally prove that our basic blind signature is secure in the (classical) random oracle model with poly-logarithmically many concurrent signing sessions following the recent work by Kastner et al. [55], assuming the subexponential hardness of the group action inverse problem (or a constant number of concurrent sessions assuming only polynomial hardness). That is, the security proof permits a poly-logarithmic number of signatures to be issued per public key in a concurrent manner. We note that extension to polynomially-many concurrent sessions is impossible, as demonstrated in work of Katasumata et al. [57].

Our second contribution is to provide an optimization of our basic blind signature using a new hardness assumption called the ζ_d-*ring* group action inverse problem (ζ_d-rGAIP), where ζ_d denotes a *d-th primitive root of unity* over $\mathbb{Z}_N$. Informally, ζ_d-rGAIP asserts that given $([\mathfrak{g}^{s\cdot\zeta_d^j}] * E_0)_{j\in[d]}$ for a random exponent $s \xleftarrow{\$} \mathbb{Z}_N$ and base elliptic curve $E_0\colon y^2 = x^3 + x$, it is difficult to solve for s. Note that when $d = 2$, we have $\zeta_2 = -1$ and we recover the standard GAIP, where $[\mathfrak{g}^{-s}] * E_0$ is the (efficiently computable) quadratic twist of $[\mathfrak{g}^s] * E_0$. At a high level, ζ_d-rGAIP allows us to use a larger challenge space for the underlying sigma protocol by increasing the public key. This in turn implies that the number of parallel repetitions can be lowered compared to our basic blind signature, and effectively, we obtain a public key size of $(128 \cdot d)$ B and signature size of roughly $(8/\log_2 d)$ KB based on ζ_d-rGAIP. Our construction is generic and works for any group actions for which the ζ_d-rGAIP is hard, however, we must show that such group actions exist for it to be useful.

Our third contribution complements our second contribution: we provide a preliminary cryptanalysis on the hardness of ζ_d-rGAIP for the CSIDH-512 parameter sets. We first show that the set of values $\{\mathsf{gcd}(\zeta_d^i - 1, N)\}_{i\in[d]}$ relates to the hardness of ζ_d-rGAIP. Informally, we create new GAIP instances over a series of subgroups of the class group, where the size of these subgroups relate to each $\mathsf{gcd}(\zeta_d^i - 1, N)$. Using known attacks against GAIP in a Pohlig-Hellman manner, we can break this newly generated GAIP instances that has a smaller order compared to the GAIP with CSIDH-512. For instance with CSIDH-512, when $d = 7$ or 8, this attack shows that ζ_d-rGAIP only has half the security of GAIP over CSIDH-512. On the other hand, for other values of d such as $d = 2, 3, 4, 5, 9, \dots$, this attack is no more effective than trying to break GAIP over CSIDH-512. In fact, when $\mathsf{gcd}(\zeta_d^i - 1, N) = N/\mathsf{poly}(n)$ for n the security parameter, we show a reduction from the ζ_d-rGAIP to GAIP, thus establishing the optimality of our attack for certain parameters such as $d = 3, 5, 9, \dots$. In the end, due to other correctness constraints, we are only able to instantiate the above optimized blind signature with $d = 4$, which leads to a public key of size 512 B and signature size of 4 KB. While our preliminary cryptanalysis shows that ζ_4-rGAIP is presumably as hard as GAIP over CSIDH-512, we leave further cryptanalysis for future work as it is not covered by our reduction to GAIP.

Our final contribution is extending our basic blind signature into a *partially* blind signature. While it is straightforward to construct a partially blind signature from a Schnorr-style blind signature in the classical group or the lattice settings, this approach fails in the isogeny setting.[2] For example, Abe and Okamoto [4] constructed the first partially blind signature, where the main idea was to hash the public message (also known as a *tag*) info to a group element $h_{\mathsf{info}} \in \mathbb{G}$ and let the signer prove that it knows either the exponent of its public key $h = g^a$ or the hashed tag h_{info}. In particular, the underlying sigma protocol proves a 1-out-of-2 (or an OR) relation. In the security proof, the reduction samples $a_{\mathsf{info}} \overset{\$}{\leftarrow} \mathbb{Z}_p$, programs the random oracle so that $h_{\mathsf{info}} = g^{a_{\mathsf{info}}}$, and uses a_{info} to simulate the signing algorithm. Unfortunately, this approach is inapplicable in the isogeny setting since we do not know how to map into the set of elliptic curves while simultaneously hiding the exponent. Note that if the exponent is known, any real-world adversary can use the reduction algorithm to forge a signature, thus rendering the scheme insecure.

To this end, we provide a new general approach to constructing partially blind signatures that may be of an independent interest. At the core of our approach is devising a sigma protocol for a *2-out-of-3* relation and embedding the tag info into the signature differently. Since the sigma protocol must also be compatible with the blind signature, we are not able to rely on any 2-out-of-3 sigma protocols for threshold relations such as Cramer-Damgård-Schnoemakers' sigma protocol [33] using Shamir's secret-sharing scheme [81]. One downside of our partially blind signature is that compared to our blind signature, it requires a signature size roughly three times as large. However, we note that even then, we still achieve a smaller signature size than the lattice-based counterparts.

[2] We note that *proving* the security of a partially blind signature is more subtle and difficult. Indeed, it was only recently that Kastner et al. [55] provided a corrected proof of the Abe-Okamoto (partially) blind signature [4].

2.2 Technical overview

We now explain our contributions in detail. We first review the Schnorr blind signature and see where it fails when translating the construction to the isogeny setting. We then explain our basic blind signature $\mathsf{CSI\text{-}Otter}$ that uses the quadratic twist and further show how to extend it to the partially blind setting. Finally, we explain the optimization using the newly introduced rGAIP assumption.

2.2.1 Reviewing the Schnorr blind signature

We first recall the Schnorr sigma/identification protocol between a prover with $(\mathsf{pk}, \mathsf{sk}) = (h = g^a, a) \in \mathbb{G} \times \mathbb{Z}_p$ and a verifier with pk. The prover samples $y \xleftarrow{\$} \mathbb{Z}_p$ and sends $Y = g^y$ to the verifier. The verifier sends a random challenge $c \xleftarrow{\$} \mathbb{Z}_p$ to the prover, where the prover replies with $r = y - a \cdot c$. The verifier is convinced that it was communicating with a prover in possession of $\mathsf{sk} = a$ if $g^r \cdot h^c = Y$. Here, if the verifier sets the challenge as $c = \mathsf{H}(Y \| \mathsf{M})$ for a message M and a hash function H modeled as a random oracle, then $\sigma = (c, r)$ serves as a signature based on the Fiat–Shamir transform [40], where the prover is the *signer* and the verifier is the *user with* M.

Clearly, this interactive signing protocol does not satisfy *blindness*, which roughly stipulates that a signature cannot be traced back to a specific signing session. In particular, when the user outputs the pair (M, σ), the signer will know in which session it signed σ—or equivalently, the signature σ can be traced back to the user—by simply checking when the hash value c included in σ was used.

The main idea of the Schnorr blind signature [29] is to let the user randomize the interaction so the session transcript becomes independent of the final signature. More explicitly, the user randomizes the interaction so that the final signature becomes $\sigma' = (c+d, r+z)$, where (d, z) is uniform over $\mathbb{Z}_p^2$ from the view of the signer. The Schnorr blind signature accomplishes this as follows: When the user receives Y as the first-sender message, it samples $(d, z) \xleftarrow{\$} \mathbb{Z}_p^2$ and sets $Y' := g^z \cdot Y \cdot h^d$. It then computes $c' = \mathsf{H}(Y' \| \mathsf{M})$ and sends $c := c' - d$ to the signer, where the signer replies with $r = y - a \cdot c$ as before. Since we have $g^r \cdot h^c = Y$, the user can multiply g^z and h^d on each side to obtain $g^{r+z} \cdot h^{c+d} = Y'$. Thus, $\sigma' = (c', r') := (c + d, r + z)$ is a valid signature for the message M. Moreover, it can be checked that this satisfies (perfect) blindness since any signature $\sigma' = (c', r')$ has an equal chance of being generated from a transcript (Y, c, r), where the probability is taken over the randomness sampled by the user.

2.2.2 Difficulty with group actions

In the above, the user is implicitly using a specific structure of the underlying Schnorr sigma protocol to randomize the interaction. Specifically, it is using the fact that $\mathbb{G}$ is a $\mathbb{Z}_p$*-module*. This allows the user to randomize the first-signer message $Y \in \mathbb{G}$ by multiplying it with the generator $g \in \mathbb{G}$ raised to the power of $z \in \mathbb{Z}_p$ and the public key $h = g^a \in \mathbb{G}$ lifted to the power of $d \in \mathbb{Z}_p$. This property has been more formally abstracted as a *linear identification protocol* [49, 50], which covers schemes based on classical groups and lattices.

Unfortunately, this does not extend to the isogeny setting since isogenies are only a *group action*. Concretely, the CSIDH group action is defined as $* : G \times \mathcal{E}\ell\ell \to \mathcal{E}\ell\ell$, where G is an ideal class group and $\mathcal{E}\ell\ell$ is a set of elliptic curves, and we further assume the structure of G is known and can be expressed as $G = \langle [\mathfrak{g}] \rangle \cong \mathbb{Z}_N$ for some $N \in \mathbb{N}$, where $\mathfrak{g}$ is the generator

[12]. Let us make an attempt to construct an isogeny-based Schnorr-style blind signature where the public key is $\mathsf{pk} = A = [\mathfrak{g}^a] * E_0 \in \mathcal{E\ell\ell}$ for a random $a \xleftarrow{\$} \mathbb{Z}_N$ and a fixed curve E_0. While the analogy of setting the first-signer message as $Y = [\mathfrak{g}^y] * E_0$ for $y \xleftarrow{\$} \mathbb{Z}_N$ works, it seems this is as far as we can get. Unlike the Schnorr blind signature, the user can only randomize Y *once from the left side*. That is, while computing $[\mathfrak{g}^z] * Y$ for a random $z \in G$ is possible, combining Y with $[\mathfrak{g}^d] * A$ is not possible since they are both set elements. We note that in the Schnorr blind signature setting, the former and latter correspond to $g^z \cdot Y$ and $Y \cdot h^d$, respectively. Since the blindness of the Schnorr blind signature hinged on the fact that the first-sender message Y can be randomized *twice*; one randomness d to hide the challenge c and another randomness z to hide the second-signer message r, it is unclear how to use isogenies to construct a blind signature while having only one way to randomize Y.

2.2.3 Using the quadratic twist

Our main observation to overcome this problem is to rely on the property that isogenies are slightly more expressive than a group action due to the *quadratic twist*. Given any $A = [\mathfrak{g}^a] * E_0$ for an unknown $a \in \mathbb{Z}_N$, we can efficiently compute its quadratic twist $[\mathfrak{g}^{-a}] * E_0$, which we denote[3] by A^{-1}.

We first explain the underlying isogeny-based sigma protocol, where we assume for now that the challenge space is $\mathcal{C} = \{-1, 1\}$. As above, the prover sends $Y = [\mathfrak{g}^y] * E_0$ for $y \xleftarrow{\$} \mathbb{Z}_N$. The verifier then sends a random challenge $c \xleftarrow{\$} \{-1, 1\}$, and the prover replies with $r = y - a \cdot c$. The verifier then verifies the "signature" $\sigma = (c, r)$ by checking whether $[\mathfrak{g}^r] * A^c = Y$, where note that A^c is well-defined for $c \in \{-1, 1\}$ even though A comes from the set of elliptic curves. For an honest execution of the protocol, we have $[\mathfrak{g}^r] * A^c = [\mathfrak{g}^r] * ([\mathfrak{g}^{a \cdot c}] * E_0) = [\mathfrak{g}^{r + a \cdot c}] * E_0 = Y$ as desired.[4]

Our idea is to randomize this sigma protocol so that the signature $\sigma = (c, r)$ becomes $\sigma' = (c \cdot d, r \cdot d + z)$, where (d, z) is uniform over $\{-1, 1\} \times \mathbb{Z}_N$ from the view of the signer. Concretely, given the first-sender message Y, the user randomizes Y by sampling random $(d, z) \xleftarrow{\$} \{-1, 1\} \times \mathbb{Z}_N$ and sets $Y' := [\mathfrak{g}^z] * Y^d$. It then computes $c' = \mathsf{H}(Y' \| \mathsf{M})$ and sends $c := c' \cdot d$. The signer replies with $r = y - a \cdot c$ as before. Since we have $[\mathfrak{g}^r] * A^c = Y$, the user can first compute $[\mathfrak{g}^{r \cdot d}] * A^{c \cdot d} = Y^d$. Namely, it performs nothing if $d = 1$, and computes the quadratic twist of both sides if $d = -1$. It then acts by $[\mathfrak{g}^z]$ to obtain $[\mathfrak{g}^{r \cdot d + z}] * A^{c \cdot d} = [\mathfrak{g}^z] * Y^d$. Since the right-hand side is Y', $\sigma' = (c', r') := (c \cdot d, r \cdot d + z)$ is a valid signature for the message M as desired. Moreover, it can be checked that we have perfect blindness since c and r are both randomized; the (multiplicative) randomness $d \in \{-1, 1\}$ hides the challenge c and the (additive) randomness $z \in \mathbb{Z}_N$ hides the response r. Put differently, any signature $\sigma' = (c', r')$ has an equal chance of being generated from a transcript (Y, c, r), where the probability is taken over the randomness sampled by the user.

Finally, to turn this basic idea into a secure blind signature, we enlarge the challenge space to be exponentially large, i.e., $\mathcal{C} = \{-1, 1\}^n$ where n is the security parameter. All the above

[3] The notation for the quadratic twist is not totally uniform in the literature. When $E/k\colon y^2 = x^3 + Ax^2 + x$ and $c \in k^\times \backslash k^{\times 2}$ one sometimes denotes $E^c/k\colon cy^2 = x^3 + Ax^2 + x$. In this work we will always have $-1 \in k^\times \backslash k^{\times 2}$ (since $k = \mathbb{F}_p$ and $p \equiv 3 \pmod 4$), and we will have $E^{-1} \cong E'\colon y^2 = x^3 - Ax^2 + x$ by the change of variables $(x, y) \mapsto (-x, y)$. So this notation—while not usually used in the CSIDH literature—is reasonable, and will be convenient for our protocol description.

[4] Note that this is a standard (optimized variant of an) isogeny-based sigma protocol where 0 is removed from the challenge space (see for instance [12]).

arguments naturally extend to this enlarged challenge space by running the protocol n times in parallel.

2.2.4 Formal security proof

A knowledgeable reader may recall that the Schnorr blind signature is not known to be secure in the random oracle model [11]. This is also the case for our described isogeny-based blind signature. The Schnorr blind signature has been generalized by Pointcheval and Stern [71, 72] and Abe and Okamoto [4] in similar but different ways to have a security proof in the random oracle model. The latter Abe-Okamoto blind signature is compatible with our isogeny-based construction, where the public key is modified to a tuple $\mathsf{pk} = (A_0, A_1) = ([\mathfrak{g}_0^a] * E_0, [\mathfrak{g}_1^a] * E_0) \in \mathcal{E}\ell\ell^2$ for a random $(a_0, a_1) \overset{\$}{\leftarrow} \mathbb{Z}_N^2$, and the secret key to $\mathsf{sk} = (\delta, a_\delta)$ for a random $\delta \overset{\$}{\leftarrow} \{0, 1\}$. The construction uses the OR composition of the underlying sigma protocol and works well with our idea using the quadratic twist. While the original proof of Abe and Okamoto [4] contained a subtle but non-trivially fixable bug, Kastner et al. [55] recently provided a somewhat generic proof for Abe-Okamoto style blind signatures. The security proof of our blind signature is established by adapting their result to our setting.

2.2.5 Turning it partially blind

As explained in Sect. 2.1, there is no analog of the Abe-Okamoto *partially* blind signature in the isogeny setting. The only reason why we could replicate the Abe-Okamoto (non-partial) blind signature in the isogeny setting was that both (A_0, A_1) in pk were set up in a way that the user did not know the secret exponents. Generating $A_1 \in \mathcal{E}\ell\ell$ as a hash of the tag info, i.e., $A_1 = \mathsf{H}(\mathsf{info})$, would have failed in the isogeny setting since we cannot do so without letting the computation of $\mathsf{H}(\cdot)$ reveal the secret exponent a_1. If a_1 is public, then the scheme becomes trivially forgeable.

Our main approach in constructing a partially blind signature is to keep the same public key $\mathsf{pk} = (A_0, A_1)$ as before but to generate another curve $A_2 = \mathsf{H}(\mathsf{info})$ *with the secret exponent* a_2. We then modify the signer to prove that it knows at least *two of the three* exponents of (A_0, A_1, A_2). The reduction will be able to extract either a secret key pair (a_0, a_2), (a_1, a_2), or (a_0, a_1) from the forgery: we can rely on the proof for the standard blind signature that the first two pairs occur with an almost equal probability independent of the secret key used by the reduction, and the third case always allows the reduction to win.

The question is then how to construct a base sigma protocol for this 2-out-of-3 relation that is compatible with the above randomization technique using the quadratic twist. For instance, we cannot use the well-known Cramer-Damgård-Schnoemakers' sigma protocol [33] using Shamir's secret-sharing scheme [81] since the challenge space $\mathcal{C} = \{-1, 1\}$ is used as a multiplicative group in our construction, rather than a field as required by Shamir's secret-sharing scheme.[5] To this end, we use a 2-out-of-3 *multiplicative* secret-sharing scheme as follows: Given a secret $c \in \{-1, 1\}$, sample $(c_0, c_1, c_2) \in \{-1, 1\}^3$ uniformly random conditioned on $c_0 \cdot c_1 \cdot c_2 = c$. We then view (c_0, c_1), (c_1, c_2), and (c_2, c_0) as the three shares. One can check that any two of the three shares allow reconstructing c, while c is information-theoretically hidden when only one share is known.

We now construct a sigma protocol for a 2-out-of-3 relation using this secret-sharing scheme as follows: the high-level idea is to assign the secret shares (c_0, c_1), (c_1, c_2), and

[5] Since parallel repetition is not required to show blindness, we only focus on the small challenge space for simplicity.

(c_2, c_0) to the exponents a_0, a_1, and a_2, respectively. In more detail, assume the prover knows the exponents a_0 and a_2. It first samples two shares $(c_1, c_2) \xleftarrow{\$} \{-1, 1\}^2$ and runs the honest-verifier zero-knowledge simulator to simulate the knowledge of the unknown exponent a_1. Specifically, it samples $(r_{1,0}, r_{1,1}) \xleftarrow{\$} \mathbb{Z}_N^2$ and sets $(Y_{1,0}, Y_{1,1}) = ([\mathfrak{g}^{r_{1,0}}] * A_1^{c_1}, [\mathfrak{g}^{r_{1,1}}] * A_1^{c_2})$. It then sets $(Y_{b,0}, Y_{b,1}) = ([\mathfrak{g}^{y_{b,0}}] * A_b, [\mathfrak{g}^{y_{b,1}}] * A_b)$ for $b \in \{0, 2\}$ by sampling the y's as before. Upon receiving $(Y_{b,0}, Y_{b,1})_{b\in\{0,1,2\}}$, the verifier returns a random $c \in \{-1, 1\}$. The prover sets the final share $c_0 = c \cdot c_1 \cdot c_2$ and computes $(r_{0,0}, r_{0,1}) = (y_{0,0} - a_0 \cdot c_0, y_{0,1} - a_0 \cdot c_1)$ and $(r_{2,0}, r_{2,1}) = (y_{2,0} - a_2 \cdot c_2, y_{2,1} - a_2 \cdot c_0)$, where recall a_2 is the publicly known exponent associated with the tag info. Finally, the prover replies with $(r_{b,0}, r_{b,1})_{b\in\{0,1,2\}}$. The verifier can check the validity of the proof by a similar check as before and will be convinced that the prover knows at least two secret exponents of $\mathsf{pk} = (A_0, A_1, A_2)$.

Building on a similar argument using the quadratic twist, we turn this 2-out-of-3 sigma protocol into a partially blind signature by allowing the user to appropriately randomize the first-signer message Y's. The user samples three randomness from $\{-1, 1\}$ to randomize the challenge (c_0, c_1, c_2) and six randomness from $\mathbb{Z}_N$ to randomize the second-signer message $(r_{b,0}, r_{b,1})_{b\in\{0,1,2\}}$. We show that the proof of Kastner et al. [55] can be slightly modified to work for this partially blind signature.

2.2.6 Optimization using higher degree roots of unity

Finally, we show how to optimize our blind signature. One of the implicit reasons why the randomization of the sigma protocol worked was because the challenge space $\mathcal{C} = \{-1, 1\}$ was a multiplicative subgroup of the ring $\mathbb{Z}_N$. We generalize this observation and consider a larger challenge space $\mathcal{C}_d = \{\zeta_d^j\}_{j\in[d]}$, where ζ_d is the d-th primitive root of unity over $\mathbb{Z}_N$,[6] i.e., $\zeta_d^d = 1$ and $\zeta_d^j \neq 1$ for any $j \in [d-1]$. $\mathcal{C}_d$ is indeed a larger multiplicative subgroup of the ring $\mathbb{Z}_N$, where setting $d = 2$ recovers the challenge space $\mathcal{C}_2 = \mathcal{C}$. The goal of the optimized scheme remains the same: we want to randomize the signature $\sigma = (c, r) \in \mathcal{C}_d \times \mathbb{Z}_N$ by $\sigma' = (c \cdot d, r \cdot d + z)$ for a random $(d, z) \xleftarrow{\$} \mathcal{C}_d \times \mathbb{Z}_N$. However, unfortunately, when we use a larger challenge space $\mathcal{C}_d$ for $d > 2$, the underlying sigma protocol no longer satisfies correctness. Recall in the most simple sigma protocol, the verifier receives $Y = [\mathfrak{g}^y] * E_0$, outputs a challenge $c \in \{-1, 1\}$, receives $r = y - a \cdot c$ and checks if $[\mathfrak{g}^r] * A^c = Y$. The final check by the verifier was computable since computing the quadratic twist (i.e., A^{-1}) was efficient. This is no longer the case for a more general $c \in \mathcal{C}_d$ since we do not know how to compute $A^j := [\mathfrak{g}^{a \cdot \zeta_d^j}] * E_0$ given only the curve $A = [\mathfrak{g}^a] * E_0 \in \mathcal{E\ell\ell}$, $j \in [d-1]$, and ζ_d with $d \geq 3$. To this end, we extend the public key to $\mathsf{pk} = (A^j)_{j\in[d]}$ to aid the verifier's computation and modify the sigma protocol to address this extension. This is where we rely on the new ζ_d-*ring* group action inverse problem (ζ_d-rGAIP) which states that given pk, it is difficult to recover the exponent $a \in \mathbb{Z}_N$. Before getting into the hardness of ζ_d-rGAIP, we finish the overview of our optimized blind signature below.

Although we are now able to construct a sigma protocol with a larger challenge space, it does not yet naturally extend to blind signatures due to the extra structure. In particular, the main issue is that when the signer sends $Y = [\mathfrak{g}^y] * E_0$ as the first message, our idea was to let the user randomize this by $[\mathfrak{g}^z] * Y^w$, where $Y^w := [\mathfrak{g}^{y \cdot \zeta_d^w}] * E_0$ for $(z, w) \xleftarrow{\$} \mathbb{Z}_N \times \mathcal{C}_d$. However, due to the same reason as above, this cannot be efficiently computed from only Y. To this end, we further extend the sigma protocol so that the prover includes all $(Y^j)_{j\in[d]}$ in the first message. While this structure cannot be efficiently checked by the verifier/user,

[6] For the overview, we will ignore when such ζ_d exists and how to find them (see Sect. 8.1 for more details).

we modify the sigma protocol so that it performs some consistency checks on these Y^j's. We show that this check is sufficient to argue blindness of the resulting blind signature even when the malicious signer is using a malformed public key, i.e., $(A^j)_{j\in[d]}$ does not have the correct ring structure.

2.2.7 Cryptanalysis of ζ_d-rGAIP

We have explained how to construct an optimized blind signature assuming the hardness of ζ_d-rGAIP. We complement our result by providing a preliminary cryptanalysis of ζ_d-rGAIP for the CSIDH-512 parameter set. We provide an attack that exploits the additional structure of ζ_d-rGAIP for specific choices of d. The insight is the difference of each curves in the public key always has a factor of $(\zeta_d^i - \zeta_d^j)$ for distinct $i, j \in [d]$ which constitutes a non-injective endomorphism over the secret key space $\mathbb{Z}_N$. By investigating these differences, we can reduce an ζ_d-rGAIP instance to a GAIP instance with a possibly smaller group than $\mathbb{Z}_N$ and recover partial information. Then, we can integrate these partial information in a Pohlig-Hellman sense. As a consequence, we can evaluate the upper bound security strength of ζ_d-rGAIP using known attacks against GAIP. For some choices of ζ_d, ζ_d-rGAIP only has half the security of GAIP for the CSIDH-512 parameters. On the other hand, for some instances of ζ_d, we show that ζ_d-rGAIP is as hard as GAIP, which demonstrates that the upper bounds obtained via our cryptanalysis are also the lower bounds. There are some instances of ζ_d-rGAIP for which our attack does not apply while also having no reduction to GAIP. We leave analysis of such instances of ζ_d-rGAIP for the CSIDH-512 parameter set as an interesting future work.

2.2.8 Isogeny-based cryptography

The roots of isogeny-based cryptography can be traced back to a 1997 talk of Couveignes, later published online in 2006 [32] and independently rediscovered by Rostovstev and Stolbunov [76]. These works propose a post-quantum key establishment protocol—the CRS protocol—whose security is based on the difficulty of the "parallelization" problem for the class group action on the set of *ordinary* elliptic curves; that is, finding $[\mathfrak{a}][\mathfrak{b}]*E$ given E, $[\mathfrak{a}]*E$, $[\mathfrak{b}]*E$, where E is an ordinary elliptic curve with endomorphism ring $\mathcal{O}$ and $[\mathfrak{a}], [\mathfrak{b}] \in \mathcal{C}\ell(\mathcal{O})$. This paralellization problem is the "Diffie-Hellman analogue" of the perhaps more natural "group action inversion" problem: given two ordinary curves E and $E' = [\mathfrak{a}] * E$, find $[\mathfrak{a}]$. The CRS scheme suffered primarily from two flaws: first, it was impractically slow—requiring approximately 458 s to establish a key at the 128-bit security level [82]—and second, Childs, Jao, and Soukharev [31] demonstrated that the CRS protocol is vulnerable to a subexponential-time attack using Kuperberg's algorithm [58], with later works [16, 19, 53] improving the attack to require only polynomial quantum space due to Regev's improved version of Kuperberg's algorithm [74].

These problems with ordinary isogeny-based protocols led researchers to instead consider protocols based on *supersingular* elliptic curves. The first such protocol was the hash function due to Charles et al. [26]. Later, De Feo, Jao, and Plût introduced the Supersingular Isogeny Diffie–Hellman (SIDH) key establishment protocol, which was later used to construct Supersingular Isogeny Key Establishment (SIKE) [52], which was a fourth round candidate in the NIST Post-Quantum Cryptography competition. Despite passive attacks on "unbalanced" variants [69, 73] and active attacks on static/ephemeral implementations [38, 45, 48], SIDH resisted cryptanalysis until 2022, when a series of papers [23, 65, 75] established that SIDH and SIKE could be broken in polynomial time. While there are proposals for countermeasures to these devastating attacks [42], the efficacy of these countermeasures has not yet been thoroughly studied.

Commutative Supersingular Isogeny Diffie-Hellman (CSIDH) was introduced in 2017 by Castryck et al. [24] as an alternative to SIDH. Unlike SIDH—which bears very little resemblance to CRS—CSIDH is very much a supersingular analogue of CRS. In CSIDH, the supersingularity of the curves involved is exploited to ensure that a torsion subgroup of very large smooth order is defined over $\mathbb{F}_{p^2}$, which allows approximately uniform random sampling and evaluation of complex multiplication to be performed very efficiently, making CSIDH orders of magnitude faster than CRS. As well, CSIDH is not known to be susceptible to any kind of adaptive attack, making it usable in the static/ephemeral setting.

The inability to uniformly sample elements of the ideal class group whose action can be computed efficiently (without knowing the relation lattice of the class group) makes it difficult to create CSIDH-based signatures. De Feo and Galbraith were the first to solve this problem in their protocol SeaSign [35], using rejection sampling to ensure that signing key information is never leaked. Later, Beullens, Kleinjung, and Vercauteren were able to compute the relation lattice of the class group used in the CSIDH-512 parameter set, and hence construct CSI-FiSh [12]: a CSIDH-based signature scheme without rejection sampling. Unfortunately, the best known classical algorithms to compute the relation lattice scale subexponentially in the CSIDH security parameter, and so it is not currently possible to extend CSI-FiSh to larger parameter sets. However, there are efficient quantum algorithms to compute these relation lattices, making CSI-FiSh a candidate for *post-post-quantum* cryptography [34]: cryptographic protocols which require a quantum computer to establish global parameters, but which are otherwise classical. A very recent work [39] shows a feasible manner to obtain the group structure using the oriented supersingular curves and imaginary quadratic orders with a large prime conductor. Though the isogeny evaluation has subexponential complexity in theory, they show a feasible result in practice by carefully choosing the parameters.

When the relation lattice of the class group is known, complex multiplication is an instance of what Couveignes [32] called a *hard homogeneous space*, and what is now often called a *cryptographic group action* [6]. While many CSIDH/CSI-FiSh-based protocols have been constructed using the group action abstractly, the CSIDH group action actually has slightly more structure than an abstract cryptographic group action. In particular, if $E/\mathbb{F}_p : y^2 = x^3 + Ax^2 + x$ has endomorphism ring $\mathcal{O}$ and $[\mathfrak{b}] \in \mathcal{C}\ell(\mathcal{O})$

$$([\mathfrak{b}] * E)^{-1} = [\mathfrak{b}]^{-1} * E^{-1}$$

where E^{-1} has Montgomery form $E^{-1}: y^2 = x^3 - Ax^2 + x$. In particular, if we take $E = E_0 : y^2 = x^3 + x$ we have $([\mathfrak{b}] * E_0)^{-1} = [\mathfrak{b}]^{-1} * E_0$, and so given $[\mathfrak{b}] * E_0$, we have an efficient way of constructing $[\mathfrak{b}]^{-1} * E_0$. This additional structure turns out to be a powerful tool, which has led to the construction of a UC-secure isogeny-based oblivious transfer [60], provably-secure isogeny-based password authenticated key establishment [1] (which had been elusive for years [10, 83]) and new techniques for fault attack-resistance of static/ephemeral CSIDH [63]. It is also a useful tool used in [12, 61] to compress the signature or the proof size.

2.2.9 Post-quantum blind signatures

The most active area of post-quantum blind signatures is those based on lattices. The first lattice-based blind signature was proposed by Rükert [77], who followed a design paradigm similar to the classical Schnorr or Okamoto–Schnorr blind signatures [72, 79]. This approach has been optimized in subsequent works [7–9, 62, 67], where BLAZE+ by Alkadri et al. [8] currently stands as the most efficient proposal. However, recently, Hauck et al. [50] showed that all constructions following the blueprint of Rükert's blind signature contain the same bug in their security proof, and provided the first *provably secure* lattice-based blind signature following a similar template with a signature size of roughly 7.9 MB.

Recently, Lyubashevsky et al. [64] constructed a novel blind signature based on a new approach using one-time signatures and OR proofs. While they can support only a bounded polynomially many signatures per public key, the signature size is small as 150 KB. In a concurrent and independent work, del Pino and Katsumata [37] and Agrawal et al. [5] showed two different methods loosely following the generic blind signature construction by Fischlin [41]. The former has a signature size of roughly 100 KB under the SIS assumption and is the first scheme to have provable security in the *quantum* random oracle model. The latter has a signature size of roughly 50 KB under a newly introduced *one-more* SIS assumption. In an independent and concurrent work to ours, Beullens et al. [15] recently took this approach one step further and constructed a lattice-based blind signature with signature size of 22 KB. The construction relies on an NIZK for proving relations of concrete hash functions.

Finally, there are a few blind signatures based on other post-quantum assumptions. Blazy et al. [17] constructs a code-based blind signature following the generic blind signature construction by Fischlin. The other is by Petzoldt et al. [70] that constructs a multivariate-based blind signature under a non-standard unforgeability notion.

3 Background

3.1 Notation

We denote the set of natural numbers and integers by $\mathbb{N}$ and $\mathbb{Z}$, respectively. We define the ring of integers modulo N, i.e., $\mathbb{Z}_N$, with representatives in $[-N/2, N/2) \cap \mathbb{Z}$. For a positive integer k, we let $[k]$ denote the set $\{1, 2, \ldots, k\}$. For a vector $\overrightarrow{h}$, h_i denotes its i-th entry and $\overrightarrow{h}_{[i]}$ denotes the vector of its first i-entries. For a distribution D, we write $x \overset{\$}{\leftarrow} D$ to denote x is sampled according to D. For a finite set S, we denote $x \overset{\$}{\leftarrow} S$ to sample x uniformly at random over S. We use $\odot$ to denote the component-wise multiplication of vectors in $\mathbb{R}$. We use $\|$ to denote the concatenation of two strings. For an element g and vector $\mathbf{a} = (a_1, \ldots, a_n)$, we use $g^{\mathbf{a}}$ as a shorthand for $(g^{a_1}, \ldots, g^{a_n})$. Moreover, for any operation $*$ defined between two elements g and h and vectors $\mathbf{a} = (a_1, \ldots, a_n)$ and $\mathbf{b} = (b_1, \ldots, b_n)$, we use $g^{\mathbf{a}} * h^{\mathbf{b}}$ as a shorthand for $(g^{a_1} * h^{b_1}, \ldots, g^{a_n} * h^{b_1})$.

3.2 (Partially) Blind signatures

We define partially blind signatures consisting of three moves, which is sufficient to capture many known protocols, e.g., [4, 54, 55]. Below, we retrieve the standard definition of (three-move) blind signatures by ignoring the tag info or alternatively setting info to a predefined value.

Definition 1 (*Partially blind signature scheme*) A three-move *partially blind signature* PBS = (PBS.KGen, PBS.S, PBS.U, PBS.Verify) with an efficiently decidable public key space $\mathcal{PK}$ consists of the following PPT algorithms:

$\mathsf{PBS.KGen}(1^n) \to (\mathsf{pk}, \mathsf{sk})$: On input the security parameter 1^n, the key generation algorithm outputs a pair of public and secret keys $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{PBS.S} = (\mathsf{PBS.S}_1, \mathsf{PBS.S}_2)$: The interactive signer algorithm consists of two phases:

$\mathsf{PBS.S}_1(\mathsf{sk}, \mathsf{info}) \to (\mathsf{state}_\mathsf{S}, \rho_{\mathsf{S},1})$: On input a secret key sk and a tag info, it outputs an internal signer state $\mathsf{state}_\mathsf{S}$ and a first-sender message $\rho_{\mathsf{S},1}$.[7]

$\mathsf{PBS.S}_2(\mathsf{state}_\mathsf{S}, \rho_\mathsf{U})) \to \rho_{\mathsf{S},2}$: On input a signer state $\mathsf{state}_\mathsf{S}$ and a user message ρ_U, it outputs a second-sender message $\rho_{\mathsf{S},2}$.

$\mathsf{PBS.U} = (\mathsf{PBS.U}_1, \mathsf{PBS.U}_2)$: The interactive user algorithm consists of two phases:

$\mathsf{PBS.U}_1(\mathsf{pk}, \mathsf{info}, \mathsf{M}, \rho_{\mathsf{S},1}) \to (\mathsf{state}_\mathsf{U}, \rho_\mathsf{U})$: On input a public key $\mathsf{pk} \in \mathcal{PK}$, a tag info, a message M, and a first-sender message $\rho_{\mathsf{S},1}$, it outputs an internal user state $\mathsf{state}_\mathsf{U}$ and a user message ρ_U.

$\mathsf{PBS.U}_2(\mathsf{state}_\mathsf{U}, \rho_{\mathsf{S},2})) \to \sigma$: On input a user state $\mathsf{state}_\mathsf{U}$ and a second-signer message $\rho_{\mathsf{S},2}$, it outputs a signature σ.

$\mathsf{PBS.Verify}(\mathsf{pk}, \mathsf{info}, \mathsf{M}, \sigma) \to 1$ or 0 : In input a public key pk, a tag info, a message M, and a signature σ, the verification algorithm outputs 1 to indicate the signature is valid, and 0 otherwise.

If the partially blind signature only accepts a unique tag info, we drop the "partially" and simply call it a *blind signature* (BS) and omit info from the syntax.

We require a partially blind signature to be complete, blind against malicious signer, and one-more unforgeable. We first define correctness.

Definition 2 (*Perfect correctness*) A three-move partially blind signature scheme PBS is *perfectly correct* if for all public and secret key pairs $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{PBS.KGen}(1^n)$ and every tag and message pair $(\mathsf{info}, \mathsf{M})$, we have

$$\Pr\left[\mathsf{PBS.Verify}(\mathsf{pk}, \mathsf{info}, \mathsf{M}, \sigma) = 1 \middle| \begin{array}{r} (\mathsf{state}_\mathsf{S}, \rho_{\mathsf{S},1}) \xleftarrow{\$} \mathsf{PBS.S}_1(\mathsf{sk}, \mathsf{info}) \\ (\mathsf{state}_\mathsf{U}, \rho_\mathsf{U}) \xleftarrow{\$} \mathsf{PBS.U}_1(\mathsf{pk}, \mathsf{info}, \mathsf{M}, \rho_{\mathsf{S},1}) \\ \rho_{\mathsf{S},2} \xleftarrow{\$} \mathsf{PBS.S}_2(\mathsf{state}_\mathsf{S}, \rho_\mathsf{U}) \\ \sigma \xleftarrow{\$} \mathsf{PBS.U}_2(\mathsf{state}_\mathsf{U}, \rho_{\mathsf{S},2}) \end{array}\right] = 1$$

The following definitions are taken from [54, 55]. Partial blindness roughly requires the transcript to be independent of the signature even if the signer choses the keys maliciously.

Definition 3 (*Partial blindness under chosen keys*) We define partial blindness of a three-move partially blind signature scheme PBS via the following game between a challenger and an adversary $\mathcal{A}$:

Setup. The challenger samples $\mathsf{coin} \in \{0, 1\}$ and runs $\mathcal{A}$ on input 1^n.

Online Phase. When $\mathcal{A}$ outputs a tag info, messages $\widetilde{\mathsf{M}}_0$ and $\widetilde{\mathsf{M}}_1$, and a public key $\mathsf{pk} \in \mathcal{PK}$, it assigns $(\mathsf{M}_0, \mathsf{M}_1) := (\widetilde{\mathsf{M}}_\mathsf{coin}, \widetilde{\mathsf{M}}_{1-\mathsf{coin}})$. $\mathcal{A}$ is then given access to oracles $\mathsf{U}_1, \mathsf{U}_2$, which behave as follows.

Oracle U_1. On input $b \in \{0, 1\}$, and a first-signer message $\rho_{\mathsf{S},1,b}$, if the session b is not yet open, the oracle marks session b as $\mathtt{opened}$ and runs $\left(\mathsf{state}_{\mathsf{U},b}, \rho_{\mathsf{U},b}\right) \xleftarrow{\$} \mathsf{PBS.U}_1\left(\mathsf{pk}, \mathsf{info}, \mathsf{M}_b, \rho_{\mathsf{S},1,b}\right)$. It returns $\rho_{\mathsf{U},b}$ to $\mathcal{A}$.

[7] We assume without loss of generality that sk includes pk and $\mathsf{state}_\mathsf{S}$ includes $(\mathsf{pk}, \mathsf{sk})$ and omit it when the context is clear. Below, we also assume that $\mathsf{state}_\mathsf{U}$ includes M.

Oracle U_2. On input $b \in \{0,1\}$ and a second-signer message $\rho_{\mathsf{S},2,b}$, if the session b is $\mathtt{opened}$, the oracle creates a signature $\sigma_b \overset{\$}{\leftarrow} \mathsf{PBS.U}_2\left(\mathsf{state}_{\mathsf{U},b}, \rho_{\mathsf{S},2,b}\right)$. It marks session b as $\mathtt{closed}$. Oracle U_2 does not output anything.

Output Determination. When both sessions are closed and $\mathsf{PBS.Verify}(\mathsf{pk}, \mathsf{info}, \mathsf{M}_b, \sigma_b) = 1$ for $b \in \{0,1\}$, the oracle returns the two signatures $(\sigma_{\mathsf{coin}}, \sigma_{1-\mathsf{coin}})$ to $\mathcal{A}$, where note that σ_{coin} (resp. $\sigma_{1-\mathsf{coin}}$) is a valid signature for $\widetilde{\mathsf{M}}_0$ (resp. $\widetilde{\mathsf{M}}_1$) regardless of the choice of coin. $\mathcal{A}$ outputs a guess coin^* for coin. We say $\mathcal{A}$ *wins* if $\mathsf{coin}^* = \mathsf{coin}$.

We say PBS is *partially blind under chosen keys* if the advantage of $\mathcal{A}$ defined as $\Pr[\mathcal{A}$ wins$]$ is negligible.

One-more unforgeability roughly ensures that at most one valid signature is generated after each execution of PBS.Sign. Formally, we have the following.

Definition 4 (*One-more-unforgeability*) We define ℓ-one-more unforgeability (ℓ-OMUF) for any $\ell \in \mathbb{N}$ of a three-move partially blind signature scheme PBS via the following game between a challenger and an adversary $\mathcal{A}$:

Setup. The challenger samples $(\mathsf{pk}, \mathsf{sk}) \overset{\$}{\leftarrow} \mathsf{PBS.KGen}(1^n)$ and runs $\mathcal{A}$ on input pk. It further initializes $\ell_{\mathsf{closed}} = 0$ and $\mathsf{opened}_{\mathsf{sid}} = \mathtt{false}$ for all $\mathsf{sid} \in \mathbb{N}$.

Online Phase. $\mathcal{A}$ is given access to oracles S_1 and S_2, which behave as follows.

Oracle S_1: On input a tag info, the oracle samples a fresh session identifier sid. It sets $\mathsf{opened}_{\mathsf{sid}} \leftarrow \mathit{true}$ and generates $(\mathsf{state}_{\mathsf{S},\mathsf{sid}}, \rho_{\mathsf{S},1}) \overset{\$}{\leftarrow} \mathsf{PBS.S}_1(\mathsf{sk}, \mathsf{info})$. Then it returns sid and the first-sender message $\rho_{\mathsf{S},1}$ to $\mathcal{A}$.

Oracle S_2: On input a user message ρ_{U} and a session identifier sid, if $\ell_{\mathsf{closed}} \geq \ell$ or $\mathsf{opened}_{\mathsf{sid}} = \mathtt{false}$, then it returns $\perp$. Otherwise, it sets $\ell_{\mathsf{closed}}++$ and $\mathsf{opened}_{\mathsf{sid}} = \mathtt{false}$. It then computes the second-signer message $\rho_{\mathsf{S},2} \overset{\$}{\leftarrow} \mathsf{PBS.S}_2(\mathsf{state}_{\mathsf{S},\mathsf{sid}}, \rho_{\mathsf{U}})$ and returns $\rho_{\mathsf{S},2}$ to $\mathcal{A}$.

Output Determination. When $\mathcal{A}$ outputs distinct tuples $(\mathsf{M}_1, \sigma_1, \mathsf{info}_1), \ldots, (\mathsf{M}_k, \sigma_k, \mathsf{info}_k)$, we say $\mathcal{A}$ *wins* if $k \geq \ell_{\mathsf{closed}} + 1$ and for all $i \in [k]$, $\mathsf{PBS.Verify}(\mathsf{pk}, \mathsf{info}_i, \mathsf{M}_i, \sigma_i) = 1$.

We say PBS is *ℓ-one-more unforgeable* if the advantage of $\mathcal{A}$ defined as $\Pr[\mathcal{A}$ wins$]$ is negligible.

3.3 Sigma protocols

Definition 5 (*Sigma protocol*) A sigma protocol Σ for an **NP** relation R is a three-move public-coin interactive protocol with two pairs of PPT algorithms $\mathsf{P} = (\mathsf{P}_1, \mathsf{P}_2)$, V with the following flow:

- The prover on input a statement and witness pair $(\mathsf{X}, \mathsf{W}) \in R$, runs $(\mathsf{com}, \mathsf{state}) \overset{\$}{\leftarrow} \mathsf{P}_1(\mathsf{X}, \mathsf{W})$ and sends a *commitment* com to the verifier.
- The verifier samples a random *challenge* $\mathsf{ch} \overset{\$}{\leftarrow} \mathcal{C}$ from a specified challenge set, and sends ch to the prover.
- The prover runs $\mathsf{rsp} \overset{\$}{\leftarrow} \mathsf{P}_2(\mathsf{state}, \mathsf{ch})$ and returns a *response* rsp to the verifier.
- The verifier runs $\mathsf{V}(\mathsf{X}, \mathsf{com}, \mathsf{ch}, \mathsf{rsp})$ and outputs 1 to indicate the prover is valid and 0 otherwise.

To be useful as an (implicit) building block for blind signatures, a sigma protocol must satisfy correctness, honest verifier zero-knowledge (HVZK), witness indistinguishability, and special soundness, defined below.

Definition 6 (*Perfect completeness*) A sigma protocol is *perfectly correct* if whenever the protocol is executed by an honest prover and verifier (that is, a prover and verifier who follow the specification of the protocol), the verifier will return "Accept" with probability 1.

Definition 7 (*Special soundness*) A sigma protocol has *special soundness* if there is an efficient (*i.e.,* polynomial-time) extractor Ext which, given two accepting transcripts $\tau_1 = (\mathsf{com}, \mathsf{ch}_1, \mathsf{rsp}_1)$ and $\tau_2 = (\mathsf{com}, \mathsf{ch}_2, \mathsf{rsp}_2)$ for the same public key X, with $\mathsf{ch}_1 \neq \mathsf{ch}_2$, produces a witness W to the statement X.

Definition 8 (*Honest verifier zero-knowledge*) A sigma protocol is *honest verifier zero-knowledge* (HVZK) if there is an efficient algorithm Sim—the *simulator*—which, given a statement X outputs a transcript $\tau = (\mathsf{com}, \mathsf{ch}, \mathsf{rsp})$ such that the distribution of outputs of Sim is identical to the distribution of transcripts of honest executions of the protocol.

Witness indistinguishability is a weaker notion compared with HVZK, where we require the interactions between a prover using a witness W_1 or W_2 satisfying $(\mathsf{X}, \mathsf{W}_1), (\mathsf{X}, \mathsf{W}_2) \in R$ are indistinguishable. Namely, the interaction does not leak which witness is being used.

We also define a *hard instance generator* for the **NP** relation R as follows.

Definition 9 (*Hard instance generator*) An **NP** relation R is associated with an *instance generator* (IG) if IG, given as input the security parameter 1^n, outputs a statement-witness pair $(\mathsf{X}, \mathsf{W}) \in R$. Moreover, we say the instance generator is *hard* if the following holds for any PPT adversary $\mathcal{A}$:

$$\Pr[(\mathsf{X}, \mathsf{W}) \leftarrow \mathsf{IG}(1^n), \mathsf{W}' \leftarrow \mathcal{A}(\mathsf{X}) : (\mathsf{X}, \mathsf{W}') \in R] = \mathsf{negl}.$$

3.4 Elliptic curves and isogenies

Let E denote an elliptic curve over a finite field $\mathbb{F}_p$ with p a large prime, and let $\mathbf{0}_E$ be the point at infinity on E. The curve E is called supersingular if and only if $\#E\left(\mathbb{F}_p\right) = p+1$. Therefore, by using point counting or Schoof's algorithm [80], one can verify the supersingularity of a given curve efficiently. Otherwise, the curve is called ordinary curve. Given two elliptic curves E and E', an isogeny ϕ is a morphism $\phi : E \to E'$, namely, isogeny is a map given by rational functions and it is a group homomorphism such that $\phi\left(\mathbf{0}_E\right) = \mathbf{0}_{E'}$. An isomorphism is an isogeny whose inverse over the algebraic closure is also an isogeny and two elliptic curves are isomorphic if and only if they have the same j-invariant. There is a one-to-one correspondence from finite subgroups of an elliptic curve to separable isogenies from said curve, up to post-composition with isomorphisms. To be more specific, any subgroup $S \subset E\left(\mathbb{F}_{p^k}\right)$ determines a (separable) isogeny $\phi : E \to E'$ with $\ker \phi = S$, i.e. $E' = E/S$. Given subgroup S, the equation for E' and the isogeny ϕ can be computed using Vélu's formulae using $O\left(\#S(k \log p)^2\right)$ bit-operations. As a result, only those isogenies who kernels are small subgroups S defined over extensions $\mathbb{F}_{p^k}$ of small degree k can be computed efficiently.

The ring of endomorphisms $\mathrm{End}(E)$ consists of all isogenies from E to itself, and $\mathrm{End}_p(E)$ denotes the ring of endomorphisms defined over $\mathbb{F}_p$.

When $E/\mathbb{F}_p$ is supersingular, the endomorphism ring $\mathrm{End}_p(E)$ is isomorphic to an order $\mathcal{O}$ of the quadratic field $\mathbb{Q}(\sqrt{-p})$ [24]. We recall that an order is a subring of $\mathbb{Q}(\sqrt{-p})$, which is also a finitely-generated $\mathbb{Z}$-module containing a basis of $\mathbb{Q}(\sqrt{-p})$ as a $\mathbb{Q}$-vector space. A fractional ideal $\mathfrak{a}$ of $\mathcal{O}$ is a finitely generated $\mathcal{O}$-submodule of $\mathbb{Q}(\sqrt{-p})$. We say that $\mathfrak{a}$ is invertible if there exists another fractional ideal $\mathfrak{b}$ of $\mathcal{O}$ such that $\mathfrak{a}\mathfrak{b} = \mathcal{O}$, and that it is principal if $\mathfrak{a} = \alpha\mathcal{O}$ for some $\alpha \in \mathbb{Q}(\sqrt{-p})$. The invertible fractional ideals of $\mathcal{O}$ form an Abelian group whose quotient by the subgroup of principal fractional ideals is finite. This quotient group is called the *ideal class group* of $\mathcal{O}$, and denoted by $\mathcal{C\ell}(\mathcal{O})$.

The ideal class group $\mathcal{C\ell}(\mathcal{O})$ acts freely and transitively on the set $\mathcal{E\ell\ell}_p(\mathcal{O}, \pi)$, which contains all supersingular elliptic curves E over $\mathbb{F}_p$-modulo isomorphisms defined over $\mathbb{F}_p$—such that there exists an isomorphism between $\mathcal{O}$ and $\mathrm{End}_p(E)$ mapping $\sqrt{-p} \in \mathcal{O}$ to the Frobenius endomorphism $\pi : (x, y) \mapsto (x^p, y^p)$. When no confusion will arise, we will abbreviate $\mathcal{E\ell\ell}_p(\mathcal{O}, \pi)$ as $\mathcal{E\ell\ell}$.

The quadratic twist of a given elliptic curve $E : y^2 = f(x)$ is $E^{-1} : dy^2 = f(x)$ where $d \in \mathbb{F}_p^\times \backslash \mathbb{F}_p^{\times 2}$. When $p = 3 \bmod 4$ and E_0 is of j-invariant 1728, then E_0 and E_0^{-1} are $\mathbb{F}_p$-isomorphic. The quadratic twist can be efficiently computed in this setting. When $p = 3 \bmod 4$, the quadratic twist $E' : -y^2 = x^3 + Ax^2 + x$ of $E_A : y^2 = x^3 + Ax^2 + x$ is $\mathbb{F}_p$-isomorphic to E_{-A} by considering $(x, y) \mapsto (-x, y)$. Further, $([\mathfrak{a}] * E_0)^{-1} = [\mathfrak{a}]^{-1} * E_0$ for any $[\mathfrak{a}] \in \mathcal{C\ell}(\mathcal{O})$. Therefore, for any curve $E \in \mathcal{E\ell\ell}_p(\mathcal{O}, \pi)$, we have, by the transitivity of the action,

$$([\mathfrak{a}] * E)^{-1} = [\mathfrak{a}]^{-1} * E^{-1}.$$

Remark 1 Throughout the rest of the paper, we consider the underlying prime $p = 3 \bmod 4$. We assume the structure of the ideal class group $G = \langle [\mathfrak{g}] \rangle \cong \mathbb{Z}_N$, justified by the Cohen-Lenstra heuristic, is known for some $N \in \mathbb{N}$ and for each $i \in [N]$ the action $[\mathfrak{g}^i] * E$ can be efficiently evaluated. The setup is justified by [12].

Let $E_0 \in \mathcal{E\ell\ell}$ be the supersingular curve of j-invariant 1728. Our cryptosystems rely on the following assumptions.

Definition 10 (*Group action inverse problem (GAIP)*) Given $(E_0, E') \in \mathcal{E\ell\ell}^2$ where $E' = [\mathfrak{g}^s] * E_0$ and $s \xleftarrow{\$} [N]$, the *group action inverse problem* is to find $[\mathfrak{g}'] \in G$ such that $[\mathfrak{g}'] * E_0 = E'$.

The problem is equivalent to finding the exponent $s \bmod N$ by considering $f(m, n) = [\mathfrak{g}^m \mathfrak{g}'^n] \star E_0$ and applying the quantum period finding algorithm.

Recall that $G \cong \mathbb{Z}_N$ and $\mathbb{Z}_N$ is a ring. We introduce a generalized version of the group action inverse problem by considering a *d-th primitive root of unity*, denoted by ζ_d, over $\mathbb{Z}_N$ such that $\zeta_d^d = 1$ and $\zeta_d^j \neq 1$ for any $j \in [d-1]$. We define the ring group action inverse problem with respect to ζ_d as follows.

Definition 11 (*ζ_d-Ring group action inverse problem (rGAIP)*) Given $(E_0, S) \in \mathcal{E\ell\ell}^{d+1}$ where $S = ([\mathfrak{g}^{s\zeta_d^j}] * E_0)_{j\in[d]}$, $s \xleftarrow{\$} [N]$ and $d|\lambda(N)$ (here λ is the Carmichael function), the *ζ_d-ring group action inversion problem* (ζ_d-rGAIP) is to recover s.

When the context is clear, we may remove d from the subscript or remove ζ_d entirely and call it rGAIP for simplicity. This problem is a generalized version of GAIP, which is a ζ_2-rGAIP with $\zeta_2 = -1$. To see this, by taking the quadratic twist of a GAIP instance $E' = [\mathfrak{g}^s] * E_0$, we have $(E', E'^{-1}) = ([\mathfrak{g}^s] * E_0, [\mathfrak{g}^{-s}] * E_0)$. Such a ζ_d exists if d is a divisor of the Carmichael function $\lambda(N)$. Concretely, if $N = \Pi p_i^{e_i}$ where p_i are distinct primes, we have $\lambda(N) = \mathsf{lcm}_i(\lambda(p_i^{e_i}))$ where

$$\lambda(p_i^{e_i}) = \begin{cases} \frac{1}{2}\varphi(p_i^{e_i}) & \text{if } p_i = 2 \wedge e_i \geq 3 \\ \varphi(p_i^{e_i}) & \text{otherwise} \end{cases}$$

where φ is the Euler phi-function. Similar to GAIP [18, 25, 43] having polynomial-time HSP algorithms for insecure group structures, the hardness of an ζ_d-rGAIP also relies on the underlying algebraic structure and the specific choice of ζ_d. In Sect. 8.2, we provide a structural analysis on the ζ_d-rGAIP for CSIDH-512 and display a few weak and hard instances depending on ζ_d. We show that for some carefully-chosen d (depending on N), ζ_d-rGAIP is essentially as hard as the original GAIP.

Finally, when constructing our optimized blind signatures in Sect. 7, we require d to satisfy a bit more requirement other than ζ_d-rGAIP being hard. Informally, we require $\eta_d = \mathsf{lcm}_{i \in [d-1]}(\mathsf{gcd}(\zeta_d^i - 1, N))$ to be small for the extractor of the underlying sigma protocol to be efficient. More details can be found in Sect. 7.

4 Generic proofs for blind Schnorr-type signatures

In this section, we review the recent work of Kastner et al. [55] that provided a proof of the Abe-Okamoto (partially) blind signature [4]. The original security proof of the one-more unforgeability in [4] contained a leap of logic in the security proof (i.e., the scheme was correct but the security proof was not), and Kastner, Loss, and Xu provided a somewhat generic proof that works for many of the blind Schnorr-type signatures [29].[8] While their focus was on the scheme by Abe and Okamoto, the proof is generic enough to capture other similar schemes (see for instance [55, Appendix F] that provides a proof sketch of [2]). Indeed, the constructions we propose fall under their generic proofs as well. To this end, we extract the minimal definitions and lemmas from [55] required to argue the security of our (partially) blind signatures. Here, we note that it is likely that one can rewrite [55] in a more generic fashion by borrowing the tools from [49]. However, we chose not to for better readability and since isogenies do not naturally endow a *linear* identification scheme as required by [49]. Finally, we emphasize that while this section is not contained in Sect. 3 (i.e., Background), we do not claim any technical novelty of it.

Below, we provide a brief overview of the proof by Kastner, Loss, and Xu and then introduce the key lemmas that need to be proven in this paper to apply their proof.

4.1 Proof overview

Loosely speaking, a blind Schnorr-type signature is a type of blind signature that builds on top of a Schnorr-type sigma protocol [78]. The signer of the blind signature is identical to the prover in a sigma protocol, while the user of the blind signature modifies the verifier in the sigma protocol by appropriately adding blindness factors. In the proof of one-more unforgeability, the adversary (i.e., a malicious user) does not care if its forgeries are blind, and thus, how the blindness is achieved can be ignored for now.

At a high level, to argue one-more unforgeability, we would like the reduction to embed a hard problem into the public key of the blind signature and appeal to the special soundness of the underlying sigma protocol to extract a solution from the forgeries. However, unlike standard Fiat–Shamir-based signatures, the reduction cannot rely on HVZK to simulate the signatures since the challenge is under the adversary's control. To simulate the interaction between the adversary, we thus allow the public key to have *two* valid secret keys, e.g.,

[8] Note that the proof in [55] relies on the fact that there are two possible signing keys per public key. Therefore, their proof does not work for the original Schnorr blind signature [29], which is known to be secure if we further rely on the algebraic group model [54].

$(\mathsf{vk} = (E_0, [\mathfrak{g}^{a_0}] * E_0, [\mathfrak{g}^{a_1}] * E_0), \mathsf{sk} = (\delta, a_\delta))$ with $\delta \in \{0, 1\}$. The reduction embeds a hard problem into one of the secret keys while simulating with the other secret key.

What makes the security proof of blind Schnorr-type signatures tricky is that even if the adversary's view is independent of the secret key being used, this alone does not complete the proof. This is because to argue that the secret key extracted via the special soundness of the underlying sigma protocol is unbiased, we need to argue that the algorithm (i.e., reduction) executing the extractor of the special soundness is unbiased. While this holds for standard Fiat–Shamir based signature schemes since the reduction can invoke HVZK, this is not the case for blind signatures. As we discussed above, since the adversary chooses the challenge, the reduction can only try to invoke witness indistinguishability. However, witness indistinguishability breaks when the reduction *rewinds* the adversary since the reduction needs to simulate two transcripts using the same first commitment of the sigma protocol. Thus, the reduction is not compatible with the definition of witness indistinguishability.

That being said since the view of the adversary (in each run) is independent of the secret key being used, intuition tells us that the extraction works: the only thing that's not working is the security proof. To overcome this issue, Kastner et al. [55] provides a detailed analysis of the probability of the reduction succeeding while implicitly relying on witness indistinguishability. We note that Abe and Okamoto [4] also rely on the same proof approach but included a subtle but non-trivially fixable flaw to compute the probability.

4.2 Key definitions, lemmas, and theorems

We extract the minimal definitions and lemmas from [55] in a self-contained manner so that the security of our (partially) blind signatures is established through several easy-to-state lemmas. For a more full exposition, we refer the readers to [55].

4.2.1 Preparation

We first assume the adversary against the one-more unforgeability game is restricted to make only $\ell + 1$ distinct hash queries to the random oracle, where $\ell + 1$ is the number of forgeries the adversary outputs. Moreover, as with any blind Schnorr-type signature, we assume each signature in the forgery is associated with a distinct hash query.[9] We also assume the public key of the (partially) blind signature has exactly two corresponding secret keys. More specifically, we assume the underlying sigma protocol is for the **NP** OR-relation R defined with respect to another **NP** relation R'. That is, $(\mathsf{X} := (\mathsf{X}'_0, \mathsf{X}'_1), \mathsf{W} := (\delta, \mathsf{W}'_\delta)) \in R$, where $(\mathsf{X}'_0, \mathsf{W}'_0), (\mathsf{X}'_1, \mathsf{W}'_1) \in R'$, X is the public key and W is the secret key. Finally, we assume the adversary's user-message ρ_U queried to the signing algorithm $\mathsf{PBS.S}_2$ satisfies $\rho_\mathsf{U} \in \mathcal{C}$, where $\mathcal{C}$ is the challenge space of the underlying sigma protocol for relation R (and R').

We first define the notion of *instances*. Roughly, an instance defines the signer's key and randomness. We present a variant of the definition of instances in [55, Definition 4] that is agnostic to the underlying sigma protocol. We provide an explicit description of instances, analogous to [55, Definition 4], when we detail our construction of (partial) blind signatures.

Definition 12 (*Instances*) Assume the public key of a blind Schnorr-type signature has exactly two corresponding secret keys $\mathsf{sk}_0 = (0, \mathsf{W}'_0)$ and $\mathsf{sk}_1 = (1, \mathsf{W}'_1)$. We define two

[9] For those unfamiliar with Schnorr-type signatures, we encourage to look at our concrete construction, where the meaning would be clear from context.

types of *instances* $\mathbf{I}$: A $\mathbf{0}$-side (resp. $\mathbf{1}$-side) instance consists of sk_0 (resp. sk_1) and the randomness used by the honest signer algorithm when the secret key is fixed to sk_0 (resp. sk_1), i.e., randomness excluding those used by the key generation algorithm.

The main argument of Kastner, Loss, and Xu boils down to arguing that the output of the extraction algorithm (i.e., forking algorithm) explained above is independent of the instances.

Let $\overrightarrow{h}$ be the vector of responses returned by the random oracle, where $|\overrightarrow{h}| = \ell + 1$, and let rand be the randomness used by the one-more unforgeability adversary. We define a deterministic wrapper algorithm $\mathcal{W}$ that simulates the interaction between the signer and the adversary given input $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$. $\mathcal{W}$ invokes the signer and the adversary on inputs $\mathbf{I}$ and rand, respectively, and uses $\overrightarrow{h}$ to answer the random oracle queries made by the adversary. We define $\mathcal{W}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ to output $\perp$ if the adversary aborts prematurely or fails to win the one-more unforgeability game, and otherwise, output what the adversary outputs. We then define the notion of *successful tuples* as follows.

Definition 13 (*Successful tuples*) We define the set of *successful tuples* as follows:

$$\mathsf{Succ} := \{(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \mid \mathcal{W}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \neq \perp\}.$$

We next define a sufficient condition to invoke the extraction algorithm of the underlying sigma protocol. This is a standard definition (often implicitly) used even for Fiat-Shamir based signatures.

Definition 14 (*Successful Forking* [55, Definition 7]) We say two successful input tuples $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in \mathsf{Succ}$ fork from each other at index $i \in [\ell + 1]$ if $\overrightarrow{h}_{[i-1]} = \overrightarrow{h}'_{[i-1]}$ but $h_i \neq h'_i$. We denote the set of hash vector pairs (h_i, h'_i) such that $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in \mathsf{Succ}$ fork at index i as $\mathrm{F}_i(\mathbf{I}, \mathsf{rand})$.

We next define the notion of transcripts. A *query transcript* denotes the user messages queried to the signer. A *full transcript* denotes the entire transcript produced by the signer and the adversary, including the final forgery.

Definition 15 (*Query transcript* [55, Definition 5]) Consider the wrapper $\mathcal{W}$ running on input $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$. The *query transcript*, denoted $\overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, is the vector of user message (ρ_U) queries made to the signing algorithm $\mathsf{PBS.S}_2$ (simulated by $\mathcal{W}$) by the adversary, ordered by sid.

Definition 16 (*Full transcript* [55, Definition 6]) Consider the wrapper $\mathcal{W}$ running on input $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$. The *full transcript*, denoted $\mathsf{trans}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, is the transcript produced between the signer and the adversary, i.e., all messages sent between the signer and user played by the adversary, including the forgeries.

We now define *partners*, which plays a key role in the analysis of [4, 55]. Informally, two tuples $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in \mathsf{Succ}$ are partners at i if they fork at this index i and produce the same query transcript. Note that this does not nencessarily imply that each tuple results in the same full transcript.

Definition 17 (*Partners* [55, Definition 8]) We say two successful tuples $(\mathbf{I}, rand, \overrightarrow{h}), (\mathbf{I}, rand, \overrightarrow{h}')$ are partners at index $i \in [\ell + 1]$ if the followings hold:

- $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ fork at index i.
- $\overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = \overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$

We denote the set of $(\overrightarrow{h}, \overrightarrow{h}')$ such that $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ are partners at index i by $\mathrm{prt}_i(\mathbf{I}, \mathsf{rand})$.

A *triangle* is another key tool introduced in [4, 55] in order to enhance the standard forking tuples with the nice properties of partners. A triangle consists of three vectors $\overrightarrow{h}, \overrightarrow{h}', \overrightarrow{h}''$ such that each two vectors fork at the same index, and additionally, $(\overrightarrow{h}, \overrightarrow{h}')$ are partners.

Definition 18 (*Triangles* [55, Definition 9]) A triangle at index $i \in [\ell + 1]$ with respect to $\mathbf{I}, \mathsf{rand}$ is a tuple of three successful tuples in the following set:

$$\triangle_i(\mathbf{I}, \mathsf{rand}) = \left\{ \begin{array}{l|l} ((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), & (\overrightarrow{h}, \overrightarrow{h}') \in \mathrm{prt}_i(\mathbf{I}, \mathsf{rand}) \\ (\mathbf{I}, \mathsf{rand}\, \overrightarrow{h}'), & (\overrightarrow{h}, \overrightarrow{h}'') \in \mathrm{F}_i(\mathbf{I}, \mathsf{rand}) \\ (\mathbf{I}, \mathsf{rand}\, \overrightarrow{h}'')) & (\overrightarrow{h}', \overrightarrow{h}'') \in \mathrm{F}_i(\mathbf{I}, \mathsf{rand}) \end{array} \right\}$$

For a triangle $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')) \in \triangle_i(\mathbf{I}, \mathsf{rand})$, we call the pair of tuples $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'))$ the base, and $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}''))$ and $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}''))$ the sides.

We next define a map that transforms a b-side instance into a $(1 - b)$-side instance for $b \in \{\mathbf{0}, \mathbf{1}\}$. Roughly, the map allows us to relate the number of triangles with a $\mathbf{0}$-side instance to those with a $\mathbf{1}$-side instance. We present a variant of the definition of instances in [55, Definition 12] that is agnostic to the underlying sigma protocol. We provide an explicit description of the map, analogous to [55, Definition 12], when we detail our construction of (partial) blind signatures.

Definition 19 (*Mapping instances via transcript*) For $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in \mathsf{Succ}$, we define $\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I})$ as a function that maps a $\mathbf{0}$-side instance $\mathbf{I}$ (resp. $\mathbf{1}$-side instance $\mathbf{I}$) to a $\mathbf{1}$-side instance $\mathbf{I}'$ (resp. $\mathbf{0}$-side instance $\mathbf{I}'$).

Finally, we formally define the witness extractor used by the reduction. We present a variant of the definition of witness extractor in [55, Definition 13] that is agnostic to the underlying sigma protocol. This is because the witness extractor's concrete description is defined using the special soundness extractor of the underlying sigma protocol, which we will do when we detail our construction of (partial) blind signatures.

Definition 20 (*Witness extraction*) Fix $\mathbf{I}, \mathsf{rand}$ and let $\overrightarrow{h}, \overrightarrow{h}' \in \mathrm{F}_i(\mathbf{I}, \mathsf{rand})$ for some $i \in [\ell + 1]$. Moreover, denote σ_i, σ_i' the signatures that correspond to h_i, h_i', respectively. We say deterministic algorithms $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ are *witness extractors* if $(\mathsf{Ext}_0(\sigma_i, \sigma_i'), \mathsf{Ext}_1(\sigma_i, \sigma_i')) \in \{(\mathsf{sk}_0, \bot), (\bot, \mathsf{sk}_1), (\mathsf{sk}_0, \mathsf{sk}_1)\}$.[10] For $b \in \{0, 1\}$, we say that the *b-side witness can be extracted from* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ *and* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ *at index i* if $\mathsf{Ext}_b(\sigma_i, \sigma_i')$ outputs sk_b.

[10] [55] defined the witness extractors in such a way that it outputs only $(\mathsf{sk}_0, \bot)$ or $(\bot, \mathsf{sk}_1)$. However, this restriction is not required as long as Lemma 1 (i.e., [55, Corollary 3]) holds. We note that we need this extra relaxation for it to be useful in our *partially* blind signature. Moreover, note that the extractors are only required to output W_b' included in $\mathsf{sk}_b = (b, \mathsf{W}_b')$. We use W_b' and sk_b interchangeably for readability.

4.2.2 Sufficient condition for one-more unforgeability

We are now prepared to formally present the main result of Kastner et al. [55]. First of all, if the map $\Phi_{\mathsf{rand},\overrightarrow{h}}$ is a bijection that preserves transcripts for any rand and $\overrightarrow{h}$, then a partner tuple with a b-side instance maps to another partner tuple with a $(1-b)$-side instance for the same rand and $\overrightarrow{h}$ (see [55, Corollary 1 and Lemma 3]). This implies that the extracted witness from a partner tuple is independent of the reduction's secret key. However, it is not clear if the reduction is able to obtain a partner tuple by rewinding. To this end, we use the sides of the triangle rather than the base (i.e., partner tuple) to extract a witness, where the main observation is that if a b-side witness can be extracted from the base of a triangle, then a b-side witness can be extracted from at least one of the sides. Then, we argue that the reduction having a b-side witness hits one corner of the base of a triangle in the first run, and then hits the top of the triangle such that it creates side with a $(1-b)$-side witness with a probability of roughly 1/2.

The main contribution of Kastner et al. [55] was to make the above high-level argument precise. Their result is mostly purely statistical and it suffices to only prove that our (partial) blind signature satisfies the following two lemmas to invoke their main theorem concerning one-more unforgeability. The first lemma shows that the blind signature is perfectly *witness indistinguishable*. This is used to establish the extracted witness from a partner tuple is independent of the reduction's secret key.

Lemma 1 ([55, Lemma 2]) *Fix* $\mathsf{rand}, \overrightarrow{h}$. *For all tuples* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in \mathsf{Succ}$, $\Phi_{\mathsf{rand},\overrightarrow{h}}$ *is a self-inverse bijection and* $\mathsf{trans}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = \mathsf{trans}(\Phi_{\mathsf{rand},\overrightarrow{h}}(\mathbf{I}), \mathsf{rand}, \overrightarrow{h})$.

The second lemma states that if a witness can be extracted from a base of a triangle, then the same witness can be extracted from at least one of its sides.

Lemma 2 ([55, Corollary 3]) *Fix* $\mathbf{I}, \mathsf{rand}$ *and let* $(\overrightarrow{h}, \overrightarrow{h}', \overrightarrow{h}'') \in \triangle_i(\mathbf{I}, \mathsf{rand})$, *for some* $i \in [\ell+1]$. *If the* ***0****-side (****1****-side) witness can be extracted from the base* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ *of the triangle at index* i, *then one can also extract the* ***0****-side (****1****-side) witness from at least one of the sides* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ *or* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ *at index* i.

The following is the main theorem of Kastner et al. [55, Theorem 1] casted slightly generally to be agnostic to the underlying hardness assumption.

Theorem 3 *Let the (partially) blind Schnorr-type signature* $(\mathsf{P})\mathsf{BS}$ *be as defined in the preparation of Sect. 4.2. In particular, assume the public key consists of two instances of the* ***NP*** *relation* R' *generated by a corresponding hard instance generator* IG *and the underlying sigma protocol has challenge space* $\mathcal{C}$.

If Lemmas 1 and 2 hold, then for all $\ell \in \mathbb{N}$, *if there exists an adversary* $\mathcal{A}$ *that makes* Q *hash queries to the random oracle and breaks the* ℓ*-one more unforgeability of* $(\mathsf{P})\mathsf{BS}$ *with advantage* $\epsilon_{\mathcal{A}} \geq \frac{C_1}{|\mathcal{C}|} \cdot \binom{Q}{\ell+1}$, *then there exists an algorithm* $\mathcal{B}$ *that breaks the hard instance generator with advantage* $\epsilon_{\mathcal{B}} \geq C_2 \cdot \frac{\epsilon_{\mathcal{A}}^2}{\binom{Q}{\ell+1}^2 \cdot (\ell+1)^3}$ *for some universal positive constants* C_1 *and* C_2.

We note that Kastner, Loss, and Xu only show the above theorem for blind signatures. They then show that it can be extended to a proof for their particular partially blind signature with a

$$\begin{array}{lcl}
\hline
\textsf{P}: \begin{array}{l}\textsf{X} = A = [\mathfrak{g}^a] * E_0 \\ \textsf{W} = a \in \mathbb{Z}_N\end{array} & & \textsf{V}: \ \textsf{X} = A = [\mathfrak{g}^a] * E_0 \\
\hline
\begin{array}{l} y \xleftarrow{\$} \mathbb{Z}_N \\ Y = [\mathfrak{g}^y] * E_0 \end{array} & \xrightarrow{\qquad\qquad Y \qquad\qquad} & \\
 & \xleftarrow{\qquad\qquad c \qquad\qquad} & c \xleftarrow{\$} \{-1, 1\} \\
r = y - a \cdot c & \xrightarrow{\qquad\qquad r \qquad\qquad} & \begin{array}{l}\text{Accept if} \\ [\mathfrak{g}^r] * A^c = Y\end{array} \\
\hline
\end{array}$$

Fig. 1 The basic Sigma protocol underlying our blind signature scheme and partially-blind signature scheme

loss of $1/T$, where T is the number of the distinct tag info queries by the adversary (see [55, Theorem 2]). However, as explained in the introduction, we cannot follow their approach since our partially blind signature must deviate from prior constructions. To this end, we notice that the same proofs and theorem above can be applied to the partially blind setting if the instances in Definition 12 can be defined independently from the tags info used by the adversary. See Sect. 6 for more details.

5 Constructing isogeny-based blind signatures

In this section, we provide our isogeny-based blind signature. We first explain the sigma protocol that underlies our isogeny-based blind signature and then show how to compile it into a blind signature.

5.1 Our basic sigma protocol for isogeny knowledge

First, we introduce the basic sigma protocol that we use to construct the OR-proofs which form the basis for our blind signature in Sect. 5 and our partially blind signature in Sect. 6. Though the protocol is essentially standard, we include this discussion because this Sigma protocol is *not* simply the protocol used in CRS [32, 76] adapted to the supersingular setting (as in CSI-FiSh [12])—rather, our proof uses the quadratic twist in a fundamental way, which is necessary when constructing our signature schemes.

To begin, our protocol is depicted in Fig. 1.

We prove that the scheme depicted in Fig. 1 is a secure sigma protocol; that is, that it satisfies perfect completeness, special soundess, and honest verifier zero-knowledge (HVZK).

Lemma 4 (Perfect completeness) *The protocol depicted in Fig. 1 is perfectly complete.*

Proof Suppose that the protocol is executed according to the specification. Then

$$[\mathfrak{g}^r] * A^c = [\mathfrak{g}^{y-a\cdot c}] * [\mathfrak{g}^{a\cdot c}] * E_0 = [\mathfrak{g}^y] * E_0 = Y$$

so that V accepts, as required. □

Lemma 5 (Special soundess) *The protocol depicted in Fig. 1 satisfies special soundness.*

Proof Using the notation of Fig. 1, without loss of generality we may assume that $c = 1$ and $c' = -1$. Then

$$r' - r = (y + a) - (y - a) = 2a.$$

Recall the parameter $p \equiv 3 \pmod 4$ implies $|\mathcal{C}\ell(\mathcal{O})|$ is odd. Therefore, we can solve for the unique value of $a \in \mathbb{Z}_N$ as

$$a \equiv 2^{-1}(r' - r) \pmod N.$$

□

Lemma 6 (Honest verifier zero-knowledge) *The protocol depicted in Fig. 1 satisfies the honest verifier zero-knowledge property.*

Proof For a fixed statement $\mathsf{X} = [\mathfrak{g}^a] * E_0$, the distribution of honest transcripts is uniform on the set

$$\begin{aligned} T &= \{(Y = [\mathfrak{g}^y] * E_0,\ c,\ r = y - a \cdot c)\ :\ y \in \mathbb{Z}_N, c \in \{-1, 1\}\} \\ &= \{(Y = [\mathfrak{g}^{r+ac}] * E_0,\ c,\ r)\ :\ r \in \mathbb{Z}_N, c \in \{-1, 1\}\} \\ &= \{(Y = [\mathfrak{g}^r] * A^c,\ c,\ r)\ :\ r \in \mathbb{Z}_n, c \in \{-1, 1\}\}. \end{aligned} \tag{1}$$

Considering Eq. 1, we see that the following procedure will perfectly simulate the honest distribution of transcripts:

1. Choose $r \in \mathbb{Z}_N$ uniformly at random.
2. Choose $c \in \{-1, 1\}$ uniformly at random.
3. Set $Y = [g^r] * A^c$.

Thus we have defined the required Sim, and so the protocol satisfies the honest verifier zero-knowledge property. □

5.2 Base sigma protocol for an OR relation

Building on the protocol of Sect. 5.1 we consider a sigma protocol to prove that the prover knows at least *one of the two secrets* corresponding to the public statement $\mathsf{X} = (A_0, A_1) = ([\mathfrak{g}^{a_0}] * E_0, [\mathfrak{g}^{a_1}] * E_0)$. The sigma protocol is depicted in Fig. 2. Note that this is a standard isogeny-based sigma protocol where 0 is removed from the challenge space (see for instance [12]). As explained in Sect. 2.2, the main reason for this slight modification is to make the (non-soundness amplified) challenge space $\{-1, 1\}$ to be a (multiplicative) subgroup of $\mathbb{Z}_N^\times$.

While these properties are implicit in the blind signature, we sketch the properties of our sigma protocol for completeness. Correctness can be verified through a routine check.

5.2.1 HVZK

Given a challenge $\mathbf{c}$, a zero-knowledge simulator Sim samples random $(\mathbf{c}_0, \mathbf{c}_1) \xleftarrow{\$} (\{-1, 1\}^n)^2$ and $(\mathbf{r}_0, \mathbf{r}_1) \xleftarrow{\$} \mathbb{Z}_N^2$ conditioned on $\mathbf{c}_0 \odot \mathbf{c}_1 = \mathbf{c}$. It then sets $\mathbf{Y}_b = [\mathfrak{g}^{\mathbf{r}_b}] * A_b^{\mathbf{c}_b}$ for $b \in \{0, 1\}$, and outputs the simulated transcript $\big((\mathbf{Y}_0, \mathbf{Y}_1), \mathbf{c}, (\mathbf{r}_0, \mathbf{r}_1, \mathbf{c}_0, \mathbf{c}_1)\big)$. Since there is a bijection between $\mathbf{r}_b$ and $\mathbf{Y}_b$ once $\mathbf{c}_b$ is fixed, this produces a transcript identically distributed as a real transcript.

P: $\mathsf{X} = (A_0, A_1) = ([\mathfrak{g}^{a_0}] * E_0, [\mathfrak{g}^{a_1}] * E_0)$ $\mathsf{W} = (\delta, a_\delta) \in \{0,1\} \times \mathbb{Z}_N$		V: $\mathsf{X} = (A_0, A_1)$
$\mathbf{y}_\delta \xleftarrow{\$} \mathbb{Z}_N^n$ $\mathbf{Y}_\delta = [\mathfrak{g}^{\mathbf{y}_\delta}] * E_0$ $\mathbf{c}_{1-\delta} \xleftarrow{\$} \{-1,1\}^n$ $\mathbf{r}_{1-\delta} \xleftarrow{\$} \mathbb{Z}_n^\delta$ $\mathbf{Y}_{1-\delta} = [\mathfrak{g}^{\mathbf{r}_{1-\delta}}] * A_{1-\delta}^{\mathbf{c}_{1-\delta}}$	$\xrightarrow{(\mathbf{Y}_0, \mathbf{Y}_1)}$	
	$\xleftarrow{\mathbf{c}}$	$\mathbf{c} \xleftarrow{\$} \{-1,1\}^n$
$\mathbf{c}_\delta = \mathbf{c} \odot \mathbf{c}_{1-\delta}$ $\mathbf{r}_\delta = \mathbf{y}_\delta - a_\delta \cdot \mathbf{c}_\delta$	$\xrightarrow{(\mathbf{r}_0, \mathbf{r}_1, \mathbf{c}_0, \mathbf{c}_1)}$	Accept if $\mathbf{c} = \mathbf{c}_0 \odot \mathbf{c}_1$ and $\forall b \in \{0,1\}, [\mathfrak{g}^{\mathbf{r}_b}] * A_b^{\mathbf{c}_b} = \mathbf{Y}_b$

Fig. 2 The base OR sigma protocol underlying our blind signature scheme

5.2.2 Witness indistinguishability

This is a direct consequence of the above since perfect HVZK implies perfect witness indistinguishability.

5.2.3 Special soundness

Let $\big((\mathbf{Y}_0, \mathbf{Y}_1), \mathbf{c}, (\mathbf{r}_0, \mathbf{r}_1, \mathbf{c}_0, \mathbf{c}_1)\big)$ and $\big((\mathbf{Y}_0, \mathbf{Y}_1), \mathbf{c}', (\mathbf{r}'_0, \mathbf{r}'_1, \mathbf{c}'_0, \mathbf{c}'_1)\big)$ be two valid transcripts such that $\mathbf{c} \neq \mathbf{c}'$. Since $\mathbf{c} \neq \mathbf{c}'$, either $\mathbf{c}_0 \neq \mathbf{c}'_0$ or $\mathbf{c}_1 \neq \mathbf{c}'_1$. Without loss of generality, assume $c_{0,1} \neq c'_{0,1}$, where $c_{0,1}$ and $c'_{0,1} \in \{-1, 1\}$ are the first elements of $\mathbf{c}_0$ and $\mathbf{c}'_0$, respectively. The extractor Ext then given such two valid transcripts outputs a witness $(0, a_0 = \frac{r_{0,1} - r'_{0,1}}{c_{0,1} - c'_{0,1}})$, where $r_{0,1}, r'_{0,1} \in \mathbb{Z}_N$ are the first elements of $\mathbf{r}_0$ and $\mathbf{r}'_0$. Note that, since $p \equiv 3 \pmod 4$, we have that N is odd, so that $c_{0,1} - c'_{0,1} \in \{-2, 2\}$ is invertible mod N. Let us verify the correctness of such an Ext. Since the two transcripts are valid, we have $[\mathfrak{g}^{r_{0,1}}] * A_0^{c_{0,1}} = [\mathfrak{g}^{r'_{0,1}}] * A_0^{c'_{0,1}}$. Plugging in $A_0 = [\mathfrak{g}^{a_0}] * E_0$, we have $[\mathfrak{g}^{r_{0,1} + c_{0,1} \cdot a_0}] * E_0 = [\mathfrak{g}^{r'_{0,1} + c'_{0,1} \cdot a_0}] * E_0$, where we use the fact $c_{0,1}, c'_{0,1} \in \{-1, 1\}$. Cleaning up the exponents, we obtain the desired a_0.

5.3 Description of our blind signature

We present our isogeny-based blind signature building on top of the base sigma protocol in Sect. 5.2. Let (p, N, E_0) be the public parameter specified as the underlying prime, the order of the group and the distinguished element, resp. Let $\mathfrak{g}$ be a generator of the ideal class group $\mathcal{C}\ell(\mathcal{O})$. We assume these parameters are provided to all algorithms. Let $\mathsf{H} : \{0, 1\}^* \to \{-1, 1\}^n$ be a hash function modeled as a random oracle in the security proof.

The following algorithms are summarized in Fig. 3.

BS.KGen (1^n): On input the security parameter 1^n, it samples a bit $\delta \xleftarrow{\$} \{0, 1\}$, $(a_0, a_1) \xleftarrow{\$} \mathbb{Z}_N^2$ and outputs a public key $\mathsf{pk} = (A_0, A_1) = ([\mathfrak{g}^{a_0}] * E_0, [\mathfrak{g}^{a_1}] * E_0)$ and secret key $\mathsf{sk} = (\delta, a_\delta)$.

```
BS.KGen(1^n)
101 :  $\delta \xleftarrow{\$} \{0,1\}$
102 :  $(a_0, a_1) \xleftarrow{\$} \mathbb{Z}_N^2$
103 :  $(A_0, A_1) \leftarrow ([\mathfrak{g}^{a_0}] * E_0, [\mathfrak{g}^{a_1}] * E_0)$
104 :  $\textbf{return}\ (\mathsf{pk} = (A_0, A_1), \mathsf{sk} = (\delta, a_\delta))$

BS.S_1(sk)
201 :  $\textbf{parse}\ (\delta, a_\delta) \leftarrow \mathsf{sk}$
202 :  $\mathbf{y}_\delta^* \leftarrow \mathbb{Z}_N^n$
203 :  $\mathbf{Y}_\delta^* \leftarrow [\mathfrak{g}^{\mathbf{y}_\delta^*}] * E_0$
204 :  $(\mathbf{c}_{1-\delta}^*, \mathbf{r}_{1-\delta}^*) \xleftarrow{\$} \{-1,1\}^n \times \mathbb{Z}_N^n$
205 :  $\mathbf{Y}_{1-\delta}^* \leftarrow [\mathfrak{g}^{\mathbf{r}_{1-\delta}^*}] * A_{1-\delta}^{\mathbf{c}_{1-\delta}^*}$
206 :  $\mathsf{state_S} \leftarrow (\mathbf{y}_\delta^*, \mathbf{c}_{1-\delta}^*, \mathbf{r}_{1-\delta}^*)$
207 :  $\textbf{return}\ (\mathsf{state_S}, \rho_{\mathsf{S},1} = (\mathbf{Y}_0^*, \mathbf{Y}_1^*))$

BS.U_1(pk, M, ρ_{S,1})
301 :  $\textbf{parse}\ (\mathbf{Y}_0^*, \mathbf{Y}_1^*) \leftarrow \rho_{\mathsf{S},1}$
302 :  $\textbf{for}\ b \in \{0,1\}$
303 :      $(\mathbf{d}_b, \mathbf{z}_b) \xleftarrow{\$} \{-1,1\}^n \times \mathbb{Z}_N^n$
304 :      $\mathbf{Z}_b \leftarrow [\mathfrak{g}^{\mathbf{z}_b}] * (\mathbf{Y}_b^*)^{\mathbf{d}_b}$
305 :  $\mathbf{c} \leftarrow \mathsf{H}(\mathbf{Z}_0 \| \mathbf{Z}_1 \| \mathsf{M})$
306 :  $\mathbf{c}^* \leftarrow \mathbf{c} \odot \mathbf{d}_0 \odot \mathbf{d}_1 \in \{-1,1\}^n$
307 :  $\mathsf{state_U} \leftarrow (\mathbf{d}_b, \mathbf{z}_b)_{b \in \{0,1\}}$
308 :  $\textbf{return}\ (\mathsf{state_U}, \rho_\mathsf{U} = \mathbf{c}^*)$

BS.S_2(state_S, ρ_U)
401 :  $\textbf{parse}\ (\mathbf{y}_\delta^*, \mathbf{c}_{1-\delta}^*, \mathbf{r}_{1-\delta}^*) \leftarrow \mathsf{state_S}$
402 :  $\textbf{parse}\ \mathbf{c}^* \leftarrow \rho_\mathsf{U}$
403 :  $\mathbf{c}_\delta^* \leftarrow \mathbf{c}^* \odot \mathbf{c}_{1-\delta}^* \in \{-1,1\}^n$
404 :  $\mathbf{r}_\delta^* \leftarrow \mathbf{y}_\delta^* - a_\delta \cdot \mathbf{c}_\delta^* \in \mathbb{Z}_N^n$
405 :  $\textbf{return}\ \rho_{\mathsf{S},2} = (\mathbf{c}_b^*, \mathbf{r}_b^*)_{b \in \{0,1\}}$

BS.U_2(state_U, ρ_{S,2})
501 :  $\textbf{parse}\ (\mathbf{d}_b, \mathbf{z}_b)_{b \in \{0,1\}} \leftarrow \mathsf{state_U}$
502 :  $\textbf{parse}\ (\mathbf{c}_b^*, \mathbf{r}_b^*)_{b \in \{0,1\}} \leftarrow \rho_{\mathsf{S},2}$
503 :  $\textbf{for}\ b \in \{0,1\}$
504 :      $(\mathbf{c}_b, \mathbf{r}_b) \leftarrow (\mathbf{c}_b^* \odot \mathbf{d}_b, \mathbf{z}_b + \mathbf{r}_b^* \odot \mathbf{d}_b)$
505 :  $\mathbf{c}' \leftarrow \mathsf{H}\Big([\mathfrak{g}^{\mathbf{r}_0}] * A_0^{\mathbf{c}_0} \| [\mathfrak{g}^{\mathbf{r}_1}] * A_1^{\mathbf{c}_1} \| \mathsf{M}\Big)$
506 :  $\textbf{if}\ \mathbf{c}_0 \odot \mathbf{c}_1 = \mathbf{c}'$
507 :      $\textbf{return}\ \sigma = (\mathbf{c}_b, \mathbf{r}_b)_{b \in \{0,1\}}$
508 :  $\textbf{return}\ \sigma = \perp$

BS.Verify(pk, M, σ)
601 :  $\textbf{parse}\ (\mathbf{c}_b, \mathbf{r}_b)_{b \in \{0,1\}} \leftarrow \sigma$
602 :  $\mathbf{c}' \leftarrow \mathsf{H}\Big([\mathfrak{g}^{\mathbf{r}_0}] * A_0^{\mathbf{c}_0} \| [\mathfrak{g}^{\mathbf{r}_1}] * A_1^{\mathbf{c}_1} \| \mathsf{M}\Big)$
603 :  $\textbf{if}\ \mathbf{c}_0 \odot \mathbf{c}_1 = \mathbf{c}'$
604 :      $\textbf{return}\ 1$
605 :  $\textbf{return}\ 0$
```

Fig. 3 Our blind signature scheme. We assume the algorithms return $\perp$ and terminate if **parse** is not in the correct format

$\mathsf{BS.S}_1(\mathsf{sk})$: The signer first samples $\mathbf{y}_\delta^* \xleftarrow{\$} \mathbb{Z}_N^n$ and sets $\mathbf{Y}_\delta^* = [\mathfrak{g}^{\mathbf{y}_\delta^*}] * E_0$. It then samples $(\mathbf{c}_{1-\delta}^*, \mathbf{r}_{1-\delta}^*) \xleftarrow{\$} \{-1, 1\}^n \times \mathbb{Z}_N^n$ and sets $\mathbf{Y}_{1-\delta}^* = [\mathfrak{g}^{\mathbf{r}_{1-\delta}^*}] * A_{1-\delta}^{\mathbf{c}_{1-\delta}^*}$. It then outputs the signer state $\mathsf{state_S} = (\mathbf{y}_\delta^*, \mathbf{c}_{1-\delta}^*, \mathbf{r}_{1-\delta}^*)$ and the first-sender message $\rho_{\mathsf{S},1} = (\mathbf{Y}_0^*, \mathbf{Y}_1^*)$.

$\mathsf{BS.U}_1(\mathsf{pk}, \mathsf{M}, \rho_{\mathsf{S},1})$: The user parses $(\mathbf{Y}_0^*, \mathbf{Y}_1^*) \leftarrow \rho_{\mathsf{S},1}$, samples $(\mathbf{d}_b, \mathbf{z}_b) \xleftarrow{\$} \{-1, 1\}^n \times \mathbb{Z}_N^n$, and computes $\mathbf{Z}_b = [\mathfrak{g}^{\mathbf{z}_b}] * (\mathbf{Y}_b^*)^{\mathbf{d}_b}$ for $b \in \{0, 1\}$. It then computes $\mathbf{c} = \mathsf{H}(\mathbf{Z}_0 \| \mathbf{Z}_1 \| \mathsf{M}) \in \{-1, 1\}^n$ and outputs the user state $\mathsf{state_U} = (\mathbf{d}_b, \mathbf{z}_b)_{b \in \{0,1\}}$ and user message $\rho_\mathsf{U} = \mathbf{c}^* = \mathbf{c} \odot \mathbf{d}_0 \odot \mathbf{d}_1$.

$\mathsf{BS.S}_2(\mathsf{state_S}, \rho_\mathsf{U})$: The signer parses $(\mathbf{y}_\delta^*, \mathbf{c}_{1-\delta}^*, \mathbf{r}_{1-\delta}^*) \leftarrow \mathsf{state_S}$, $\mathbf{c}^* \leftarrow \rho_\mathsf{U}$, sets $\mathbf{c}_\delta^* = \mathbf{c}^* \odot \mathbf{c}_{1-\delta}^* \in \{-1, 1\}^n$, and computes $\mathbf{r}_\delta^* = \mathbf{y}_\delta^* - a_\delta \cdot \mathbf{c}_\delta^* \in \mathbb{Z}_N^n$.[11] It then outputs the second-signer message $\rho_{\mathsf{S},2} = (\mathbf{c}_b^*, \mathbf{r}_b^*)_{b \in \{0,1\}}$.

$\mathsf{BS.U}_2(\mathsf{state_U}, \rho_{\mathsf{S},2})$: The user parses $(\mathbf{d}_b, \mathbf{z}_b)_{b \in \{0,1\}} \leftarrow \mathsf{state_U}$, $(\mathbf{c}_b^*, \mathbf{r}_b^*)_{b \in \{0,1\}} \leftarrow \rho_{\mathsf{S},2}$ and sets $(\mathbf{c}_b, \mathbf{r}_b) = (\mathbf{c}_b^* \odot \mathbf{d}_b, \mathbf{z}_b + \mathbf{r}_b^* \odot \mathbf{d}_b)$ for $b \in \{0, 1\}$. It then checks if

$$\mathbf{c}_0 \odot \mathbf{c}_1 = \mathsf{H}\Big([\mathfrak{g}^{\mathbf{r}_0}] * A_0^{\mathbf{c}_0} \| [\mathfrak{g}^{\mathbf{r}_1}] * A_1^{\mathbf{c}_1} \| \mathsf{M}\Big). \tag{2}$$

[11] Recall that we assume $\mathsf{state_S}$ includes sk (cf. Footnote 7).

If it holds, it outputs a signature $\sigma = (\mathbf{c}_b, \mathbf{r}_b)_{b\in\{0,1\}} \in \left(\{-1,1\}^n \times \mathbb{Z}_N^n\right)^2$, and otherwise a $\perp$.

$\mathsf{BS.Verify}(\mathsf{pk}, \mathsf{M}, \sigma)$: The verifier outputs 1 if Eq. 2 holds, and otherwise 0.

The correctness, blindness, and one-more unforgeability of our blind signature are provided in the subsequent sections.

5.4 Proof of correctness and blindness

Correctness can be checked by a routine calculation. For completeness, we provide the proof below.

Theorem 7 (Correctness) *The blind signature scheme in Fig. 3 is (perfectly) correct.*

Proof To show correctness, it suffices to show that Eq. 2 holds when both the signer and user follow the protocol. First, it can be checked that we have $\mathbf{Y}_b^* = [\mathfrak{g}^{\mathbf{r}_b^*}] * A_b^{\mathbf{c}_b^*}$ for $b \in \{0,1\}$. The case $b = 1-\delta$ holds by definition and the other case holds due to the correctness of the base OR sigma protocol (see Sect. 5.2). Then, substituting $(\mathbf{c}_b, \mathbf{r}_b) = (\mathbf{c}_b^* \odot \mathbf{d}_b, \mathbf{z}_b + \mathbf{r}_b^* \odot \mathbf{d}_b)$ for $b \in \{0,1\}$, we have

$$\begin{aligned}[\mathfrak{g}^{\mathbf{r}_b}] * A_b^{\mathbf{c}_b} &= [\mathfrak{g}^{\mathbf{z}_b + \mathbf{r}_b^* \odot \mathbf{d}_b}] * A_b^{\mathbf{c}_b^* \odot \mathbf{d}_b} \\ &= [\mathfrak{g}^{\mathbf{z}_b}] * \left([\mathfrak{g}^{\mathbf{r}_b^* \odot \mathbf{d}_b}] * A_b^{\mathbf{c}_b^* \odot \mathbf{d}_b}\right) = [\mathfrak{g}^{\mathbf{z}_b}] * (\mathbf{Y}_b^*)^{\mathbf{d}_b} = \mathbf{Z}_b. \end{aligned} \quad (3)$$

Finally, since $\mathbf{c} = \mathbf{c}^* \odot \mathbf{d}_0 \odot \mathbf{d}_1 = \mathbf{c}_0^* \odot \mathbf{c}_1^* \odot \mathbf{d}_0 \odot \mathbf{d}_1 = \mathbf{c}_0 \odot \mathbf{c}_1$, where $\mathbf{c} = \mathsf{H}(\mathbf{Z}_0 \| \mathbf{Z}_1 \| \mathsf{M})$, we obtain Eq. 2 as desired. Note that we use the fact that $x \odot x = 1$ for any $x \in \{-1,1\}$ in the first equality. □

The proof of blindness is also standard. Since checking A is a valid elliptic curve can be done efficiently and for such valid A, there exists a unique $a \in \mathbb{Z}_N$ such that $[\mathfrak{g}^a] * E_0 = A$, our blind signature is secure even against a malicious server outputting an arbitrary public key.

Theorem 8 (Blindness) *The blind signature scheme in Fig. 3 is (perfectly) blind under chosen keys.*

Proof It suffices to show that for any valid public key pk, any first and second-signer messages $\rho_{\mathsf{S},1} = (\mathbf{Y}_0^*, \mathbf{Y}_1^*)$ and $\rho_{\mathsf{S},2} = (\mathbf{c}_b^*, \mathbf{r}_b^*)_{b\in\{0,1\}} \in (\{-1,1\}^n \times \mathbb{Z}_N^n)^2$, and valid signature $\sigma = (\mathbf{c}_b, \mathbf{r}_b)_{b\in\{0,1\}} \in (\{-1,1\}^n \times \mathbb{Z}_N^n)^2$, there exists a unique and pair-wise distinct user state $\mathsf{state}_\mathsf{U} = (\mathbf{d}_b, \mathbf{z}_b)_{b\in\{0,1\}} \in \left(\{-1,1\}^n \times \mathbb{Z}_N^n\right)^2$ that could have generated σ. In other words, it suffices to show that fixing an arbitrary $(\mathsf{pk}, \rho_{\mathsf{S},1}, \rho_{\mathsf{S},2})$, there exists a bijection between a valid σ and $\mathsf{state}_\mathsf{U}$. Here, note that any public key $\mathsf{pk} = (A_0, A_1)$ output by the adversary (i.e., malicious signer) $\mathcal{A}$ can be efficiently checked to be valid elliptic curves (i.e., supersingularity). Below, we let $(a_0, a_1) \in \mathbb{Z}_N^2$ be the unique secret key $\mathsf{sk} = (a_0, a_1)$ such that $(A_0, A_1) = ([\mathfrak{g}^{a_0}] * E_0, [\mathfrak{g}^{a_1}] * E_0)$.

Let us fix $\mathsf{sk} = (a_0, a_1)$ (hence pk), $\rho_{\mathsf{S},1} = (\mathbf{Y}_0^*, \mathbf{Y}_1^*)$, $\rho_{\mathsf{S},2} = (\mathbf{c}_b^*, \mathbf{r}_b^*)_{b\in\{0,1\}}$, and a valid signature $\sigma = (\mathbf{c}_b, \mathbf{r}_b)_{b\in\{0,1\}}$. Let us further define the user state $\mathsf{state}_\mathsf{U} = (\mathbf{d}_b, \mathbf{z}_b)_{b\in\{0,1\}}$ as $\mathbf{d}_b = \mathbf{c}_b \odot \mathbf{c}_b^*$ and $\mathbf{z}_b = \mathbf{r}_b - \mathbf{r}_b^* \odot \mathbf{d}_b$ for $b \in \{0,1\}$. Following Eq. 3 from right to left, we have $\mathbf{Z}_b = [\mathfrak{g}^{\mathbf{r}_b}] * A_b^{\mathbf{c}_b}$ for $b \in \{0,1\}$. Combining this with σ being a valid signature, we have $\mathbf{c}_0 \odot \mathbf{c}_1 = \mathsf{H}\left([\mathfrak{g}^{\mathbf{r}_0}] * A_0^{\mathbf{c}_0} \| [\mathfrak{g}^{\mathbf{r}_1}] * A_1^{\mathbf{c}_1} \| \mathsf{M}\right) = \mathsf{H}(\mathbf{Z}_0 \| \mathbf{Z}_1 \| \mathsf{M})$. Therefore, $\mathsf{state}_\mathsf{U}$ is indeed a user state that results in the valid signature σ. Moreover, for any choice of $\rho_{\mathsf{S},2}$ and any $\sigma \neq \sigma'$, it can be checked that the corresponding user states $\mathsf{state}_\mathsf{U}$ and $\mathsf{state}'_\mathsf{U}$ defined as above are distinct. Hence, there is a bijection between a valid signature and a user state. This concludes the proof. $\square$

5.5 Proof of one-more unforgeability

Our proof of OMUF consists of preparing the necessary tools to invoke Theorem 3. Specifically, we define instances (see Definition 12), the map $\Phi_{\mathsf{rand},\overrightarrow{h}}$ (see Definition 19), the witness extractors $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ (see Definition 20) and prove that Lemmas 1 and 2 hold.

Below, we denote $\overrightarrow{X}$ as a shorthand for a vector $(X^{(1)}, \ldots, X^{(\ell)})$ and endow $\overrightarrow{X}$ with the same operations defined for $X^{(k)}$ by operating them component wise. Moreover, recall rand denotes the adversary's randomness, and $\overrightarrow{h} = (\mathbf{c}^{(1)}, \ldots, \mathbf{c}^{(\ell)})$ is the random oracle's response vector conditioned on the adversary making only ℓ random oracle queries. Finally, once the instance, adversary's randomness and hash output tuple $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ is fixed, the query transcript $\overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$—the vector of user message ρ_U queries made to the signing algorithm $\mathsf{BS.S}_2$—is defined. We denote this as $\overrightarrow{\mathbf{c}^*}$ below to be consistent with the notations used in our construction.

5.5.1 Preparation: instances

Let us first define the $\mathbf{0}$-side instance $\mathbf{I}_0$ and the $\mathbf{1}$-side instance $\mathbf{I}_1$. Below, we assume the adversary against the one-more unforgeability game makes ℓ-signing queries in total.

A $\mathbf{0}$-side instance $\mathbf{I}_0 = (0, a_0, A_1, \overrightarrow{\mathbf{y}_0^*}, \overrightarrow{\mathbf{c}_1^*}, \overrightarrow{\mathbf{r}_1^*})$ is defined as follows:

- $(0, a_0)$: The secret key sk when $\delta = 0$.
- A_1 : The part of the public key $\mathsf{pk} = (A_0, A_1)$ whose secret key is unknown.
- $\mathbf{y}_0^{*(k)}$: The exponent of the commitment $\mathbf{Y}_0^{*(k)}$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 0$ such that $\mathbf{Y}_0^{*(k)} = [\mathfrak{g}^{\mathbf{y}_0^{*(k)}}] * E_0$.
- $\mathbf{c}_1^{*(k)}$: The simulated challenge in the k-th ($k \in [\ell]$) first-sender message when $\delta = 0$.
- $\mathbf{r}_1^{*(k)}$: The exponent of the commitment $\mathbf{Y}_1^{*(k)}$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 0$ such that $\mathbf{Y}_1^{*(k)} = [\mathfrak{g}^{\mathbf{r}_1^{*(k)}}] * A_1^{\mathbf{c}_1^{*(k)}}$.

A $\mathbf{1}$-side instance $\mathbf{I}_1 = (1, a_1, A_0, \overrightarrow{\mathbf{y}_1^*}, \overrightarrow{\mathbf{c}_0^*}, \overrightarrow{\mathbf{r}_0^*})$ is defined as follows:

- $(1, a_1)$: The secret key sk when $\delta = 1$.
- A_0 : The part of the public key $\mathsf{pk} = (A_0, A_1)$ whose secret key is unknown.
- $\mathbf{y}_1^{*(k)}$: The exponent of the commitment $\mathbf{Y}_1^{*(k)}$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 0$ such that $\mathbf{Y}_1^{*(k)} = [\mathfrak{g}^{\mathbf{y}_1^{*(k)}}] * E_0$.
- $\mathbf{c}_0^{*(k)}$: The simulated challenge in the k-th ($k \in [\ell]$) first-sender message when $\delta = 1$.
- $\mathbf{r}_0^{*(k)}$: The exponent of the commitment $\mathbf{Y}_0^{*(k)}$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 0$ such that $\mathbf{Y}_0^{*(k)} = [\mathfrak{g}^{\mathbf{r}_0^{*(k)}}] * A_1^{\mathbf{c}_0^{*(k)}}$.

```
$\mathsf{Ext}_0(\sigma, \sigma')$
101 : if $\exists t \in [n]$ s.t. $c_{0,t} \neq c'_{0,t}$
102 :     return $a_0 = \frac{r_{0,t} - r'_{0,t}}{c_{0,t} - c'_{0,t}}$
103 : return $\perp$
```

```
$\mathsf{Ext}_1(\sigma, \sigma')$
101 : if $\exists t \in [n]$ s.t. $c_{1,t} \neq c'_{1,t}$
102 :     return $a_1 = \frac{r_{1,t} - r'_{1,t}}{c_{1,t} - c'_{1,t}}$
103 : return $\perp$
```

Fig. 4 Witness extractors for our blind signature. In the above, $\sigma = (\mathbf{c}_k, \mathbf{r}_k)_{k\in\{0,1\}}$ and $\sigma' = (\mathbf{c}'_k, \mathbf{r}'_k)_{k\in\{0,1\}}$, where $\mathbf{c}_k, \mathbf{c}'_k$ live in $\{-1, 1\}^n$ and $\mathbf{r}_k, \mathbf{r}'_k$ live in $\mathbb{Z}_N^n$. Non-bold font indicates the entries of a vector

5.5.2 Preparation: map $\Phi_{\mathsf{rand}, \overrightarrow{h}}$

We next define the map $\Phi_{\mathsf{rand}, \overrightarrow{h}}$ that maps a $\mathbf{0}$-side instance $\mathbf{I}_0$ into a $\mathbf{1}$-side instance $\mathbf{I}_1$ and vice versa. Concretely, a $\mathbf{0}$-side instance $\mathbf{I}_0 = (0, a_0, A_1, \overrightarrow{\mathbf{y}_0^*}, \overrightarrow{\mathbf{c}_1^*}, \overrightarrow{\mathbf{r}_1^*})$ maps to a $\mathbf{1}$-side instance $\mathbf{I}_1$ such that

$$\mathbf{I}_1 = \left(1,\ a_1,\ A_0 = [\mathfrak{g}^{a_0}] * E_0,\ \overrightarrow{\mathbf{y}_1^*} = \overrightarrow{\mathbf{r}_1^*} + a_1 \cdot \overrightarrow{\mathbf{c}_1^*},\ \overrightarrow{\mathbf{c}_0^*} = \overrightarrow{\mathbf{c}^*} \odot \overrightarrow{\mathbf{c}_1^*},\ \overrightarrow{\mathbf{r}_0^*} = \overrightarrow{\mathbf{y}_0^*} - a_0 \cdot \overrightarrow{\mathbf{c}_0^*}\right),$$

where a_1 is such that $[\mathfrak{g}^{a_1}] * E_0 = A_1$ and recall that $\overrightarrow{\mathbf{c}^*} = \overrightarrow{e}(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h})$. On the other hand, a $\mathbf{1}$-side instance $\mathbf{I}_1 = (1, a_1, A_0, \overrightarrow{\mathbf{y}_1^*}, \overrightarrow{\mathbf{c}_0^*}, \overrightarrow{\mathbf{r}_0^*})$ maps to a $\mathbf{0}$-side instance $\mathbf{I}_0$ such that

$$\mathbf{I}_0 = \left(0,\ a_0,\ A_1 = [\mathfrak{g}^{a_1}] * E_0,\ \overrightarrow{\mathbf{y}_0^*} = \overrightarrow{\mathbf{r}_0^*} + a_0 \cdot \overrightarrow{\mathbf{c}_0^*},\ \overrightarrow{\mathbf{c}_1^*} = \overrightarrow{\mathbf{c}^*} \odot \overrightarrow{\mathbf{c}_0^*},\ \overrightarrow{\mathbf{r}_1^*} = \overrightarrow{\mathbf{y}_1^*} - a_1 \cdot \overrightarrow{\mathbf{c}_1^*}\right),$$

where a_0 is such that $[\mathfrak{g}^{a_0}] * E_0 = A_0$ and recall that $\overrightarrow{\mathbf{c}^*} = \overrightarrow{e}(\mathbf{I}_1, \mathsf{rand}, \overrightarrow{h})$.

5.5.3 Preparation: witness extractors (Ext_0, Ext_1)

Fix $\mathbf{I}$, rand and let $(\overrightarrow{h}, \overrightarrow{h}') \in \mathrm{F}_i(\mathbf{I}, \mathsf{rand})$ for some $i \in [\ell + 1]$. Let us denote $\sigma = (\mathbf{c}_b, \mathbf{r}_b)_{b\in\{0,1\}}$ and $\sigma' = (\mathbf{c}'_b, \mathbf{r}'_b)_{b\in\{0,1\}}$ the signatures that correspond to $\mathbf{c}^{(i)}$ and $\mathbf{c}'^{(i)}$, respectively, where recall $\mathbf{c}^{(i)}$ (resp. $\mathbf{c}'^{(i)}$) is the i-th entry of $\overrightarrow{h}$ (resp. $\overrightarrow{h}'$). In particular, we have $\mathbf{c}_0 \odot \mathbf{c}_1 = \mathbf{c}^{(i)}$ and $\mathbf{c}'_0 \odot \mathbf{c}'_1 = \mathbf{c}'^{(i)}$. We define the witness extractors (Ext_0, Ext_1) as in Fig. 4.

The following lemma establishes the correctness of the witness extractors.

Lemma 9 (Ext_0, Ext_1) *in Fig. 4 satisfy the definition of witness extractors in Definition 20.*

Proof By the definition of $\mathrm{F}_i(\mathbf{I}, \mathsf{rand})$ (see Definition 14), we have $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in \mathsf{Succ}$ and $\mathbf{c}^{(i)} \neq \mathbf{c}'^{(i)}$. The former implies that the two signatures σ and σ' are valid. Concretely, we have

$$\mathbf{c}^{(i)} = \mathbf{c}_0 \odot \mathbf{c}_1 = \mathsf{H}\Big([\mathfrak{g}^{\mathbf{r}_0}] * A_0^{\mathbf{c}_0} \| [\mathfrak{g}^{\mathbf{r}_1}] * A_1^{\mathbf{c}_1} \| \mathsf{M}\Big)$$
$$\mathbf{c}'^{(i)} = \mathbf{c}'_0 \odot \mathbf{c}'_1 = \mathsf{H}\Big([\mathfrak{g}^{\mathbf{r}'_0}] * A_0^{\mathbf{c}'_0} \| [\mathfrak{g}^{\mathbf{r}'_1}] * A_1^{\mathbf{c}'_1} \| \mathsf{M}\Big).$$

Moreover, since $\overrightarrow{h}$ and $\overrightarrow{h}'$ agree up to the i-th entry and the challenger and adversary's randomness are fixed, the input to the hash functions agree. Namely, we have

$$[\mathfrak{g}^{\mathbf{r}_b}] * A_b^{\mathbf{c}_b} = [\mathfrak{g}^{\mathbf{r}'_b}] * A_b^{\mathbf{c}'_b} \text{ for } b \in \{0, 1\} \ \wedge\ \mathsf{M} = \mathsf{M}'.$$

Since $\mathbf{c}^{(i)} \neq \mathbf{c}'^{(i)}$, we must have $\mathbf{c}_0 \neq \mathbf{c}'_0$ or $\mathbf{c}_1 \neq \mathbf{c}'_1$. Based on the special-soundness of the underlying sigma protocol (see Sect. 5.2), one of Ext_0 or Ext_1 always outputs a valid secret key. This completes the proof. □

5.5.4 Proof of one-more unforgeability

We prove the following two lemmas required to invoke the main theorem Theorem 3.

Lemma 10 *Lemma 1 holds for our definition of the map* $\Phi_{\mathsf{rand}, \overrightarrow{h}}$.

Proof Since the proof for the $\mathbf{0}$-side and $\mathbf{1}$-side instances $\mathbf{I}_0$ and $\mathbf{I}_1$ are analogous, we only consider the $\mathbf{0}$-side instance. For any $\mathsf{rand}, \overrightarrow{h}$, let us consider the query transcript $\overrightarrow{e}(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h}) = \overrightarrow{\mathbf{c}^*}$, i.e., the vector of user message ρ_U queries made by the adversary to the signing algorithm $\mathsf{BS.S}_2$. Since the underlying sigma protocol is perfectly witness indistinguishable (see Sect. 5.2), for each $i \in [\ell]$ and $\mathbf{c}^{*(i)}$, there is a set of randomness that the signer with a secret key $(1, a_1)$ (i.e., a $\mathbf{1}$-side witness) could have used to produce the same view (i.e., first and second-signer messages) to the adversary. Concretely, this set of randomness is exactly those defined by $\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}_0)$. Hence, we have $\mathsf{trans}(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h}) = \mathsf{trans}(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}_0), \mathsf{rand}, \overrightarrow{h})$ as desired. Moreover, it is easy to check that $\Phi_{\mathsf{rand}, \overrightarrow{h}}(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}_0))$ from the definition of $\Phi_{\mathsf{rand}, \overrightarrow{h}}$. Hence, it is a bijection as desired. This completes the proof. □

Lemma 11 *Lemma 2 holds for our definition of the witness extractors* $(\mathsf{Ext}_0, \mathsf{Ext}_1)$.

Proof Since the proof of $\mathbf{0}$-side and $\mathbf{1}$-side is analogous, we only consider the $\mathbf{0}$-side case. We prove the lemma by contradiction. Suppose the $\mathbf{0}$-side witness can be extracted from the base $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ at index i, but cannot be extracted from either of the sides $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$, $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ or $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$. By Lemma 9, the assumption holds if and only if $\mathbf{c}_0 = \mathbf{c}''_0$ and $\mathbf{c}'_0 = \mathbf{c}''_0$. As a result, $\mathbf{c}_0 = \mathbf{c}'_0$. By Lemma 9, the $\mathbf{0}$-side witness cannot be extracted from $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$. However, this contradicts our assumption. □

Combining everything together, we obtain the following.

Theorem 12 (One-more unforgeability) *The blind signature scheme in Fig. 3 is one-more unforgeable. To be more specific, for all $\ell \in \mathbb{N}$, if there exists an adversary $\mathcal{A}$ that makes Q hash queries to the random oracle and breaks the ℓ-one more unforgeability of* BS *with advantage $\epsilon_{\mathcal{A}} \geq \frac{C_1}{2^n} \cdot \binom{Q}{\ell+1}$, then there exists an algorithm $\mathcal{B}$ that breaks the* GAIP *problem with advantage $\epsilon_{\mathcal{B}} \geq C_2 \cdot \frac{\epsilon_{\mathcal{A}}^2}{\binom{Q}{\ell+1}^2 \cdot (\ell+1)^3}$ for some universal positive constants C_1 and C_2.*

Remark 2 Assuming that GAIP is subexponentially hard—more precisely, assuming that no polynomial-time adversary can solve GAIP with probability better than $2^{-\log^{\omega(1)} n}$, where n is the security parameter—this implies that the blind signature scheme of Fig. 3 is secure in the regime of poly-logarithmically-many concurrent sessions. This is because a polynomial-time adversary makes $Q = n^{O(1)} = 2^{O(\log n)}$ hash queries, and poly-logarithmically-many

$$
\begin{array}{lcl}
\mathsf{P}\text{: } \begin{array}{l}\mathsf{X} = (A_0, A_1, A_2) = ([\mathfrak{g}^{a_0}] * E_0, [\mathfrak{g}^{a_1}] * E_0, [\mathfrak{g}^{a_2}] * E_0) \\ \mathsf{W} = (\delta, a_\delta, a_2) \in \{0,1\} \times \mathbb{Z}_N^2\end{array} & & \mathsf{V}\text{: } \mathsf{X} = (A_0, A_1, A_2) \\
\hline
\begin{array}{l}
\text{For } j \in \{0,1\} \\
\quad (\mathbf{y}_{\delta,j}, \mathbf{y}_{2,j}) \xleftarrow{\$} (\mathbb{Z}_N^n)^2 \\
\quad \mathbf{Y}_{\delta,j} = [\mathfrak{g}^{\mathbf{y}_{\delta,j}}] * E_0 \\
\quad \mathbf{Y}_{2,j} = [\mathfrak{g}^{\mathbf{y}_{2,j}}] * E_0 \\
\quad \mathbf{c}_{[1-\delta+j]_3} \xleftarrow{\$} \{-1,1\}^n \\
\quad \mathbf{r}_{1-\delta,j} \xleftarrow{\$} \mathbb{Z}_N^n \\
\quad \mathbf{Y}_{1-\delta,j} = [\mathfrak{g}^{\mathbf{r}_{1-\delta,j}}] * A_{1-\delta}^{\mathbf{c}_{[1-\delta+j]_3}}
\end{array} & \xrightarrow{(\mathbf{Y}_{k,j})_{\substack{k\in[0:2]\\ j\in\{0,1\}}}} & \\
 & \xleftarrow{\quad\mathbf{c}\quad} & \mathbf{c} \xleftarrow{\$} \{-1,1\}^n \\
\begin{array}{l}
\mathbf{c}_{[3-\delta]_3} = \mathbf{c} \odot \mathbf{c}_{[1-\delta]_3} \odot \mathbf{c}_{[2-\delta]_3} \\
\text{For } j \in \{0,1\} \\
\quad \mathbf{r}_{\delta,j} = \mathbf{y}_{\delta,j} - a_\delta \cdot \mathbf{c}_{[\delta+j]_3} \\
\quad \mathbf{r}_{2,j} = \mathbf{y}_{2,j} - a_2 \cdot \mathbf{c}_{[2+j]_3}
\end{array} & \xrightarrow[(\mathbf{c}_k)_{k\in[0:2]}]{(\mathbf{r}_{k,j})_{\substack{k\in[0:2]\\ j\in\{0,1\}}},} & \begin{array}{l}
\text{Accept if } \mathbf{c} = \mathbf{c}_0 \odot \mathbf{c}_1 \odot \mathbf{c}_2 \\
\wedge\ \forall (k,j) \in [0:2] \times \{0,1\}, \\
\qquad [\mathfrak{g}^{\mathbf{r}_{k,j}}] * A_k^{\mathbf{c}_{[k+j]_3}} = \mathbf{Y}_{k,j}
\end{array}
\end{array}
$$

Fig. 5 The base 2-out-of-3 sigma protocol underlying our partially blind signature scheme. Recall $[0 : 2]$ denotes the set $\{0, 1, 2\}$ and $[x]_3$ is a shorthand for $x \bmod 3$

concurrent sessions means $\ell = \log^{O(1)} n$. The theorem guarantees that the advantage $\epsilon_{\mathcal{A}}$ of a polynomial-time adversary in the one-more unforgeability game satisfies

$$\epsilon_{\mathcal{A}} \leq \binom{Q}{\ell+1}\sqrt{\frac{(\ell+1)^3}{C_2 2^{\log^{\omega(1)} n}}} = 2^{\log^{O(1)} n - \log^{\omega(1)} n} = \mathsf{negl}(n).$$

If we assume only that no polynomial-time adversary can solve GAIP with non-negligible probability—that is, such an adversary's success probability is $2^{-\omega(\log n)}$—then we still have security against a constant number of concurrent sessions, since with $\ell = O(1)$ we have

$$\epsilon_{\mathcal{A}} \leq \binom{Q}{\ell+1}\sqrt{\frac{(\ell+1)^3}{C_2 2^{\omega(\log n)}}} = 2^{O(\log n) - \omega(\log n)} = \mathsf{negl}(n).$$

Proof We define the hard instance generator IG to output a GAIP problem instance. Then, the proof follows from the above Lemmas 10 and by Theorem 3, i.e., the main theorem of Kastner et al. [55]. □

6 Extension to partially blind signatures

In this section, we provide our isogeny-based partially blind signature. We first explain the sigma protocol that underlies our isogeny-based partially blind signature and then show how to compile it into a partially blind signature.

6.1 Base sigma protocol for a 2-out-of-3 relation

We consider a sigma protocol to prove that the prover knows at least *two out of the three* secrets corresponding to the public statement $\mathsf{X} = (A_0, A_1, A_2) = ([\mathfrak{g}^{a_0}] * E_0, [\mathfrak{g}^{a_1}] * E_0, [\mathfrak{g}^{a_2}] * E_0)$. The sigma protocol is depicted in Fig. 5. Since the secret a_2 for A_2 will be known by the signer *and* user in our partially blind signature, we assume the prover always knows the secret a_2 and proves knowledge of one other secret a_0 or a_1 in our sigma protocol.

While these properties are implicit in the partially blind signature, we sketch the properties of our sigma protocol for completeness.

6.1.1 Correctness

Observe that the prover creates six first-flow commitments $(\mathbf{Y}_{k,j})_{(k,j)\in[0:2]\times\{0,1\}}$, where $(\mathbf{Y}_{k,j})_{j\in\{0,1\}}$ is used for the k-th statement A_k, and the challenges associated with $\mathbf{Y}_{k,j}$ are defined as $\mathbf{c}_{[k+j]_3}$. Specifically, we have the correspondence $(\mathbf{Y}_{0,0}, \mathbf{Y}_{0,1}) \mapsto (\mathbf{c}_0, \mathbf{c}_1)$, $(\mathbf{Y}_{1,0}, \mathbf{Y}_{1,1}) \mapsto (\mathbf{c}_1, \mathbf{c}_2)$, and $(\mathbf{Y}_{2,0}, \mathbf{Y}_{2,1}) \mapsto (\mathbf{c}_2, \mathbf{c}_0)$. Correctness then follows from a routine check.

6.1.2 HVZK

Given a challenge $\mathbf{c}$, a zero-knowledge simulator Sim samples random $(\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2) \xleftarrow{\$} (\{-1, 1\}^n)^3$ and $(\mathbf{r}_{k,j})_{(k,j)\in[0:2]\times\{0,1\}} \xleftarrow{\$} \mathbb{Z}_N^6$ conditioned on $\mathbf{c}_0 \odot \mathbf{c}_1 \odot \mathbf{c}_2 = \mathbf{c}$. It then sets $\mathbf{Y}_{k,j} = [\mathfrak{g}^{\mathbf{r}_{k,j}}] * A_k^{\mathbf{c}_{[k+j]_3}}$ for $(k, j) \in [0:2] \times \{0, 1\}$, and outputs the simulated transcript $\big((\mathbf{Y}_{k,j})_{(k,j)\in[0:2]}, \mathbf{c}, ((\mathbf{r}_{k,j}, \mathbf{c}_k)_{k\in[0:2]})_{j\in\{0,1\}}\big)$. Since there is a bijection between $\mathbf{r}_{k,j}$ and $\mathbf{Y}_{k,j}$ once $\mathbf{c}_{[k+j]_3}$ is fixed, this produces a transcript identically distributed as a real transcript.

6.1.3 Witness indistinguishability

This is a direct consequence of the above since perfect HVZK implies perfect witness indistinguishability.

6.1.4 Special soundness

Let $\big((\mathbf{Y}_{k,j})_{(k,j)\in[0:2]}, \mathbf{c}, ((\mathbf{r}_{k,j}, \mathbf{c}_k)_{k\in[0:2]})_{j\in\{0,1\}}\big)$ and $\big((\mathbf{Y}_{k,j})_{(k,j)\in[0:2]}, \mathbf{c}', ((\mathbf{r}'_{k,j}, \mathbf{c}'_k)_{k\in[0:2]})_{j\in\{0,1\}}\big)$ be two valid transcripts such that $\mathbf{c} \neq \mathbf{c}'$. Since $\mathbf{c} \neq \mathbf{c}'$, there exists $k \in [0:2]$ such that $\mathbf{c}_k \neq \mathbf{c}'_k$. Without loss of generality, assume $c_{0,1} \neq c'_{0,1}$, where $c_{0,1}$ and $c'_{0,1} \in \{-1, 1\}$ are the first elements of $\mathbf{c}_0$ and $\mathbf{c}'_0$, respectively. The extractor Ext then given such two valid transcripts outputs a witness $(0, a_0 = \frac{r_{0,0,1}-r'_{0,0,1}}{c_{0,1}-c'_{0,1}}, a_2 = \frac{r_{2,1,1}-r'_{2,1,1}}{c_{0,1}-c'_{0,1}})$, where $(r_{0,0,1}, r'_{0,0,1}, r_{2,1,1}, r'_{2,1,1}) \in \mathbb{Z}_N^4$ are the first elements of $(\mathbf{r}_{0,0}, \mathbf{r}'_{0,0}, \mathbf{r}_{2,1}, \mathbf{r}'_{2,1})$. Let us verify the correctness of such an Ext. Since the two transcripts are valid, we have $[\mathfrak{g}^{r_{0,0,1}}] * A_0^{c_{0,1}} = [\mathfrak{g}^{r'_{0,0,1}}] * A_0^{c'_{0,1}}$ and $[\mathfrak{g}^{r_{2,1,1}}] * A_0^{c_{0,1}} = [\mathfrak{g}^{r'_{2,1,1}}] * A_0^{c'_{0,1}}$. Plugging in $A_0 = [\mathfrak{g}^{a_0}] * E_0$ and $A_2 = [\mathfrak{g}^{a_2}] * E_0$ and following the same argument as in Sect. 5.2, we obtain the desired (a_0, a_2).

6.2 Description of our partially blind signature

We are now able to present our isogeny-based partially blind signature. Let (p, N, E_0) be the public parameters, $[\mathfrak{g}]$ be a generator in $\mathcal{C}\ell(\mathcal{O})$, and $\mathsf{H} : \{0, 1\}^* \to \{-1, 1\}^n$ as defined in Sect. 5. We also require another hash function $\mathsf{G} : \{0, 1\}^* \to \mathbb{Z}_N$ that is modeled as a random oracle. Note that H and G can be implemented by a single random oracle by using domain separation. The following algorithms are summarized in Fig. 6.

PBS.KGen (1^n): On input the security parameter 1^n, it samples a bit $\delta \xleftarrow{\$} \{0, 1\}$, $(a_0, a_1) \xleftarrow{\$} \mathbb{Z}_N^2$ and outputs a public key $\mathsf{pk} = (A_0, A_1) = ([\mathfrak{g}^{a_0}] * E_0, [\mathfrak{g}^{a_1}] * E_0)$ and secret key $\mathsf{sk} = (\delta, a_\delta)$.

```
$\mathsf{PBS.KGen}(1^n)$
101 :  $\delta \xleftarrow{\$} \{0,1\}$
102 :  $(a_0, a_1) \xleftarrow{\$} \mathbb{Z}_N^2$
103 :  $(A_0, A_1) = ([\mathfrak{g}^{a_0}] * E_0, [\mathfrak{g}^{a_1}] * E_0)$
104 :  return $(\mathsf{pk} = (A_0, A_1), \mathsf{sk} = (\delta, a_\delta))$

$\mathsf{PBS.S}_1(\mathsf{sk}, \mathsf{info})$
201 :  parse $(\delta, a_\delta) \leftarrow \mathsf{sk}$
202 :  for $j \in \{0,1\}$
203 :      $(\mathbf{y}^*_{\delta,j}, \mathbf{y}^*_{2,j}) \xleftarrow{\$} (\mathbb{Z}_N^n)^2$
204 :      $\mathbf{Y}^*_{\delta,j} = [\mathfrak{g}^{\mathbf{y}^*_{\delta,j}}] * E_0$
205 :      $\mathbf{Y}^*_{2,j} = [\mathfrak{g}^{\mathbf{y}^*_{2,j}}] * E_0$
206 :      $(\mathbf{c}^*_{[1-\delta+j]_3}, \mathbf{r}^*_{1-\delta,j}) \xleftarrow{\$} \{0,1\}^n \times \mathbb{Z}_N^n$
207 :      $\mathbf{Y}^*_{1-\delta,j} = [\mathfrak{g}^{\mathbf{r}^*_{1-\delta,j}}] * A_{1-\delta}^{\mathbf{c}^*_{[1-\delta+j]_3}}$
208 :  $\mathsf{state}_\mathsf{S} = (\mathbf{y}^*_{\delta,j}, \mathbf{y}^*_{2,j}, \mathbf{c}^*_{1-\delta,j}, \mathbf{r}^*_{1-\delta,j})_{j\in\{0,1\}}$
209 :  $\rho_{\mathsf{S},1} = (\mathbf{Y}^*_{k,j})_{(k,j)\in[0:2]\times\{0,1\}}$
210 :  return $(\mathsf{state}_\mathsf{S}, \rho_{\mathsf{S},1})$

$\mathsf{PBS.U}_1(\mathsf{pk}, \mathsf{info}, \mathsf{M}, \rho_{\mathsf{S},1})$
301 :  parse $(\mathbf{Y}^*_{k,j})_{(k,j)\in[0:2]\times\{0,1\}} \leftarrow \rho_{\mathsf{S},1}$
302 :  for $k \in [0:2]$
303 :      $\mathbf{d}_k \xleftarrow{\$} \{-1,1\}^n$
304 :      for $j \in \{0,1\}$
305 :          $\mathbf{Z}_{k,j} = [\mathfrak{g}^{\mathbf{z}_{k,j}}] * (\mathbf{Y}^*_{k,j})^{\mathbf{d}_{[k+j]_3}}$
306 :  $\mathbf{c} = \mathsf{H}((\mathbf{Z}_{k,j})_{(k,j)\in[0:2]\times\{0,1\}} \| \mathsf{info} \| \mathsf{M})$
307 :  $\mathbf{c}^* = \mathbf{c} \odot \mathbf{d}_0 \odot \mathbf{d}_1 \odot \mathbf{d}_2 \in \{-1,1\}^n$
308 :  $\mathsf{state}_\mathsf{U} = (\mathbf{d}_k, (\mathbf{z}_{k,j})_{j\in\{0,1\}})_{k\in[0:2]}$
309 :  return $(\mathsf{state}_\mathsf{U}, \rho_\mathsf{U} = \mathbf{c}^*)$

$\mathsf{PBS.S}_2(\mathsf{state}_\mathsf{S}, \rho_\mathsf{U})$
401 :  parse $(\mathbf{y}^*_{\delta,j}, \mathbf{y}^*_{2,j}, \mathbf{c}^*_{1-\delta,j}, \mathbf{r}^*_{1-\delta,j})_{j\in\{0,1\}} \leftarrow \mathsf{state}_\mathsf{S}$
402 :  parse $\mathbf{c}^* \leftarrow \rho_\mathsf{U}$
403 :  $a_2 = \mathsf{G}(\mathsf{info})$
404 :  $\mathbf{c}^*_{[3-\delta]_3} = \mathbf{c}^* \odot \mathbf{c}^*_{[1-\delta]_3} \odot \mathbf{c}^*_{[2-\delta]_3} \in \{-1,1\}^n$
405 :  for $j \in \{0,1\}$
406 :      $\mathbf{r}^*_{\delta,j} = \mathbf{y}^*_{\delta,j} - a_\delta \cdot \mathbf{c}^*_{[\delta+j]_3} \in \mathbb{Z}_N^n$
407 :      $\mathbf{r}^*_{2,j} = \mathbf{y}^*_{2,j} - a_2 \cdot \mathbf{c}^*_{[2+j]_3} \in \mathbb{Z}_N^n$
408 :  return $\rho_{\mathsf{S},2} = (\mathbf{c}^*_k, (\mathbf{r}^*_{k,j})_{j\in\{0,1\}})_{k\in[0:2]}$

$\mathsf{PBS.U}_2(\mathsf{state}_\mathsf{U}, \rho_{\mathsf{S},2})$
501 :  parse $(\mathbf{d}_k, (\mathbf{z}_{k,j})_{j\in\{0,1\}})_{k\in[0:2]} \leftarrow \mathsf{state}_\mathsf{U}$
502 :  parse $(\mathbf{c}^*_k, (\mathbf{r}^*_{k,j})_{j\in\{0,1\}})_{k\in[0:2]} \leftarrow \rho_{\mathsf{S},2}$
503 :  $a_2 = \mathsf{G}(\mathsf{info})$
504 :  $A_2 = [\mathfrak{g}^{a_2}] * E_0$
505 :  for $k \in [0:2]$
506 :      $\mathbf{c}_k = \mathbf{c}^*_k \odot \mathbf{d}_k$
507 :      for $j \in \{0,1\}$
508 :          $\mathbf{r}_{k,j} = \mathbf{z}_{k,j} + \mathbf{r}^*_{k,j} \odot \mathbf{d}_{[k+j]_3}$
509 :  $\mathbf{c}' = \mathsf{H}\Big(([\mathfrak{g}^{\mathbf{r}_{k,j}}] * A_k^{\mathbf{c}_{[k+j]_3}})_{(k,j)\in[0:2]\times\{0,1\}} \| \mathsf{info} \| \mathsf{M}\Big).$
510 :  if $\mathbf{c}_0 \odot \mathbf{c}_1 \odot \mathbf{c}_2 = \mathbf{c}'$
511 :      return $\sigma = (\mathbf{c}_k, (\mathbf{r}_{k,j})_{j\in\{0,1\}})_{k\in[0:2]}$
512 :  return $\sigma = \bot$

$\mathsf{PBS.Verify}(\mathsf{pk}, \mathsf{info}, \mathsf{M}, \sigma)$
601 :  parse $(\mathbf{c}_k, (\mathbf{r}_{k,j})_{j\in\{0,1\}})_{k\in[0:2]} \leftarrow \sigma$
602 :  $\mathbf{c}' = \mathsf{H}\Big(([\mathfrak{g}^{\mathbf{r}_{k,j}}] * A_k^{\mathbf{c}_{[k+j]_3}})_{(k,j)\in[0:2]\times\{0,1\}} \| \mathsf{info} \| \mathsf{M}\Big)$
603 :  if $\mathbf{c}_0 \odot \mathbf{c}_1 \odot \mathbf{c}_2 = \mathbf{c}'$
604 :      return 1
605 :  return 0
```

Fig. 6 Our partially blind signature scheme. We assume the algorithms return $\bot$ and terminate if **parse** is not in the correct format. Recall $[0:2]$ denotes the set $\{0, 1, 2\}$ and $[x]_3$ is a shorthand for x mod 3

$\mathsf{PBS.S}_1(\mathsf{sk}, \mathsf{info})$: The signer performs the following for $j \in \{0, 1\}$: It samples $(\mathbf{y}^*_{\delta,j}, \mathbf{y}^*_{2,j}) \xleftarrow{\$} (\mathbb{Z}_N^n)^2$ and sets $(\mathbf{Y}^*_{\delta,j}, \mathbf{Y}^*_{2,j}) = ([\mathfrak{g}^{\mathbf{y}^*_{\delta,j}}] * E_0, [\mathfrak{g}^{\mathbf{y}^*_{2,j}}] * E_0)$. It then samples $(\mathbf{c}^*_{[1-\delta+j]_3}, \mathbf{r}^*_{1-\delta,j}) \xleftarrow{\$} \{-1, 1\}^n \times \mathbb{Z}_N^n$ and sets $\mathbf{Y}^*_{1-\delta,j} = [\mathfrak{g}^{\mathbf{r}^*_{1-\delta,j}}] * A_{1-\delta}^{\mathbf{c}^*_{[1-\delta+j]_3}}$. Finally, it outputs the signer state $\mathsf{state}_\mathsf{S} = (\mathbf{y}^*_{\delta,j}, \mathbf{y}^*_{2,j}, \mathbf{c}^*_{1-\delta,j}, \mathbf{r}^*_{1-\delta,j})_{j\in\{0,1\}}$ and the first-sender message $\rho_{\mathsf{S},1} = (\mathbf{Y}^*_{k,j})_{(k,j)\in[0:2]\times\{0,1\}}$.

$\mathsf{PBS.U}_1(\mathsf{pk}, \mathsf{info}, \mathsf{M}, \rho_{\mathsf{S},1})$: The user parses $(\mathbf{Y}^*_{k,j})_{(k,j)\in[0:2]\times\{0,1\}} \leftarrow \rho_{\mathsf{S},1}$. It then samples $\mathbf{d}_k \xleftarrow{\$} \{-1, 1\}^n$, $\mathbf{z}_{k,j} \xleftarrow{\$} \mathbb{Z}_N^n$, and computes $\mathbf{Z}_{k,j} = [\mathfrak{g}^{\mathbf{z}_{k,j}}] * (\mathbf{Y}^*_{k,j})^{\mathbf{d}_{[k+j]_3}}$ for $(k, j) \in [0 : 2] \times \{0, 1\}$. It then computes $\mathbf{c} = \mathsf{H}\Big((\mathbf{Z}_{k,j})_{(k,j)\in[0:2]\times\{0,1\}} \| \mathsf{info} \| \mathsf{M}\Big) \in \{-1, 1\}^n$ and outputs the user state $\mathsf{state}_\mathsf{U} = (\mathbf{d}_k, (\mathbf{z}_{k,j})_{j\in\{0,1\}})_{k\in[0:2]}$ and user message $\rho_\mathsf{U} = \mathbf{c}^* = \mathbf{c} \odot \mathbf{d}_0 \odot \mathbf{d}_1 \odot \mathbf{d}_2$.

$\mathsf{PBS.S}_2(\mathsf{state}_\mathsf{S}, \rho_\mathsf{U})$: The signer computes $a_2 = \mathsf{G}(\mathsf{info}) \in \mathbb{Z}_N$, parses $(\mathbf{y}^*_{\delta,j}, \mathbf{y}^*_{2,j}, \mathbf{c}^*_{1-\delta,j}, \mathbf{r}^*_{1-\delta,j})_{j\in\{0,1\}} \leftarrow \mathsf{state}_\mathsf{S}$, $\mathbf{c}^* \leftarrow \rho_\mathsf{U}$ and sets $\mathbf{c}^*_{[3-\delta]_3} = \mathbf{c}^* \odot \mathbf{c}^*_{[1-\delta]_3} \odot \mathbf{c}^*_{[2-\delta]_3} \in \{-1, 1\}^n$. It then computes $\mathbf{r}^*_{\delta,j} = \mathbf{y}^*_{\delta,j} - a_\delta \cdot \mathbf{c}^*_{[\delta+j]_3} \in \mathbb{Z}_N^n$ and $\mathbf{r}^*_{2,j} = \mathbf{y}^*_{2,j} - a_2 \cdot \mathbf{c}^*_{[2+j]_3} \in \mathbb{Z}_N^n$ for $j \in \{0, 1\}$. Finally, it outputs the second-signer message $\rho_{\mathsf{S},2} = (\mathbf{c}^*_k, (\mathbf{r}^*_{k,j})_{j\in\{0,1\}})_{k\in[0:2]}$.

$\mathsf{PBS.U}_2(\mathsf{state}_\mathsf{U}, \rho_{\mathsf{S},2})$: The user first computes $a_2 = \mathsf{G}(\mathsf{info}) \in \mathbb{Z}_N$ and sets $A_3 = [\mathfrak{g}^{a_2}] * E_0$. It then parses $(\mathbf{d}_k, (\mathbf{z}_{k,j})_{j\in\{0,1\}})_{k\in[0:2]} \leftarrow \mathsf{state}_\mathsf{U}$, $(\mathbf{c}^*_k, (\mathbf{r}^*_{k,j})_{j\in\{0,1\}})_{k\in[0:2]} \leftarrow \rho_{\mathsf{S},2}$ and sets $\mathbf{c}_k = \mathbf{c}^*_k \odot \mathbf{d}_k$ and $\mathbf{r}_{k,j} = \mathbf{z}_{k,j} + \mathbf{r}^*_{k,j} \odot \mathbf{d}_{[k+j]_3}$ for $(k, j) \in [0:2] \times \{0, 1\}$. It then checks if

$$\mathbf{c}_0 \odot \mathbf{c}_1 \odot \mathbf{c}_2 = \mathsf{H}\Big(([\mathfrak{g}^{\mathbf{r}_{k,j}}] * A_k^{\mathbf{c}_{[k+j]_3}})_{(k,j)\in[0:2]\times\{0,1\}} \| \mathsf{info} \| \mathsf{M}\Big). \tag{4}$$

If it holds, it outputs a signature $\sigma = (\mathbf{c}_k, (\mathbf{r}_{k,j})_{j\in\{0,1\}})_{k\in[0:2]} \in \big(\{-1,1\}^n \times (\mathbb{Z}_N^n)^2\big)^3$, and otherwise a $\perp$.

$\mathsf{PBS.Verify}(\mathsf{pk}, \mathsf{M}, \sigma)$: The verifier outputs 1 if Eq. 4 holds, and otherwise 0.

The correctness, blindness, and one-more unforgeability of our blind signature are provided in the subsequent sections.

6.3 Proof of correctness and blindness

Correctness can be checked by a routine calculation. For completeness, we provide the proof below.

Theorem 13 *The partially blind signature scheme in Fig. 6 is (perfectly) correct.*

Proof To show correctness, it suffices to show that Eq. 4 holds when both the signer and user follow the protocol. First, it can be checked that we have $\mathbf{Y}^*_{k,j} = [\mathfrak{g}^{\mathbf{r}^*_{k,j}}] * A_k^{\mathbf{c}^*_{[k+j]_3}}$ for $(k, j) \in [0:2] \times \{0, 1\}$. The case $k = 1 - \delta$ holds by definition and the other cases hold due to the correctness of the base 2-out-of-3 sigma protocol (see Sect. 6.1). Then, plugging in $\mathbf{c}_k = \mathbf{c}^*_k \odot \mathbf{d}_k$ and $\mathbf{r}_{k,j} = \mathbf{z}_{k,j} + \mathbf{r}^*_{k,j} \odot \mathbf{d}_{[k+j]_3}$ for $(k, j) \in [0:2] \times \{0, 1\}$, we have

$$\begin{aligned}
[\mathfrak{g}^{\mathbf{r}_{k,j}}] * A_k^{\mathbf{c}_{[k+j]_3}} &= [\mathfrak{g}^{\mathbf{z}_{k,j} + \mathbf{r}^*_{k,j} \odot \mathbf{d}_{[k+j]_3}}] * A_k^{\mathbf{c}^*_{[k+j]_3} \odot \mathbf{d}_{[k+j]_3}} \\
&= [\mathfrak{g}^{\mathbf{z}_{k,j}}] * \Big([\mathfrak{g}^{\mathbf{r}^*_{k,j} \odot \mathbf{d}_{[k+j]_3}}] * A_k^{\mathbf{c}^*_{[k+j]_3} \odot \mathbf{d}_{[k+j]_3}}\Big) \\
&= [\mathfrak{g}^{\mathbf{z}_{k,j}}] * (\mathbf{Y}^*_{k,j})^{\mathbf{d}_{[k+j]_3}} = \mathbf{Z}_{k,j}.
\end{aligned} \tag{5}$$

Finally, since $\mathbf{c} = \mathbf{c}^* \odot \mathbf{d}_0 \odot \mathbf{d}_1 \odot \mathbf{d}_2 = \mathbf{c}^*_0 \odot \mathbf{c}^*_1 \odot \mathbf{c}^*_2 \odot \mathbf{d}_0 \odot \mathbf{d}_1 \odot \mathbf{d}_2 = \mathbf{c}_0 \odot \mathbf{c}_1 \odot \mathbf{c}_2$, where $\mathbf{c} = \mathsf{H}((\mathbf{Z}_{k,j})_{(k,j)\in[0:2]\times\{0,1\}} \| \mathsf{info} \| \mathsf{M})$, we obtain Eq. 4 as desired. Note that we use the fact that $x \odot x = 1$ for any $x \in \{-1, 1\}$ in the first equality. □

The proof of blindness is also standard. Since checking A is a valid elliptic curve can be done efficiently and for such valid A, there exists a unique $a \in \mathbb{Z}_N$ such that $[\mathfrak{g}^a] * E_0 = A$, our partially blind signature is secure even against a malicious server outputting an arbitrary public key.

Theorem 14 *The partially blind signature scheme in Fig. 6 is (perfectly) blind under chosen keys.*

Proof It suffices to show that for any valid public key pk, tag info, any first and second-signer messages $\rho_{\mathsf{S},1} = (\mathbf{Y}^*_{k,j})_{(k,j)\in[0:2]\times\{0,1\}}$ and $\rho_{\mathsf{S},2} = (\mathbf{c}^*_k, (\mathbf{r}^*_{k,j})_{j\in\{0,1\}})_{k\in[0:2]} \in (\{-1,1\}^n \times (\mathbb{Z}_N^n)^2)^3$, and valid signature $(\mathbf{c}_k, (\mathbf{r}_{k,j})_{j\in\{0,1\}})_{k\in[0:2]} \in (\{-1,1\}^n \times (\mathbb{Z}_N^n)^2)^3$, there exists a unique and pair-wise distinct user state $\mathsf{state}_\mathsf{U} = (\mathbf{d}_k, (\mathbf{z}_{k,j})_{j\in\{0,1\}})_{k\in[0:2]} \in (\{-1,1\}^n \times (\mathbb{Z}_N^n)^2)^3$ that could have generated σ. In other words, it suffices to show that fixing an arbitrary $(\mathsf{pk}, \mathsf{info}, \rho_{\mathsf{S},1}, \rho_{\mathsf{S},2})$, there exists a bijection between a valid σ and $\mathsf{state}_\mathsf{U}$. Here, note that any public key $\mathsf{pk} = (A_0, A_1)$ output by the adversary (i.e., malicious signer) $\mathcal{A}$ can be efficiently checked to be valid elliptic curves (i.e., supersingularity). Below, we let $(a_0, a_1) \in \mathbb{Z}_N^2$ be the unique secret key $\mathsf{sk} = (a_0, a_1)$ such that $(A_0, A_1) = ([\mathfrak{g}^{a_0}] * E_0, [\mathfrak{g}^{a_1}] * E_0)$ and set $a_2 = \mathsf{G}(\mathsf{info})$ and $A_2 = [\mathfrak{g}^{a_2}] * E_0$.

Let us fix $\mathsf{sk} = (a_0, a_1)$ (hence pk), (a_2, A_2) (hence info), $\rho_{\mathsf{S},1} = (\mathbf{Y}^*_{k,j})_{(k,j)\in[0:2]\times\{0,1\}}$ and $\rho_{\mathsf{S},2} = (\mathbf{c}^*_k, (\mathbf{r}^*_{k,j})_{j\in\{0,1\}})_{k\in[0:2]}$, and a valid signature $\sigma = (\mathbf{c}_k, (\mathbf{r}_{k,j})_{j\in\{0,1\}})_{k\in[0:2]}$. Let us further define the user state $\mathsf{state}_\mathsf{U} = (\mathbf{d}_k, (\mathbf{z}_{k,j})_{j\in\{0,1\}})_{k\in[0:2]}$ as $\mathbf{d}_k = \mathbf{c}_k \odot \mathbf{c}^*_k$ and $\mathbf{z}_{k,j} = \mathbf{r}_{k,j} - \mathbf{r}^*_{k,j} \odot \mathbf{d}_{[k+j]_3}$ for $(k, j) \in [0 : 2] \times \{0, 1\}$. Following Eq. 5 from right to left, we have $\mathbf{Z}_{k,j} = [\mathfrak{g}^{\mathbf{r}_{k,j}}] * A_k^{\mathbf{c}_{[k+j]_3}}$ for $(k, j) \in [0 : 2] \times \{0, 1\}$. Combining this with σ being a valid signature, we have $\mathbf{c}_0 \odot \mathbf{c}_1 \odot \mathbf{c}_2 = \mathsf{H}\Big(([\mathfrak{g}^{\mathbf{r}_{k,j}}] * A_k^{\mathbf{c}_{[k+j]_3}})_{(k,j)\in[0:2]\times\{0,1\}} \| \mathsf{info} \| \mathsf{M}\Big) = \mathsf{H}\Big((\mathbf{Z}_{k,j})_{(k,j)\in[0:2]\times\{0,1\}} \| \mathsf{info} \| \mathsf{M}\Big)$. Therefore, $\mathsf{state}_\mathsf{U}$ is indeed a user state that results in the valid signature σ. Moreover, for any choice of $\rho_{\mathsf{S},2}$ and any $\sigma \neq \sigma'$, it can be checked that the corresponding user states $\mathsf{state}_\mathsf{U}$ and $\mathsf{state}'_\mathsf{U}$ defined as above are distinct. Hence, there is a bijection between a valid signature and a user state. This concludes the proof. $\square$

6.4 Proof of one-more unforgeability

Our proof of OMUF consists of preparing the necessary tools to invoke Theorem 3. Specifically, we define instances (see Definition 12), the map $\Phi_{\mathsf{rand},\vec{h}}$ (see Definition 19), the witness extractors $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ (see Definition 20) and prove that Lemmas 1 and 2 hold. We refer the readers to Sect. 5.5 for some of the notations used below.

6.4.1 Preparation: instances

Let us first define the $\mathbf{0}$-side instance $\mathbf{I}_0$ and the $\mathbf{1}$-side instance $\mathbf{I}_1$. Below, we assume the adversary against the one-more unforgeability game makes ℓ signing queries in total.

A $\mathbf{0}$-side instance $\mathbf{I}_0 = (0, a_0, A_1, \overrightarrow{\mathbf{y}^*_{0,0}}, \overrightarrow{\mathbf{y}^*_{0,1}}, \overrightarrow{\mathbf{c}^*_1}, \overrightarrow{\mathbf{c}^*_2}, \overrightarrow{\mathbf{r}^*_{1,0}}, \overrightarrow{\mathbf{r}^*_{1,1}}, \overrightarrow{\mathbf{y}^*_{2,0}}, \overrightarrow{\mathbf{y}^*_{2,1}})$ is defined as follows:

- $(0, a_0)$: The secret key sk when $\delta = 0$.
- A_1: The part of the public key $\mathsf{pk} = (A_0, A_1)$ whose secret key is unknown.
- $(\mathbf{y}^{*(k)}_{0,0}, \mathbf{y}^{*(k)}_{0,1})$: The exponent of the commitment $(\mathbf{Y}^{*(k)}_{0,0}, \mathbf{Y}^{*(k)}_{0,1})$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 0$ such that $(\mathbf{Y}^{*(k)}_{0,0}, \mathbf{Y}^{*(k)}_{0,1}) = ([\mathfrak{g}^{\mathbf{y}^{*(k)}_{0,0}}] * E_0, [\mathfrak{g}^{\mathbf{y}^{*(k)}_{0,1}}] * E_0)$.
- $(\mathbf{c}^{*(k)}_1, \mathbf{c}^{*(k)}_2)$: The simulated challenge in the k-th ($k \in [\ell]$) first-sender message when $\delta = 0$.
- $(\mathbf{r}^{*(k)}_{1,0}, \mathbf{r}^{*(k)}_{1,1})$: The exponent of the commitment $(\mathbf{Y}^{*(k)}_{1,0}, \mathbf{Y}^{*(k)}_{1,1})$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 0$ such that $(\mathbf{Y}^{*(k)}_{1,0}, \mathbf{Y}^{*(k)}_{1,1}) = ([\mathfrak{g}^{\mathbf{r}^{*(k)}_{1,0}}] * A_1^{\mathbf{c}^{*(k)}_1}, [\mathfrak{g}^{\mathbf{r}^{*(k)}_{1,1}}] * A_1^{\mathbf{c}^{*(k)}_2})$.
- $(\mathbf{y}^{*(k)}_{2,0}, \mathbf{y}^{*(k)}_{2,1})$: The exponent of the commitment $(\mathbf{Y}^{*(k)}_{2,0}, \mathbf{Y}^{*(k)}_{2,1})$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 0$ such that $(\mathbf{Y}^{*(k)}_{2,0}, \mathbf{Y}^{*(k)}_{2,1}) = ([\mathfrak{g}^{\mathbf{y}^{*(k)}_{2,0}}] * E_0, [\mathfrak{g}^{\mathbf{y}^{*(k)}_{2,1}}] * E_0)$..

A $\mathbf{1}$-side instance $\mathbf{I}_1 = (1, a_1, A_0, \overrightarrow{\mathbf{y}^*_{1,0}}, \overrightarrow{\mathbf{y}^*_{1,1}}, \overrightarrow{\mathbf{c}^*_0}, \overrightarrow{\mathbf{c}^*_1}, \overrightarrow{\mathbf{r}^*_{0,0}}, \overrightarrow{\mathbf{r}^*_{0,1}}, \overrightarrow{\mathbf{y}^*_{2,0}}, \overrightarrow{\mathbf{y}^*_{2,1}})$ is defined as follows:

- $(1, a_1)$: The secret key sk when $\delta = 1$.
- A_0: The part of the public key $\mathsf{pk} = (A_0, A_1)$ whose secret key is unknown.

- $(\mathbf{y}^{*(k)}_{1,0}, \mathbf{y}^{*(k)}_{1,1})$: The exponent of the commitment $(\mathbf{Y}^{*(k)}_{1,0}, \mathbf{Y}^{*(k)}_{1,1})$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 1$ such that $(\mathbf{Y}^{*(k)}_{1,0}, \mathbf{Y}^{*(k)}_{1,1}) = ([\mathfrak{g}^{\mathbf{y}^{*(k)}_{1,0}}] * E_0, [\mathfrak{g}^{\mathbf{y}^{*(k)}_{1,1}}] * E_0)$.
- $(\mathbf{c}^{*(k)}_{0}, \mathbf{c}^{*(k)}_{1})$: The simulated challenge in the k-th ($k \in [\ell]$) first-sender message when $\delta = 1$.
- $(\mathbf{r}^{*(k)}_{0,0}, \mathbf{r}^{*(k)}_{0,1})$: The exponent of the commitment $(\mathbf{Y}^{*(k)}_{0,0}, \mathbf{Y}^{*(k)}_{0,1})$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 1$ such that $(\mathbf{Y}^{*(k)}_{0,0}, \mathbf{Y}^{*(k)}_{0,1}) = ([\mathfrak{g}^{\mathbf{r}^{*(k)}_{0,0}}] * A_0^{\mathbf{c}^{*(k)}_0}, [\mathfrak{g}^{\mathbf{r}^{*(k)}_{0,1}}] * A_1^{\mathbf{c}^{*(k)}_1})$.
- $(\mathbf{y}^{*(k)}_{2,0}, \mathbf{y}^{*(k)}_{2,1})$: The exponent of the commitment $(\mathbf{Y}^{*(k)}_{2,0}, \mathbf{Y}^{*(k)}_{2,1})$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 1$ such that $(\mathbf{Y}^{*(k)}_{2,0}, \mathbf{Y}^{*(k)}_{2,1}) = ([\mathfrak{g}^{\mathbf{y}^{*(k)}_{2,0}}] * E_0, [\mathfrak{g}^{\mathbf{y}^{*(k)}_{2,1}}] * E_0)$.

In the above, note that the randomness $(\overrightarrow{\mathbf{y}_{2,0}}, \overrightarrow{\mathbf{y}_{2,1}})$ associated with the tags $\overrightarrow{\mathsf{info}}$ are identical for both instances, and moreover, chosen independently of the tags queried by the adversary. This will be a crucial observation when applying Theorem 3, which focuses on the one-more unforgeability of blind signatures, to the partially blind signature setting.

6.4.2 Preparation: map $\Phi_{\mathsf{rand}, \vec{h}}$

We next define the map $\Phi_{\mathsf{rand}, \vec{h}}$ that maps a $\mathbf{0}$-side instance $\mathbf{I}_0$ into a $\mathbf{1}$-side instance $\mathbf{I}_1$ and vice versa. Concretely, a $\mathbf{0}$-side instance $\mathbf{I}_0 = (0, a_0, A_1, \overrightarrow{\mathbf{y}^*_{0,0}}, \overrightarrow{\mathbf{y}^*_{0,1}}, \overrightarrow{\mathbf{c}^*_1}, \overrightarrow{\mathbf{c}^*_2}, \overrightarrow{\mathbf{r}^*_{1,0}}, \overrightarrow{\mathbf{r}^*_{1,1}}, \overrightarrow{\mathbf{y}^*_{2,0}}, \overrightarrow{\mathbf{y}^*_{2,1}})$, $\Phi_{\mathsf{rand}, \vec{h}}(\mathbf{I}_0)$ maps to a $\mathbf{1}$-side instance $\mathbf{I}_1$ given by

$$\mathbf{I}_1 = \left(\begin{array}{lll} & a_1 \text{ such that } [\mathfrak{g}^{a_1}] * E_0 = A_1, & A_0 = [\mathfrak{g}^{a_0}] * E_0, \\ 1, & \overrightarrow{\mathbf{y}^*_{1,0}} = \overrightarrow{\mathbf{r}^*_{1,0}} + a_1 \cdot \overrightarrow{\mathbf{c}^*_1}, & \overrightarrow{\mathbf{y}^*_{1,1}} = \overrightarrow{\mathbf{r}^*_{1,1}} + a_1 \cdot \overrightarrow{\mathbf{c}^*_2}, \overrightarrow{\mathbf{y}^*_{2,0}}, \overrightarrow{\mathbf{y}^*_{2,1}} \\ & \overrightarrow{\mathbf{c}^*_0} = \overrightarrow{\mathbf{c}^*} \odot \overrightarrow{\mathbf{c}^*_1} \odot \overrightarrow{\mathbf{c}^*_2}, & \overrightarrow{\mathbf{c}^*_1}, \\ & \overrightarrow{\mathbf{r}^*_{0,0}} = \overrightarrow{\mathbf{y}^*_{0,0}} - a_0 \cdot \overrightarrow{\mathbf{c}^*_0}, & \overrightarrow{\mathbf{r}^*_{0,1}} = \overrightarrow{\mathbf{y}^*_{0,1}} - a_0 \cdot \overrightarrow{\mathbf{c}^*_1}, \end{array} \right),$$

where recall that $\overrightarrow{\mathbf{c}^*} = \vec{e}(\mathbf{I}_0, \mathsf{rand}, \vec{h})$.

On the other hand, a $\mathbf{1}$-side instance $\mathbf{I}_1 = (1, a_1, A_0, \overrightarrow{\mathbf{y}^*_{1,0}}, \overrightarrow{\mathbf{y}^*_{1,1}}, \overrightarrow{\mathbf{c}^*_0}, \overrightarrow{\mathbf{c}^*_1}, \overrightarrow{\mathbf{r}^*_{0,0}}, \overrightarrow{\mathbf{r}^*_{0,1}}, \overrightarrow{\mathbf{y}^*_{2,0}}, \overrightarrow{\mathbf{y}^*_{2,1}})$, $\Phi_{\mathsf{rand}, \vec{h}}(\mathbf{I}_1)$ maps to a $\mathbf{0}$-side instance $\mathbf{I}_0$ such that

$$\mathbf{I}_0 = \left(\begin{array}{lll} & a_0 \text{ such that } [\mathfrak{g}^{a_0}] * E_0 = A_0, & A_1 = [\mathfrak{g}^{a_1}] * E_0, \\ 0, & \overrightarrow{\mathbf{y}^*_{0,0}} = \overrightarrow{\mathbf{r}^*_{0,0}} + a_0 \cdot \overrightarrow{\mathbf{c}^*_0}, & \overrightarrow{\mathbf{y}^*_{0,1}} = \overrightarrow{\mathbf{r}^*_{0,1}} + a_0 \cdot \overrightarrow{\mathbf{c}^*_1}, \overrightarrow{\mathbf{y}^*_{2,0}}, \overrightarrow{\mathbf{y}^*_{2,1}} \\ & \overrightarrow{\mathbf{c}^*_1}, & \overrightarrow{\mathbf{c}^*_2} = \overrightarrow{\mathbf{c}^*} \odot \overrightarrow{\mathbf{c}^*_0} \odot \overrightarrow{\mathbf{c}^*_1}, \\ & \overrightarrow{\mathbf{r}^*_{1,0}} = \overrightarrow{\mathbf{y}^*_{1,0}} - a_1 \cdot \overrightarrow{\mathbf{c}^*_1}, & \overrightarrow{\mathbf{r}^*_{1,1}} = \overrightarrow{\mathbf{y}^*_{1,1}} - a_1 \cdot \overrightarrow{\mathbf{c}^*_2}, \end{array} \right),$$

where recall that $\overrightarrow{\mathbf{c}^*} = \vec{e}(\mathbf{I}_1, \mathsf{rand}, \vec{h})$.

6.4.3 Preparation: witness extractors $(\mathsf{Ext}_0, \mathsf{Ext}_1)$

Fix $\mathbf{I}, \mathsf{rand}$ and let $(\vec{h}, \vec{h}') \in \mathrm{F}_i(\mathbf{I}, \mathsf{rand})$ for some $i \in [\ell + 1]$. Let $\sigma = (\mathbf{c}_k, (\mathbf{r}_{k,j})_{j \in \{0,1\}})_{k \in [0:2]}$ and $\sigma' = (\mathbf{c}'_k, (\mathbf{r}'_{k,j})_{j \in \{0,1\}})_{k \in [0:2]}$ be the signatures that correspond to $\mathbf{c}^{(i)}$ and $\mathbf{c}'^{(i)}$, respectively, where $\mathbf{c}^{(i)}$ (resp. $\mathbf{c}'^{(i)}$) is the i-th entry of $\vec{h}$ (resp. $\vec{h}'$). In particular, we have $\mathbf{c}_0 \odot \mathbf{c}_1 \odot \mathbf{c}_2 = \mathbf{c}^{(i)}$ and $\mathbf{c}'_0 \odot \mathbf{c}'_1 \odot \mathbf{c}'_2 = \mathbf{c}'^{(i)}$. We define the witness extractors $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ as in Fig. 7.

The following lemma establishes the correctness of the witness extractors.

```
Ext_0(σ, σ')
101 :  if $\exists t \in [n]$ s.t. $c_{0,t} \neq c'_{0,t}$
102 :      return $a_0 = \frac{r_{0,0,t} - r'_{0,0,t}}{c_{0,t} - c'_{0,t}}$
103 :  elseif $\exists t \in [n]$ s.t. $c_{1,t} \neq c'_{1,t}$
104 :      return $a_0 = \frac{r_{0,1,t} - r'_{0,1,t}}{c_{1,t} - c'_{1,t}}$
105 :  return $\perp$

Ext_1(σ, σ')
101 :  if $\exists t \in [n]$ s.t. $c_{1,t} \neq c'_{1,t}$
102 :      return $a_1 = \frac{r_{1,0,t} - r'_{1,0,t}}{c_{1,t} - c'_{1,t}}$
103 :  elseif $\exists t \in [n]$ s.t. $c_{2,t} \neq c'_{2,t}$
104 :      return $a_1 = \frac{r_{1,1,t} - r'_{1,1,t}}{c_{2,t} - c'_{2,t}}$
105 :  return $\perp$
```

Fig. 7 Witness extractors for our partially blind signature. In the above, $\sigma = (\mathbf{c}_k, (\mathbf{r}_{k,j})_{j\in\{0,1\}})_{k\in[0:2]}$ and $\sigma' = (\mathbf{c}'_k, (\mathbf{r}'_{k,j})_{j\in\{0,1\}})_{k\in[0:2]}$, where $\mathbf{c}_k, \mathbf{c}'_k$ live in $\{-1, 1\}^n$ and $\mathbf{r}_{k,j}, \mathbf{r}'_{k,j}$ live in $\mathbb{Z}_N^n$. Non-bold font indicates the entries of a vector

Lemma 15 $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ *in Fig. 7 satisfy Definition 20.*

Proof By the definition of $\mathrm{F}_i(\mathbf{I}, \mathsf{rand})$ (see Definition 14), we have $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in \mathsf{Succ}$ and $\mathbf{c}^{(i)} \neq \mathbf{c}'^{(i)}$. The former implies that the two signatures σ and σ' are valid. Concretely, we have

$$\mathbf{c}^{(i)} = \mathbf{c}_0 \odot \mathbf{c}_1 \odot \mathbf{c}_2 = \mathsf{H}\Big(([\mathfrak{g}^{\mathbf{r}_{k,j}}] * A_k^{\mathbf{c}_{[k+j]_3}})_{(k,j)\in[0:2]\times\{0,1\}} \| \mathsf{info} \| \mathsf{M}\Big)$$
$$\mathbf{c}'^{(i)} = \mathbf{c}'_0 \odot \mathbf{c}'_1 \odot \mathbf{c}'_2 = \mathsf{H}\Big(([\mathfrak{g}^{\mathbf{r}'_{k,j}}] * A_k^{\mathbf{c}'_{[k+j]_3}})_{(k,j)\in[0:2]\times\{0,1\}} \| \mathsf{info}' \| \mathsf{M}'\Big).$$

Moreover, since $\overrightarrow{h}$ and $\overrightarrow{h}'$ agree up to the i-th entry and the challenger and adversary's randomness are fixed, the input to the hash functions agree. Namely, we have

$$[\mathfrak{g}^{\mathbf{r}_{k,j}}] * A_k^{\mathbf{c}_{[k+j]_3}} = [\mathfrak{g}^{\mathbf{r}'_{k,j}}] * A_k^{\mathbf{c}'_{[k+j]_3}} \text{ for } (k, j) \in [0:2] \times \{0, 1\} \ \wedge\ (\mathsf{info}, \mathsf{M}) = (\mathsf{info}', \mathsf{M}').$$

Due to the special soundness of the underlying sigma protocol (see Sect. 6.1), the witness extractors Ext_0 and Ext_1 each outputs a valid secret key from the $\mathbf{0}$-side and $\mathbf{1}$-side instances, respectively. Moreover, since $\mathbf{c}^{(i)} \neq \mathbf{c}'^{(i)}$, we must have $\mathbf{c}_k \neq \mathbf{c}'_k$ for some $k \in [0:2]$. Thus, at least one of Ext_0 or Ext_1 always outputs a valid secret key; if $\mathbf{c}_1 \neq \mathbf{c}'_1$, then they both output a valid secret key. This completes the proof. □

6.4.4 Proof of one-more unforgeability

We prove the following two lemmas required to invoke the main theorem Theorem 3.

Lemma 16 *Lemma 1 holds for the map* $\Phi_{\mathsf{rand},\overrightarrow{h}}$.

Proof Since the proof for the $\mathbf{0}$-side and $\mathbf{1}$-side instances $\mathbf{I}_0$ and $\mathbf{I}_1$ are analogous, we only consider the $\mathbf{0}$-side instance. For any $\mathsf{rand}, \overrightarrow{h}$, let us consider the query transcript $\overrightarrow{e}(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h}) = \overrightarrow{\mathbf{c}^*}$, i.e., the vector of user message ρ_U queries made by the adversary to the signing algorithm $\mathsf{PBS.S}_2$. Since the underlying sigma protocol is perfectly witness indistinguishable (see Sect. 6.1), for each $i \in [\ell]$ and $\mathbf{c}^{*(i)}$, there is a set of randomness that the signer with a secret key $(1, a_1)$ (i.e., a $\mathbf{1}$-side witness) could have used to produce the same view (i.e., first and second-signer messages) to the adversary. Concretely, this set of randomness is exactly those defined by $\Phi_{\mathsf{rand},\overrightarrow{h}}(\mathbf{I}_0)$. Hence, we have

$\mathsf{trans}(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h}) = \mathsf{trans}(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}_0), \mathsf{rand}, \overrightarrow{h})$ as desired. Moreover, it is easy to check that $\Phi_{\mathsf{rand}, \overrightarrow{h}}(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}_0))$ from the definition of $\Phi_{\mathsf{rand}, \overrightarrow{h}}$. Hence, it is a bijection as desired. This completes the proof. □

Lemma 17 *Lemma 2 holds for the witness extractors* $(\mathsf{Ext}_0, \mathsf{Ext}_1)$.

Proof Since the proof of $\mathbf{0}$-side and $\mathbf{1}$-side witnesses are analogous, we only consider the $\mathbf{0}$-side witness. Suppose the $\mathbf{0}$-side witness can be extracted from base $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ at index i, but cannot be extracted from either of the sides $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$, $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ or $(\mathbf{I}, \mathsf{rand}, \mathsf{H})$, $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$. Due to the description of our witness extractors $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ in Fig. 7, we have $(\mathbf{c}'_0, \mathbf{c}'_1) = (\mathbf{c}''_0, \mathbf{c}''_1)$ and $(\mathbf{c}_0, \mathbf{c}_1) = (\mathbf{c}''_0, \mathbf{c}''_1)$ if the $\mathbf{0}$-side witness cannot be extracted from either of the sides. This implies that $(\mathbf{c}_0, \mathbf{c}_1) = (\mathbf{c}'_0, \mathbf{c}'_1)$. However, this means that Ext_0 fails to extract a $\mathbf{0}$-side witness, thus contradicting our assumption. This completes the proof. □

Combining everything together, we obtain the following.

Theorem 18 (One-more unforgeability) *The partially blind signature scheme in Fig. 6 is one-more unforgeable. More precisely, for all* $\ell \in \mathbb{N}$, *if there exists an adversary* $\mathcal{A}$ *that makes* Q *hash queries to the random oracle and breaks the* ℓ*-one more unforgeability of our* PBS *with advantage* $\epsilon_{\mathcal{A}} \geq \frac{C_1}{2^n} \cdot \binom{Q}{\ell+1}$, *then there exists an algorithm* $\mathcal{B}$ *that breaks the* GAIP *problem with advantage* $\epsilon_{\mathcal{B}} \geq C_2 \cdot \frac{\epsilon_{\mathcal{A}}^2}{\binom{Q}{\ell+1}^2 \cdot (\ell+1)^3}$ *for some universal positive constants* C_1 *and* C_2.

Proof We define the hard instance generator IG to output a GAIP instance. Then, the proof follows from the above Lemmas 1 and 2 and by invoking Theorem 3, i.e., the main theorem of Kastner, Loss, and Xu [55]. To be precise, [55, Theorem 1] is for blind signatures and not the partially blind variant—however, it can be checked that the same proof applies to our partially blind signature by observing that our definition of $\mathbf{0}$-side and $\mathbf{1}$-side instances are defined *independently* of the tags $\overrightarrow{\mathsf{info}}$ used by the adversary, where note that $\overrightarrow{\mathsf{info}}$ is implicitly defined by $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$. In particular, the probability that the reduction extracts the correct witness (i.e., the witness not used by the reduction), can be bounded following the same argument as [55, Theorem 1]. □

Remark 3 (Comparing to the Abe-Okamoto partially blind signature) We note that the reason why the same argument does not hold for the Abe-Okamoto partially blind signature [4] is that the tag info is explicitly required to define the instances. In more detail, the Abe-Okamoto partially blind signature only has one secret key $a_0 \in \mathbb{Z}_p$ attached to the verification key $h_0 = g^{a_0} \in \mathbb{G}$. To sign with respect to a tag info, the signer hashes info to a group element h_{info} and then performs an OR proof that it knows a secret key to either h_0 or h_{info}. In the security proof, the reduction hashes info to a group element $h_{\mathsf{info}} = g^{a_{\mathsf{info}}}$ while knowing the exponent a_{info}. In case the adversary is restricted to use only one tag info, the proof can define the $\mathbf{0}$-side and $\mathbf{1}$-side instances by using a_0 and a_{info}, respectively, and in particular independently of the adversary's randomness. However, when there is more than one tag, we can no longer define a well-defined $\mathbf{1}$-side instance. This is why Kastner, Loss and Xu and Abe and Okamoto first prove the single-tag setting and then prove the multi-tag setting by guessing which tag info the adversary forges on.

7 Optimization using higher degree roots of unity

We investigate the possibility of reducing the signature size by exploiting the $\mathbb{Z}$-module structure of the ideal class group. In this section, we present a generalized construction of the blind signature presented in Sect. 5 based on a new assumption, the *ring group action inverse problem* (rGAIP), which is a generalized version of the group action inverse problem (GAIP).

In Sects. 7.4 to 7.6, we provide the proofs of the correctness, blindness, and OMUF of the construction under the assumption that rGAIP is hard and discuss the applicability of the partialness technique given in Sect. 6. In Sect. 8, we provide analysis on the hardness of the rGAIP for the CSIDH-512 parameter set and show that not all rGAIP instances are equally difficult.

7.1 Overview and preparation

7.1.1 Notations

We summarize some notations unique to this section. We use $\mathbb{Z}_d$ to denote the set $\{0, \ldots, d-1\}$. Moreover, any vector is indexed from 0, e.g., $\mathbf{a} \in \mathbb{Z}_d^\kappa$ is expressed as $(a_0, \ldots, a_{\kappa-1})$. With an overload of notations, for any integer j, we define the bold font $\mathbf{j}$ as the length-κ vector $(j, \ldots, j)$. For any positive integer d and $a \in \mathbb{Z}$ or $\mathbb{Z}_d$, we use $[a]_d$ to denote $(a \bmod d) \in \mathbb{Z}_d$. For the simplicity of the notations, we use the exponent of $\langle\zeta\rangle$ to represent the challenge space of a sigma protocol with an understanding that $\langle\zeta\rangle$ is the d-th primitive root of unity. That is, we will draw a challenge c from $\mathbb{Z}_d$. The operation between the challenges is thereby the addition $c_0 + c_1$, corresponding to the multiplication of $\zeta^{c_0+c_1} = \zeta^{c_0}\zeta^{c_1}$.

7.1.2 Overview

It is a natural attempt to reduce the signature size by considering a larger public key space. Indeed, as shown in [12, Sect. 5.1], such an optimization is possible for standard signature schemes by relaxing GAIP to the multi-target GAIP. As a result, the soundness error of the underlying sigma protocol in a single round decreases to $\frac{1}{2S-1}$ from $\frac{1}{3}$ for a public key size S. Since the number of repetitions is decreased to $\frac{n}{\log_2(2S-1)}$, this technique makes it possible to decrease the signature size, signing, or verification time at the cost of increased public key size. For isogeny-based protocols—which are generally slow but offer small key sizes—this is a very favorable tradeoff.

Unfortunately, a natural adaptation of the same relaxation will not apply to our case because the multi-target GAIP does not offer the particular structure that our blind signature requires. Roughly speaking, a main component of our blind signature requires a user/verifier to compute $[\mathfrak{g}^{z+y^*d}] * E_0$ while only given $[\mathfrak{g}^{y^*}] * E_0 \in \mathcal{E\ell\ell}$, $z \in \mathbb{Z}_N$ and d. This is only feasible by using the quadratic twist which is when $d \in \{-1, 1\}$. An unstructured random public key not only fails to benefit the user but also breaches the group structure of the challenge space since d is no longer restricted in $\{-1, 1\}$.

To this end, we present a novel technique that allows us to trade off between efficiency and the signature size using a structured public key. The high-level idea is fairly simple: to generalize the concept of the quadratic twist in the sense of the group action relation. In the previous section, both parties compute the action of $[\mathfrak{g}^r]$ on a curve E_0 or E_0^{-1} with respect to the challenge $c \in \mathbb{Z}_3^\times = \{-1, 1\}$. Recall that $([\mathfrak{g}^r] * E_0)^{-1} = [\mathfrak{g}^{-r}] * E_0$. In other words, the challenge $c \in \mathbb{Z}_3^\times = \{-1, 1\}$ is encoded into $\mathfrak{g}^c$. Since -1 is a second primitive root of unity over $\mathbb{Z}_N$, the challenge space, as a (multiplicative) group, induces an action on $\mathcal{E}\ell\ell$ by computing the twist.

We generalize the concept by expanding the challenge space to $\langle\zeta\rangle = \{1, \zeta, \zeta^2, \ldots, \zeta^{d-1}\}$, where $d \in \mathbb{N}$ and ζ, a d-th primitive root of unity over $\mathbb{Z}_N^\times$; that is, ζ satisfies $\zeta^d = 1$ and $\zeta^j \neq 1$ for any $j \in [d-1]$. Note that $\langle\zeta\rangle$ is naturally a multiplicative (sub)group, which offers the operation over the challenge space. The action $(r, c) \in \mathbb{Z}_N \times \mathbb{Z}_d$ on a curve $E_0 \in \mathcal{E}\ell\ell$ is defined to be $[\mathfrak{g}^{r\zeta^c}] * E_0$. When $k = 2$ and ζ can be taken to be -1, this is identical to the scheme in the previous section. However, unlike the case $d = 2$ where we have the formula derived from the quadratic twist, when $d \geq 3$ the signer is required to compute $[\mathfrak{g}^{y^*_{b,j}\zeta}] * E_0$ for each $(b, j) \in [2] \times [\kappa]$ in $\mathsf{BS.S}_1$ to aid the user's computation.

7.1.3 Preparation

Our construction requires one more property from the d-th primitive root of unity ζ to be useful.

Looking ahead, when we construct a sigma protocol for the rGAIP relation, the special soundness extractor must solve for the secret exponent $a \in \mathbb{Z}_N$, given $c_1, c_2 \in \mathbb{Z}_N^2$ and $r_1 = y + a\zeta^{c_1}, r_2 = y + a\zeta^{c_2} \pmod N$ for an unknown a and y. If $\mathbb{Z}_N$ is a finite field, then this is trivial. However, in general when $\mathbb{Z}_N$ is a ring, such a may not be efficiently computable. One sufficient condition would be to only use a $d \in \mathbb{Z}_N$ such that $(\zeta^{c_1} - \zeta^{c_2})$ is invertible over $\mathbb{Z}_N$ for all distinct $(c_1, c_2) \in \mathbb{Z}_N^2$. However, this is an overly restrictive requirement and we thus make the following relaxed requirement.

Requirement 1 *We require* $\eta_d = \mathsf{lcm}_{i\in[d-1]}(\mathsf{gcd}(\zeta^i - 1, N)) = \mathsf{poly}$.

The requirement is equivalent to finding a d which divides the totient of many maximal prime power divisors of the class number (see Sect. 8.1 about the existence of, and a method for finding, such a root). Informally, when η_d is polynomial in the security parameter n, then we can brute force all $a \in \mathbb{Z}_N$ such that $a \cdot (\zeta^{c_1} - \zeta^{c_2}) = z$ for a given $(c_1, c_2, z) \in \mathbb{Z}_N^3$. Formally, we have the following.

Lemma 19 *Let (N, d, ζ) be a public parameter where the factorization of N is known and let $\eta_d = \mathsf{lcm}_{i\in[d-1]}(\mathsf{gcd}(\zeta^i - 1, N))$. Then, there exists an extractor Ext' that takes as input the public parameter and $(r_1, r_2, c_1, c_2) \in \mathbb{Z}_N^2 \times \mathbb{Z}_d^2$ where c_1, c_2 are distinct with relations $r_1 = y + a\zeta_d^{c_1}, r_2 = y + a\zeta_d^{c_2} \pmod N$, and outputs a list containing $a \in \mathbb{Z}_N$ of size not greater than η_d in time $\mathsf{poly}(\eta_d)$.*

Proof By calculating $(r_1 - r_2)\zeta_d^{-c_2} = a(\zeta_d^{c_1-c_2} - 1)$, the extractor solves for a by solving the linear equation lifted to the prime power factor of N, then using the Chinese remainder theorem to obtain a list of candidates of a. The size of the list is the number of solutions for the linear equation, which is at most η_d. □

7.2 Base sigma protocol with a large challenge space

We first introduce the base sigma protocol with a larger challenge space assuming Requirement 1. This is depicted in Fig. 8 with the boxed components omitted.

We will show the correctness, HVZK, and, importantly, special soundness of this sigma protocol.

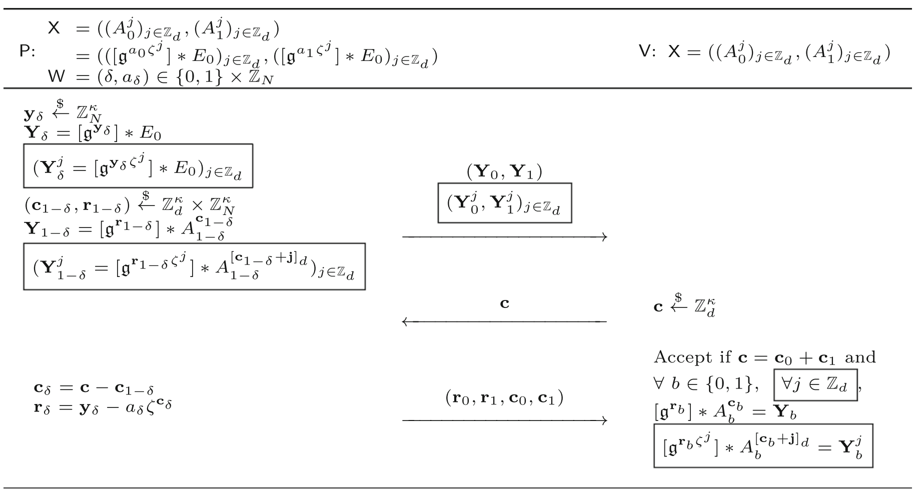
P: $\mathsf{X} = ((A_0^j)_{j\in\mathbb{Z}_d}, (A_1^j)_{j\in\mathbb{Z}_d}) = (([\mathfrak{g}^{a_0\zeta^j}] * E_0)_{j\in\mathbb{Z}_d}, ([\mathfrak{g}^{a_1\zeta^j}] * E_0)_{j\in\mathbb{Z}_d})$
$\mathsf{W} = (\delta, a_\delta) \in \{0,1\} \times \mathbb{Z}_N$
V: $\mathsf{X} = ((A_0^j)_{j\in\mathbb{Z}_d}, (A_1^j)_{j\in\mathbb{Z}_d})$
$\mathbf{y}_\delta \xleftarrow{\$} \mathbb{Z}_N^\kappa$
$\mathbf{Y}_\delta = [\mathfrak{g}^{\mathbf{y}_\delta}] * E_0$
$\boxed{(\mathbf{Y}_\delta^j = [\mathfrak{g}^{\mathbf{y}_\delta\zeta^j}] * E_0)_{j\in\mathbb{Z}_d}}$
$(\mathbf{c}_{1-\delta}, \mathbf{r}_{1-\delta}) \xleftarrow{\$} \mathbb{Z}_d^\kappa \times \mathbb{Z}_N^\kappa$
$\mathbf{Y}_{1-\delta} = [\mathfrak{g}^{\mathbf{r}_{1-\delta}}] * A_{1-\delta}^{\mathbf{c}_{1-\delta}}$
$\boxed{(\mathbf{Y}_{1-\delta}^j = [\mathfrak{g}^{\mathbf{r}_{1-\delta}\zeta^j}] * A_{1-\delta}^{[\mathbf{c}_{1-\delta}+\mathbf{j}]_d})_{j\in\mathbb{Z}_d}}$
$(\mathbf{Y}_0, \mathbf{Y}_1)$
$\boxed{(\mathbf{Y}_0^j, \mathbf{Y}_1^j)_{j\in\mathbb{Z}_d}}$
$\mathbf{c}$
$\mathbf{c} \xleftarrow{\$} \mathbb{Z}_d^\kappa$
$\mathbf{c}_\delta = \mathbf{c} - \mathbf{c}_{1-\delta}$
$\mathbf{r}_\delta = \mathbf{y}_\delta - a_\delta\zeta^{\mathbf{c}_\delta}$
$(\mathbf{r}_0, \mathbf{r}_1, \mathbf{c}_0, \mathbf{c}_1)$
Accept if $\mathbf{c} = \mathbf{c}_0 + \mathbf{c}_1$ and
$\forall\, b \in \{0,1\}$, $\boxed{\forall j \in \mathbb{Z}_d}$,
$[\mathfrak{g}^{\mathbf{r}_b}] * A_b^{\mathbf{c}_b} = \mathbf{Y}_b$
$\boxed{[\mathfrak{g}^{\mathbf{r}_b\zeta^j}] * A_b^{[\mathbf{c}_b+\mathbf{j}]_d} = \mathbf{Y}_b^j}$

Fig. 8 The base sigma protocol with a large challenge space, where the box is to be ignored. Recall $\mathbb{Z}_d = \{0, 1, \ldots, d-1\}$. A_b^j denotes $[\mathfrak{g}^{a_b\zeta^j}] * E_0$ for $j \in \mathbb{Z}_d$ and the vector $A_b^{[\mathbf{c}]_d}$ denotes $(A_b^{[c_0]_d}, \ldots, A_b^{[c_{\kappa-1}]_d})$ where $\mathbf{c} = (c_0, \ldots, c_{\kappa-1}) \in \mathbb{Z}^\kappa$. If $\mathbf{c} \in \mathbb{Z}_d^\kappa$, then $A_b^{[\mathbf{c}]_d}$ is simply $A_b^{\mathbf{c}}$. Other notations are explained in the paragraph above Sect. 7.2. The base sigma protocol can be made compatible with blind signatures by running the boxed lines instead of the preceding non-boxed lines

7.2.1 Correctness

It suffices to show the equation.

$$[\mathfrak{g}^{\mathbf{r}_b}] * A_b^{\mathbf{c}_b} = \mathbf{Y}_b \tag{6}$$

for $b \in \{0, 1\}$. For the case $b = 1-\delta$, the equation holds naturally. For the case $b = \delta$, we have

$$\begin{aligned}[\mathfrak{g}^{\mathbf{r}_\delta}] * A_\delta^{\mathbf{c}_\delta} &= [\mathfrak{g}^{\mathbf{y}_\delta - a_\delta\zeta^{\mathbf{c}_\delta}}] * A_\delta^{\mathbf{c}_\delta} \\ &= [\mathfrak{g}^{\mathbf{y}_\delta - a_\delta\zeta^{\mathbf{c}_\delta}}] * \left([\mathfrak{g}^{a_\delta\zeta^{\mathbf{c}_\delta}}] * E_0\right) \\ &= \mathbf{Y}_\delta,\end{aligned}$$

where we use the fact that $A_\delta^c = [\mathfrak{g}^{a_\delta\zeta^c}] * E_0$ for any $c \in \mathbb{Z}_d$.

7.2.2 HVZK

Given a challenge $\mathbf{c} \in \mathbb{Z}_d^\kappa$, a zero-knowledge simulator Sim samples random $(\mathbf{c}_0, \mathbf{c}_1) \xleftarrow{\$} \mathbb{Z}_d^\kappa$ conditioned on $\mathbf{c}_0 + \mathbf{c}_1 = \mathbf{c}$. Then, for each $b \in \{0, 1\}$, the simulator generates $\mathbf{r}_b \xleftarrow{\$} \mathbb{Z}_N^\kappa$ and $\mathbf{Y}_b = [\mathfrak{g}^{\mathbf{r}_b}] * A_b^{\mathbf{c}_b}$, and outputs $((\mathbf{Y}_0, \mathbf{Y}_1), \mathbf{c}, (\mathbf{r}_0, \mathbf{r}_1, \mathbf{c}_0, \mathbf{c}_1))$.

Since there is a bijection between $\mathbf{r}_b$ and $\mathbf{Y}_b$ once $\mathbf{c}_b$ is fixed, this produces a transcript identically distributed as a real transcript.

7.2.3 Witness indistinguishability

This is a direct consequence of the above since perfect HVZK implies perfect witness indistinguishability.

7.2.4 Special soundness

It suffices to show that special soundness holds in the case that $\kappa = 1$. Let $((Y_0, Y_1), c, (r_0, r_1, c_0, c_1))$, and $((Y_0, Y_1), c', (r'_0, r'_1, c'_0, c'_1))$ be two valid transcripts. Since $c = c_0 + c_1$, $c = c'_0 + c'_1$ and $c \neq c'$, we assume $c_0 \neq c'_0$ without loss of generality. We have $r_0, r'_0 \in \mathbb{Z}_N$, and distinct $c_0, c'_0 \in \mathbb{Z}_d$ which satisfy $r_0 = y + a_0\zeta_d^{c_0}$, $r'_0 = y + a_0\zeta_d^{c'_0} \pmod N$ where y, a_0 are unknown. Since we assume Requirement 1 holds, we can use the extractor $\mathsf{Ext}'(r_0, r'_0, c_0, c'_0)$ in Lemma 19 to obtain a list of size $\eta = \mathsf{lcm}_{i\in[d-1]}(\mathsf{gcd}(\zeta^i - 1, N)) = \mathsf{poly}$ containing $a_0 \in \mathbb{Z}_N$ in polynomial time. We can find a_0 from the list by running through each element in the list and checking if it maps to the statement $(A_0^j)_{j\in\mathbb{Z}_d}$ or $(A_1^j)_{j\in\mathbb{Z}_d}$. Here, we implicitly assume the statement is honestly generated and that this check always terminates.

Before explaining our blind signature, we make a subtle but important modification to our base sigma protocol depicted in Fig. 8 with the boxes. As explained in the introduction, this modification is required since the user of the blind signature is required to randomize $\mathbf{Y}_b = [\mathfrak{g}^{\mathbf{y}_b}] * E_0$ for $b \in \{0, 1\}$ to $[\mathfrak{g}^{\mathbf{z}_b}] * \left([\mathfrak{g}^{\mathbf{y}_b\zeta^{\mathbf{d}_b}}] * E_0\right)$, where $(\mathbf{z}_b, \mathbf{d}_b) \xleftarrow{\$} \mathbb{Z}_N^\kappa \times \mathbb{Z}_d^\kappa$, which is no longer possible when $d \geq 3$. We will give the details of this construction in the following subsection. This extra components also play a key role when proving blindness with malicious keys.

7.3 Enhancing the base sigma protocol for blind signatures

Before explaining our blind signature, we make a subtle but important modification to our base sigma protocol. To understand this modification, notice that if we tried to use a similar idea as in the prior sections to blind $\mathbf{Y}_b = [\mathfrak{g}^{\mathbf{y}_b}] * E_0$ for $b \in \{0, 1\}$, the user must randomize it to a value $[\mathfrak{g}^{\mathbf{z}_b}] * \left([\mathfrak{g}^{\mathbf{y}_b\zeta^{\mathbf{d}_b}}] * E_0\right)$, where $(\mathbf{z}_b, \mathbf{d}_b) \xleftarrow{\$} \mathbb{Z}_N^\kappa \times \mathbb{Z}_d^\kappa$. This was doable when $d = 2$, since $\zeta = -1$ and $[\mathfrak{g}^{\mathbf{y}_b\zeta^{\mathbf{d}_b}}] * E_0$ is simply the quadratic twist of $\mathbf{Y}_b$. However, in general, such a computation cannot be performed. To this end, we let the prover include components that will later help the user in the blind signature. This extension to our basic sigma protocol is illustrated in Fig. 8, where the box represents the modification. The prover sends $[\mathfrak{g}^{\mathbf{y}_b\zeta^j}] * E_0$ for all $j \in \mathbb{Z}_d$ so that the user in the blind signature can choose whichever one based on the $\mathbf{d}_d$ it samples. We also modify the verifier of the base sigma protocol to check that $[\mathfrak{g}^{\mathbf{y}_b\zeta^{\mathbf{d}_b}}] * E_0$ were generated correctly. Below, we show that the extended sigma protocol satisfies correctness and HVZK. Since the extended sigma protocol includes the transcript of the base sigma protocol, special soundness is inherited.

7.3.1 Correctness

It suffices to show that

$$[\mathfrak{g}^{\mathbf{r}_b\zeta^j}] * A_b^{[\mathbf{c}_b+\mathbf{j}]_d} = \mathbf{Y}_b^j$$

for any $(b, j) \in \{0, 1\} \times \mathbb{Z}_d$. For the case $b = 1 - \delta$, the equation holds by definition. For the case $b = \delta$, we have

$$\begin{aligned}
[\mathfrak{g}^{\mathbf{r}_\delta\zeta^j}] * A_\delta^{[\mathbf{c}_\delta+\mathbf{j}]_d} &= [\mathfrak{g}^{\mathbf{y}_\delta\zeta^j - a_\delta\zeta^{\mathbf{c}_\delta+\mathbf{j}}}] * A_\delta^{[\mathbf{c}_\delta+\mathbf{j}]_d} \\
&= [\mathfrak{g}^{\mathbf{y}_\delta\zeta^j - a_\delta\zeta^{\mathbf{c}_\delta+\mathbf{j}}}] * \left([\mathfrak{g}^{a_\delta\zeta^{\mathbf{c}_\delta+\mathbf{j}}}] * E_0\right) \\
&= \mathbf{Y}_\delta^j,
\end{aligned}$$

where we use the fact that $A_\delta^{[c]_d} = [\mathfrak{g}^{a_\delta \zeta^c}] * E_0$ for any $c \in \mathbb{Z}$.

7.3.2 HVZK

Given a challenge $\mathbf{c} \in \mathbb{Z}_d^\kappa$, a zero-knowledge simulator Sim samples random $(\mathbf{c}_0, \mathbf{c}_1) \xleftarrow{\$} \mathbb{Z}_d^\kappa$ conditioned on $\mathbf{c}_0 + \mathbf{c}_1 = \mathbf{c}$. Then, for each $(b, j) \in \{0,1\} \times \mathbb{Z}_d$, the simulator generates $\mathbf{r}_b \xleftarrow{\$} \mathbb{Z}_N^\kappa$ and $\mathbf{Y}_b^j = [\mathfrak{g}^{\mathbf{r}_b \zeta^j}] * A_b^{[\mathbf{c}_b + \mathbf{j}]_d}$, and outputs $((\mathbf{Y}_0^j, \mathbf{Y}_1^j)_{j \in \mathbb{Z}_d}, \mathbf{c}, (\mathbf{r}_0, \mathbf{r}_1, \mathbf{c}_0, \mathbf{c}_1))$ Since for every $j \in \mathbb{Z}_d$, there is a bijection between $\mathbf{r}_b$ and $\mathbf{Y}_b^j$ once $\mathbf{c}_b$ is fixed, this produces a transcript identically distributed as a real transcript.

7.4 Description of our optimized blind signature

We present our optimized isogeny-based blind signature building upon of the enhanced base sigma protocol in Sect. 7.2. Let (p, N, E_0) be the public parameter and $\mathfrak{g}$ be a generator of the ideal class group $\mathcal{C}\ell(\mathcal{O})$ as in Sect. 5. Let ζ to be a d-th root of unity. We assume these parameters are provided to all algorithms. The parameter $\kappa \in \mathbb{N}$ indicates the number of repetition of the underlying sigma protocol such that $d^\kappa \geq 2^n$. Let $\mathsf{H} : \{0,1\}^* \to \mathbb{Z}_d^\kappa$ be a hash function modeled as a random oracle. The following algorithms are summarized in Fig. 9.

$\mathsf{BS.KGen}(1^n)$: On input the security parameter 1^n, it samples a bit $\delta \xleftarrow{\$} \{0,1\}$, $(a_0, a_1) \xleftarrow{\$} \mathbb{Z}_N^2$, and outputs a public key $\mathsf{pk} = ((A_0^j)_{j \in \mathbb{Z}_d}, (A_1^j)_{j \in \mathbb{Z}_d})$ where $A_b^j = [\mathfrak{g}^{a_b \zeta^j}] * E_0$ for $(b, j) \in \{0,1\} \times \mathbb{Z}_d$, and secret key $\mathsf{sk} = (\delta, a_\delta)$.

$\mathsf{BS.S}_1(\mathsf{sk})$: The signer first samples $\mathbf{y}_\delta^* \xleftarrow{\$} \mathbb{Z}_N^\kappa$ and sets $\mathbf{Y}_\delta^{j*} = [\mathfrak{g}^{\mathbf{y}_\delta^* \zeta^j}] * E_0$ for $j \in \mathbb{Z}_d$. It then samples $(\mathbf{c}_{1-\delta}^*, \mathbf{r}_{1-\delta}^*) \xleftarrow{\$} \mathbb{Z}_d^\kappa \times \mathbb{Z}_N^\kappa$ and sets $\mathbf{Y}_{1-\delta}^{j*} = [\mathfrak{g}^{\mathbf{r}_{1-\delta}^* \zeta^j}] * A_{1-\delta}^{\mathbf{c}_{1-\delta}^* + \mathbf{j}}$ for $j \in \mathbb{Z}_d$. It then outputs the signer state $\mathsf{state}_\mathsf{S} = (\mathbf{y}_\delta^*, \mathbf{c}_{1-\delta}^*, \mathbf{r}_{1-\delta}^*)$ and the first-sender message $\rho_{\mathsf{S},1} = (\mathbf{Y}_0^{j*}, \mathbf{Y}_1^{j*})_{j \in \mathbb{Z}_d}$.

$\mathsf{BS.U}_1(\mathsf{pk}, \mathsf{M}, \rho_{\mathsf{S},1})$: The user parses $(\mathbf{Y}_0^{j*}, \mathbf{Y}_1^{j*})_{j \in \mathbb{Z}_d} \leftarrow \rho_{\mathsf{S},1}$, samples $(\mathbf{d}_b, \mathbf{z}_b) \xleftarrow{\$} \mathbb{Z}_d^\kappa \times \mathbb{Z}_N^\kappa$, and computes $\mathbf{Z}_b = [\mathfrak{g}^{\mathbf{z}_b}] * \left(Y_{b,0}^{d_{b,0}*}, \ldots, Y_{b,\kappa-1}^{d_{b,\kappa-1}*}\right)$ for $b \in \{0,1\}$. Here, note that $Y_{b,j}^{d_{b,j}*}$ denotes the j-th ($j \in \mathbb{Z}_d$) element of $\mathbf{Y}_b^{d_{b,j}*} \in \mathcal{E}\ell\ell^\kappa$ and $d_{b,j}$ is the j-th element of $\mathbf{d}_b \in \mathbb{Z}_d^\kappa$. It then computes $\mathbf{c} = \mathsf{H}(\mathbf{Z}_0 \| \mathbf{Z}_1 \| \mathsf{M}) \in \mathbb{Z}_d^\kappa$ and outputs the user state $\mathsf{state}_\mathsf{U} = (\mathbf{d}_0, \mathbf{d}_1, \mathbf{z}_0, \mathbf{z}_1)$ and user message $\rho_\mathsf{U} = \mathbf{c}^* = \mathbf{c} - \mathbf{d}_0 - \mathbf{d}_1$.

$\mathsf{BS.S}_2(\mathsf{state}_\mathsf{S}, \rho_\mathsf{U})$: The signer parses $(\mathbf{y}_\delta^*, \mathbf{c}_{1-\delta}^*, \mathbf{r}_{1-\delta}^*) \leftarrow \mathsf{state}_\mathsf{S}$, $\mathbf{c}^* \leftarrow \rho_\mathsf{U}$, sets $\mathbf{c}_\delta^* = \mathbf{c}^* + \mathbf{c}_{1-\delta}^* \in \mathbb{Z}_d^\kappa$, and computes $\mathbf{r}_\delta^* = \mathbf{y}_\delta^* - a_\delta \zeta^{\mathbf{c}_\delta^*} \in \mathbb{Z}_N^\kappa$. It then outputs the second-signer message $\rho_{\mathsf{S},2} = (\mathbf{c}_0^*, \mathbf{c}_1^*, \mathbf{r}_0^*, \mathbf{r}_1^*)$.

$\mathsf{BS.U}_2(\mathsf{state}_\mathsf{U}, \rho_{\mathsf{S},2})$: The user parses $(\mathbf{d}_0, \mathbf{d}_1, \mathbf{z}_0, \mathbf{z}_1) \leftarrow \mathsf{state}_\mathsf{U}$, $(\mathbf{c}_0^*, \mathbf{c}_1^*, \mathbf{r}_0^*, \mathbf{r}_1^*) \leftarrow \rho_{\mathsf{S},2}$ and checks if $[\mathfrak{g}^{\mathbf{r}_b^* \zeta^j}] * A_b^{[\mathbf{c}_b^* + \mathbf{j}]_d} = \mathbf{Y}_b^{j*}$ holds for all $(b, j) \in \{0,1\} \times \mathbb{Z}_d$. If not, it outputs $\perp$. Otherwise, it sets $(\mathbf{c}_b, \mathbf{r}_b) = (\mathbf{c}_b^* + \mathbf{d}_b, \mathbf{z}_b + \mathbf{r}_b^* \zeta^{\mathbf{d}_b}) \in \mathbb{Z}_d^\kappa \times \mathbb{Z}_N^\kappa$ for $b \in \{0,1\}$. It then checks if

$$\mathbf{c}_0 + \mathbf{c}_1 = \mathsf{H}\Big([\mathfrak{g}^{\mathbf{r}_0}] * A_0^{\mathbf{c}_0} \| [\mathfrak{g}^{\mathbf{r}_1}] * A_1^{\mathbf{c}_1} \| \mathsf{M}\Big). \tag{7}$$

If it holds, it outputs a signature $\sigma = (\mathbf{c}_0, \mathbf{c}_1, \mathbf{r}_0, \mathbf{r}_1) \in (\mathbb{Z}_d^\kappa)^2 \times (\mathbb{Z}_N^\kappa)^2$, and otherwise $\perp$.

$\mathsf{BS.Verify}(\mathsf{pk}, \mathsf{M}, \sigma)$: The verifier outputs 1 if Eq. 7 holds, and otherwise 0.

```
BS.KGen$(1^n)$
101 :  $(a_0, a_1, \delta) \xleftarrow{\$} \mathbb{Z}_N^2 \times \{0,1\}$
102 :  $(A_0^j)_{j\in\mathbb{Z}_d} \leftarrow ([\mathfrak{g}^{a_0\zeta^j}] * E_0)_{j\in\mathbb{Z}_d}$
103 :  $(A_1^j)_{j\in\mathbb{Z}_d} \leftarrow ([\mathfrak{g}^{a_1\zeta^j}] * E_0)_{j\in\mathbb{Z}_d}$
104 :  $\mathsf{pk} \leftarrow ((A_0^j)_{\in\mathbb{Z}_d}, (A_1^j)_{j\in\mathbb{Z}_d})$
105 :  return $(\mathsf{pk}, \mathsf{sk} = (\delta, a_\delta))$
```

```
BS.S$_1$(sk)
201 :  parse $(\delta, a_\delta) \leftarrow \mathsf{sk}$
202 :  $\mathbf{y}_\delta^* \xleftarrow{\$} \mathbb{Z}_N^\kappa$
203 :  $(\mathbf{c}_{1-\delta}^*, \mathbf{r}_{1-\delta}^*) \xleftarrow{\$} \mathbb{Z}_d^\kappa \times \mathbb{Z}_N^\kappa$
204 :  for $j \in \mathbb{Z}_d$
205 :      $\mathbf{Y}_\delta^{j*} = [\mathfrak{g}^{\mathbf{y}_\delta^*\zeta^j}] * E_0$
206 :      $\mathbf{Y}_{1-\delta}^{j*} = [\mathfrak{g}^{\mathbf{r}_{1-\delta}^*\zeta^j}] * A_{1-\delta}^{\mathbf{c}_{1-\delta}^*+\mathbf{j}}$
207 :  $\mathsf{state}_\mathsf{S} = (\mathbf{y}_\delta^*, \mathbf{c}_{1-\delta}^*, \mathbf{r}_{1-\delta}^*)$
208 :  $\rho_{\mathsf{S},1} = (\mathbf{Y}_0^{j*}, \mathbf{Y}_1^{j*})_{j\in\mathbb{Z}_d}$
209 :  return $(\mathsf{state}_\mathsf{S}, \rho_{\mathsf{S},1})$
```

```
BS.U$_1$(pk, M, $\rho_{\mathsf{S},1}$)
301 :  parse $(\mathbf{Y}_0^{j*}, \mathbf{Y}_1^{j*})_{j\in\mathbb{Z}_d} \leftarrow \rho_{\mathsf{S},1}$
302 :  for $b \in \{0,1\}$ do
303 :      $(\mathbf{d}_b, \mathbf{z}_b) \xleftarrow{\$} \mathbb{Z}_d^\kappa \times \mathbb{Z}_N^\kappa$
304 :      $\mathbf{Z}_b = [\mathfrak{g}^{\mathbf{z}_b}] * \left(Y_{b,0}^{d_{b,0}*}, \dots, Y_{b,\kappa-1}^{d_{b,\kappa-1}*}\right)$
305 :  $\mathbf{c} \ = \mathsf{H}(\mathbf{Z}_0 || \mathbf{Z}_1 || \mathsf{M})$
306 :  $\mathbf{c}^* = \mathbf{c} - \mathbf{d}_0 - \mathbf{d}_1 \in \mathbb{Z}_d^\kappa$
307 :  $\mathsf{state}_\mathsf{U} \leftarrow (\mathbf{d}_0, \mathbf{d}_1, \mathbf{z}_0, \mathbf{z}_1)$
308 :  return $\left(\mathsf{state}_\mathsf{U}, \rho_\mathsf{U} = \mathbf{c}^*\right)$
```

```
BS.S$_2$(state$_\mathsf{S}$, $\rho_\mathsf{U}$)
401 :  parse $(\mathbf{y}_\delta^*, \mathbf{c}_{1-\delta}^*, \mathbf{r}_{1-\delta}^*) \leftarrow \mathsf{state}_\mathsf{S}$
402 :  parse $\mathbf{c}^* \leftarrow \rho_\mathsf{U}$
403 :  $\mathbf{c}_\delta^* \leftarrow \mathbf{c}^* - \mathbf{c}_{1-\delta}^*$
404 :  $\mathbf{r}_\delta^* \leftarrow \mathbf{y}_\delta^* - a_\delta\zeta^{\mathbf{c}_\delta^*}$
405 :  return $\rho_{\mathsf{S},2} = \left(\mathbf{c}_0^*, \mathbf{c}_1^*, \mathbf{r}_0^*, \mathbf{r}_1^*\right)$
```

```
BS.U$_2$(state$_\mathsf{U}$, $\rho_{\mathsf{S},2}$)
501 :  parse $(\mathbf{d}_0, \mathbf{d}_1, \mathbf{z}_0, \mathbf{z}_1) \leftarrow \mathsf{state}_\mathsf{U}$
502 :  parse $\left(\mathbf{c}_0^*, \mathbf{c}_1^*, \mathbf{r}_0^*, \mathbf{r}_1^*\right) \leftarrow \rho_{\mathsf{S},2}$
503 :  for $(b, j) \in \{0,1\} \times \mathbb{Z}_d$
504 :      if $[\mathfrak{g}^{\mathbf{r}_b^*\zeta^j}] * A_b^{[\mathbf{c}_b^*+\mathbf{j}]_d} \neq \mathbf{Y}_b^{j*}$
505 :          return $\sigma = \perp$
506 :  for $b \in \{0,1\}$
507 :      $\mathbf{c}_b \leftarrow \mathbf{c}_b^* + \mathbf{d}_b$
508 :      $\mathbf{r}_b \leftarrow \mathbf{z}_b + \mathbf{r}_b^*\zeta^{\mathbf{d}_b}$
509 :  $\mathbf{c}' = \mathsf{H}\left([\mathfrak{g}^{\mathbf{r}_0}] * A_0^{\mathbf{c}_0} \,\|\, [\mathfrak{g}^{\mathbf{r}_1}] * A_1^{\mathbf{c}_1} \,\|\, \mathsf{M}\right)$
510 :  if $\mathbf{c}_0 + \mathbf{c}_1 = \mathbf{c}'$
511 :      return $\sigma = (\mathbf{c}_0, \mathbf{c}_1, \mathbf{r}_0, \mathbf{r}_1)$
512 :  return $\sigma = \perp$
```

```
BS.Verify(pk, M, $\sigma$)
601 :  parse $(\mathbf{c}_0, \mathbf{c}_1, \mathbf{r}_0, \mathbf{r}_1) \leftarrow \sigma$
602 :  $\mathbf{c}' = \mathsf{H}\left([\mathfrak{g}^{\mathbf{r}_0}] * A_0^{\mathbf{c}_0} \,\|\, [\mathfrak{g}^{\mathbf{r}_1}] * A_1^{\mathbf{c}_1} \,\|\, \mathsf{M}\right)$
603 :  if $\mathbf{c}_0 + \mathbf{c}_1 = \mathbf{c}'$
604 :      return 1
605 :  return 0
```

Fig. 9 The optimized version of the blind signature where H is a hash function and ζ is a d-th primitive root of unity. Recall $\mathbb{Z}_d = \{0, 1, \dots, d-1\}$ and that we use the notations $\mathbf{d} = (d_0, \dots, d_{\kappa-1}) \in \mathbb{Z}_d^\kappa$ and $\mathbf{Y}^j = (Y_0^j, \dots, Y_{\kappa-1}^j) \in \mathcal{E}\ell\ell^\kappa$. Moreover, if $\mathbf{c} \in \mathbb{Z}_d^\kappa$, then $A_b^{[\mathbf{c}]_d}$ is simply $A_b^{\mathbf{c}}$ for $b \in \{0, 1\}$. See the caption of Fig. 8 for further explanation on the notations

Remark 4 One can observe that the only source of overhead in the communication bandwidth compared to the blind signature in Sect. 5 is in $\mathsf{BS.S}_1$. The bandwidth is increased by a factor of $\frac{d\kappa}{2n}$.

Remark 5 We remark that it is possible to fuse our partial blindness technique and the generalized construction in this section and obtain an optimized PBS variant. By doing so, we can obtain a PBS with a smaller signature size based on the rGAIP. Roughly, there are three sequences of the curves in the public statement $(A_0, A_1, A_2) = \left(([\mathfrak{g}^{a_0\zeta_j}] * E_0)_{j\in\mathbb{Z}_d}, ([\mathfrak{g}^{a_1\zeta_j}] * E_0)_{j\in\mathbb{Z}_d}, ([\mathfrak{g}^{a_2\zeta_j}] * E_0)_{j\in\mathbb{Z}_d}\right)$ where the secret key of the third public key is derived from the public information. The underlying sigma protocol is to prove for a two-out-of-three secret corresponding to this statement.

However, given the proofs in Sect. 6 and in this section, we expect the proof to be highly involved. We leave this as a future work.

7.5 Proof of correctness and blindness

The subsection shows that our blind signature presented in Sect. 7.4 has (perfect) correctness and blindness.

Theorem 20 *The blind signature scheme in Fig. 9 is (perfectly) correct.*

Proof To show correctness, it suffices to show the equation

$$\mathbf{c}_0 + \mathbf{c}_1 = \mathsf{H}\left([\mathfrak{g}^{\mathbf{r}_0}] * A_0^{\mathbf{c}_0} \ \| \ [\mathfrak{g}^{\mathbf{r}_1}] * A_1^{\mathbf{c}_1} \ \| \ \mathsf{M}\right)$$

holds when both the signer and user follow the protocol.

From the description of $\mathsf{BS.U}_1$, $\mathsf{BS.S}_2$ and $\mathsf{BS.U}_2$, we have $\mathbf{c} = \mathbf{c}^* + \mathbf{d}_0 + \mathbf{d}_1$, $\mathbf{c}^* = \mathbf{c}_1^* + \mathbf{c}_2^*$, and $\mathbf{c}_b = \mathbf{c}_b^* + \mathbf{d}_b$ for $b \in \{0, 1\}$. Therefore, we have $\mathbf{c} = \mathbf{c}_0 + \mathbf{c}_1$, which shows the l.h.r. equation. It remains to show $\mathbf{Z}_b = [\mathfrak{g}^{\mathbf{r}_b}] * A_b^{\mathbf{c}_b}$ for each $b \in \{0, 1\}$. Following the definition of $\mathbf{Z}_b$ computed by $\mathsf{BS.U}_1$, we have

$$\begin{aligned}
\mathbf{Z}_b &= [\mathfrak{g}^{\mathbf{z}_b}] * \left(Y_{b,0}^{d_{b,0}*}, \ldots, Y_{b,\kappa-1}^{d_{b,\kappa-1}*}\right) \\
&= [\mathfrak{g}^{\mathbf{z}_b}] * \left([\mathfrak{g}^{r^*_{b,0}\zeta^{d_{b,0}}}] * A_b^{[c^*_{b,0}+d_{b,0}]_d}, \ldots, [\mathfrak{g}^{r^*_{b,\kappa-1}\zeta^{d_{b,\kappa-1}}}] * A_b^{[c^*_{b,\kappa-1}+d_{b,\kappa-1}]_d}\right) \qquad (8) \\
&= \left([\mathfrak{g}^{z_{b,0}+r^*_{b,0}\zeta^{d_{b,0}}}] * A_b^{[c^*_{b,0}+d_{b,0}]_d}, \ldots, [\mathfrak{g}^{z_{b,\kappa-1}+r^*_{b,\kappa-1}\zeta^{d_{b,\kappa-1}}}] * A_b^{[c^*_{b,\kappa-1}+d_{b,\kappa-1}]_d}\right) \\
&= [\mathfrak{g}^{\mathbf{r}_b}] * A_b^{\mathbf{c}_b}, \qquad (9)
\end{aligned}$$

where Eq. 8 follows from the check performed by $\mathsf{BS.U}_2$ and Eq. 9 follows from the definition of $(\mathbf{c}_b, \mathbf{r}_b)$. □

Next, we will show the generalized blind signature has perfect blindness. Notably, blindness holds even under adversarially chosen keys. This is a strong property since if a malicious signer uses malformed supersingular curves in $\mathcal{E}\ell\ell$ without the ring structure as the public key, the user cannot detect this. The main reason why we can argue perfect blindness is that if the public key is malformed, then the pair of curves in the first message $(\mathbf{Y}_0^{j*}, \mathbf{Y}_1^{j*})_{j\in\mathbb{Z}_d}$ is also malformed in a controlled manner. If there exists one user state that leads to a valid signature, then we can argue that the first message must be in a specific (but possibly incorrect) form regardless of the user state. Using this, we are able to establish a bijection between an arbitrary user state and a valid signature conditioning on a fixed first and second signature messages and a user message. Namely, any valid signature could have been produced with an equal probability.

Theorem 21 *The blind signature scheme in Fig. 9 is (perfectly) blind under chosen keys.*

Proof Let $(\rho_{\mathsf{S},1,0}, \rho_{\mathsf{S},2,0})$ and $(\rho_{\mathsf{S},1,1}, \rho_{\mathsf{S},2,1})$ be the two sets of first and second-signer message pairs the adversary $\mathcal{A}$ queries to oracles U_1 and U_2. Moreover, let $\rho_{\mathsf{U},b}$ be the user message returned by oracle U_1 when $\mathcal{A}$ queries with $\rho_{\mathsf{S},1,b}$ for $b \in \{0, 1\}$, and let $(\sigma_{\mathsf{coin}}, \sigma_{1-\mathsf{coin}})$ be the two signatures $\mathcal{A}$ sees at the end, where note that these two corresponds to $\widetilde{\mathsf{M}}_0$ and $\widetilde{\mathsf{M}}_1$, respectively, regardless of the choice of $\mathsf{coin} \in \{0, 1\}$. We call $(\rho_{\mathsf{S},1,b}, \rho_{\mathsf{U},b}, \rho_{\mathsf{S},2,b})_{b\in\{0,1\}}$ the *view* of $\mathcal{A}$. To prove perfect blindness, it suffices to prove that the view is independent of $\mathsf{coin} \in \{0, 1\}$. In other words, since the randomness used by oracle U_1 is defined by $(\mathsf{state}_{\mathsf{U},b})_{b\in\{0,1\}}$ and oracle U_2 is deterministic, we prove that there exist two sets of states $(\mathsf{state}_{\mathsf{U},b}^{(0)})_{b\in\{0,1\}}$ and $(\mathsf{state}_{\mathsf{U},b}^{(1)})_{b\in\{0,1\}}$ that can be sampled by oracle U_1 with an equal probability such that they generate the same view to $\mathcal{A}$ but produce a different pair of signatures (σ_0, σ_1) and (σ_1, σ_0), respectively. Considering that the set of valid signature space and user randomness/state space is identical, we prove a stronger statement that for any non-aborting (partial) view $(\rho_{\mathsf{S},1,0}, \rho_{\mathsf{U},0}, \rho_{\mathsf{S},2,0})$ of $\mathcal{A}$, there is a bijection between a valid signature σ_0 on message M_0 and a state $\mathsf{state}_{\mathsf{U},0}$ of the oracle U_1. Below, we drop the subscript 0 for readability.

Let us denote the first and second-signer message as $\rho_{\mathsf{S},1} = (\mathbf{Y}_0^{j*}, \mathbf{Y}_1^{j*})_{j\in\mathbb{Z}_d}$, $\rho_{\mathsf{S},2} = (\mathbf{c}_0^*, \mathbf{c}_1^*, \mathbf{r}_0^*, \mathbf{r}_1^*)$, a user message as $\rho_\mathsf{U} = \mathbf{c}^*$, and a valid signature for message M as $\sigma = (\mathbf{c}_0, \mathbf{c}_1, \mathbf{r}_0, \mathbf{r}_1) \in (\mathbb{Z}_d^\kappa)^2 \times (\mathbb{Z}_N^\kappa)^2$. Here, note that any public key $\mathsf{pk} = ((A_0^j)_{j\in\mathbb{Z}_d}, (A_1^j)_{j\in\mathbb{Z}_d})$ output by the adversary (i.e., malicious signer) $\mathcal{A}$ can be efficiently checked to be valid elliptic curves (i.e., supersingularity) but cannot be checked if it has the correct cyclic structure.

We define a map between the signature $\sigma = (\mathbf{c}_0, \mathbf{c}_1, \mathbf{r}_0, \mathbf{r}_1)$ and user state $\mathsf{state}_\mathsf{U} = (\mathbf{d}_0, \mathbf{d}_1, \mathbf{z}_0, \mathbf{z}_1)$ by $\mathbf{d}_b = \mathbf{c}_b - \mathbf{c}_b^*$ and $\mathbf{z}_b = \mathbf{r}_b - \mathbf{r}_b^* \cdot \zeta^{\mathbf{d}_b}$ for $b \in \{0, 1\}$. It is easy to check that once the view (or $\rho_{\mathsf{S},2} = (\mathbf{c}_0^*, \mathbf{c}_1^*, \mathbf{r}_0^*, \mathbf{r}_1^*)$) is fixed, then this mapping is indeed a bijection. It remains to prove that this $\mathsf{state}_\mathsf{U}$ is a state that produces σ.

Observe that if $\mathsf{BS.U}_1(\mathsf{pk}, \mathsf{M}, \rho_{\mathsf{S},1})$ samples $\mathsf{state}_\mathsf{U}$, then it computes $\mathbf{Z}_b = [\mathfrak{g}^{\mathbf{z}_b}] * \left(Y_{b,0}^{d_{b,0}^*}, \dots, Y_{b,\kappa-1}^{d_{b,\kappa-1}^*}\right)$ for $b \in \{0, 1\}$ using $\rho_{\mathsf{S},1}$. It then sets $\mathbf{c}' = \mathsf{H}(\mathbf{Z}_0||\mathbf{Z}_1||\mathsf{M})$ and defines $\rho'_\mathsf{U} = \mathbf{c}'^* = \mathbf{c}' - \mathbf{d}_0 - \mathbf{d}_1$. Moreover, due to restrictions on the blindness game, the view is non-aborting for at least one state $\mathsf{state}_\mathsf{U}$. Combining this with the fact that the first check performed by $\mathsf{BS.U}_2(\mathsf{state}_\mathsf{U}, \rho_{\mathsf{S},2})$ only depends on $\rho_{\mathsf{S},2}$, and in particular independent of $\mathsf{state}_\mathsf{U}$, we have $[\mathfrak{g}^{\mathbf{r}_b^* \zeta^j}] * A_b^{[\mathbf{c}_b^* + \mathbf{j}]_d} = \mathbf{Y}_b^{j*}$ for $j \in \mathbb{Z}_d$ and any state $\mathsf{state}_\mathsf{U}$. Therefore, $\mathsf{BS.U}_2$ always outputs σ as desired since the signature σ is assumed to be valid.

It remains to check that $\rho'_\mathsf{U} = \mathbf{c}'^*$ generated by $\mathsf{BS.U}_1$ is the desired $\rho_\mathsf{U} = \mathbf{c}^*$ to complete the proof. Since σ is valid and due to the definition of $\mathsf{state}_\mathsf{U}$, we have $\mathbf{c}_0^* + \mathbf{c}_1^* + \mathbf{d}_0 + \mathbf{d}_1 = \mathsf{H}\left([\mathfrak{g}^{\mathbf{r}_0}] * A_0^{\mathbf{c}_0} \;\|\; [\mathfrak{g}^{\mathbf{r}_1}] * A_1^{\mathbf{c}_1} \;\|\; \mathsf{M}\right)$. Moreover, since the view is non-aborting, we are guaranteed that $\mathbf{c}^* = \mathbf{c}_0^* + \mathbf{c}_1^*$. Therefore, if $\mathbf{Z}_b = [\mathfrak{g}^{\mathbf{r}_b}] * A_b^{\mathbf{c}_b}$ for $b \in \{0, 1\}$, then we can conclude that $\mathbf{c}^* = \mathbf{c}'^*$ as desired. This can be checked as follows, where we use the fact that $[\mathfrak{g}^{\mathbf{r}_b^* \zeta^j}] * A_b^{[\mathbf{c}_b^* + \mathbf{j}]_d} = \mathbf{Y}_b^{j*}$ for $j \in \mathbb{Z}_d$ in the second equality:

$$\begin{aligned}
\mathbf{Z}_b &= [\mathfrak{g}^{\mathbf{z}_b}] * \left(Y_{b,0}^{d_{b,0}*}, \dots, Y_{b,\kappa-1}^{d_{b,\kappa-1}*}\right)\\
&= [\mathfrak{g}^{\mathbf{z}_b}] * \left([\mathfrak{g}^{r_{b,0}^* \zeta^{d_{b,0}}}] * A_b^{[c_{b,0}^* + d_{b,0}]_d}, \dots, [\mathfrak{g}^{r_{b,\kappa-1}^* \zeta^{d_{b,\kappa-1}}}] * A_b^{[c_{b,\kappa-1}^* + d_{b,\kappa-1}]_d}\right)\\
&= \left([\mathfrak{g}^{z_{b,0} + r_{b,0}^* \zeta^{d_{b,0}}}] * A_b^{[c_{b,0}^* + d_{b,0}]_d}, \dots, [\mathfrak{g}^{z_{b,\kappa-1} + r_{b,\kappa-1}^* \zeta^{d_{b,\kappa-1}}}] * A_b^{[c_{b,\kappa-1}^* + d_{b,\kappa-1}]_d}\right)\\
&= [\mathfrak{g}^{\mathbf{r}_b}] * A_b^{\mathbf{c}_b}.
\end{aligned}$$

This completes the proof. □

7.6 Proof of one-more unforgeability

Our proof of OMUF consists of preparing the necessary tools present in Sect. 4 to invoke Theorem 3. Specifically, we define instances $\mathbf{I}_0, \mathbf{I}_1$ (see Definition 12), the map $\Phi_{\mathsf{rand}, \overrightarrow{h}}$ (see Definition 19), the witness extractors $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ (see Definition 20) and prove that Lemmas 1 and 2 hold. We refer the readers to Sect. 5.5 for some of the notations used below.

Preparation: instances

Let us first define the $\mathbf{0}$-side instance $\mathbf{I}_0$ and the $\mathbf{1}$-side instance $\mathbf{I}_1$. Below, we assume the adversary against the one-more unforgeability game makes ℓ-signing queries in total.

A $\mathbf{0}$-side instance $\mathbf{I}_0 = (0, a_0, \mathbf{A}_1, \overrightarrow{\mathbf{y}_0^*}, \overrightarrow{\mathbf{r}_1^*}, \overrightarrow{\mathbf{c}_1^*})$ is defined as follows:

- $(0, a_0)$: The secret key sk when $\delta = 0$.

- $\mathbf{A}_1$: The part of the public key $\mathsf{pk} = (\mathbf{A}_0 = (A_0^j)_{j\in\mathbb{Z}_d}, \mathbf{A}_1 = (A_1^j)_{j\in\mathbb{Z}_d})$ whose secret key is unknown.
- $\mathbf{y}_0^{*(k)}$: The exponent of the commitment $(\mathbf{Y}_0^{j*(k)})_{j\in\mathbb{Z}_d}$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 0$ such that $\mathbf{Y}_0^{j*(k)} = [\mathfrak{g}^{\mathbf{y}_0^{*(k)}\zeta^j}] * E_0$ for each $j \in \mathbb{Z}_d$.
- $\mathbf{c}_1^{*(k)}$: The simulated challenge in the k-th ($k \in [\ell]$) first-sender message when $\delta = 0$.
- $\mathbf{r}_1^{*(k)}$: The exponent of the commitment $\mathbf{Y}_1^{j*(k)}$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 0$ such that $\mathbf{Y}_1^{j*(k)} = [\mathfrak{g}^{\mathbf{r}_1^{*(k)}\zeta^j}] * A_1^{[\mathbf{c}_1^{*(k)}+\mathbf{j}]_d}$ for each $j \in \mathbb{Z}_d$.

A **1**-side instance $\mathbf{I}_1 = (1, a_1, \mathbf{A}_0, \overrightarrow{\mathbf{y}_1^*}, \overrightarrow{\mathbf{r}_0^*}, \overrightarrow{\mathbf{c}_0^*})$ is defined as follows:

- $(1, a_1)$: The secret key sk when $\delta = 1$.
- $\mathbf{A}_0$: The part of the public key $\mathsf{pk} = (\mathbf{A}_0 = (A_0^j)_{j\in\mathbb{Z}_d}, \mathbf{A}_1 = (A_1^j)_{j\in\mathbb{Z}_d})$ whose secret key is unknown.
- $\mathbf{y}_1^{*(k)}$: The exponent of the commitment $(\mathbf{Y}_1^{j*(k)})_{j\in\mathbb{Z}_d}$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 1$ such that $(\mathbf{Y}_1^{j*(k)})_{j\in\mathbb{Z}_d} = [\mathfrak{g}^{\mathbf{y}_1^{*(k)}\zeta^j}] * E_0$ for each $j \in \mathbb{Z}_d$.
- $\mathbf{c}_0^{*(k)}$: The simulated challenge in the k-th ($k \in [\ell]$) first-sender message when $\delta = 1$.
- $\mathbf{r}_0^{*(k)}$: The exponent of the commitment $\mathbf{Y}_0^{j*(k)}$ in the k-th ($k \in [\ell]$) first-sender message when $\delta = 1$ such that $\mathbf{Y}_0^{j*(k)} = [\mathfrak{g}^{\mathbf{r}_0^{*(k)}\zeta^j}] * A_0^{[\mathbf{c}_0^{*(k)}+\mathbf{j}]_d}$ for each $j \in \mathbb{Z}_d$.

7.6.1 Preparation: Map $\Phi_{\mathsf{rand},\vec{h}}$

We next define the map $\Phi_{\mathsf{rand},\vec{h}}$ that maps a **0**-side instance $\mathbf{I}_0$ into a **1**-side instance $\mathbf{I}_1$ and vice versa. Concretely, a **0**-side instance $\mathbf{I}_0 = (0, a_0, \mathbf{A}_1, \overrightarrow{\mathbf{y}_0^*}, \overrightarrow{\mathbf{r}_1^*}, \overrightarrow{\mathbf{c}_1^*})$ maps to a **1**-side instance $\mathbf{I}_1$ such that

$$\mathbf{I}_1 = (1, a_1,\ A_0 = (A_0^j)_{j\in\mathbb{Z}_d} = ([\mathfrak{g}^{a_0\zeta^j}] * E_0)_{j\in\mathbb{Z}_d},\ \overrightarrow{\mathbf{y}_1^*} = \overrightarrow{\mathbf{r}_1^*} + a_1\zeta^{\overrightarrow{\mathbf{c}_1^*}},\ \overrightarrow{\mathbf{c}_0^*} = \overrightarrow{\mathbf{c}^*} - \overrightarrow{\mathbf{c}_1^*},\ \overrightarrow{\mathbf{r}_0^*} = \overrightarrow{\mathbf{y}_0^*} - a_0\zeta^{\overrightarrow{\mathbf{c}_0^*}}),$$

where $a_1 \in \mathbb{Z}_N$ such that $A_1^0 = [\mathfrak{g}^{a_1}] * E_0$ and recall that $\overrightarrow{\mathbf{c}^*} = \vec{e}(\mathbf{I}_0, \mathsf{rand}, \vec{h})$. On the other hand, a **1**-side instance $\mathbf{I}_1 = (1, a_1, \mathbf{A}_0, \overrightarrow{\mathbf{y}_1^*}, \overrightarrow{\mathbf{r}_0^*}, \overrightarrow{\mathbf{c}_0^*})$ maps to a **0**-side instance $\mathbf{I}_0$ such that

$$\mathbf{I}_0 = (0, a_0,\ \mathbf{A}_1 = (A_1^j)_{j\in\mathbb{Z}_d} = ([\mathfrak{g}^{a_1\zeta^j}] * E_0)_{j\in\mathbb{Z}_d},\ \overrightarrow{\mathbf{y}_0^*} = \overrightarrow{\mathbf{r}_0^*} + a_0\zeta^{\overrightarrow{\mathbf{c}_0^*}},\ \overrightarrow{\mathbf{c}_1^*} = \overrightarrow{\mathbf{c}^*} - \overrightarrow{\mathbf{c}_0^*},\ \overrightarrow{\mathbf{r}_1^*} = \overrightarrow{\mathbf{y}_1^*} - a_1\zeta^{\overrightarrow{\mathbf{c}_1^*}})$$

where $a_0 \in \mathbb{Z}_N$ such that $A_0^0 = [\mathfrak{g}^{a_0}] * E_0$ and recall that $\overrightarrow{\mathbf{c}^*} = \vec{e}(\mathbf{I}_1, \mathsf{rand}, \vec{h})$.

7.6.2 Preparation: witness extractors (Ext_0, Ext_1)

Fix $\mathbf{I}$, rand and let $(\vec{h}, \vec{h}') \in F_i(\mathbf{I}, \mathsf{rand})$ for some $i \in [\ell + 1]$. Moreover, denote the two signatures $\sigma = (\mathbf{c}_0, \mathbf{c}_1, \mathbf{r}_0, \mathbf{r}_1), \sigma' = (\mathbf{c}'_0, \mathbf{c}'_1, \mathbf{r}'_0, \mathbf{r}'_1)$ be the signatures that correspond to $\mathbf{c}^{(i)}$ and $\mathbf{c}'^{(i)}$, respectively, where recall $\mathbf{c}^{(i)}$ (resp. $\mathbf{c}'^{(i)}$) is the i-th entry of $\vec{h}$ (resp. $\vec{h}'$). In particular, we have $\mathbf{c}_0 + \mathbf{c}_1 = \mathbf{c}^{(i)}$ and $\mathbf{c}'_0 + \mathbf{c}'_1 = \mathbf{c}'^{(i)}$. We define the witness extractors $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ as in Fig. 10.

The following lemma establishes the correctness of the witness extractors.

```
Ext_0(σ, σ')
101 :  if $\exists t \in [\kappa]$ s.t. $c_{0,t} \neq c'_{0,t}$
102 :      $L \leftarrow \mathsf{Ext}'(r_{0,t}, r'_{0,t}, c'_{0,t}, c_{0,t})$
103 :      for $a' \in L$
104 :          if $[\mathfrak{g}^{a'}] * E_0 = A_0^0$
105 :          return $a'$
106 :  return $\perp$
```

```
Ext_1(σ, σ')
101 :  if $\exists t \in [n]$ s.t. $c_{1,t} \neq c'_{1,t}$
102 :      $L \leftarrow \mathsf{Ext}'(r_{1,t}, r'_{1,t}, c'_{1,t}, c_{1,t})$
103 :      for $a' \in L$
104 :          if $[\mathfrak{g}^{a'}] * E_0 = A_1^0$
105 :          return $a'$
106 :  return $\perp$
```

Fig. 10 Witness extractors for our generalized blind signature for σ, σ'. In the above, $\sigma = (\mathbf{c}_0, \mathbf{c}_1, \mathbf{r}_0, \mathbf{r}_1)$ and $\sigma' = (\mathbf{c}'_0, \mathbf{c}'_1, \mathbf{r}'_0, \mathbf{r}'_1)$, where $\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}'_0, \mathbf{c}'_1$ live in $\mathbb{Z}_d^\kappa$ and $\mathbf{r}_0, \mathbf{r}_1, \mathbf{r}'_0, \mathbf{r}'_1$ live in $\mathbb{Z}_N^\kappa$. Ext' is the extractor in Lemma 19. Non-bold font indicates the entries of a vector

Lemma 22 $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ *in Fig. 10 satisfy the definition of witness extractors in Definition 20.*

Proof By the definition of $\mathrm{F}_i(\mathbf{I}, \mathsf{rand})$ (see Definition 14), we have $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in \mathsf{Succ}$ and $\mathbf{c}^{(i)} \neq \mathbf{c}'^{(i)}$. The former implies that the two signatures σ and σ' are valid. Concretely, we have

$$\mathbf{c}^{(i)} = \mathbf{c}_0 + \mathbf{c}_1 = \mathsf{H}\Big([\mathfrak{g}^{\mathbf{r}_0}] * A_0^{\mathbf{c}_0} \parallel [\mathfrak{g}^{\mathbf{r}_1}] * A_1^{\mathbf{c}_1} \parallel \mathsf{M}\Big)$$
$$\mathbf{c}'^{(i)} = \mathbf{c}'_0 + \mathbf{c}'_1 = \mathsf{H}\Big([\mathfrak{g}^{\mathbf{r}'_0}] * A_0^{\mathbf{c}'_0} \parallel [\mathfrak{g}^{\mathbf{r}'_1}] * A_1^{\mathbf{c}'_1} \parallel \mathsf{M}'\Big).$$

Moreover, since $\overrightarrow{h}$ and $\overrightarrow{h}'$ agree up to the i-th entry and the challenger and adversary's randomness are fixed, the input to the hash functions agree. Namely, we have

$$[\mathfrak{g}^{\mathbf{r}_0}] * A_0^{\mathbf{c}_0} = [\mathfrak{g}^{\mathbf{r}'_0}] * A_0^{\mathbf{c}'_0} \quad \wedge \quad [\mathfrak{g}^{\mathbf{r}_1}] * A_1^{\mathbf{c}_1} = [\mathfrak{g}^{\mathbf{r}'_1}] * A_1^{\mathbf{c}'_1} \quad \wedge \quad \mathsf{M} = \mathsf{M}'$$

Since $\mathbf{c}^{(i)} \neq \mathbf{c}'^{(i)}$, we must have $\mathbf{c}_0 \neq \mathbf{c}'_0$ or $\mathbf{c}_1 \neq \mathbf{c}'_1$. Thus, due to the special soundness of the underlying sigma protocol (see Sect. 7.2) one of Ext_0 or Ext_1 always outputs a valid secret key. This completes the proof. □

7.6.3 Proof of one-more unforgeability

We have the following two lemmas required to invoke the main theorem Theorem 25. Since the proof is almost identical to our earlier proofs in Sect. 5.5, we omit the proof of the lemmas.

Lemma 23 *Lemma 1 holds for our definition of the map* $\Phi_{\mathsf{rand}, \overrightarrow{h}}$ *above.*

Lemma 24 *Lemma 2 holds for our definition of the witness extractors* $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ *Fig. 10.*

Combining everything together, we obtain the following.

Theorem 25 (One-more unforgeability) *The partially blind signature scheme in Fig. 9 is one-more unforgeable. More precisely, for all $\ell \in \mathbb{N}$, if there exists an adversary $\mathcal{A}$ that makes Q hash queries to the random oracle and breaks the ℓ-one more unforgeability of our scheme with advantage $\epsilon_\mathcal{A} \geq \frac{C_1}{d^\kappa} \cdot \binom{Q}{\ell+1}$, then there exists an algorithm $\mathcal{B}$ that breaks the ζ-*rGAIP *problem with advantage $\epsilon_\mathcal{B} \geq C_2 \cdot \frac{\epsilon_\mathcal{A}^2}{\binom{Q}{\ell+1}^2 \cdot (\ell+1)^3}$ for some universal positive constants C_1 and C_2. Note we use a d-th primitive root of unity ζ and κ denotes the number of parallel repetitions of the underlying sigma protocol.*

Proof Upon receiving an rGAIP instance, the wrapper proceeds as described in Sect. 4.2. The proof follows from the above Lemmas 23 and 24 and Theorem 3 (i.e., [55, Theorem 1]) and the result follows. □

8 Analysis of ring GAIP

This section analyzes the ζ_d-ring group action inverse problem (ζ_d-rGAIP) over CSIDH-512. Section 8.1 discusses the existence of primitive d-th roots in $\mathbb{Z}_N^\times$, and a method to find one which satisfies Requirement 1. Section 8.2 recalls the most efficient classical and quantum algorithms against GAIP and presents a structural attack on ζ_d-rGAIP which effectively reduces ζ_d-rGAIP for a few choices of d to a GAIP instance with a smaller group size compared to the original group considered by ζ_d-rGAIP. In Sect. 8.3, we complement our cryptanalysis by proving that ζ_d-rGAIP for a few choices of d is as hard as GAIP defined over the same group. This shows optimality of our structural attack for ζ_d-rGAIP for some choices of d. We note that the concrete value of d's that admit an attack or a reduction depends on the concrete CSIDH-512 parameter set.

8.1 Finding a root of unity and satisfying requirement 1

We briefly discuss the existence of and a process for finding a primitive d-th root of unity $\zeta_d \in \mathbb{Z}_N^\times$ which satisfies Requirement 1. Firstly, it is a straightforward consequence of the fundamental theorem of finitely-generated abelian groups and the definition of $\lambda(N)$ that $\mathbb{Z}_N^\times \cong \mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2} \times \cdots \times \mathbb{Z}_{n_r}$ where $n_1 \mid n_2 \mid \cdots \mid n_r$ and $n_r = \lambda(N)$, so that a d-th root of unity exists if and only if d is a divisor of $\lambda(N)$—here, $\lambda(\cdot)$ is the Carmichael function.

To find such a root for a given valid d, the most intuitive method, perhaps, is to start with a primitive $\lambda(N)$-th root of unity $\zeta_{\lambda(N)}$, and compute $\zeta_{\lambda(N)}^{\frac{\lambda(N)}{d}}$, which will have order exactly d. Unfortunately, this may result in a d-th root of unity that does not meet Requirement 1 (even when one exists which satisfies Requirement 1). In particular, we have to ensure that ζ is a generator modulo all but a small collection of small prime power divisors of N to conclude $\eta_d = \mathsf{lcm}_{i\in[d-1]}(\mathsf{gcd}(\zeta^i - 1, N)) = poly(n)$. First, let $N = 2^f p_1^{e_1} p_2^{e_2} \cdots p_t^{e_t}$ be the prime decomposition of N—note that we treat the prime 2 separately for later convenience.[12] By the Chinese remainder theorem we have

$$\mathbb{Z}_N^\times \cong \mathbb{Z}_{2^f}^\times \times \mathbb{Z}_{p_1^{e_1}}^\times \times \mathbb{Z}_{p_2^{e_2}}^\times \cdots \times \mathbb{Z}_{p_t^{e_t}}^\times.$$

Now, for each j such that $d \mid p_j - 1$, we may find an element γ_j of $\mathbb{Z}_{p_j^{e_j}}^\times$ of order precisely d, since $\mathbb{Z}_{p_j^{e_j}}$ is cyclic of order $p_j^{e_j-1}(p_j - 1)$.[13] Let $I_d = \{j \,:\, 1 \le j \le t \text{ and } d \mid p_j\}$. Applying the Chinese remainder theorem again, there exists a solution $\zeta_d \pmod N$ to the

[12] In CSIDH this is not necessary, since the choice of $p \equiv 3 \pmod 4$ guarantees that $N = \#\mathcal{C}\ell(\mathcal{O})$ is odd. But this result extends to other groups, and so we treat it in full generality.

[13] Since p_j is odd, we have $\lambda(p_j^{e_j}) = \varphi(p_j^{e_j}) = p^{e_j-1}(p_j - 1)$ so that $\mathbb{Z}_{p_j^{e_j}}^\times$ is cyclic of order divisible by d.

system

$$\zeta_d \equiv \begin{cases} \gamma_j \pmod{p_j} & \text{if } j \in I_d \\ 1 \pmod{p_j^{e_j}} & \text{if } j \in [t] \setminus I_d \\ 1 \pmod{2^f} & \end{cases}.$$

Note that this ζ_d is coprime to N, and moreover for $i = 1, 2, \ldots d-1$ we have

$$\begin{aligned} \zeta_d^i &\not\equiv 1 \pmod{p_j} \text{ for } j \in I_d \\ \zeta_d^i &\equiv 1 \pmod{p_j^{e_j}} \text{ for } j \in [t] \setminus I_d \\ \zeta_d^i &\equiv 1 \pmod{2^f} \end{aligned}$$

so that $\eta_d = \mathsf{lcm}_{i \in [d-1]}(\mathsf{gcd}(\zeta_d^i - 1, N)) = 2^f \prod_{j \in [t] \setminus I_d} p_j^{e_j}$. In particular, if d is chosen so that $2^f \prod_{j \in [t] \setminus I_d} p_j = poly(n)$, our ζ_d satisfies Requirement 1, as required.

Concretely, for the CSIDH-512 parameter sets we have

$$\begin{aligned} N &= 3 \times 37 \times 1407181 \times 51593604295295867744293584889 \\ &\quad \times 31599414504681995853008278745587832204909, \\ \lambda(N) &= 2^3 \times 3^2 \times 5 \times 7^2 \times 47 \times 71 \times 499 \times 43872112495999887537664613 \\ &\quad \times 111265544030570407933127742061928986637, \end{aligned}$$

and we can construct the following primitive d-th roots of unity with respect to CSIDH-512 following this method for $2 \leq d \leq 9$, 47 and 499:

$$\begin{aligned} \zeta_2 &= -1 \\ \zeta_3 &= 2477699437908495650371104512535948994004956355404732770089878647330138924903\,49 \\ \zeta_4 &= 847249911467870199377355343817339592122893618913963633620986484656468775794\,5 \\ \zeta_5 &= 724530243246883951871818693965099412690399512626895792249146926276748199981\,75 \\ \zeta_7 &= 728604689428996897384604951715181217845042118483738631839298086369175557888\,57 \\ \zeta_8 &= 179680810279510028621279942318029725212447549505150327666400650549608102102\,90 \\ \zeta_9 &= 1445324672113289129383144298979303579836224540652760782552429210942581039527\,04 \\ \zeta_{47} &= 628478118037960958300537125640801634748544703297957974485612968823593372682\,0 \\ \zeta_{499} &= 277169907100153008535427356676756656338281710679312797172941829358721485079\,72. \end{aligned}$$

Remark 6 In the list above, we only display d that is a prime power. For other composite divisors of $\lambda(N)$, one can obtain the corresponding root by multiplication. For instance, we can obtain $\zeta_{23453} = \zeta_{47}\zeta_{499}$.

For the CSIDH-512 parameter set, the totients of the small prime divisors of N have the following (maximal) small prime power divisors:

$$\begin{aligned} \varphi(3) &: 2 \\ \varphi(37) &: 2^2,\ 3^2 \\ \varphi(1407181) &: 2^2,\ 3,\ 5,\ 47,\ 499 \\ \varphi(51593604295295867744293584889) &: 2^3,\ 3,\ 7^2 \\ \varphi(31599414504681995853008278745587832204909) &: 2^2,\ 71. \end{aligned}$$

This implies that for the CSIDH-512 parameter we can find only a 4th root of unity meeting Requirement 1 (with $\eta_4 = 3$) since only $\mathbb{Z}_3^\times$ has no cyclic subgroups of order 4. For example, for any 3rd root of unity ζ_3, we always have a 134-bit divisor of $\mathsf{gcd}(\zeta_3, N)$. Therefore,

ζ_4-rGAIP over CSIDH-512 is the candidate hardness assumption that can be used for our optimized blind signature construction.

In the next subsection, we show that the hardness of ζ_d-rGAIP varies with the choice of ζ_d. Since we believe ζ_d-rGAIP may be of independent interest, we waive Requirement 1 when considering the cryptanalysis.

8.2 Cryptanalysis and structural attack on rGAIP

In the previous section, we showed how to choose a root ζ_d according to the decomposition of the multiplication group of $\mathbb{Z}_N^\times$. In this section, we show that the underlying structure of ζ_d in each component is related to the security of ζ_d-rGAIP by presenting a concrete cryptanalysis on the overstretched ζ_d-rGAIP with respect to the CSIDH-512 parameters.

Generic attacks on GAIP

The best known classical algorithm against GAIP is the meet-in-the-middle attack [46, 47] with time complexity $O(\sqrt{|\mathcal{C}\ell(\mathcal{O})|}) = O(\sqrt[4]{p})$ against GAIP.

The best-known quantum algorithm against GAIP is Kuperburg's algorithm [19, 58, 59, 68, 74]. Typically, given a challenge E to find $a \in \mathbb{Z}_N$ such that $E = [\mathfrak{g}^a] * E_0$, we have a hidden shift problem by defining $f(x) = [\mathfrak{g}^x] * E_0$ and $g(x) = [\mathfrak{g}^x] * E$, the permutations f, g over $\mathcal{E}\ell\ell$ are hidden shifted by a. By applying Kuperburg's algorithm, one can solve GAIP in time complexity $2^{O(\sqrt{\log(|\mathcal{C}\ell(\mathcal{O})|)})}$. It is not clear whether the additional structure can give an advantage to the adversary by reducing the group size *in general*. The subset $\{1, \zeta_d, \ldots, \zeta_d^{d-1}\}$ forms a group with multiplication instead of addition. Modifying the group action by restricting to the multiplication subgroup of $\mathbb{Z}_N^\times$ does not give a feasible g with a hidden shift a. Also, ζ generates the additive group $\mathbb{Z}_N$, so that the quotient group does not help in this case.

8.2.1 Structural attack on rGAIP

Let ζ_d be a d-th primitive root of unity and N be the class number.

We show that the underlying structure of the root in each component of $\mathbb{Z}_N^\times$ is related to security by displaying a structural attack against ζ_d-rGAIP and the efficacy of the attack is related to each $\mathsf{gcd}(\zeta_d^i - 1, N)$. We remark that the structural attack requires the order N to be squarefree. The requirement complies with the Cohen–Lenstra conjecture and our instantiation satisfies the requirement.

The high-level strategy of our structural attack is to break down a ζ_d-rGAIP instance into several GAIP instances over smaller subgroups or quotient groups. The idea is to exploit the differential information of any two curves in the instance and launch a Pohlig-Hellman-type attack. Recall that the instance is of the form $(X_0 = [\mathfrak{g}^a] * E_0, X_1 = [\mathfrak{g}^{a\zeta_d}] * E_0, \ldots, X_{d-1} = [\mathfrak{g}^{a\zeta_d^{d-1}}] * E_0)$. For any two curves X_i, X_j in the instance, there exists a unique group element $[\mathfrak{g}_{ij}] = [\mathfrak{g}^{a\zeta_d^j - a\zeta_d^i}] \in \mathcal{C}\ell(\mathcal{O})$ such that $[\mathfrak{g}_{ij}] * X_i = X_j$. Therefore, recovering the differential element $[\mathfrak{g}_{ij}]$ gives information about a. Typically, it is difficult to recover such $[\mathfrak{g}_{ij}]$ due to the size of the group and considering the GAIP of (X_i, X_j). However, depending on the knowledge of η_d derived from the public ζ_d, the hardness of the GAIP of the structural (X_i, X_j) can be reduced.

The following two lemmata show the decomposition into two GAIP instances with smaller groups.

Lemma 26 *Let $(\mathcal{C\ell}(\mathcal{O}), \mathcal{E\ell\ell}, E_0, *)$ be known-order EGA (KOEGA) where $\mathcal{C\ell}(\mathcal{O})$ is cyclic, and the other parameters are as described above. Let $m = \mathsf{gcd}(\zeta_d^j - \zeta_d^i, N)$, and let $m' = \frac{N}{m}$. Define $G_{ij} = \langle[\mathfrak{g}^{\zeta_d^j - \zeta_d^i}]\rangle$ and note that it is a subgroup of $\mathcal{C\ell}(\mathcal{O})$ with $[\mathfrak{g}_{ij}] \in G_{ij}$. Hence, one can recover $[\mathfrak{g}_{ij}]$ and $a' = a \cdot (\zeta_d^j - \zeta_d^i) \mod m'$ by solving* GAIP *problem of (X_i, X_j) over the subgroup G_{ij}.*

Proof G_{ij} is a cyclic subgroup generated by $[\mathfrak{g}^{\zeta_d^j - \zeta_d^i}]$ and $[\mathfrak{g}_{ij}] \in G_{ij}$ by definition. □

Observe that the quotient group $\mathcal{C\ell}(\mathcal{O})/G_{ij}$ has order m, and in fact the map $\phi\colon \mathcal{C\ell}(\mathcal{O})/G_{ij} \to \langle\mathfrak{g}^{m'}\rangle$ defined by $\phi(\overline{[\mathfrak{a}]}) = [\mathfrak{a}^{m'}]$ is an isomorphism. Solving the instance (X_i, X_j) yields $a \equiv a' \pmod{m}$. Observe that $[\mathfrak{g}^{(a-a')}] * ([\mathfrak{g}^{a'}] * E_0) = [\mathfrak{g}^a] * E_0 = X_0$, and moreover that $m' \mid (a - a')$. Thus, to find a, it suffices to solve the GAIP instance $([\mathfrak{g}^{a'}] * E_0, X_0)$ over $\mathbb{Z}_m \cong \langle\mathfrak{g}^{m'}\rangle \cong \mathcal{C\ell}(\mathcal{O})/G_{ij}$ to find $a - a'$. If a' has been recovered by solving the GAIP instance (X_i, X_j) over G_{ij}, we can then easily recover a. Thus we have established Lemma 27.

Lemma 27 *Let $(\mathcal{C\ell}(\mathcal{O}), \mathcal{E\ell\ell}, E_0, *)$ be KOEGA where $\mathcal{C\ell}(\mathcal{O})$ is cyclic and the order is squarefree, and let the other parameters be as described above. If $[\mathfrak{g}_{ij}] \in G_{ij}$, then we can recover $[\mathfrak{g}^a]$ by solving the GAIP instance $([\mathfrak{g}^{a'}] * E_0, X_0)$ over $\mathcal{C\ell}(\mathcal{O})/G_{ij}$.*

We see that the main strength against our structural attack depends on the GAIP hardness with the group size of $\max(|G_{ij}|, |G/G_{ij}|)$. Choosing an appropriate sequence of $(i_\ell, j_\ell)_{\ell=1}^k$, the root ζ_d gives the following ascending chain: $\{1\} = G_1 < G_2 < \cdots < G_k = \mathcal{C\ell}(\mathcal{O})$, where for each $\ell \in [k-1]$, $G_\ell = G_{ij}$ for some distinct $i, j \in [d]$.

We can conclude as follows.

Theorem 28 *Let $(\mathcal{C\ell}(\mathcal{O}), \mathcal{E\ell\ell}, E_0, *)$ be KOEGA where $\mathcal{C\ell}(\mathcal{O})$ is cyclic and the order is squarefree. Let ζ be a d-th root of unity for G. Let*

$$\{1\} = G_1 < G_2 < \cdots < G_k = \mathcal{C\ell}(\mathcal{O})$$

*be a chain for G where $G_\ell = \{[\mathfrak{g}^{n(\zeta^{j_\ell} - \zeta^{i_\ell})}] \mid n \in \mathbb{Z}_N\}$ for some $i_\ell, j_\ell \in \{0, \ldots, d-1\}$ for $\ell \in [k-1]$. Given a GAIP adversary $\mathcal{A}$ over the KOEGA model, there exists an ζ-*rGAIP *adversary $\mathcal{B}$ running in polynomial time with $O(k)$ queries to $\mathcal{A}$ such that* [14]

$$\mathsf{Adv}^{\zeta\text{-rGAIP}}_{(\mathcal{C\ell}(\mathcal{O}), \mathcal{E\ell\ell}, E_0, *)}(\mathcal{B}) \geq \prod_{\ell=1}^{k-1} \mathsf{Adv}^{\mathsf{GAIP}}_{(G_{\ell+1}/G_\ell, \mathcal{E\ell\ell}, X'_\ell, *)}(\mathcal{A}).$$

where X'_ℓ is some element of $\mathcal{E\ell\ell}$ chosen in the reduction.

Proof We proceed by induction on k. When $k = 2$ it is trivial, since $G_2/G_1 \cong \mathcal{C\ell}(\mathcal{O})$, so the algorithm $\mathcal{B}$ simply calls $\mathcal{A}$ on (X_0, X_1) and returns its output. For larger k, we apply Lemmas 26 and 27 to find that, to solve a ζ-rGAIP instance, it suffices to solve GAIP instances in G_{k-1} and G_k/G_{k-1}. By the inductive hypothesis, there exists a polynomial-time adversary $\mathcal{B}'$ making $O(k)$ calls to $\mathcal{A}$ which solves GAIP in G_{k-1} with advantage $\mathsf{Adv}^{\zeta\text{-rGAIP}}_{(\mathcal{C\ell}(\mathcal{O}), \mathcal{E\ell\ell}, E_0, *)}(\mathcal{B}') \geq \prod_{\ell=1}^{k-1} \mathsf{Adv}^{\mathsf{GAIP}}_{(G_{\ell+1}/G_\ell, \mathcal{E\ell\ell}, X'_\ell, *)}(\mathcal{A})$, and $\mathcal{A}$ solves the GAIP instance in G_k/G_{k-1} with advantage $\mathsf{Adv}^{\mathsf{GAIP}}_{(G_k/G_{k-1}, \mathcal{E\ell\ell}, X'_k, *)}(\mathcal{A})$ by definition. Our algorithm $\mathcal{B}$ is as

[14] Over KOEGA $(\mathcal{C\ell}(\mathcal{O}), \mathcal{E\ell\ell}, E_0, *)$, the group G can be represented by $\oplus_{i=1}^d \mathbb{Z}_{m_i}$. Therefore, it induces the KOEGA $(G_{\ell+1}/G_\ell, \mathcal{E\ell\ell}, X'_\ell, *)$ for any ℓ in a canonical way by using the operation from $(\mathcal{C\ell}(\mathcal{O}), \mathcal{E\ell\ell}, E_0, *)$.

Table 1 The upper row denotes ζ_d-rGAIP over CSIDH-512

ζ_d-rGAIP	ζ_2	ζ_3	ζ_4	ζ_5	ζ_7	ζ_8	ζ_9	ζ_{47}	ζ_{499}
GAIP with Group Size in $\log_2$	257	251	255	236	161	134	251	236	236

Using our cryptanalysis in Sect. 8.2, we reduce each ζ_d-rGAIP instance into a GAIP instance with a group size summarized in the lower row. Note that GAIP over CSIDH-512 is equivalent to ζ_2-rGAIP over CSIDH-512

follows: call $\mathcal{B}'$ to solve the GAIP instance in G_{k-1}, and then call $\mathcal{A}$ to solve the GAIP instance in G_k/G_{k-1}. This algorithm is clearly polynomial-time and makes $O(k)$ calls to $\mathcal{A}$; moreover, it succeeds with probability

$$\begin{aligned}\mathsf{Adv}^{\zeta\text{-rGAIP}}_{(\mathcal{C}\ell(\mathcal{O}),\mathcal{E}\ell\ell,E_0,*)}(\mathcal{B}) &\geq \mathsf{Adv}^{\mathsf{GAIP}}_{(G_k/G_{k-1},\mathcal{E}\ell\ell,X'_k,*)}(\mathcal{A})\cdot\prod_{\ell=1}^{k-2}\mathsf{Adv}^{\mathsf{GAIP}}_{(G_{\ell+1}/G_\ell,\mathcal{E}\ell\ell,X'_\ell,*)}(\mathcal{A})\\ &=\prod_{\ell=1}^{k-1}\mathsf{Adv}^{\mathsf{GAIP}}_{(G_{\ell+1}/G_\ell,\mathcal{E}\ell\ell,X'_\ell,*)}(\mathcal{A})\end{aligned}$$

as required. □

Using the aforementioned structural attack, the hardness of ζ_d-rGAIP is determined by the size of the largest quotient group $G_{\ell+1}/G_\ell$ for some $\ell \in [k-1]$.

Remark 7 We note that $\mathsf{gcd}(\zeta_d^i - 1, N)$ is divisible by a prime divisor p of N if and only if $\zeta_d^{\frac{d}{\mathsf{gcd}(i,d)}} \equiv 1 \pmod p$. Thus we only need to calculate $\mathsf{gcd}(\zeta_d^{d'} - 1, N)$ for every divisor d' of d to find η_d. In particular, when d is prime, we need only compute $\mathsf{gcd}(\zeta_d - 1, N)$ to find η_d. Therefore, we only need to consider $\mathsf{gcd}(\zeta_d - 1, N)$ for $d = 3, 5, 7, 11, 47, 499$ for the CSIDH-512 parameter set.

As a consequence, we reduce each ζ_d-rGAIP instance to a GAIP instance with a group size determined by ζ_d. This is summarized in Table 1. For ζ_8, we have a chain $\{1\} = G_1 < G_2 < G_3 < G_4 < G_5 = \mathcal{C}\ell(\mathcal{O})$ where G_2, G_3, G_4 is of size $\mathsf{gcd}(\zeta_8 - 1, N)$, $\mathsf{gcd}(\zeta_8^2 - 1, N)$, $\mathsf{gcd}(\zeta_8^4 - 1, N)$, respectively, and the largest quotient group is $|G_2/G_1| \approx 2^{134}$, which demonstrates the invulnerability of ζ_8-rGAIP. For instance, for ζ_3 we have a chain $\{1\} = G_1 < G_2 < G_3 = \mathcal{C}\ell(\mathcal{O})$ where G_2 is of size 37 and the largest quotient group is $|G_3/G_2| \approx 2^{251}$. For ζ_4, ζ_{47} and ζ_{499} we have a chain $\{1\} = G_1 < G_2 < G_3 = \mathcal{C}\ell(\mathcal{O})$ where G_2 is of size 1407181 with the largest quotient group $|G_3/G_2| \approx 2^{236}$. Our cryptanalysis gives an upper bound of ζ_d-rGAIP from the perspective of GAIP. Importantly, ζ_4-rGAIP which we use for our optimized blind signature only seems to lose 2 bits of security compared with ζ_2-rGAIP, or equivalently, GAIP over CSIDH-512.

8.3 Equivalence between GAIP and rGAIP

We complement our cryptanalysis by showing that our attack is optimal for some parameters. Although a few instances of ζ_d-rGAIP were shown to be significantly weaker than the original GAIP over CSIDH-512, we present a surprising condition that allows to reduce ζ_d-rGAIP to the original GAIP. This shows that the attack in Table 1 is optimal for those specific choices of ζ_d. We note that though the condition does not cover all cases (including ζ_4 which meets Requirement 1), the result gives us some guidance of the hardness of ζ_d-rGAIP.

8.3.1 Large $\mathsf{gcd}(\zeta_d - 1, N) \approx N$

Note first that in this case we do not know how to have an efficient extractor in our optimized sigma protocol due to the large value of η_d (see Lemma 19). Requirement 1 is not satisfied.

It is clear that GAIP is never easier than ζ_d-rGAIP. The key insight of the reverse reduction is that when $\mathsf{gcd}(\zeta_d - 1, N) \approx N$ (or $\mathsf{gcd}(\zeta_d - 1, N) = N/\mathsf{poly}$ to be precise), given a GAIP instance we can generate a ζ_d-rGAIP instance by trial and error. Additionally, the success rate can also be amplified by repeatedly invoking the ζ_d-rGAIP oracle and testing the correctness.

Concretely, given $X_0 = [\mathfrak{g}^a] * E_0$ and access to an ζ_d-rGAIP adversary $\mathcal{A}$ for a d-th root of unity ζ_d, we can construct a GAIP adversary $\mathcal{B}$ which invokes $\mathcal{A}$ on input $(X_0, [a'] * X_0, [a'^{\zeta_d}] * X_0, \dots, [a'^{\zeta_d^{d-1}}] * X_0)$ where a' is sampled uniformly at random from the subgroup $\{r^{\zeta_d - 1} | r \in \mathcal{C}\ell(\mathcal{O})\}$. Then, $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs. Since the subgroup is of size $N/\mathsf{gcd}(\zeta_d - 1, N) = ploy(n)$, the adversary $\mathcal{B}$ invokes $\mathcal{A}$ on a well-formed instance with probability $\mathsf{gcd}(\zeta_d - 1, N)/N$, which is non-negligible. We thus obtain the following theorem.

Theorem 29 *Given any ζ_d-rGAIP adversary $\mathcal{A}$ for a known-order effective group action of the group size N, there exists a GAIP adversary $\mathcal{B}$ in time d over the same action such that* $\mathsf{Adv}^{\zeta_d\text{-rGAIP}}(\mathcal{A}) \leq \frac{N}{\mathsf{gcd}(\zeta_d - 1, N)} \cdot \mathsf{Adv}^{\mathsf{GAIP}}(\mathcal{B})$.

As a consequence, we know that for CSIDH-512 we have $\zeta_3, \zeta_9, \zeta_5, \zeta_{47}, \zeta_{499}$-rGAIPs are as hard as the original GAIP with a reduction loss of factors 37, 37, 1407181, 1407181, 1407181 respectively. Similarly, $\zeta_{117265} = \zeta_5 \zeta_{47} \zeta_{499}$ also has a reduction loss of a factor 1407181.

9 Performance

We present an overall performance in Table 2 for our protocols instantiated using CSIDH-512. As explained in Sect. 8, we instantiate the ζ_d-rGAIP assumption with the 4-th root of unity ζ_4 as it is the only parameter that satisfies Requirement 1 while being presumably as hard as GAIP over CSIDH-512. We also analyze the trade-off between our basic blind signature in Sect. 5 and the optimized blind signature using a d-th primitive root of unity in Sect. 7. This helps us illustrate the effect of the value d on our optimized scheme and may be useful in the future when new group actions where ζ_d-rGAIP is hard are discovered.

The public key is d times larger compared to the basic scheme in general, which can be halved when d is even and $\zeta^{\frac{d}{2}} = -1$. Let $w = \log_2(N)/8$ denote the byte size of a class group element in $\mathbb{Z}_N$ and approximately $2w$ for one elliptic curve in $\mathcal{E}\ell\ell$; for example $w \approx 32$ for a CSIDH-512 group. In Sect. 5, the sender and user bandwidths and the signature size of the basic blind signature are $4wn$ B, $n/8$ B (i.e., one hash), and $2n(w + n/8)$ B, respectively. On the other hand, in Sect. 7 the sender and user bandwidths and the signature size of the optimized blind signature are $2\kappa(wd + w + \log_2 d)$ B, $(\kappa \log_2 d)/8$ B, and $2\kappa(w + \log_2 d)$ B, respectively. Now, given the security parameter n, the number of repetitions κ with a d-th primitive root of unity is required to satisfy $d^\kappa = 2^n$, i.e., $n = \kappa \log_2 d$. Therefore, the communication cost of the signer is increased by roughly $\frac{d\kappa}{2n}$, while the signature is decreased by roughly $\frac{n}{\kappa}$. The computation cost is increased by a factor of $\frac{d\kappa}{2n}$ in group action evaluations for both the signer and the user. Concretely, when $d = 4$, we have $n = 2\kappa$ and thus the signature size is reduced by approximately 50%.

It takes roughly 40 ms to perform an action on a 2.70 GHz processor [12, 24], and we can estimate the running time in terms of the number of the isogeny action. Since the signing

Table 2 The overall performance of our blind signature family regarding the bandwidth, the secret key size, the public size, and the signature size using CSIDH-512

	Bandwidth.S	Bandwidth.U	$\|\mathsf{sk}\|$	$\|\mathsf{pk}\|$	$\|\sigma\|$	Assumption
Basic. (Fig. 3)	16 KB	16 B	16 B	128 B	8 KB	GAIP
Fig. 9 with ζ_4	64 KB	16 B	16 B	512 B	4 KB	ζ_4-rGAIP
PBS. (Fig. 6)	48 KB	16 B	16 B	128 B	24 KB	GAIP

We take $n = 128$ and sk is generated by a seed of n bits. The first two rows are our blind signatures and the final row is our (unoptimized) partially blind signature

(respectively, verifying) process requires 6×128 (respectively, 2×128) actions in Sect. 5, it takes 30 s (respectively, 10 s) for the procedure.

Acknowledgements Shuichi Katsumata was partially supported by JST, CREST Grant Number JPMJCR22M1 and by JST, AIP Acceleration Research JPMJCR22U5. Yi-Fu Lai is supported by the European Union (ERC AdG REWORC - 101054911). Yi-Fu Lai, Jason T. LeGrow, and Ling Qin were supported in part by the Ministry for Business, Innovation and Employment of New Zealand. Jason T. LeGrow was supported in part by the Commonwealth of Virginia's Commonwealth Cyber Initiative (CCI), an investment in the advancement of cyber R&D, innovation, and workforce development. For more information about CCI, visit url: www.cyberinitiative.org.

Data availability Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

Declarations

Conflict of interest The authors declare that they have no financial or non-financial interests.

References

1. Abdalla M., Eisenhofer T., Kiltz E., Kunzweiler S., Riepel D.: Password-authenticated key exchange from group actions. In: Dodis Y., Shrimpton T., et al. (eds.) CRYPTO 2022, Part II. LNCS, vol. 13508, pp. 699–728. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15979-4_24.
2. Abe M.: A secure three-move blind signature scheme for polynomially many signatures. In: Pfitzmann B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 136–151. Springer, Cham (2001).https://doi.org/10.1007/3-540-44987-6_9.
3. Abe M., Fujisaki E.: How to date blind signatures. In: Kim K., Matsumoto T. (eds.) ASIACRYPT'96. LNCS, vol. 1163, pp. 244–251. Springer, New York (1996).https://doi.org/10.1007/BFb0034851.
4. Abe M., Okamoto T.: Provably secure partially blind signatures. In: Bellare M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 271–286. Springer, Berlin (2000). https://doi.org/10.1007/3-540-44598-6_17.
5. Agrawal S., Kirshanova E., Stehlé D., Yadav A.: Practical, round-optimal lattice-based blind signatures. In: Yin H., Stavrou A., Cremers C., Shi E. (eds.) ACM CCS 2022, pp. 39–53. ACM Press, New York (2022).https://doi.org/10.1145/3548606.3560650.

6. Alamati N., De Feo L., Montgomery H., Patranabis S.: Cryptographic group actions and applications. In: Moriai S., Wang H. (eds.) ASIACRYPT 2020, Part II. LNCS, vol. 12492, pp. 411–439. Springer, Berlin (2020). https://doi.org/10.1007/978-3-030-64834-3_14.
7. Alkeilani Alkadri N., El Bansarkhani R., Buchmann J.: BLAZE: practical lattice-based blind signatures for privacy-preserving applications. In: Bonneau J., Heninger N. (eds.) FC 2020. LNCS, vol. 12059, pp. 484–502. Springer, Berlin (2020).https://doi.org/10.1007/978-3-030-51280-4_26.
8. Alkeilani Alkadri N., El Bansarkhani R., Buchmann J.: On lattice-based interactive protocols: an approach with less or no aborts. In: Liu J.K., Cui H. (eds.) ACISP 20. LNCS, vol. 12248, pp. 41–61. Springer, Berlin (2020).https://doi.org/10.1007/978-3-030-55304-3_3.
9. Alkeilani Alkadri N., Harasser P., Janson C.: BlindOR: an efficient lattice-based blind signature scheme from OR-proofs. In: Conti M., Stevens M., Krenn S. (eds.) CANS 21. LNCS, vol. 13099, pp. 95–115. Springer, Berlin (2021).https://doi.org/10.1007/978-3-030-92548-2_6.
10. Azarderakhsh R., Jao D., Koziel B., LeGrow J.T., Soukharev V., Taraskin O.: How not to create an isogeny-based PAKE. In: Conti M., Zhou J., Casalicchio E., Spognardi A. (eds.) ACNS 20, Part I. LNCS, vol. 12146, pp. 169–186. Springer, Berlin (2020).https://doi.org/10.1007/978-3-030-57808-4_9.
11. Baldimtsi F., Lysyanskaya A.: On the security of one-witness blind signature schemes. In: Sako K., Sarkar P. (eds.) ASIACRYPT 2013, Part II. LNCS, vol. 8270, pp. 82–99. Springer, Berlin (2013).https://doi.org/10.1007/978-3-642-42045-0_5.
12. Beullens W., Kleinjung T., Vercauteren F.: CSI-FiSh: efficient isogeny based signatures through class group computations. In: Galbraith S.D., Moriai S. (eds.) ASIACRYPT 2019, Part I. LNCS, vol. 11921, pp. 227–247. Springer, Berlin (2019).https://doi.org/10.1007/978-3-030-34578-5_9.
13. Beullens W., Katsumata S., Pintore F.: Calamari and Falafl: logarithmic (linkable) ring signatures from isogenies and lattices. In: Moriai S., Wang H. (eds.) ASIACRYPT 2020, Part II. LNCS, vol. 12492, pp. 464–492. Springer, Berlin (2020).https://doi.org/10.1007/978-3-030-64834-3_16.
14. Beullens W., Dobson S., Katsumata S., Lai Y.-F., Pintore F.: Group signatures and more from isogenies and lattices: generic, simple, and efficient. In: Dunkelman O., Dziembowski S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 95–126. Springer, Berlin (2022).https://doi.org/10.1007/978-3-031-07085-3_4.
15. Beullens W., Lyubashevsky V., Nguyen N.K., Seiler G.: Lattice-based blind signatures: short, efficient, and round-optimal. Cryptology ePrint Archive, Paper 2023/077. https://eprint.iacr.org/2023/077 (2023).
16. Biasse J.-F., Iezzi A., Jacobson M.J. Jr.: A note on the security of CSIDH. In: Chakraborty D., Iwata T. (eds.) INDOCRYPT 2018. LNCS, vol. 11356, pp. 153–168. Springer, Berlin (2018).https://doi.org/10.1007/978-3-030-05378-9_9.
17. Blazy O., Gaborit P., Schrek J., Sendrier N.: A code-based blind signature. In: 2017 IEEE International Symposium on Information Theory (ISIT), pp. 2718–2722 (2017). IEEE.
18. Bonnetain X., Naya-Plasencia M.: Hidden shift quantum cryptanalysis and implications. In: Peyrin T., Galbraith S. (eds.) ASIACRYPT 2018, Part I. LNCS, vol. 11272, pp. 560–592. Springer, Berlin (2018).https://doi.org/10.1007/978-3-030-03326-2_19.
19. Bonnetain X., Schrottenloher A.: Quantum security analysis of CSIDH. In: Canteaut A., Ishai Y. (eds.) EUROCRYPT 2020, Part II. LNCS, vol. 12106, pp. 493–522. Springer, Berlin (2020).https://doi.org/10.1007/978-3-030-45724-2_17.
20. Brands S.: Untraceable off-line cash in wallets with observers (extended abstract). In: Stinson D.R. (ed.) CRYPTO'93. LNCS, vol. 773, pp. 302–318. Springer, Berlin (1994).https://doi.org/10.1007/3-540-48329-2_26.
21. Buser M., Dowsley R., Esgin M., Gritti C., Kasra K.S., Kuchta V., LeGrow J., Liu J., Phan R., Sakzad A.: A survey on exotic signatures for post-quantum blockchain: challenges and research directions. ACM Comput. Surv. **55**(12), 1–32 (2023).
22. Camenisch J., Lysyanskaya A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Pfitzmann B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer, Berlin (2001).https://doi.org/10.1007/3-540-44987-6_7.
23. Castryck W., Decru T.: An efficient key recovery attack on SIDH. In: Hazay C., Stam M. (eds.) EUROCRYPT 2023, Part V. LNCS, vol. 14008, pp. 423–447. Springer, Cham (2023).https://doi.org/10.1007/978-3-031-30589-4_15.
24. Castryck W., Lange T., Martindale C., Panny L., Renes J.: CSIDH: an efficient post-quantum commutative group action. In: Peyrin T., Galbraith S. (eds.) ASIACRYPT 2018, Part III. LNCS, vol. 11274, pp. 395–427. Springer, Berlin (2018).https://doi.org/10.1007/978-3-030-03332-3_15.
25. Castryck W., Dooms A., Emerencia C., Lemmens A.: A fusion algorithm for solving the hidden shift problem in finite abelian groups. In: Cheon J.H., Tillich J.-P. (eds.) Post-Quantum Cryptography-12th International Workshop, PQCrypto 2021, pp. 133–153. Springer (2021).https://doi.org/10.1007/978-3-030-81293-5_8.

26. Charles D.X., Lauter K.E., Goren E.Z.: Cryptographic hash functions from expander graphs. J. Cryptol. **22**(1), 93–113 (2009). https://doi.org/10.1007/s00145-007-9002-x.
27. Chaum D.: Blind signatures for untraceable payments. In: Chaum D., Rivest R.L., Sherman A.T. (eds.) CRYPTO'82, pp. 199–203. Plenum Press, New York (1982).
28. Chaum D.: Elections with unconditionally-secret ballots and disruption equivalent to breaking RSA. In: Günther C.G. (ed.) EUROCRYPT'88. LNCS, vol. 330, pp. 177–182. Springer, Berlin (1988). https://doi.org/10.1007/3-540-45961-8_15.
29. Chaum D., Pedersen T.P.: Wallet databases with observers. In: Brickell E.F. (ed.) CRYPTO'92. LNCS, vol. 740, pp. 89–105. Springer, Berlin (1993).https://doi.org/10.1007/3-540-48071-4_7.
30. Chaum D., Fiat A., Naor M.: Untraceable electronic cash. In: Goldwasser S. (ed.) CRYPTO'88, vol. 403, pp. 319–327. LNCS. Springer, Berlin (1990).
31. Childs A., Jao D., Soukharev V.: Constructing elliptic curve isogenies in quantum subexponential time. J. Math. Cryptol. **8**(1), 1–29 (2014). https://doi.org/10.1515/jmc-2012-0016.
32. Couveignes J.-M.: Hard Homogeneous Spaces. Cryptology ePrint Archive, Report 2006/291. https://eprint.iacr.org/2006/291 (2006).
33. Cramer R., Damgård I., Schoenmakers B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt Y. (ed.) CRYPTO'94, vol. 839, pp. 174–187. LNCS. Springer, Berlin (1994). https://doi.org/10.1007/3-540-48658-5_19.
34. De Feo L.: SeaSign: Compact Isogeny Signatures from Class Group Actions. Talk at Eurocrypt 2019 (2019). http://defeo.lu/docet/assets/slides/2019-05-23-eurocrypt.pdf.
35. De Feo L., Galbraith S.D.: SeaSign: compact isogeny signatures from class group actions. In: Ishai Y., Rijmen V. (eds.) EUROCRYPT 2019, Part III. LNCS, vol. 11478, pp. 759–789. Springer, Cham (2019).https://doi.org/10.1007/978-3-030-17659-4_26.
36. De Feo L., Kohel D., Leroux A., Petit C., Wesolowski B.: SQISign: compact post-quantum signatures from quaternions and isogenies. In: Moriai S., Wang H. (eds.) ASIACRYPT 2020, Part I. LNCS, vol. 12491, pp. 64–93. Springer, Berlin (2020).https://doi.org/10.1007/978-3-030-64837-4_3.
37. del Pino R., Katsumata S.: A new framework for more efficient round-optimal lattice-based (partially) blind signature via trapdoor sampling. In: Dodis Y., Shrimpton T. (eds.) CRYPTO 2022, Part II. LNCS, vol. 13508, pp. 306–336. Springer, Berlin (2022).https://doi.org/10.1007/978-3-031-15979-4_11.
38. Dobson S., Galbraith S.D., LeGrow J., Ti Y.B., Zobernig L.: An adaptive attack on 2-sidh. Int. J. Comput. Math. **5**(4), 282–299 (2020). https://doi.org/10.1080/23799927.2020.1822446.
39. Feo L.D., Fouotsa T.B., Kutas P., Leroux A., Merz S.-P., Panny L., Wesolowski B.: SCALLOP: scaling the CSI-FiSh. Cryptology ePrint Archive, Paper 2023/058. https://eprint.iacr.org/2023/058 (2023).
40. Fiat A., Shamir A.: How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 186–194. Springer, Berlin (1987). https://doi.org/10.1007/3-540-47721-7_12.
41. Fischlin M.: Round-optimal composable blind signatures in the common reference string model. In: Dwork C. (ed.) CRYPTO 2006, vol. 4117, pp. 60–77. LNCS. Springer, Berlin (2006).
42. Fouotsa T.B., Moriya T., Petit C.: M-SIDH and MD-SIDH: countering SIDH attacks by masking information. In: Hazay C., Stam M. (eds.) EUROCRYPT 2023, Part V. LNCS, vol. 14008, pp. 282–309. Springer, Berlin (2023).https://doi.org/10.1007/978-3-031-30589-4_10.
43. Friedl K., Ivanyos G., Magniez F., Santha M., Sen P.: Hidden translation and translating coset in quantum computing. SIAM J. Comput. **43**(1), 1–24 (2014).
44. Fujioka A., Okamoto T., Ohta K.: A practical secret voting scheme for large scale elections. In: AUSCRYPT, pp. 244–251 (1992). Springer.
45. Galbraith S.D., Lai Y.-F.: Attack on sheals and heals: The second wave of gpst. In: Post-Quantum Cryptography: 13th International Workshop, PQCrypto 2022, Virtual Event, September 28–30, 2022, Proceedings, pp. 399–421 (2022). Springer.
46. Galbraith S., Stolbunov A.: Improved algorithm for the isogeny problem for ordinary elliptic curves. Appl. Algebra Eng. Commun. Comput. **24**(2), 107–131 (2013).
47. Galbraith S.D., Hess F., Smart N.P.: Extending the GHS Weil descent attack. In: Knudsen L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 29–44. Springer (2002).https://doi.org/10.1007/3-540-46035-7_3.
48. Galbraith S.D., Petit C., Shani B., Ti Y.B.: On the security of supersingular isogeny cryptosystems. In: Cheon J.H., Takagi T. (eds.) ASIACRYPT 2016, Part I. LNCS, vol. 10031, pp. 63–91. Springer (2016).https://doi.org/10.1007/978-3-662-53887-6_3.
49. Hauck E., Kiltz E., Loss J.: A modular treatment of blind signatures from identification schemes. In: Ishai Y., Rijmen V. (eds.) EUROCRYPT 2019, Part III. LNCS, vol. 11478, pp. 345–375. Springer, (2019).https://doi.org/10.1007/978-3-030-17659-4_12.

50. Hauck E., Kiltz E., Loss J., Nguyen N.K.: Lattice-based blind signatures, revisited. In: Micciancio D., Ristenpart T. (eds.) CRYPTO 2020, Part II. LNCS, vol. 12171, pp. 500–529. Springer (2020).https://doi.org/10.1007/978-3-030-56880-1_18.
51. Hendrickson S., Iyengar J., Pauly T., Valdez S., Wood C.A.: Private Access Tokens. Internet-Draft draft-private-access-tokens-01. Internet Engineering Task Force. Work in Progress (2022). https://datatracker.ietf.org/doc/draft-private-access-tokens/.
52. Jao D., Azarderakhsh R., Campagna M., Costello C., De Feo L., Hess B., Jalali A., Koziel B., LaMacchia B., Longa P., Naehrig M., Renes J., Soukharev V., Urbanik D., Pereira G., Karabina K., Hutchinson A.: Supersingular isogeny key encapsulation. Technical report, National Institute of Standards and Technology (2017).
53. Jao D., LeGrow J., Leonardi C., Ruiz-Lopez L.: A subexponential-time, polynomial quantum space algorithm for inverting the cm group action. J. Math. Cryptol. **14**(1), 129–138 (2020). https://doi.org/10.1515/jmc-2015-0057.
54. Kastner J., Loss J., Xu J.: On pairing-free blind signature schemes in the algebraic group model. In: PKC, pp. 468–497 (2022). Springer.
55. Kastner J., Loss J., Xu J.: The Abe-Okamoto partially blind signature scheme revisited. In: Agrawal S., Lin D. (eds.) ASIACRYPT 2022, Part IV. LNCS, vol. 13794, pp. 279–309. Springer (2022). https://doi.org/10.1007/978-3-031-22972-5_10.
56. Katsumata S., Lai Y.-F., LeGrow J.T., Qin L.: CSI -otter: Isogeny-based (partially) blind signatures from the class group action with a twist. In: CRYPTO 2023, Part III. LNCS, pp. 729–761. Springer (2023).https://doi.org/10.1007/978-3-031-38548-3_24.
57. Katsumata S., Lai Y.-F., Reichle M.: Breaking Parallel ROS: Implication for Isogeny and Lattice-based Blind Signatures. Cryptology ePrint Archive, Paper 2023/1603. https://eprint.iacr.org/2023/1603 (2023).
58. Kuperberg G.: A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. SIAM J. Comput. **35**(1), 170–188 (2005). https://doi.org/10.1137/S0097539703436345.
59. Kuperberg G.: Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem. arXiv:1112.3333 (2011).
60. Lai Y.-F., Galbraith S.D., Delpech de Saint Guilhem C.: Compact, efficient and UC-secure isogeny-based oblivious transfer. In: Canteaut A., Standaert F.-X. (eds.) EUROCRYPT 2021, Part I. LNCS, vol. 12696, pp. 213–241. Springer (2021). https://doi.org/10.1007/978-3-030-77870-5_8.
61. Lai Y.-F.: CAPYBARA and TSUBAKI: verifiable random functions from group actions and isogenies. Cryptology ePrint Archive, Report 2023/182. https://eprint.iacr.org/2023/182 (2023).
62. Le H.Q., Susilo W., Khuc T.X., Bui M.K., Duong D.H.: A blind signature from module latices. In: Dependable and Secure Computing (DSC), pp. 1–8 (2019). IEEE.
63. LeGrow J.T.: A faster method for fault attack resistance in static/ephemeral CSIDH. J. Cryptogr. Eng. pp. 1–12 (2023).
64. Lyubashevsky V., Nguyen N.K., Plançon M.: Efficient lattice-based blind signatures via gaussian one-time signatures. In: Hanaoka G., Shikata J., Watanabe Y. (eds.) PKC 2022, Part II. LNCS, vol. 13178, pp. 498–527. Springer (2022).https://doi.org/10.1007/978-3-030-97131-1_17.
65. Maino L., Martindale C., Panny L., Pope G., Wesolowski B.: A direct key recovery attack on SIDH. In: Hazay C., Stam M. (eds.) EUROCRYPT 2023, Part V. LNCS, vol. 14008, pp. 448–471. Springer, (2023).https://doi.org/10.1007/978-3-031-30589-4_16.
66. Okamoto T., Ohta K.: Universal electronic cash. In: Feigenbaum J. (ed.) CRYPTO'91. LNCS, vol. 576, pp. 324–337. Springer (1992).https://doi.org/10.1007/3-540-46766-1_27.
67. Papachristoudis D., Hristu-Varsakelis D., Baldimtsi F., Stephanides G.: Leakage-Resilient Lattice-Based Partially Blind Signatures. Cryptology ePrint Archive, Report 2019/1452. https://eprint.iacr.org/2019/1452 (2019).
68. Peikert C.: He gives C-sieves on the CSIDH. In: Canteaut A., Ishai Y. (eds.) EUROCRYPT 2020, Part II. LNCS, vol. 12106, pp. 463–492. Springer (2020).https://doi.org/10.1007/978-3-030-45724-2_16.
69. Petit C.: Faster algorithms for isogeny problems using torsion point images. In: Takagi T., Peyrin T. (eds.) ASIACRYPT 2017, Part II. LNCS, vol. 10625, pp. 330–353. Springer (2017).https://doi.org/10.1007/978-3-319-70697-9_12.
70. Petzoldt A., Szepieniec A., Mohamed M.S.E.: A practical multivariate blind signature scheme. In: Kiayias A. (ed.) FC 2017. LNCS, vol. 10322, pp. 437–454. Springer (2017).
71. Pointcheval D., Stern J.: Security proofs for signature schemes. In: Maurer U.M. (ed.) EUROCRYPT'96. LNCS, vol. 1070, pp. 387–398. Springer (1996).https://doi.org/10.1007/3-540-68339-9_33.
72. Pointcheval D., Stern J.: Security arguments for digital signatures and blind signatures. J. Cryptol. **13**(3), 361–396 (2000). https://doi.org/10.1007/s001450010003.

73. Quehen V., Kutas P., Leonardi C., Martindale C., Panny L., Petit C., Stange K.E.: Improved torsion-point attacks on SIDH variants. In: Malkin T., Peikert C. (eds.) CRYPTO 2021, Part III. LNCS, vol. 12827, pp. 432–470. Springer, Virtual Event (2021). https://doi.org/10.1007/978-3-030-84252-9_15.
74. Regev O.: A subexponential time algorithm for the dihedral hidden subgroup problem with polynomial space. (2004).
75. Robert D.: Breaking SIDH in polynomial time. In: Hazay C., Stam M. (eds.) EUROCRYPT 2023, Part V. LNCS, vol. 14008, pp. 472–503. Springer (2023).https://doi.org/10.1007/978-3-031-30589-4_17.
76. Rostovtsev A., Stolbunov A.: Public-key cryptosystem based on isogenies. Cryptology ePrint Archive, Report 2006/145. https://eprint.iacr.org/2006/145 (2006).
77. Rückert M.: Lattice-based blind signatures. In: Abe M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 413–430. Springer (2010).https://doi.org/10.1007/978-3-642-17373-8_24.
78. Schnorr C.-P.: Efficient identification and signatures for smart cards. In: Brassard G. (ed.) CRYPTO'89. LNCS, vol. 435, pp. 239–252. Springer (1990).https://doi.org/10.1007/0-387-34805-0_22.
79. Schnorr C.-P.: Security of blind discrete log signatures against interactive attacks. In: Qing S., Okamoto T., Zhou J. (eds.) ICICS 01. LNCS, vol. 2229, pp. 1–12. Springer (2001).
80. Schoof R.: Counting points on elliptic curves over finite fields. Journal de théorie des nombres de Bordeaux **7**(1), 219–254 (1995).
81. Shamir A.: How to share a secret. Commun. ACM **22**(11), 612–613 (1979).
82. Stolbunov A.: Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves. Adv. Math. Commun. **4**(2), 215–235 (2010). https://doi.org/10.3934/amc.2010.4.215.
83. Taraskin O., Soukharev V., Jao D., LeGrow J.T.: Towards isogeny-based password-authenticated key establishment. J. Math. Cryptol. **15**(1), 18–30 (2021). https://doi.org/10.1515/jmc-2020-0071.
84. VPN by Google One, explained. https://one.google.com/about/vpn/howitworks. (2022).
85. Yi X., Lam K.-Y.: A new blind ECDSA scheme for bitcoin transaction anonymity. In: Galbraith S.D., Russello G., Susilo W., Gollmann D., Kirda E., Liang Z. (eds.) ASIACCS 19, pp. 613–620. ACM Press (2019). https://doi.org/10.1145/3321705.3329816.

Authors and Affiliations

Shuichi Katsumata[1,2] · **Yi-Fu Lai**[3,4] · **Jason T. LeGrow**[5] · **Ling Qin**[3]

✉ Shuichi Katsumata
shuichi.katsumata@pqshield.com

✉ Yi-Fu Lai
yi-fu.lai@ruhr-uni-bochum.de

✉ Jason T. LeGrow
jlegrow@vt.edu

Ling Qin
lqin276@aucklanduni.ac.nz

1 PQShield, Ltd., 267 Banbury Road, Oxford OX2 7HQ, UK

2 AIST, 2-3-26, Aomi, Koto-Ku, Tokyo 135-0064, Japan

3 Department of Mathematics, University of Auckland, 38 Princes Street, Auckland 1010, New Zealand

4 Department of Computer Science, Ruhr-University Bochum, Universitätsstraße 150, 44801 Bochum, Germany

5 Department of Mathematics, Virginia Polytechnic Institute and State University, 225 Stanger Street, Blacksburg, VA 24061, USA