Compact Privacy Protocols from Post-Quantum and Timed Classical Assumptions

Jonathan Bootle[1], Anja Lehmann[2⋆], Vadim Lyubashevsky[1], and Gregor Seiler[1,3]

[1] IBM Research – Zurich, Switzerland
[2] Hasso Plattner Institute, University of Potsdam, Germany
[3] ETH Zurich, Switzerland

Abstract. While basic lattice-based primitives like encryption and digital signature schemes are already fairly short, more advanced privacy-preserving protocols (e.g. group signatures) that are believed to be post-quantum secure have outputs of at least several hundred kilobytes. In this paper, we propose a framework for building privacy protocols with significantly smaller parameter sizes whose secrecy is based on post-quantum assumptions, but soundness additionally assumes that some classical assumption, e.g., the discrete logarithm problem (DLP), is hard to break within a short amount of time.

The main ingredients of our constructions are statistical zero-knowledge proofs of knowledge for certain relations, whose soundness rely on the hardness of solving the discrete logarithm problem for a *fresh* DLP instance per proof. This notion has recently been described by the term *quantum annoyance*. Using such proofs, while also enforcing that they be completed in a fixed amount of time, we then show how to construct privacy-preserving primitives such as (dynamic) group signatures and DAA schemes, where soundness is based on the hardness of the "timed" discrete logarithm problem and SIS. The outputs of our schemes are significantly shorter ($\approx 30X$) than purely lattice-based schemes.

1 Introduction

Lattice cryptography is a particularly attractive post-quantum alternative to classical cryptographic schemes based on factoring and discrete log. Its main appeal is that one can build basic primitives, such as encryption and digital signature schemes, with relatively short outputs (1 - 3 KB) with the added bonus of sometimes being faster than the classical analogues. When one looks at more advanced privacy-preserving primitives such as group signatures, verifiable encryption, etc., the situation is considerably less attractive. For example, while group signatures based on elliptic curve pairings are only 160 Bytes [34], the smallest lattice-based group signatures, in which keys don't grow linearly with the number of group members, are approximately 600 KB [16].

Despite a considerable amount of research, it's looking very unlikely that even basic privacy-preserving primitives will be reduced to sizes of less than a

⋆ Work done while being at IBM Research - Zurich.

few hundred kilobytes. This is due to the general approach used in constructing privacy-preserving schemes, such as group signatures, which lacks efficient lattice-based building blocks. The authority gives out a secret key to a particular user by signing the user's identity. To authenticate himself, the user then produces a ZKPoK of the signature on his identity.[4] Because creating an efficient zero-knowledge proof generally requires algebraic structure in the underlying statement, one generally uses standard-model (rather than one secure in the random oracle model) digital signature schemes for the authority's signature rather than rely on schemes that use a hash function modeled as a random oracle. And it is this requirement of a standard-model signature scheme that is the main culprit in the large output sizes of privacy-preserving schemes constructed in the above manner.

In this work we propose a framework for a middle-ground solution which addresses some of the main security problems posed by the eventual coming of quantum computers. One of the biggest concerns today is that communication in the pre-quantum world can be harvested and then eventually decrypted when quantum computers are eventually built. The main result of this paper is a framework for constructing compact privacy schemes where secrecy is either information-theoretic or based on post-quantum assumptions, while soundness is based on classical ones. Because only the soundness is classical, our schemes are not susceptible to the aforementioned harvesting attacks, and are therefore safe to use in the pre-quantum era.

If full-fledged quantum computers arrive and there are still no acceptably compact fully quantum-safe privacy schemes, then one can still continue using our schemes in certain situations. Firstly, they are *quantum annoying* (c.f. [22]), in the sense that breaking soundness requires solving a fresh discrete logarithm instance for each new forgery. This may be good enough in instances where the forgery payoff is less than the cost to use a quantum computer for the attack. In addition, we show that our schemes can be made to satisfy a stronger security notion by relying on "timed" versions of classical assumptions in which the prover must produce a response in a limited amount of time. This implies that a successful cheating prover can be used to solve the underlying problem in a fixed time interval, which may remain a difficult problem well into the post-quantum era (see the discussions in e.g. [25, 23, 29]).

1.1 Our Techniques

Since the main culprit for inefficient lattice-based privacy schemes are standard-model signatures, we propose avoiding them altogether, and instead construct a proof of knowledge of (possibly short) vectors x, y, s, when given public matrices

[4] If the user wants to sign a message, then he transforms the interactive authentication protocol into a non-interactive one via the Fiat-Shamir framework and uses the message to create the challenge.

/ vectors A, B, C, z over some polynomial ring, satisfying

$$[A\ B] \cdot \begin{bmatrix} x \\ \tilde{F}(y) \end{bmatrix} = z \wedge H(F(y)) = Cs. \tag{1}$$

We then show that such proofs are enough for constructing privacy-preserving primitives such as group signatures and DAA schemes. In some constructions, $F = \tilde{F}$ will be (the same) one-way functions, while in others F will be a one-way function while $\tilde{F}$ will just be the identity. The function H is a cryptographic hash function.

The soundness of our proof is based on the assumed intractability of the discrete logarithm problem. More precisely, the prover shows that he either knows the (short) solution x, y, s satisfying the above relation (which means he knows a solution to a lattice problem), or he is able to find $e_i \in \mathbb{Z}$ satisfying $\prod g_i^{e_i} = 1$ for random generators g_i of some group.[5] While the discrete logarithm problem is not quantum secure, the only place in which it is used in our constructions is for guaranteeing the soundness of the zero-knowledge proofs. The zero-knowledge property itself is statistical and hence the privacy of the secrets is not affected by the (quantum) power of the adversary.

By letting the generators g_i be freshly chosen by the verifier (or some randomness beacon) at the time the proof is started, the ZKP already becomes "quantum annoying" as for each forgery the (quantum) adversary must solve a new DLP instance. Moreover, if the running-time of the proof is restricted, i.e. the verifier will not accept the proof if it takes more than Δ time, then one can base the soundness of the proof on the "timed" discrete logarithm assumption, in which the relation $\prod g_i^{e_i} = 1$ must be solved in a fixed amount of time. If this amount of time is short, then this problem may remain hard even for quantum computers.

Proof Approach. Our zero-knowledge proof of (1) builds on the works in [12, 13, 17]. One of the contributions of [12, 13] was showing an efficient proof of the pre-image y satisfying $H(y) = z$, where H is an arbitrary circuit, based on the hardness of discrete log. These works also showed how to prove linear relations (in the exponent) of Pederesen commitments and applications to range proofs. The work of [17] utilized these techniques to give faster proofs of knowledge of a short vector x satisfying $Ax = z$ for a public matrix A and vector z over the polynomial ring $R_q = \mathbb{Z}_q[X]/(X^d + 1)$.

When $F = \tilde{F}$, we can rewrite (1) as

$$[A\ B] \cdot \begin{bmatrix} x \\ r \end{bmatrix} = z \wedge F(y) = r \wedge H(r) = t \wedge Cs = t, \tag{2}$$

and then proving (1) is equivalent to proving knowledge of x, y, s, r, t, with some of these needing to have coefficients in a certain range, satisfying the above.

[5] We will use multiplicative notation for discrete log.

Similarly, if $\tilde{F}$ is the identity, then one can rewrite (1) as

$$[A\ B] \cdot \begin{bmatrix} x \\ y \end{bmatrix} = z \wedge F(y) = r \wedge H(r) = t \wedge Cs = t, \tag{3}$$

The first part of the conjunction in both (2) and (3) can be proven using [17], while the last one is similar except the t is also secret. The other two parts can be proven using the techniques from [12, 13] applied to general circuits.

While the proofs in [13] are very compact, their main drawback is that the proof and verification time grows (more than) linearly in the number of gates in the circuit and proving the knowledge of a pre-image of a SHA-256 function (mapping 512 to 256 bits) takes approximately 20 seconds. In contrast, our schemes will require hash functions that map onto the space of a polynomial ring, which is around ten thousand bits. The proofs in [13] are based on the discrete logarithm assumption, which naturally lend themselves to proving statements over fields of large prime order. Therefore, we would like to use a hash-function built around arithmetic over such fields. MiMC [3] is a family of hash functions designed with precisely this in mind and we analyze the number of multiplication gates required for their evaluation.

Applications. We then show that proving (1) is enough for constructing schemes like group signatures and DAA schemes. While we only provide a few examples of what privacy-preserving schemes can be built from (1), there should be numerous other related schemes that can be constructed using this approach. Intuitively, constructing privacy-preserving primitives can be done by obtaining a signature on an identity from an issuer and proving knowledge of this signature in conjunction with supplementary information connected to the identity (c.f. [14]). One can then view the right part of (1) as a GPV-type signature scheme where the signature of the message (identity) $F(y)$ is s, and then the left side of the conjunction is a relation involving the message/identity y and some supplementary data x. The intuitive reason for why one may want to use $F(y)$ instead of y as the message is that one may wish to sometimes expose $F(y)$ but never expose her secret y. Using the image of the secret $F(y)$ as her identity, and then proving relations about the pre-image, allows the user to ascertain her knowledge of the secret without ever having to reveal it.

Our construction of a group signature scheme results in signatures of approximately 20 KB based on the hardness of standard lattice problems (i.e. NTRU and LWE) and the timed DL assumption. We also give a construction of a DAA scheme in in the full version of this paper, where the proofs are tweaked for the setting where attestation are generated jointly by a resource-constrained TPM and powerful host.

1.2 Related Work

In this paper, we demonstrate the feasibility of our framework by giving a concrete construction of a group signature scheme. Since the foundational work of

[5], there have been many constructions of such schemes with security based on various problems. The schemes based on the hardness of the discrete logarithm problem are compact, but not quantum-safe, while those based on the hardness of lattice problems are quantum-safe, but have large signatures and/or public keys. We give a comparison to our scheme in Table 1.

	Size		(Security) Properties		
Scheme	*gpk*	sign.	dynamic	non-frameability	quantum-safe
DS18 [18]	1.29	1.96	✓	✓	✘
BBS04 [11]	1.05	0.43	✘	✓	✘
dPLS [16]	120	580	✘	✘	✓
ESSLL [21]	9000	48	✓	✘	✓
This Work	5.5	20	✓	✓	(✓)

Table 1: Output sizes (in KB) of discrete log, lattice-based, and our group signatures. For pairing based schemes using CP5-663 pairing curve (128 bit security level, 256 bit order curve). The public key size (and opening time) in [21] grows linearly with the number of users. The size given in the table is for 1000 users.

1.3 Open Problems

The main result of our work is a framework for constructing privacy-preserving primitives based on lattice assumptions and the timed discrete logarithm problem. The advantage of this approach is that our protocols enjoy significantly shorter outputs than purely lattice-based (or any purely quantum-safe) schemes. The main drawback of our concrete instantiation of this framework, which uses Bulletproofs along with the MiMC hash function, is that the proofs require millions of group operations, which would take a substantial amount of time for an honest prover.

The most interesting open question is thus to obtain faster solutions which may involve constructing different hash functions along with compatible discrete-log proof systems. There is currently related work, sponsored by the Ethereum Foundation, to create a STARK-friendly hash function [7, 1], with several proposals already offering significant improvements over MiMC (e.g. [26, 4, 2]). Research into such hash functions is still in its infancy and there is reason to believe that we could eventually have hash functions which are very amenable to Bullet-proof style zero-knowledge proofs.

2 Preliminaries

In this section we introduce the building blocks needed for our privacy protocols.

Lattices. For $x, c \in \mathbb{R}^d$ and $\sigma \in \mathbb{R}^+$, we define the Gaussian function $\rho_{c,\sigma}(x) = \exp\left(\frac{-\|x-c\|^2}{2\sigma^2}\right)$, and for a lattice $\mathcal{L}$, we define the distribution $D_{\mathcal{L},c,\sigma}(x)$ to be 0 whenever $x \notin \mathcal{L}$ and $D_{\mathcal{L},c,\sigma}(x) = \frac{\rho_{c,\sigma}(x)}{\sum_{v\in\mathcal{L}} \rho_{c,\sigma}(v)}$. when $x \in \mathcal{L}$. When we omit the $\mathcal{L}$

from the above equation, it is assumed that the lattice is $\mathbb{Z}^d$ (where d is evident from context). Omitting the c implies that $c = 0$.

We will denote by R_q the polynomial ring $\mathbb{Z}_q[X]/(X^d + 1)$ and define the norm of elements in R_q as the norm of its coefficients. As an additive group, the polynomial ring $R = \mathbb{Z}[X]/(X^d + 1)$ has an obvious mapping to $\mathbb{Z}^d$ and so we can write $v \leftarrow D_\sigma$ to signify sampling a random centered element from R.

For polynomials $a, t \in R$, we can define a $2d$-dimensional shifted lattice[6]

$$\mathcal{L}^{\perp}_{a,t} = \{(s_1, s_2) \in R^2 \ : \ as_1 + s_2 = t \bmod q\}$$

and we define the distribution $D^{\perp}_{a,t,\sigma}(x)$ to be 0 whenever $x \notin \mathcal{L}^{\perp}_{a,t}$ and

$$D^{\perp}_{a,t,\sigma}(x) = \frac{\rho_\sigma(x)}{\sum\limits_{v \in \mathcal{L}^{\perp}_{a,t}} \rho_\sigma(v)} \tag{4}$$

In general, given a random $a, t \in R_q$, it is hard (as hard as the Ring-SIS problem [33, 32]) to sample according to $D^{\perp}_{a,t,\sigma}$ for small σ. One can do such sampling, however, when given a special trapdoor basis for the lattice $\mathcal{L}^{\perp}_{a,0}$. The smaller the vectors in the trapdoor, the smaller the σ can be in the distribution. A way to create a particularly small trapdoor can be done over NTRU lattices, in particular when $a = f/g$ for two polynomials f, g with small coefficients [27, 19, 35]. In particular, one can create an a, together with a trapdoor matrix T_a that allows one to sample (using a sampling algorithm from [24, 20]) from $D^{\perp}_{a,t,\sigma}$, for any $t \in R_q$, for $\sigma \approx 1.5\sqrt{q}$.

NTRU Signature. This trapdoor sampling algorithm almost directly leads to a rather compact digital signature scheme, in the random oracle model, based on the hardness of finding short vectors in NTRU lattices. The public key is $a = f/g$, while the signing key is T_a. If we model the hash function H_{R_q} as a random oracle, then to sign a message m, the signer samples $s_1, s_2 \leftarrow D^{\perp}_{a,H_{R_q}(m),\sigma}$ and outputs s_1, s_2 (or just s_1 since s_2 can be computed from s_1 and m) as the signature. The signature is valid if $\|s_1\|, \|s_2\| \leq 1.1\sigma\sqrt{d} = 1.65\sqrt{qd}$. In this paper, we will use MiMC as the cryptographic hash function.

NTRU Encryption. The key generation procedure of the NTRU encryption scheme [28] consists of creating two polynomials with small $(-1/0/1)$ coefficients $f, g \in R_q$ and outputting the public key as $h = f/g$ and secret key g. Encryption of a message m with $0/1$ coefficients involves generating an $r, e \in R_q$ with small coefficients and outputting the ciphertext $v = 2(hr + e) + m$. To decrypt v, one would compute $m = (vg \bmod q)/g \bmod 2$.

Lattice-Based Zero-Knowledge Proofs. Our protocols will use a combination of various lattice and discrete-log based zero-knowledge proofs from the literature.

In general, for a public $\vec{A} \in R_q^{n \times m}$ and $\vec{t} \in R_q^n$, the prover knows a secret $\vec{s} \in R_q^m$ with small coefficients such that $\vec{A}\vec{s} = \vec{t}$. Ideally, he would like to give a proof

[6] A *shifted lattice* is a lattice shifted by some vector v. Note that a shifted lattice does not have the property that the sum of any two vectors is in the shifted lattice.

of this $\vec{s}$, but such proofs are rather costly in their communication complexity. In some scenarios, however, the high cost may be acceptable. For example, joining a group (or registering a TPM) only needs to be done once and there are generally no strict restrictions on the time of communication complexity. An example of a proof in which a vector $\vec{s}$ is taken from a set with $\|\vec{s}\|_\infty \leq \alpha$ and the prover can produce a proof

$$\pi = \mathsf{ZKP}\{\vec{s} \ : \ \vec{A}\vec{s} = \vec{t}, \|\vec{s}\|_\infty \leq \alpha\} \tag{5}$$

is given in [30]. The proof is a variation of Stern's proof of knowledge of a near codeword [37] and each iteration of the scheme has soundness error 2/3. A more efficient proof that has soundness error $1/2d$ was introduced by Benhamouda et al. [8] where the prover uses his knowledge of $\vec{s}$ to prove the knowledge of a vector $\bar{\vec{s}}$ satisfying $\vec{A}\bar{\vec{s}} = 2\vec{t}$ where $\|\bar{\vec{s}}\| > \|\vec{s}\|$. In particular, given an $\vec{s} = \begin{bmatrix} s_1 \\ \dots \\ s_m \end{bmatrix}$ such that $\|s_i\| \leq \alpha$, it produces a zero-knowledge proof

$$\pi = \mathsf{ZKP}\{\bar{\vec{s}} \ : \ \vec{A}\bar{\vec{s}} = 2\vec{t}, \|\bar{\vec{s}}\| \leq 33\alpha d^{1.5} m\sqrt{\lambda}\} \tag{6}$$

In App. A we explicitly provide the prover and verifier algorithms for this relation since they were only given for an interactive, asymptotic version in [8].

Hash Functions with Efficient Proofs. For our privacy protocols we need a hash function that allows for efficient zero-knowledge proofs that a hash was correctly computed and that the prover knows a pre-image of the hash value. We will use zero-knowledge proofs based on the DL assumption, which naturally lend themselves to proving statements over fields of large prime order. Thus, we would like to use a hash function built around arithmetic over such fields.

MiMC [3] is a family of hash functions designed with precisely this in mind. MiMC hash functions are based on the sponge construction [10]. The construction works by cubing the input over the field, adding randomly chosen constant values, and repeating the process many times. We give a more detailed overview of the MiMC hash function and our parameter choices in App. B

3 Timed Zero-Knowledge Proofs

In this section we describe our idea of quantum-annoying and timed zero-knowledge proofs (ZKP), describe how they can be made non-interactive via a beacon service, and realized using a combination of lattice/bulletproofs.

More precisely, we consider ZKPs for generalized statements that prove an exact relation as in (5), but follow the proof system recently introduced in [17]. The proof system uses a CRS made up of random group elements $g_1, \dots, g_n$, and assuming the DL problem is hard, it allows to prove knowledge of a witness for various NP statements. For example, the protocol of [17] actually proves is that the prover knows a SIS solution $\mathbf{s}$ *or* a non-trivial discrete logarithm relation between $g_1, \dots, g_n$. The advantage of this technique is that the proofs

can be very short, but the disadvantage is that the running time of the prover and verifier is long (e.g. for typical parameters in [17] it was 10 - 20 seconds). Formally, the proof in [17] gives a proof of a disjunction

$$\pi = \mathsf{ZKP}\{\vec{s}, \{r_i\} \ : \ \mathtt{DLR}\,(\{g_i\}, \{r_i\}) = 1 \vee \vec{A}\vec{s} = \vec{t},\ \|\vec{s}\|_\infty \leq \alpha\}. \tag{7}$$

where g_i are public elements of some group G and $\vec{A}, \vec{t}$ are as before.

Generalizing the proof system of [17], we obtain zero-knowledge proofs of the following form in which the prover proves that they know a witness w for relation $\mathcal{R}_q$ or for relation $\mathcal{R}_c$: $\mathsf{ZKP}\{(w) : (x_c, w) \in \mathcal{R}_c \ \vee \ (x_q, w) \in \mathcal{R}_q\}$. In our proof systems, a witness for $\mathcal{R}_c$ will always be a non-trivial DL relation, and $\mathcal{R}_q$ will be the collection of statements and witnesses we are actually interested in.

Quantum Annoying & Timed Proofs. In this plain form, the soundness of the above proof relies on the weaker of both relations, i.e., the DL assumption even though it also proves a lattice relation. We can transform the proof into a quantum annoying version [22] by simply letting the verifier freshly choose g_i when the proof starts. As g_i are not longer long-term parameters, this forces the adversary to solve a fresh DL instance for every proof it wants to forge.

By requiring the prover to produce a proof within a short amount of time, we can further strengthen this approach such that the problem likely remains hard even for quantum computers (or is at least prohibitively expensive to solve). That is, the verifier only accepts a proof when the prover correctly responds within some fixed short time Δ. The soundness of our ZKP then even holds against a quantum adversary under the additional assumption that the DL problem is hard to solve within a short amount of time. We will refer to such an assumption as Δ-hardness. In App. C we provide a more formal treatment of such timed ZKPs and discuss their relation to quantum annoyance.

Non-interactive Timed Proofs. Finally, in our privacy protocols we want to use signature proofs of knowledge, i.e., non-interactive ZKPs that follow the Fiat-Shamir paradigm and "sign" a message m by including m in the challenge hash of the NIZK. To maintain the short-term validity aspect in this non-interactive form, we will rely on a beacon and time-stamp service $\mathcal{T}$.

This trusted entity $\mathcal{T}$ has a signing key pair (ssk, spk) and serves a double purpose: First, it regularly publishes signed tuples (t, b, σ) with $\sigma \xleftarrow{\$} \mathsf{Sign}(ssk, (t, b))$ for a time t and random beacon b. We will use b to deterministically generate fresh DL instances $(g_1, \ldots, g_n) \leftarrow G(b, m)$ where G is simply a hash function that outputs group elements of some group $\mathcal{C}$.

The prover first obtains such a timed beacon (t, b, σ), derives fresh DL instances and computes $\pi = \mathsf{NIZK}\{w, \{r_i\} \ : \ \mathtt{DLR}\,(\{g_i\}, \{r_i\}) = 1 \vee (x_q, w) \in \mathcal{R}_q\}(m)$. It then sends $h \leftarrow H(\pi)$ to $\mathcal{T}$ which will return $t', \sigma' \xleftarrow{\$} \mathsf{Sign}(ssk, (t', h))$, i.e., $\mathcal{T}$ time-stamps the hash h for time t'. The non-interactive timed proof output by the prover consists of $(\pi, t, t', b, \sigma, \sigma')$.

For the sake of brevity we use the following shorthand to refer to non-interactive timed proofs of such a form and with running time Δ:

$$\mathsf{ZKP}^{\Delta}_{\mathsf{DLR}}\{w_q : (x_q, w_q) \in \mathcal{R}_q\}.$$

Finally, we stress that while soundness is quantum-annoying or timed, we require the zero-knowledge property of the proof to hold statistically.

Building Timed ZKPs. To build our timed ZKPs needed for our group signature and DAA scheme, we use Bulletproofs [13] (instantiated with MiMC) and the proof system from [17] in a mostly black-box manner. The algorithms in our privacy protocols rely on complex relations made up of combinations of the DL, SIS and pre-image relations of the form $Func(f) := \{\mathbf{u} \in \{0,1\}^m, \mathbf{v} \in \{0,1\}^n : f(\mathbf{u}) = \mathbf{v}\}$. We describe how to realize such proofs from the mentioned proofs systems, and the tweaks that should be made, in App D.

4 Group Signature Scheme

A dynamic group signature allows users to sign messages on behalf of a group without revealing their individual identity. Group membership is managed by an issuer $\mathcal{I}$ that lets users $\mathcal{U}$ dynamically join the group. The anonymity of a user can be lifted through a dedicated opening authority $\mathcal{O}$ that can reveal the identity of the signer behind a particular signature in a verifiable manner. More precisely, a group signature Π_{GS} consists of the following algorithms:

$\mathsf{GKg}(1^\lambda) \to (gpk, isk, osk)$**:** On input the security parameter 1^λ it outputs a group public key gpk, and the secret keys isk, osk for the issuer and opener.

$\mathsf{UKg}(1^\lambda) \to (upk, usk)$**:** Outputs the private and public key of a user.

$\langle \mathsf{Join}(gpk, upk, usk), \mathsf{Issue}(isk, reg) \rangle \to (gsk, reg')$**:** A user can join the group by running an interactive join protocol with the issuer. The user's output is his signing key gsk, and the issuer outputs an updated registration table reg'.

$\mathsf{Sign}(gpk, gsk, \mu) \to \Sigma$**:** On input a group public key gpk, a user's secret signing key gsk and a message μ outputs a signature Σ.

$\mathsf{Verify}(gpk, \mu, \Sigma) \to 1/0$**:** Verifies a signature Σ against the group public key gpk.

$\mathsf{Open}(gpk, osk, reg, \Sigma, \mu) \to (upk, \tau)/\perp$**:** This algorithms uses the opener's secret key osk to recover the identity of the signer of Σ for message μ. It outputs a claimed signer upk and proof τ, or $\perp$ to indicate failure.

$\mathsf{Judge}(gpk, upk, \Sigma, \mu, \tau) \to 1/0$**:** This deterministic judge algorithm verifies the proof τ, i.e., whether the user with public key upk is the signer of Σ.

The user secret keys will be uniformly random 2λ-bit strings from the set $\mathcal{N}$. We define a one-way function $F : \mathcal{N} \to R_+$ which maps a user's secret key ρ to his public key $upk \in R_+$. We will assume that inverting this function (for random input $\rho \in \mathcal{N}$) is λ-hard. A part of the signature will be an encryption of the user identity (and nonce), and we will use the Naor-Yung approach of encrypting the same message under two different public NTRU keys (or where one of the public keys is indistinguishable from random), and provide a zero-knowledge proof of this fact.

Ring	R_q	$\mathbb{Z}_q[X]/(X^d+1)$
Ring Modulus	q	12289
Ring Dimension	d	1024
Standard Deviation	$\sigma = 1.5\sqrt{q}$	
usk space	$\mathcal{N}$	$\{0,1\}^{2\lambda}$
Encryption randomness	$R_\pm$	$\{-1,0,1\}^d \subset R_q$
upk space	R_+	$\{0,1\}^d \subset R_q$
Credential (*gsk*) space	$\mathcal{S}$	$s \in \mathbb{Z}_q[X]/(X^d+1)$, s.t. $\|s\| \leq 1.5\sigma\sqrt{d}$
Signature size	$\|\Sigma\|$	$19.86KB$

Table 2: Proposed parameters for our group signature

Key Generation: The issuer's key consists of a public $a \in R_q$ together with a secret trapdoor T_a that will allow him to sample $s_1, s_2 \sim D^{\perp}_{a,t,\sigma}$ with $\sigma = 1.5\sqrt{q}$. The reference for this algorithm as well as the construction of the trapdoor T_a is discussed in Section 2. The opener's public key will be $h = f/g$ where $f, g \leftarrow R_\pm$ and his secret opening key will be (f, g).

A user's key is as described above, i.e. it sets $usk = \rho$ chosen uniformly at random from $\mathcal{N}$, and will define $upk = F_{R_+}(\rho)$ as his public key where F is a λ-hard one-way function.

Algorithm 1 $\mathsf{GKg}(1^\lambda)$

Output: $gpk := (a, h, h')$, $isk := T_a$, $osk := (f, g)$.
1: $(a, T_a) \leftarrow$ NTRUTrapdoor.
2: $f, g, f', g' \leftarrow R_\pm$. If g, g' is not invertible mod q or mod 2, re-sample it.
3: $h := f/g$, $h' := f'/g'$.

Join: When a user with keys $usk = \rho$, $upk = F_{R_+}(\rho)$ wants to join the group, it send *upk* to the issuer. This *upk* is the value to which all of his actions can be traced to by the opener. The issuer then samples $s_1, s_2 \leftarrow D_{a,t,\sigma}$, for $t = H_{R_q}(F_{R_+}(\rho))$ and sends s_1, s_2 to the group member. The member will use ρ and the polynomials s_1, s_2 as his signing credentials. Observe that s_1, s_2 is the GPV signature of the message $F_{R_+}(\rho)$ when the GPV signature is instantiated with the concrete hash function H_{R_q}.

Sign: If a member with credentials (ρ, s_1, s_2), as above, wishes to sign μ, he creates two NTRU encryptions of the message $F_{R_+}(\rho)$ with respect to the public keys h and h' and gives a zero-knowledge proof that he knows the randomness and the message underlying the ciphertexts, as well as the knowledge of ρ, s_1, s_2 satisfying $as_1 + s_2 = H_{R_q}(F_{R_+}(\rho))$ and the fact that $F_{R_+}(\rho)$ is the message in the ciphertext. The μ is signed via its insertion in the random oracle during the Fiat-Shamir transform.

The reason that we need two NTRU "encryptions" is to achieve CCA security via the Naor-Yung transform. While the Naor-Yung approach is usually not the most practical way of building CCA-secure schemes, it actually incurs little

Algorithm 2 $\langle \mathsf{Join}(gpk, upk, usk), \mathsf{Issue}(isk, reg)\rangle$

Input: $usk = \rho$, $upk = F_{R_+}(\rho)$, $gpk = (a, h, h')$, $isk = T_a$, reg
Output: User: $gsk = (s_1, s_2, \rho)$, Issuer: updated registr. table reg'.
1: User: Send upk to the Issuer
2: Issuer: Check that $upk \notin reg$. Sample $s_1, s_2 \leftarrow D^{\perp}_{a,t,\sigma}$ for $t := H_{R_q}(upk)$. Send s_1, s_2 to the User, output $reg' = reg \cup \{upk\}$.
3: User: If $as_1 + s_2 = H_{R_q}(F_{R_+}(\rho))$, output $gsk = (s_1, s_2 \in \mathcal{S}^2, \rho)$.

Algorithm 3 $\mathsf{Sign}(gpk, gsk, \mu)$:

Input: $gsk = (s_1, s_2, \rho)$ s.t. $as_1 + s_2 = H_{R_q}(\rho)$, $gpk = (a, h, h')$, μ
Output: Signature $\Sigma := (u, u', \pi)$
1: $e_1, e_2, e_1', e_2' \leftarrow R_{\pm}$
2: $u := 2(he_1 + e_2) + F_{R_+}(\rho)$, $u' := 2(he_1' + e_2') + F_{R_+}(\rho)$
3: $\pi := \mathsf{ZKP}^{\Delta}_{\mathsf{DLR}}\{(s_1, s_2, e_1, e_2, e_1', e_2', \rho) : as_1 + s_2 = H_{R_q}(F_{R_+}(\rho)) \ \wedge \ 2(he_1 + e_2) + F_{R_+}(\rho) = u \ \wedge \ 2(h'e_1' + e_2') + F_{R_+}(\rho) = u' \ \wedge \ s_1, s_2 \in \mathcal{S} \ \wedge \ e_1, e_2, e_1', e_2' \in R_{\pm} \ \wedge \ \rho \in \mathcal{N}\}(\mu)$
4: **return** $\Sigma := (u, u', \pi)$

overhead in our case because providing proofs of ciphertext correctness would be necessary even if we were only aiming for CPA security. For CCA security, we just need to prove two equations instead of one.

Verify: For verification, the Verifier simply checks the validity of the proof.

Open: The opener checks the proof in the signature and performs NTRU decryption of the ciphertext u using his secret key g. If the decrypted public key is contained in the registration table, he gives a zero-knowledge proof that the opening is correct. In particular, he proves that he knows the secret keys g, f that form the public key h (i.e. $f/g = h$) and also that the multiplication $gu = 2v+gm$ where v is a polynomial with coefficients less than $q/4 - d/2$. If this is satisfied, then decryption is indeed valid because $gu \bmod q = 2v + gm$ in $\mathcal{R}$, which follows from the smallness of v and the fact that all the coefficients of gm are at most d. Therefore decryption, which requires reducing the above modulo 2 guarantees that $gu \bmod q \bmod 2 = gm$. Hence the correct decryption of u is m.

Judge: The Judge checks that the opener's proofs are valid. If it is, he concludes that the opener revealed the correct identity.

4.1 Security of the Group Signature Scheme

We now show that our dynamic group signature scheme is secure according to the established notions by Bellare et al. [6], i.e., it satisfies *anonymity*, *traceabilty* and *non-frameability*. The detailed proof of the following theorem is given in the full version.

Algorithm 4 $\mathsf{Verify}(gpk, \Sigma, \mu)$:

Input: $\Sigma = (u, u', \pi)$, $gpk = (a, h, h')$, μ
Output: Output 1 iff the verification passes
1: **return** 1 iff π is valid wrt u, u', gpk and μ.

Algorithm 5 $\mathsf{Open}(gpk, osk, reg, \Sigma, \mu)$:

Input: $\Sigma = (u, u', \pi)$, message μ, $gpk = (a, h, h')$, $osk = (f, g)$, registration table reg.
Output: Identity $upk = m$, and proof of valid decryption τ, or $\perp$.
1: $m := (gu \bmod q)/g \bmod 2$.
2: **return** $\perp$ if $\mathsf{Verify}(gpk, \Sigma, \mu) \neq 1$ or $m \notin reg$
3: $\tau := \mathsf{ZKP}^{\Delta}_{\mathsf{DLR}}\{(f, g, v) : hg - f = 0 \wedge ug = 2v + gm \wedge f, g \in R_{\pm} \wedge v \in \mathcal{R} \text{ s.t. } \|v\|_\infty < q/4 - d/2\}$
4: **return** (m, τ)

Theorem 1. *Our group signature is fully anonymous, traceable and non-frameable when the underlying NTRU encryption scheme is CPA secure, the underlying GPV signature scheme is unforgeable, F is one-way, the proof system $\mathsf{ZKP}^{\Delta}_{\mathsf{DLR}}$ is special sound and zero-knowledge, and* DLR *is Δ-hard.*

Hardness We now briefly analyze the concrete security of the underlying lattice schemes in our group signature scheme for the parameters given in Table 2. This means we assess the complexity of some known lattice attacks on our instantiations of the NTRU encryption scheme and the GPV signature scheme.

For NTRU we focus on the primal key recovery attack, see [9] for more details and an overview of other attacks, in particular meet-in-the-middle and hybrid attacks. Given $h \in R_q$, the problem is to find two short polynomials $f, g \in R_q$ such that $gh = f$ in R_q. By lifting the equation to R, this gives a lattice of dimension $2d$ and volume q^d. Now one can hope that certain coefficients of g are zero, say k many, $0 \leq k < d$, and search for a solution in the corresponding sublattice of dimension $2d-k$. This gives a speed-up despite the reduced success probability. Furthermore, we can restrict the search to the sublatice corresponding to only $m \leq d$ of the equations over $\mathbb{Z}^d$, leaving us with a lattice of dimension $d - k + m$ and volume q^m. The general strategy then is to apply the BKZ basis reduction algorithm to the basis of an optimally chosen sublatice with a large enough block size β so that our target solution will be found. When using John Schanck's estimation scripts [36], we find that for $m = 889$ we would require a block size $\beta = 712$. Costing only one call to an SVP algorithm in dimensions 712 in the so-called Core-SVP methodology gives a time complexity of about 2^{208} when using the best known classical sieving algorithms and a complexity of 2^{188} when also considering quantum speed-ups.

For the GPV signature scheme we focus on the forgery attack. Here the adversary needs to find a short solution $s_1, s_2 \in R_q$ such that $\|s_i\| \leq 1.5\sigma\sqrt{d}$ and $as_1 + s_2 = t$ for a random t. This gives a lattice of dimension $2d+1$ and volume q^d. But unlike in the case of NTRU we do not search for a particular very short solu-

Algorithm 6 $\mathsf{Judge}(gpk, upk, \Sigma, \mu, \tau)$:

Input: $\Sigma = (u, u', \pi), \mu, gpk = (a, h, h'),\ upk = m$, and the opener's proof τ
Output: Output 1 iff the user with upk is the signer of Σ
1: **return** 1 iff $\mathsf{Verify}(gpk, \Sigma, \mu) = 1$ and τ is valid wrt gpk, upk, u.

tion. Any solution fulfilling the bound is fine and it is clearly sufficient to search in a sublattice of dimension $n \leq 2d+1$. The BKZ algorithm with blocksize β finds a solution of length $\delta^n q^{d/n}$ where heuristically $\delta = (\beta(\pi\beta)^{1/\beta}/(2\pi e))^{1/(2(\beta-1))}$. We find that we need $\delta < 1.00226$ and hence a block size of $\beta \geq 875$. Finding a shortest vector in dimension 875 costs 2^{255} classically and 2^{232} quantumly.

4.2 Costs and Sizes

We want to analyze the sizes of the signatures Σ in our group signature scheme and the cost of computing and verifying them in terms of numbers of elliptic curve scalar multiplications. By far the largest element of a signature Σ is the proof π. This proof essentially consists of two parts. In the first part the linear equations for u, u' and $H(upk)$ are proven. The second part is concerned with the nonlinear equations $\|s_i\| \leq 1.5\sigma\sqrt{d}$, $upk = F(\rho)$ and $t = H(upk)$. For the first part we use the proof system from [17] but we further split the proof into two parts involving secret polynomials with coefficients in $\{-1, 0, 1\}$ and $\{-(q-1)/2, \ldots, (q-1)/2\}$, respectively. Note that the l2-norm bound on s_1, s_2 is proven separately and hence it is sufficient for the linear proof of $as_1 + s_2 = H(upk)$ to only include the bound $\|s_i\|_\infty \leq (q-1)/2$. From the formulas in [17] we find that the two linear proofs have combined size 75 group elements plus 6 elements in $\mathbb{Z}_p$. The non-linear proof has size 48 group elements and 5 field elements. Since we use a 521-bit curve, for example NIST P-521, the three proofs have a combined size of about $16.36KB$. The two NTRU encryptions consist of two uniform elements in R_q with size 1.75KB each. So in total a signature Σ has size 19.86 KB. See Section D for more explicit details on how the proofs are conducted.

For the number of exponentiations we find from the formulas in [17] and [13] that the prover has to compute 2.047.271 scalar multiplications for the linear proofs and 11.620.232 scalar multiplications for the non-linear proof. So in total the prover needs to compute 13.7 million scalar multiplications. The verifier has to compute at total number of 4 million scalar multiplications.

Acknowledgements This work was supported by the SNSF ERC starting transfer grant FELICITY and the EU Horizon 2020 project FutureTPM (No.779391).

References

1. STARK-friendly hash challenge, 2019. https://starkware.co/hash-challenge/.

2. M. R. Albrecht, L. Grassi, L. Perrin, S. Ramacher, C. Rechberger, D. Rotaru, A. Roy, and M. Schofnegger. Feistel structures for mpc, and more. Cryptology ePrint Archive, Report 2019/397, 2019. `https://eprint.iacr.org/2019/397`.
3. M. R. Albrecht, L. Grassi, C. Rechberger, A. Roy, and T. Tiessen. Mimc: Efficient encryption and cryptographic hashing with minimal multiplicative complexity. In *Advances in Cryptology - ASIACRYPT*, pages 191–219, 2016.
4. A. Aly, T. Ashur, E. Ben-Sasson, S. Dhooghe, and A. Szepieniec. Design of symmetric-key primitives for advanced cryptographic protocols. Cryptology ePrint Archive, Report 2019/426, 2019. `https://eprint.iacr.org/2019/426`.
5. M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *EUROCRYPT*, pages 614–629, 2003.
6. M. Bellare, H. Shi, and C. Zhang. Foundations of group signatures: The case of dynamic groups. In *Topics in Cryptology - CT-RSA*, pages 136–153, 2005.
7. E. Ben-Sasson. Stark-friendly hash, 2019. https://medium.com/starkware/stark-friendly-hash-tire-kicking-8087e8d9a246.
8. F. Benhamouda, J. Camenisch, S. Krenn, V. Lyubashevsky, and G. Neven. Better zero-knowledge proofs for lattice encryption and their application to group signatures. In *ASIACRYPT*, pages 551–572, 2014.
9. D. J. Bernstein, C. Chuengsatiansup, T. Lange, and C. van Vredendaal. NTRU prime: Reducing attack surface at low cost. In *SAC*, pages 235–260, 2017.
10. G. Bertoni, J. Daemen, M. Peeters, and G. V. Assche. On the indifferentiability of the sponge construction. In *Advances in Cryptology - EUROCRYPT*, pages 181–197, 2008.
11. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Advances in Cryptology - CRYPTO*, pages 41–55, 2004.
12. J. Bootle, A. Cerulli, P. Chaidos, J. Groth, and C. Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *EUROCRYPT*, pages 327–357, 2016.
13. B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *IEEE Symposium on Security and Privacy, SP*, pages 315–334, 2018.
14. J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In *SCN*, pages 268–289, 2002.
15. R. Chaabouni, H. Lipmaa, and A. Shelat. Additive combinatorics and discrete logarithm based range protocols. In *ACISP*, pages 336–351, 2010.
16. R. del Pino, V. Lyubashevsky, and G. Seiler. Lattice-based group signatures and zero-knowledge proofs of automorphism stability. In *CCS*, pages 574–591, 2018.
17. R. del Pino, V. Lyubashevsky, and G. Seiler. Short discrete log proofs for fhe and ring-lwe ciphertexts. In *PKC*, 2019.
18. D. Derler and D. Slamanig. Highly-efficient fully-anonymous dynamic group signatures. In *AsiaCCS*, pages 551–565, 2018.
19. L. Ducas, V. Lyubashevsky, and T. Prest. Efficient identity-based encryption over NTRU lattices. In *ASIACRYPT*, pages 22–41, 2014.
20. L. Ducas and T. Prest. Fast fourier orthogonalization. In *ISSAC*, pages 191–198, 2016.
21. M. F. Esgin, R. K. Zhao, R. Steinfeld, J. K. Liu, and D. Liu. Matrict: Efficient, scalable and post-quantum blockchain confidential transactions protocol. In *CCS*, pages 567–584. ACM, 2019.
22. L. D. Feo, S. Masson, C. Petit, and A. Sanso. Verifiable delay functions from supersingular isogenies and pairings. In *Asiacrpt*, 2019.

23. A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland. Surface codes: Towards practical large-scale quantum computation. *Phys. Rev. A*, 86:032324, Sep 2012.
24. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.
25. C. Gidney. Why will quantum computers be slow? `http://algassert.com/post/1800`. Last accessed Monday 3rd February, 2020., 2018.
26. L. Grassi, D. Kales, D. Khovratovich, A. Roy, C. Rechberger, and M. Schofnegger. Starkad and poseidon: New hash functions for zero knowledge proof systems. Cryptology ePrint Archive, Report 2019/458, 2019. `https://eprint.iacr.org/2019/458`.
27. J. Hoffstein, N. Howgrave-Graham, J. Pipher, J. H. Silverman, and W. Whyte. Ntrusign: Digital signatures using the ntru lattice. In *CT-RSA*, pages 122–140, 2003.
28. J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: A ring-based public key cryptosystem. In *ANTS*, pages 267–288, 1998.
29. B. Lekitsch, S. Weidt, A. G. Fowler, K. Mølmer, S. J. Devitt, C. Wunderlich, and W. K. Hensinger. Blueprint for a microwave trapped ion quantum computer. *Science Advances*, 3(2), 2017.
30. S. Ling, K. Nguyen, D. Stehlé, and H. Wang. Improved zero-knowledge proofs of knowledge for the ISIS problem, and applications. In *PKC*, pages 107–124, 2013.
31. V. Lyubashevsky. Lattice signatures without trapdoors. In *EUROCRYPT*, pages 738–755, 2012.
32. V. Lyubashevsky and D. Micciancio. Generalized compact knapsacks are collision resistant. In *ICALP (2)*, pages 144–155, 2006.
33. C. Peikert and A. Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *TCC*, pages 145–166, 2006.
34. D. Pointcheval and O. Sanders. Short randomizable signatures. In *Topics in Cryptology - CT-RSA*, pages 111–126, 2016.
35. T. Prest, P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Ricosset, G. Seiler, W. Whyte, , and Z. Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2017. https://csrc.nist.gov/projects/post-quantum-cryptography/ round-1-submissions.
36. J. M. Schanck. Security estimator for lattice based cryptosystems, 2019.
37. J. Stern. A new identification scheme based on syndrome decoding. In *CRYPTO*, pages 13–21, 1993.

A Lattice-Based ZKP for Relation 6

Below we provide the prover and verifier algorithms for relation 6 adapted from [8].

If $R_q = \mathbb{Z}_q[X]/(X^d+1)$, then we define the set $\mathcal{M} = \{0, \pm x^i \ 0 \leq i < d\}$. The size of $\mathcal{M}$ is $2d+1$. We also define a parameter λ which controls the soundness error of the proof. The soundness error will be $|\mathcal{M}|^{-\lambda} \approx d^{-\lambda-1}$. For example, if $d = 2048$, then to get the soundness error to be less than 2^{-128}, we need to set $\lambda = 11$.

The proof in Algorithm 7 uses Gaussian-based rejection sampling and can be shown to be zero-knowledge, and requiring 3 iterations on average, using [31,

Algorithm 7 Prover

Input: Secret $\vec{s} = \begin{bmatrix} s_1 \\ \dots \\ s_m \end{bmatrix} \in R_q^m$ s.t. $\|s_i\| \leq \alpha$ and public $\vec{A} \in R_q^{n \times m}, \vec{t} = \vec{A}\vec{s} \in R_q^n$.

Output: $\pi = (\vec{z} \in R_q^\lambda, (c_1, \dots, c_\lambda) \in \mathcal{M}^\lambda)$

1: $\sigma := 11\alpha\sqrt{m\lambda}$; for $i = 1$ to λ, $\vec{y}_i \leftarrow D_\sigma$, $\vec{w}_i := \vec{A}\vec{y}_i$

2: $(c_1, \dots, c_\lambda) := H_{\mathcal{M}^\lambda}(\vec{A}, \vec{t}, \vec{w}_1, \dots, \vec{w}_\lambda)$; $\vec{v} := \begin{bmatrix} c_1\vec{s} \\ \cdots \\ c_\lambda\vec{s} \end{bmatrix} \in R_q^{m\lambda}$

3: $\vec{z} = \begin{bmatrix} \vec{z}_1 \\ \cdots \\ \vec{z}_\lambda \end{bmatrix} := \begin{bmatrix} \vec{y}_1 \\ \cdots \\ \vec{y}_\lambda \end{bmatrix} + \vec{v} \in R_q^{m\lambda}$

4: with probability $1 - \frac{D_\sigma(\vec{z})}{3D_{\vec{v},\sigma}(\vec{z})}$, goto 1

5: **return** $\vec{z}, (c_1, \dots, c_\lambda)$

Algorithm 8 Verifier

Input: $\vec{A} \in R_q^{n \times m}, \vec{t} = \vec{A}\vec{s} \in R_q^n$, $\pi = (\vec{z} \in R_q^\lambda, (c_1, \dots, c_\lambda) \in \mathcal{M}^\lambda)$

Output: Output 1 iff $\pi = \mathsf{ZKP}\{\bar{\vec{s}} : \vec{A}\bar{\vec{s}} = 2\vec{t}, \|\bar{\vec{s}}\| \leq 3\sigma d^{1.5}\sqrt{m} = 33\alpha d^{1.5} m\sqrt{\lambda}\}$

write $\begin{bmatrix} \vec{w}_1 \\ \dots \\ \vec{w}_\lambda \end{bmatrix} := \begin{bmatrix} \vec{A}\vec{z}_1 - c_1\vec{t} \\ \dots \\ \vec{A}\vec{z}_\lambda - c_\lambda\vec{t} \end{bmatrix}$

Accept iff $(c_1, \dots, c_\lambda) = H_{\mathcal{M}^\lambda}(\vec{A}, \vec{t}, \vec{w}_1, \dots, \vec{w}_\lambda)$ and $\|\vec{z}_i\| \leq 1.5\sigma\sqrt{dm}$

Theorem 4.6]. If $|\mathcal{M}|^\lambda > 2^{128}$, then a prover succeeding with probability greater than $\approx 2^{-128}$ can be rewound to produce two solutions $\vec{A}\vec{z}_i = \vec{w}_i + c_i\vec{t}$ and $\vec{A}\vec{z}'_i = \vec{w}_i + c'_i\vec{t}$ for distinct $c_i \in \mathcal{M}$. These can be combined to form the solution

$$\vec{A}(\vec{z}_i - \vec{z}'_i)/(c_i - c'_i) = \vec{t}.$$

By [8, Lemma 3.1], we know that for $c_i \neq c'_i \in \mathcal{M}$, the quotient $2/(c_i - c'_i)$ is a polynomial with coefficients in $\{-1, 0, 1\}$ and therefore has ℓ_2-norm at most $\sqrt{d}$. The parameters for the size of $\bar{\vec{s}}$ in (6) then follow from the parameters in Algorithms 7 and 8.

B Hash Functions with Efficient Proofs

In our group signature and DAA scheme, we need to use a hash function that allows for efficient zero-knowledge proofs that a hash was correctly computed and that the prover knows a pre-image of the hash value. We will use zero-knowledge proofs based on the discrete logarithm assumption, which naturally lend themselves to proving statements over fields of large prime order. Therefore, we would like to use a hash-function built around arithmetic over such fields.

The MiMC Hash Function Family. MiMC [3] is a family of hash functions designed with precisely this in mind. MiMC hash functions are based on the sponge

construction [10]. The construction works by cubing the input over the field, adding randomly chosen constant values, and repeating the process many times.

For fixed input size, output size, and security level, the MiMC family includes a range of hash functions with a trade-off between the size of the prime field used and the number of multiplication gates in a circuit which verifies correct computing of the hash function. Later, in our choices of zero-knowledge proof-system, we will see that for every multiplication in the circuit, the prover must perform some exponentiations over a cryptographic group. Therefore, in the two cases below, we have carefully selected the parameters of the MiMC hash functions in order to minimise the computational burden on the prover. To specify an MiMC hash function, one must give the desired security level and the 'rate' of the round function, which determines the prime field to be used.

As part of our schemes, we will use a pre-image resistant function (later referred to as F_{R_+}) to protect the user's secret key. We instantiate this function with an MiMC hash function with an input length of 256 bits and an output length of 1,024 bits. The circuit used to prove knowledge of a hash pre-image has 60,192 multiplication gates. We will also use a hash-function, modelled as a random oracle, which maps the output of the previous function onto a ring element from $\mathbb{Z}_q[X]/(X^d + 1)$. In this case, we use an MiMC hash function with an input length of 1,024 bits and an output length of 14,336 bits. For the new, larger input and output sizes, the circuit used to prove knowledge of a hash pre-image has 831,577 multiplication gates.

In both cases, we use MiMC hash functions with capacity 512, and a 521-bit prime. This choice of parameters comes from our requirement that the hash function has 256 bits of classical security and therefore 128 bits of quantum security against collision-finding attacks. For 256 bits of classical security, the internal workings of the hash function force us to use a prime of at least 512 bits. Hence, we use a 521-bit prime so that we can use a standardised NIST elliptic curve, for which we expect highly optimised implementations of curve operations compared with unstandardised curves.

C Quantum Annoying and Timed ZKPs

The core observation behind our timed ZKPs is that while certain hard problems, such as the discrete logarithm problem, can be solved in polynomial-time by (sufficiently sized) quantum computers, it is likely that solving them won't be instantaneous or at least prohibitively expensive. Thus, forcing the adversary to solve a *fresh* DLP instance for each proof might render the attack infeasible.

This property has recently been described as *quantum annoyance* [22] and formalized through a two stage adversary. Roughly, in an offline pre-computation phase the adversary is granted full quantum power, but gets restricted to be classical when turning to an online phase.

We now apply this concept to zero-knowledge proofs, more precisely, we consider ZKPs for generalized statements following the form of equation (7) of the proof system recently introduced in [17]. The proof system uses a CRS made

up of random group elements $g_1, \ldots, g_n$, and assuming the DL problem is hard, it allows to prove knowledge of a witness for various NP statements. For example, the protocol of [17] actually proves is that the prover knows a SIS solution $\mathbf{s}$ *or* a non-trivial discrete logarithm relation between $g_1, \ldots, g_n$. Generalizing this idea we consider proofs of the form: $\mathsf{ZKP}\{(w) : (x_q, w) \in \mathcal{R}_q \ \vee \ (x_c, w) \in \mathcal{R}_c\}$, where $\mathcal{R}$ denotes a NP relation and w is a witness for a statement x if $(x, w) \in \mathcal{R}$.

In this plain form, the soundness of the proof relies on the weaker of both relations, i.e., the DL assumption in the case of [17] even though it also proves a lattice relation. We can transform the proof into a quantum annoying (and later timed) version by simply letting the verifier freshly choose x_c (i.e., g_i in our concrete scheme) when the proof starts.

Let $x \xleftarrow{\$} \mathsf{Gen}(1^\lambda, \mathcal{L})$ be a generator that produces a random instance $x \in \mathcal{L}$ for security parameter 1^λ and language $\mathcal{L} = \{x \mid \exists w : (x, w) \in \mathcal{R}\}$. We can then formulate quantum-annoying soundness for an interactive proof protocol $(\mathcal{P}, \mathcal{V})$ for statements $(x_q, w) \in \mathcal{R}_q \ \vee \ (x_c, w) \in \mathcal{R}_c$ as follows: For any efficient adversary $(\mathcal{A}_1, \mathcal{A}_2)$ — where $\mathcal{A}_1$ is quantum, and $\mathcal{A}_2$ is classical — running the following game

1. sample random $x_q \xleftarrow{\$} \mathsf{Gen}(1^\lambda, \mathcal{L}_q)$
2. $\mathsf{st} \xleftarrow{\$} \mathcal{A}_1(x_q)$
3. sample random $x_c \xleftarrow{\$} \mathsf{Gen}(1^\lambda, \mathcal{L}_c)$
4. where $\mathsf{Pr}\left[\langle \mathcal{A}_2(\mathsf{st}, x_q, x_c), \mathcal{V}(x_q, x_c)\rangle = 1\right] > \epsilon$

there exist an efficient extractor $\mathcal{E}$ with rewindable black-box access to $\mathcal{A}_2$ that outputs w s.t. $(x_q, w) \in \mathcal{R}_q \vee (x_c, w) \in \mathcal{R}_c$ with probability $\geq \epsilon / \mathsf{poly}(1^\lambda)$.

Generally, the online adversary $\mathcal{A}_2$ can be seen as a resource-restricted adversary that cannot break the classical problem. While quantum-annoyance models the resource restriction by simply limiting $\mathcal{A}_2$ to be classical, we can also be more generous and give $\mathcal{A}_2$ quantum power, yet restrict its running time.

That is, the verifier only accepts a proof when the prover correctly responds within some fixed short time Δ. The soundness of our ZKP then even holds against a full quantum adversary under the additional assumption that the problem $\mathcal{R}_c$ is hard to solve within a short amount of time. We will refer to such an assumption as Δ-hardness.

Note that there are subtle constraints on how to choose the time Δ for a concrete ZKP instantiation based on a Δ'-hard problem. For satisfying completeness, Δ must be chosen large enough, such that honest provers can still complete the proof (for $\mathcal{L}_q$) in time. For soundness, Δ depends on the loss in the reduction, i.e., the running time of the extractor that will be used to break the Δ'-hard problem needs to be taken into account. We leave a more formal treatment of these relations as interesting future work.

D Zero-Knowledge Proofs for Group Signature Algorithms

In this section, we explain how to give the zero-knowledge proofs for the group signature algorithms of Section 4 in terms of the proof systems of [17] for SIS

relations and [13] for more complicated relations with less special structure available.

Both proof systems rely on the discrete logarithm assumption.

Definition 1 (Discrete Log Relation). *For all PPT adversaries $\mathcal{A}$ and for all $n \geq 2$ there exists a negligible function $\mu(\lambda)$ such that*

$$P\left[\begin{matrix}\mathcal{C} = \mathcal{G}(1^\lambda),\ g_1, \ldots, g_n \leftarrow \mathcal{C}; \\ a_1, \ldots, a_n \in \mathbb{Z} \leftarrow \mathcal{A}(G, g_1, \ldots, g_n)\end{matrix} : \exists a_i \neq 0 \wedge \prod_{i=1}^{n} g_i^{a_i} = 1\right] \leq \mu(\lambda)$$

For $n \geq 2$, this is equivalent to the discrete logarithm assumption.

Sign: A zero-knowledge proof of the following statement is computed:

$$\mathsf{ZKP}^{\Delta}_{\mathsf{DLR}}\left\{\begin{matrix} & as_1 + s_2 = H_{R_q}(F_{R_+}(\rho)) \\ & \wedge\ 2(he_1 + e_2) + F_{R_+}(\rho) = u \\ s_1, s_2, & \wedge\ 2(h'e'_1 + e'_2) + F_{R_+}(\rho) = u' \\ e_1, e_2, e'_1, e'_2, \rho\ : & \wedge\ s_1, s_2 \in \mathcal{S}\ \wedge\ \rho \in \mathcal{N} \\ & \wedge\ e_1, e_2, e'_1, e'_2 \in R_\pm \end{matrix}\right\}(\mu)$$

The conditions in this relation can be rewritten as follows, with appropriate size bounds on different elements. Set $k = F_{R_\pm}(\rho)$ and $l = H_{R_q}(k)$.

$$\begin{bmatrix} 2h & 2 & 0 & 0 & 1 \\ 0 & 0 & 2h' & 2 & 1 \end{bmatrix} \cdot \begin{bmatrix} e_1 \\ e_2 \\ e'_1 \\ e'_2 \\ k \end{bmatrix} = \begin{bmatrix} u \\ u' \end{bmatrix} \wedge \begin{bmatrix} a & 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} s_1 \\ s_2 \\ l \end{bmatrix} = 0$$

$$\wedge\ k = F_{R_\pm}(\rho)\ \wedge\ l = H_{R_q}(k)$$

We prove the necessary conditions as follows. We use the proof system of [17] to give a zero knowledge proof for the first linear equation, which has an infinity norm bound of 1 on e_1, e_2, e'_1, e'_2 and k. The size of this proof is roughly 76 group elements and 6 field elements for the parameters that we have chosen. We also use the same proof system from [17] to give a zero-knowledge proof for the second linear equation, with an infinity norm bound of q on s_1, s_2 and l.

The remaining conditions that we have to check are the conditions $k = F_{R_\pm}(\rho)$, $l = H_{R_q}(k)$, and the fact that the ℓ_2-norms of s_1 and s_2 are bounded by $1.5\sigma\sqrt{d}$. We use the proof system of [13] to achieve this. This proof system works with general arithmetic circuits. The number of multiplication gates in the circuit required to prove these conditions is the sum of the sizes of the circuits for $F_{R_\pm}$ and H_{R_q}, plus roughly 2096 extra multiplications which are used for checking that the norms of s_1 and s_2 are bounded correctly. The extra multiplication gates compute the squares of the ℓ_2 norms of each of s_1 and s_2, using 2048 multiplications, check that roughly 48 values are bits by checking that when multiplying them with their complements, the result is zero, and then show that the squares of the ℓ_2 norms are represented by the binary values, so that

the norms must be in the correct range. Since we have already used the proof system [17] to check that the infinity norms of s_1 and s_2 are bounded, and we work over a prime field with a much larger modulus than the base ring of s_1 and s_2, we need not worry about overflow when computing the squares of the ℓ_2 norms. We give zero-knowledge proofs of arithmetic circuit satisfiability and prove all of these things using one single proof from [13]. This proof contributes 48 group elements and 5 finite field elements.

In order to use these proof systems, and be sure that certain secret values are consistent across the different proofs, we need to make some adjustments. The first tweak is to split some of the long commitments made in the protocols into several parts, to allow values to be shared between the two proof systems. This is described in the full version. Separate commitments to k and s_1, s_2, l allow these values to be shared between the first two proofs for linear relations and the third proof for non-linear relations.

The second tweak is to modify the protocol of [17] so that it works even if we are proving that the entries of the secret vector lie in an interval whose width is not a power of 2. This is easily achieved using techniques from [15]. The idea is that a binary expansion of the form $\sum_i x_i 2^i$ uniquely expresses every integer in a given interval whose width is a power of 2, but if we change the powers of two in the expression to other values, we can obtain (possibly non-unique) binary expansions for other intervals which suffice for the purpose of giving range proofs. This change has no impact on proof size.

Open: The following zero-knowledge proof is needed:

$$\mathsf{ZKP}^{\Delta}_{\mathsf{DLR}} \left\{ (f, g, v) \ : \ \begin{matrix} hg - f = 0 \ \wedge \ ug = 2v + gm \\ \wedge \ f, g \in R_{\pm} \ \wedge \ v \in \mathcal{R} \text{ s.t. } \|v\|_\infty < q/4 - d/2 \end{matrix} \right\}$$

The conditions in this relation can be rewritten as follows, with appropriate size bounds on different elements.

$$\begin{bmatrix} h & 1 & 0 \\ u & 0 & 2 \end{bmatrix} \cdot \begin{bmatrix} g \\ -f \\ -v \end{bmatrix} = \begin{bmatrix} 0 \\ m \end{bmatrix}$$

This relation is proved by using the proof system from [17] twice. The first proof proves the linear relation from the first row of the matrix, which does not include v. Therefore, the proof system can be used with norm bound 1. The second proof proves the linear relation from the second row of the matrix, which does include v, and therefore works with norm bound $q/4 - d_2$. As with the signing algorithm, we use the adjustments described to make sure that the preimage values are consistent across the two proofs.