Lattice-Based Blind Signatures: Short, Efficient, and Round-Optimal

Ward Beullens[1], Vadim Lyubashevsky[1], Ngoc Khanh Nguyen[2], and Gregor Seiler[1]

[1] IBM Research Europe, Zurich
[2] EPFL, Lausanne

Abstract. We give a construction of a 2-round blind signature scheme based on the hardness of standard lattice problems (Ring/Module-SIS/LWE and NTRU) with a signature size of 22 KB. The protocol is round-optimal and has a transcript size that can be as small as 60 KB. This blind signature is around 4 times shorter than the most compact lattice-based scheme based on standard assumptions of del Pino and Katsumata (Crypto 2022) and around 2 times shorter than the scheme of Agrawal et al. (CCS 2022) based on their newly-proposed one-more-ISIS assumption. We also give a construction of a "keyed-verification" blind signature scheme in which the verifier and the signer need to share a secret key. The signature size in this case is only 48 bytes, but more work needs to be done to explore the efficiency of the protocol which generates the signature.

1 Introduction

Blind signatures, protocols in which a user obtains a signature of a message from a signer without the latter learning anything about the message, are one of the foundational primitives of privacy-based cryptography [Cha82]. Their importance is steadily increasing due to the emergence of digital currency and the population's growing privacy expectations. There have been many practical instantiations of this primitive from various assumptions, and there are currently concrete proposals to deploy these primitives on national levels to create a central bank digital currency (e.g. [CM22]).

Another important recent development is the ongoing transition of cryptography based on assumptions such as factoring and discrete log to cryptography based on conjectured quantum-safe assumptions. With the 6 year NIST post-quantum standardization effort having recently produced four public-key schemes (one encryption and three digital signature) to be standardized in the next year [NIS22], the NSA is now mandating that a full transition to quantum-safe cryptography must occur by the beginning of the 2030s [NSA22]. It is quite probable that this timeline will closely coincide with the large-scale deployment of blind signatures and other forthcoming privacy-based cryptographic schemes; which strongly implies that these will therefore need to be quantum-safe as well. But while there are many very efficient blind signatures based on classical assumptions, there has been a dearth of research on quantum-safe blind signatures.[3]

The state of affairs has been recently improved with two proposals for lattice-based blind signatures which, while less efficient than their classical counterparts, are certainly in the realm of practicality. The scheme of [dPK22] is a blind signature based on standard lattice assumptions[4] with signature sizes being around 100 KB and total communication being under a megabyte. The scheme in [AKSY22] is based on the new one-more-ISIS assumption proposed in that paper, with the advantage being that the scheme is more efficient, being around 45KB in size and also requiring less communication to create a signature.

In our work, we improve on this state of the art by presenting a practical blind signature based on standard lattice assumptions which shrinks the signature size down to approximately 22 KB. We also propose a *keyed-verification* blind signature scheme in which the signature is only 48 bytes, but the downsides of the latter are that the generation of the signature is less efficient and verification requires the signer's secret key.

[3] The "quantum-safe" in the title of [CM22] only refers to a particular security property – the overall scheme, however, is not quantum-safe because it is based on RSA.

[4] All the lattice assumptions in the papers we discuss and those in the current paper are over lattices with polynomial structure.

Our schemes are round-optimal, meaning that the signature is generated after 2 rounds. Round-optimality is useful in practice because communication latency is often a major bottleneck in interactive protocols. It can be additionally useful because it allows off-line pre-computation of an entire part of the protocol for one of the parties, which could allow even a fairly low-powered user to participate in the protocol. Moreover, 2-round blind signatures need not worry about delicate issues involving concurrent executions of the signing protocol, which can complicate the design of multi-round blind signatures and have led to concrete attacks in the past [BLL+21].

Scheme	Assumption	Signature Size
[AKSY22]	one-more-ISIS + LWE + NTRU	45 KB
[dPK22]	SIS + LWE	100 KB
Blind Signature (This work)	SIS + LWE + NTRU	22 KB
Keyed-Verification Blind Signature (This Work)	SIS + LWE	48 B

Table 1: Potentially Practical Round-Optimal Lattice-Based Blind Signatures. All the assumptions are for the versions of the specified lattice problems over polynomial rings.

Below we give a detailed sketch of our constructions. We will assume that the reader is somewhat familiar with basic lattice cryptography.

1.1 Blind Signature

The starting point of our construction is the generic blind signature construction of Fischlin [Fis06] and its more efficient adaptation for lattices, based on the one-more-ISIS assumption by Agrawal et al. [AKSY22]. The public keys in [AKSY22] consist of polynomial matrices $\mathbf{A}$ and $\mathbf{B}$, where $\mathbf{A}$ is created together with a trapdoor that allows for efficient pre-image sampling of short $\vec{\mathbf{s}}$ satisfying $\mathbf{A}\vec{\mathbf{s}} = \vec{\mathbf{t}}$ for arbitrary $\vec{\mathbf{t}}$, and $\mathbf{B}$ is a random CRS. The aforementioned trapdoor serves as the signer's secret key. The scheme also uses a cryptographic hash function $\mathsf{H}(\cdot)$ modeled as a random oracle.

The user, who wants to get μ signed, commits to $\mathsf{H}(\mu)$ using randomness $\vec{\mathbf{r}}$, which is a vector of small norm, as

$$\vec{\mathbf{c}} := \mathbf{B}\vec{\mathbf{r}} + \mathsf{H}(\mu) \tag{1}$$

and send $\vec{\mathbf{c}}$ to the signer.[5] The signer then proceeds to sign this commitment using a lattice-based signature scheme based on (the soon-to-be NIST standardized) FALCON [PFH+17]. In particular, the signer uses his secret trapdoor to create a short vector $\vec{\mathbf{s}}$ satisfying $\mathbf{A}\vec{\mathbf{s}} = \vec{\mathbf{c}}$ and sends $\vec{\mathbf{s}}$ to the user. In order to create a signature for μ, the user reveals μ and creates a NIZK proof π that proves knowledge of short $\vec{\mathbf{s}}, \vec{\mathbf{r}}$ satisfying

$$\mathbf{A}\vec{\mathbf{s}} = \mathbf{B}\vec{\mathbf{r}} + \mathsf{H}(\mu). \tag{2}$$

The NIZK proof π can be efficiently created using the recent linear-size zero-knowledge proofs from [LNP22].

The unforgeabilty property of blind signatures requires that an adversary who makes k signing queries should only be able to produce at most k valid signatures. A technical issue that arises in protocols that are constructed in the same vein as Fischlin's is that proving security requires us to use a successful adversary to obtain $k+1$ valid witnesses to some relation (in the case of the above scheme, $\vec{\mathbf{s}}$ and $\vec{\mathbf{r}}$). But because the adversary only gives a zero-knowledge proof, we can't immediately obtain these. One could consider extracting $k+1$ times using rewinding, but this would exponentially (in k) blow up the soundness error in the security proof. One therefore needs to

[5] All the operations in our paper are over the usual polynomial rings of the form $\mathbb{Z}_q[X]/(X^d+1)$.

modify the proof system by additionally creating an encryption to the witness and then producing a sound NIZK which includes the original statement as well as the fact that the encryption is of the witness.[6] The protocol of [AKSY22] thus also includes an encryption of the witness $\vec{\mathbf{r}}$, $\vec{\mathbf{s}}$ as part of the signature and this incurs approximately a factor 2 blow-up in the signature size.

We modify the Fischlin/Agrawal et al. construction in two ways in order to remove the new $\mathsf{one\text{-}more\text{-}ISIS}$ assumption and the ciphertext that's part of the signature. The need for the new assumption in [AKSY22] comes from the fact that the $\vec{\mathbf{c}}$ from (1) that the signer produces a pre-image for is not necessarily random because the user can first compute $\mathsf{H}(\mu)$ and then choose the $\vec{\mathbf{r}}$ conditioned on $\mathsf{H}(\mu)$ in some way so as to influence the distribution of $\vec{\mathbf{c}}$. The need for the $\mathsf{one\text{-}more\text{-}ISIS}$ [7] assumption comes exactly from the fact that the signer can be forced to produce pre-images of vectors that are non-random; whereas the GPV [GPV08] and FALCON [PFH$^+$17] signatures avoid this assumption by making sure that the signer only outputs short pre-images of random oracle outputs. In order to ensure that $\vec{\mathbf{c}}$ is truly random, we modify it to be

$$\vec{\mathbf{c}} = \mathbf{B}\vec{\mathbf{r}} + \mathsf{H}(\mu, \mathsf{H}(\vec{\mathbf{r}})) \tag{3}$$

and also include a NIZK proof that $\vec{\mathbf{c}}$ is correctly constructed. We also include an "encryption to the sky" of the witness $\vec{\mathbf{r}}$ and μ and include the proof of the encryption in the NIZK. The signature for μ is then modified to be $(\rho = \mathsf{H}(\vec{\mathbf{r}}), \pi)$, where π is a NIZK proof of knowledge of a short $\vec{\mathbf{s}}$ and $\vec{\mathbf{r}}$ satisfying

$$\mathbf{A}\vec{\mathbf{s}} = \mathbf{B}\vec{\mathbf{r}} + \mathsf{H}(\mu, \rho). \tag{4}$$

Importantly, there is no need to encrypt the witness $\vec{\mathbf{r}}$ and $\vec{\mathbf{s}}$ and, unlike in the proof of (3), the user does not need to prove that $\rho = \mathsf{H}(\vec{\mathbf{r}})$.

Efficiency. The proof π for (4) is constructed using the general technique from [LNP22] for proving simple lattice relations and we compute the proof size (and thus the size of the blind signature, since μ and ρ are just 32 bytes each) to be approximately 22 KB. Because the framework of [LNP22] is quite recent, there haven't been any implementations of it, and so we aren't able to provide the running time of the prover and the verifier. But earlier versions of lattice-based zero-knowledge proofs upon which [LNP22] is based had implementations in which the prover and verifier had running times on the order of milliseconds (c.f. [LNS20]) for proving similar relations.

The NIZK produced by the user in the first step (3) is proving the validity of cryptographic hash function inputs and outputs, which is computationally less efficient than the NIZK that just proves the simple lattice relations. Nevertheless, because this NIZK is not part of the signature, we do not need to overly optimize its size and there are various ways to make it rather efficient. The instantiation of our blind signature, which uses a FALCON-like domain (i.e. a ring $\mathbb{Z}_q[X]/(X^{512}+1)$ where q is less than 2^{13}), requires a hash function that outputs $13 * 512/8 = 832$ bytes, which is essentially 26 SHA outputs. Modern quantum-safe zero-knowledge proof systems, whose security relies only on properties of cryptographic hash functions, can construct such proofs in under 20 seconds and verify them in under 10 seconds (c.f. [BFH$^+$20, Table 1]). The proof sizes are several hundred kilobytes, but as previously mentioned, this is not part of the signature size. Additionally, because the proof is sent in the first message of the 2-round signing protocol, the user can compute it offline. There are other things besides hash functions that need to be proved in (3), but those are just basic lattice relations which, at worst, could be proved using the linear-sized proofs from [LNP22] at a cost of another few hundred kilobytes.

A promising recent approach which may produce even faster and shorter proofs for R1CS and lattice relations is the new sub-linear protocol LaBRADOR [BS22] which creates very short ($\approx$ 60KB) proofs for R1CS instances of reasonable size (up to 2^{30} constraints) based on the hardness of

[6] This is sometimes referred to as "encryption to the sky" because in the real protocol, decryption is not performed (in fact, the public key is usually not even validly constructed and is just a random string that is indistinguishable from an actual public key.

[7] This assumption roughly states that an algorithm that is given k pre-images on targets of his choice cannot produce pre-images of $k+1$ random targets.

standard lattice problems. Because lattice-based schemes are generally more efficient than hash-based ones (e.g. lattice-based signatures are several orders of magnitude faster than hash-based ones), one should expect that the running-time and memory consumption of lattice-based zero-knowledge proofs will also be considerably smaller. And finally, there are also proposals for replacing SHA as the cryptographic hash function in scenarios that require efficient zero-knowledge proofs, which should also result in a major speed-up (c.f. [BGL20]), and should take the prover time of (3) to well under a second. We leave the implementation of [LNP22, BS22] and their adaptations to our blind signature scheme as important future work.

Security. The NIZK proof of (3) produced by the user in the first move plays two roles. The first, as mentioned, it allows the security proof of the blind signature to be based on standard lattice problems. The intuition for this is that by putting $\vec{\mathbf{r}}$ as an input to $\mathsf{H}(\cdot)$, we force the adversary to pick the $\vec{\mathbf{r}}$ before querying the random oracle – thus the right-hand side of (3) is random and not under the control of the user. The second useful role is that because we additionally include an "encryption to the sky" of the witness $\mu, \vec{\mathbf{r}}$ during this move, one will not need any encryption to the sky as part of the signature output. The reason is that we only need to obtain *one* new signature from the adversary instead of $k+1$ because we can already obtain k valid signatures from the adversary's signing queries because we can obtain μ and $\vec{\mathbf{r}}$ via decryption. Thus all we need to do is use rewinding (or reprogramming the random oracle in the non-interactive setting when applying Fiat-Shamir) on one proof, which does not result in an exponential loss of soundness.

The security of our scheme is proved in two steps. First, we consider a regular (i.e. non-blind) signature scheme similar to GPV [GPV08] and [PFH$^+$17] in which the signature of a message μ consists of short vectors $\vec{\mathbf{r}}$, $\vec{\mathbf{s}}$ and a bit-string $\rho = \mathsf{H}(\vec{\mathbf{r}})$ satisfying $\mathbf{A}\vec{\mathbf{s}} = \mathbf{B}\vec{\mathbf{r}} + \mathsf{H}(\mu, \rho)$.[8] The verification only checks that the more "relaxed" condition from (4) is satisfied – that is, it does not additionally make sure that $\rho = \mathsf{H}(\vec{\mathbf{r}})$.

Using similar techniques, which are only slightly more complicated due to the nested random oracle, as showing that the GPV signature is based on SIS, it can be shown that breaking the above signature scheme is as hard as SIS – that is, finding a short non-zero vector $\vec{\mathbf{x}}$ satisfying $[\mathbf{A}||\mathbf{B}] \cdot \vec{\mathbf{x}} = \vec{\mathbf{0}}$. Given random $\mathbf{A}, \mathbf{B}$, the reduction publishes them as the public key. By the NTRU assumption, the random $\mathbf{A}$ is indistinguishable from one with an NTRU trapdoor. If the adversary ever queries $\mathsf{H}(\vec{\mathbf{r}})$ on a fresh $\vec{\mathbf{r}}$, the simulator assigns a random value as the output. If the adversary queries $\mathsf{H}(\mu, \rho)$, then the simulator checks whether ρ is one of the outputs for some previously-queried $\mathsf{H}(\vec{\mathbf{r}})$. If it's not, then he assigns it a random value (this query can never be used in a signature query because the adversary does not know the pre-image of ρ). If $\rho = \mathsf{H}(\vec{\mathbf{r}})$ for some previously-queried $\vec{\mathbf{r}}$, then the simulator chooses a short $\vec{\mathbf{s}}$ from the correct distribution (a Gaussian with a small standard deviation), computes $\mathbf{A}\vec{\mathbf{s}}$, and programs $\mathsf{H}(\mu, \rho) := \mathbf{A}\vec{\mathbf{s}} - \mathbf{B}\vec{\mathbf{r}}$. Because the entropy of $\vec{\mathbf{s}}$ is large enough, $\mathbf{A}\vec{\mathbf{s}}$ is uniformly random, and thus $\mathsf{H}(\mu, \rho)$ is random. If the adversary makes a signing query using a $\mu, \vec{\mathbf{r}}$ for which an $\vec{\mathbf{s}}$ was already chosen, the simulator outputs this $\vec{\mathbf{s}}$ as the signature. Otherwise, he creates the $\vec{\mathbf{s}}$ and the value for $\mathsf{H}(\mu, \mathsf{H}(\vec{\mathbf{r}}))$ as above, and outputs $\vec{\mathbf{s}}$.

When the adversary outputs a forgery $\mu, \rho, \vec{\mathbf{r}}, \vec{\mathbf{s}}$ satisfying (4), he necessarily needed to have previously queried $\mathsf{H}(\mu, \rho)$. This implies that the simulator already knows a tuple $(\vec{\mathbf{r}}', \vec{\mathbf{s}}')$ satisfying $\mathbf{A}\vec{\mathbf{s}}' = \mathbf{B}\vec{\mathbf{r}}' + \mathsf{H}(\mu, \rho)$. From this equation and the forgery satisfying (4), he obtains $\mathbf{A}(\vec{\mathbf{s}} - \vec{\mathbf{s}}') - \mathbf{B}(\vec{\mathbf{r}} - \vec{\mathbf{r}}') = \vec{\mathbf{0}}$. Since the $\vec{\mathbf{s}}'$ is unpredictable to the adversary, there is a high probability that $(\vec{\mathbf{s}} - \vec{\mathbf{s}}') \neq \vec{\mathbf{0}}$, and thus we obtain a short non-zero $\vec{\mathbf{x}}$ satisfying $[\mathbf{A}||\mathbf{B}] \cdot \vec{\mathbf{x}} = \vec{\mathbf{0}}$.

Having proved that the above signature scheme is secure in the random oracle model, we can assume that it is secure when H is instantiated with some concrete cryptographic hash function (e.g. SHA). We then use the assumption that the signature scheme with the concrete hash function is secure to prove the security of the blind signature. In particular, we assume that we have an adversary

[8] The GPV and FALCON schemes do not have the $\vec{\mathbf{r}}$ vector (or one can think of it as being $\vec{\mathbf{0}}$). It is only needed in our scheme because it serves as the commitment randomness in the first step.

who can forge a fresh message in the blind signature scheme, and use it to break the signature.[9] To simulate the signer, we obtain the first message of the adversarial user and, assuming that the NIZK is sound, we can decrypt the witness $\vec{\mathbf{r}}$ and μ. We then send this $\vec{\mathbf{r}}$ and μ to the signing oracle to obtain a signature $\vec{\mathbf{s}}$ and forward it to the adversary. If the adversary is to win the blind signature game, he needs to produce a signature of some message μ that he has not queried. If he outputs a μ and ρ and a NIZK proof of knowledge of an $\vec{\mathbf{r}}$ and $\vec{\mathbf{s}}$ satisfying (4), then we can extract this one witness $\vec{\mathbf{r}}, \vec{\mathbf{s}}$ and output $\mu, \vec{\rho}, \vec{\mathbf{r}}, \vec{\mathbf{s}}$ as a valid signature forgery.

1.2 Keyed-Verification Blind Signature

Keyed-verification anonymous credentials, introduced in [CMZ14], are more efficient anonymous credential schemes in the scenario where the credential issuer is also the verifier (or, more generally, the verifier must possess some secret key associated with the issuer). This naturally fits into the context of a blind signature scheme in the scenario where the entity issuing the signature is also the verifier. This could, for example, occur in an e-cash application where the merchant who is paid by the user does not do the check for double-spending and validity by himself, but instead sends the coin and its signature to the issuing bank in order to have it checked. While this scenario restricts who can do the verification, the advantage is that the resulting scheme can be more compact.

A Keyed-Verification Blind Signature scheme can be constructed generically from a *Verifiable Oblivious PRF* (VOPRF). A VOPRF is an interactive protocol between a server and a client. The server has a key K for some PRF F, and during the protocol, the client should learn $F(K, \mu)$ for a message μ of his choosing while learning nothing about K; and the server should similarly learn nothing about μ.

Converting a VOPRF into a Keyed-Verification Blind Signature is simple – the signer (server) and the user (client) execute a protocol in which the user learns $F(K, \mu)$, which is then the user's signature of μ. One can generically construct a quantum-resistant VOPRF using standard MPC techniques, but the resulting scheme will not be round-optimal (c.f. the discussion in [ADDS21, Appendix E].) There does, however, exist a construction of a round-optimal lattice-based VOPRF [ADDS21], which adapts the classical "hashed Diffie Hellman" technique (c.f. [CHL22] for a good survey) to lattices, and it serves as a starting point for our construction.

Because we need a blind signature rather than VOPRF, we are able to give a more efficient construction by relaxing what we need out of the VOPRF. In particular, instead of the client obtaining an evaluation of $F(K, \mu)$ for a message μ of his choosing, in our version he obtains $F(K, (\mu, \rho))$, where ρ is a randomly-generated value outside of his control. While this is not a VOPRF, it perfectly suffices for our blind signature because the user can simply output ρ along with μ.[10] Constructing this weakened version of a VOPRF turns out to be more efficient than the standard VOPRF construction from [ADDS21] mainly because it does not require "noise drowning" which is needed in [ADDS21] for the server's privacy.[11] Not needing to add large errors allows our scheme to have a smaller overall system modulus. This, in turn, allows to take a significantly smaller lattice dimension and should improve the running time of the scheme by at least an order of magnitude.

Our construction. The public key consists of a random square polynomial matrix $\mathbf{B}$ and a vector $\vec{\mathbf{t}}^T = \vec{\mathbf{s}}^T\mathbf{B} + \vec{\mathbf{e}}_1{}^T$ where $\vec{\mathbf{s}}$ and $\vec{\mathbf{e}}$ are low-norm polynomial vectors with $\vec{\mathbf{s}}$ being the secret key. The user, similar to (3), commits to his message μ as

$$\vec{\mathbf{c}} := \mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{e_2}} + \mathsf{H}\left(\mu, \mathsf{H}(\vec{\mathbf{r}}, \vec{\mathbf{e_2}})\right), \tag{5}$$

[9] The blindness of the scheme is easy to see because (3) is a hiding commitment and everything else is proved using zero-knowledge proofs.

[10] It would be interesting to see whether this weaker version of a VOPRF also has other applications.

[11] Noise drowning is a term for a technique used in lattice cryptography in which a very large random integer is added to some element drawn from a narrow distribution in order to make the resulting sum statistically close to the distribution of the large integer.

and creates a NIZK for the statement the commitment is of the right form. As in (3), putting the randomness into the hash ensures that the (uniform) distribution of $\vec{\mathbf{c}}$ is outside of the user's control.[12]

Upon receiving the $\vec{\mathbf{c}}$ from the user, the signer computes an LWE sample using his secret $\vec{\mathbf{s}}$ as $h = \vec{\mathbf{s}}^T\vec{\mathbf{c}} + e_3$ and proves that it is a valid LWE sample and that the $\vec{\mathbf{s}}$ is the same $\vec{\mathbf{s}}$ that was used to construct the public key $\vec{\mathbf{t}}^T$. Now we observe that if we define $\vec{\mathbf{u}} = \mathsf{H}\left(\mu, \mathsf{H}(\vec{\mathbf{r}}, \vec{\mathbf{e_2}})\right)$, then

$$h - \vec{\mathbf{t}}^T\vec{\mathbf{r}} = \underbrace{\vec{\mathbf{s}}^T\vec{\mathbf{u}}}_{\text{uniform}} + \underbrace{\vec{\mathbf{s}}^T\vec{\mathbf{e_2}} + e_3 - \vec{\mathbf{e_1}}^T\vec{\mathbf{r}}}_{\text{small}}. \tag{6}$$

If the modulus is large enough and the user outputs $\rho = \mathsf{H}(\vec{\mathbf{r}}, \vec{\mathbf{e_2}})$ in his signature of μ, then both the user and the signer would be able to compute $\lceil h - \vec{\mathbf{t}}^T\vec{\mathbf{r}}\rfloor = \lceil\vec{\mathbf{s}}^T\vec{\mathbf{u}}\rfloor$, where $\lceil\cdot\rfloor$ is the rounding of each integer of the polynomial to the highest few (e.g. 2) order bits. The user thus outputs $\rho, \lceil h - \vec{\mathbf{t}}^T\vec{\mathbf{r}}\rfloor$ as the signature of μ, which is only 48 bytes in length.

Efficiency. The signature size of the scheme is very short, and verification is quite efficient since it doesn't involve any zero-knowledge proofs. The protocol itself, however, is less efficient than the one for the basic blind signature from the previous subsection. While the NIZK that the signer sends to the user is a proof of lattice relations and can be created using the protocol from [LNP22], the proof of (5) does require proving knowledge of inputs and outputs of cryptographic hash functions. In order to ensure that $\lceil h - \vec{\mathbf{t}}^T\vec{\mathbf{r}}\rfloor = \lceil\vec{\mathbf{s}}^T\vec{\mathbf{u}}\rfloor$, we needed to use a modulus that is a lot larger than the largest coefficient of $\vec{\mathbf{s}}^T\vec{\mathbf{e_2}} + e_3 - \vec{\mathbf{e_1}}^T\vec{\mathbf{r}}$. This requires taking $q \approx 2^{150}$ and the overall dimension of the LWE instance to be ≈ 6000. This means that the hash function maps onto a domain of size $\approx 2^{17}$ bytes, which would require computing 2^{12} SHA functions. This is around 100X larger than the 26 SHA functions needed to hash to the domain in the blind signature scheme from the previous section. Using current techniques (e.g. [BFH+20]), this would take around 30 minutes, which is too slow for many applications. Nevertheless, we think that using a more NIZK-compatible hash function as well as exploring the eventual speed-up which should arise from using lattice-based zero-knowledge proofs (e.g. [BS22]) could result in this number coming down to something more manageable. So unlike our main blind signature proposal, which is already practical, we think that there is still work left to be done before our Keyed-Verification Blind Signature can be considered for real-world applications. Still, the possibility of having a round-optimal quantum-safe scheme with a very short signature size is a very intriguing prospect that we believe should encourage research toward these efficiency improvements.

Security. The privacy of the user's μ is preserved because the commitment $\vec{\mathbf{c}}$ is hiding $\mathsf{H}(\mu, \rho)$ and because $\lceil h - \vec{\mathbf{t}}^T\vec{\mathbf{r}}\rfloor$ being equal to $\lceil\vec{\mathbf{s}}^T\vec{\mathbf{u}}\rfloor$, with all but negligible probability, implies that the $\vec{\mathbf{e}}_2$ and $\vec{\mathbf{r}}$ generated by the user are hidden.

To show the privacy of the signer, we first note that the fact that $\vec{\mathbf{c}}$ is generated uniformly at random (and cannot be influenced by the user) implies that $h = \vec{\mathbf{s}}^T\vec{\mathbf{c}} + e_3$ is indistinguishable from uniform based on MLWE. Note that while the user can't control the $\vec{\mathbf{c}}$, he can send the same one twice. Therefore it's important that the signer also uses the same error e_3 for creating the h. This is why the signer generates the e_3 using a PRF (using a secret key K) that takes as input the $\vec{\mathbf{c}}$. Therefore the h will be the same for the same $\vec{\mathbf{c}}$. Furthermore, because $\vec{\mathbf{u}} = \mathsf{H}(\mu, \rho)$ is an output of the random oracle and the modulus is large, the value $\lceil\vec{\mathbf{s}}^T\vec{\mathbf{u}}\rfloor$ does not give out any additional information because it can be derived from the information already known by the adversarial user (i.e. $\lceil\vec{\mathbf{s}}^T\vec{\mathbf{u}}\rfloor = \lceil h - \vec{\mathbf{t}}^T\vec{\mathbf{r}}\rfloor$).

The soundness of our scheme follows from the fact that for a new μ', $\vec{\mathbf{u}} = \mathsf{H}(\mu', \rho')$ is uniformly random, and so by the LWE assumption $\lceil\vec{\mathbf{s}}^T\vec{\mathbf{u}}\rfloor$ is indistinguishable from a completely random

[12] In this protocol we explicitly separate out the "secret" and the "error" of the LWE equation as $\vec{\mathbf{r}}$ and $\vec{\mathbf{e}}_2$, whereas in the blind signature in the previous section, the two were lumped together into $\vec{\mathbf{r}}$ and $\mathbf{B}$ was extended to include an identity matrix. The reason for this is that the current protocol uses a "Diffie-Hellman"-like protocol (in which the separation between the secret and error is important for efficiency) whereas the previous one used trapdoor sampling (where the secret and the error have the exact same function and so can be lumped together).

element of some small entropy depending on the rounding function, and thus the adversary has a negligible chance of producing a $v' = \lceil \vec{\mathbf{s}}^T \vec{\mathbf{u}} \rfloor$ that would be accepted.

Comparison to generically using the VOPRF from [ADDS21]. The reason our scheme is more efficient than the VOPRF of [ADDS21] is that in their scheme the first move of the client/user was the commitment $\vec{\mathbf{c}} := \mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{e_2}} + \mathsf{H}(\mu)$ along with a zero-knowledge proof proving the correctness of the construction.[13] Because the input to H was not tied to $\vec{\mathbf{r}}$ and $\vec{\mathbf{e}}_2$, the user could later send $\vec{\mathbf{c}}' = \mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{e_2}}' + \mathsf{H}(\mu)$ and then subtract the responses $h = \vec{\mathbf{s}}^T\vec{\mathbf{c}} + e_3$ and $h' = \vec{\mathbf{s}}^T\vec{\mathbf{c}}' + e_3'$ to obtain $\vec{\mathbf{s}}^T(\vec{\mathbf{e}}_2 - \vec{\mathbf{e}}_2') + (e_3 - e_3')$. In order for this to not leak information about $\vec{\mathbf{s}}^T$, the errors e_3, e_3' needed to be significantly larger than $\vec{\mathbf{s}}^T(\vec{\mathbf{e}}_2 - \vec{\mathbf{e}}_2')$ so as to "drown out" any effect of $\vec{\mathbf{s}}$ on the difference. This results in the modulus being at least $q = 2^{256}$ and would require the overall lattice dimension to increase by some noticeable factor as well over the one used in our scheme. We would estimate that the size of the outputs of the hashes in (5) would increase by at least an order of magnitude, which would result in needing to prove that many more hash functions, which would make the running time of the proof significantly less efficient.

Acknowledgements. This work is supported by the EU H2020 ERC Project 101002845 PLAZA.

2 Preliminaries

2.1 Notation.

Let q be a positive integer and let d be a power of two. We denote by $\mathcal{R}_q$ the ring $\mathbb{Z}[x]/(q, x^d + 1)$. Let p be a positive integer less than q. We define a coefficient-wise rounding function $\lceil \cdot \rfloor_p : \mathcal{R}_q \to \mathcal{R}_p$ that multiplies the central representant of each coefficient by p/q and rounds to the nearest integer. If p is clear from the context we omit the subscript p.

2.2 Blind signatures

We provide the correctness and security definitions of blind signatures [Cha82]. A round-optimal blind signature scheme consists of the following five algorithms:

- $\mathsf{KeyGen}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk})$: Takes the security parameter as input and outputs a secret key sk and a public key pk.
- $\mathsf{Sign}_1(\mathsf{pk}, \mu) \to (\rho_1, S_\mathcal{U})$: This is the first part of a signing protocol between a user $\mathcal{U}$ and a signer $\mathcal{S}$, where $\mathcal{U}$ uses a public key and a message $\mu \in \{0,1\}^*$, to compute a state $S_\mathcal{U}$ and a first message ρ_1, which they send to $\mathcal{S}$.
- $\mathsf{Sign}_2(\mathsf{sk}, \rho_1) \to \rho_2$: The second part of the interaction, where $\mathcal{S}$ uses their signing key sk and a first message ρ and outputs a second message ρ_2, which they send to $\mathcal{U}$.
- $\mathsf{Sign}_3(\rho_2, S_\mathcal{U}) \to \mathsf{sig}$: The last part of the interaction, where $\mathcal{U}$ uses ρ_2 and $S_\mathcal{U}$ to derive a signature sig or $\perp$ if the signing protocol failed.
- $\mathsf{Verify}(\mathsf{pk}, \mu, \mathsf{sig}) \to \{0,1\}$: A verification algorithm that takes as input a public key pk, a message μ and a signature sig and outputs a bit to indicate whether the signature is deemed valid (1) or invalid (0).

All the algorithms take the security parameter λ as input (sometimes implicitly). We also define a *keyed-verification* variant of blind signatures, where the Verify algorithm additionally requires sk as input.

We require (keyed-verification) blind signatures to satisfy three properties: correctness, which says that honestly generated signatures should verify with probability close to 1, anonymity, which says that the signer cannot link signatures to the interaction with which they were created, and one-more

[13] The H used in [ADDS21] was not a cryptographic hash function, but a lattice-based PRF [BP14], but this does not matter for this discussion.

unforgeability, which says that if a user is allowed to execute the signing protocol Q times it can obtain no more than Q valid signatures. These properties are formalized in Figure 1.

Definition 2.1 (correctness). *A (keyed-verification) blind signature* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *is correct if there exists a negligible function* negl*, such that for any message* μ *we have*

$$\Pr\left[\mathsf{Verify}(\mathsf{sk},\mathsf{pk},\mu,\mathsf{sig}) = 1 \;\middle|\; \begin{array}{c}(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)\\ (\rho_1, S_{\mathcal{U}}) \leftarrow \mathsf{Sign}_1(\mathsf{pk},\mu)\\ \rho_2 \leftarrow \mathsf{Sign}_2(\mathsf{sk},\rho_1)\\ \mathsf{sig} \leftarrow \mathsf{Sign}_3(\rho_2, S_{\mathcal{U}})\end{array}\right] \geq 1 - \mathsf{negl}(\lambda)\,.$$

Definition 2.2 (anonymity). *A (keyed-verification) blind signature* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *is anonymous if for every polynomial-time three-part stateful adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$ *there exists a negligible function* negl *such that for any two messages* μ_0, μ_1 *we have*

$$\left|\Pr\left[\mathcal{A}_3(\mathsf{sig}^0, \mathsf{sig}^1) = b \;\middle|\; \begin{array}{c}\mathsf{pk} \leftarrow \mathcal{A}_1(1^\lambda), b \overset{\$}{\leftarrow} \{0,1\}\\ (\rho_1^0, S_{\mathcal{U}}^0) \leftarrow \mathsf{Sign}_1(\mathsf{pk},\mu_0), (\rho_1^1, S_{\mathcal{U}}^1) \leftarrow \mathsf{Sign}_1(\mathsf{pk},\mu_1)\\ \rho_2^b, \rho_2^{1-b} \leftarrow \mathcal{A}_2(\rho_1^b, \rho_1^{1-b})\\ \mathsf{sig}^0 \leftarrow \mathsf{Sign}_3(\rho_2^0, S_{\mathcal{U}}^0), \mathsf{sig}^1 \leftarrow \mathsf{Sign}_3(\rho_2^1, S_{\mathcal{U}}^1)\\ \textit{If } \mathsf{sig}^0 = \bot \textit{ or } \mathsf{sig}^1 = \bot\textit{, then } (\mathsf{sig}^0, \mathsf{sig}^1) \leftarrow (\bot, \bot)\end{array}\right] - \frac{1}{2}\right| < \mathsf{negl}(\lambda)\,.$$

Definition 2.3 (one-more unforgeability). *A (keyed-verification) blind signature* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *is one-more unforgeable if for every polynomial-time adversary* $\mathcal{A}$ *that makes at most* Q *queries (*Q *is a function of* λ*, bounded above by a polynomial) there exists a negligible function* negl *such that*

$$\Pr\left[\begin{array}{c}\forall 1 \leq i < j \leq Q+1 : \mu_i \neq \mu_j\\ \forall 1 \leq i \leq Q+1 : \mathsf{Verify}(\mathsf{sk},\mathsf{pk},\mu_i,\mathsf{sig}_i) = 1\end{array} \;\middle|\; \begin{array}{c}(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)\\ \{(\mu_i, \mathsf{sig}_i)\}_{i \in [Q+1]} \leftarrow \mathcal{A}^{\mathsf{Sign}_2(\mathsf{sk},\cdot)),\mathsf{Verify}(\mathsf{sk},\mathsf{pk},\cdot,\cdot)}(\mathsf{pk})\end{array}\right] < \mathsf{negl}(\lambda)\,.$$

In the case of keyed-verification blind signatures, the adversary has access to a verification oracle that on input (μ, sig) *returns* $\mathsf{Verify}(\mathsf{sk}, \mathsf{pk}, \mu, \mathsf{sig})$.

Fig. 1: The three security definitions of a (keyed-verification) blind signature scheme: correctness, anonymity and one-more unforgeability. For keyed-verification blind signatures the definitions are slightly different, with the modifications given in gray

.

3 Blind Signature

We first introduce the basic signature scheme that forms the starting point in the construction of our blind signature scheme. The basic signature is based on Falcon [PFH+17], which is an instantiation of the GPV framework [GPV08]. It uses an NTRU trapdoor, and comes with fast FFT-type algorithms for computing the trapdoor basis and its Gram-Schmidt orthogonalization. In slightly simplified notation, a Falcon public key is the matrix $\boldsymbol{A} = (\boldsymbol{t}, \boldsymbol{1}) \in \mathcal{R}_q^{1\times 2}$ defining an NTRU lattice $\mathcal{L}_q^\perp(\boldsymbol{A})$. The corresponding private key is a trapdoor basis for the lattice that makes it possible to sample relatively narrow Gaussian vectors in any lattice coset. A signature for a message $\mu \in \{0,1\}^*$ consists of a Gaussian distributed short preimage vector $\vec{\boldsymbol{s}} \in \mathcal{R}_q^2$ such that $\boldsymbol{A}\vec{\boldsymbol{s}} = \mathsf{H}(\mu)$, i.e. a Gaussian vector in the lattice coset $(\boldsymbol{0}, \mathsf{H}(\mu))^T + \mathcal{L}_q^\perp(\boldsymbol{A})$. Here the hash function H on the right hand side is modeled as a random oracle that maps onto $\mathcal{R}_q$. We use a deterministic variant of Falcon where the randomness for the preimage sampler comes from a PRG that is seeded with a key κ in the secret key and the message μ. We extend Falcon by adding a second uniformly random non-trapdoored matrix $\boldsymbol{B} \in \mathcal{R}_q^{1\times 2}$ to the public key, and a random short vector $\vec{\boldsymbol{r}} \in \mathcal{R}_q^2$ to the message that can be chosen by the user in order to blind the right hand side. More precisely, a signature for a message $(\mu, \vec{\boldsymbol{r}})$ consists of a Gaussian short preimage $\vec{\boldsymbol{s}}$ such that $\boldsymbol{A}\vec{\boldsymbol{s}} = \boldsymbol{B}\vec{\boldsymbol{r}} + \mathsf{H}(\rho, \mu)$, where $\rho = \mathsf{G}(\vec{\boldsymbol{r}}) \in \{0,1\}^{2\lambda}$ for a second random oracle G. Despite the additional term $\boldsymbol{B}\vec{\boldsymbol{r}}$, the right hand side $\boldsymbol{c} = \boldsymbol{B}\vec{\boldsymbol{r}} + \mathsf{H}(\mathsf{G}(\vec{\boldsymbol{r}}), \mu)$ is still given by a random oracle evaluated at the message $(\mu, \vec{\boldsymbol{r}})$. This is important for the security proof as it means that an adversary can only use the signing oracle in the forgery game to query preimages for independently random polynomials $\boldsymbol{c}$ and not for specific ones of his choosing. On the other hand, an important observation for our blind signature is that the relation between ρ and $\vec{\boldsymbol{r}}$ via the random oracle G does not need to be checked in signature verification. The reason is that $\vec{\boldsymbol{r}}$ has to be short and therefore a forged signed message gives a short preimage for $\boldsymbol{h} = \mathsf{H}(\rho, \mu)$ under the matrix $(\boldsymbol{A}, -\boldsymbol{B})$. We summarize the basic signature scheme in Figure 2.

Now, when a signature is viewed solely as a signature for μ, it consists of the random bit string ρ and the short vectors $\vec{\boldsymbol{r}}$, $\vec{\boldsymbol{s}}$ such that $\boldsymbol{A}\vec{\boldsymbol{s}} = \boldsymbol{B}\vec{\boldsymbol{r}} + \mathsf{H}(\rho, \mu) = \boldsymbol{B}\vec{\boldsymbol{r}} + \boldsymbol{h} = \boldsymbol{c}$. This makes it very easy to prove knowledge of signatures without revealing $\boldsymbol{c}$. One can reveal ρ, and prove knowledge of $\vec{\boldsymbol{r}}$ and $\vec{\boldsymbol{s}}$ in zero-knowledge without the complication of proving random oracles. Revealing ρ means that $\boldsymbol{h}$ is revealed, but not the full right-hand side $\boldsymbol{c}$. Indeed, the random oracle G hides $\vec{\boldsymbol{r}}$ and then $\boldsymbol{B}\vec{\boldsymbol{r}}$ blinds $\boldsymbol{c}$ under the Ring-LWE assumption. So in our blind signature scheme, the user can request signatures by sending a polynomial $\boldsymbol{c}$ as this is all that is needed for the signature sampling, and then anonymously prove knowledge of a signature. But in order for this to be secure the user must prove well-formedness of $\boldsymbol{c}$ in his signature queries, i.e. he must prove that $\boldsymbol{c} = \boldsymbol{B}\vec{\boldsymbol{r}} + \mathsf{H}(\mathsf{G}(\vec{\boldsymbol{r}}), \mu)$ for a short vector $\vec{\boldsymbol{r}}$.

```
keygen($\boldsymbol{B}$), $\boldsymbol{B} \in \mathcal{R}_q^{1\times 2}$
$\boldsymbol{f}, \boldsymbol{g} \leftarrow D_{\sigma_0}^d \subset \mathcal{R}_q$
$\kappa \leftarrow \{0,1\}^\lambda$
$\boldsymbol{A} = (\boldsymbol{f}\boldsymbol{g}^{-1}, \boldsymbol{1}) \in \mathcal{R}_q^{1\times 2}$
Extend $\boldsymbol{f}, \boldsymbol{g}$ to a trapdoor basis
    $\boldsymbol{T} \in \mathcal{R}_q^{2\times 2}$ for the lattice $\mathcal{L}_q^\perp(\boldsymbol{A})$
pk = $(\boldsymbol{A}, \boldsymbol{B})$, sk = $(\boldsymbol{T}, \boldsymbol{B}, \kappa)$
```

```
sign($\mu$, $\vec{\boldsymbol{r}}$, sk), $\mu \in \{0,1\}^*$, $\vec{\boldsymbol{r}} \in \mathcal{R}_q^2$
If $\|\vec{\boldsymbol{r}}\| > \beta_r$, return $\perp$
Parse sk = $(\boldsymbol{T}, \boldsymbol{B}, \kappa)$
$\rho = \mathsf{G}(\vec{\boldsymbol{r}})$
$\boldsymbol{c} = \boldsymbol{B}\vec{\boldsymbol{r}} + \mathsf{H}(\rho, \mu)$
Preimage sample $\vec{\boldsymbol{s}} \leftarrow D_\sigma^{2d}$ such that $\boldsymbol{A}\vec{\boldsymbol{s}} = \boldsymbol{c}$
    by using the trapdoor basis $\boldsymbol{T}$, with randomness
    seeded from $(\kappa, \mu)$
If $\|\vec{\boldsymbol{s}}\| > \beta_s$, resample
sig = $(\rho, \vec{\boldsymbol{s}})$
```

```
verify($\mu$, $\vec{\boldsymbol{r}}$, sig, pk)
Parse sig = $(\rho, \vec{\boldsymbol{s}})$, pk = $(\boldsymbol{A}, \boldsymbol{B})$
If $\|\vec{\boldsymbol{r}}\| > \beta_r$ or $\|\vec{\boldsymbol{s}}\| > \beta_s$, return 0
If $\boldsymbol{A}\vec{\boldsymbol{s}} \neq \boldsymbol{B}\vec{\boldsymbol{r}} + \mathsf{H}(\rho, \mu)$, return 0
Return 1
```

Fig. 2: The basic signature scheme.

Theorem 3.1. *The signature scheme from Figure 2 is one-more unforgeable under the NTRU and Ring-SIS assumptions when the hash functions* G *and* H *are modeled as random oracles. More precisely, for an adversary A in the signature forgery game that makes up to Q queries to the (random) oracles and has advantage* $\mathsf{Adv}_{\mathsf{SIG}}(A)$*, there exists an adversary B that distinguishes NTRU public keys* $\boldsymbol{f}\boldsymbol{g}^{-1}$ *from uniformly random with advantage* $\mathsf{Adv}_{\mathsf{NTRU}}(B)$*, and an adversary C that solves Ring-SIS with norm bound* $\beta = 2\sqrt{\beta_s^2 + \beta_r^2}$ *with advantage* $\mathsf{Adv}_{\mathsf{RSIS}_\beta}(C)$*, such that*

$$\mathsf{Adv}_{\mathsf{SIG}}(A) \leq \mathsf{Adv}_{\mathsf{NTRU}}(B) + \mathsf{Adv}_{\mathsf{RSIS}_\beta}(C) + \frac{Q}{2^{\lambda-1}}.$$

Proof. First, we simulate the signing oracle for the adversary without making use of the NTRU Trapdoor by exercising control over the random oracles G and H. We implement these oracles as shown in Figure 3. Notice that the algorithms for G and H produce uniformly random outputs for new queries and consistent outputs for repeated queries. For consistency, the algorithms maintain

```
$\mathsf{G}(\vec{\boldsymbol{r}})$
If $\exists \rho \in \{0,1\}^{2\lambda}$ s.t. $(\vec{\boldsymbol{r}}, \rho) \in R$,
    $\rho \leftarrow \{\rho' \in \{0,1\}^{2\lambda} \mid (\vec{\boldsymbol{r}}, \rho') \in R\}$
    Return $\rho$
$\rho \leftarrow \{0,1\}^{2\lambda}$
$R = R \cup \{(\vec{\boldsymbol{r}}, \rho)\}$
Return $\rho$
```

```
$\mathsf{H}(\rho, \mu)$
If $\exists \boldsymbol{h} \in \mathcal{R}_q$ s.t. $(\rho, \mu, \boldsymbol{h}) \in S$,
    $\boldsymbol{h} \leftarrow \{\boldsymbol{h}' \in \mathcal{R}_q \mid (\rho, \mu, \boldsymbol{h}') \in S\}$
    Return $\boldsymbol{h}$
$\vec{\boldsymbol{s}} \leftarrow D_\sigma^{2d} \subset \mathcal{R}_q^2$
If $\|\vec{\boldsymbol{s}}\| > \beta_s$, resample
If $\exists \vec{\boldsymbol{r}} \in \mathcal{R}_q^2$ s.t. $\|\vec{\boldsymbol{r}}\| \leq \beta_r$ and $(\vec{\boldsymbol{r}}, \rho) \in R$,
    $\vec{\boldsymbol{r}} \leftarrow \{\vec{\boldsymbol{r}}' \in \mathcal{R}_q^2 \mid \|\vec{\boldsymbol{r}}'\| \leq \beta_r \wedge (\vec{\boldsymbol{r}}', \rho) \in R\}$
Else,
    $\vec{\boldsymbol{r}} \leftarrow D_{\sigma_0}^{2d} \subset \mathcal{R}_q^2$
    If $\|\vec{\boldsymbol{r}}\| > \beta_r$, resample
$\boldsymbol{h} = \boldsymbol{A}\vec{\boldsymbol{s}} - \boldsymbol{B}\vec{\boldsymbol{r}}$
$S = S \cup \{(\rho, \mu, \boldsymbol{h})\}$
$S' = S' \cup \{(\vec{\boldsymbol{r}}, \vec{\boldsymbol{s}}, \boldsymbol{h})\}$
Return $\boldsymbol{h}$
```

```
$\mathsf{sign}'(\mu, \vec{\boldsymbol{r}})$
$\rho = \mathsf{G}(\vec{\boldsymbol{r}})$
$\boldsymbol{h} = \mathsf{H}(\rho, \mu)$
$(\vec{\boldsymbol{r}}', \vec{\boldsymbol{s}}') \leftarrow \{(\vec{\boldsymbol{r}}', \vec{\boldsymbol{s}}') \in \mathcal{R}_q^2 \times \mathcal{R}_q^2 \mid (\vec{\boldsymbol{r}}', \vec{\boldsymbol{s}}', \boldsymbol{h}) \in S'\}$
If $\vec{\boldsymbol{r}}' \neq \vec{\boldsymbol{r}}$, abort
Return $\vec{\boldsymbol{s}}'$
```

Fig. 3: The oracles for the forgery game. G and H are random oracles and sign' is a simulation of the signing oracle that makes use of the relation S' maintained by H.

the relations R and S. For random outputs, G just samples a uniformly random $\rho \in \{0,1\}^{2\lambda}$ for every new query. H first checks if the argument $\rho \in \{0,1\}^{2\lambda}$ was output by G before and if there exists a corresponding input $\vec{\boldsymbol{r}}$ in the relation R that is short. In this case the algorithm chooses one such $\vec{\boldsymbol{r}}$ at random. Otherwise H just samples a short Gaussian $\vec{\boldsymbol{r}}$ over $\mathbb{Z}^{2d}$. Next, H samples $\vec{\boldsymbol{s}} \leftarrow D_\sigma^{2n}$ and programs the output $\boldsymbol{h}$ such that $\boldsymbol{A}\vec{\boldsymbol{s}} = \boldsymbol{B}\vec{\boldsymbol{r}} + \boldsymbol{h}$. Here $\boldsymbol{h}$ is statistically indistinguishable from uniformly random because the standard deviation σ is above smoothing for the sublattice $\mathcal{L}_q^{\perp}(\boldsymbol{A})$. The preimage $(\vec{\boldsymbol{r}}, \vec{\boldsymbol{s}})$ is recorded in the relation S'. Observe that the relation S' contains short preimages $(\vec{\boldsymbol{r}}, \vec{\boldsymbol{s}})$ for all polynomials $\boldsymbol{h}$ that have ever been output by H.

Now, for a signing query $(\mu, \vec{\boldsymbol{r}})$ the simulated oracle sign' runs $\rho \leftarrow \mathsf{G}(\vec{\boldsymbol{r}})$ followed by $\boldsymbol{h} \leftarrow \mathsf{H}(\rho, \mu)$. Then the relation S' contains a preimage $(\vec{\boldsymbol{r}}', \vec{\boldsymbol{s}})$ such that $\boldsymbol{A}\vec{\boldsymbol{s}} = \boldsymbol{B}\vec{\boldsymbol{r}}' + \boldsymbol{h}$ and sign' chooses one such preimage at random. If $\vec{\boldsymbol{r}}' \neq \vec{\boldsymbol{r}}$ the forgery game is aborted. Otherwise $(\rho, \vec{\boldsymbol{s}})$ is returned as the signature to the adversary. We now analyze the probability that the aborting condition $\vec{\boldsymbol{r}}' \neq \vec{\boldsymbol{r}}$ happens. Firstly, it could happen if there existed a collision for $\boldsymbol{h}$ in S, i.e. if H outputted $\boldsymbol{h}$ for different inputs. Since the space for $\boldsymbol{h}$ is huge (more than 2^{6000} elements, depending on parameters) this in fact can not happen. Secondly, abort can happen if there is a collision for ρ in R. Since the space for ρ is of size $2^{2\lambda}$ the probability for this is essentially $Q^2/2^{2\lambda+1}$. Finally, it can happen if the adversary called $\mathsf{H}(\rho, \mu)$ before ρ was output by G. The probability for this is bounded by $Q/2^{\lambda}$.

So, the new game with simulated signing oracle has statistical distance at most $Q/2^{\lambda-1}$ from the original game. We can then replace the left polynomial $\boldsymbol{t} = \boldsymbol{f}\boldsymbol{g}^{-1}$ in the matrix $\boldsymbol{A}$ from the public key by a uniformly random $\boldsymbol{t}$. The adversary A can only distinguish this with advantage $\mathsf{Adv}_{\mathsf{NTRU}}(B)$.

Finally, assume the adversary produces a forged signature $(\rho, \vec{\boldsymbol{s}})$ for a message $(\mu, \vec{\boldsymbol{r}})$ that he did not use in any of his signing queries. That is, we get $\boldsymbol{A}\vec{\boldsymbol{s}} = \boldsymbol{B}\vec{\boldsymbol{r}} + \mathsf{H}(\rho, \mu)$. Then in the relation S' there also is a short preimage $(\vec{\boldsymbol{r}}', \vec{\boldsymbol{s}}')$ for the hash $\boldsymbol{h} = \mathsf{H}(\rho, \mu)$. We show that $(\vec{\boldsymbol{r}}, \vec{\boldsymbol{s}}) \neq (\vec{\boldsymbol{r}}', \vec{\boldsymbol{s}}')$ with high probability. Assume that $\vec{\boldsymbol{r}} = \vec{\boldsymbol{r}}'$. Then, given $\boldsymbol{h}$, the conditioned distribution for $\vec{\boldsymbol{s}}'$ is the Gaussian distribution with standard deviation σ in the coset $(\mathbf{0}, \boldsymbol{B}\vec{\boldsymbol{r}} + \boldsymbol{h})^T + \mathcal{L}_q^{\perp}(\boldsymbol{A})$. Since the min-entropy of this distribution is large, the probability that $\vec{\boldsymbol{s}}' = \vec{\boldsymbol{s}}$ is small. Hence, we have a non-trivial Ring-SIS solution $\boldsymbol{A}(\vec{\boldsymbol{s}} - \vec{\boldsymbol{s}}') - \boldsymbol{B}(\vec{\boldsymbol{r}} - \vec{\boldsymbol{r}}') = \mathbf{0}$ of length $2\sqrt{\beta_s^2 + \beta_r^2}$ with high probability.

Signer		User
$\boldsymbol{T} \in \mathcal{R}_q^{2\times 2}, \boldsymbol{B} \in \mathcal{R}_q^{1\times 2}$, pk		$\boldsymbol{A}, \boldsymbol{B} \in \mathcal{R}_q^{1\times 2}$, pk
		$\vec{\mathbf{r}} \leftarrow D_{\sigma_0} \in \mathcal{R}_q^2$ If $\Vert\vec{\mathbf{r}}\Vert > \beta_r$, resample $\mathbf{c} = \mathbf{B}\vec{\mathbf{r}} + \mathsf{H}\left(\mathsf{G}(\vec{\mathbf{r}}), \mu\right)$ $r \leftarrow \{0,1\}^{\lambda}$ $\mathsf{ct} = \mathsf{PKE.Enc}(\vec{\mathbf{r}}, \mu; \mathsf{pk}, r)$
	$\xleftarrow{\mathbf{c}, \mathsf{ct}, \pi_1}$	$\pi_1 = \mathsf{NIZK}_1.P\left(\vec{\mathbf{r}}, \mu, r \,\middle\vert\, \begin{array}{l} \mathbf{c} = \mathbf{B}\vec{\mathbf{r}} + \mathsf{H}\left(\mathsf{G}(\vec{\mathbf{r}}), \mu\right) \\ \wedge\, \mathsf{ct} = \mathsf{enc}(\vec{\mathbf{r}}, \mu; \mathsf{pk}, r) \\ \wedge\, \Vert\vec{\mathbf{r}}\Vert \le \beta_r \end{array}\right)$
If $\mathsf{NIZK}_1.V(\pi_1) = 0$, abort $\vec{\mathbf{s}} \leftarrow \mathbf{A}_\sigma^{-1}(\mathbf{c}) \in \mathcal{R}_q^2$ If $\Vert\vec{\mathbf{s}}\Vert > \beta_s$, resample	$\xrightarrow{\vec{\mathbf{s}}}$	
		$\rho = \mathsf{G}(\vec{\mathbf{r}})$ $\mathbf{h} = \mathsf{H}(\rho, \mu)$ $\pi_2 = \mathsf{NIZK}_2.P\left(\vec{\mathbf{r}}, \vec{\mathbf{s}} \,\middle\vert\, \begin{array}{l} \mathbf{A}\vec{\mathbf{s}} = \mathbf{B}\vec{\mathbf{r}} + \mathbf{h} \\ \wedge\, \Vert\vec{\mathbf{s}}\Vert \le \beta_s \wedge \Vert\vec{\mathbf{r}}\Vert \le \beta_r \end{array}\right)$ Signature for μ is ρ, π_2
Verifying a signature: $\mathbf{h} = \mathsf{H}(\rho, \mu)$ Return $\mathsf{NIZK}_2.V(\pi_2)$		

Fig. 4: Our first blind signature scheme. NIZK_1 is a sound zero-knowledge proof system, NIZK_2 is an adaptively knowledge-sound zero-knowledge proof system, and PKE a CPA-secure public-key encryption scheme. pk is a public key for PKE. A corresponding secret key is not known to anybody.

So since the basic signature scheme is secure in the random oracle model, we can assume that the scheme where H and G are instantiated with a concrete hash function such as SHA2 is secure. Now we turn to the blind signature scheme as given in Figure 4 and prove its security by reducing it to the unforgeability of the basic signature scheme and the soundness and zero-knowledge properties of the NIZKs. For technical reasons in the security proof, we also need that the user encrypts the witness to the sky. But we do not think that this is actually needed for security in practice.

Theorem 3.2. *The blind signature scheme in Figure 4 is one-more unforgeable based on the unforgeability of the basic signature scheme and of the soundness of the NIZKs. More precisely, for every adversary A that makes at most Q oracle queries in the one-more unforgeability game, there exist adversaries B and C against the soundness of the NIZK and the unforgeability of the basic signature scheme, respectively, such that*

$$\mathsf{Adv}_{\mathsf{BSIG}}(A) \le (Q+1)\mathsf{Adv}_{\mathsf{SND}}(B) + \mathsf{Adv}_{\mathsf{SIG}}(C).$$

Proof. In the reduction, we can assume that we know the secret key corresponding to the public key pk for the encryption to the sky. So when the adversary A makes a signing query we first verify the proof π_1. If this is successful we decrypt the ciphertext ct and get a message μ and a vector $\vec{\boldsymbol{r}} \in \mathcal{R}_q^2$. The proof shows that $\boldsymbol{c} = \boldsymbol{B}\vec{\boldsymbol{r}} + \mathsf{H}(\mathsf{G}(\vec{\boldsymbol{r}}), \mu)$ and $\Vert\vec{\boldsymbol{r}}\Vert \le \beta_r$. By the soundness property of the NIZK, the adversary can not come up with a valid proof for a false statement, except with

Parameter	Instantiation	Explanation
λ	128	security parameter
d	512	degree of the cyclotomic ring $\mathcal{R}$
q	7933	modulus
D_{σ_0}	$\mathcal{U}(S_2)$	prob. distribution for polynomials of $\vec{\mathbf{r}}$
σ	232	standard deviation
β_r	$2\sqrt{md}$	norm bound on $\vec{\mathbf{r}}$
β_s	$\sigma\sqrt{2nd}$	norm bound on $\vec{\mathbf{s}}$

Table 2: Overview of parameters and notation. Here, $\mathcal{U}(S_2)$ denotes a uniformly random distribution over polynomials in $\mathcal{R}$ with coefficients between -2 and 2.

advantage $\mathsf{Adv}_{\mathsf{SND}}(B)$. So we assume that the statement is true. Then we can query the signing oracle from the unforgeability game with the message $(\mu, \vec{\boldsymbol{r}})$ and get a signature $(\rho, \vec{\boldsymbol{s}})$ for it. We return $\vec{\boldsymbol{s}}$ to the adversary.

Now, when the adversary successfully produces $Q{+}1$ signed messages $(\mu_i, \rho_i, \pi_{2,i})$ for $i = 1, \ldots, Q{+}1$, we can check for each if the message was used in one of the at most Q signing queries. So there exists a signed message (μ^*, ρ^*, π_2^*) where the message μ^* has not been used. We can then use the extractor for the NIZK to extract a witness $(\vec{\boldsymbol{s}}^*, \vec{\boldsymbol{r}}^*)$ such that $\boldsymbol{A}\vec{\boldsymbol{s}}^* = \boldsymbol{B}\vec{\boldsymbol{r}}^* + \mathsf{H}(\rho^*, \mu^*)$, $\|\vec{\boldsymbol{s}}\| \leq \beta_s$ and $\|\vec{\boldsymbol{r}}\| \leq \beta_r$. So we have a forged signature $(\rho^*, \vec{\boldsymbol{s}}^*)$ for the message $(\mu^*, \vec{\boldsymbol{r}}^*)$ that has not been queried in the unforgeability game for the basic signature scheme.

Theorem 3.3. *The blind signature scheme in Figure 4 is anonymous based on the zero-knowledge property of the NIZK. More precisely, for every adversary A in the anonymity game there exists an adversary B that can distinguish simulated NIZK proofs from real ones with advantage* $\mathsf{Adv}_{\mathsf{ZK}}(B)$*, an adversary C against the CPA security of the encryption scheme with advantage* $\mathsf{Adv}_{\mathsf{CPA}}(C)$*, and an adversary D that can distinguish Ring-LWE with advantage* $\mathsf{Adv}_{\mathsf{RLWE}}(D)$*, such that*

$$\mathsf{Adv}_{\mathsf{ANON}}(A) \leq \mathsf{Adv}_{\mathsf{ZK}}(B) + \mathsf{Adv}_{\mathsf{CPA}}(C) + \mathsf{Adv}_{\mathsf{RLWE}}(D).$$

Proof. First we replace the two NIZKs in the signing protocol and the signature generation by their simulations. This can only be distinguished by the adversary with advantage $\mathsf{Adv}_{\mathsf{ZK}}(B)$. Then the two proofs π_1 and π_2 do not depend on μ, $\vec{\boldsymbol{r}}$ and $\vec{\boldsymbol{s}}$ anymore. It remains to show that the right-hand sides $\vec{\boldsymbol{c}} = \boldsymbol{B}\vec{\boldsymbol{r}} + \mathsf{H}(\mathsf{G}(\vec{\boldsymbol{r}}), \mu)$ and the ciphertexts ct do not leak information about the messages, given that the adversary also gets the hashes $\rho = \mathsf{G}(\vec{\boldsymbol{r}})$ and $\boldsymbol{h} = \mathsf{H}(\rho, \mu)$. The ciphertexts for the two different messages can only be distinguished with advantage $\mathsf{Adv}_{\mathsf{CPA}}(C)$. For the right-hand sides $\boldsymbol{c}$ we prove that the pair $(\boldsymbol{B}\vec{\boldsymbol{r}}, \mathsf{H}'(\vec{\boldsymbol{r}}))$ is indistinguishable from uniformly random under the Ring-LWE assumption when H' is modeled as a random oracle. So in the reduction to Ring-LWE we get a polynomial $\boldsymbol{t}$ that is either a Ring-LWE sample or uniformly random. We sample a uniformly random $\boldsymbol{h} \in \mathcal{R}_q$ and send $(\boldsymbol{t}, \boldsymbol{h})$ to the adversary. When the adversary calls the random oracle for the first time with a short vector $\vec{\boldsymbol{r}} \in \mathcal{R}_q^2$ such that $\boldsymbol{B}\vec{\boldsymbol{r}} = \boldsymbol{t}$ and $\|\vec{\boldsymbol{r}}\| \leq \beta_r$ we return $\boldsymbol{h}$, otherwise we return a uniformly random polynomial. This perfectly simulates the view of the adversary and he can only distinguish the two cases when he can distinguish Ring-LWE, with advantage $\mathsf{Adv}_{\mathsf{RLWE}}(D)$.

Parameter selection. We measure the hardness of the computational lattice problems with the root Hermite factor δ and aim for $\delta < 1.0045$ similarly as in [BLS19, ALS20, ENS20, LNS21] for 128-bit security. For Module-SIS, we applied the standard methodology from [MR09, GN08]. As for Module-LWE, we used the LWE-Estimator by Albrecht et al. [APS15].

We propose the parameters in Table 2. Namely, we set $(q, d) = (7933, 512)$ where 7933 is a prime congruent to 5 modulo 8. For the commitment randomness $\vec{\boldsymbol{r}}$ generated by the user, we need $\mathsf{Adv}_{\mathsf{RLWE}}(D)$ to be negligible. By setting D_{σ_0} to be uniform distribution over polynomials in $\mathcal{R}_q$ with coefficients between -2 and 2, we obtain the root Hermite factor 1.0042.

We use NTRU lattices [DLP14, PFH+17] to instantiate preimage sampling. Namely, let

$$\sigma = 1.17\sqrt{q} \cdot \frac{1}{\pi}\sqrt{\frac{1}{2}\ln\left(2+\frac{2}{\epsilon}\right)} \quad \text{where} \quad \epsilon = \frac{2^{-\lambda}}{4d}.$$

Then, there exist two PPT algorithms $\mathsf{NTRU.TrapGen}$ (c.f. [DLP14, Algorithm 2]) and $\mathsf{NTRU.SamplePre}$ (c.f. [DLP14, Theorem 1]) such that for any $c \in \mathcal{R}_q$ and $\mathsf{NTRU.TrapGen}(q,d) \to (t, \mathbf{T}) \in \mathcal{R}_q \times \mathcal{R}_q^{2\times 2}$:

$$\Delta\left(\mathsf{NTRU.SamplePre}(t, \mathbf{T}, c, \sigma), \left[t\ 1\right]_\sigma^{-1}(c)\right) \leq 2^{-\lambda}.$$

Further, we discuss the bounds on the secret vectors. Since coefficients of $\vec{\mathbf{r}}$ are between -2 and 2, we set $\beta_r = 2\sqrt{md}$. As for $\vec{\mathbf{s}}$, we use the general tail-bound property of discrete Gaussians over lattices [AGHS13, Lemma 3] and set $\beta_s = \mathfrak{s}\sqrt{2nd}$.

As for the first message sent by the user, we need an IND-CPA encryption scheme enc and the proof π_1 to satisfy correctness, zero-knowledge, and soundness. As discussed in the introduction, π_1 can be instantiated fairly efficiently using [BS22] or [BFH+20]. Since this part does not contribute to the signature size, we leave picking a concrete instantiation of π_1 as future work. Recall that the signature is a non-interactive zero-knowledge proof of knowledge of $\vec{\mathbf{s}}$ and $\vec{\mathbf{r}}$ such that (i) $\mathbf{A}\vec{\mathbf{s}} = \mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{h}}$, (ii) $\|\vec{\mathbf{s}}\| \leq \beta_s$, and (iii) $\|\vec{\mathbf{r}}\| \leq \beta_r$. Using the standard way to map equations over $\mathcal{R}_q$ to $\mathbb{Z}_q$ using rotational matrices, one can equivalently prove knowledge of $\vec{\mathbf{s}} \in \mathbb{Z}_q^{2d}$ and $\vec{\mathbf{r}} \in \mathbb{Z}_q^{2d}$ such that:

$$\mathbf{A}\vec{\mathbf{s}} = \mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{h}} \pmod{q}, \quad \|\vec{\mathbf{s}}\| \leq \beta_s, \quad \|\vec{\mathbf{r}}\| \leq \beta_r$$

where $(\mathbf{A}, \mathbf{B}, \vec{\mathbf{h}})$ is a statement. To this end, we apply the recent lattice-based zero-knowledge framework by Lyubashevsky et al. [LNP22] to prove the relations above. Hence as required by the [LNP22], we pick the proof system modulus $\hat{q}$ to be a product of two primes congruent to 5 modulo 8, where the smaller one is $q = 7933$. The reason is that $\mathbf{A}\vec{\mathbf{s}} = \mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{h}} \pmod{q}$ is equivalent to $(\frac{\hat{q}}{q} \cdot \mathbf{A})\vec{\mathbf{s}} = (\frac{\hat{q}}{q} \cdot \mathbf{B})\vec{\mathbf{r}} + \frac{\hat{q}}{q} \cdot \vec{\mathbf{h}} \pmod{\hat{q}}$, thus we transformed the original relation over $\mathbb{Z}_q$ to $\mathbb{Z}_{\hat{q}}$ and we can apply directly the protocol from [LNP22, Figure 10]. We compute the parameters using the SAGE code provided by [LNP22], and additionally make use of small optimizations (related to rejection sampling) introduced in the follow-up work [LN22]. For our parameters in Table 2, we obtain the proof system modulus $\hat{q} = 2199023260649 \approx 2^{41}$ and the total proof size is 21.5KB. Including 2λ bits of ρ, the total signature size can be bounded by 22KB. We also include the public and secret key calculation in Table 3. Recall that since $\mathbf{B}$ is uniformly random, we can only store the seed for generation, hence the size of a is the most expensive part of the public key. As for the secret key $\mathbf{T}$, we use the naive bound that the coefficients of $\mathbf{T}$ are less than $1.17\sqrt{q}$ in the absolute value.

Public key	Secret key	Signature
1KB	2KB	22KB

Table 3: Public key, user secret key, and signature sizes of our first blind signature scheme.

Finally, for the unforgeability property, we need to make sure that $\mathsf{Adv}_{\mathsf{RSIS}_\beta}(C)$ is negligible for $\beta = 2\sqrt{\beta_s^2 + \beta_r^2}$. From the parameters in Table 2, we obtain the root Hermite factor 1.0043.

4 Keyed-Verification Blind Signatures

4.1 Description of the scheme.

Our scheme is parametrized by the following values (functions of the security parameter). Given these values, the protocol is described in Figure 5.

- q, the MLWE modulus,

- d, the dimension of the ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d+1)$,
- n, the in number of $\mathcal{R}_q$ elements in the MLWE secret $\vec{\mathbf{s}}$,
- p, the modulus used for the rounding $\lceil\cdot\rfloor_p$,
- D_e, an error distribution on $\mathcal{R}_q$,
- β_s, β_e, ℓ_2-norm bounds,
- $\mathsf{PKE} = (\mathsf{keygen}, \mathsf{enc}, \mathsf{dec})$, a public key encryption scheme with randomness space $\mathcal{R}$.
- $\mathsf{H}_{\mathcal{R}_q^n}, \mathsf{H}_{\{0,1\}^{2\lambda}}, \mathsf{H}_{D_e}$, hash functions that respectively output elemens of $\mathcal{R}_q$, $\{0,1\}^{2\lambda}$, and elements of $\mathcal{R}_q$ sampled from the error distribution D_e distribution conditioned on their ℓ_2-norm being less than β_e,
- two non-interactive zero-knowledge proofs systems NIZK_1, NIZK_2 for their respective relations

$$\mathsf{R}_1 = \left\{ ((\vec{\mathbf{c}}, \mathsf{ct}, \mathsf{pk}), (\vec{\mathbf{r}}, \vec{\mathbf{e_2}}, \mu)) \;\middle|\; \begin{array}{c} \|\vec{\mathbf{r}}\| \le \beta_s, \|\vec{\mathbf{e_2}}\| \le \beta_s, \\ \mathsf{ct} = \mathsf{enc}(\mathsf{pk}, (\vec{\mathbf{r}}, \vec{\mathbf{e_2}}, \mu)) \\ \rho := \mathsf{H}_{\{0,1\}^{2\lambda}}(\vec{\mathbf{r}}, \vec{\mathbf{e_2}}) \\ \vec{\mathbf{c}} := \mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{e_2}} + \mathsf{H}_{\mathcal{R}_q^n}(\mu, \rho) \end{array} \right\}, \text{ and}$$

$$\mathsf{R}_2 = \left\{ ((\vec{\mathbf{c}}, \vec{\mathbf{t}}, h), (\vec{\mathbf{s}}, \vec{\mathbf{e_1}}, e_3)) \;\middle|\; \begin{array}{c} \|\vec{\mathbf{s}}\| < \beta_s, \|\vec{\mathbf{e_1}}\| < \beta_s, \|e_3\| < \beta_e, \\ h = \vec{\mathbf{s}}^T \vec{\mathbf{c}} + e_3 \\ \vec{\mathbf{t}}^T = \vec{\mathbf{s}}^T \mathbf{B} + \vec{\mathbf{e_1}} \end{array} \right\}.$$

4.2 Security Analysis

We first prove the correctness and anonymity of our scheme.

Theorem 4.1. *Suppose NIZK_1 and NIZK_2 are correct, and that $\sqrt{d}\left(2d\beta_s^2 + \beta_e\right)p/q$ is negligible, then the keyed-verification blind signature of Figure 5 is correct. If in addition, NIZK_1 is zero-knowledge, NIZK_2 is adaptively knowledge-sound, the PKE is IND-CPA secure, and the MLWE_{n,D_e} and $\mathsf{MSIS}_{n,2n,2\sqrt{2}\beta_s}$ problems are hard, then the blind signature is anonymous.*

Proof. **correctness.** Due to the correctness of the NIZK protocols the proofs will pass verification with overwhelming probability, so the interactive signing protocol outputs $(\rho, v) = (\mathsf{H}_{\{0,1\}^{2\lambda}}(\vec{\mathbf{r}}, \vec{\mathbf{e}}_2), \lceil h - \vec{\mathbf{t}}^T\vec{\mathbf{r}}\rfloor)$, where $h = \vec{\mathbf{s}}^T(\mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{e_2}} + \vec{\mathbf{u}}) + e_3$, and $\vec{\mathbf{t}}^T = \vec{\mathbf{s}}^T\mathbf{B} + \vec{\mathbf{e_1}}^{\mathbf{T}}$, where $\vec{\mathbf{u}} = \mathsf{H}_{\mathcal{R}_q^n}(\mu, \rho)$. So we have

$$\begin{aligned} v &= \lceil \vec{\mathbf{s}}^T(\mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{e_2}} + \vec{\mathbf{u}}) + e_3 - (\vec{\mathbf{s}}^T\mathbf{B} + \vec{\mathbf{e_1}}^{\mathbf{T}})\vec{\mathbf{r}} \rfloor \\ &= \lceil \underbrace{\vec{\mathbf{s}}^T\vec{\mathbf{u}}}_{\text{uniform}} + \underbrace{\vec{\mathbf{s}}^T\vec{\mathbf{e_2}} + e_3 - \vec{\mathbf{e_1}}^T\vec{\mathbf{r}}}_{\text{small}} \rfloor . \end{aligned}$$

Where the $\vec{\mathbf{s}}^T\vec{\mathbf{u}}$ term is uniformly random. The sum of the error terms $\vec{\mathbf{s}}^T\vec{\mathbf{e_2}} + e_3 - \vec{\mathbf{e_1}}^T\vec{\mathbf{r}}$ has ℓ_2-norm at most $2d\beta_s^2 + \beta_e$. Correctness fails if the error terms affect the rounding of one of the coefficients. If a coefficient of the error terms has value e, then the probability that $\lceil a + e\rfloor_p \neq \lceil a\rfloor_p$ for a uniformly random $a \in \mathbb{Z}_q$ is at most $|e|p/q$, because there are p boundaries where the value of $\lceil\cdot\rfloor_p$ changes, and for each boundary, the probability that it ends up between a and $a + e$ is $|e|/q$. By a union bound over all coefficients, it follows that the probability that an error with an ℓ_1-norm of x affects the rounding of a random element of $\mathcal{R}_q$ is at most px/q. Since in general $\|\cdot\|_1 \le \sqrt{d}\,\|\cdot\|$, it follows that the verification fails with probability at most

$$\mathsf{negl}(\lambda) + \sqrt{d}\left(2d\beta_s^2 + \beta_e\right)p/q\,,$$

which is negligible by assumption.

anonymity. We prove anonymity through a sequence of games.

Signer		User
$\vec{\mathbf{s}} \in \mathcal{R}_q^n, K \in \{0,1\}^\lambda, \vec{\mathbf{e}}_1 \leftarrow D_e^n$		$\mathbf{B}, \vec{\mathbf{t}}^T = \vec{\mathbf{s}}^T\mathbf{B} + \vec{\mathbf{e_1}}^{\mathbf{T}}, \mathsf{pk}$
		$\vec{\mathbf{r}}, \vec{\mathbf{e_2}} \leftarrow D_r \in \mathcal{R}_q^n : \lVert \vec{\mathbf{r}} \rVert \le \beta_s, \lVert \vec{\mathbf{e}}_2 \rVert \le \beta_s,$ $\mathsf{ct} := \mathsf{enc}(\mathsf{pk}, (\vec{\mathbf{r}}, \vec{\mathbf{e_2}}, \mu))$ $\vec{\mathbf{c}} := \mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{e_2}} + \mathsf{H}_{\mathcal{R}_q^n}\left(\mu, \mathsf{H}_{\{0,1\}^{2\lambda}}(\vec{\mathbf{r}}, \vec{\mathbf{e_2}})\right)$
	$\xleftarrow{\vec{\mathbf{c}}, \mathsf{ct}, \pi_1}$	$\pi_1 := \mathsf{NIZK}_1.\mathsf{P}\left((\vec{\mathbf{r}}, \vec{\mathbf{e_2}}, \mu) \;\middle\vert\; \begin{array}{c} \lVert \vec{\mathbf{r}} \rVert \le \beta_s, \lVert \vec{\mathbf{e_2}} \rVert \le \beta_s, \\ \mathsf{ct} = \mathsf{enc}(\mathsf{pk}, (\vec{\mathbf{r}}, \vec{\mathbf{e_2}}, \mu)) \\ \rho := \mathsf{H}_{\{0,1\}^{2\lambda}}(\vec{\mathbf{r}}, \vec{\mathbf{e_2}}) \\ \vec{\mathbf{c}} := \mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{e_2}} + \mathsf{H}_{\mathcal{R}_q^n}(\mu, \rho) \end{array} \right)$
If $\mathsf{NIZK}_1.\mathsf{V}(\pi_1) = \bot$**:** Return $\bot$ $e_3 := \mathsf{H}_{D_s}(K, \vec{\mathbf{c}}) \in \mathcal{R}_q$ $h := \vec{\mathbf{s}}^T\vec{\mathbf{c}} + e_3$		
$\pi_2 := \mathsf{NIZK}_2.\mathsf{P}\left((\vec{\mathbf{s}}, \vec{\mathbf{e_1}}, e_3) \;\middle\vert\; \begin{array}{c} \lVert \vec{\mathbf{s}} \rVert < \beta_s, \\ \lVert \vec{\mathbf{e_1}} \rVert < \beta_s, \\ \lVert e_3 \rVert < \beta_e, \\ h = \vec{\mathbf{s}}^T\vec{\mathbf{c}} + e_3 \\ \vec{\mathbf{t}}^T = \vec{\mathbf{s}}^T\mathbf{B} + \vec{\mathbf{e_1}} \end{array} \right)$	$\xrightarrow{h, \pi_2}$	
		If $\mathsf{NIZK}_2.\mathsf{V}(\pi_2) = \bot$**:** Return $\bot$ $\rho := \mathsf{H}_{\{0,1\}^{2\lambda}}(\vec{\mathbf{r}}, \vec{\mathbf{e}}_2)$ $v := \lceil h - \vec{\mathbf{t}}^T\vec{\mathbf{r}} \rfloor$ Return $\mathsf{sig} = (\rho, v)$
Verify($\mathsf{sig} = (\rho, v), \mu$) $\vec{\mathbf{u}} := \mathsf{H}_{\mathcal{R}_q^n}(\mu, \rho)$ Check that $v = \lceil \vec{\mathbf{s}}^T\vec{\mathbf{u}} \rfloor$		

Fig. 5: Our keyed-verification blind signature scheme, parametrized by the values and subroutines given in Section 4.1. $\mathbf{B}$ is a publicly known random matrix $\in \mathcal{R}_q^{n\times n}$ and pk is a public key for PKE for which no corresponding secret key is not known to anybody.

- The first game, Game_0, is precicely the anonymity game of Definition 2.2.
- In the next game, Game_1, the challenger simulates the random oracle and uses the ZK-simulator to produce the proof π_1, rather than computing the proof honestly. The ZK property of NIZK_1 implies that the distinguishing advantage of the adversary in Game_1 differs from its distinguishing advantage in Game_0 by at most a negligible amount.
- In Game_2, the challenger encrypts a dummy message instead of $(\vec{\mathbf{r}}, \vec{\mathbf{e_2}}, \mu)$. It follows from the $\mathsf{IND\text{-}CPA}$ security of the PKE, that the distinguishing advantage of the adversary in Game_2 differs from its distinguishing advantage in Game_1 by at most a negligible amount.
- In Game_3, the challenger uses the adaptive knowledge soundness of NIZK_2 two times to extract $\vec{\mathbf{s}}^b, \vec{\mathbf{e_1}}^b, e_3^b$ for $b \in \{0,1\}$ such that $\lVert \vec{\mathbf{s}}^b \rVert \le \beta_s$ and $\lVert \vec{\mathbf{e}}_1^b \rVert \le \beta_s$, $\lVert e_3^b \rVert \le \beta_e$, $\vec{\mathbf{t}}^T = \vec{\mathbf{s}}^{bT}\mathbf{B} + \vec{\mathbf{e_1}}^{\mathbf{bT}}$, and $h^b = \vec{\mathbf{s}}^{bT}\vec{\mathbf{c}}^b + e_3^b$ for $b \in \{0,1\}$ in expected polynomial time. If $\vec{\mathbf{s}}^0 \neq \vec{\mathbf{s}}^1$, then the challenger can output the MSIS solution $(\vec{\mathbf{s}}^0 - \vec{\mathbf{s}}^1, \vec{\mathbf{e}}_1^0 - \vec{\mathbf{e}}_1^1)$ for the matrix $\big(\mathbf{B}\;\mathbf{1}\big)$, which contradicts the assumed hardness of $\mathsf{MSIS}_{n,2n,2\sqrt{2}\beta_s}$. So with all but a negligible probability $\vec{\mathbf{s}} = \vec{\mathbf{s}}'$. If the extractor fails or if $\vec{\mathbf{s}} \neq \vec{\mathbf{s}}'$ the game aborts and output s a random bit (which happens with negligible probability), otherwise the game proceeds as in Game_2, so the distinguishing advantage of the adversary in Game_3 differs from its distinguishing advantage in Game_2 by at most a negligible amount.
- In Game_4, if $\mathsf{sig}^0 \neq \bot$ and $\mathsf{sig}^1 \neq \bot$ the challenger uses the extracted value of $\vec{\mathbf{s}}$ to replace both signatures by $(\rho^b, v^b = \lceil \vec{\mathbf{s}}\mathsf{H}_{\mathcal{R}_q^n}(\mu^b, \rho^b) \rfloor)$ for $b \in \{0,1\}$, regardless of the values h^b sent by the

adversary. This changes the signatures with probability at most $\sqrt{d}\left(2d\beta_s^2 + \beta_e\right)p/q$, which is negligible by assumption, so the distinguishing advantage of the adversary in Game_4 differs from its distinguishing advantage in Game_3 by at most a negligible amount.

- In Game_5, the challenger sets $\vec{\mathbf{c}}^b$ and ρ uniformly at random, instead of $\vec{\mathbf{c}}^b = \mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{e}}_\mathbf{2} + \rho^\mathbf{b}$ and $\rho = \mathsf{H}_{\{0,1\}^{2\lambda}}(\vec{\mathbf{r}}, \vec{\mathbf{e}}_2)$. This does only affect the distinguishing advantage of the adversary by at most a negligible amount, because otherwise the adversary could be used to break the MLWE_{n,D_e} problem if we model $\mathsf{H}_{\{0,1\}^\lambda}$ as a random oracle, as argued in the proof of Theorem 3.3.

Finally, note that the distinguishing advantage of the adversary in Game_5 is exactly zero, since the bit b chosen by the challenger is now information-theoretically hidden from the adversary. Therefore, the distinguishing advantage of the adversary for Game_0 must have been negligible.

Before we prove the one-more unforgeability of our keyed-verification blind signature, we introduce a new hard problem, *hashed learning with errors*. Then we prove that the hashed learning with errors problem is as hard as the usual learning with errors problem in the random oracle model, and we reduce the one-more unforgeability of our blind signature to the hardness of the hashed learning with errors problem.

Definition 4.2 (hashed module learning with errors). *Let $\mathsf{H}_{\mathcal{R}_q^n}$ be a hash function that outputs elements of $\mathcal{R}_q^n$, and let H_{D_e} be some hash function that samples elements from $\mathcal{R}_q$ according to the error distribution D_e. Let $\vec{\mathbf{s}} \in \mathcal{R}_q^n$ be some vector over $\mathcal{R}_q$, and $K \in \{0,1\}^\lambda$ a bitstring, then we define an oracle $\mathcal{O}_{\vec{\mathbf{s}},K,\mathsf{H}_{\mathcal{R}_q^n},\mathsf{H}_{D_e}}$ that when queried on a message $\mu \in \{0,1\}^*$, returns $\vec{\mathbf{s}}^T\mathsf{H}_{\mathcal{R}_q^n}(\mu) + \mathsf{H}_{D_e}(K,\mu)$.*

The hashed module learning with errors ($\mathsf{HMLWE}_{\mathsf{H}_{\mathcal{R}_q^n},\mathsf{H}_{D_e}}$) problem asks to distinguish $\mathcal{O}_{\vec{\mathbf{s}},K,\mathsf{H}_{\mathcal{R}_q^n},\mathsf{H}_{D_e}}$ for some $\vec{\mathbf{s}}'$ drawn from D_e^n and $K \xleftarrow{\$} \{0,1\}^\lambda$ from a random oracle that outputs elements of $\mathcal{R}_q$ uniformly at random.

We will prove that our blind signature scheme is secure, assuming the HMLWE problem is hard from some concrete hash function such as SHA-2 or SHA-3. To argue why this assumption is reasonable, notice that if the hash functions are modeled as random oracles, then the $\mathsf{HMLWE}_{\mathsf{H}_1,\mathsf{H}_2}$ is as hard as the standard MLWE_{n,D_e} problem.

Lemma 4.3. *The $\mathsf{HMLWE}_{\mathsf{H}_{\mathcal{R}_q^n},\mathsf{H}_{D_e}}$ problem is at least as hard as the MLWE_{n,D_e} problem with an unlimited amount of samples if $\mathsf{H}_{\mathcal{R}_q^n}$ and H_{D_e} are modeled as random oracles.*

Proof. We give a reduction from MLWE_{n,D_e} to $\mathsf{HMLWE}_{\mathsf{H}_{\mathcal{R}_q^n},\mathsf{H}_{D_e}}$ that works as follows: each time the HMLWE adversary makes a query for a fresh message μ, the reduction asks the MLWE challenger for a fresh MLWE sample $(\vec{\mathbf{a}}, b)$, and stores $(\mu, \vec{\mathbf{a}}, b)$ in a table. If the query was a random oracle query for $\mathsf{H}_{\mathcal{R}_q^n}$, then the reduction answers with $\vec{\mathbf{a}}$, and if the query was a HMLWE query, then it answers with b. If the HMLWE adversary makes a query for a message μ for which there already is an entry $(\mu, \vec{\mathbf{a}}, b)$ in the table, then the reduction returns $\vec{\mathbf{a}}$ or b if the query was a random oracle query for H_1 or a HMLWE query respectively. The reduction simulates the random oracle H_{D_e} honestly.

Let $K \xleftarrow{\$} \{0,1\}^\lambda$ be some key. If the reduction is receiving real MLWE samples with a secret $\vec{\mathbf{s}}$, then the reduction behaves as $(\mathsf{H}_{\mathcal{R}_q^n}, \mathsf{H}_{D_e}, \mathcal{O}_{\vec{\mathbf{s}},K,\mathsf{H}_{\mathcal{R}_q^n},\mathsf{H}_{D_e}}$, as long as the adversary does not query H_{D_e} on an input K, μ, which happens only with a negligible probability since K has high min-entropy and is information-theoretically hidden from the adversary. If the reduction is receiving random samples, then the reduction behaves as three random oracles. Therefore the distinguishing advantage of the reduction against MLWE equals the distinguishing advantage of the adversary against HMLWE up to a negligible amount due to the adversary guessing K.

We will also assume that the function

$$f_{\mathbf{B}} : (\mu, \vec{\mathbf{r}}, \vec{\mathbf{e}}_2) \mapsto \mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{e}}_\mathbf{2} + \mathsf{H}_{\mathcal{R}_q^n}(\mu, \mathsf{H}_{\{0,1\}^{2\lambda}}(\vec{\mathbf{r}}, \vec{\mathbf{e}}_2))$$

is collision-resistant, which is a reasonable assumption because it is true if $\mathsf{H}_{\mathcal{R}_q^n}$ and $\mathsf{H}_{\{0,1\}^{2\lambda}}$ are modeled as random oracles. Now we are ready to prove the one-more unforgeability of our keyed-verification blind signature.

Theorem 4.4. *Assume the* $\mathsf{HMLWE}_{\mathsf{H}_{\mathcal{R}_q^n},\mathsf{H}_{D_e}}$ *problem is hard, that the function* $f_{\mathbf{B}}$ *as defined above is collision-resistant, and that* PKE *is correct. Furthermore, assume that* NIZK_1 *is adaptively sound, and* NIZK_2 *is zero-knowledge, that* $\sqrt{d}\beta_e p/q$ *is a negligible function, and that* $\lceil u \rfloor_p$ *has high min-entropy for* $u \xleftarrow{\$} \mathcal{R}_q$*. Then the keyed-verification blind signature of Figure 5 is one-more unforgeable.*

Proof. Let $\mathcal{A}$ be a polynomial-time adversary. We prove that $\mathcal{A}$ only has a negligible probability of winning the one-more unforgeability game using a sequence of games. Let ε_i be the winning probability of the adversary $\mathcal{A}$ in Game_i.

- The first game, Game_0 is exactly the one-more unforgeabilty game of Definition 2.3, played by $\mathcal{A}$.
- In the next game, Game_1 the challenger runs $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{PKE.keygen}$ and replaces the random public key in the crs by pk. The public key in the CRS is indistinguishable from the one generated by $\mathsf{PKE.keygen}$, so $\varepsilon_0 - \varepsilon_1 = \mathsf{negl}(\lambda)$.
- In Game_2, the challenger simulates the random oracle queries for the second NIZK, and uses the zero-knowledge simulator of NIZK_2 to generate the π_2 proofs in the resposes to the signing queries. By the zero-knowledge property, the simulated proofs are indistinguishable from real proofs, so $\varepsilon_1 - \varepsilon_2 = \mathsf{negl}(\lambda)$.
- In Game_3, the challenger uses sk to decrypt ct each time the adversary makes a signing query $(\vec{\mathbf{c}}, \mathsf{ct}, \pi_1)$, and the game aborts if $\mathsf{dec}(\mathsf{sk}, \mathsf{ct})$ does not return $\vec{\mathbf{r}}, \vec{\mathbf{e_2}}, \mu$ such that $\|\vec{\mathbf{r}}\|, \|\vec{\mathbf{e_2}}\| \leq \beta_s$, and $\vec{\mathbf{c}} = \mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{e_2}} + \mathsf{H}_{\mathcal{R}_q^n}(\mu, \mathsf{H}_{\{0,1\}^{2\lambda}}(\vec{\mathbf{r}}, \vec{\mathbf{e}}_2))$. It follows from the adaptive soundness of NIZK_1 and the correctness of PKE, that the abort only occurs with negligible probability. So, $\varepsilon_2 - \varepsilon_3 = \mathsf{negl}(\lambda)$.
- In Game_4, the challenger changes the way it samples e_3 when answering signing queries, using the information they get from decrypting ct. Instead of $e_3 \leftarrow \mathsf{H}_{D_e}(K, \vec{\mathbf{c}})$ it puts $e_3 \leftarrow \mathsf{H}_{S_e}(K, \mu, \vec{\mathbf{r}}, \vec{\mathbf{e}}_2)$. This does not alter the view of the adversary if the adversary does not make queries of the form $(K, \star)$ (which happens with overwhelming probability because K has high min-entropy), and if the adversary does not make queries with the same $\vec{\mathbf{c}}$, but different $(\mu, \vec{\mathbf{r}}, \vec{\mathbf{e}}_2)$. The latter can only happen with negligible probability since we assume the function $f_{\mathbf{B}}$ is collision-resistant.
- In Game_5 the challenger simulates for itself the $\mathcal{O}_{\vec{\mathbf{s}},K,\mathsf{H}_{\mathcal{R}_q^n},\mathsf{H}_{D_e}}$ oracle (from now on we omit the subscript for brevity) and queries it on $1, \ldots, n$ to produce $\vec{\mathbf{t}}^T$. It calls $H_{\mathcal{R}_q^n}$ on inputs $1, \ldots, n$ to get the n collumns of $\mathbf{B}$. The challenger answers signature queries with $h = \mathbf{B}\vec{\mathbf{r}} + \vec{\mathbf{e_2}} + \mathcal{O}(\mu, \mathsf{H}_{\{\mathbf{0},\mathbf{1}\}^{2\lambda}}(\vec{\mathbf{r}}, \vec{\mathbf{e_2}}))$. It answers verification oracle queries $(\mu, (\rho, v))$ with 1 if $v = \lceil \mathcal{O}(\mu, \rho) \rfloor$ and with 0 otherwise. We almost haven't changed the game, just rewritten it in terms of the oracle $\mathcal{O}$. The only difference between Game_4 and Game_5 is that the verification oracle in Game_4 checks $v = \lceil \vec{\mathbf{s}}^T \vec{\mathbf{u}} \rfloor$ and in Game_5 it checks $v = \lceil \vec{\mathbf{s}}^T \vec{\mathbf{u}} + \mathsf{H}_{D_e}(K, \mu, \rho) \rfloor$. Since $\sqrt{d}\beta_e p/q$ is negligible, the probability that this changes the outcome of any of the at most polynomially many verification queries is negligible, and therefore $\varepsilon_4 - \varepsilon_5 = \mathsf{negl}(\lambda)$.
- In Game_6 the challenger behaves like in Game_5 except that it simulates a random oracle and uses it instead of the $\mathcal{O}$ oracle. We can construct a polynomial-time distinguisher $\mathcal{B}^{\mathcal{A}}$ that distinguishes between $\mathcal{O}$ and a random oracle with advantage $|\varepsilon_6 - \varepsilon_5|$, so it follows from the assumed hardness of HMLWE that $\varepsilon_6 - \varepsilon_5 = \mathsf{negl}(\lambda)$.

In Game_6, we say that a verification query $(\mu, \mathsf{sig} = (\rho, v))$ is "fresh" if there was no signing query that made the challenger sample $\mathsf{H}_{\mathcal{R}_q}$ on input (μ, ρ). The probability that a fresh verification query results in 1 is the probability that $v = \lceil u \rfloor$ for $u \xleftarrow{\$} \mathcal{R}_q$. Since $\lceil u \rfloor$ is assumed to have high

Parameter	Instantiation	Explanation
λ	128	security parameter
d	64	degree of the cyclotomic ring $\mathcal{R}$
q	$\approx 2^{152}$	modulus
p	4	modulus after rounding
n	93	MLWE-rank
D_e	D_1	discrete Gaussian distribution with standard deviation 1
β_s	78	norm bound on $\vec{\mathbf{s}}$, $\vec{\mathbf{e}}_1$, $\vec{\mathbf{r}}$, $\vec{\mathbf{e}}_2$
β_e	8	norm bound on e_3
$\lvert\mathsf{pk}\rvert$	110 KB	public key size
$\lvert\mathsf{sig}\rvert$	48 B	signature size

Table 4: Overview of parameters and notation for our keyed-verification blind signature.

min-entropy, the probability that any of the (at most polynomially many) fresh verification queries results in a 1 is negligible. Without loss of generality, the adversary outputs only signatures for which it queried the verification oracle. (If this is not already the case we can define a new adversary that verifies its signatures before outputting them.) If the messages in the $(Q+1)$ message-signature pairs returned by the verifier are pairwise distinct, then at least one of the $Q+1$ verification queries is fresh, since the adversary only makes Q queries to the signing oracle. Therefore, the probability that all the $Q+1$ signatures are valid is negligible.

Since each game transition decreases the winning probability by at most a negligible amount, and in the last game Game_6 the winning probability is negligible, it follows that the adversary only has a negligible probability of winning the one-more unforgeability game Game_0.

4.3 Parameter selection

To achieve a security level of 2^λ We require that our parameters from Section 4.1 satisfy the following constraints:

- It requires 2^λ resources to solve the MLWE_{n,D_e} problem,
- $\sqrt{d}\left(2d\beta_s^2 + \beta_e\right) p/q < 2^{-\lambda}$ to ensure that the rounding function is unaffected by the errors with overwhelming probability, and
- $d \log(p) \gtrsim \lambda$ to ensure that $\lceil u \rfloor$ has λ bits of min-entropy for uniformly random $u \xleftarrow{\$} \mathcal{R}_q$.
- $\mathsf{NIZK}_1, \mathsf{NIZK}_2$, and PKE reach the security level 2^λ.

To satisfy these constraints for $\lambda = 128$, one can use parameters as in Table 4: a ring $\mathcal{R}_q$ of dimension $d = 64$ with 152-bit modulus q, MLWE rank $n = 93$, and errors drawn from a distribution with standard deviation $e = 1$ to get a root Hermite factor of $\delta = 1.00445$, as per the LWE estimator by Albrecht et al. [APS15]. We put $p = 4$, $\beta_s = 78$, and $\beta_e = 8$. In this case, $\vec{\mathbf{c}}$ is $dn \log q$ bits large, which amounts to 110 KB, and the size of $(\vec{\mathbf{r}}, \vec{\mathbf{e}}_2)$ is roughly 40 KB. This means that NIZK_1 needs to prove that a hash function with 40 KB of input and 110 KB of output is computed correctly. The size of the signature $\mathsf{sig} = (\rho, v)$ is $2\lambda + d \log p$ bits which is only 48 Bytes.

References

ADDS21. Martin R. Albrecht, Alex Davidson, Amit Deo, and Nigel P. Smart. Round-optimal verifiable oblivious pseudorandom functions from ideal lattices. In *Public Key Cryptography (2)*, volume 12711 of *Lecture Notes in Computer Science*, pages 261–289. Springer, 2021. https://eprint.iacr.org/2019/1271.

AGHS13. Shweta Agrawal, Craig Gentry, Shai Halevi, and Amit Sahai. Discrete gaussian leftover hash lemma over infinite domains. In *ASIACRYPT (1)*, volume 8269 of *Lecture Notes in Computer Science*, pages 97–116. Springer, 2013.

AKSY22. Shweta Agrawal, Elena Kirshanova, Damien Stehlé, and Anshu Yadav. Practical, round-optimal lattice-based blind signatures. In *CCS*, pages 39–53. ACM, 2022.

ALS20. Thomas Attema, Vadim Lyubashevsky, and Gregor Seiler. Practical product proofs for lattice commitments. In *CRYPTO (2)*, volume 12171 of *Lecture Notes in Computer Science*, pages 470–499. Springer, 2020.

APS15. Martin R Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015.

BFH$^+$20. Rishabh Bhadauria, Zhiyong Fang, Carmit Hazay, Muthuramakrishnan Venkitasubramaniam, Tiancheng Xie, and Yupeng Zhang. Ligero++: A new optimized sublinear IOP. In *CCS*, pages 2025–2038. ACM, 2020.

BGL20. Eli Ben-Sasson, Lior Goldberg, and David Levit. STARK friendly hash - survey and recommendation. *IACR Cryptol. ePrint Arch.*, page 948, 2020.

BLL$^+$21. Fabrice Benhamouda, Tancrède Lepoint, Julian Loss, Michele Orrù, and Mariana Raykova. On the (in)security of ROS. In *EUROCRYPT (1)*, volume 12696 of *Lecture Notes in Computer Science*, pages 33–53. Springer, 2021.

BLS19. Jonathan Bootle, Vadim Lyubashevsky, and Gregor Seiler. Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. In *CRYPTO (1)*, volume 11692 of *Lecture Notes in Computer Science*, pages 176–202. Springer, 2019.

BP14. Abhishek Banerjee and Chris Peikert. New and improved key-homomorphic pseudorandom functions. In *CRYPTO (1)*, volume 8616 of *Lecture Notes in Computer Science*, pages 353–370. Springer, 2014. https://eprint.iacr.org/2014/074.

BS22. Ward Beullens and Gregor Seiler. LaBRADOR: Compact proofs for R1CS from module-sis. *IACR Cryptol. ePrint Arch.*, page 1341, 2022.

Cha82. David Chaum. Blind signatures for untraceable payments. In *CRYPTO*, pages 199–203. Plenum Press, New York, 1982.

CHL22. Sílvia Casacuberta, Julia Hesse, and Anja Lehmann. SoK: Oblivious pseudorandom functions. In *EuroS&P*, pages 625–646. IEEE, 2022. https://eprint.iacr.org/2022/302.

CM22. David Chaum and Thomas Moser. eCash 2.0: Inalienably private and quantum-resistant to counterfeiting,, 2022. https://chaum.com/publications/#iTK6j8.

CMZ14. Melissa Chase, Sarah Meiklejohn, and Greg Zaverucha. Algebraic MACs and keyed-verification anonymous credentials. In *CCS*, pages 1205–1216. ACM, 2014.

DLP14. Léo Ducas, Vadim Lyubashevsky, and Thomas Prest. Efficient identity-based encryption over NTRU lattices. In *ASIACRYPT*, pages 22–41, 2014.

dPK22. Rafaël del Pino and Shuichi Katsumata. A new framework for more efficient round-optimal lattice-based (partially) blind signature via trapdoor sampling. In *CRYPTO (2)*, volume 13508 of *Lecture Notes in Computer Science*, pages 306–336. Springer, 2022.

ENS20. Muhammed F. Esgin, Ngoc Khanh Nguyen, and Gregor Seiler. Practical exact proofs from lattices: New techniques to exploit fully-splitting rings. In *ASIACRYPT (2)*, pages 259–288, 2020.

Fis06. Marc Fischlin. Round-optimal composable blind signatures in the common reference string model. In *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 60–77. Springer, 2006.

GN08. Nicolas Gama and Phong Q. Nguyen. Predicting lattice reduction. In *EUROCRYPT*, pages 31–51, 2008.

GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.

LN22. Vadim Lyubashevsky and Ngoc Khanh Nguyen. BLOOM: bimodal lattice one-out-of-many proofs and applications. *IACR Cryptol. ePrint Arch.*, page 1307, 2022.

LNP22. Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In *CRYPTO (2)*, volume 13508 of *Lecture Notes in Computer Science*, pages 71–101. Springer, 2022.

LNS20. Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. Practical lattice-based zero-knowledge proofs for integer relations. In *CCS*, pages 1051–1070. ACM, 2020.

LNS21. Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. Shorter lattice-based zero-knowledge proofs via one-time commitments. In *Public Key Cryptography (1)*, pages 215–241. Springer, 2021.

MR09. Daniele Micciancio and Oded Regev. Lattice-based cryptography. In *Post-quantum cryptography*, pages 147–191. Springer, 2009.

NIS22. NIST. Status report on the third round of the nist post-quantum cryptography standardization process, 2022. https://csrc.nist.gov/publications/detail/nistir/8413/final.

NSA22. NSA. Announcing the commercial national security algorithm suite 2.0, 2022. https://media.defense.gov/2022/Sep/07/2003071834/-1/-1/0/CSA_CNSA_2.0_ALGORITHMS_.PDF.

PFH$^+$17. Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, , and Zhenfei Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2017. https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.