Constant-size Group Signatures from Lattices

San Ling, Khoa Nguyen, Huaxiong Wang, Yanhong Xu

Division of Mathematical Sciences,
School of Physical and Mathematical Sciences,
Nanyang Technological University, Singapore.
{lingsan,khoantt,hxwang,xu0014ng}@ntu.edu.sg

Abstract. Lattice-based group signature is an active research topic in recent years. Since the pioneering work by Gordon, Katz and Vaikuntanathan (Asiacrypt 2010), ten other schemes have been proposed, providing various improvements in terms of security, efficiency and functionality. However, in all known constructions, one has to fix the number N of group users in the setup stage, and as a consequence, the signature sizes are dependent on N.

In this work, we introduce the first constant-size group signature from lattices, which means that the size of signatures produced by the scheme is independent of N and only depends on the security parameter λ. More precisely, in our scheme, the sizes of signatures, public key and users' secret keys are all of order $\widetilde{\mathcal{O}}(\lambda)$. The scheme supports dynamic enrollment of users and is proven secure in the random oracle model under the Ring Short Integer Solution (RSIS) and Ring Learning With Errors (RLWE) assumptions. At the heart of our design is a zero-knowledge argument of knowledge of a valid message-signature pair for the Ducas-Micciancio signature scheme (Crypto 2014), that may be of independent interest.

Keywords. lattice-based cryptography, constant-size group signatures, zero-knowledge proofs, Ducas-Micciancio signature

1 Introduction

Group signature, introduced by Chaum and van Heyst [18], is a fundamental anonymity primitive which allows members of a group to sign messages on behalf of the whole group. Yet, users are kept accountable for the signatures they issue since a tracing authority can identify them should the need arise. These two appealing features allow group signatures to find applications in various real-life scenarios, such as digital right management, anonymous online communications, e-commerce systems, and much more. On the theoretical front, designing secure and efficient group signature schemes is interesting and challenging, since those advanced constructions usually require a sophisticated combination of carefully chosen cryptographic ingredients: digital signatures, encryption schemes, and zero-knowledge protocols. Numerous group signature schemes have been proposed in the last quarter-century, some of which produce very short signatures [2,8]. In the setting of bilinear groups, many schemes [1,28,12,40] achieved

constant-size signatures, which means that the group signatures only contain $\mathcal{O}(1)$ number of group elements. In other words, the signature sizes in those schemes only depend on the security parameter and are independent of the number N of group users. In the lattice setting, however, none of the existing constructions achieved this feature.

LATTICE-BASED GROUP SIGNATURES. Lattice-based cryptography has been an exciting research area since the seminal works of Regev [55] and Gentry et al. [24]. Lattices not only allow to build powerful primitives (e.g., [23,25]) that have no feasible instantiations in conventional number-theoretic cryptography, but they also provide several advantages over the latter, such as conjectured resistance against quantum adversaries and faster arithmetic operations. Along with other primitives, lattice-based group signature has received noticeable attention in recent years. The first scheme was introduced by Gordon, Katz and Vaikuntanathan [26] whose solution produced signature size linear in the number of group users N. Camenisch et al. [16] then extended [26] to achieve anonymity in the strongest sense. Later, Laguillaumie et al. [32] put forward the first scheme with the signature size logarithmic in N, at the cost of relatively large parameters. Simpler and more efficient solutions with $\mathcal{O}(\log N)$ signature size were subsequently given by Nguyen et al. [52] and Ling et al. [42]. Libert et al. [37] obtained substantial efficiency improvements via a construction based on Merkle trees which eliminates the need for GPV trapdoors [24]. More recently, a scheme supporting message-dependent opening (MDO) feature [56] was proposed in [39]. All the schemes mentioned above are designed for static groups, and all have signature sizes dependent on N.

Three lattice-based group signatures that have certain dynamic features were proposed by Langlois et al. [33], Libert et al. [35], and Ling et al. [43]. The first one is a scheme with verifier-local revocation (VLR) [9], which means that only the verifiers need to download the up-to-date group information. The second one addresses the orthogonal problem of dynamic user enrollments, which was formalized by Kiayias and Yung [31] and by Bellare et al. [5]. The third one is a fully dynamic scheme that supports both features, following Bootle et al.'s model [10]. Again, all these three schemes have signature size $\mathcal{O}(\log N)$.

In all existing works on lattice-based group signatures, for various reasons, one has to fix the number $N = \mathsf{poly}(\lambda)$, where λ is the security parameter, in the setup stage. For the schemes from [26,16,32,33,42,52,39,35] - which are based on full-fledged lattice-based ordinary signatures [24,11,17], this is due to the fact that their security reductions have to guess a target user with probability $1/N$, and cannot go through unless N is known in advance. For the schemes from [37,43] - which associate group users with leaves in lattice-based Merkle hash trees - this is because the size N of the trees has to be determined so that the setup algorithm succeeds. As a consequence, the parameters of those schemes, including the signature sizes, are unavoidably dependent on N. This state-of-affairs is somewhat unsatisfactory, considering that the constant-size feature has been achieved in the pairing setting. This inspires us to investigate the problem of designing constant-size lattice-based group signatures.

Our Results and Techniques. We introduce the first constant-size group signature scheme from lattices. Here, by "constant-size", we mean that the signature size is independent of the number of group users N, as in the context of pairing-based group signatures [28,12]. The crucial difference between our scheme and previous works on lattice-based group signatures is that we do not have to fix N in the setup phase. As a result, the execution of the scheme is totally independent of N. The sizes of the public key, users' signing keys and signatures are of order $\widetilde{\mathcal{O}}(\lambda)$. A comparison between our schemes and previous works, in terms of asymptotic efficiency and functionality, is given in Table 1.

The scheme operates in Bellare et al.'s model for partially dynamic groups [5], and is proven secure under the hardness of the Ring Short Integer Solution (RSIS) and the Ring Learning With Errors (RLWE) problems. As for all known lattice-based group signatures, our security analysis is in the random oracle model.

Scheme	Sig. Size	Group PK size	Signer's SK size	Functionality
GKV [26]	$\widetilde{\mathcal{O}}(\lambda^2 \cdot N)$	$\widetilde{\mathcal{O}}(\lambda^2 \cdot N)$	$\widetilde{\mathcal{O}}(\lambda^2)$	static
CNR [16]	$\widetilde{\mathcal{O}}(\lambda^2 \cdot N)$	$\widetilde{\mathcal{O}}(\lambda^2)$	$\widetilde{\mathcal{O}}(\lambda^2)$	static
LLLS [32]	$\widetilde{\mathcal{O}}(\lambda \cdot \ell)$	$\mathcal{O}(\lambda^2 \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda^2)$	static
LLNW [33]	$\widetilde{\mathcal{O}}(\lambda \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda^2 \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda \cdot \ell)$	VLR
NZZ [52]	$\widetilde{\mathcal{O}}(\lambda + \ell^2)$	$\widetilde{\mathcal{O}}(\lambda^2 \cdot \ell^2)$	$\widetilde{\mathcal{O}}(\lambda^2)$	static
LNW-I [42]	$\widetilde{\mathcal{O}}(\lambda \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda^2 \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda)$	static
LNW-II [42]	$\widetilde{\mathcal{O}}(\lambda \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda) + \ell$	static
LLNW [37]	$\widetilde{\mathcal{O}}(\lambda \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda^2 + \lambda \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda \cdot \ell)$	static
LLM+ [35]	$\widetilde{\mathcal{O}}(\lambda \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda^2 \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda)$	partially dynamic
LMN [39]	$\widetilde{\mathcal{O}}(\lambda \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda^2 \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda)$	MDO
LNWX [43]	$\widetilde{\mathcal{O}}(\lambda \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda^2 + \lambda \cdot \ell)$	$\widetilde{\mathcal{O}}(\lambda) + \ell$	fully dynamic
Ours	$\widetilde{\mathcal{O}}(\lambda)$	$\widetilde{\mathcal{O}}(\lambda)$	$\widetilde{\mathcal{O}}(\lambda)$	partially dynamic

Table 1. Comparison of known lattice-based group signatures, in terms of asymptotic efficiency and functionality. The comparison is done based on two governing parameters: security parameter λ and the maximum expected number of group users $N = 2^{\ell}$. Among the listed schemes, the LNW-II [42] scheme and ours are the only ideal-lattice-based constructions, while other schemes rely on various SIS and LWE assumptions in the general-lattice setting.

Our scheme relies on the RSIS-based signature scheme by Ducas and Micciancio [20], which exploits the "confined guessing" technique [7] in the ring setting to achieve short public key. We employ the stateful and adaptively secure version

of the scheme, described in [21], which suffices for building group signatures and which allows to work with even shorter key.

The scheme follows the usual sign-then-encrypt-then-prove approach for constructing group signatures. Each user generates a secret-public key pair $(\mathbf{x}, p)$ and becomes a certified group member once receiving a Ducas-Micciancio signature on his public key p. When generating group signatures, the user first encrypts his public key p to ciphertext $\mathbf{c}$ via a CCA-secure encryption scheme obtained by applying the Naor-Yung transformation [51] to a variant of the RL-WE-based scheme by Lyubashevsky, Peikert and Regev [47]. Then he proves in zero-knowledge that: (i) he has a valid secret key $\mathbf{x}$ corresponding to p; (ii) he possesses a Ducas-Micciancio signature on p; and (iii) $\mathbf{c}$ is a correct ciphertext of p. The protocol is then transformed into a signature via the Fiat-Shamir heuristic [22].

To instantiate the above approach, we design a zero-knowledge argument of knowledge of a valid message-signature pair for the Ducas-Micciancio signature, which is based on Stern's framework [57]. We observe that a similar protocol for the Boyen signature [11] was proposed by Ling et al. [42], but their method is sub-optimal in terms of efficiency. We thus propose a refined technique that allows to achieve better communication cost, and hence, shorter signature size. We believe that our protocol is of independent interest. Indeed, apart from group signatures, zero-knowledge protocols for valid message-signature pairs are essential ingredients for designing various privacy-enhancing constructions, such as anonymous credentials [15], compact e-cash [14,38], policy-based signatures [3,19], and much more.

On the practical front, as all known lattice-based group signatures, our scheme is not truly practical. Even though the scheme produces signatures of constant size $\widetilde{\mathcal{O}}(\lambda)$, due to a large poly-logarithmic factor contained in the $\widetilde{\mathcal{O}}$ notation, the signature size is too big to be really useful in practice. We, however, hope that our result will inspire more efficient constructions in the near future.

2 Background

NOTATIONS. The set $\{1, \ldots, n\}$ is denoted by $[n]$. If S is a finite set, then $x \xleftarrow{\$} S$ means that x is chosen uniformly at random from S. When concatenating column vectors $\mathbf{x} \in \mathbb{R}^m$ and $\mathbf{y} \in \mathbb{R}^k$, for simplicity, we use the notation $(\mathbf{x} \| \mathbf{y}) \in \mathbb{R}^{m+k}$ instead of $(\mathbf{x}^\top \| \mathbf{y}^\top)^\top$.

2.1 Rings, RSIS and RLWE

Let $q \geq 3$ be a positive integer and let $\mathbb{Z}_q = [-\frac{q-1}{2}, \frac{q-1}{2}]$. Consider rings of the form $R = \mathbb{Z}[X]/(\Phi_{2n}(X))$ and $R_q = (R/qR)$, where n is a power of 2 and $\Phi_{2n}(X) = X^n + 1$ is the cyclotomic polynomial of degree n.

We will use the coefficient embedding $\tau : R_q \to \mathbb{Z}_q^n$ that maps ring element $v = v_0 + v_1 \cdot X + \ldots + v_{n-1} \cdot X^{N-1} \in R_q$ to vector $\tau(v) = (v_0, v_1, \ldots, v_{n-1})^\top \in \mathbb{Z}_q^n$. We will also use the ring homomorphism $\mathsf{rot} : R_q \to \mathbb{Z}_q^{n \times n}$ that maps $a \in R_q$ to

matrix $\mathsf{rot}(a) = \big[\tau(a) \mid \tau(a \cdot X) \mid \ldots \mid \tau(a \cdot X^{n-1})\big] \in \mathbb{Z}_q^{n \times n}$ (see, e.g., [49,58]). These functions allow us to interpret the product $y = a \cdot v$ over R_q as the matrix-vector product $\tau(y) = \mathsf{rot}(a) \cdot \tau(v) \bmod q$.

When working with vectors over R_q, we often abuse the notations rot and τ. If $\mathbf{A} = [a_1 \mid \ldots \mid a_m] \in R_q^{1 \times m}$, then we denote by $\mathsf{rot}(A)$ the matrix

$$\mathsf{rot}(\mathbf{A}) = \big[\mathsf{rot}(a_1) \mid \ldots \mid \mathsf{rot}(a_m)\big] \in \mathbb{Z}_q^{n \times mn}.$$

If $\mathbf{v} = (v_1, \ldots, v_m)^\top \in R_q^m$, then we let $\tau(\mathbf{v}) = (\tau(v_1) \| \ldots \| \tau(v_m)) \in \mathbb{Z}_q^{mn}$. Note that, if $y = \mathbf{A} \cdot \mathbf{v}$ over R_q, then we have $\tau(y) = \mathsf{rot}(\mathbf{A}) \cdot \tau(\mathbf{v}) \bmod q$.

For $a = a_0 + a_1 \cdot X + \ldots + a_{n-1} \cdot X^{N-1} \in R$, we define $\|a\|_\infty = \max_i(|a_i|)$. Similarly, for vector $\mathbf{b} = (b_1, \ldots, b_{\mathsf{m}})^\top \in R^{\mathsf{m}}$, we define $\|\mathbf{b}\|_\infty = \max_j(\|b_j\|_\infty)$.

We now recall the average-case problems RSIS and RLWE associated with the rings R, R_q, as well as their hardness results.

Definition 1 ([44,54,45]). The $\mathsf{RSIS}_{n,m,q,\beta}$ problem is as follows. Given a uniformly random $\mathbf{A} = [a_1 \mid \ldots \mid a_m] \in R_q^{1 \times m}$, find a non-zero vector $\mathbf{x} = (x_1, \ldots, x_m)^\top \in R^m$ such that $\|\mathbf{x}\|_\infty \leq \beta$ and $\mathbf{A} \cdot \mathbf{x} = a_1 \cdot x_1 + \ldots + a_m \cdot x_m = 0$.

For $m > \frac{\log q}{\log(2\beta)}$, $\gamma = 16\beta mn \log^2 n$, and $q \geq \frac{\gamma\sqrt{n}}{4 \log n}$, the $\mathsf{RSIS}_{n,m,q,\beta}$ problem is at least as hard as $\mathsf{SVP}_\gamma^\infty$ in any ideal in the ring R (see, e.g., [44]).

Definition 2 ([46]). Let $n, m \geq 1$, $q \geq 2$, and let χ be a probability distribution on R. For $s \in R_q$, let $A_{s,\chi}$ be the distribution obtained by sampling $a \xleftarrow{\$} R_q$ and $e \hookleftarrow \chi$, and outputting the pair $(a, a \cdot s + e) \in R_q \times R_q$. The $\mathsf{RLWE}_{n,m,q,\chi}$ problem (the Hermite-Normal-Form version) asks to distinguish m samples chosen according to $\mathcal{A}_{s,\chi}$ (for $s \hookleftarrow \chi$) and m samples chosen according to the uniform distribution over $R_q \times R_q$.

Let $q = \mathsf{poly}(n)$ be a prime power. Let $B = \widetilde{\mathcal{O}}(n^{5/4})$ be an integer and χ be a B-bounded distribution on R, i.e., it outputs samples $e \in R$ such that $\|e\|_\infty \leq B$ with overwhelming probability in n. Then, for $\gamma = n^2(q/B)(nm/\log(nm))^{1/4}$, the $\mathsf{RLWE}_{n,m,q,\chi}$ problem is at least as hard as $\mathsf{SVP}_\gamma^\infty$ in any ideal in the ring R, via a polynomial-time quantum reduction (see, e.g., [46,48,34,53]).

2.2 Decompositions

We next recall the integer decomposition technique from [41]. For any $B \in \mathbb{Z}_+$, define $\delta_B := \lfloor \log_2 B \rfloor + 1 = \lceil \log_2(B+1) \rceil$ and the sequence $B_1, \ldots, B_{\delta_B}$, where $B_j = \lfloor \frac{B + 2^{j-1}}{2^j} \rfloor$, for each $j \in [1, \delta_B]$. As observed in [41], it satisfies $\sum_{j=1}^{\delta_B} B_j = B$ and any integer $v \in [0, B]$ can be decomposed to $\mathsf{idec}_B(v) = (v^{(1)}, \ldots, v^{(\delta_B)})^\top \in \{0,1\}^{\delta_B}$ such that $\sum_{j=1}^{\delta_B} B_j \cdot v^{(j)} = v$. This decomposition procedure is described in a deterministic manner as follows:

1. $v' := v$
2. For $j = 1$ to δ_B do:

(i) If $v' \geq B_j$ then $v^{(j)} := 1$, else $v^{(j)} := 0$;
(ii) $v' := v' - B_j \cdot v^{(j)}$.

3. Output $\mathsf{idec}_B(v) = (v^{(1)}, \ldots, v^{(\delta_B)})^\top$.

In this work, we will employ the above decomposition procedure when working with polynomials in the ring R_q. Specifically, for $B \in [1, \frac{q-1}{2}]$, we define the injective function rdec_B that maps $a \in R_q$ such that $\|a\|_\infty \leq B$ to $\mathbf{a} \in R^{\delta_B}$ such that $\|\mathbf{a}\|_\infty \leq 1$, which works as follows.

1. Let $\tau(a) = (a_0, \ldots, a_{n-1})^\top$. For each i, let $\sigma(a_i) = 0$ if $a_i = 0$; $\sigma(a_i) = -1$ if $a_i < 0$; and $\sigma(a_i) = 1$ if $a_i > 0$.
2. $\forall i$, compute $\mathbf{w}_i = \sigma(a_i) \cdot \mathsf{idec}_B(|a_i|) = (w_{i,1}, \ldots, w_{i,\delta_B})^\top \in \{-1, 0, 1\}^{\delta_B}$.
3. Form the vector $\mathbf{w} = (\mathbf{w}_0 \| \ldots \| \mathbf{w}_{n-1}) \in \{-1, 0, 1\}^{n\delta_B}$, and let $\mathbf{a} \in R^{\delta_B}$ be such that $\tau(\mathbf{a}) = \mathbf{w}$.
4. Output $\mathsf{rdec}_B(a) = \mathbf{a}$.

When working with vectors of ring elements, e.g., $\mathbf{v} = (v_1, \ldots, v_m)^\top$ such that $\|\mathbf{v}\|_\infty \leq B$, then we let $\mathsf{rdec}_B(\mathbf{v}) = \big(\mathsf{rdec}_B(v_1) \| \ldots \| \mathsf{rdec}_B(v_m)\big) \in R^{m\delta_B}$.
Now, $\forall m, B \in \mathbb{Z}_+$, we define matrices $\mathbf{H}_B \in \mathbb{Z}^{n \times n\delta_B}$ and $\mathbf{H}_{m,B} \in \mathbb{Z}^{nm \times nm\delta_B}$ as

$$\mathbf{H}_B = \begin{bmatrix} B_1 \ldots B_{\delta_B} & & \\ & \ddots & \\ & & B_1 \ldots B_{\delta_B} \end{bmatrix}, \quad \text{and} \quad \mathbf{H}_{m,B} = \begin{bmatrix} \mathbf{H}_B & & \\ & \ddots & \\ & & \mathbf{H}_B \end{bmatrix}.$$

Then we have

$$\tau(a) = \mathbf{H}_B \cdot \tau(\mathsf{rdec}_B(a)) \bmod q \quad \text{and} \quad \tau(\mathbf{v}) = \mathbf{H}_{m,B} \cdot \tau(\mathsf{rdec}_B(\mathbf{v})).$$

For simplicity of presentation, when $B = \frac{q-1}{2}$, we will use the notation rdec instead of $\mathsf{rdec}_{\frac{q-1}{2}}$, and $\mathbf{H}$ instead of $\mathbf{H}_{\frac{q-1}{2}}$.

2.3 A Variant of the Ducas-Micciancio Signature scheme

We recall a variant of the Ducas-Micciancio signature scheme [20,21], which is to used to design a (partially) dynamic group signature scheme as in the model of Bellare, Shi and Zhang [5]. Specifically, we use it to enroll new users.

In their papers, Ducas and Micciancio proposed two versions of signature schemes from ideal lattices: non-stateful and stateful. Note that in a group signature scheme, there are at most polynomial number of users. Therefore, it is reasonable to assume there are at most polynomial number of signature queries to the Ducas-Micciancio signature scheme. Under this assumption, the stateful version not only reduces the security loss of the proof, but also allows better parameters ([21, Section 4.1]), compared with the non-stateful version. We also note that in a group signature scheme, the signature scheme used to enroll users should be adaptively secure. To achieve adaptive security, we thus embed the chameleon hash function [21, Appendix B.3] into the above non-adaptively secure version.

Now we summarize the stateful and adaptively secure version of Ducas-Micciancio signature scheme below. Following [20,21], throughout this work, let $c > 1$ be some real constant and $\alpha_0 \geq 1/(c-1)$. Let $d \geq \log_c(\omega(\log n))$ be an integer and $\{c_0, c_1 \cdots, c_d\}$ be a strictly increasing integer sequence with $c_0 = 0$ and $c_i = \lfloor \alpha_0 c^i \rfloor$ for $i \in [d]$. Define $\mathcal{T}_i = \{0,1\}^{c_i}$ for $i \in [d]$. For a tag $t = (t_0, t_1 \ldots, t_{c_d-1})^\top \in \mathcal{T}_d$, let $t_{[i]} = (t_{c_{i-1}}, \ldots, t_{c_i-1})^\top$. Then we can check that $t = (t_{[1]} \| t_{[2]} \| \ldots \| t_{[d]})$. Identify each tag $t \in \mathcal{T}_d$ as $t(X) = \sum_{j=0}^{c_d-1} t_j X^j \in R$ and $t_{[i]}$ as $t_{[i]}(X) = \sum_{j=c_{i-1}}^{c_i-1} t_j X^j \in R$.

This variant works with the following parameters. Given the security parameter λ, the key generation algorithm works as follows.

- Choose parameter $n = \mathcal{O}(\lambda)$ being a power of 2, and modulus $q = 3^k$ for some positive integer k. Let $R = \mathbb{Z}[X]/(X^n + 1)$ and $R_q = R/qR$.
- Also, let $\ell = \lfloor \log \frac{q-1}{2} \rfloor + 1$, $m \geq 2\lceil \log q \rceil + 2$, and $\overline{m} = m + k$.
- Let integer d and sequence $c_0, \ldots, c_d$ as described above. Let $\beta = \widetilde{\mathcal{O}}(n)$ be a integer.
- Let $S \in \mathbb{Z}$ be a state initialized to 0.

The verification key consists of the following:

$$\mathbf{A}, \mathbf{F}_0 \in R_q^{1 \times \overline{m}}; \quad \mathbf{A}_{[0]}, \ldots, \mathbf{A}_{[d]} \in R_q^{1 \times k}; \quad \mathbf{F}, \mathbf{F}_1 \in R_q^{1 \times \ell}; \quad u \in R_q$$

while the signing key is a Micciancio-Peikert [50] trapdoor matrix $\mathbf{R} \in R_q^{m \times k}$.

To sign a message $p \in R_q$, let $\mathbf{p} = \mathsf{rdec}(p) \in R^\ell$ whose coefficients are in the set $\{-1, 0, 1\}$. The signer then proceeds as follows.

- Set the tag $t = (t_0, t_1 \ldots, t_{c_d-1})^\top \in \mathcal{T}_d$, where $S = \sum_{j=0}^{c_d-1} 2^j \cdot t_j$, and compute $\mathbf{A}_t = [\mathbf{A} | \mathbf{A}_{[0]} + \sum_{i=1}^{d} t_{[i]} \mathbf{A}_{[i]}] \in R_q^{1 \times (\overline{m}+k)}$. Update S to $S + 1$.
- Sample $\mathbf{r} \in R^{\overline{m}}$ such that $\|\mathbf{r}\|_\infty \leq \beta$.
- Let $y = \mathbf{F}_0 \cdot \mathbf{r} + \mathbf{F}_1 \cdot \mathbf{p} \in R_q$ and $u_p = \mathbf{F} \cdot \mathsf{rdec}(y) + u \in R_q$.
- Using the trapdoor matrix $\mathbf{R}$, generate a ring vector $\mathbf{v} \in R^{\overline{m}+k}$ such that $\mathbf{A}_t \cdot \mathbf{v} = u_p$ and $\|\mathbf{v}\|_\infty \leq \beta$.
- Output the tuple $(t, \mathbf{r}, \mathbf{v})$ as a signature for $p \in R_q$.

To verify a signature tuple $(t, \mathbf{r}, \mathbf{v})$ on message $p \in R_q$, the verifier computes the matrix $\mathbf{A}_t$ as above and checks the following conditions hold or not. If yes, he outputs 1. Otherwise, he outputs 0.

$$\begin{cases} \mathbf{A}_t \cdot \mathbf{v} = \mathbf{F} \cdot \mathsf{rdec}(\mathbf{F}_0 \cdot \mathbf{r} + \mathbf{F}_1 \cdot \mathsf{rdec}(p)) + u, \\ \|\mathbf{r}\|_\infty \leq \beta, \quad \|\mathbf{v}\|_\infty \leq \beta. \end{cases}$$

Remark 1. We remark that $\mathbf{p} = \mathsf{rdec}(p) \in R^\ell$ and $\mathsf{rdec}(y) \in R^\ell$ are ring vectors with coefficients in the set $\{-1, 0, 1\}$ while Ducas-Micciancio signature scheme handles ring vectors with binary coefficients. However, this does not affect the security of the Ducas-Micciancio signature scheme.

Lemma 1 ([20,21]). *If we assume there are at most polynomial number of signature queries and the* $\mathsf{RSIS}_{n,\overline{m},q,\widetilde{\mathcal{O}}(n^2)}$ *problem is hard, then the above variant of Ducas-Micciancio signature scheme is existentially unforgeable against adaptive chosen message attacks.*

2.4 Zero-Knowledge Argument Systems and Stern-like Protocols

We will work with statistical zero-knowledge argument systems, namely, interactive protocols where the zero-knowledge property holds against *any* cheating verifier, while the soundness property only holds against *computationally bounded* cheating provers. More formally, let the set of statements-witnesses $\mathrm{R} = \{(y, w)\} \in \{0,1\}^* \times \{0,1\}^*$ be an NP relation. A two-party game $\langle \mathcal{P}, \mathcal{V} \rangle$ is called an interactive argument system for the relation R with soundness error e if the following conditions hold:

- $\mathsf{Completeness}$. If $(y, w) \in \mathrm{R}$ then $\Pr\big[\langle \mathcal{P}(y, w), \mathcal{V}(y) \rangle = 1\big] = 1$.
- $\mathsf{Soundness}$. If $(y, w) \notin \mathrm{R}$, then $\forall$ PPT $\widehat{\mathcal{P}}$: $\Pr[\langle \widehat{\mathcal{P}}(y, w), \mathcal{V}(y) \rangle = 1] \leq e$.

An argument system is called statistical zero-knowledge if there exists a PPT simulator $\mathcal{S}(y)$ having oracle access to any $\widehat{\mathcal{V}}(y)$ and producing a simulated transcript that is statistically close to the one of the real interaction between $\mathcal{P}(y, w)$ and $\widehat{\mathcal{V}}(y)$. A related notion is argument of knowledge, which requires the witness-extended emulation property. For protocols consisting of 3 moves (*i.e.*, commitment-challenge-response), witness-extended emulation is implied by *special soundness* [27], where the latter assumes that there exists a PPT extractor which takes as input a set of valid transcripts with respect to all possible values of the "challenge" to the same "commitment", and outputs w' such that $(y, w') \in \mathrm{R}$.

Stern-like protocols. The statistical zero-knowledge arguments of knowledge presented in this work are Stern-like [57] protocols. In particular, they are Σ-protocols in the generalized sense defined in [29,6] (where 3 valid transcripts are needed for extraction, instead of just 2). The basic protocol consists of 3 moves: commitment, challenge, response. If a statistically hiding and computationally binding string commitment scheme, such as the KTX scheme [30], is employed in the first move, then one obtains a statistical zero-knowledge argument of knowledge (ZKAoK) with perfect completeness, constant soundness error $2/3$. In many applications, the protocol is repeated $\kappa = \omega(\log \lambda)$ times to make the soundness error negligibly small in λ.

An abstraction of Stern's protocol. We recall an abstraction of Stern's protocol, proposed in [35]. Let K, L, q be positive integers, where $L \geq K$ and $q \geq 2$, and let VALID be a subset of $\{-1, 0, 1\}^L$. Suppose that $\mathcal{S}$ is a finite set such that one can associate every $\phi \in \mathcal{S}$ with a permutation Γ_ϕ of L elements, satisfying the following conditions:

$$\begin{cases} \mathbf{w} \in \mathsf{VALID} \iff \Gamma_\phi(\mathbf{w}) \in \mathsf{VALID}, \\ \text{If } \mathbf{w} \in \mathsf{VALID} \text{ and } \phi \text{ is uniform in } \mathcal{S}, \text{ then } \Gamma_\phi(\mathbf{w}) \text{ is uniform in } \mathsf{VALID}. \end{cases} \tag{1}$$

We aim to construct a statistical ZKAoK for the following abstract relation:

$$\mathrm{R}_{\text{abstract}} = \left\{(\mathbf{M}, \mathbf{u}), \mathbf{w} \in \mathbb{Z}_q^{K \times L} \times \mathbb{Z}_q^K \times \mathsf{VALID} : \mathbf{M} \cdot \mathbf{w} = \mathbf{u} \bmod q.\right\}$$

The conditions in (1) play a crucial role in proving in ZK that $\mathbf{w} \in \mathsf{VALID}$: To do so, the prover samples $\phi \xleftarrow{\$} \mathcal{S}$ and let the verifier check that $\Gamma_\phi(\mathbf{w}) \in \mathsf{VALID}$, while the latter cannot learn any additional information about $\mathbf{w}$ thanks to the randomness of ϕ. Furthermore, to prove in ZK that the linear equation holds, the prover samples a masking vector $\mathbf{r}_w \xleftarrow{\$} \mathbb{Z}_q^L$, and convinces the verifier instead that $\mathbf{M} \cdot (\mathbf{w} + \mathbf{r}_w) = \mathbf{M} \cdot \mathbf{r}_w + \mathbf{u} \bmod q$.

The interaction between prover $\mathcal{P}$ and verifier $\mathcal{V}$ is described in Figure 1. The protocol employs a statistically hiding and computationally binding string commitment scheme COM (e.g., the RSIS-based scheme from [30]).

1. **Commitment:** Prover samples $\mathbf{r}_w \xleftarrow{\$} \mathbb{Z}_q^L$, $\phi \xleftarrow{\$} \mathcal{S}$ and randomness ρ_1, ρ_2, ρ_3 for COM. Then he sends $\mathrm{CMT} = \big(C_1, C_2, C_3\big)$ to the verifier, where

$$C_1 = \mathsf{COM}(\phi, \mathbf{M} \cdot \mathbf{r}_w \bmod q; \rho_1), \;\; C_2 = \mathsf{COM}(\Gamma_\phi(\mathbf{r}_w); \rho_2),$$
$$C_3 = \mathsf{COM}(\Gamma_\phi(\mathbf{w} + \mathbf{r}_w \bmod q); \rho_3).$$

2. **Challenge:** The verifier sends a challenge $Ch \xleftarrow{\$} \{1, 2, 3\}$ to the prover.
3. **Response:** Depending on Ch, the prover sends RSP computed as follows:
   - $Ch = 1$: Let $\mathbf{t}_w = \Gamma_\phi(\mathbf{w})$, $\mathbf{t}_r = \Gamma_\phi(\mathbf{r}_w)$, and $\mathrm{RSP} = (\mathbf{t}_w, \mathbf{t}_r, \rho_2, \rho_3)$.
   - $Ch = 2$: Let $\phi_2 = \phi$, $\mathbf{w}_2 = \mathbf{w} + \mathbf{r}_w \bmod q$, and $\mathrm{RSP} = (\phi_2, \mathbf{w}_2, \rho_1, \rho_3)$.
   - $Ch = 3$: Let $\phi_3 = \phi$, $\mathbf{w}_3 = \mathbf{r}_w$, and $\mathrm{RSP} = (\phi_3, \mathbf{w}_3, \rho_1, \rho_2)$.

Verification: Receiving RSP, the verifier proceeds as follows:

- $Ch = 1$: Check that $\mathbf{t}_w \in \mathsf{VALID}$, $C_2 = \mathsf{COM}(\mathbf{t}_r; \rho_2)$, $C_3 = \mathsf{COM}(\mathbf{t}_w + \mathbf{t}_r \bmod q; \rho_3)$.
- $Ch = 2$: Check that $C_1 = \mathsf{COM}(\phi_2, \mathbf{M} \cdot \mathbf{w}_2 - \mathbf{u} \bmod q; \rho_1)$, $C_3 = \mathsf{COM}(\Gamma_{\phi_2}(\mathbf{w}_2); \rho_3)$.
- $Ch = 3$: Check that $C_1 = \mathsf{COM}(\phi_3, \mathbf{M} \cdot \mathbf{w}_3; \rho_1)$, $\;C_2 = \mathsf{COM}(\Gamma_{\phi_3}(\mathbf{w}_3); \rho_2)$.

In each case, the verifier outputs 1 if and only if all the conditions hold.

Fig. 1: Stern-like ZKAoK for the relation $\mathrm{R}_{\text{abstract}}$.

Theorem 1 ([35]). *Assume that* COM *is a statistically hiding and computationally binding string commitment scheme. Then, the protocol in Figure 1 is a statistical* ZKAoK *with perfect completeness, soundness error* $2/3$*, and communication cost* $\mathcal{O}(L \log q)$*. In particular:*

- *There exists a polynomial-time simulator that, on input* $(\mathbf{M}, \mathbf{u})$*, outputs an accepted transcript statistically close to that produced by the real prover.*

– *There exists a polynomial-time knowledge extractor that, on input a commitment* CMT *and* 3 *valid responses* $(\mathrm{RSP}_1, \mathrm{RSP}_2, \mathrm{RSP}_3)$ *to all* 3 *possible values of the challenge Ch, outputs* $\mathbf{w}' \in \mathsf{VALID}$ *such that* $\mathbf{M} \cdot \mathbf{w}' = \mathbf{u} \bmod q$.

The proof of the Theorem 1, appeared in [35], employs standard simulation and extraction techniques for Stern-like protocols (e.g., [30,41,36]). For completeness, we recall the proof in Appendix B.

Looking ahead, all the relations we consider in this work (Sections 3.2 and 4.2), will be reduced to instances of the above abstract protocol.

3 ZKAoK for the Ducas-Micciancio Signature Scheme

This section presents our statistical zero-knowledge argument of knowledge for a valid message-signature pair for the Ducas-Micciancio signature scheme [20,21]. In the process, we will need to prove knowledge of a witness vector of the "mixing" form

$$\big(\mathbf{z} \parallel t_0 \cdot \mathbf{z} \parallel \ \ldots \ \parallel t_{c_d - 1} \cdot \mathbf{z} \big), \tag{2}$$

where $\mathbf{z} \in \{-1, 0, 1\}^{\mathfrak{m}}$ and $\mathbf{t} = (t_0, \ldots, t_{c_d - 1})^\top \in \{0, 1\}^{c_d}$ for some positive integers $\mathfrak{m}$ and c_d.

We note that, in their ZK protocol for the Boyen signature [11], Ling et al. [42] also derived a vector of similar form. To handle such a vector in the Stern's framework [57], Ling et al. used a permutation in the symmetric group $\mathcal{S}_{3\mathfrak{m}}$ to hide the value of $\mathbf{z}$ and a permutation in the symmetric group $\mathcal{S}_{2c_d}$ to hide the value of $\mathbf{t}$. As a consequence, the cost of communicating the permutations from the prover to the verifier is $3\mathfrak{m} \log \mathfrak{m} + 2c_d \log c_d$ bits. This is sub-optimal, because the cost is much larger than the number of secret bits.

In Section 3.1, we put forward a refined permuting technique in which the total cost for the permutations is exactly the total bit-size of $\mathbf{z}$ and $\mathbf{t}$. We then employ this technique as a building block for our ZK protocol in Section 3.2.

3.1 A Refined Permuting Technique

We first observe that the coefficients of the vector described in (2) are highly correlated: most of them are products of t_i and z_j, where both t_i and z_j do appear at other positions. Thus, to prove the well-formedness of such a vector, we have to solve two sub-problems: (i) proving that a secret integer z is an element of the set $\{-1, 0, 1\}$; (ii) proving that a secret integer y is the product of secret integers $t \in \{0, 1\}$ and $z \in \{-1, 0, 1\}$. Furthermore, these sub-protocols must be compatible and extendable, so that we can additionally prove that the *same* t and z satisfy other relations.

Technique for proving that $z \in \{-1, 0, 1\}$**.** For any integer a, let us denote by $[a]_3$ the integer $a' \in \{-1, 0, 1\}$ such that $a' = a \bmod 3$. For integer $z \in \{-1, 0, 1\}$, we define the 3-dimensional vector $\mathsf{enc}_3(z)$ as follows:

$$\mathsf{enc}_3(z) = \big([z + 1]_3, [z]_3, [z - 1]_3\big)^\top \in \{-1, 0, 1\}^3.$$

Namely, $\mathsf{enc}_3(-1) = (0,-1,1)^\top$, $\mathsf{enc}_3(0) = (1,0,-1)^\top$ and $\mathsf{enc}_3(1) = (-1,1,0)^\top$.

Now, for any integer $e \in \{-1,0,1\}$, define the permutation π_e that transforms vector $\mathbf{v} = (v^{(-1)}, v^{(0)}, v^{(1)})^\top \in \mathbb{Z}^3$ into vector

$$\pi_e(\mathbf{v}) = (v^{([-e-1]_3)}, v^{([-e]_3)}, v^{([-e+1]_3)})^\top.$$

We then observe that, for any $z, b \in \{-1,0,1\}$, the following equivalence holds.

$$\mathbf{v} = \mathsf{enc}_3(z) \iff \pi_e(\mathbf{v}) = \mathsf{enc}_3([z+e]_3). \tag{3}$$

In the framework of Stern's protocol, the above technique in fact allows us to prove knowledge of $z \in \{-1,0,1\}$, where z may satisfy other relations. To do this, we first extend z to $\mathbf{v} = \mathsf{enc}_3(z)$. Then, to show that $\mathbf{v}$ is a well-formed extension, we pick a uniformly random $e \xleftarrow{\$} \{-1,0,1\}$, and send $\pi_e(\mathbf{v})$ to the verifier. Thanks to the equivalence observed in (3), when seeing that $\pi_e(\mathbf{v}) = \mathsf{enc}_3([z+e]_3)$, the verifier should be convinced that $\mathbf{v} = \mathsf{enc}_3(z)$, which implies that $z \in \{-1,0,1\}$. Meanwhile, since e acts as a "one-time pad", the value of z is completely hidden from the verifier. Furthermore, to prove that z satisfies other relations, we can use the same "one-time pad" e at other appearances of z. An example of that is to prove that z is involved in a product $t \cdot z$, which we now present.

Technique for proving that $y = t \cdot z$. For any $b \in \{0,1\}$, we denote by $\overline{b}$ the bit $1-b$. The addition operation modulo 2 is denoted by $\oplus$.

For any $t \in \{0,1\}$ and $z \in \{-1,0,1\}$, we construct the 6-dimensional integer vector $\mathsf{ext}(t,z) \in \{-1,0,1\}^6$ as follows:

$$\mathsf{ext}(t,z) = \big(\ \overline{t} \cdot [z+1]_3,\ \ t \cdot [z+1]_3,\ \ \overline{t} \cdot [z]_3,\ \ t \cdot [z]_3,\ \ \overline{t} \cdot [z-1]_3,\ \ t \cdot [z-1]_3\ \big)^\top.$$

Now, for any $b \in \{0,1\}$ and $e \in \{-1,0,1\}$, we define the permutation $\psi_{b,e}(\cdot)$ that transforms vector

$$\mathbf{v} = \big(v^{(0,-1)}, v^{(1,-1)}, v^{(0,0)}, v^{(1,0)}, v^{(0,1)}, v^{(1,1)}\big)^\top \in \mathbb{Z}^6$$

into vector

$$\psi_{b,e}(\mathbf{v}) = \big(v^{(b,[-e-1]_3)},\ v^{(\overline{b},[-e-1]_3)},\ v^{(b,[-e]_3)},\ v^{(\overline{b},[-e]_3)},\ v^{(b,[-e+1]_3)},\ v^{(\overline{b},[-e+1]_3)}\big)^\top.$$

We then observe that the following equivalence holds for any $t, b \in \{0,1\}$ and any $z, e \in \{-1,0,1\}$.

$$\mathbf{v} = \mathsf{ext}(t,z) \iff \psi_{b,e}(\mathbf{v}) = \mathsf{ext}(\ t \oplus b,\ [z+e]_3\ ). \tag{4}$$

Example 1. Let $t = 1$ and $z = -1$. Then we have

$$\mathbf{v} = \mathsf{ext}(t,z) = (0,0,0,-1,0,1)^\top = (v^{(0,-1)}, v^{(1,-1)}, v^{(0,0)}, v^{(1,0)}, v^{(0,1)}, v^{(1,1)})^\top.$$

Suppose that $b = 0$ and $e = 1$, then

$$\psi_{b,e}(\mathbf{v}) = (v^{(0,1)}, v^{(1,1)}, v^{(0,-1)}, v^{(1,-1)}, v^{(0,0)}, v^{(1,0)})^\top = (0, 1, 0, 0, 0, -1)^\top,$$

which is equal to $\mathsf{ext}(1, 0) = \mathsf{ext}(1 \oplus 0, [-1 + 1]_3)$.

In the framework of Stern's protocol, the above technique will be used to prove that $y = t \cdot z$, as follows. We first extend y to $\mathbf{v} = \mathsf{ext}(t, z)$. Then, to prove that $\mathbf{v}$ is well-formed, we sample $b \xleftarrow{\$} \{0, 1\}$, $e \xleftarrow{\$} \{-1, 0, 1\}$, and demonstrate to the verifier that $\psi_{b,e}(\mathbf{v}) = \mathsf{ext}(\, t \oplus b,\, [z + e]_3\,)$. Thanks to the equivalence observed in (4), the verifier should be convinced that $\mathbf{v}$ is well-formed, implying that the original integer y does have the form $t \cdot z$. Meanwhile, the random integers b, e essentially act as "one-time pads" that perfectly hide the values of t and z, respectively. Moreover, if we want to prove that the same t, z appear elsewhere, we can use the same b, e at those places.

Next, we will describe the somewhat straightforward generalizations of the above two core techniques, which enable us to prove knowledge of vector $\mathbf{z} \in \{-1, 0, 1\}^{\mathfrak{m}}$ as well as vector of the form (2). Based on the above discussions, one can see that the target is to obtain equivalences similar to (3) and (4), which are useful in Stern's framework.

Proving that $\mathbf{z} \in \{-1, 0, 1\}^{\mathfrak{m}}$. For any vector $\mathbf{a} \in \mathbb{Z}^{\mathfrak{m}}$, we will also use the notation $[\mathbf{a}]_3$ to denote the vector $\mathbf{a}' \in \{-1, 0, 1\}^{\mathfrak{m}}$ such that $\mathbf{a}' = \mathbf{a} \bmod 3$.

For $\mathbf{z} = (z_1, \ldots, z_{\mathfrak{m}})^\top \in \{-1, 0, 1\}^{\mathfrak{m}}$, we define the following extension:

$$\mathsf{enc}(\mathbf{z}) = \big(\, \mathsf{enc}_3(z_1) \,\|\, \ldots \,\|\, \mathsf{enc}_3(z_{\mathfrak{m}}) \,\big) \in \{-1, 0, 1\}^{3\mathfrak{m}}.$$

For any vector $\mathbf{e} = (e_1, \ldots, e_{\mathfrak{m}})^\top \in \{-1, 0, 1\}^{\mathfrak{m}}$, we define the permutation $\mathit{\Pi}_{\mathbf{e}}$ that acts as follows. When applied to vector $\mathbf{v} = (\mathbf{v}_1 \| \ldots \| \mathbf{v}_{\mathfrak{m}}) \in \mathbb{Z}^{3\mathfrak{m}}$ consisting of $\mathfrak{m}$ blocks of size 3, it transforms $\mathbf{v}$ into vector:

$$\mathit{\Pi}_{\mathbf{e}}(\mathbf{v}) = \big(\pi_{e_1}(\mathbf{v}_1) \| \ldots \| \pi_{e_{\mathfrak{m}}}(\mathbf{v}_{\mathfrak{m}})\big).$$

It then follows from (3) that the following holds for any $\mathbf{z}, \mathbf{e} \in \{-1, 0, 1\}^{\mathfrak{m}}$.

$$\mathbf{v} = \mathsf{enc}(\mathbf{z}) \iff \mathit{\Pi}_{\mathbf{e}}(\mathbf{v}) = \mathsf{enc}([\mathbf{z} + \mathbf{e}]_3). \tag{5}$$

Handling a "mixing" vector. We now tackle the "mixing" vector discussed earlier, i.e.,

$$\mathbf{y} = \big(\, \mathbf{z} \,\|\, t_0 \cdot \mathbf{z} \,\|\, \ \ldots\ \,\|\, t_{c_d - 1} \cdot \mathbf{z} \,\big).$$

For any $\mathbf{z} = (z_1, \ldots, z_{\mathfrak{m}})^\top \in \{-1, 0, 1\}^{\mathfrak{m}}$ and $\mathbf{t} = (t_0, \ldots, t_{c_d - 1})^\top \in \{0, 1\}^{c_d}$, we define vector $\mathsf{mix}(\mathbf{t}, \mathbf{z}) \in \{-1, 0, 1\}^{3\mathfrak{m} + 6\mathfrak{m} c_d}$ of the form:

$$\big(\, \mathsf{enc}(\mathbf{z}) \,\|\, \mathsf{ext}(t_0, z_1) \,\|\, \ldots \,\|\, \mathsf{ext}(t_0, z_{\mathfrak{m}}) \,\|\, \ldots \,\|\, \mathsf{ext}(t_{c_d - 1}, z_1) \,\|\, \ldots \,\|\, \mathsf{ext}(t_{c_d - 1}, z_{\mathfrak{m}}) \,\big),$$

which is an extension of vector $\mathbf{y}$. Next, for $\mathbf{b} = (b_0, \cdots, b_{c_d-1})^\top \in \{0,1\}^{c_d}$ and $\mathbf{e} = (\mathbf{e}_1, \ldots, e_{\mathfrak{m}}) \in \{-1,0,1\}^{\mathfrak{m}}$, we define the permutation $\Psi_{\mathbf{b},\mathbf{e}}$ that acts as follows. When applied to vector

$$\mathbf{v} = \big(\mathbf{v}_{-1} \parallel \mathbf{v}_{0,1} \parallel \ldots \parallel \mathbf{v}_{0,\mathfrak{m}} \parallel \ldots \parallel \mathbf{v}_{c_d-1,1} \parallel \ldots \parallel \mathbf{v}_{c_d-1,\mathfrak{m}}\big) \in \mathbb{Z}^{3\mathfrak{m}+6\mathfrak{m}c_d},$$

where block $\mathbf{v}_{-1}$ has length $3\mathfrak{m}$ and each block $\mathbf{v}_{i,j}$ has length 6, it transforms $\mathbf{v}$ into vector

$$\begin{aligned}\Psi_{\mathbf{b},\mathbf{e}}(\mathbf{v}) = \big(\mathit{\Pi}_{\mathbf{e}}(\mathbf{v}_{-1}) \| \;\; & \psi_{b_0,e_1}(\mathbf{v}_{0,1}) \| \ldots \| \psi_{b_0,e_{\mathfrak{m}}}(\mathbf{v}_{0,\mathfrak{m}}) \| \ldots \| \\ & \psi_{b_{c_d-1},e_1}(\mathbf{v}_{c_d-1,1}) \| \ldots \| \psi_{b_{c_d-1},e_{\mathfrak{m}}}(\mathbf{v}_{c_d-1,\mathfrak{m}})\big).\end{aligned}$$

Then, observe that the following desirable equivalence holds for all $\mathbf{t}, \mathbf{b} \in \{0,1\}^{c_d}$ and $\mathbf{z}, \mathbf{e} \in \{-1,0,1\}^{\mathfrak{m}}$.

$$\mathbf{v} = \mathsf{mix}(\mathbf{t}, \mathbf{z}) \iff \Psi_{\mathbf{b},\mathbf{e}}(\mathbf{v}) = \mathsf{mix}(\, \mathbf{t} \oplus \mathbf{b},\, [\mathbf{z} + \mathbf{e}]_3 \,). \tag{6}$$

3.2 Zero-Knowledge Protocol for the Ducas-Micciancio Signature

We now present our statistical ZKAoK of a valid message-signature pair for the Ducas-Micciancio signature scheme. Let $n, q, m, k, \overline{m}, \ell, \beta, d, c_0, \ldots, c_d$ as specified in Section 2.3. The protocol can be summarized as follows.

- The public input consists of

$$\mathbf{A}, \mathbf{F}_0 \in R_q^{1\times\overline{m}}; \quad \mathbf{A}_{[0]}, \ldots, \mathbf{A}_{[d]} \in R_q^{1\times k}; \quad \mathbf{F}, \mathbf{F}_1 \in R_q^{1\times\ell}; \quad u \in R_q.$$

- The prover's secret input consists of message $p \in R_q$ and signature $(t, \mathbf{r}, \mathbf{v})$, where

$$\begin{cases} t = (t_0, \ldots, t_{c_1-1}, \ldots, t_{c_{d-1}}, \ldots, t_{c_d-1})^\top \in \{0,1\}^{c_d}; \\ \mathbf{r} \in R^{\overline{m}}; \quad \mathbf{v} = (\mathbf{s} \| \mathbf{z}) \in R^{\overline{m}+k}; \quad \mathbf{s} \in R^{\overline{m}}; \quad \mathbf{z} \in R^k; \end{cases}$$

- The prover's goal is to prove in zero-knowledge that $\|\mathbf{r}\|_\infty \le \beta$, $\|\mathbf{v}\|_\infty \le \beta$, and that the equation

$$\mathbf{A} \cdot \mathbf{s} + \mathbf{A}_{[0]} \cdot \mathbf{z} + \sum_{i=1}^{d} \mathbf{A}_{[i]} \cdot t_{[i]} \cdot \mathbf{z} = \mathbf{F} \cdot \mathbf{y} + u \tag{7}$$

holds for $\big\{t_{[i]} = \sum_{j=c_{i-1}}^{c_i-1} t_j \cdot X^j\big\}_{i=1}^{d}$ and

$$\mathbf{y} = \mathsf{rdec}\left(\mathbf{F}_0 \cdot \mathbf{r} + \mathbf{F}_1 \cdot \mathsf{rdec}(p)\right) \in R^\ell. \tag{8}$$

Our strategy is to reduce the considered statement to an instance of the abstract protocol from Section 2.4. The reduction consists of 2 steps.

Decomposing-Unifying. In the first step, we will employ the decomposition techniques from Section 2.2 together with the notations rot and τ from

Section 2.1 to transform equations (7) and (8) into one equation of the form $\mathbf{M}_0 \cdot \mathbf{w}_0 = \mathbf{u} \bmod q$, where $\mathbf{M}_0, \mathbf{u}$ are public, and the coefficients of vector $\mathbf{w}_0$ are in the set $\{-1, 0, 1\}$.

Let $\mathbf{s}^\star = \tau(\mathsf{rdec}_\beta(\mathbf{s})) \in \{-1,0,1\}^{n\overline{m}\delta_\beta}$, $\mathbf{z}^\star = \tau(\mathsf{rdec}_\beta(\mathbf{z})) \in \{-1,0,1\}^{nk\delta_\beta}$ and $\mathbf{r}^\star = \tau(\mathsf{rdec}_\beta(\mathbf{r})) \in \{-1,0,1\}^{n\overline{m}\delta_\beta}$. Then, we observe that, equation (7) is equivalent to,

$$[\mathsf{rot}(\mathbf{A}_{[0]}) \cdot \mathbf{H}_{k,\beta}] \cdot \mathbf{z}^\star + \sum_{i=1}^{d} \sum_{j=c_{i-1}}^{c_i - 1} [\mathsf{rot}(\mathbf{A}_{[i]} \cdot X^j) \cdot \mathbf{H}_{k,\beta}] \cdot t_j \cdot \mathbf{z}^* +$$

$$[\mathsf{rot}(\mathbf{A}) \cdot \mathbf{H}_{\overline{m},\beta}] \cdot \mathbf{s}^\star - [\mathsf{rot}(\mathbf{F})] \cdot \tau(\mathbf{y}) = \tau(u) \bmod q,$$

and equation (8) is equivalent to

$$[\mathsf{rot}(\mathbf{F}_0) \cdot \mathbf{H}_{\overline{m},\beta}] \cdot \mathbf{r}^\star + [\mathsf{rot}(\mathbf{F}_1)] \cdot \tau(\mathsf{rdec}(p)) - [\mathbf{H}] \cdot \tau(\mathbf{y}) = \mathbf{0} \bmod q.$$

Now, using basic algebra, we can manipulate the two derived equations: rearranging the secret vectors and combining them, as well as concatenating the public matrices (namely, those written inside $[\,\cdot\,]$) accordingly. As a result, we obtain an unifying equation of the form:

$$\mathbf{M}_0 \cdot \mathbf{w}_0 = \mathbf{u} \bmod q,$$

where $\mathbf{u} = (\tau(u) \parallel \mathbf{0}) \in \mathbb{Z}_q^{2n}$ and $\mathbf{M}_0$ are public, and $\mathbf{w}_0 = (\mathbf{w}_1 \parallel \mathbf{w}_2)$, with

$$\begin{cases} \mathbf{w}_1 = (\mathbf{z}^\star \parallel t_0 \cdot \mathbf{z}^\star \parallel \ \ldots \ \parallel t_{c_d - 1} \cdot \mathbf{z}^\star) \in \{-1,0,1\}^{(k\delta_\beta + c_d k \delta_\beta)n}; \\ \mathbf{w}_2 = (\mathbf{s}^\star \parallel \mathbf{r}^\star \parallel \tau(\mathbf{y}) \parallel \tau(\mathsf{rdec}(p))) \in \{-1,0,1\}^{(\overline{m}\delta_\beta + \ell)2n}. \end{cases}$$

Extending-Permuting. In this second step, we will transform the equation $\mathbf{M}_0 \cdot \mathbf{w}_0 = \mathbf{u} \bmod q$ obtained in the first step into an equation of the form $\mathbf{M} \cdot \mathbf{w} = \mathbf{u} \bmod q$, where the secret vector $\mathbf{w}$ satisfies the conditions required by the abstract protocol. In the process, we will employ the techniques introduced in Section 3.1.

Specifically, we extend the blocks of vector $\mathbf{w}_0 = (\mathbf{w}_1 \| \mathbf{w}_2)$ as follows.

$$\begin{aligned} \mathbf{w}_1 &\mapsto \mathbf{w}_1' = \mathsf{mix}\big(t, \mathbf{z}^\star\big) \in \{-1,0,1\}^{L_1}; \\ \mathbf{w}_2 &\mapsto \mathbf{w}_2' = \mathsf{enc}(\mathbf{w}_2) \in \{-1,0,1\}^{L_2}. \end{aligned} \tag{9}$$

Then we form vector $\mathbf{w} = (\mathbf{w}_1' \| \mathbf{w}_2') \in \{-1,0,1\}^L$, where

$$L = L_1 + L_2;\ L_1 = (k\delta_\beta + 2c_d k\delta_\beta)3n;\ L_2 = (\overline{m}\delta_\beta + \ell)6n.$$

At the same time, we insert suitable zero-columns to matrix $\mathbf{M}_0$ to obtain matrix $\mathbf{M} \in \mathbb{Z}_q^{2n \times L}$ such that $\mathbf{M} \cdot \mathbf{w} = \mathbf{M}_0 \cdot \mathbf{w}_0$.

Up to this point, we have transformed the considered relations into one equation of the desired form $\mathbf{M} \cdot \mathbf{w} = \mathbf{u} \bmod q$. We now specify the set VALID that contains the obtained vector $\mathbf{w}$, the set $\mathcal{S}$ and permutations $\{\Gamma_\phi : \phi \in \mathcal{S}\}$, such that the conditions in (1) hold.
Define VALID as the set of all vectors $\mathbf{v}' = (\mathbf{v}_1' \| \mathbf{v}_2') \in \{-1,0,1\}^L$, satisfying the following:

– There exist $t \in \{0,1\}^{c_d}$ and $\mathbf{z}^\star \in \{-1,0,1\}^{nk\delta_\beta}$ such that $\mathbf{v}'_1 = \mathsf{mix}(t, \mathbf{z}^\star)$.
– There exists $\mathbf{w}_2 \in \{-1,0,1\}^{(\overline{m}\delta_\beta+\ell)2n}$ such that $\mathbf{v}'_2 = \mathsf{enc}(\mathbf{w}_2)$.

Clearly, our vector $\mathbf{w}$ belongs to this tailored set VALID.

Now, let $\mathcal{S} = \{0,1\}^{c_d} \times \{-1,0,1\}^{nk\delta_\beta} \times \{-1,0,1\}^{(\overline{m}\delta_\beta+\ell)2n}$, and associate every element $\phi = (\mathbf{b}, \mathbf{e}, \mathbf{f}) \in \mathcal{S}$ with permutation $\mathit{\Gamma}_\phi$ that acts as follows. When applied to vector $\mathbf{v}^\star = (\mathbf{v}_1^\star \| \mathbf{v}_2^\star) \in \mathbb{Z}^L$, where $\mathbf{v}_1^\star \in \mathbb{Z}^{L_1}$ and $\mathbf{v}_2^\star \in \mathbb{Z}^{L_2}$, it transforms $\mathbf{v}^\star$ into vector

$$\mathit{\Gamma}_\phi(\mathbf{v}^\star) = \big(\mathit{\Psi}_{\mathbf{b},\mathbf{e}}(\mathbf{v}_1^\star) \parallel \mathit{\Pi}_{\mathbf{f}}(\mathbf{v}_2^\star) \big).$$

Based on the equivalences observed in (5) and (6), it can be checked that VALID, $\mathcal{S}$ and $\mathit{\Gamma}_\phi$ satisfy the conditions specified in (1). In other words, we have reduced the considered statement to an instance of the abstract protocol from Section 2.4.

The interactive protocol. Given the above preparations, our interactive protocol works as follows.

– The public input consists of matrix $\mathbf{M}$ and vector $\mathbf{u}$, which are built from $\mathbf{A}$, $(\mathbf{A}_{[0]}, \ldots, \mathbf{A}_{[d]}, \mathbf{F}, \mathbf{F}_0, \mathbf{F}_1, u)$, as discussed above.
– The prover's witness is vector $\mathbf{w} \in \mathsf{VALID}$, which is obtained from the original witnesses $(p, t, \mathbf{r}, \mathbf{v})$, as described above.

Both parties then run the protocol of Figure 1. The protocol uses the KTX string commitment scheme COM, which is statistically hiding and computationally binding under the (R)SIS assumption. We therefore obtain the following result, as a corollary of Theorem 1.

Theorem 2. *Assume that* COM *is a statistically hiding and computationally binding string commitment scheme. Then the protocol described above is a statistical* ZKAoK *of a valid message-signature pair for the Ducas-Micciancio signature scheme, with perfect completeness, soundness error* $2/3$ *and communication cost* $\widetilde{\mathcal{O}}(\lambda)$.

Proof. For simulation, we simply run the simulator of Theorem 1. As for extraction, we invoke the knowledge extractor of Theorem 1 to obtain a vector $\mathbf{w}' \in \mathsf{VALID}$ such that $\mathbf{M} \cdot \mathbf{w}' = \mathbf{u} \bmod q$. Then, by "backtracking" the transformations being done, we can extract from $\mathbf{w}'$ a satisfying witness $(p', t', \mathbf{r}', \mathbf{v}')$ for the considered statement.

The perfect completeness, soundness error and communication cost of the protocol directly follow from those of the abstract protocol in Section 2.4. In particular, the communication cost is:

$$\mathcal{O}(L \cdot \log q) = \mathcal{O}\big((k\delta_\beta + 2c_d k\delta_\beta)3n \cdot \log q + (\overline{m}\delta_\beta + \ell)6n \cdot \log q\big) = \mathcal{O}(n \cdot \log^4 n) = \widetilde{\mathcal{O}}(\lambda),$$

for the setting of parameters for the Ducas-Micciancio signature in Section 2.3. □

4 Constant-Size Group Signatures from Lattices

In Section 4.1, we recall the syntax, correctness and security requirements of the (partially) dynamic group signatures, as in the model of Bellare, Shi and Zhang [5]. In Section 4.2, we describe our main zero-knowledge argument, which will be used as a building block in our group signature scheme constructed in Section 4.3.

4.1 Dynamic Group Signatures

In this section, we recall the syntax, correctness and security definitions of the (partially) dynamic group signatures, as put forward by Bellare, Shi and Zhang [5]. A dynamic group signature scheme involves a trusted party who generates the initial keys, an authority named issuer, an authority named opener and a set of users who are potential group members. The scheme consists of the following polynomial-time algorithms.

$\mathsf{GKg}(\lambda)$**:** Given the security parameter λ, the trusted party runs this algorithm to generate a triple $(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok})$. The issue key ik is given to the issuer, the opening key ok is given to the opener and the group public key gpk is made public.

$\mathsf{UKg}(\lambda)$**:** A user who intends to be a group member runs this algorithm to obtain a personal key pair $(\mathsf{upk}, \mathsf{usk})$. It is assumed that upk is public.

$\langle \mathsf{Join}, \mathsf{Iss} \rangle$**:** This is an interactive protocol run by the issuer and a user. If it completes successfully, the issuer registers this user to the group and this user becomes a group member. The final state of the Join is the secret signing key gsk_i while the final state of the Iss is the registration information $\mathbf{reg}[i]$ stored in the registration table $\mathbf{reg}$.

$\mathsf{Sign}(\mathsf{gpk}, \mathsf{gsk}_i, M)$**:** A group member, using his group signing key gsk_i, runs this algorithm to obtain a signature Σ on message M.

$\mathsf{Verify}(\mathsf{gpk}, M, \Sigma)$**:** This algorithm outputs 1/0 indicating whether or not Σ is a valid signature on message M, with respect to the group public key gpk.

$\mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathbf{reg}, M, \Sigma)$**:** Given gpk, a message-signature pair (M, Σ) and ok, the opener, who has read-access to the registration table $\mathbf{reg}$, runs this algorithm to obtain a pair (i, Π_{open}), where $i \in \mathbb{N} \cup \{\perp\}$. In case $i = \perp$, $\Pi_{\mathsf{open}} = \perp$.

$\mathsf{Judge}(\mathsf{gpk}, M, \Sigma, i, \mathsf{upk}_i, \Pi_{\mathsf{open}})$**:** This algorithm outputs 1/0 to check whether or not Π_{open} is a proof that i produced Σ, with respect to the group public key gpk and message M.

Now we recall the correctness and security definitions of dynamic group signatures below.

Correctness requires that for any signature generated by honest group members, the following should hold: the signature should be valid; the opening algorithm, given the message and signature, should correctly identify the signer; the proof returned by the opening algorithm should be accepted by the judge.

Full Anonymity requires that it is infeasible to recover the identity of a signer from a signature, even if the adversary is given access to the opening oracle. As pointed out by [4,5], it is sufficient that the adversary is unable to distinguish which of two signers of its choice signed a targeted message of its choice.

Traceability requires that every valid signature should be traced to some group member and the opener is able to generate a proof accepted by the judge.

Non-frameability requires that the adversary is unable to generate a proof, which is accepted by the judge, that an honest user generated a valid signature unless this user really did generate this signature.

Formal definitions of correctness and security requirements are deferred to Appendix A.

4.2 The Underlying Zero-Knowledge Argument System

Before describing our group signature scheme in Section 4.3, let us first present the statistical ZKAoK that will be invoked by the signer when generating group signatures. The protocol is an extension of the one for the Ducas-Micciancio signature from Section 3.2, for which the prover additionally convinces the verifier of the following two facts.

1. He knows a secret key $\mathbf{x} \in R^m$ corresponding to the public key $p \in R_q$, which satisfies $\|\mathbf{x}\|_\infty \leq 1$ and $\mathbf{B} \cdot \mathbf{x} = p$. Here, $\mathbf{B} \in R_q^{1\times m}$ is a public matrix.

2. He has correctly encrypted the vector $\mathsf{rdec}(p) \in R^\ell$ to a given ciphertext $(\mathbf{c}_{1,1}, \mathbf{c}_{1,2}, \mathbf{c}_{2,1}, \mathbf{c}_{2,2}) \in (R_q^\ell)^4$, under public key $(\mathbf{a}, \mathbf{b}_1, \mathbf{b}_2) \in (R_q^\ell)^3$. To this end, he proves that equations

$$\mathbf{c}_{i,1} = \mathbf{a} \cdot g_i + \mathbf{e}_{i,1}, \quad \mathbf{c}_{i,2} = \mathbf{b}_i \cdot g_i + \mathbf{e}_{i,2} + \lfloor q/4 \rfloor \cdot \mathsf{rdec}(p), \tag{10}$$

   hold for B-bounded randomness $g_1, g_2 \in R$, and $\mathbf{e}_{1,1}, \mathbf{e}_{2,1}, \mathbf{e}_{1,2}, \mathbf{e}_{2,2} \in R^\ell$.

As the transformations for the "Ducas-Micciancio layer" have been established in Section 3.2, in the following, we only specify the transformations with respect to the newly appeared relations.

We will first apply the decomposition techniques in Section 2.2 to the secret objects.

- Let $\mathbf{x}^\star = \tau(\mathbf{x}) \in \{-1, 0, 1\}^{nm}$.
- For $i \in \{1, 2\}$, compute $\mathbf{g}_i^\star = \tau(\mathsf{rdec}_B(g_i)) \in \{-1, 0, 1\}^{n\delta_B}$.
- For $i \in \{1, 2\}$, compute $\mathbf{e}_{i,1}^\star = \tau(\mathsf{rdec}_B(\mathbf{e}_{i,1}))$ and $\mathbf{e}_{i,2}^\star = \tau(\mathsf{rdec}_B(\mathbf{e}_{i,2}))$. Note that they are vectors in $\{-1, 0, 1\}^{n\ell\delta_B}$.

Then the equation $\mathbf{B} \cdot \mathbf{x} = p$ can be translated as

$$[\mathsf{rot}(\mathbf{B})] \cdot \mathbf{x}^\star - [\mathbf{H}] \cdot \tau(\mathsf{rdec}(p)) = \mathbf{0}^n \bmod q. \tag{11}$$

Meanwhile, let $\mathbf{a} = (a_1, \ldots, a_\ell)^\top$, $\{\mathbf{b}_i = (b_{i,1}, \ldots, b_{i,\ell})^\top\}_{i=1,2}$, then equations in (10) can be rewritten as, for $i = 1, 2$,

$$\begin{bmatrix} \mathsf{rot}(a_1) \cdot \mathbf{H}_B \\ \vdots \\ \mathsf{rot}(a_\ell) \cdot \mathbf{H}_B \end{bmatrix} \cdot \mathbf{g}_i^\star + [\mathbf{H}_{\ell,B}] \cdot \mathbf{e}_{i,1}^\star = \tau(\mathbf{c}_{i,1}) \bmod q; \quad (12)$$

$$\begin{bmatrix} \mathsf{rot}(b_{i,1}) \cdot \mathbf{H}_B \\ \vdots \\ \mathsf{rot}(b_{i,\ell}) \cdot \mathbf{H}_B \end{bmatrix} \cdot \mathbf{g}_i^\star + [\mathbf{H}_{\ell,B}] \cdot \mathbf{e}_{i,2}^\star + \lfloor q/4 \rfloor \cdot \tau(\mathsf{rdec}(p)) = \tau(\mathbf{c}_{i,2}) \bmod q. \quad (13)$$

Before proceeding further, let us recall that, in the protocol for the Ducas-Micciancio signature from Section 3.2, at the end of the Decomposing-Unifying step, we did combine the secret objects into vectors $\mathbf{w}_1, \mathbf{w}_2$ of the form:

$$\begin{cases} \mathbf{w}_1 = (\mathbf{z}^\star \parallel t_0 \cdot \mathbf{z}^\star \parallel \ \ldots \ \parallel t_{c_d-1} \cdot \mathbf{z}^\star) \in \{-1, 0, 1\}^{(k\delta_\beta + c_d k \delta_\beta)n}; \\ \mathbf{w}_2 = (\mathbf{s}^\star \parallel \mathbf{r}^\star \parallel \tau(\mathbf{y}) \parallel \tau(\mathsf{rdec}(p))) \in \{-1, 0, 1\}^{(\overline{m}\delta_\beta + \ell)2n}. \end{cases}$$

Since vector $\tau(\mathsf{rdec}(p))$ has been counted as a block of vector $\mathbf{w}_2$, we now combine the newly appeared secret vectors in equations (11), (12), (13) into vector

$$\mathbf{w}_3 = \big(\mathbf{x}^\star \parallel \mathbf{g}_1^\star \parallel \mathbf{g}_2^\star \parallel \mathbf{e}_{1,1}^\star \parallel \mathbf{e}_{1,2}^\star \parallel \mathbf{e}_{2,1}^\star \parallel \mathbf{e}_{2,2}^\star \big) \in \{-1, 0, 1\}^{nm + 2n\delta_B + 4n\ell\delta_B},$$

and let $\mathbf{w}_4 = (\mathbf{w}_2 \| \mathbf{w}_3) \in \{-1, 0, 1\}^{L_4'}$, for $L_4' = (\overline{m}\delta_\beta + \ell)2n + nm + 2n\delta_B + 4n\ell\delta_B$.

Next, we extend $\mathbf{w}_4$ to vector $\mathbf{w}_4' = \mathsf{enc}(\mathbf{w}_4) \in \{-1, 0, 1\}^{L_4}$, where $L_4 = 3L_4'$, and form the vector

$$\widetilde{\mathbf{w}} = \big(\mathbf{w}_1' \| \mathbf{w}_4'\big) \in \{-1, 0, 1\}^{\widetilde{L}},$$

where $\mathbf{w}_1' = \mathsf{mix}\big(t, \mathbf{z}^\star\big) \in \{-1, 0, 1\}^{L_1}$ is the "mixing vector" obtained in (9), and $\widetilde{L} = L_1 + L_4$.

We remark that, by suitably concatenating/extending the matrices and vectors derived from the public input, we can obtain public matrix $\widetilde{\mathbf{M}}$ and public vector $\widetilde{\mathbf{u}}$ such that $\widetilde{\mathbf{M}} \cdot \widetilde{\mathbf{w}} = \widetilde{\mathbf{u}} \bmod q$. Having obtained this desired equation, we now proceed as in Section 3.2.

Define $\widetilde{\mathsf{VALID}}$ as the set of all vectors $\mathbf{v}' = (\mathbf{v}_1' \| \mathbf{v}_4') \in \{-1, 0, 1\}^{\widetilde{L}}$, satisfying the following:

- There exist $t \in \{0, 1\}^{c_d}$ and $\mathbf{z}^\star \in \{-1, 0, 1\}^{nk\delta_\beta}$ such that $\mathbf{v}_1' = \mathsf{mix}(t, \mathbf{z}^\star)$.
- There exists $\mathbf{w}_4 \in \{-1, 0, 1\}^{L_4'}$ such that $\mathbf{v}_4' = \mathsf{enc}(\mathbf{w}_4)$.

It can be seen that vector $\widetilde{\mathbf{w}}$ belongs to $\widetilde{\mathsf{VALID}}$.

Now, let $\widetilde{\mathcal{S}} = \{0, 1\}^{c_d} \times \{-1, 0, 1\}^{nk\delta_\beta} \times \{-1, 0, 1\}^{L_4'}$, and associate every element $\phi = (\mathbf{b}, \mathbf{e}, \mathbf{f}) \in \mathcal{S}$ with permutation $\widetilde{\Gamma}_\phi$ that acts as follows. When applied to vector $\mathbf{v}^\star = (\mathbf{v}_1^\star \| \mathbf{v}_4^\star) \in \mathbb{Z}^{\widetilde{L}}$, where $\mathbf{v}_1^\star \in \mathbb{Z}^{L_1}$ and $\mathbf{v}_2^\star \in \mathbb{Z}^{L_4}$, it transforms $\mathbf{v}^\star$ into vector

$$\widetilde{\Gamma}_\phi(\mathbf{v}^\star) = \big(\Psi_{\mathbf{b},\mathbf{e}}(\mathbf{v}_1^\star) \parallel \Pi_{\mathbf{f}}(\mathbf{v}_4^\star) \big).$$

Based on the equivalences observed in (5) and (6), it can be checked that $\widetilde{\mathsf{VALID}}$, $\widetilde{\mathcal{S}}$ and $\widetilde{\Gamma}_\phi$ satisfy the conditions specified in (1). In other words, we have reduced the considered statement to an instance of the abstract protocol from Section 2.4.

The interactive protocol. Given the above preparations, our interactive protocol works as follows.

- The public input consists of matrix $\widetilde{\mathbf{M}}$ and vector $\widetilde{\mathbf{u}}$, which are built from $\mathbf{A}$, $(\mathbf{A}_{[0]}, \ldots, \mathbf{A}_{[d]}$, $\mathbf{F}, \mathbf{F}_0, \mathbf{F}_1$, $u)$, and $\mathbf{B}$, $\mathbf{c}_{1,1}$, $\mathbf{c}_{1,2}$, $\mathbf{c}_{2,1}$, $\mathbf{c}_{2,2}$, $\mathbf{a}, \mathbf{b}_1, \mathbf{b}_2$, as discussed in Section 3.2 and above.
- The prover's witness is vector $\widetilde{\mathbf{w}} \in \widetilde{\mathsf{VALID}}$, which is obtained from the original witnesses $(p, t, \mathbf{r}, \mathbf{v}, \mathbf{x}, g_1, g_2, \mathbf{e}_{1,1}, \mathbf{e}_{2,1}, \mathbf{e}_{1,2}, \mathbf{e}_{2,2})$, as described in Section 3.2 and above.

Both parties then run the protocol of Figure 1. The protocol uses the KTX string commitment scheme COM, which is statistically hiding and computationally binding under the (R)SIS assumption. We therefore obtain the following result, as a corollary of Theorem 1.

Theorem 3. *Assume that* COM *is a statistically hiding and computationally binding string commitment scheme. Then the protocol described above is a statistical* ZKAoK *for the considered statement, with perfect completeness, soundness error* $2/3$ *and communication cost* $\widetilde{\mathcal{O}}(\lambda)$.

Proof. For simulation, we simply run the simulator of Theorem 1. As for extraction, we invoke the knowledge extractor of Theorem 1 to obtain a vector $\widetilde{\mathbf{w}}' \in \widetilde{\mathsf{VALID}}$ such that $\widetilde{\mathbf{M}} \cdot \widetilde{\mathbf{w}}' = \widetilde{\mathbf{u}} \bmod q$. Then, by "backtracking" all the transformations being done, we can extract from vector $\widetilde{\mathbf{w}}'$ a satisfying witness $(p', t', \mathbf{r}', \mathbf{v}', \mathbf{x}', g_1', g_2', \mathbf{e}_{1,1}', \mathbf{e}_{2,1}', \mathbf{e}_{1,2}', \mathbf{e}_{2,2}')$ for the considered statement.

The perfect completeness, soundness error and communication cost of the protocol directly follow from those of the abstract protocol in Section 2.4. In particular, the communication cost is:

$$\mathcal{O}(\widetilde{L} \cdot \log q) = \mathcal{O}\big((k\delta_\beta + c_d k \delta_\beta) n \cdot \log q + ((\overline{m}\delta_\beta + \ell)n + nm + n\delta_B + n\ell\delta_B) \cdot \log q\big),$$

which is of order $\mathcal{O}(n \cdot \log^4 n) = \widetilde{\mathcal{O}}(\lambda)$, for the setting of parameters we use in the group signature scheme of Section 4.3. □

4.3 Description of Our Scheme

In the description below, the Ducas-Micciancio signature scheme [20,21] as described in Section 2.3 is used to design a group signature scheme for (partially) dynamic groups. Group public key consists of three parts: (i) a verification key from the Ducas-Micciancio signature scheme, (ii) *two* public keys of an extended version of LPR encryption scheme [47] and (iii) a public matrix $\mathbf{B}$ for users to generate their short secret vectors together with public syndromes as user key pairs. The issue key is the corresponding signing key of the verification key while

the opening key is *any one* of the corresponding secret keys of the two public keys.

When a user joins the group, it first generates a short vector together with a public syndrome using matrix $\mathbf{B}$. It then interacts with the issuer. The issuer signs the public syndrome of this user using the issue key. If the interaction completes successfully, the user obtains a signature on his syndrome from the issuer while the issuer registers this user to the group.

Once registered as a group member, the user can sign messages on behalf of the group. When signing a message, it first encrypts the public syndrome twice using the two public keys. The user then generates a ZKAoK of his syndrome, of the signature on the syndrome obtained from the issuer, of the short vector corresponding to his syndrome and of randomness used in the encryptions of the syndrome. This ZKAoK protocol is repeated $\kappa = \omega(\log \lambda)$ times to achieve negligible soundness error and made non-interactive via Fiat-Shamir transform [22]. The signature then consists of the $\mathsf{NIZKAoK}$ Π_{gs} and the two ciphertexts of the syndrome. Note that the ZK argument together with double encryption enables CCA-security of the underlying encryption scheme, which is known as the Naor-Yung transformation [51]. This enables full anonymity of our group signature scheme.

When one needs to know the validity of a signature, one simply verifies Π_{gs}. In case of dispute, the opener can decrypt the syndrome using his opening key. To prevent corrupted opening, the opener is required to generate a $\mathsf{NIZKAoK}$ of correct opening Π_{open}. Only when Π_{open} is a valid proof, will the judger accept the opening result. Details of the scheme are described below.

$\mathsf{GKg}(\lambda)$**:** Given the security parameter λ, the trusted party proceeds as follows.

- Choose parameter $n = \mathcal{O}(\lambda)$ being a power of 2, and modulus $q = \widetilde{\mathcal{O}}(n^4)$, where $q = 3^k$ for some positive integer k. Let $R = \mathbb{Z}[X]/(X^n + 1)$ and $R_q = R/qR$.
  Also, let $\ell = \lfloor \log \frac{q-1}{2} \rfloor + 1$, $m \geq 2\lceil \log q \rceil + 2$, and $\overline{m} = m + k$.
- Choose integer d and sequence $c_0, \ldots, c_d$ as described in Section 2.3.
- Choose integer bounds $\beta = \widetilde{\mathcal{O}}(n)$, $B = \widetilde{\mathcal{O}}(n^{5/4})$, and let χ be a B-bounded distribution over R.
- Let $\mathcal{H}_{\mathsf{FS}} : \{0,1\}^* \to \{1,2,3\}^\kappa$, where $\kappa = \omega(\log \lambda)$, be a collision-resistant hash function, to be modelled as a random oracle in the Fiat-Shamir transformations [22].
- Let COM be the statistically hiding and computationally binding commitment scheme from [30], to be used in our zero-knowledge argument systems.
- Draw a uniformly random matrix $\mathbf{B} \in R_q^{1 \times m}$.
- Generate verification key

$$\mathbf{A}, \mathbf{F}_0 \in R_q^{1 \times \overline{m}}; \quad \mathbf{A}_{[0]}, \ldots, \mathbf{A}_{[d]} \in R_q^{1 \times k}; \quad \mathbf{F}, \mathbf{F}_1 \in R_q^{1 \times \ell}; \quad u \in R_q$$

  and signing key $\mathbf{R} \in R_q^{m \times k}$ for the Ducas-Micciancio signature scheme, as described in Section 2.3.

– Initialize the Naor-Yung double-encryption mechanism [51] with an extended version of the LPR encryption scheme [47] that allows to encrypt $\{-1,0,1\}$ ring vectors of length ℓ. Specifically, sample $s_1, s_2 \hookleftarrow \chi$, $\mathbf{e}_1, \mathbf{e}_2 \hookleftarrow \chi^\ell$, $\mathbf{a} \xleftarrow{\$} R_q^\ell$, and compute

$$\mathbf{b}_1 = \mathbf{a} \cdot s_1 + \mathbf{e}_1 \in R_q^\ell; \quad \mathbf{b}_2 = \mathbf{a} \cdot s_2 + \mathbf{e}_2 \in R_q^\ell.$$

Set the public parameter pp, the group public key gpk, the issue key ik and the opening key ok as follows:

$$\mathsf{pp} = \{n, q, k, R, R_q, \ell, m, \overline{m}, \chi, d, c_0, \ldots, c_d, B, \beta, \kappa, \mathcal{H}_{\mathsf{FS}}, \mathsf{COM}, \mathbf{B}\},$$

$$\mathsf{gpk} = \{\mathsf{pp}, \mathbf{A}, \{\mathbf{A}_{[j]}\}_{j=0}^d, \mathbf{F}, \mathbf{F}_0, \mathbf{F}_1, u, \mathbf{a}, \mathbf{b}_1, \mathbf{b}_2\},$$
$$\mathsf{ik} = \mathbf{R}, \quad \mathsf{ok} = (s_1, \mathbf{e}_1).$$

The trusted party then makes gpk public and sends ik to the issuer and ok to the opener.
Assume that after receiving ik from the trusted party, the issuer initializes his internal state $S = 0$ and the registration table $\mathbf{reg}$.

$\mathsf{UKg}(\mathsf{gpk})$**:** The user samples $\mathbf{x} \in R^m$, whose coefficients are uniformly random in the set $\{-1,0,1\}$. Then he computes $p = \mathbf{B} \cdot \mathbf{x} \in R_q$. Set $\mathsf{upk} = p$ and $\mathsf{usk} = \mathbf{x}$.

$\langle \mathsf{Join}, \mathsf{Iss} \rangle$**:** When receiving the joining request from a user with public key $\mathsf{upk} = p$, the issuer verifies that upk was not previously used by a registered user, and aborts if this is not the case. Otherwise, he proceeds as follows.

– Set the tag $t = (t_0, t_1 \ldots, t_{c_d-1})^\top \in \mathcal{T}_d$, where $S = \sum_{j=0}^{c_d-1} 2^j \cdot t_j$, and compute $\mathbf{A}_t = [\mathbf{A} | \mathbf{A}_{[0]} + \sum_{i=1}^d t_{[i]} \mathbf{A}_{[i]}] \in R_q^{1 \times (\overline{m}+k)}$.

– Using the signing key $\mathbf{R}$, generate a Ducas-Micciancio signature $(t, \mathbf{r}, \mathbf{v})$ on message $\mathsf{rdec}(p) \in R^\ell$ - whose coefficients are in $\{-1,0,1\}$. As described in Section 2.3, one has $\mathbf{r} \in R^{\overline{m}}$, $\mathbf{v} \in R^{\overline{m}+k}$ and

$$\begin{cases} \mathbf{A}_t \cdot \mathbf{v} = \mathbf{F} \cdot \mathsf{rdec}(\mathbf{F}_0 \cdot \mathbf{r} + \mathbf{F}_1 \cdot \mathsf{rdec}(p)) + u, \\ \|\mathbf{r}\|_\infty \leq \beta, \quad \|\mathbf{v}\|_\infty \leq \beta. \end{cases} \tag{14}$$

The issuer then sends the triple $(t, \mathbf{r}, \mathbf{v})$ to the user. The latter sets his group signing key as $\mathsf{gsk} = (t, \mathbf{r}, \mathbf{v}, \mathbf{x})$ while the former stores $\mathbf{reg}[S] = p$ and updates S to $S + 1$.

$\mathsf{Sign}(\mathsf{gpk}, \mathsf{gsk}_i, M)$**:** To sign a message $M \in \{0,1\}^*$ using $\mathsf{gsk} = (t, \mathbf{r}, \mathbf{v}, \mathbf{x})$, the group member who has public key $p \in R_q$ proceeds as follows.

– Encrypt the ring vector $\mathsf{rdec}(p) \in R_q^\ell$ with coefficients in $\{-1,0,1\}$ twice. Namely, for each $i \in \{1,2\}$, sample $g_i \hookleftarrow \chi$, $\mathbf{e}_{i,1} \hookleftarrow \chi^\ell$, and $\mathbf{e}_{i,2} \hookleftarrow \chi^\ell$ and compute

$$\begin{aligned} \mathbf{c}_i &= (\mathbf{c}_{i,1}, \mathbf{c}_{i,2}) \\ &= \big(\mathbf{a} \cdot g_i + \mathbf{e}_{i,1}, \quad \mathbf{b}_i \cdot g_i + \mathbf{e}_{i,2} + \lfloor q/4 \rfloor \cdot \mathsf{rdec}(p)\big) \in R_q^\ell \times R_q^\ell. \end{aligned}$$

– Generate a NIZKAoK Π_{gs} to demonstrate the possession of a valid tuple

$$\zeta = (t, \mathbf{r}, \mathbf{v}, \mathbf{x}, p, g_1, g_2, \mathbf{e}_{1,1}, \mathbf{e}_{2,1}, \mathbf{e}_{1,2}, \mathbf{e}_{2,2}) \tag{15}$$

such that

(i) The conditions from (14) hold.

(ii) $\mathbf{c}_1$ and $\mathbf{c}_2$ are both correct encryptions of $\mathsf{rdec}(p)$ with B-bounded randomness $g_1, \mathbf{e}_{1,1}, \mathbf{e}_{1,2}$ and $g_2, \mathbf{e}_{2,1}, \mathbf{e}_{2,2}$, respectively.

(iii) $\|\mathbf{x}\|_\infty \leq 1$ and $\mathbf{B} \cdot \mathbf{x} = p$.

This is done by running the argument system described in Section 4.2. The protocol is an extension of the one for the Ducas-Micciancio signature from Section 3.2, in which the prover additionally proves statements (ii) and (iii). The protocol is repeated $\kappa = \omega(\log \lambda)$ times to achieve negligible soundness error and made non-interactive via Fiat-Shamir heuristic [22] as a triple $\Pi_{\mathsf{gs}} = (\{\mathrm{CMT}_i\}_{i=1}^{\kappa}, \mathrm{CH}, \{\mathrm{RSP}_i\}_{i=1}^{\kappa})$ where $\mathrm{CH} = \mathcal{H}_{\mathsf{FS}}(M, \{\mathrm{CMT}_i\}_{i=1}^{\kappa}, \xi)$ with

$$\xi = (\mathbf{A}, \mathbf{A}_{[0]}, \ldots, \mathbf{A}_{[d]}, \mathbf{F}, \mathbf{F}_0, \mathbf{F}_1, u, \mathbf{B}, \mathbf{a}, \mathbf{b}_1, \mathbf{b}_2, \mathbf{c}_1, \mathbf{c}_2) \tag{16}$$

– Output the group signature $\Pi = (\Pi_{\mathsf{gs}}, \mathbf{c}_1, \mathbf{c}_2)$.

$\mathsf{Verify}(\mathsf{gpk}, M, \Sigma)$**:** Given the inputs, this algorithm proceeds as follows.

1. Parse Σ as $\Sigma = \big(\{\mathrm{CMT}_i\}_{i=1}^{\kappa}, (Ch_1, \ldots, Ch_\kappa), \{\mathrm{RSP}\}_{i=1}^{\kappa}, \mathbf{c}_1, \mathbf{c}_2\big)$. If $(Ch_1, \ldots, Ch_\kappa) \neq \mathcal{H}_{\mathsf{FS}}\big(M, \{\mathrm{CMT}_i\}_{i=1}^{\kappa}, \xi\big)$, then return 0, where ξ is as in (16).
2. For each $i \in [\kappa]$, run the verification phase of the protocol in Section 4.2 to check the validity of RSP_i with respect to CMT_i and Ch_i. If any of the conditions does not hold, then return 0.
3. Return 1.

$\mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathbf{reg}, M, \Sigma)$**:** Let $\mathsf{ok} = (s_1, \mathbf{e}_1)$ and $\Sigma = (\Pi_{\mathsf{gs}}, \mathbf{c}_1, \mathbf{c}_2)$. This algorithm then does the following.

1. Use s_1 to decrypt $\mathbf{c}_1 = (\mathbf{c}_{1,1}, \mathbf{c}_{1,2})$ as follows.
   (a) It computes
   $$\mathbf{p}'' = \frac{\mathbf{c}_{1,2} - \mathbf{c}_{1,1} \cdot s_1}{\lfloor q/4 \rfloor}.$$
   (b) For each coeffcient of $\mathbf{p}''$,
   - if it is closer to 0 than to -1 and 1, then round it to 0;
   - if it is closer to -1 than to 0 and 1, then round it to -1;
   - if it is closer to 1 than to 0 and -1, then round it to 1.

   (c) Denote the rounded $\mathbf{p}''$ as $\mathbf{p}' \in R_q^\ell$ with coefficients in $\{-1, 0, 1\}$.
   (d) Let $p' \in R_q$ such that $\tau(p') = \mathbf{H} \cdot \tau(\mathbf{p}')$. Recall that $\mathbf{H} \in \mathbb{Z}_q^{n \times n\ell}$ is the decomposition matrix for elements of R_q (see Section 2.2).
2. If **reg** does not include an entry p', then return $(\perp, \perp)$.

3. Otherwise, generate a $\mathsf{NIZKAoK}$ Π_{open} to demonstrate the possession of a tuple $(s_1, \mathbf{e}_1, \mathbf{y}) \in R_q \times R_q^\ell \times R_q^\ell$

$$\begin{cases} \|s_1\|_\infty \leq B;\ \|\mathbf{e}_1\|_\infty \leq B;\ \|\mathbf{y}\|_\infty \leq \lceil q/10 \rceil; \\ \mathbf{a} \cdot s_1 + \mathbf{e}_1 = \mathbf{b}_1; \\ \mathbf{c}_{1,2} - \mathbf{c}_{1,1} \cdot s_1 = \mathbf{y} + \lfloor q/4 \rfloor \cdot \mathsf{rdec}(p'). \end{cases} \tag{17}$$

We remark that conditions in (17) involve only linear secret objects with bounded norms, and can be handled using the Stern-like techniques from Sections 3.2 and 4.2. As a result, we can obtain a statistical ZKAoK for the considered statement. The protocol is repeated $\kappa = \omega(\log \lambda)$ times to achieve negligible soundness error and made non-interactive via the Fiat-Shamir heuristic as a triple $\Pi_{\mathsf{Open}} = (\{\mathrm{CMT}_i\}_{i=1}^{\kappa}, \mathrm{CH}, \{\mathrm{RSP}\}_{i=1}^{\kappa})$, where

$$\mathrm{CH} = \mathcal{H}_{\mathsf{FS}}\big(\{\mathrm{CMT}_i\}_{i=1}^{\kappa}, \mathbf{a}, \mathbf{b}_1, M, \Sigma, p'\big) \in \{1,2,3\}^{\kappa}. \tag{18}$$

4. Output $(p', \Pi_{\mathsf{Open}})$.

$\mathsf{Judge}(\mathsf{gpk}, M, \Sigma, p', \Pi_{\mathsf{open}})$**:** If Verify algorithm outputs 0, then this algorithm returns 0. Otherwise, this algorithm then verifies the argument Π_{Open} w.r.t. common input $(\mathbf{a}, \mathbf{b}_1, M, \Sigma, p')$, in a similar manner as in algorithm Verify. If Π_{open} does not verify, then return 0; otherwise, return 1.

4.4 Analysis of the Scheme

Efficiency. We first analyze the efficiency of the scheme described in Section 4.3, with respect to security parameter λ.

- The public key gpk has bit-size $\mathcal{O}(\lambda \cdot \log^2 \lambda) = \widetilde{\mathcal{O}}(\lambda)$.
- The signing key gsk_i has bit-size $\mathcal{O}(\lambda \cdot \log^2 \lambda) = \widetilde{\mathcal{O}}(\lambda)$.
- The size of a signature Σ is dominated by that of the Stern-like $\mathsf{NIZKAoK}$ Π_{gs}, which is $\mathcal{O}(\widetilde{L} \cdot \log q) \cdot \omega(\log \lambda)$, where $\widetilde{L}$ denotes the bit-size of a vector $\widetilde{\mathbf{w}} \in \widetilde{\mathsf{VALID}}$ as described in Section 4.2. Recall $\mathcal{O}(\widetilde{L} \cdot \log q) = \mathcal{O}(\lambda \cdot \log^4 \lambda)$. As a result, Σ has bit-size $\mathcal{O}(\lambda \cdot \log^4 \lambda) \cdot \omega(\log \lambda) = \widetilde{\mathcal{O}}(\lambda)$.
- The Stern-like $\mathsf{NIZKAoK}$ Π_{open} has bit-size $\mathcal{O}(\lambda \cdot \log^3 \lambda) \cdot \omega(\log \lambda) = \widetilde{\mathcal{O}}(\lambda)$.

Correctness. The correctness of the above group signature scheme relies on the following facts: (i) the underlying argument systems to generate Π_{gs} and Π_{open} are perfectly complete; (ii) the underlying encryption scheme, which is an extended version of LPR encryption scheme [47] is correct.

Specifically, for an honest user, when he signs a message on behalf of the group, he is able to demonstrate the possession of a valid tuple ζ of the form (15). With probability 1, Π_{gs} is accepted by the Verify algorithm, which is implied by

the perfect completeness of the argument system to generate Π_{gs}. As for the correctness of the Open algorithm, note that

$$\begin{aligned}\mathbf{c}_{1,1} - \mathbf{c}_{1,2} \cdot s_1 &= \mathbf{b}_1 \cdot g_1 + \mathbf{e}_{1,2} + \lfloor q/4 \rfloor \cdot \mathsf{rdec}(p) - (\mathbf{a} \cdot g_1 + \mathbf{e}_{1,1}) \cdot s_1 \\ &= (\mathbf{a} \cdot s_1 + \mathbf{e}_1) \cdot g_1 + \mathbf{e}_{1,2} + \lfloor q/4 \rfloor \cdot \mathsf{rdec}(p) - (\mathbf{a} \cdot g_1 + \mathbf{e}_{1,1}) \cdot s_1 \\ &= \mathbf{e}_1 \cdot g_1 + \mathbf{e}_{1,2} - \mathbf{e}_{1,1} \cdot s_1 + \lfloor q/4 \rfloor \cdot \mathsf{rdec}(p)\end{aligned}$$

where $\|\mathbf{e}_1\|_\infty \leq B$, $\|s_1\|_\infty \leq B$, $\|g_1\|_\infty \leq B$, $\|\mathbf{e}_{1,1}\|_\infty \leq B$, $\|\mathbf{e}_{1,2}\|_\infty \leq B$. Recall $B = \widetilde{\mathcal{O}}(n^{5/4})$ and $q = \widetilde{\mathcal{O}}(n^4)$. Hence we have:

$$\|\mathbf{e}_1 \cdot g_1 + \mathbf{e}_{1,2} - \mathbf{e}_{1,1} \cdot s_1\|_\infty \leq 2n \cdot B^2 + B = \widetilde{\mathcal{O}}(n^{3.5}) \leq \lceil \frac{q}{10} \rceil = \widetilde{\mathcal{O}}(n^4).$$

With probability 1, the rounding procedure described in the Open algorithm recovers $\mathsf{rdec}(p)$ and hence outputs p, which is the actual signer. Thus the opener is able to identify the signer of a signature and hence correctness of the Open algorithm holds.

As the opener correctly recovers $\mathsf{rdec}(p)$ and p, it possesses a valid tuple $(s_1, \mathbf{e}_1, \mathbf{y})$ satisfying conditions in (17). It then follows from the perfect completeness of the argument system to generate Π_{open}, the judge will accept the opening result outputted by the opener and hence correctness of the Judge algorithm holds.

Security. In Theorem 4, we prove that our scheme satisfies the security requirements of the Bellare, Shi and Zhang's model [5]. For the proof of non-framebility, we will use the following simple lemma.

Lemma 2. *Let $\mathbf{B} \in R_q^{1 \times m}$, where $m \geq 2\lceil \log q \rceil + 2$. If $\mathbf{x}$ is a uniformly random element of R^m such that $\|\mathbf{x}\|_\infty \leq 1$, then with probability at least $1 - 2^{-n}$, there exists another $\mathbf{x}' \in R^m$ such that $\|\mathbf{x}'\|_\infty \leq 1$ and $\mathbf{B} \cdot \mathbf{x} = \mathbf{B} \cdot \mathbf{x}' \in R_q$.*

Proof. Note that there are in total 3^{nm} elements $\mathbf{x} \in R^m$ such that $\|\mathbf{x}\|_\infty \leq 1$. Among them, there exist at most $q^n - 1$ elements that do not have $\mathbf{x}'$ such that $\mathbf{B} \cdot \mathbf{x} = \mathbf{B} \cdot \mathbf{x}'$. Hence, the probability that a uniformly random $\mathbf{x}$ has a corresponding $\mathbf{x}'$ for which $\mathbf{B} \cdot \mathbf{x} = \mathbf{B} \cdot \mathbf{x}'$ is at least

$$\frac{3^{nm} - q^n + 1}{3^{nm}} = 1 - \frac{q^n - 1}{3^{nm}} > 1 - \frac{q^n}{2^n q^n} = 1 - 2^{-n}.$$

□

Theorem 4. *Assume that the Stern-like argument systems used in Section 4.3 are simulation-sound. Then, in the random oracle model, the given group signature scheme satisfies full anonymity, traceability and non-frameability under the* RLWE *and* RSIS *assumptions.*

In the random oracle model, the proof of Theorem 4 relies on the following facts:

1. The Stern-like zero-knowledge argument systems being used are simulation-sound;

2. The underlying encryption scheme, which is an extended version of the LPR encryption scheme [47], via the Naor-Yung transformation [51], is IND-CCA secure;
3. The variant of Ducas-Micciancio signature scheme described in Section 2.3 with at most polynomial number of signature queries is existentially unforgeable against adaptive chosen message attacks [20,21];
4. For a properly generated user key pair $(\mathbf{x}, p)$, it is infeasible to find $\mathbf{x}' \in R_q^m$ such that $\|\mathbf{x}'\|_\infty \leq 1$, $\mathbf{x}' \neq \mathbf{x}$ and $\mathbf{B} \cdot \mathbf{x}' = p$.

The proof of Theorem 4 is established by Lemmas 3-5 given below.

Lemma 3. *Assume that the* $\mathsf{RLWE}_{n,\ell,q,\chi}$ *problem is hard. Then the given group signature scheme is fully anonymous in the random oracle model.*

The detailed proof of Lemma 3 is given in Appendix C.

Lemma 4. *Assume that the* $\mathsf{RSIS}^{\infty}_{n,\overline{m},q,\widetilde{\mathcal{O}}(n^2)}$ *problem is hard. Then the given group signature scheme is traceable in the random oracle model.*

Proof. We prove traceability by contradiction. Suppose that $\mathcal{A}$ succeeds with non-negligible advantage ϵ. Then we build a PPT algorithm $\mathcal{B}$ that, with non-negligible probability, breaks the unforgeability of the Ducas-Micciancio signature scheme from Section 2.3, which is based on the hardness of the $\mathsf{RSIS}^{\infty}_{n,\overline{m},q,\widetilde{\mathcal{O}}(n^2)}$ problem. It then follows that our construction is traceable.

When given the verification key of the Ducas-Micciancio signature scheme, the simulator $\mathcal{B}$ runs the experiment $\mathbf{Exp}^{\mathsf{trace}}_{\mathsf{GS},\mathcal{A}}(\lambda)$ faithfully. $\mathcal{B}$ can answer all oracle queries made by $\mathcal{A}$ except when $\mathcal{A}$ queries the *send to issuer* SndToI oracle or *add user* AddU oracle. However, $\mathcal{B}$ can resort to his oracle queries of the signature scheme. In these two cases, $\mathcal{B}$ enrolls the corresponding user to the group. When $\mathcal{A}$ halts, it outputs $(M^*, \Pi^*_{\mathsf{gs}}, \mathbf{c}_1^*, \mathbf{c}_2^*)$. With non-negligible probability ϵ, $\mathcal{A}$ wins the experiment. Parse $\Pi^*_{\mathsf{gs}} = (\{\mathrm{CMT}_i^*\}_{i=1}^{\kappa}, \mathrm{CH}^*, \{\mathrm{RSP}_i^*\}_{i=1}^{\kappa})$. Let

$$\xi^* = (\mathbf{A}, \mathbf{A}_{[0]}, \ldots, \mathbf{A}_{[d]}, \mathbf{F}, \mathbf{F}_0, \mathbf{F}_1, u, \mathbf{B}, \mathbf{a}, \mathbf{b}_1, \mathbf{b}_2, \mathbf{c}_1^*, \mathbf{c}_2^*).$$

Then $\mathrm{CH}^* = \mathcal{H}_{\mathsf{FS}}\big(M^*, \{\mathrm{CMT}_i^*\}_{i=1}^{\kappa}, \xi^*\big)$ and RSP_i^* is a valid response w.r.t. CMT_i^* and CH_i^* for $i \in [\kappa]$ by the fact that $\mathcal{A}$ wins and hence $(\Pi^*_{\mathsf{gs}}, \mathbf{c}_1^*, \mathbf{c}_2^*)$ is a valid signature on message M^*.

We claim that $\mathcal{A}$ had queried $\big(M^*, \{\mathrm{CMT}_i^*\}_{i=1}^{\kappa}, \xi^*\big)$ to the hash oracle $\mathcal{H}_{\mathsf{FS}}$ with overwhelming probability. Otherwise, the probability of guessing correctly the value of $\mathcal{H}_{\mathsf{FS}}\big(M^*, \{\mathrm{CMT}_i^*\}_{i=1}^{\kappa}, \xi^*\big)$ is at most $3^{-\kappa}$, which is negligible. Therefore, with probability $\epsilon' = \epsilon - 3^{-\kappa}$, $\mathcal{A}$ had queried the hash oracle $\mathcal{H}_{\mathsf{FS}}$. Denote by $\theta^* \in \{1, 2, \ldots, Q_H\}$ the index of this specific query, where Q_H is the total number of hash queries made by $\mathcal{A}$.

Algorithm $\mathcal{B}$ then runs at most $32 \cdot Q_H/\epsilon'$ executions of $\mathcal{A}$. For each new run, it is exactly the same as the original run until the point of θ^*-th query to the hash oracle $\mathcal{H}_{\mathsf{FS}}$. From this point on, $\mathcal{B}$ replies $\mathcal{A}$'s hash queries with uniformly random and independent values for each new run. This guarantees that the input of θ^*-th

query $\mathcal{A}$ made to $\mathcal{H}_{\mathsf{FS}}$ is the tuple $\left(M^*, \{\mathrm{CMT}_i^*\}_{i=1}^{\kappa}, \xi^*\right)$ for each new run while the output of this hash query is uniformly random and independent for each new run. To this point, by the forking lemma of Brickell et al. [13], with probability $\geq 1/2$, $\mathcal{B}$ obtains 3-fork involving the same tuple $\left(M^*, \{\mathrm{CMT}_i^*\}_{i=1}^{\kappa}, \xi^*\right)$ with pairwise distinct hash values $\mathrm{CH}_{\theta^*}^{(1)}, \mathrm{CH}_{\theta^*}^{(2)}, \mathrm{CH}_{\theta^*}^{(3)} \in \{1,2,3\}^{\kappa}$ and corresponding valid responses $\mathrm{RSP}_{\theta^*}^{(1)}$, $\mathrm{RSP}_{\theta^*}^{(2)}$, $\mathrm{RSP}_{\theta^*}^{(3)}$. A simple calculation shows that with probability $1 - (\frac{7}{9})^{\kappa}$, we have $\{\mathsf{CH}_{\theta^*,j}^{(1)}, \mathsf{CH}_{\theta^*,j}^{(2)}, \mathsf{CH}_{\theta^*,j}^{(3)}\} = \{1,2,3\}$ for some $j \in \{1,2,\ldots,\kappa\}$.

Therefore, $\mathrm{RSP}_{\theta^*,j}^{(1)}$, $\mathrm{RSP}_{\theta^*,j}^{(2)}$, $\mathrm{RSP}_{\theta^*,j}^{(3)}$ are 3 valid responses for all the challenges $1,2,3$ w.r.t. the same commitment CMT_j^*. Since COM is computationally binding, $\mathcal{B}$ is able to extract the witness

$$t^* \in \mathcal{T}_d; \mathbf{r}^* \in R_q^{\overline{m}}; \mathbf{v}^* \in R_q^{\overline{m}+k}; \mathbf{p}^* \in R_q^{\ell},$$

such that $\|\mathbf{r}^*\|_{\infty} \leq \beta$, $\|\mathbf{v}^*\|_{\infty} \leq \beta$, $\|\mathbf{p}^*\|_{\infty} \leq 1$ and

$$\mathbf{A}_{t^*} \cdot \mathbf{v}^* = \mathbf{F} \cdot \mathsf{rdec}(\mathbf{F}_0 \cdot \mathbf{r}^* + \mathbf{F}_1 \cdot \mathbf{p}^*) + u,$$

and $\mathbf{c}_1^*, \mathbf{c}_2^*$ are correct encryptions of $\mathbf{p}^*$.

Since $\mathcal{A}$ wins the game, either we have (i) the Open algorithm outputs $(\perp, \perp)$ or (ii) the Open algorithm output $(p', \Pi_{\mathsf{open}}^*)$ with $p' \neq \perp$ but the Judge algorithm rejects the opening result.

Case (i) implies that, if $\mathbf{c}_1^*$ is decrypted to $\mathbf{p}'$ and $p' \in R_q$ such that $\tau(p') = \mathbf{H} \cdot \tau(\mathbf{p}') \in \mathbb{Z}_q^n$, then p' is not in the registration table. From the extraction, we know that $\mathbf{c}_1^*$ will be decrypted to $\mathbf{p}^*$ by the correctness of our encryption scheme. Therefore, the *intermediate* opening result $\mathbf{p}'$ is equal to $\mathbf{p}^*$. On the other hand, the fact that p' is not in the registration table implies that $\mathcal{B}$ did not enroll p' to the group, that is, $\mathcal{B}$ did not query p' to his challenger when $\mathcal{A}$ made the AddU oracle queries or SndToI oracle queries. To summarize, $\mathcal{B}$ did not query signature on p' and $\mathcal{B}$ extracts a signature $(t^*, \mathbf{r}^*, \mathbf{v}^*)$ on $\mathbf{p}^* = \mathbf{p}'$ such that $\tau(p') = \mathbf{H} \cdot \tau(\mathbf{p}')$. Therefore $(\mathbf{p}^*, t^*, \mathbf{r}^*, \mathbf{v}^*)$ is a valid forgery of the Ducas-Micciancio signature scheme.

Case (ii) implies that, if $\mathbf{c}_1^*$ is decrypted to $\mathbf{p}'$ and $p' \in R_q$ such that $\tau(p') = \mathbf{H} \cdot \tau(\mathbf{p}') \in \mathbb{Z}_q^n$, then p' is in the registration table and Π_{open}^* generated by $\mathcal{B}$ is not accepted by the Judge algorithm. From the extraction, we know that $\mathbf{c}_1^*$ will be decrypted to $\mathbf{p}^*$ by the correctness of our encryption scheme. Therefore, the *intermediate* opening result $\mathbf{p}'$ is equal to $\mathbf{p}^*$. On the other hand, we claim that $\mathsf{rdec}(p') \neq \mathbf{p}' = \mathbf{p}^*$. Otherwise, $\mathsf{rdec}(p') = \mathbf{p}' = \mathbf{p}^*$, then $\mathcal{B}$ possesses valid witness to generate the proof Π_{open}^*. By the perfect completeness of the underlying argument system generating Π_{open}^*, it will be accepted by the judge algorithm with probability 1. This is a contradiction and hence we obtain $\mathsf{rdec}(p') \neq \mathbf{p}' = \mathbf{p}^*$. Recall that in the $\langle \mathsf{Join}, \mathsf{Iss} \rangle$ algorithm, the issuer only generates signature on $\mathsf{rdec}(p')$. So $\mathcal{B}$ only queries the signature on $\mathsf{rdec}(p')$ and hence $(\mathbf{p}^*, t^*, \mathbf{r}^*, \mathbf{v}^*)$ is a valid forgery of the Ducas-Micciancio signature scheme.

Therefore, with probability at least $\frac{1}{2} \cdot (\epsilon - 3^{-\kappa})(1 - (\frac{7}{9})^{\kappa})$, which is non-negligible, $\mathcal{B}$ breaks the unforgeability of the Ducas-Micciancio signature scheme. This concludes the proof.

□

Lemma 5. *Assume that the* $\mathsf{RSIS}^{\infty}_{n,m,q,1}$ *problem is hard. Then the given group signature scheme is non-frameable in the random oracle model.*

Proof. We prove non-frameability by contradiction. Suppose that $\mathcal{A}$ succeeds with non-negligible advantage ϵ. Then we build a PPT algorithm $\mathcal{B}$ that solves a $\mathsf{RSIS}_{n,m,q,1}$ instance $\mathbf{B} \in R_q^{1 \times m}$ with non-negligible probability.

After $\mathcal{B}$ is given a RSIS instance matrix $\mathbf{B}$, it runs the experiment $\mathbf{Exp}^{\mathsf{nf}}_{\mathsf{GS},\mathcal{A}}$ faithfully. $\mathcal{B}$ can answer all the oracle queries made by $\mathcal{A}$ since $\mathcal{B}$ knows all the keys. When $\mathcal{A}$ halts, it outputs $(M^*, \Pi^*_{\mathsf{gs}}, \mathbf{c}_1^*, \mathbf{c}_2^*, p^*, \Pi^*_{\mathsf{open}})$. With non-negligible probability ϵ, $\mathcal{A}$ wins the experiment.

The fact that $\mathcal{A}$ wins the game implies $(M^*, \Pi^*_{\mathsf{gs}}, \mathbf{c}_1^*, \mathbf{c}_2^*)$ is a valid message-signature pair that was not queried before. By the same extraction technique as in Lemma 4, we can extract witness $\mathbf{x}' \in R_q^m$ and $\mathbf{p}' \in R_q^{\ell}$ such that $\mathbf{x}', \mathbf{p}'$ have coefficients in $\{-1, 0, 1\}$, $\mathbf{B} \cdot \mathbf{x}' = p'$ with $\tau(p') = \mathbf{H} \cdot \tau(\mathbf{p}')$ and $\mathbf{c}_1^*, \mathbf{c}_2^*$ are correct encryptions of $\mathbf{p}'$. By the correctness of the encryption scheme being used, $\mathbf{c}_1^*$ will be decrypted to $\mathbf{p}'$.

The fact that $\mathcal{A}$ wins the game also implies $(p^*, \Pi^*_{\mathsf{open}})$ is accepted by the Judge algorithm. It follows from the soundness of the argument system used to generate Π^*_{open} that $\mathbf{c}_1^*$ will be decrypted to $\mathsf{rdec}(p^*)$. Therefore, we have $\mathbf{p}' = \mathsf{rdec}(p^*)$ and hence $p' = p^*$. Note that $\mathcal{A}$ wins the game also implies that p^* is an honest user with $\mathsf{gsk} \neq \perp$ and $\mathcal{A}$ did not query the user secret key $\mathbf{x}^*$ that corresponds to p^*. Thus we obtain: $\mathbf{B} \cdot \mathbf{x}' = p' = p^* = \mathbf{B} \cdot \mathbf{x}^*$, where $\mathbf{x}^*$ has coefficients in $\{-1, 0, 1\}$. By Lemma 2, $\mathbf{x}' \neq \mathbf{x}^*$ with probability at least $1/2$. In the case they are not equal, we obtain a non-zero vector $\mathbf{y} = \mathbf{x}' - \mathbf{x}^*$ such that $\mathbf{B} \cdot \mathbf{y} = 0$ and $\|\mathbf{y}\|_{\infty} = 1$.

Therefore, with probability at least $\frac{1}{2} \cdot (\epsilon - 3^{-\kappa})(1 - (\frac{7}{9})^{\kappa}) \cdot \frac{1}{2}$, which is non-negligible, $\mathcal{B}$ solves a $\mathsf{RSIS}_{n,m,q,1}$ instance $\mathbf{B} \in R_q^{1 \times m}$. This concludes the proof.

□

Acknowledgements. The authors would like to thank Benoît Libert and the anonymous reviewers of PKC 2018 for helpful comments and discussions. The research is supported by Singapore Ministry of Education under Research Grant MOE2016-T2-2-014(S).

References

1. G. Ateniese, J. Camenisch, S. Hohenberger, and B. de Medeiros. Practical group signatures without random oracles. *IACR Cryptology ePrint Archive*, 2005:385, 2005.

2. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *CRYPTO 2000*, volume 1880 of *LNCS*, pages 255–270. Springer, 2000.
3. M. Bellare and G. Fuchsbauer. Policy-based signatures. In *PKC 2014*, volume 8383 of *LNCS*, pages 520–537. Springer, 2014.
4. M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 614–629. Springer, 2003.
5. M. Bellare, H. Shi, and C. Zhang. Foundations of group signatures: The case of dynamic groups. In *CT-RSA 2005*, volume 3376 of *LNCS*, pages 136–153. Springer, 2005.
6. 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 2014*, volume 8873 of *LNCS*, pages 551–572. Springer, 2014.
7. F. Böhl, D. Hofheinz, T. Jager, J. Koch, and C. Striecks. Confined guessing: New signatures from standard assumptions. *J. Cryptology*, 28(1):176–208, 2015.
8. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, 2004.
9. D. Boneh and H. Shacham. Group signatures with verifier-local revocation. In *ACM CCS 2004*, pages 168–177. ACM, 2004.
10. J. Bootle, A. Cerulli, P. Chaidos, E. Ghadafi, and J. Groth. Foundations of fully dynamic group signatures. In *ACNS 2016*, volume 9696 of *LNCS*, pages 117–136, 2016.
11. X. Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In *PKC 2010*, volume 6056 of *LNCS*, pages 499–517. Springer, 2010.
12. X. Boyen and B. Waters. Full-domain subgroup hiding and constant-size group signatures. In *PKC 2007*, volume 4450 of *LNCS*, pages 1–15. Springer, 2007.
13. E. F. Brickell, D. Pointcheval, S. Vaudenay, and M. Yung. Design validations for discrete logarithm based signature schemes. In *PKC 2000*, volume 1751 of *LNCS*, pages 276–292. Springer, 2000.
14. J. Camenisch, S. Hohenberger, and A. Lysyanskaya. Compact e-cash. In *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 302–321. Springer, 2005.
15. J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118. Springer, 2001.
16. J. Camenisch, G. Neven, and M. Rückert. Fully anonymous attribute tokens from lattices. In *SCN 2012*, volume 7485 of *LNCS*, pages 57–75. Springer, 2012.
17. D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 523–552. Springer, 2010.
18. D. Chaum and E. van Heyst. Group signatures. In *EUROCRYPT 1991*, volume 547 of *LNCS*, pages 257–265. Springer, 1991.
19. S. Cheng, K. Nguyen, and H. Wang. Policy-based signature scheme from lattices. *Des. Codes Cryptography*, 81(1):43–74, 2016.
20. L. Ducas and D. Micciancio. Improved short lattice signatures in the standard model. In *CRYPTO 2014*, volume 8616 of *LNCS*, pages 335–352. Springer, 2014.
21. L. Ducas and D. Micciancio. Improved short lattice signatures in the standard model. *IACR Cryptology ePrint Archive*, 2014:495, 2014.

22. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO 1986*, volume 263 of *LNCS*, pages 186–194. Springer, 1986.
23. C. Gentry. Fully homomorphic encryption using ideal lattices. In *STOC 2009*, pages 169–178. ACM, 2009.
24. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC 2008*, pages 197–206. ACM, 2008.
25. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Predicate encryption for circuits from LWE. In *CRYPTO 2015*, volume 9216 of *LNCS*, pages 503–523. Springer, 2015.
26. S. D. Gordon, J. Katz, and V. Vaikuntanathan. A group signature scheme from lattice assumptions. In *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 395–412. Springer, 2010.
27. J. Groth. Evaluating security of voting schemes in the universal composability framework. In *ACNS 2004*, volume 3089 of *LNCS*, pages 46–60. Springer, 2004.
28. J. Groth. Fully anonymous group signatures without random oracles. In *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 164–180. Springer, 2007.
29. A. Jain, S. Krenn, K. Pietrzak, and A. Tentes. Commitments and efficient zero-knowledge proofs from learning parity with noise. In *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 663–680. Springer, 2012.
30. A. Kawachi, K. Tanaka, and K. Xagawa. Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In *ASIACRYPT 2008*, volume 5350 of *LNCS*, pages 372–389. Springer, 2008.
31. A. Kiayias and M. Yung. Secure scalable group signature with dynamic joins and separable authorities. *Int. Journal of Security and Networks*, 1(1):24–45, 2006.
32. F. Laguillaumie, A. Langlois, B. Libert, and D. Stehlé. Lattice-based group signatures with logarithmic signature size. In *ASIACRYPT 2013*, volume 8270 of *LNCS*, pages 41–61. Springer, 2013.
33. A. Langlois, S. Ling, K. Nguyen, and H. Wang. Lattice-based group signature scheme with verifier-local revocation. In *PKC 2014*, volume 8383 of *LNCS*, pages 345–361. Springer, 2014. Corrected full version: `http://eprint.iacr.org/2014/033`.
34. A. Langlois and D. Stehlé. Worst-case to average-case reductions for module lattices. *Des. Codes Cryptography*, 75(3):565–599, 2015.
35. B. Libert, S. Ling, F. Mouhartem, K. Nguyen, and H. Wang. Signature schemes with efficient protocols and dynamic group signatures from lattice assumptions. In *ASIACRYPT 2016*, volume 10032 of *LNCS*, pages 373–403. Springer, 2016.
36. B. Libert, S. Ling, F. Mouhartem, K. Nguyen, and H. Wang. Zero-knowledge arguments for matrix-vector relations and lattice-based group encryption. In *ASIACRYPT 2016*, volume 10032 of *LNCS*, pages 101–131. Springer, 2016.
37. B. Libert, S. Ling, K. Nguyen, and H. Wang. Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In *EUROCRYPT 2016*, volume 9666 of *LNCS*, pages 1–31. Springer, 2016.
38. B. Libert, S. Ling, K. Nguyen, and H. Wang. Zero-knowledge arguments for lattice-based prfs and applications to e-cash. In *ASIACRYPT 2017*, volume 10626 of *LNCS*, pages 304–335. Springer, 2017.
39. B. Libert, F. Mouhartem, and K. Nguyen. A lattice-based group signature scheme with message-dependent opening. In *ACNS 2016*, volume 9696 of *LNCS*, pages 137–155. Springer, 2016.

40. B. Libert, T. Peters, and M. Yung. Short group signatures via structure-preserving signatures: Standard model security from simple assumptions. In *CRYPTO 2015*, volume 9216 of *LNCS*, pages 296–316. Springer, 2015.
41. S. Ling, K. Nguyen, D. Stehlé, and H. Wang. Improved zero-knowledge proofs of knowledge for the ISIS problem, and applications. In *PKC 2013*, volume 7778 of *LNCS*, pages 107–124. Springer, 2013.
42. S. Ling, K. Nguyen, and H. Wang. Group signatures from lattices: Simpler, tighter, shorter, ring-based. In *PKC 2015*, volume 9020 of *LNCS*, pages 427–449. Springer, 2015.
43. S. Ling, K. Nguyen, H. Wang, and Y. Xu. Lattice-based group signatures: Achieving full dynamicity with ease. In *ACNS 2017*, volume 10355 of *LNCS*, pages 293–312. Springer, 2017.
44. V. Lyubashevsky and D. Micciancio. Generalized compact knapsacks are collision resistant. In *ICALP 2006*, volume 4052 of *LNCS*, pages 144–155. Springer, 2006.
45. V. Lyubashevsky, D. Micciancio, C. Peikert, and A. Rosen. SWIFFT: A modest proposal for FFT hashing. In *FSE 2008*, volume 5086 of *LNCS*, pages 54–72. Springer, 2008.
46. V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, 2010.
47. V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. *J. ACM*, 60(6):43:1–43:35, 2013.
48. V. Lyubashevsky, C. Peikert, and O. Regev. A toolkit for ring-lwe cryptography. In *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 35–54. Springer, 2013.
49. D. Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *Computational Complexity*, 16(4):365–411, 2007.
50. D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, 2012.
51. M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *ACM 1990*, pages 427–437. ACM, 1990.
52. P. Q. Nguyen, J. Zhang, and Z. Zhang. Simpler efficient group signatures from lattices. In *PKC 2015*, volume 9020 of *LNCS*, pages 401–426. Springer, 2015.
53. C. Peikert, O. Regev, and N. Stephens-Davidowitz. Pseudorandomness of ring-lwe for any ring and modulus. In *STOC 2017*, pages 461–473. ACM, 2017.
54. C. Peikert and A. Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *TCC 2006,*, volume 3876 of *LNCS*, pages 145–166. Springer, 2006.
55. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *ACM 2005*, pages 84–93. ACM, 2005.
56. Y. Sakai, K. Emura, G. Hanaoka, Y. Kawai, T. Matsuda, and K. Omote. Group signatures with message-dependent opening. In *Pairing 2012*, volume 7708 of *LNCS*, pages 270–294. Springer, 2012.
57. J. Stern. A new paradigm for public key identification. *IEEE Trans. Information Theory*, 42(6):1757–1768, 1996.
58. K. Xagawa. Improved (hierarchical) inner-product encryption from lattices. *IACR Cryptology ePrint Archive*, 2015:249, 2015.

A Dynamic Group Signatures

A (partially) dynamic group signature scheme must satisfy correctness and security requirements: full anonymity, traceability and non-frameability, as put forward by Bellare, Shi and Zhang [5].

Correctness and security requirements of the dynamic group signatures will be modelled via experiments, where the adversary has access to some oracles. We therefore recall the oracles as specified by [5].

$\mathsf{AddU}(i)$**:** This *add user* oracle adds an honest user to the group and returns upk.

$\mathsf{CruptU}(i, \mathsf{upk})$**:** This *corrupt user* oracle allows the adversary to corrupt this user i and set this user pubic key to be upk chosen by the adversary.

$\mathsf{SndToI}(i, M_{\mathsf{in}})$**:** After corrupting user i, the adversary runs this *send to issuer* oracle to interact with the honest and Iss-executing issuer while the adversary plays the role of user i. If it completes successful, the oracle returns M_{out} to the adversary and stores $\mathbf{reg}[i]$ to the registration table as the final state of the Iss algorithm.

$\mathsf{SndToU}(i, M_{\mathsf{in}})$**:** This *send to user* oracle is run, by the adversary who corrupts the issuer, to interact with an honest and Join-executing user i while the adversary plays the role of the issuer. If it completes successful, the oracle returns M_{out} to the adversary and stores gsk_i as the final state of the Join algorithm.

$\mathsf{RevealU}(i)$**:** This *reveal user* oracle reveals all the secret keys $(\mathsf{usk}_i, \mathsf{gsk}_i)$ of user i to the adversary.

$\mathsf{RReg}(i)$**:** This *read registration table* oracle allows the adversary to read $\mathbf{reg}[i]$.

$\mathsf{WReg}(i, \rho)$**:** This *write registration table* oracle allows the adversary to change $\mathbf{reg}[i]$ to arbitrarily string ρ chosen by the adversary.

$\mathsf{Sign}(i, M)$**:** This *signing* oracle allows the adversary to query a signature of the message M under the group signing key gsk_i, as long as user i is an honest user and gsk_i is defined.

$\mathsf{Ch}_b(i_0, i_1, M)$**:** Given the inputs (i_0, i_1, M) from the adversary attacking anonymity, this *challenge* oracle returns a signature Σ of the message M under the group signing key gsk_{i_b} of user i_b to the adversary and stores (M, Σ) in a set G.

$\mathsf{Open}(M, \Sigma)$**:** This *opening* oracle allows the adversary to query the opening result of the pair (M, Σ) as long as (M, Σ) was not obtained from the *challenge* oracle.

Readers are referred to Bellare, Shi, Zhang [5, Section 3] for detailed descriptions of the above oracles.

Definition 3. *For any security parameter λ and any* PPT *adversary $\mathcal{A}$, define correctness and security experiments in Figure 2.*

For correctness, traceability and non-frameability, the advantage of the adversary is defined as the probability of outputting 1 *in the corresponding experiment.*

For anonymity, the advantage is defined as the absolute difference of probability of outputting 1 *between experiment* $\mathbf{Exp}^{\mathsf{anon}-1}_{\mathsf{GS},\mathcal{A}}$ *and experiment* $\mathbf{Exp}^{\mathsf{anon}-0}_{\mathsf{FDGS},\mathcal{A}}$.

A (partially) dynamic group signature scheme is said to be correct and secure (i.e., fully anonymous, traceable and non-frameable) if the advantages of the adversary in all considered experiments are negligible in λ.

$\underline{\mathbf{Exp}^{\mathsf{corr}}_{\mathsf{GS},\mathcal{A}}(\lambda)}$

$(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GKg}(\lambda)$; $\mathrm{CU} \leftarrow \emptyset$; $\mathrm{HU} \leftarrow \emptyset$; $(i, M) \leftarrow \mathcal{A}(\mathsf{gpk} : \mathsf{AddU}, \mathsf{RReg})$

If $i \notin \mathrm{HU}$ then return 0; If $\mathsf{gsk}_i = \epsilon$ then return 0

$\Sigma \leftarrow \mathsf{Sign}(\mathsf{gpk}, \mathsf{gsk}_i, M)$; If $\mathsf{Verify}(\mathsf{gpk}, M, \Sigma) = 0$ then return 1

$(j, \Pi_{\mathsf{open}}) \leftarrow \mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathbf{reg}, M, \Sigma)$; If $j \neq i$ then return 1

If $\mathsf{Judge}(\mathsf{gpk}, M, \Sigma, i, \mathsf{upk}_i, \Pi_{\mathsf{open}}) = 0$ then return 1 else return 0

$\underline{\mathbf{Exp}^{\mathsf{anon}-b}_{\mathsf{GS},\mathcal{A}}(\lambda)}$ // $b \in \{0, 1\}$

$(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GKg}(\lambda)$; $\mathrm{CU} \leftarrow \emptyset$; $\mathrm{HU} \leftarrow \emptyset$; $G \leftarrow \emptyset$

$b' \leftarrow \mathcal{A}(\mathsf{gpk}, \mathsf{ik} : \mathsf{Ch}_b, \mathsf{Open}, \mathsf{SndToU}, \mathsf{WReg}, \mathsf{RevealU}, \mathsf{CruptU})$

Return b'

$\underline{\mathbf{Exp}^{\mathsf{trace}}_{\mathsf{GS},\mathcal{A}}(\lambda)}$

$(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GKg}(\lambda)$; $\mathrm{CU} \leftarrow \emptyset$; $\mathrm{HU} \leftarrow \emptyset$;

$(M, \Sigma) \leftarrow \mathcal{A}(\mathsf{gpk}, \mathsf{ok} : \mathsf{SndToI}, \mathsf{AddU}, \mathsf{RReg}, \mathsf{RevealU}, \mathsf{CruptU})$

If $\mathsf{Verify}(\mathsf{gpk}, M, \Sigma) = 0$ then return 0; $(i, \Pi_{\mathsf{open}}) \leftarrow \mathsf{Open}(\mathsf{gpk}, \mathsf{ok}, \mathbf{reg}, M, \Sigma)$;

If $i = \perp$ or $\mathsf{Judge}(\mathsf{gpk}, M, \Sigma, i, \mathsf{upk}_i, \Pi_{\mathsf{open}}) = 0$ then return 1 else return 0

$\underline{\mathbf{Exp}^{\mathsf{nf}}_{\mathsf{GS},\mathcal{A}}(\lambda)}$

$(\mathsf{gpk}, \mathsf{ik}, \mathsf{ok}) \leftarrow \mathsf{GKg}(\lambda)$; $\mathrm{CU} \leftarrow \emptyset$; $\mathrm{HU} \leftarrow \emptyset$;

$(M, \Sigma, i, \Pi_{\mathsf{open}}) \leftarrow \mathcal{A}(\mathsf{gpk}, \mathsf{ok}, \mathsf{ik} : \mathsf{SndToU}, \mathsf{WReg}, \mathsf{Sign}, \mathsf{RevealU}, \mathsf{CruptU})$

If $\mathsf{Verify}(\mathsf{gpk}, M, \Sigma) = 0$ then return 0;

if the following are all true then return 1 else return 0:

- $i \in \mathrm{HU}$ and $\mathsf{gsk}_i \neq \epsilon$ and $\mathsf{Judge}(\mathsf{gpk}, M, \Sigma, i, \mathsf{upk}_i, \Pi_{\mathsf{open}}) = 1$
- $\mathcal{A}$ did not query $\mathsf{RevealU}(i)$ and $\mathsf{Sign}(i, M)$

Fig. 2: Experiments to define correctness and security requirements of a dynamic group signature scheme.

B Proof of Theorem 1

We provide the proof of Theorem 1, as appeared in [35]. We first restate the theorem.

Theorem 5. *Assume that* COM *is a statistically hiding and computationally binding string commitment scheme. Then, the protocol in Figure 1 is a statisti-*

cal ZKAoK *with perfect completeness, soundness error* $2/3$*, and communication cost* $\mathcal{O}(L \log q)$*. In particular:*

- *There exists a polynomial-time simulator that, on input* $(\mathbf{M}, \mathbf{u})$*, outputs an accepted transcript statistically close to that produced by the real prover.*
- *There exists a polynomial-time knowledge extractor that, on input a commitment* CMT *and* 3 *valid responses* $(\mathrm{RSP}_1, \mathrm{RSP}_2, \mathrm{RSP}_3)$ *to all* 3 *possible values of the challenge* Ch*, outputs* $\mathbf{w}' \in \mathsf{VALID}$ *such that* $\mathbf{M} \cdot \mathbf{w}' = \mathbf{u} \bmod q$*.*

Proof. It can be checked that the protocol has perfect completeness: If an honest prover follows the protocol, then he always gets accepted by the verifier. It is also easy to see that the communication cost is bounded by $\mathcal{O}(L \log q)$.

We now prove that the protocol is a statistical zero-knowledge argument of knowledge.

Zero-Knowledge Property. We construct a PPT simulator SIM interacting with a (possibly dishonest) verifier $\widehat{\mathcal{V}}$, such that, given only the public input, SIM outputs with probability negligibly close to $2/3$ a simulated transcript that is statistically close to the one produced by the honest prover in the real interaction.

The simulator first chooses a random $\overline{Ch} \in \{1, 2, 3\}$ as a prediction of the challenge value that $\widehat{\mathcal{V}}$ will *not* choose.

Case $\overline{Ch} = 1$: Using basic linear algebra over $\mathbb{Z}_q$, SIM computes a vector $\mathbf{w}' \in \mathbb{Z}_q^L$ such that $\mathbf{M} \cdot \mathbf{w}' = \mathbf{u} \bmod q$. Next, it samples $\mathbf{r}_w \xleftarrow{\$} \mathbb{Z}_q^L$, $\phi \xleftarrow{\$} \mathcal{S}$, and randomness ρ_1, ρ_2, ρ_3 for COM. Then, it sends the commitment $\mathrm{CMT} = \big(C_1', C_2', C_3'\big)$ to $\widehat{\mathcal{V}}$, where

$$C_1' = \mathsf{COM}(\phi, \mathbf{M} \cdot \mathbf{r}_w; \rho_1),$$
$$C_2' = \mathsf{COM}(\Gamma_\phi(\mathbf{r}_w); \rho_2), \quad C_3' = \mathsf{COM}(\Gamma_\phi(\mathbf{w}' + \mathbf{r}_w); \rho_3).$$

Receiving a challenge Ch from $\widehat{\mathcal{V}}$, the simulator responds as follows:

- If $Ch = 1$: Output $\perp$ and abort.
- If $Ch = 2$: Send $\mathrm{RSP} = \big(\phi, \mathbf{w}' + \mathbf{r}_w, \rho_1, \rho_3\big)$.
- If $Ch = 3$: Send $\mathrm{RSP} = \big(\phi, \mathbf{r}_w, \rho_1, \rho_2\big)$.

Case $\overline{Ch} = 2$: SIM samples $\mathbf{w}' \xleftarrow{\$} \mathsf{VALID}$, $\mathbf{r}_w \xleftarrow{\$} \mathbb{Z}_q^L$, $\phi \xleftarrow{\$} \mathcal{S}$, and randomness ρ_1, ρ_2, ρ_3 for COM. Then it sends the commitment $\mathrm{CMT} = \big(C_1', C_2', C_3'\big)$ to $\widehat{\mathcal{V}}$, where

$$C_1' = \mathsf{COM}(\phi, \mathbf{M} \cdot \mathbf{r}_w; \rho_1),$$
$$C_2' = \mathsf{COM}(\Gamma_\phi(\mathbf{r}_w); \rho_2), \quad C_3' = \mathsf{COM}(\Gamma_\phi(\mathbf{w}' + \mathbf{r}_w); \rho_3).$$

Receiving a challenge Ch from $\widehat{\mathcal{V}}$, the simulator responds as follows:

- If $Ch = 1$: Send $\mathrm{RSP} = \big(\Gamma_\phi(\mathbf{w}'), \Gamma_\phi(\mathbf{r}_w), \rho_2, \rho_3\big)$.
- If $Ch = 2$: Output $\perp$ and abort.

– If $Ch = 3$: Send RSP $= \big(\phi, \mathbf{r}_w, \rho_1, \rho_2\big)$.

Case $\overline{Ch} = 3$: SIM samples $\mathbf{w}' \xleftarrow{\$} \mathsf{VALID}$, $\mathbf{r}_w \xleftarrow{\$} \mathbb{Z}_q^L$, $\phi \xleftarrow{\$} \mathcal{S}$, and randomness ρ_1, ρ_2, ρ_3 for COM. Then it sends the commitment CMT $= \big(C_1', C_2', C_3'\big)$ to $\widehat{\mathcal{V}}$, where $C_2' = \mathsf{COM}(\Gamma_\phi(\mathbf{r}_w); \rho_2)$, $C_3' = \mathsf{COM}(\Gamma_\phi(\mathbf{w}' + \mathbf{r}_w); \rho_3)$ as in the previous two cases, while

$$C_1' = \mathsf{COM}(\phi, \mathbf{M} \cdot (\mathbf{w}' + \mathbf{r}_w) - \mathbf{u}; \rho_1).$$

Receiving a challenge Ch from $\widehat{\mathcal{V}}$, it responds as follows:

– If $Ch = 1$: Send RSP computed as in the case $(\overline{Ch} = 2, Ch = 1)$.
– If $Ch = 2$: Send RSP computed as in the case $(\overline{Ch} = 1, Ch = 2)$.
– If $Ch = 3$: Output $\perp$ and abort.

We observe that, in every case we have considered above, since COM is statistically hiding, the distribution of the commitment CMT and the distribution of the challenge Ch from $\widehat{\mathcal{V}}$ are statistically close to those in the real interaction. Hence, the probability that the simulator outputs $\perp$ is negligibly close to $1/3$. Moreover, one can check that whenever the simulator does not halt, it will provide an accepted transcript, the distribution of which is statistically close to that of the prover in the real interaction. In other words, we have constructed a simulator that can successfully impersonate the honest prover with probability negligibly close to $2/3$.

Argument of Knowledge. Suppose that $\mathrm{RSP}_1 = (\mathbf{t}_w, \mathbf{t}_r, \rho_2, \rho_3)$, $\mathrm{RSP}_2 = (\phi_2, \mathbf{w}_2, \rho_1, \rho_3)$, $\mathrm{RSP}_3 = (\phi_3, \mathbf{w}_3, \rho_1, \rho_2)$ are 3 valid responses to the same commitment $\mathrm{CMT} = (C_1, C_2, C_3)$, with respect to all 3 possible values of the challenge. The validity of these responses implies that:

$$\begin{cases} \mathbf{t}_w \in \mathsf{VALID}; \\ C_1 = \mathsf{COM}(\phi_2, \mathbf{M} \cdot \mathbf{w}_2 - \mathbf{u} \bmod q; \rho_1) = \mathsf{COM}(\phi_3, \mathbf{M} \cdot \mathbf{w}_3; \rho_1); \\ C_2 = \mathsf{COM}(\mathbf{t}_r; \rho_2) = \mathsf{COM}(\Gamma_{\phi_3}(\mathbf{w}_3); \rho_2); \\ C_3 = \mathsf{COM}(\mathbf{t}_w + \mathbf{t}_r \bmod q; \rho_3) = \mathsf{COM}(\Gamma_{\phi_2}(\mathbf{w}_2); \rho_3). \end{cases}$$

Since COM is computationally binding, we can deduce that

$$\begin{cases} \mathbf{t}_w \in \mathsf{VALID};\ \phi_2 = \phi_3;\ \mathbf{t}_r = \Gamma_{\phi_3}(\mathbf{w}_3);\ \mathbf{t}_w + \mathbf{t}_r = \Gamma_{\phi_2}(\mathbf{w}_2) \bmod q; \\ \mathbf{M} \cdot \mathbf{w}_2 - \mathbf{u} = \mathbf{M} \cdot \mathbf{w}_3 \bmod q. \end{cases} \tag{19}$$

Since $\mathbf{t}_w \in \mathsf{VALID}$, if we let $\mathbf{w}' = [\Gamma_{\phi_2}]^{-1}(\mathbf{t}_w)$, then $\mathbf{w}' \in \mathsf{VALID}$. Furthermore, we have

$$\Gamma_{\phi_2}(\mathbf{w}') + \Gamma_{\phi_2}(\mathbf{w}_3) = \Gamma_{\phi_2}(\mathbf{w}_2) \bmod q,$$

which implies that $\mathbf{w}' + \mathbf{w}_3 = \mathbf{w}_2 \bmod q$, and that $\mathbf{M} \cdot \mathbf{w}' + \mathbf{M} \cdot \mathbf{w}_3 = \mathbf{M} \cdot \mathbf{w}_2 \bmod q$. As a result, we have $\mathbf{M} \cdot \mathbf{w}' = \mathbf{u} \bmod q$. This concludes the proof. □

C Proof of Lemma 3

In this section, we prove Lemma 3. First, let us recall the lemma.

Lemma 6. *Assume that the* $\mathsf{RLWE}_{n,\ell,q,\chi}$ *problem is hard. Then the given group signature scheme is fully anonymous in the random oracle model.*

Proof. We prove this lemma using a series of games. In the first game, the challenger runs the experiment $\mathbf{Exp}^{\mathsf{anon}-0}_{\mathsf{GS},\mathcal{A}}(\lambda)$ while in the last game, the challenger runs the experiment $\mathbf{Exp}^{\mathsf{anon}-1}_{\mathsf{GS},\mathcal{A}}(\lambda)$. Denote W_i be the event that the adversary outputs 1 in Game i.

Game 0**:** This is exactly the experiment $\mathbf{Exp}^{\mathsf{anon}-0}_{\mathsf{GS},\mathcal{A}}(\lambda)$, where the adversary receives a challenged signature $\Sigma^* \leftarrow \mathsf{Sign}(\mathsf{gpk}, \mathsf{gsk}_{i_0}, M)$ in the challenge phase. So $\Pr[W_0] = \Pr[\mathbf{Exp}^{\mathsf{anon}-0}_{\mathsf{GS},\mathcal{A}}(\lambda) = 1]$.

Game 1**:** This game is the same as Game 0 with one exception: the challenger will keep both decryption keys of the ring-based variant of Regev's encryption scheme instead of erasing the second decryption key $(s_2, \mathbf{e}_2)$. However, the view of the adversary $\mathcal{A}$ is still the same as in Game 0. Therefore, $\Pr[W_0] = \Pr[W_1]$.

Game 2**:** This game is identical to Game 1 except that it generates simulated proofs for the *opening* oracle queries by programming the random oracle $\mathcal{H}_{\mathsf{FS}}$. Note that the challenger still follows the original game (that is, it uses s_1 to decrypt $\mathbf{c}_1$) to identify the real signer. The view of the adversary, however, is statistically indistinguishable between Game 1 and Game 2 by statistical zero-knowledge property of our argument system generating Π_{Open}. Therefore we have $\Pr[W_1] \stackrel{s}{\approx} \Pr[W_2]$.

Game 3**:** This game modifies Game 2 as follows. It uses s_2 instead of s_1 to decrypt the ciphertext $\mathbf{c}_2$ for the *opening* oracle queries. The view of $\mathcal{A}$ is the same as in Game 2 until event F_1, where $\mathcal{A}$ queries the *opening* oracle a valid signature $(\Pi_{\mathsf{gs}}, \mathbf{c}_1, \mathbf{c}_2)$ with $\mathbf{c}_1, \mathbf{c}_2$ encrypting distinct messages, happens. Since F_1 breaks the soundness of our argument generating Π_{gs}, we have $|\Pr[W_2] - \Pr[W_3]| \leq \Pr[F_1] \leq \mathbf{Adv}^{\mathsf{sound}}_{\Pi_{\mathsf{gs}}}(\lambda) = \mathsf{negl}(\lambda)$.

Game 4**:** This game changes Game 3 in the following way. It generates a simulated proof Π^*_{gs} in the *challenge* phase by programming the random oracle $\mathcal{H}_{\mathsf{FS}}$ even though the challenger has the correct witness to generate a real proof. The view is indistinguishable to $\mathcal{A}$ by statistical zero-knowledge property of our argument system generating Π^*_{gs}. Therefore we have $\Pr[W_3] \stackrel{s}{\approx} \Pr[W_4]$.

Game 5**:** In this game, we modify Game 4 by modifying the distribution of the challenged signature $\Sigma^* = (\Pi^*_{\mathsf{gs}}, \mathbf{c}^*_1, \mathbf{c}^*_2)$ as follows. Recall that in Game 4, both $\mathbf{c}^*_1$ and $\mathbf{c}^*_2$ encrypt the same message, i.e., $\mathsf{rdec}(p_{i_0})$. Here we change $\mathbf{c}^*_1$ to be encryption of $\mathsf{rdec}(p_{i_1})$ and keep $\mathbf{c}^*_2$ unchanged. By the semantic security of the underlying encryption scheme for public key $(\mathbf{a}, \mathbf{b}_1)$ (which is implied by the $\mathsf{RLWE}_{n,\ell,q,\chi}$ assumption since we no longer use s_1 to open

signatures), the change made in this game is negligible to the adversary. Therefore we have $|\Pr[W_4] - \Pr[W_5]| = \mathsf{negl}(\lambda)$.

Game 6**:** This game follows Game 5 with one change: it switches back to use s_1 for the *opening* oracle queries and erases $(s_2, \mathbf{e}_2)$ again. This modification is indistinguishable to the adversary until event F_2, where $\mathcal{A}$ queries the *opening* oracle a valid signature $(\Pi_{\mathsf{gs}}, \mathbf{c}_1, \mathbf{c}_2)$ with $\mathbf{c}_1, \mathbf{c}_2$ encrypting different messages, happens. However event F_2 breaks the simulation soundness of the argument system generating Π_{gs}. Therefore we have $|\Pr[W_5] - \Pr[W_6]| \leq \mathsf{Adv}^{\mathsf{ss}}_{\Pi_{\mathsf{gs}}}(\lambda) = \mathsf{negl}(\lambda)$.

Game 7**:** In this game, we modify Game 6 by modifying the distribution of the challenged signature $\Sigma^* = (\Pi^*_{\mathsf{gs}}, \mathbf{c}^*_1, \mathbf{c}^*_2)$ again. It changes $\mathbf{c}^*_2$ to be encryption of $\mathsf{rdec}(p_{i_1})$ in the *challenge* phase. By the semantic security of our encryption scheme for public key $(\mathbf{a}, \mathbf{b}_2)$ (which is implied by the $\mathsf{RLWE}_{n,\ell,q,\chi}$ assumption as we no longer use s_2 to open signatures), the change is negligible to the adversary. Hence we have $|\Pr[W_6] - \Pr[W_7]| = \mathsf{negl}(\lambda)$. Note that now both $\mathbf{c}^*_1$ and $\mathbf{c}^*_2$ encrypt the same message, i.e., $\mathsf{rdec}(p_{i_1})$, so Π^*_{gs} is a simulated proof even though the challenger has correct witness.

Game 8**:** In the game, we modify Game 7 in the following way. We switch to generate a real proof Π^*_{gs} in the *challenge* phase. By the statistical zero-knowledge property of our argument system generating Π^*_{gs}, the view of $\mathcal{A}$ is indistinguishable between Game 7 and Game 8. Therefore we have $\Pr[W_7] \overset{s}{\approx} \Pr[W_8]$.

Game 9**:** This game changes Game 8 in one aspect. It produces real proofs for the *opening* oracle queries. By the statistical zero-knowledge property of our argument system generating Π_{Open}, we have Game 8 and Game 9 are statistically indistinguishable to the adversary, i.e., we have $\Pr[W_8] \overset{s}{\approx} \Pr[W_9]$. This is actually the experiment $\mathbf{Exp}^{\mathsf{anon}-1}_{\mathsf{FDGS},\mathcal{A}}(\lambda)$. Hence, we obtain that $\Pr[W_9] = \Pr[\mathbf{Exp}^{\mathsf{anon}-1}_{\mathsf{FDGS},\mathcal{A}}(\lambda) = 1]$.

As a result, we have that

$$|\Pr[\mathbf{Exp}^{\mathsf{anon}-1}_{\mathsf{FDGS},\mathcal{A}}(\lambda) = 1] - \Pr[\mathbf{Exp}^{\mathsf{anon}-0}_{\mathsf{FDGS},\mathcal{A}}(\lambda) = 1]| = \mathsf{negl}(\lambda),$$

and hence our scheme is fully anonymous. □