To appear in SCN 2012

Fully Anonymous Attribute Tokens from Lattices

Jan Camenisch[1], Gregory Neven[1], and Markus Rückert[2]*

[1] IBM Research – Zurich
{jca,nev}@zurich.ibm.com
[2] markus.rueckert@cased.de

Abstract. Anonymous authentication schemes such as group signatures and anonymous credentials are important privacy-protecting tools in electronic communications. The only currently known scheme based on assumptions that resist quantum attacks is the group signature scheme by Gordon et al. (ASIACRYPT 2010). We present a generalization of group signatures called *anonymous attribute tokens* where users are issued attribute-containing credentials that they can use to anonymously sign messages and generate tokens revealing only a subset of their attributes. We present two lattice-based constructions of this new primitive, one with and one without opening capabilities for the group manager. The latter construction directly yields as a special case the first lattice-based group signature scheme offering full anonymity (in the random-oracle model), as opposed to the practically less relevant notion of chosen-plaintext anonymity offered by the scheme of Gordon et al. We also extend our scheme to protect users from framing attacks by the group manager, where the latter creates tokens or signatures in the name of honest users. Our constructions involve new lattice-based tools for aggregating signatures and verifiable CCA2-secure encryption.

Keywords Anonymous attribute tokens, group signatures, lattices, post-quantum cryptography.

* This work was supported by CASED (www.cased.de) and ECRYPT II (http://www.ecrypt.eu.org/).

1 Introduction

We all increasingly use electronic services in our daily lives. To do so, we currently have no choice but to provide plenty of personal information for authorization, billing purposes, or as part of the terms and conditions of service providers. Dispersing all these personal information erodes our privacy and puts us at risk of abuse of this information by criminals. Therefore, these services and their authentication mechanisms should be built in a way that minimizes the disclosed personal information. For instance, to access a resource, users should not need to identify themselves but rather only to prove to the resource provider that they possess the necessary attributes (e.g., rights or properties) which are required for the access. In fact, in Europe it is widely acknowledged that to secure the future digital infrastructure one must employ this kind of attribute-based access control and use so-called attribute-based credentials or minimal disclose tokens (see, e.g., [RIS10,IA11]).

The cryptographic research literature has put forth a large body of protocols that allow for privacy-friendly access control. For instance, group signature [CvH91] and identity escrow [KP98] schemes allow a user to prove that she has authorization (i.e., is member of a group of people who all share the same property) without revealing her identity. Nevertheless, in case of abuse of this anonymity, group signature and identity escrow schemes allow a designated party to lift the anonymity and to identify the abusing user. The generalization of these schemes are anonymous credentials or pseudonym systems [Cha81,Bra99,CL01b,LRSW99]. Such schemes feature a *plurality* of organizations who assign *attributes* to users by issuing attribute-containing credentials. Users are known to the different issuers under different pseudonyms. Later, when users need to authenticate somewhere, they can do so in the most privacy-protecting manner, i.e., users can just prove that they possess credentials asserting them the attributes required by the authentication policy.

It is well known that the cryptographic assumptions underlying all known realizations of these privacy-protecting schemes can be broken with quantum computers. The only exception to this is the group signature scheme by Gordon, Katz, and Vaikuntanathan [GKV10]. Their scheme works on ordinary computers but is based on the hardness of lattice problems, which are believed to be immune to quantum computers. While so far only small quantum computers breaking toy keys could be built, it seems very plausible that in just a few years computers breaking currently used keys can be built [Los10]. Even if quantum computers are not considered an immediate threat, the hardness of lattice problems against sub-exponential time adversaries and their provable worst-case to average-case relation makes it desirable to build cryptographic schemes from these problems.

In this paper we provide a number of new schemes for privacy-protecting authentication with security based on lattice problems in the random-oracle model. In particular, as our first contribution, we define and present an *anonymous attribute token scheme without anonymity revocation* (AAT–R). Here, a user can obtain a credential from a group manager or issuer, the credential containing the attributes that the manager wants to assert to the user. Later, the user can anonymously authenticate to a verifier by generating an authentication token from her credential, the token revealing only a subset of the attributes that are contained in the credential. Such authentication tokens are anonymous, i.e., a token containing a set of attributes could originate from any user who has been asserted a superset of these attributes. Minimal disclosure tokens as implemented by Microsoft's U-Prove [BP10] are an example of an AAT–R scheme.

As our second and main contribution, we extend our scheme to an *anonymous attribute token scheme with anonymity revocation* (AAT+R), where the group manager additionally has the power to reveal the identity of the user who generated a given token. Group signatures can be seen as special case of AAT+R schemes where the manager issues to all users a credential without attributes (or a single attribute with a fixed value). Our scheme provides anonymity to honest users in the presence of adversaries with adaptive access to the opening functionality. This is a major improvement over the group signature scheme of Gordon et al., who provide a much weaker form of anonymity. In their model, anonymity may break down for *all* users in the system as soon as a *single* signature (or token in our terminology) is opened, even for users who never misbehaved and never had their tokens opened. Hence, their scheme can only be used as long as no signature (token) is opened—an event that users are typically not even aware of. This is a severe limitation that we overcome.

We furthermore show how our AAT–R and AAT+R schemes can be combined to obtain a new AAT+R scheme that protects users from *framing* by a dishonest group manager. That is, in this resulting third scheme, no one except the user herself can produce tokens that when opened will be attributed to the user. This is a further property that the Gordon et al. group signature scheme does not provide and which we believe is rather important when one wants to have accountability. Group signatures obtained from our AAT+R schemes do not only provide better security compared to the Gordon et al. scheme, but also offer other advantages: the manager's public and secret key are independent of the number of users (versus linear in their scheme[1]) and users can join dynamically (in theirs, all the users' keys need to be generated at setup time). Thus, while our main focus is on anonymous attribute token schemes, we present as a corollary the first lattice-based, non-frameable group signature scheme with full anonymity.

As an aside, to construct our scheme, we improve upon known tools and introduce a number of new building blocks, which we believe are of interest in their own right. We provide a verifiable encryption proof protocol for the CCA2-secure encryption scheme of Peikert [Pei09] and introduce and construct single-signer aggregate signatures as a restricted, but useful, form of aggregate signatures [BGLS03].

Related Work. We do not claim anonymous attribute token to be a new primitive: the U-Prove scheme [BP10] and the signature scheme with its proof protocols by Camenisch and Lysyanskaya [CL04] actually realize instantiations of it based on the discrete logarithm assumption and the strong RSA assumption, respectively. Nevertheless, to the best of our knowledge, an anonymous attribute token scheme (with or without anonymity revocation) has never been formally defined. As we have pointed out already, group signature schemes can be seen as a special case of AAT+R schemes.

Several group signature schemes have been proposed in the literature. Most of these are based on strong RSA [ACJT00,AST02,CL01a] or on bilinear maps [BBS04,BS04,CL04,DP06,BCN$^+$10]. The scheme due to Gordon et al. [GKV10] is the only based on assumptions that resist attacks by quantum computers.

Attribute-based signatures [MPR11] are a related primitive where signatures cannot be opened and where the signer can prove any predicate over the attributes that can be expressed as a monotone span program, which includes circuits of threshold gates. Attribute-based group signatures [Kha07] are a similar primitive where signatures can be opened by a dedicated authority, and is thereby closely related to our notion of AAT+R schemes. Unfortunately, however, the security notions proposed in [Kha07] are flawed.[2]

Ring signatures [ST01] are another privacy-enabling primitive which can be seen as an adhoc group signature scheme without a central group manager and without the possibility for anonymity revocation. Ring signatures can also be constructed from our AAT–R scheme, as we shall point out later.

[1] Note that secret keys can always be made of constant length by storing the random seed used to generate the key instead of the key itself. Likewise, one can always publish the hash of the public key instead of the public key itself. The first trick involves re-generating keys, which is particularly costly in lattice-based schemes that use trapdoors. The latter trick comes at the cost of having to attach the full public key to each signature or token.

[2] The "selective-policy" anonymity notion of [Kha07] allows linkability of signatures when a signer signs the same message with the same set of revealed attributes twice. The traceability notion merely implies that any valid signature will open to some user. There is no guarantee that it opens to the actual signer behind the signature, however, nor does the notion offer any protection against users claiming attributes that they do not possess.

The most general privacy-enabling primitive are probably anonymous credential systems with additional features such as proving predicates over attributes, cryptographic pseudonyms, and partially blind issuing protocols to protect users against framing attacks by malicious issuers. While they are quite close to anonymous attribute token schemes, we leave it as an open problem to construct a full-fledged anonymous credential system based on lattices.

Organization of the Paper. After a brief preliminary section, we define anonymous attribute token schemes in Section 3. Then, we introduce, analyze, and discuss the building blocks for our constructions in Section 4, followed by our constructions in Section 4.2. Based on these results, we describe how to achieve group signatures and restricted anonymous credential systems in Section 6. There, we also discuss open research problems before we conclude the paper in Section 7.

2 Preliminaries

The statement $x \leftarrow_{\$} X$ means that x is chosen uniformly at random from the finite set X. A function is negligible if it vanishes faster than $1/p(n)$ for any polynomial p. All logarithms are base 2 and we identify $\{1, \ldots, k\}$ with $[k]$ and $(x_i)_{i=a}^{b}$ with $(x_a, \ldots, x_b)$. Furthermore, $[a, b]_{\mathbb{Z}} := [a, b] \cap \mathbb{Z}$. Instead of $a \equiv b \pmod q$, we simply write $a \equiv b$. When we write "$\|$", we mean the concatenation of strings or matrix columns. The concatenation of two vectors $\mathbf{x}, \mathbf{y}$ is denoted $[\mathbf{x}, \mathbf{y}]$. The notation $\#S$ denotes the cardinality of a finite set S.

In this work, we only require full-rank lattices. A lattice in $\mathbb{R}^n$ is a discrete subgroup $\Lambda = \{\sum_{i=1}^{n} x_i \, \mathbf{b}_i \mid x_i \in \mathbb{Z}\}$, typically represented by a matrix $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_n] \in \mathbb{Z}^{n \times n}$ of $\mathbb{R}$-linearly independent vectors. The matrix $\mathbf{B}$ is a basis of the lattice Λ and we write $\Lambda = \Lambda(\mathbf{B})$. The number of linearly independent vectors in $\mathbf{B}$ is the dimension $\dim(\Lambda)$. For a lattice $\Lambda(\mathbf{B})$ with $\mathbf{B} \in \mathbb{Z}^{n \times n}$ define the (full-rank) dual lattice as the set of all $\mathbf{x} \in \mathbb{R}^n$ with $\langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z}$ for all $\mathbf{y} \in \Lambda(\mathbf{B})$. The Gram-Schmidt orthogonalization (GSO) $\tilde{\mathbf{B}} = [\tilde{\mathbf{b}}_1 \| \ldots \| \tilde{\mathbf{b}}_n]$ of the columns of $\mathbf{B}$ is recursively computed by letting $\tilde{\mathbf{b}}_{i+1}$ be the orthogonal projection of $\mathbf{b}_{i+1}$ onto $\mathit{span}(\tilde{\mathbf{b}}_1, \ldots, \tilde{\mathbf{b}}_i)^{\perp}$. The length of $\mathbf{B}$ is defined as $\|\mathbf{B}\| := \max_{i \in [n]}(\|\mathbf{b}_i\|_2)$.

One of the main computational problems in lattices is the approximate shortest vector problem (SVP). Given a basis $\mathbf{B}$ of Λ and an approximation factor $\gamma \geq 1$, the task is to find a non-zero vector $\mathbf{v} \in \Lambda$ with length at most γ times the length of a shortest vector in Λ. A related problem is the approximate shortest independent vector problem (SIVP), where one is supposed to find a set $\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ of linearly independent vectors in Λ such that $\max_i \|\mathbf{v}_i\|_2 \leq \gamma \lambda_n$. Here, λ_n denotes the n-th successive minimum of Λ, which is the smallest radius of a sphere that contains n linearly independent lattice vectors. For polynomial (in the dimension) approximation factors, which are relevant for cryptography, the best known algorithms require exponential space × time, e.g., [MV10].

In cryptography, we use lattices of a special form, which we call *q-ary lattices*: for $q \in \mathbb{N}$, $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, we define $\Lambda_q^{\perp}(\mathbf{A}) := \{\mathbf{v} \in \mathbb{Z}^m : \mathbf{A}\,\mathbf{v} \equiv \mathbf{0}\}$. Its, up to scaling, dual lattice $\Lambda_q(\mathbf{A})$ is defined as $\{\mathbf{w} \in \mathbb{Z}^m : \exists \mathbf{e} \in \mathbb{Z}^n \text{ s.t. } \mathbf{A}^t\mathbf{e} \equiv \mathbf{w}\}$. The main computational problem in $\Lambda_q^{\perp}(\mathbf{A})$ is the following "short integer solution" (SIS) problem: given n, m, q, uniformly random $\mathbf{A}$, and a norm bound $1 \leq \nu < q$, find $\mathbf{v} \in \Lambda_q^{\perp}(\mathbf{A})$ with $0 < \|\mathbf{v}\|_2 \leq \nu$. The SIS problem was introduced and analyzed by Ajtai [Ajt96] but there are numerous improvements to the analysis [MR07,GPV08]. We will also use the (equivalent) inhomogeneous problem ISIS, where the task is to find a short vector $\mathbf{x}$ that solves $\mathbf{A}\mathbf{x} \equiv \mathbf{y}$ given $\mathbf{y}$. For $\nu \leq \mathsf{poly}(n)$, prime $q \geq \nu g(n)$ for $g(n) = \omega(\sqrt{n \log(n)})$, and $m \geq 2n \log(q)$, the average-case $\mathsf{SIS}(n, m, q, \nu)$ is at least as hard as SIVP with $\gamma = \nu \tilde{\mathcal{O}}(\sqrt{n})$ in the worst case. For $\Lambda_q(\mathbf{A})$, we consider the following "learning with errors" (LWE) problem: given $n, m, q, \mathbf{A}$, and m "noisy" inner products $\mathbf{b} \equiv \mathbf{A}^t\mathbf{s} + \mathbf{e} \bmod q$, where $\mathbf{e}$ is chosen from a certain error distribution Ψ over $\mathbb{Z}^m$. The task is to recover $\mathbf{s} \in \mathbb{Z}_q^n$. This search version of LWE is at least as hard as solving the decision problem, i.e., distinguish $(\mathbf{A}, \mathbf{b})$ from uniform. The *standard* error distribution is a spherical discretized normal distribution Ψ_α^m with width parameter to $\alpha = \alpha(n) \in (0,1)$. For prime $q > 2\sqrt{n}/\alpha$ and $m \leq \mathsf{poly}(n)$, these problems are, on the average, at least as hard as SIVP with $\gamma = \tilde{\mathcal{O}}(n/\alpha)$ in the worst case [Reg09] under a quantum reduction. A similar classical reduction can be found in [Pei09] at the expense of more constraints. We will use a different, true discrete Gaussian error distribution as defined below.

Gentry et al. [GPV08] define a special type of one-way trapdoor function called a *preimage samplable function.* For parameters $n \in \mathbb{N}$, $q = q(n) = \mathsf{poly}(n)$, $m = m(n) = \Omega(n\log(q))$, $\tilde{L} = \tilde{L}(n) = \mathcal{O}(n\log(n))$, $\rho(n) = \omega(\sqrt{\log(n)})$, and $\eta \geq \tilde{L}\rho(m)$ this one-way trapdoor function is defined as follows.

- $\mathsf{GPVGen}(1^n)$ generates a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, distributed statistically close to uniform, and a secret trapdoor $\mathbf{T} \in \mathbb{Z}^{m\times m}$ such that $\mathbf{AT} \equiv \mathbf{0}$ and $\left\|\tilde{\mathbf{T}}\right\| \leq \tilde{L}$.
- The one-way function associated to $\mathbf{A}$ is $f_{\mathbf{A}} : \mathbb{Z}^m \to \mathbb{Z}_q^n : \mathbf{x} \mapsto \mathbf{Ax} \pmod{q}$.
- $\mathsf{GPVInvert}(\mathbf{A}, \mathbf{T}, \mathbf{y}, \eta)$ samples elements from $f_{\mathbf{A}}^{-1}(\mathbf{y})$ so that $(\mathbf{x}, \mathbf{Ax} \pmod{q})$ as well as $(\mathsf{GPVInvert}(\mathbf{A}, \mathbf{T}, \mathbf{y}, \eta), \mathbf{y})$ are statistically close for $\mathbf{x} \sim D_{\mathbb{Z}^m,\eta}$ and $\mathbf{y} \leftarrow_\$ \mathbb{Z}_q^n$ for a certain distribution D, defined below.
- The samples $\mathbf{x}$ returned by $\mathsf{GPVInvert}$ have a conditional min-entropy of $\omega(\log(n))$, conditioned on $\mathbf{Ax} \equiv \mathbf{y}$ and $\|\mathbf{x}\|_2 \leq \eta\sqrt{m}$ (or, $\|\mathbf{x}\|_\infty \leq \eta\rho(m)$). Refer to [GPV08,AP09,Pei10] for further details.

Let Λ be a lattice. We define the distribution $D_{\Lambda,\eta,\mathbf{c}}$ with parameter η as in [GPV08]: for all $\mathbf{x} \in \Lambda + \mathbf{c}$, it is $D_{\Lambda,\eta,\mathbf{c}}(\mathbf{x}) = \frac{D_\eta(\mathbf{x})}{\sum_{\mathbf{y}\in\Lambda+\mathbf{c}} D_\eta(\mathbf{y})}$ for $D_\eta(\mathbf{x}) = 1/\eta^m \exp(-\pi\|\mathbf{x}\|^2/\eta^2)$. For $\mathbf{c} = \mathbf{0}$, we write $D_{\Lambda,\eta}$. Note that, as in [GKV10], this distribution will serve as an error distribution for LWE later.

Theorem 1 ([GPV08]). *The family is collision-resistant if* $\mathsf{SIS}(n, m, q, 2\eta\sqrt{m})$ *is hard.*

The GPV signature scheme [GPV08] is essentially a full-domain hash scheme [BR93] based on this one-way function. It uses $\mathbf{A}$ as a public key and the trapdoor $\mathbf{T}$ as the signing key. A signature on message M is a vector $\boldsymbol{\sigma}$ such that $\mathbf{A}\boldsymbol{\sigma} \equiv \mathsf{H}(M)$ and $\|\boldsymbol{\sigma}\|_2 \leq \eta\sqrt{m}$ which can be computed using the probabilistic $\mathsf{GPVInvert}$ algorithm.[3] Signing is stateful, i.e., when the same message is signed twice, the same signature is returned.

3 Syntax and Security of Anonymous Attribute Tokens

An anonymous attribute token (AAT) scheme can be seen as an extension of group signatures or as a simplification of anonymous credentials where the issuer can assign a list of attributes to a user's signing key. When authenticating to a verifier, the user can selectively reveal some of these attributes in a token and convince the verifier that she has a valid credential (i.e., signing key with attributes) certifying the claimed attribute values, without revealing any information about the non-revealed attributes and without making her tokens linkable – that is, more linkable than directly implied by the revealed attributes. We define and design two kind of schemes: AAT without anonymity revocation (AAT–R) where anonymity is absolute, i.e., opening tokens is impossible, even for the issuer; and AAT with anonymity revocation (AAT+R), where the manager can uncover the user who created a given token. Minimal disclosure tokens as implemented by Microsoft's U-Prove [BP10] are an example of an AAT–R scheme.

Our syntax and security definitions take inspiration from those for group signatures as put forward by Bellare et al. [BMW03], but we add support for dynamic issuing of credentials. We first lay out the definitions for the revocable anonymity setting (AAT+R), and then explain the differences to the AAT–R setting. We note that an AAT+R scheme does *not* trivially yield an AAT–R scheme, because in the former the manager can always open tokens, while the latter requires that even the manager cannot link tokens. The inverse relation does not hold either due to the lack of an opening algorithm in AAT–R schemes.

[3] With negligible probability, $\mathsf{GPVInvert}$ returns $\sigma = \mathbf{0}$ or $\|\sigma\|_2 > \eta\sqrt{m}$. In this case, the algorithm starts over.

Syntax of AAT+R schemes. An AAT+R scheme is parameterized by security parameter n, maximum number of users $u_{\max}$, and maximum number of attributes per credential ℓ_{max}, and is defined by the following algorithms.

- The manager runs $\mathsf{MKeyGen}$ on $1^n, u_{\max}$ to generate his public key *mpk* and corresponding secret key *msk*.
- When a user with index u requests a credential for an ordered list of attribute values $(a_i)_{i=1}^{\ell}$, with $\ell \leq \ell_{max}$, the manager runs Issue on input *msk*, u, and $(a_i)_{i=1}^{\ell}$ to generate a credential *cred*.
- A user generates an authentication token τ revealing a subset of attribute values $(a_i)_{i \in R}$ for $R \subseteq [\ell]$ and authenticating a message M by running the $\mathsf{GenToken}$ algorithm on input *mpk*, *cred*, $(a_i)_{i=1}^{\ell}$, R, and M. The message M can be any string; in practice, it could encode authentication context information such as the identity of the verifier, a timestamp, a session identifier, or a random nonce.
- To verify a token, the verifier runs the VToken algorithm on input *mpk*, the token τ, the set R, the revealed attribute values $(a_i)_{i \in R}$, and the message M. It outputs 1 or 0, indicating the validity of τ.
- Using the Open algorithm on input *msk*, a token τ, a set R, the revealed attributes $(a_i)_{i \in R}$, and a message M, the manager recovers the index u of the user that generated the token.

Correctness is defined in the straightforward way that any honestly generated token will be accepted. Security consists of *anonymity*, requiring that tokens generated by the same user cannot be linked, and *traceability*, requiring that no adversary can produce a token that cannot be opened or that, when opened, falsely incriminates an honest user.

Anonymity of AAT+R schemes. We consider *full anonymity* here, in other works (e.g., [BBS04]) often referred to as CCA2-anonymity, where the adversary has access to an opening oracle. The adversary $\mathcal{A}$ is given the manager's public key *mpk* as input. It has access to an *initialization oracle*, an *issuing oracle*, and an *opening oracle*, which offer the following functionalities.

- The initialization oracle, on input user index u and attribute values $(a_i)_{i=1}^{\ell}$, generates a credential $\mathit{cred}_u \leftarrow_{\$} \mathsf{Issue}(\mathit{isk}, u, (a_i)_{i=1}^{\ell})$. The oracle does not generate any direct output to $\mathcal{A}$, but stores cred_u locally, outside $\mathcal{A}$'s view. It can only be queried once for each user u. Once user u has been initialized, the adversary can query the issuing and token generation oracles for u.
- The issuing oracle, on input a user index u, returns cred_u if a credential for u was previously initialized, or $\perp$ otherwise.
- The opening oracle, on input token τ, attribute indices $R \subseteq [\ell]$, attribute values $(a_i)_{i \in R}$ and message M, returns $u \leftarrow \mathsf{Open}(\mathit{msk}, \tau, R, (a_i)_{i \in R}, M)$.

At the end of the first phase, $\mathcal{A}$ outputs user indices $u_0, u_1 \in [u_{\max}]$, a set $R \subseteq [\ell]$, and a message M. Let $(a_{i,0})_{i=1}^{\ell_0}$ and $(a_{i,1})_{i=1}^{\ell_1}$ be the attributes with which u_0 and u_1 were associated by the initialization oracle, respectively. If one of u_0 or u_1 has not been initialized, or if $a_{i,0} \neq a_{i,1}$ from some $i \in R$, then $\mathcal{A}$ loses the game. Otherwise, the challenger chooses a random bit b, generates a token $\tau^* \leftarrow_{\$} \mathsf{GenToken}(\mathit{ipk}, \mathit{opk}, \mathit{cred}_{u_b}, (a_{i,b})_{i=1}^{\ell_b}, R, M)$ and hands it to $\mathcal{A}$. The latter is allowed to make any additional oracle queries except submitting τ^* to the opening oracle. Eventually it outputs a bit b' and wins the game if $b' = b$.

Traceability of AAT+R schemes. The adversary $\mathcal{A}$ is given as input the manager's public key *mpk*. Apart from the initialization, issuing, and opening oracles described above, it has access to a *token generation oracle* offering the following functionality.

- The token generation oracle, on input user index u, attribute indices $R \subseteq [\ell]$, and message M, returns a token $\tau \leftarrow_{\$} \mathsf{GenToken}(\mathit{mpk}, \mathit{cred}_u, (a_i)_{i=1}^{\ell}, R, M)$ and returns τ to the adversary if a credential for u was previously initialized, or returns $\perp$ otherwise.

At the end of the game, $\mathcal{A}$ outputs τ^*, R^*, $(a_i^*)_{i\in R^*}$, and M^*. Let $u^* \leftarrow \mathsf{Open}(msk, \tau^*, R^*, (a_i^*)_{i\in R^*}, M^*)$ be the index of the user to whom the token is attributed by the opening algorithm. The adversary wins the game if $\mathsf{VToken}(mpk, ipk, R^*, (a_i^*)_{i\in R^*}, M^*) = 1$ and either

- $\mathcal{A}$ initialized u^* with attributes $(a_i)_{i=1}^{\ell}$ such that $a_i \neq a_i^*$ for some $i \in R^*$, or
- $\mathcal{A}$ never queried the issuing oracle on u^* and never queried a token by u^* on M^* and R^*.

Syntax and security of AAT–R schemes. An AAT–R scheme does not have an Open algorithm. It does, however, have an additional VCred algorithm that a user runs, upon receiving a credential *cred*, on input *mpk*, *cred*, $(a_i)_{i=1}^{\ell}$, to check whether *cred* is a well-formed credential. The algorithm returns 1 in case it is well-formed, or 0 if not.

We define a stronger anonymity notion for AAT–R than for AAT+R. The adversary $\mathcal{A}$ is given the manager's keys *mpk* and *msk* as input. At the end of the first phase, $\mathcal{A}$ outputs user indices $u_0, u_1 \in [u_{\max}]$, credentials $cred_{u_0}, cred_{u_1}$, lists of attribute values $(a_{i,0})_{i=1}^{\ell_0}, (a_{i,1})_{i=1}^{\ell_1}$, a set $R \subseteq [\min(\ell_0, \ell_1)]$, and a message M. If $\mathsf{VCred}(mpk, cred_b, (a_{i,b})_{i=1}^{\ell_b}) = 0$ for either of $b \in \{0,1\}$ or if $a_{i,0} \neq a_{i,1}$ from some $i \in R$, then $\mathcal{A}$ loses the game. Otherwise, the challenger chooses a random bit b, generates a token $\tau^* \leftarrow_{\$} \mathsf{GenToken}(mpk, cred_{u_b}, (a_{i,b})_{i=1}^{\ell_b}, R, M)$ and hands it to $\mathcal{A}$. The latter outputs a bit b' and wins the game if $b' = b$.

The traceability notion for AAT+R is replaced with the notion of *unforgeability* for AAT–R. In the unforgeability experiment, the adversary is given *mpk* as input. It has access to the same initialization, issuing, and token generation oracles as in the traceability game above. The adversary wins the game if $\mathsf{VToken}(mpk, \tau^*, R^*, (a_i^*)_{i\in R^*}, M^*) = 1$ and if for *all* users u initialized with attributes $(a_i)_{i=1}^{\ell}$ such that $a_i = a_i^*$ for all $i \in R^*$, $\mathcal{A}$ never queried the issuing oracle on u and never queried a token for u, M^*, R^*.

4 An Anonymous Attribute Token Scheme without Revocation

Our anonymous attribute token schemes build upon techniques in the GKV group signature scheme by Gordon et al. [GKV10]. We briefly recall their scheme and explain the fundamental differences in the way we issue credentials (signing keys) and generate tokens (signatures). In the GKV scheme, each user u is assigned a matrix $\mathbf{A}_u$ as public key and a corresponding trapdoor matrix $\mathbf{T}_u$ as signing key. To group-sign a message M, user u first uses $\mathbf{T}_u$ to compute a GPV signature [GPV08] $\boldsymbol{\sigma}_u$ on M, this GPV-signature being a *short* vector such that $\mathbf{A}_u\boldsymbol{\sigma}_u \equiv \mathsf{H}(M)$, where H is a hash function. She generates a "fake" GPV-signature $\boldsymbol{\sigma}_v$ for all other users $v \neq u$ through Gaussian elimination, i.e., $\boldsymbol{\sigma}_v$ will be a *long* vector such that $\mathbf{A}_v\boldsymbol{\sigma}_v \equiv \mathsf{H}(M)$. She subsequently encrypts each of these signatures using a variant of the Regev encryption scheme [Reg09] to obtain ciphertexts $\boldsymbol{\tau}_v = \mathbf{B}_v\mathbf{s} + \boldsymbol{\sigma}_v$ for $v = 1, \ldots, u_{\max}$, where $\mathbf{B}_v$ are matrices such that $\mathbf{A}_v\mathbf{B}_v^t \equiv 0$ and which are included in the group's public key. The encrypted GPV-signatures can still be verified by checking whether or not $\mathbf{A}_v\boldsymbol{\tau}_v \equiv \mathsf{H}(M)$ holds. The group signature contains the vectors $\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_{u_{\max}}$ plus a non-interactive witness-indistinguishable proof [MV03] that at least one of the encrypted GPV-signatures is actually short. Group signatures can be opened by decrypting $\boldsymbol{\tau}_v$ using a trapdoor $\mathbf{S}_v$ associated to $\mathbf{B}_v$ and checking which of the signatures $\boldsymbol{\sigma}_v$ is short.

Our AAT–R scheme uses only a single pair of matrices $\mathbf{A}, \mathbf{B}$ for the entire group, as opposed to a pair of matrices for each user. Only the manager knows the trapdoor $\mathbf{T}$ corresponding to $\mathbf{A}$. To prevent anyone, including the manager, from knowing a trapdoor corresponding to $\mathbf{B}$, the latter matrix is determined by a common reference string. The credential of a user u is a list of GPV signatures $\boldsymbol{\sigma}_{u,i}$ such that $\mathbf{A}\boldsymbol{\sigma}_{u,i} \equiv \mathsf{H}(u\|i\|a_i)$. A first idea to create a token for attribute a_i and message M could be to encrypt $\boldsymbol{\sigma}_{u,i}$ as in the GKV scheme and include M as an argument to the random oracle in the non-interactive proof that one of the ciphertexts $\boldsymbol{\tau}_v$ encrypts a short vector.

The problem with this approach, however, is that two signatures by the same user u can be linked by checking whether $\boldsymbol{\tau}_u - \boldsymbol{\tau'}_u$ is a lattice point. This can be fixed by re-randomizing the GPV signatures, for both real and fake ones, with a small short random $\mathbf{x} \sim D_{\mathbb{Z}^{m+n},\eta}$. To enable verifiability, we compute $\mathbf{y} \leftarrow \mathbf{A}\mathbf{x} \bmod q$ and append a non-interactive witness-indistinguishable proof of knowledge of a short vector $\mathbf{x}'$ such that $\mathbf{A}\mathbf{x}' \equiv \mathbf{y}$. This proof is the Fiat-Shamir transformation of a generalization of Lyubashevsky's identification scheme [Lyu08a], where the message M is included as an argument in the hash.

signatures is short.

This approach of treating each attribute separately has the obvious disadvantage that it blows up the signature size with a factor of $\#R \leq \ell$. We can obtain shorter tokens by observing that GPV signatures support a limited form of aggregation [BGLS03]. Namely, the GPV signatures $\boldsymbol{\sigma}_{u,i}$ for $i \in R$ can be summed up to form an aggregate signature $\boldsymbol{\alpha}_u \leftarrow \sum_{i \in R} \boldsymbol{\sigma}_{u,i}$. The aggregate satisfies $\mathbf{A}\boldsymbol{\alpha}_u \equiv \sum_{i \in R} \mathsf{H}(u \| i \| a_i)$ and is still "somewhat" short. Enabling such aggregation in Section 4.1.4 comes at the price of having to choose slightly larger security parameters, but only by a factor of $\log(\#R)$.

4.1 Cryptographic Ingredients

4.1.1 Sampling Orthogonal Lattices with Trapdoors Revisited. Gordon et al. [GKV10] present an algorithm that, given a matrix $\mathbf{B} \in \mathbb{Z}_q^{n \times (m+n)}$, samples a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times (m+n)}$ and an associated trapdoor $\mathbf{T} \in \mathbb{Z}^{n \times (m+n)}$ such that $\mathbf{A}\mathbf{B}^t \equiv \mathbf{0}$. We give a construction method based on [GKV10] that is more efficient and allows for better (i.e., shorter) trapdoors.

Proposition 1. *There exists a probabilistic polynomial-time (PPT) algorithm* $\mathsf{OrthoSamp}$ *that, on input* $\mathbf{B} = \mathbf{B}_1 \| \mathbf{B}_2 \in \mathbb{Z}_q^{n \times (m+n)}$ *with* $\mathbf{B}_2 \in (\mathbb{Z}_q^{n \times n})^*$*, outputs a pair* $(\mathbf{A}, \mathbf{T}) \in \mathbb{Z}_q^{n \times (m+n)} \times \mathbb{Z}^{n \times (m+n)}$ *such that (1)* $\mathbf{A}\mathbf{B}^t \equiv \mathbf{0}$*; (2)* $\mathbf{A}$ *is distributed statistically close to uniform (conditioned on* $\mathbf{A}\mathbf{B}^t \equiv \mathbf{0}$*); (3)* $\mathbf{A}\mathbf{T} \equiv \mathbf{0}$*; and (4)* $\left\|\tilde{\mathbf{T}}\right\| \leq \tilde{L}$.

From [CHKP10], we adopt the notion of extending a lattices basis to a larger dimension. The corresponding algorithm $\mathsf{ExtBasis}$ takes as input a matrix $\mathbf{A}_1$, a basis $\mathbf{T}_1$ of $\Lambda_q^\perp(\mathbf{A}_1)$, and an extension $\mathbf{A}_2$. It picks a uniformly random $\mathbf{V} \in \mathbb{Z}_q^{m \times n}$ such that $\mathbf{A}_1\mathbf{V} \equiv -\mathbf{A}_2$. Its output is a basis $\mathbf{T} = \left(\begin{array}{c|c} \mathbf{T}_1 & \mathbf{V} \\ \hline \mathbf{0} & \mathbf{I}_n \end{array}\right)$ of $\Lambda_q(\mathbf{A})$ for $\mathbf{A} = \mathbf{A}_1 \| \mathbf{A}_2$ with $\left\|\tilde{\mathbf{T}}\right\| \leq \left\|\tilde{\mathbf{T}_1}\right\| \leq \tilde{L}$.

Proof. First, generate $(\mathbf{A}_1, \mathbf{T}_1) \leftarrow \mathsf{GPVGen}(1^n)$. Then, set $\mathbf{A}_2 \leftarrow -\mathbf{A}_1\mathbf{B}_1^t(\mathbf{B}_2^{-1})^t = [\mathbf{a}_1^{(2)}, \ldots, \mathbf{a}_n^{(2)}]$ and compute the basis $\mathbf{T} \leftarrow \mathsf{ExtBasis}(\mathbf{A}_1, \mathbf{T}_1, \mathbf{A}_2)$. Output $\mathbf{A} = \mathbf{A}_1 \| \mathbf{A}_2$ and $\mathbf{T}$. The output satisfies (1) because $\mathbf{A}\mathbf{B}^t \equiv \mathbf{A}_1\mathbf{B}_1^t + \mathbf{A}_2\mathbf{B}_2^t \equiv \mathbf{A}_1\mathbf{B}_1^t - \mathbf{A}_1\mathbf{B}_1^t(\mathbf{B}_2^{-1})^t\mathbf{B}_2^t \equiv \mathbf{0}$. It satisfies (2) because the output $\mathbf{A}_1$ of GPVGen is distributed statistically close to uniform. It satisfies (3) because $\mathbf{A}\mathbf{T} \equiv \mathbf{A}_1\mathbf{T}_1 \| (\mathbf{A}_1\mathbf{V} + \mathbf{A}_2) \equiv \mathbf{0}$. Finally, to see that it satisfies (4), recall that $\mathbf{T}_1$ is a basis of $\mathbb{R}^m$. Thus, after GSO, we arrive at $\tilde{\mathbf{T}} = \left(\begin{array}{c|c} \tilde{\mathbf{T}}_1 & \mathbf{0} \\ \hline \mathbf{0} & I_n \end{array}\right)$ and, in consequence, have $\left\|\tilde{\mathbf{T}}\right\| = \left\|\tilde{\mathbf{T}_1}\right\| \leq \tilde{L}$. □

Notice that essentially the same procedure can be used to compute an orthogonal $\mathbf{A}$ such that $\mathbf{A}\mathbf{B}^t \equiv \mathbf{0}$ without a trapdoor for $\Lambda_q^\perp(\mathbf{A})$. Just sample a uniformly random matrix $\mathbf{A}_1$ in the first step and omit all subsequent steps that involve the trapdoor $\mathbf{T}_1$.

In our security proofs, we will require that a pair $(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{B}, \mathbf{T}_\mathbf{B})$ does not reveal in which order they were generated by $\mathsf{OrthoSamp}$ as stated by the following proposition. The proof is given in Appendix A.

Proposition 2. *Let* $X_1 = (\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{B}, \mathbf{T}_\mathbf{B})$ *and* $X_2 = (\mathbf{C}, \mathbf{T}_\mathbf{C}, \mathbf{D}, \mathbf{T}_\mathbf{D})$ *be random variables where*

$$\begin{array}{lcl} (\mathbf{B}_1, \mathbf{T}_{\mathbf{B}_1}) \leftarrow \mathsf{GPVGen}(1^n) & & (\mathbf{C}_1, \mathbf{T}_{\mathbf{C}_1}) \leftarrow \mathsf{GPVGen}(1^n) \\ \mathbf{B}_2 \leftarrow_{\$} (\mathbb{Z}_q^{n\times n})^* & & \mathbf{C}_2 \leftarrow_{\$} (\mathbb{Z}_q^{n\times n})^* \\ \mathbf{T}_\mathbf{B} \leftarrow \mathsf{ExtBasis}(\mathbf{B}_1, \mathbf{T}_{\mathbf{B}_1}, \mathbf{B}_2) & \textit{and} & \mathbf{T}_\mathbf{C} \leftarrow \mathsf{ExtBasis}(\mathbf{C}_1, \mathbf{T}_{\mathbf{C}_1}, \mathbf{C}_2) \\ (\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{OrthoSamp}(\mathbf{B}) & & (\mathbf{D}, \mathbf{T}_\mathbf{D}) \leftarrow \mathsf{OrthoSamp}(\mathbf{C}) \ . \end{array}$$

Then, X_1 and X_2 are statistically indistinguishable.

Observe that we have applied a simplification to the above proposition, where we choose $\mathbf{B}_2$ and $\mathbf{C}_2$ directly from the set of invertible matrices. Whenever the proposition is applied in our schemes, this property can be easily ensured by repeating the sampling procedure a small number of times. For our parameters, a good approximation of the ratio $\left|(\mathbb{Z}_q^{n\times n})^*\right| / \left|\mathbb{Z}_q^{n\times n}\right|$ is $e^{-1/(q-1)}$ and a lower bound is $(1-1/q)^n$. Since the choice of q is mainly governed by the worst-case to average-case reduction for SIS, demanding that $q \gg \nu$ for $\mathsf{SIS}(n, m, q, \nu)$, it will exceed $\eta\sqrt{m+n} = \Omega(n^{1.5}\log^{1.5}(n))$ in all our schemes. Hence, the fraction of invertible matrices over $\mathbb{Z}_q^{n\times n}$ is very close to 1.

All in all, our method differs from the corresponding lemma of [GKV10] in that we always use GPVGen in dimension m instead of sampling a trapdoor in dimension $m+n$ (as in [GKV10]) directly. Instead, we explicitly control how the trapdoor is extended to the super lattice. Hence, we have more control over the "shape" of the $(m+n)$-dimensional input trapdoor to OrthoSamp.

Our sampling algorithm differs from that of [GKV10] in that there, the trapdoor $\mathbf{T}$ is constructed via the basis randomizer RandBasis of Cash et al. [CHKP10]. It entails extending $\mathbf{T}_1$ to a basis for $\Lambda_q^\perp(\mathbf{A})$, sampling (via GPVInvert) a set of $\Omega((m+n)^2)$ short vectors from the lattice $\Lambda_q^\perp(\mathbf{A})$, a computation of the Hermite Normal Form, and the ToBasis algorithm of [MG02, Lemma 7.1]. Here, we simply append an easily computable matrix $\mathbf{V}$ to $\mathbf{T}_1$, which has two advantages. First, our method with complexity $\widetilde{O}(nm^2)$ is more efficient than [GKV10], where the complexity is dominated by $\widetilde{\Omega}((m+n)^4)$ for sampling $(m+n)^2$ lattice vectors using [Pei10]. Second, the quality of the trapdoor in [GKV10] is $\left\|\tilde{\mathbf{T}}\right\| = \Omega(\sqrt{m+n}\tilde{L})$ because GPVInvert (in dimension $m+n$) outputs vectors of length $\eta\sqrt{m+n} = \widetilde{\Omega}(\sqrt{m+n}\tilde{L})$. We have $\left\|\tilde{\mathbf{T}}\right\| = \tilde{L}$, which allows tighter security proofs w.r.t. the SIS problem. However, we do not obtain a nice bound on $\|\mathbf{T}\|$, which does not matter for most applications because they apply Babai's Nearest Plane algorithm [Bab86] or variants thereof [GPV08,Kle00], which only rely on $\left\|\tilde{\mathbf{T}}\right\|$ being small.

Efficient Sampling with Orthogonal Trapdoors. We apply a slight, well-known improvement to GPVInvert whenever we apply it in dimension $m+n$, i.e., whenever we call $\mathsf{GPVInvert}(\mathbf{A}, \mathbf{T}, \mathbf{t}, \eta)$ for $(\mathbf{A}, \mathbf{T})$ being output by OrthoSamp. Instead of sampling directly using $\mathbf{T}$, we use the upper-left part $\mathbf{T}_1$ of $\mathbf{T}$ and the following algorithm: 1. Sample $\mathbf{x}_2 \sim D_{\mathbb{Z}^n,\eta}$; 2. Call $\mathbf{x}_1 \leftarrow \mathsf{GPVInvert}(\mathbf{A}_1, \mathbf{T}_1, \mathbf{t} - \mathbf{A}_2\mathbf{x}_2, \eta)$; 3. Output $\mathbf{x}_1\|\mathbf{x}_2$. The result has norm at most $\eta\sqrt{m+n}$.

4.1.2 Verifiable Encryption of GPV Signatures.

As mentioned in the construction sketch, we will "encrypt" GPV signatures with a variant of the "dual" encryption scheme [GPV08]. To this end, we define the following family of one-way trapdoor functions based on the LWE problem. For ease of exposition, we will slightly abuse the terms *encryption* for this trapdoor one-way function and *ciphertext* for an image under this trapdoor in the subsequent sections. Fix a truncated error distribution Ψ over $\mathbb{Z}^m$ with support D_Ψ. Other parameters are the same as for GPV signatures.

- Keys are generated using $\mathsf{GPVGen}(1^n)$, yielding a public key $\mathbf{B}$ and corresponding trapdoor $\mathbf{S}$.
- The one-way function associated to $\mathbf{B}$ is $g_\mathbf{B} : \mathbb{Z}_q^n \times \mathbb{Z}^m \to \mathbb{Z}_q^m : (\mathbf{s}, \mathbf{e}) \mapsto \mathbf{B}^t\mathbf{s} + \mathbf{e} \bmod q$.
- $\mathsf{LWEInvert}(\mathbf{B}, \mathbf{S}, \boldsymbol{\tau})$ uses $\mathbf{S}$ to find a vector $\mathbf{B}^t\mathbf{s}'$ that is close to $\boldsymbol{\tau}$. Then, it computes $\mathbf{e}' \leftarrow \boldsymbol{\tau} - \mathbf{B}^t\mathbf{s}'$ and returns $(\mathbf{s}', \mathbf{e}')$.

Note that we modified $\mathsf{LWEInvert}()$ as to output $(\mathbf{s}', \mathbf{e}')$ instead of just $\mathbf{s}'$ as defined by Peikert [Pei09]. We will use $\Psi = \sum_{i=1}^{\ell} D_{\mathbb{Z}^m,\eta} = D_{\mathbb{Z}^m,\sqrt{\ell}\eta}$ and $D_\Psi = \{\mathbf{v} \in \mathbb{Z}^m : \|\mathbf{v}\|_2 \leq \eta\sqrt{m}\}$. Correctness follows from [Pei09] with $q(n) \geq \tilde{L}^2\rho^2(m)$, security as a one-way function follows from [Reg09,Pei09,GKV10].

Theorem 2 ([GKV10]). *The family is one-way if $g_{\mathbf{B}}(\mathbf{s}, \mathbf{e})$ is indistinguishable from uniform for $\mathbf{s} \leftarrow_{\$} \mathbb{Z}_q^n$ and $\mathbf{e} \sim \Psi$. It is hard to distinguish from uniform for Ψ if decision* LWE *is hard with the standard noise distribution $\Psi^m_{\sqrt{\ell}\eta/(q\sqrt{2})}$.*

Also note that if matrices $\mathbf{B}, \mathbf{A}, \mathbf{S}$ are generated via the GPVGen and $\mathsf{OrthoSamp}$ and $\boldsymbol{\sigma}$ is a GPV signature such that $\mathbf{A}\boldsymbol{\sigma} \equiv \mathsf{H}(M)$, then the "encrypted" signature $\boldsymbol{\tau} \leftarrow \mathbf{B}^t\mathbf{s} + \boldsymbol{\sigma} \bmod q$ can still be verified by checking that $\mathbf{A}\boldsymbol{\tau} \equiv \mathsf{H}(M)$. However, we need to ensure that the "noise" $\boldsymbol{\sigma}$ is small, which is why we require the following witness-indistinguishable proof of membership (WIPoM) system for bounded-distance decodeability (BDD).

4.1.3 Efficient Proofs for Lattice Problems.

As mentioned in the construction sketch, we need two non-interactive proofs for our scheme: a proof that at least one of a number of ciphertexts encrypts a short vector, and a proof of knowledge of $\mathbf{x}$ such that $\mathbf{A}\mathbf{x} \equiv \mathbf{y}$.

WIPoM for BDD. We use a variant $\mathcal{L}_{\mathsf{BDD}}(\gamma, \beta) := \{\mathcal{L}^{\mathsf{YES}}_{\mathsf{BDD}}(\gamma, \beta), \mathcal{L}^{\mathsf{NO}}_{\mathsf{BDD}}(\gamma, \beta)\}$ of the γ-GapCVP language [Reg10] for lattices $\Lambda_q(\mathbf{B})$. The "YES" and "NO" instances for words $(\mathbf{B}, \boldsymbol{\tau}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^m$ are defined as:

$$\begin{aligned} &\mathcal{L}^{\mathsf{YES}}_{\mathsf{BDD}}(\gamma, \beta) := \{(\mathbf{B}, \boldsymbol{\tau}) | \exists \mathbf{s} \in \mathbb{Z}_q^n : \left\|\boldsymbol{\tau} - \mathbf{B}^t\mathbf{s} \bmod q\right\|_2 \leq \beta\} \\ \text{and}\quad &\mathcal{L}^{\mathsf{NO}}_{\mathsf{BDD}}(\gamma, \beta) := \{(\mathbf{B}, \boldsymbol{\tau}) | \forall \mathbf{s} \in \mathbb{Z}_q^n : \left\|\boldsymbol{\tau} - \mathbf{B}^t\mathbf{s} \bmod q\right\|_2 > \gamma\beta\}. \end{aligned}$$

The norms above are computed by taking the absolute smallest representative modulo q of the coordinates, i.e., integers in the interval $[\frac{1-q}{2}, \frac{q-1}{2}]$. Micciancio and Vadhan [MV03] proposed a statistically zero-knowledge proof of membership $(\mathcal{P}_{\mathsf{BDD}}, \mathcal{V}_{\mathsf{BDD}})$ with one-bit challenges, together with a simulator $\mathcal{S}_{\mathsf{BDD}}$ that on input $\mathbf{B}, \boldsymbol{\tau}$ produces a transcript $(\mathit{cmt}, \mathit{ch}, \mathit{rsp})$. The k-bit parallelized proof system $(\mathcal{P}_{\mathsf{pBDD}}, \mathcal{V}_{\mathsf{pBDD}})$ is still statistical honest-verifier zero-knowledge, as the simulator $\mathcal{S}_{\mathsf{pBDD}}$ can generate n conversations using $\mathcal{S}_{\mathsf{BDD}}$ and output the concatenation of all transcripts. The corresponding proof of membership requires $\widetilde{\mathcal{O}}(n)$ bits of communication for all $\gamma = \Omega(\sqrt{n})$, using $k = \omega(\log(n))$ for a negligible soundness error [MV03].

Using standard techniques [CDS94,MV03,SCPY08], one can efficiently convert $(\mathcal{P}_{\mathsf{pBDD}}, \mathcal{V}_{\mathsf{pBDD}})$ into a sound WIPoM $(\mathcal{P}_{\vee\text{-}\mathsf{pBDD}}, \mathcal{V}_{\vee\text{-}\mathsf{pBDD}})$ for the OR-combination of such statements:

$$\mathcal{L}^{\mathsf{YES}}_{\vee\text{-}\mathsf{BDD}}(\gamma, \beta, u_{\max}) := \{((\mathbf{B}, \boldsymbol{\tau}_v))_{v=1}^{u_{\max}} | \exists v \in [u_{\max}]\, \exists \mathbf{s}_v \in \mathbb{Z}_q^n : \left\|\boldsymbol{\tau}_v - \mathbf{B}^t\mathbf{s}_v\right\|_2 \leq \beta\}.$$

The "NO" instance is defined analogously. The resulting prover $\mathcal{P}_{\vee\text{-}\mathsf{pBDD}}$ generates simulated transcripts by running $\mathcal{S}_{\mathsf{pBDD}}(\mathbf{B}, \boldsymbol{\tau}_v)$ for all $v \neq u$ and runs the real prover $\mathcal{P}_{\mathsf{pBDD}}((\mathbf{B}, \boldsymbol{\tau}_u), \mathbf{s}_u)$ to obtain the transcript for user u, using as a challenge the XOR of the given challenge of the $\mathcal{P}_{\vee\text{-}\mathsf{pBDD}}$ proof and the simulated challenges for $v \neq u$. This proof system is also statistical honest-verifier zero-knowledge since the simulator $\mathcal{S}_{\vee\text{-}\mathsf{pBDD}}$ can generate correctly distributed conversation transcripts by generating $u_{\max}$ transcripts using $\mathcal{S}_{\mathsf{pBDD}}(\mathbf{B}, \boldsymbol{\tau}_v)$ and setting the main challenge to be the XOR of the challenges in the generated transcripts. We will use the non-interactive variant of the proof system using the Fiat-Shamir transformation [FS86] where the challenge $\mathit{ch}_\vee$ is generated through a random oracle.

Signatures from ISIS. In addition, our constructions will require a signature scheme based on a generalized version of the witness-indistinguishable identification scheme due to Lyubashevsky [Lyu08a] that we recall and generalize in Appendix C. The main difference to [Lyu08a] is that we require an entirely different distribution of secret keys to make the scheme applicable in our context. To be more precise, the format of these keys plays a crucial role in ensuring anonymity.

The secret key is a short vector $\mathbf{x} \sim D_{\mathbb{Z}^{m+n},\eta}$, while the public key consists of a matrix $\mathbf{A} \leftarrow_{\$} \mathbb{Z}^{n\times(m+n)}$ and the vector $\mathbf{y} \leftarrow \mathbf{A}\mathbf{x} \bmod q$. It follows a typical three-move structure where the prover first generates a commitment and internal state $(cmt_{\mathsf{ISIS}}, st) \leftarrow_{\$} \mathsf{Comm}_{\mathsf{ISIS}}(\mathbf{A})$ and sends cmt_{ISIS} to the verifier. The verifier then chooses and sends back a challenge $ch_{\mathsf{ISIS}} \leftarrow_{\$} \{0,1\}^t$, upon which the prover sends the response $rsp_{\mathsf{ISIS}} \leftarrow_{\$} \mathsf{Resp}_{\mathsf{ISIS}}(\mathbf{x}, st, ch_{\mathsf{ISIS}})$. The verifier accepts iff $\mathsf{Verify}_{\mathsf{ISIS}}(\mathbf{A}, \mathbf{y}, cmt_{\mathsf{ISIS}}, ch_{\mathsf{ISIS}}, rsp_{\mathsf{ISIS}}) = 1$. The protocol has a non-zero completeness error, i.e., probability of rejecting an honest prover, of at most $2^{-t/14}$, which is negligible for $t = \omega(\log(n))$.

The identification scheme has been shown statistically witness-indistinguishable and secure under active attack assuming that the ISIS problem related to $\mathbf{A}, \mathbf{y}$ is hard [Lyu08a, Theorem 13]. The latter property is proved through a rewinding argument that from two valid transcripts $(cmt_{\mathsf{ISIS}}, ch_{\mathsf{ISIS}}, rsp_{\mathsf{ISIS}})$, $(cmt_{\mathsf{ISIS}}, ch'_{\mathsf{ISIS}}, rsp'_{\mathsf{ISIS}})$ extracts a vector $\mathbf{x}' \leftarrow \mathsf{Ext}_{\mathsf{ISIS}}(cmt_{\mathsf{ISIS}}, ch_{\mathsf{ISIS}}, rsp_{\mathsf{ISIS}}, ch'_{\mathsf{ISIS}}, rsp'_{\mathsf{ISIS}})$ such that $\mathbf{A}\mathbf{x}' \equiv \mathbf{y}$ and $\|\mathbf{x}'\|_2 \leq \widetilde{\mathcal{O}}(n^{1.5})$.

We will turn the identification scheme into a signature scheme using the Fiat-Shamir transformation. Note that the non-zero completeness error is less of an issue for signatures, as the signer can always generate a new signature in the unlikely event that an invalid signature is generated.

4.1.4 Single-Signer Aggregate Signatures. To make the token length logarithmic[4] instead of linear the number of attributes, we observe that GPV signatures support a restricted form of aggregation [BGLS03] where up to ℓ_{max} signatures by the same signer can be compressed to the size of a single signature. Namely, given $\ell \leq \ell_{max}$ signatures $(\boldsymbol{\sigma}_i)_{i=1}^{\ell}$, the aggregate $\boldsymbol{\alpha} \leftarrow \sum_{i=1}^{\ell} \boldsymbol{\sigma}_i$ can be verified by checking that $\ell \leq \ell_{max}$, that $0 < \|\boldsymbol{\alpha}\|_2 \leq \ell\eta\sqrt{m}$, and that $\mathbf{A}\boldsymbol{\alpha} \equiv \sum_{i=1}^{\ell} \mathsf{H}(M_i)$.

Because of the similarity in structure between GPV signatures and the above single-signer aggregate scheme, the latter inherits the mechanisms to verifiably encrypt aggregate signatures from Section 4.1.2. A more detailed description and a proof of the following theorem can be found in Appendix B.

Theorem 3. *The above single-signer aggregate signature scheme is existentially unforgeable in the random oracle model if the* $\mathsf{SIS}(n, m, q, 2\ell_{max}\eta\sqrt{m})$ *problem is hard.*

4.2 Scheme and Security

In the following, we describe an anonymous attribute token scheme AAT–R = $(\mathsf{IKeyGen}, \mathsf{Issue}, \mathsf{GenToken}, \mathsf{VToken})$ with security parameter n based on hard lattice problems. The scheme uses random oracles $\mathsf{H} : \{0,1\}^* \rightarrow \mathbb{Z}_q^{m+n}$, $\mathsf{F} : \{0,1\}^* \rightarrow \{0,1\}^k$, and $\mathsf{G} : \{0,1\}^* \rightarrow \{0,1\}^t$, as well as a uniformly distributed common reference string $\mathbf{B} \in \mathbb{Z}_q^{n\times(m+n)}$ that is a valid input to $\mathsf{OrthoSamp}$.

$\mathsf{MKeyGen}(1^n, u_{\max})$**:** The manager runs $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{OrthoSamp}(\mathbf{B})$ and sets $mpk \leftarrow \mathbf{A}$ and $msk \leftarrow (\mathbf{A}, \mathbf{T})$.

$\mathsf{Issue}(msk, u, (a_i)_{i=1}^{\ell})$**:** For all $i \in [\ell]$, the manager computes $\boldsymbol{\sigma}_{u,i} \leftarrow \mathsf{GPVInvert}(\mathbf{A}, \mathbf{T}, \mathsf{H}(u\|i\|a_i), \eta)$ and returns $cred = (u, (\boldsymbol{\sigma}_{u,i})_{i=1}^{\ell})$.

$\mathsf{VCred}(mpk, cred, (a_i)_{i=1}^{\ell})$**:** The user parses $cred = (u, (\boldsymbol{\sigma}_i)_{i=1}^{\ell})$ and outputs 1 iff $\mathbf{A}\boldsymbol{\sigma}_{u,i} \equiv \mathsf{H}(u\|i\|a_i)$ and $\|\boldsymbol{\sigma}_{u,i}\|_2 \leq \eta\sqrt{m+n}$ for all $i \in [\ell]$.

$\mathsf{GenToken}(mpk, cred, (a_i)_{i=1}^{\ell}, R, M)$**:** Let $cred = (u, (\boldsymbol{\sigma}_{u,i})_{i=1}^{\ell})$. The user first chooses a random $\mathbf{x} \sim D_{\mathbb{Z}^{m+n},\eta}$, computes $\mathbf{y} \leftarrow \mathbf{A}\mathbf{x} \bmod q$, and creates a signature $(ch_{\mathsf{ISIS}}, rsp_{\mathsf{ISIS}})$ by running $(cmt_{\mathsf{ISIS}}, st) \leftarrow_{\$} \mathsf{Comm}_{\mathsf{ISIS}}(\mathbf{A})$, setting $ch_{\mathsf{ISIS}} \leftarrow \mathsf{G}(\mathbf{y}\|cmt_{\mathsf{ISIS}}\|M)$, and computing $rsp_{\mathsf{ISIS}} \leftarrow \mathsf{Resp}_{\mathsf{ISIS}}(\mathbf{x}, st, ch_{\mathsf{ISIS}})$. In the unlikely event that $\mathsf{Verify}_{\mathsf{ISIS}}(\mathbf{A}, \mathbf{y}, cmt_{\mathsf{ISIS}}, ch_{\mathsf{ISIS}}, rsp_{\mathsf{ISIS}}) = 0$,

[4] While an aggregate signature may seem constant in length, the security parameter actually needs to grow logarithmically in ℓ_{max} for security.

she simply repeats these steps.
For all $v \in [u_{\max}] \setminus \{u\}$, she picks a uniformly random $\tilde{\boldsymbol{\alpha}}_v$ such that $\mathbf{A}\tilde{\boldsymbol{\alpha}}_v \equiv \sum_{i \in R} \mathsf{H}(v \| i \| a_i)$ using Gauss elimination, chooses $\mathbf{s}_v \leftarrow_{\$} \mathbb{Z}_q^n$ and computes $\boldsymbol{\tau}_v \leftarrow \mathbf{B}^t\mathbf{s}_v + \tilde{\boldsymbol{\alpha}}_v + \mathbf{x} \bmod q$. For her own index u, she chooses $\mathbf{s}_u \leftarrow_{\$} \mathbb{Z}_q^n$ and computes $\boldsymbol{\tau}_u \leftarrow \mathbf{B}^t\mathbf{s}_u + \boldsymbol{\alpha}_u + \mathbf{x} \bmod q$, where $\boldsymbol{\alpha}_u \leftarrow \sum_{i \in R} \boldsymbol{\sigma}_{u,i}$. She generates a non-interactive proof $(\mathit{cmt}_{\vee}, \mathit{rsp}_{\vee}) \leftarrow \mathcal{P}_{\vee\text{-pBDD}}(((\mathbf{B}, \boldsymbol{\tau}_v))_{v=1}^{u_{\max}}, u, \mathbf{s}_u)$ using as challenge $\mathit{ch}_{\vee} = \mathsf{F}(\mathbf{B} \| (\boldsymbol{\tau}_v)_{v=1}^{u_{\max}} \| \mathit{cmt}_{\vee} \| (a_i)_{i=1}^{\ell} \| R \| M)$. Finally, the resulting token becomes $\tau \leftarrow (\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_{u_{\max}}, \mathbf{y}, \mathit{cmt}_{\mathsf{ISIS}}, \mathit{rsp}_{\mathsf{ISIS}}, \mathit{cmt}_{\vee}, \mathit{rsp}_{\vee})$.

$\mathsf{VToken}(\mathit{mpk}, \tau, R, (a_i)_{i \in R}, M)$**:** The verifier accepts a token if $\mathsf{Verify}_{\mathsf{ISIS}}(\mathbf{A}, \mathbf{y}, \mathit{cmt}_{\mathsf{ISIS}}, \mathsf{G}(\mathbf{y} \| \mathit{cmt}_{\mathsf{ISIS}} \| M), \mathit{rsp}_{\mathsf{ISIS}}) = 1$, if $\mathbf{A}\boldsymbol{\tau}_v \equiv \sum_{i \in R} \mathsf{H}(v \| i \| a_i) + \mathbf{y}$ for all $v \in [u_{\max}]$, and if $\mathcal{V}_{\vee\text{-pBDD}}$ accepts the proof $(\mathit{cmt}_{\vee}, \mathit{rsp}_{\vee})$ for statement $((\mathbf{B}, \boldsymbol{\tau}_v))_{v=1}^{u_{\max}}$ and challenge $\mathit{ch}_{\vee} = \mathsf{F}(\mathbf{B} \| (\boldsymbol{\tau}_v)_{v=1}^{u_{\max}} \| \mathit{cmt}_{\vee} \| (a_i)_{i=1}^{\ell} \| R \| M)$. Otherwise, the verifier rejects the token.

Security statements and proof sketches are provided below, full proofs can be found in Appendix D and E.

Theorem 4. *The above anonymous attribute token scheme is anonymous in the random oracle model if* LWE *is hard for* $\Psi = D_{\mathbb{Z}^{m+n}, \eta}$.

Proof (sketch). Using the hardness of the decision LWE problem and the statistical zero-knowledge property of the proof system for BDD, we show that the adversary's view is independent of the coin flip $b \leftarrow_{\$} \{0,1\}$ in the experiment. We achieve this in a series of indistinguishable games, with the first game being the anonymity experiment for $b = 0$ and the last one being the same experiment with $b = 1$. The overall strategy is as follows. We generate all keys honestly and pass them to adversary, who responds with two user indices $u_0, u_1 \in [u_{\max}]$, credentials $\mathit{cred}_{u_0}, \mathit{cred}_{u_1}$, and requests a token for u_b. Now, instead of using just the short vectors in cred_{u_0} to generate and return a token, we modify the experiment to eventually use cred_{u_0} and cred_{u_1} to create a token with two vectors that are close the $\Lambda_q(\mathbf{B})$.

The assumptions are used in the following way. When replacing the vector $\boldsymbol{\tau}_{u_1} \leftarrow \mathbf{B}^t\mathbf{s}_{u_1} + \mathbf{x} + \tilde{\boldsymbol{\alpha}}_{u_1} \bmod q$ (far from $\Lambda_q(\mathbf{B})$) with a vector $\boldsymbol{\tau}_{u_1} \leftarrow \mathbf{B}^t\mathbf{s}_{u_1} + \mathbf{x} + \boldsymbol{\alpha}_{u_1} \bmod q$ (close to $\Lambda_q(\mathbf{B})$), the underlying argument uses the decision LWE assumption to establish computational indistinguishability. Then, when switching from using a witness for u_0 to using a witness for u_1, we exploit the statistical witness-indistinguishability of the proof system for BDD in $\Lambda_q(\mathbf{B})$.

Theorem 5. *The above anonymous attribute token scheme is unforgeable in the random oracle model if* $\mathsf{SIS}(n, m+n, q, (2\ell_{max}+1)\eta\sqrt{m+n} + \widetilde{\mathcal{O}}(n^{1.5}))$ *is hard and the decision* LWE *problem with noise distribution* Ψ *is hard.*

Proof (sketch). We simulate the issuing oracle without knowing a trapdoor for $\Lambda_q^{\perp}(\mathbf{A})$ in the standard way, i.e., by generating random short GPV signatures and programming the random oracle H. Rather than generating tokens based on these signatures, we compute long vectors that satisfy the verification equation; the tokens thus obtained are indistinguishable from tokens generated by short vectors if the decision LWE problem is hard. We simulate the BDD proof without a witness using the simulator $\mathcal{S}_{\vee\text{-pBDD}}$ and by programming the random oracle F. As a consequence, the adversary's view is independent of the stored GPV signatures of uncorrupted users.

To use the forged token $\boldsymbol{\tau}^*$ to solve the SIS problem, we change the way the manager's keys are generated by taking $\mathbf{A}$ from the SIS problem instance, generating $(\mathbf{B}, \mathbf{S}) \leftarrow \mathsf{OrthoSamp}(\mathbf{A})$, and programming the common reference string that defines $\mathbf{B}$. The matrix $\mathbf{A}$ is an admissible input to $\mathsf{OrthoSamp}$ with high probability and the argument only requires a non-negligible fraction of matrices $\mathbf{A}$ to work. By Proposition 2, this change in how keys are generated does not affect the adversary's view. Using $\mathbf{B}$, we can "decrypt" the vectors $\boldsymbol{\tau}_v^*$ included in $\boldsymbol{\tau}^*$ and, by the soundness of $\mathcal{V}_{\vee\text{-pBDD}}$, we can recover a user index u^* and a short vector $\boldsymbol{\rho}$ such that $\mathbf{A}\boldsymbol{\rho} \equiv \sum_{i \in R^*} \mathsf{H}(u^* \| i \| a_i^*) + \mathbf{y}^*$. Then, we use the knowledge extractor for $(\mathit{cmt}_{\mathsf{ISIS}}^*, \mathit{rsp}_{\mathsf{ISIS}}^*)$ in

$\boldsymbol{\tau}^*$ to extract a short vector $\mathbf{x}'$ such that $matA\mathbf{x}' \equiv \mathbf{y}^*$. The vector $\boldsymbol{\alpha}' \leftarrow \boldsymbol{\rho} - \mathbf{x}'$ is therefore also short and a solution to $\mathbf{A}\boldsymbol{\alpha}' \equiv \sum_{i \in R^*} \mathsf{H}(u^* \| i \| a_i^*)$. The simulator internally knows a second vector $\boldsymbol{\alpha}$ satisfying the same conditions, namely the sum of the stored GPV signatures, which with overwhelming probability is different from $\boldsymbol{\alpha}'$ since the adversary's view is independent of the stored signatures. Therefore, $\boldsymbol{\alpha} - \boldsymbol{\alpha}'$ is a short vector for $\Lambda_q^\perp(\mathbf{A})$.

5 An Anonymous Attribute Token Scheme with Revocation

To add opening functionality to the AAT–R scheme of the previous section, we generate the matrix $\mathbf{B}$ with an embedded trapdoor $\mathbf{S}$ using $\mathsf{OrthoSamp}$, as done in [GKV10]. To achieve full anonymity, however, we need to be able to respond to opening queries. For this purpose, we borrow techniques from Rosen and Segev [RS09] and Peikert [Pei09] to obtain CCA-security for the LWE encryption scheme by using "correlated" ciphertexts. One problem is that the verifier needs a way to check that the included ciphertexts are valid, i.e., correctly correlated, without having the trapdoor $\mathbf{S}$. We solve this problem by a clever use of the $\mathcal{P}_{\mathsf{V\text{-}pBDD}}$ proof system so that it simultaneously proves that a ciphertext contains a short vector and is correctly correlated.

5.1 Cryptographic Ingredients

Correlated trapdoor one-way functions. Following Rosen and Segev [RS09] and Peikert [Pei09], we define the following family of correlated trapdoor one-way functions CTLWE with parameters $n, m, \kappa \in \mathbb{N}$.

Key generation: The algorithm $\mathsf{CTGen}(1^n, 1^m, 1^\kappa)$ generates $(\mathbf{B}_{0,1}, \mathbf{S}_{0,1}) \leftarrow \mathsf{GPVGen}(1^n)$, chooses a random matrix $\mathbf{B}_{0,2} \leftarrow_\$ \mathbb{Z}_q^{n \times n}$, and sets $\mathbf{B}_0 = \mathbf{B}_{0,1} \| \mathbf{B}_{0,2}$ and $\mathbf{S}_0 = \left(\begin{array}{c|c} \mathbf{S}_{0,1} & \mathbf{V} \\ \hline \mathbf{0} & I_n \end{array}\right)$ such that $\mathbf{B}_{0,1}\ \mathbf{V} \equiv -\mathbf{B}_{0,2}$. It also generates random matrices $\mathbf{B}_i \leftarrow_\$ \mathbb{Z}_q^{n \times (m+n)}$ for $i = 1, \ldots, \kappa$. The public key is $(\mathbf{B}_0, \ldots, \mathbf{B}_\kappa)$, the corresponding trapdoor is $\mathbf{S}_0$.

Evaluation: The one-way function associated to $(\mathbf{B}_0, \ldots, \mathbf{B}_\kappa)$ is $g_{(\mathbf{B}_0,\ldots,\mathbf{B}_\kappa)} : \mathbb{Z}_q^n \times (\mathbb{Z}_q^{m+n})^{\kappa+1} \to (\mathbb{Z}_q^{m+n})^{\kappa+1} : (\mathbf{s}, \mathbf{e}_0, \ldots, \mathbf{e}_\kappa) \mapsto (\mathbf{B}_0^t \mathbf{s} + \mathbf{e}_0 \bmod q, \ldots, \mathbf{B}_\kappa^t \mathbf{s} + \mathbf{e}_\kappa \bmod q)$.

Inversion: Algorithm $\mathsf{CTInvert}((\mathbf{B}_0, \ldots, \mathbf{B}_\kappa), \mathbf{S}_0, (\mathbf{b}_0, \ldots, \mathbf{b}_\kappa))$ runs $(\mathbf{s}', \mathbf{e}'_0) \leftarrow \mathsf{LWEInvert}(\mathbf{B}_0, \mathbf{S}_0, \mathbf{b}_0)$ and computes $\mathbf{e}'_i \leftarrow \mathbf{b}_i - \mathbf{B}_i^t \mathbf{s}' \bmod q$ for $i = 1, \ldots, \kappa$. If $\|\mathbf{e}'_i\|_2 \leq \eta\sqrt{m}$ for all $i \in \{0, \ldots, \kappa\}$ then it returns $(\mathbf{s}', \mathbf{e}'_0, \ldots, \mathbf{e}'_\kappa)$, else it returns $\perp$. The norm of $\mathbf{e}'_i$ is computed in a natural way by first selecting the absolute smallest representative first.

If the above function family is evaluated on uniformly random $\mathbf{s} \leftarrow_\$ \mathbb{Z}^n$ and $\mathbf{e}_i \sim \Psi$, then $g_{(\mathbf{B}_0,\ldots,\mathbf{B}_\kappa)}(\mathbf{s}, \mathbf{e}_0, \ldots, \mathbf{e}_\kappa)$ is indistinguishable from random if LWE is hard for Ψ [RS09,Pei09].

One-time signatures. A one-time signature scheme $\mathcal{OTS}$ is a triple of algorithms ($\mathsf{OTKeygen}$, $\mathsf{OTSign}, \mathsf{OTVerify}$) where $\mathsf{OTKeygen}(1^n)$ outputs a verification key $otvk \in \{0,1\}^\kappa$ and a signing key $otsk$; algorithm $\mathsf{OTSign}(otsk, M)$ outputs a signature $otsig$, and $\mathsf{OTVerify}(otvk, M, otsig)$ outputs 1 or 0 indicating whether the signature is valid. The security notion of strong existential unforgeability under one-time chosen-message attack requires that no adversary, on input $otvk$ and after a single signature query M yielding $otsig \leftarrow_\$ \mathsf{OTSign}(otsk, M)$, can output M^*, $otsig^*$ such that $\mathsf{OTVerify}(otvk, M^*, otsig^*) = 1$ and $(M^*, otsig^*) \neq (M, otsig)$.

5.2 Scheme and Security

We now describe fully anonymous AAT+R scheme with the same parameters n, $u_{\max}$, t, m as the AAT–R scheme from Section 4.2, plus the signature length κ. The scheme uses random oracles $\mathsf{H} : \{0,1\}^* \to \mathbb{Z}_q^{m+n}$, $\mathsf{F} : \{0,1\}^* \to \{0,1\}^k$ and $\mathsf{G} : \{0,1\}^* \to \{0,1\}^t$.

$\mathsf{MKeyGen}(1^n, u_{\max})$**:** The manager runs $\mathsf{CTGen}(1^n, 1^m, 1^\kappa)$ to generate matrices $((\mathbf{B}_0, \mathbf{B}_{1,0} \ldots, \mathbf{B}_{\kappa,0}), \mathbf{S}_0)$ and generates κ additional random matrices $\mathbf{B}_{1,1}, \ldots, \mathbf{B}_{\kappa,1} \leftarrow_\$ \mathbb{Z}_q^{n\times(m+n)}$. He also runs $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{OrthoSamp}(\mathbf{B}_0)$. The public key is $mpk \leftarrow (\mathbf{A}, \mathbf{B}_0, \mathbf{B}_{1,0}, \mathbf{B}_{1,1}, \ldots, \mathbf{B}_{\kappa,0}, \mathbf{B}_{\kappa,1})$, the secret key is $msk \leftarrow (\mathbf{T}, \mathbf{S}_0, mpk)$. If $\mathbf{B}_0$ is not accepted by $\mathsf{OrthoSamp}$, it is re-sampled.

$\mathsf{Issue}(msk, u, (a_i)_{i=1}^{\ell})$**:** For all $i \in [\ell]$, the manager computes $\boldsymbol{\sigma}_{u,i} \leftarrow \mathsf{GPVInvert}(\mathbf{A}, \mathbf{T}, \mathsf{H}(u\|i\|a_i), \eta)$ and outputs $cred = (u, (\boldsymbol{\sigma}_{u,i})_{i=1}^{\ell})$.

$\mathsf{GenToken}(mpk, cred, (a_i)_{i=1}^{\ell}, R, M)$**:** Let $cred = (u, (\boldsymbol{\sigma}_{u,i})_{i=1}^{\ell})$. The user u proceeds as follows:

1. She chooses $\mathbf{x}_0 \sim D_{\mathbb{Z}^{m+n},\eta}$, computes $\mathbf{y}_0 \leftarrow \mathbf{A}\mathbf{x}_0 \bmod q$, and creates a signature $(cmt_{\mathsf{ISIS}}, rsp_{\mathsf{ISIS}})$ using $ch_{\mathsf{ISIS}} = \mathsf{G}(\mathbf{y}_0\|cmt_{\mathsf{ISIS}}\|M)$ as the challenge.
2. She computes $\boldsymbol{\rho}_u \leftarrow \sum_{i\in R} \boldsymbol{\sigma}_{u,i} + \mathbf{x}_0 \bmod q$, chooses $\mathbf{s}_u \leftarrow_\$ \mathbb{Z}_q^n$ and computes $\boldsymbol{\tau}_{u,0} \leftarrow \mathbf{B}_0^t\mathbf{s}_u + \boldsymbol{\rho}_u \bmod q$.
3. For all $v \in [u_{\max}] \setminus \{u\}$, she computes $\tilde{\boldsymbol{\alpha}}_v$ such that $\mathbf{A}\tilde{\boldsymbol{\alpha}}_v \equiv \sum_{i\in R} \mathsf{H}(v\|i\|a_i)$ using Gauss elimination, computes $\boldsymbol{\rho}_v \leftarrow \tilde{\boldsymbol{\alpha}}_v + \mathbf{x}_0 \bmod q$, chooses $\mathbf{s}_v \leftarrow_\$ \mathbb{Z}_q^n$ and computes $\boldsymbol{\tau}_{v,0} \leftarrow \mathbf{B}_0^t\mathbf{s}_v + \boldsymbol{\rho}_v \bmod q$.
4. She generates a signature key pair $(otvk, otsk) \leftarrow \mathsf{OTKeygen}(1^n)$. Let $otvk = otvk_1\|\ldots\| otvk_\kappa$.
5. For all $v \in [u_{\max}]$ and $i \in [\kappa]$ she chooses $\mathbf{x}_{v,i} \sim D_{\mathbb{Z}^{m+n},\eta}$ and computes $\boldsymbol{\tau}_{v,i} \leftarrow \mathbf{B}_{i,otvk_i}^t\mathbf{s}_v + \mathbf{x}_{v,i} \bmod q$.
6. Let $\mathbf{B}_{otvk} = [\mathbf{B}_0\|\mathbf{B}_{1,otvk_1}\|\ldots\|\mathbf{B}_{\kappa,otvk_\kappa}] \in \mathbb{Z}_q^{n\times(\kappa+1)(m+n)}$. For all $v \in [u_{\max}]$ let $\mathbf{x}_v = [\boldsymbol{\rho}_v, \mathbf{x}_{v,1}, \ldots, \mathbf{x}_{v,\kappa}] \in \mathbb{Z}_q^{(\kappa+1)(m+n)}$ and $\boldsymbol{\tau}_v = [\boldsymbol{\tau}_{v,0}, \ldots, \boldsymbol{\tau}_{v,\kappa}] \in \mathbb{Z}_q^{(\kappa+1)(m+n)}$. Then for all $v \in [u_{\max}]$ we have that $\boldsymbol{\tau}_v \equiv \mathbf{B}_{otvk}^t\mathbf{s}_v + \mathbf{x}_v$, and for user u we have that $\|\mathbf{x}_u\|_2 \leq (\#R + \kappa + 1)\eta\sqrt{m+n}$. The signer can therefore create a non-interactive proof $(cmt_\vee, rsp_\vee) \leftarrow \mathcal{P}_{\vee\text{-}\mathsf{pBDD}}(((\mathbf{B}_{otvk}, \boldsymbol{\tau}_v))_{v=1}^{u_{\max}}, u, \mathbf{s}_u)$ using $ch_\vee = \mathsf{F}(\mathbf{B}_{otvk}\| (\boldsymbol{\tau}_v)_{v=1}^{u_{\max}} \|cmt_\vee\| (a_i)_{i=1}^{\ell} \|R\|M)$ as a challenge to simultaneously prove that one of the vectors $\boldsymbol{\tau}_v$ encrypts a short vector $\boldsymbol{\alpha}_v$ and that all ciphertexts $\boldsymbol{\tau}_v$ are well-formed, i.e., that all components $\boldsymbol{\tau}_{v,i}$ are underlain by the same vector $\mathbf{s}_v$.
7. Finally, the signer generates a one-time signature $otsig \leftarrow \mathsf{OTSign}(otsk, (\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_{u_{\max}}, \mathbf{y}_0, cmt_{\mathsf{ISIS}}, rsp_{\mathsf{ISIS}}, cmt_\vee, rsp_\vee))$.

The token is the tuple $\tau \leftarrow (\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_{u_{\max}}, \mathbf{y}_0, cmt_{\mathsf{ISIS}}, rsp_{\mathsf{ISIS}}, cmt_\vee, rsp_\vee, otvk, otsig)$.

$\mathsf{VToken}(mpk, \tau, R, (a_i)_{i\in R}, M)$**:** The verifier checks that $\mathsf{Verify}_{\mathsf{ISIS}}(\mathbf{A}, \mathbf{y}_0, cmt_{\mathsf{ISIS}}, \mathsf{G}(\mathbf{y}_0\|cmt_{\mathsf{ISIS}}\|M), rsp_{\mathsf{ISIS}}) = 1$, that $\mathbf{A}\boldsymbol{\tau}_{v,0} \equiv \sum_{i\in R} \mathsf{H}(v\|i\|a_i) + \mathbf{y}_0$ for all $v \in [u_{\max}]$, that $\mathsf{OTVerify}(otvk, (\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_{u_{\max}}, \mathbf{y}_0, cmt_{\mathsf{ISIS}}, rsp_{\mathsf{ISIS}}, cmt_\vee, rsp_\vee), otsig) = 1$, and that $\mathcal{V}_{\vee\text{-}\mathsf{pBDD}}$ accepts the proof $(cmt_\vee, rsp_\vee)$ for statement $(\mathbf{B}_{otvk}, \boldsymbol{\tau}_v)_{v=1}^{u_{\max}}$ and challenge $ch_\vee = \mathsf{F}(\mathbf{B}_{otvk}\| (\boldsymbol{\tau}_v)_{v=1}^{u_{\max}} \|cmt_\vee\| (a_i)_{i=1}^{\ell} \| R\|M)$. He accepts the token if all of these conditions hold, otherwise he rejects it.

$\mathsf{Open}(msk, \tau, R, (a_i)_{i\in R}, M)$**:** If $\mathsf{VToken}(mpk, \tau, R, (a_i)_{i\in R}, M) = 0$, then the manager returns $\perp$. Otherwise, for each $v \in [u_{\max}]$, the opener computes $(\mathbf{s}'_v, \boldsymbol{\rho}'_v, \mathbf{x}'_{v,1}, \ldots, \mathbf{x}'_{v,\kappa}) \leftarrow \mathsf{CTInvert}((\mathbf{B}_0, \mathbf{B}_{1,otvk_1}, \ldots, \mathbf{B}_{\kappa,otvk_\kappa}), \mathbf{S}_0, (\boldsymbol{\tau}_{v,0}, \ldots, \boldsymbol{\tau}_{v,\kappa}))$. It returns the first v where $\mathsf{CTInvert}$ does not return $\perp$ and $\boldsymbol{\rho}'_v \leq (\#R + 1)\eta\sqrt{m+n}$, or it returns $\perp$ if no such v exists.

Theorem 6. *The above AAT+R scheme is anonymous in the random oracle model if* LWE *is hard for* $\Psi = D_{\mathbb{Z}^{m+n},\eta}$ *and if* $\mathcal{OTS}$ *is existentially unforgeable under one-time chosen-message attack.*

Proof (sketch). In the proof, we combine ideas from the security proof of [Pei09] and Theorem 4, so that we can answer opening queries adaptively, without losing anonymity for all users. The technique essentially uses the one-time signature scheme to reject mal-formed or "recycled" tokens to the opener. Furthermore, it allows inserting a challenge from the LWE problem into one of the encryption matrices $\mathbf{B}_{i,j}$ and still be able decrypt via a trapdoor for any one of the other lattices. This makes the proof somewhat more involved than for the AAT–R scheme in Theorem 4; we refer to Appendix F for details. □

Theorem 7. *The above AAT+R scheme is traceable in the random oracle model if* $\mathsf{SIS}(n, m+n, q, (2\ell_{max}+1)\eta\sqrt{m+n}+\widetilde{\mathcal{O}}(n^{1.5}))$ *is hard and the decision* LWE *problem with noise distribution* Ψ *is hard.*

Proof (Sketch). The proof idea is largely the same as for the unforgeability of the AAT–R scheme (Theorem 5), i.e., the simulator reveals its internally generated GPV signatures for issue queries, but uses "fake" signatures and simulated BDD proofs to respond to token generation queries. However, this is complicated by the fact that responses to the opening oracle have to be consistent with "recycled" BDD proofs and honest for all other inputs. We refer to Appendix G for the full proof. □

6 Extensions

Achieving Non-Frameability. For our AAT+R scheme, the group manager needs to be trusted not to frame users by generating tokens on their behalf and falsely hold the users responsible for the tokens. Many group signature schemes based on classical assumptions offer security against these type of attacks by offering *non-frameabilty* [BSZ05]. The GKV scheme, as of yet the only group signature scheme based on post-quantum assumptions, does not offer this property.

In this section we describe how we can obtain non-frameabilty for our construction. Due to space limitations, we only give a high-level description here. The main idea is to run a AAT+R and a AAT–R scheme in parallel. The schemes are then merged so that a token is only valid if it contains a valid token for both schemes. The AAT–R scheme ensures that users cannot be framed, while the AAT+R scheme ensures that tokens can be opened. We briefly sketch the necessary changes to merge the two schemes.

Each user u generates its own random matrix $\mathbf{B}_u$ and generates its own master key pair $(\mathbf{A}_u, \mathbf{T}_u)$ for the AAT–R scheme based on $\mathbf{B}_u$. She then issues a credential cred_u to herself with a single attribute with a fixed value. The matrices $\mathbf{A}_u, \mathbf{B}_u$ become part of the scheme's public key, while cred_u is the user's secret key. The manager then generates a public key $(\mathbf{A}, \mathbf{B}_0, \mathbf{B}_{1,0}, \mathbf{B}_{1,1}, \ldots, \mathbf{B}_{\kappa,0}, \mathbf{B}_{\kappa,1})$ and corresponding secret key $(\mathbf{T}, \mathbf{S}_0)$ for the AAT+R scheme. Attribute-containing credentials $\widehat{\mathit{cred}}_u$ are issued using the Issue algorithm of the AAT+R scheme.

The token generation and verification algorithms of both schemes are merged by combining the WI proofs for cred_u and $\widehat{\mathit{cred}}_u$ to prove that the signer knows short vectors $\boldsymbol{\alpha}, \hat{\boldsymbol{\alpha}}$ for the same index u, i.e., the language becomes

$$\mathcal{L}^{\mathsf{YES}}_{\mathsf{V\text{-}BDD}}(\gamma, \beta, u_{\max}) := \left\{ ((\mathbf{B}_v, \boldsymbol{\tau}_v, \mathbf{B}_{otvk}, \hat{\boldsymbol{\tau}}_v))_{v=1}^{u_{\max}} \mid \exists v \in [u_{\max}]\ \exists (\mathbf{s}_v, \hat{\mathbf{s}}_v) \in \mathbb{Z}_q^n \times \mathbb{Z}_q^n : \right.$$
$$\left. \left\| \boldsymbol{\tau}_v - \mathbf{B}_v^t \mathbf{s}_v \right\|_2 \leq \beta \quad \wedge \quad \left\| \hat{\boldsymbol{\tau}}_v - \hat{\mathbf{B}}_{otvk}^t \hat{\mathbf{s}}_v \right\|_2 \leq (\#R + 1)\beta \right\} .$$

Results for Group Signatures. While our goal was to construct anonymous attribute token schemes, our results directly imply better lattice-based group signature schemes. As mentioned earlier, a group signature scheme can be seen as a special case of a AAT+R scheme with a single attribute with a fixed value. From our AAT+R scheme in Section 5, we obtain the first group signature scheme based on lattices that enjoys full anonymity, i.e., anonymity that is preserved under an attack where the adversary has access to an opening oracle. This form of anonymity is standard for group signatures [BMW03] but not achieved by the GKV scheme. Compared to their scheme, ours also has the advantage of shorter manager keys: both the public and the secret key are linear in the number of group members in the GKV scheme, versus constant in ours. When constructing a group signature scheme from the non-frameable AAT+R scheme sketched above, one furthermore obtains the first group signature based on lattices offering non-frameability.

Results for Ring Signatures. A ring signature scheme [RST01] lets users anonymously sign messages in name of ad-hoc groups, composed by the user at the time of signing. Lattice-based ring signatures can be built by letting each user generate matrices $\mathbf{A}_u, \mathbf{B}_u, \mathbf{T}_u$ for our AAT–R scheme, self-issue a credential $cred_u$ with a single attribute with a fixed value, publish $\mathbf{A}_u, \mathbf{B}_u$ as a public key and keep $cred_u$ as a signing key. To create a signature for a group of users $U \subseteq [u_{\max}]$, the user essentially generates a token for our AAT–R scheme using the language

$$\mathcal{L}^{\mathsf{YES}}_{\mathsf{V\text{-}BDD}}(\gamma, \beta, U) \;:=\; \{((\mathbf{B}_v, \boldsymbol{\tau}_v))_{v \in U} \,|\exists v \in [u_{\max}]\, \exists \mathbf{s}_v \in \mathbb{Z}_q^n : \left\|\boldsymbol{\tau}_v - \mathbf{B}_v^t \mathbf{s}_v\right\|_2 \leq \beta\} \;.$$

7 Conclusion

We have provided the first full-blown lattice-based group signature scheme. In our scheme, the anonymity of honest users is also maintained after other users' signatures are opened and, in addition, it protect users against framing by a malicious group manager. Moreover, we have provided the first lattice-based anonymous attribute-based token systems (with and without anonymity revocation) as a major step towards anonymous credential systems. Extending these schemes to full-fledged anonymous credential systems seems possible but remains a challenging open problem.

References

[ACJT00] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *2000*, volume 1880 of *Lecture Notes in Computer Science*. Springer, 2000.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC*, pages 99–108. ACM, 1996.

[AP09] Joël Alwen and Chris Peikert. Generating shorter bases for hard random lattices. In Susanne Albers and Jean-Yves Marion, editors, *STACS*, volume 09001 of *Dagstuhl Seminar Proceedings*, pages 75–86. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2009.

[AST02] Giuseppe Ateniese, Dawn Xiaodong Song, and Gene Tsudik. Quasi-efficient revocation in group signatures. In *Financial Cryptography 2002*, Lecture Notes in Computer Science, pages 183–197. Springer, 2002.

[Bab86] László Babai. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2004.

[BCN$^+$10] Patrik Bichsel, Jan Camenisch, Gregory Neven, Nigel P. Smart, and Bogdan Warinschi. Get shorty via group signatures without encryption. In Juan A. Garay and Roberto De Prisco, editors, *Security and Cryptography for Networks – 2010*, volume 6280 of *Lecture Notes in Computer Science*, pages 381–398. Springer, 2010.

[BGLS03] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 416–432. Springer, 2003.

[BMW03] Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 614–629. Springer, 2003.

[BN06] Mihir Bellare and Gregory Neven. Multi-signatures in the plain public-key model and a general forking lemma. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security*, pages 390–399. ACM, 2006.

[BP10] Stefan Brands and Christian Paquin. U-prove cryptographic specification v1.0. `http://connect.microsoft.com/site642/Downloads/DownloadDetails.aspx?DownloadID=26953`, 2010.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS 1993*. ACM, 1993.

[Bra99] Stefan Brands. *Rethinking Public Key Infrastructure and Digital Certificates— Building in Privacy.* PhD thesis, Eindhoven Institute of Technology, 1999.

[BS04] Dan Boneh and Hovav Shacham. Group signatures with verifier-local revocation. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick McDaniel, editors, *ACM CCS 2004*, pages 168–177. ACM, 2004.

[BSZ05] Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of group signatures: The case of dynamic groups. volume 3376 of *Lecture Notes in Computer Science*, pages 136–153. Springer, 2005.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo G. Desmedt, editor, *Advances in Cryptology — CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer Verlag, 1994.

[Cha81] David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, February 1981.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Gilbert [Gil10], pages 523–552.

[CL01a] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. `http://eprint.iacr.org/2001`, 2001.

[CL01b] Jan Camenisch and Anna Lysyanskaya. Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation. In Birgit Pfitzmann, editor, *Advances in Cryptology — EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 93–118. Springer Verlag, 2001.

[CL04] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Matthew K. Franklin, editor, *Advances in Cryptology — CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 56–72. Springer Verlag, 2004.

[CvH91] David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *Advances in Cryptology — EUROCRYPT '91*, volume 547 of *Lecture Notes in Computer Science*, pages 257–265. Springer-Verlag, 1991.

[DP06] Cécile Delerablée and David Pointcheval. Dynamic fully anonymous short group signatures. In Phong Q. Nguyen, editor, *Progress in Cryptology - VIETCRYPT 06*, volume 4341 of *Lecture Notes in Computer Science*, pages 193–210. Springer, 2006.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.

[Gil10] Henri Gilbert, editor. *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*. Springer, 2010.

[GKV10] Dov Gordon, Jonathan Katz, and Vinod Vaikuntanathan. A group signature scheme from lattice assumptions. Available at `http://www.cs.umd.edu/~jkatz/` (June 28, 2010), 2010. To appear in ASIACRYPT 2010.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *STOC*, pages 197–206. ACM, 2008.

[IA11] Danish National IT and Telecom Agency. New digital security models. `http://digitaliser.dk/resource/896495`, 2011.

[Kha07] Dalia Khader. Attribute based group signatures. Cryptology ePrint Archive, Report 2007/159, version 20080112:115123, 2007.

[Kle00] Philip N. Klein. Finding the closest lattice vector when it's unusually close. In *SODA*, pages 937–941, 2000.

[KP98] Joe Kilian and Erez Petrank. Identity escrow. In Hugo Krawczyk, editor, *Advances in Cryptology — CRYPTO '98*, volume 1642 of *Lecture Notes in Computer Science*, pages 169–185, Berlin, 1998. Springer Verlag.

[Los10] Daniel Loss. Personal communication, 2010.

[LRSW99] Anna Lysyanskaya, Ron Rivest, Amit Sahai, and Stefan Wolf. Pseudonym systems. In Howard Heys and Carlisle Adams, editors, *Selected Areas in Cryptography*, volume 1758 of *Lecture Notes in Computer Science*. Springer Verlag, 1999.

[Lyu08a] Vadim Lyubashevsky. Lattice-based identification schemes secure under active attacks. In Ronald Cramer, editor, *Public Key Cryptography*, volume 4939 of *Lecture Notes in Computer Science*, pages 162–179. Springer, 2008.

[Lyu08b] Vadim Lyubashevsky. *Towards Practical Lattice-Based Cryptography.* PhD thesis, University of California, San Diego, 2008.

[Lyu09] Vadim Lyubashevsky. Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 598–616. Springer, 2009.

[MG02] Daniele Micciancio and Shafi Goldwasser. *Complexity of Lattice Problems: a cryptographic perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, Boston, Massachusetts, March 2002.

[MPR11] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures. Lecture Notes in Computer Science. Springer, 2011.

[MR07] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007.

[MV03] Daniele Micciancio and Salil P. Vadhan. Statistical zero-knowledge proofs with efficient provers: Lattice problems and more. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 282–298. Springer, 2003.

[MV10] Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In Michael Mitzenmacher and Leonard J. Schulman, editors, *STOC*, pages 351–358. ACM, 2010.

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *STOC*, pages 333–342. ACM, 2009.

[Pei10] Chris Peikert. An efficient and parallel gaussian sampler for lattices. In Tal Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 80–97. Springer, 2010.

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *J. Cryptology*, 13(3):361–396, 2000.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009.

[Reg10] Oded Regev. On the complexity of lattice problems with polynomial approximation factors. In *The LLL Algorithm*, Information Security and Cryptography. Springer, 2010.

[RIS10] RISEPTIS. Trust in the information society - a report of the advisory board. `http://www.think-trust.eu/general/news/finalreport.html`, 2010.

[RS09] Alon Rosen and Gil Segev. Chosen-ciphertext security via correlated products. In Omer Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 419–436. Springer, 2009.

[RST01] Ronald Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In *2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 552–565. Springer, 2001.

[Rüc10a] Markus Rückert. Adaptively secure identity-based identification from lattices without random oracles. In Juan A. Garay and Roberto De Prisco, editors, *SCN*, volume 6280 of *Lecture Notes in Computer Science*, pages 345–362. Springer, 2010.

[Rüc10b] Markus Rückert. Lattice-based blind signatures. In Masayuki Abe, editor, *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 413–430. Springer, 2010.

[SCPY08] Alfredo De Santis, Giovanni Di Crescenzo, Giuseppe Persiano, and Moti Yung. On monotone formula composition of perfect zero-knowledge languages. *SIAM J. Comput.*, 38(4):1300–1329, 2008.

[ST01] Adi Shamir and Yael Tauman. Improved online/offline signature schemes. In Joe Kilian, editor, *Advances in Cryptology — CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 355–367. Springer Verlag, 2001.

A Proof of Proposition 2

Proof (Proposition 2). Split matrices $\mathbf{A} = \mathbf{A}_1 \| \mathbf{A}_2$, $\mathbf{B} = \mathbf{B}_1 \| \mathbf{B}_2$, $\mathbf{C} = \mathbf{C}_1 \| \mathbf{C}_2$, and $\mathbf{D} = \mathbf{D}_1 \| \mathbf{D}_2$ as above. The matrices $\mathbf{A}_1, \mathbf{B}_1, \mathbf{B}_2, \mathbf{C}_1, \mathbf{C}_2, \mathbf{D}_1$ are uniformly random by definition or statistically close to uniform (via GPVGen). Moreover, we know that $\mathbf{A}_2$ is distributed statistically close to uniform subject to $\mathbf{A}_2 = -\mathbf{A}_1 \mathbf{B}_1^t (\mathbf{B}_2^{-1})^t$. Equivalently, we can write $\mathbf{B}_2 = -\mathbf{B}_1 \mathbf{A}_1^t (\mathbf{A}_2^{-1})^t$, which is exactly how $\mathbf{D}_2$ is constructed with respect to $\mathbf{D}_1$ and $\mathbf{C}$. As for the trapdoors, we argue that the $\mathbf{T}_{\mathbf{X}_1}$, for $\mathbf{X} \in \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$, are pairwise independent and follow the same distribution, namely the output distribution of $\mathsf{GPVGen}(1^n)$. The extensions $\mathbf{T}_{\mathbf{X}_2} = \mathbf{V}_{\mathbf{X}} \| \mathbf{I}_n$, for $\mathbf{X} \in \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$, are distributed uniformly random conditioned on the consistency requirement $\mathbf{X}_1 \mathbf{V}_{\mathbf{X}} \equiv -\mathbf{X}_2$ in $\mathsf{ExtBasis}$. Hence, both random variables are within negligible statistical distance. □

B Proof of Theorem 3

We introduce the notion of a single-signer aggregate signature scheme, which can be seen as a restricted form of aggregate signatures [BGLS03] where only signatures by the same signer can be aggregated. The syntax of the key generation algorithm and signing algorithm are the same as for standard signatures. The aggregation algorithm Agg takes as input a public key pk, a list of messages $M_1, \ldots, M_\ell$, and a list of signatures $\sigma_1, \ldots, \sigma_\ell$ of these messages under pk, and outputs an aggregate signature α. The aggregate verification algorithm $\mathsf{AVerify}$, on input pk, messages $M_1, \ldots, M_\ell$, and aggregate α, returns 1 if α is deemed valid for $M_1, \ldots, M_\ell$ under pk, and returns 0 if not.

Existential unforgeability is defined as for aggregate signatures: the adversary, given pk as input and access to a signing oracle, wins if it can output a forged aggregate α^* on messages

$M_1^*, \ldots, M_{\ell^*}^*$ so that $\mathsf{AVerify}(pk, \alpha^*, (M_1^*, \ldots, M_{\ell^*}^*)) = 1$ where at least one of the messages M_i^* was never queried to the signing oracle.

The scheme we propose is parameterized with a maximum number ℓ_{max} of signatures per aggregate. Key generation and signing are as for the GPV scheme, aggregation and verification are described below.

- $\mathsf{Agg}(pk, (\boldsymbol{\sigma}_i)_{i=1}^{\ell}, (M_i)_{i=1}^{\ell})$ outputs $\boldsymbol{\alpha} \leftarrow \sum_{i=1}^{\ell} \boldsymbol{\sigma}_i$.
- $\mathsf{Verify}(pk, (M_i)_{i=1}^{\ell}, \boldsymbol{\alpha})$ returns 1 if $\ell \leq \ell_{max}$, $0 < \|\boldsymbol{\alpha}\|_2 \leq \ell\eta\sqrt{m}$, and $\mathbf{A}\boldsymbol{\alpha} \equiv \sum_{i=1}^{\ell} \mathsf{H}(M_i)$, or returns 0 otherwise.

The scheme is correct due to the linearity of $\mathbf{A}$. Let $d^\infty := \eta\rho$ be the upper bound for the infinity norm of a single signature. Then, an ℓ-aggregate requires $m\log_2(\ell d^\infty)$ bits of storage, as opposed to $m\ell\log_2(d^\infty)$ for ℓ individual GPV signatures. The proof of the following theorem is given in Appendix A.

Theorem 8. *The above single-signer aggregate signature scheme is existentially unforgeable in the random oracle model if the* $\mathsf{SIS}(n, m, q, 2\ell_{max}\eta\sqrt{m})$ *problem is hard.*

Proof. Given a forger $\mathcal{A}$, we construct an algorithm $\mathcal{B}$ that solves the SIS problem as follows. On input $\mathbf{A}$, $\mathcal{B}$ runs $\mathcal{A}$ on input $\mathbf{A}$. Algorithm $\mathcal{B}$ keeps two associative arrays $\boldsymbol{\sigma}[\cdot]$ and $\mathbf{h}[\cdot]$. We assume without loss of generality that $\mathcal{A}$ never makes the same random oracle or signing query twice, and that it always queries $\mathsf{H}(M)$ before querying a signature on M or before using M in its forgery.

When $\mathcal{A}$ makes a random oracle query $\mathsf{H}(M)$, $\mathcal{B}$ chooses $\boldsymbol{\sigma}[M] \sim D_{\Lambda,\eta}$, computes $\mathbf{h}[M] \leftarrow \mathbf{A}\boldsymbol{\sigma}[M] \bmod q$, and returns $\mathbf{h}[M]$. When $\mathcal{A}$ queries the signing oracle on M, $\mathcal{B}$ returns $\boldsymbol{\sigma}[M]$.

Eventually, $\mathcal{A}$ outputs its forgery $\boldsymbol{\alpha}^*$ for messages $M_1^*, \ldots, M_{\ell^*}^*$ such that $\mathbf{A}\boldsymbol{\alpha}^* \equiv \sum_{i=1}^{\ell^*} \mathbf{h}[M_i^*]$ and $\|\boldsymbol{\alpha}^*\|_2 \leq \ell^*\eta\sqrt{m}$. If $\boldsymbol{\alpha} = \sum_{i=1}^{\ell^*} \boldsymbol{\sigma}[M_i^*]$, then we also have that $\mathbf{A}\boldsymbol{\alpha} \equiv \sum_{i=1}^{\ell^*} \mathbf{h}[M_i^*]$ and $\|\boldsymbol{\alpha}\|_2 \leq \ell^*\eta\sqrt{m}$. For $\mathbf{x} = \boldsymbol{\alpha}^* - \boldsymbol{\alpha}$ we therefore have that $\mathbf{A}\mathbf{x} \equiv \mathbf{0}$ and by the triangle inequality that $\|\mathbf{x}\|_2 \leq 2\ell^*\eta\sqrt{m}$. Moreover, there is at least one message M_i^* that $\mathcal{A}$ did not query to the signing oracle. Therefore, $\mathcal{A}$'s view is independent of $\boldsymbol{\sigma}[M_i^*]$, so with overwhelming probability $1 - 1/2^{\omega(n)}$, $\boldsymbol{\alpha}^* \neq \boldsymbol{\alpha}$, and hence $\mathbf{x} \neq \mathbf{0}$, so that $\mathbf{x}$ is a valid solution to the SIS problem. □

C Lyubashevsky's Identification Scheme

In this section, we present a generalization of Lyubashevsky's identification scheme [Lyu08a]. Let n, m, q be parameters as per Section 2 and let $\mathbf{A}$ be a matrix that is shared by all users. In the original, the prover possesses a secret key $\mathbf{x} \in \{0,1\}^m$ and an associated public key $\mathbf{y} \leftarrow \mathbf{A}\mathbf{x} \bmod q$. Using $\mathbf{x}$ she can compute a witness-indistinguishable proof of knowledge of a short vector $\mathbf{x}' \in \mathbb{Z}^m$ such that $\mathbf{A}\mathbf{x}' \equiv \mathbf{y}$ and $\|\mathbf{x}'\|_2 \leq 10(m+n)^{1.5}\eta\rho(m) - 2\eta\rho(m+n)\sqrt{m+n} = \widetilde{\mathcal{O}}(n^{1.5})$.

Our setting is different for two reasons. First, we work over $\mathbb{Z}^{m+n}$ instead of over $\mathbb{Z}^m$. Second, our secret keys $\mathbf{x}$ are distributed according to $D_{\mathbb{Z}^{m+n},\eta}$ with $\|\mathbf{x}\|_\infty \leq \eta\rho(m) =: d$. As a consequence, the entire protocol needs to be modified. However, we stay as close as possible to the original protocol and leave especially soundness and completeness errors unchanged. Hence, the proofs of witness-indistinguishability, soundness, and completeness stay conceptually the same. Refer to Figure 1 for the protocol details.

Notice that we did not attempt to optimize the scheme for efficiency, but it is easy to transfer it into the ideal lattice setting in analogy to [Lyu09,Rüc10a]. There, the efficiency improvements of [Lyu08b,Rüc10b] apply.

Now we briefly sketch the analysis of the scheme, focusing on the differences from the original construction.

$$\begin{array}{lcl} \mathsf{Comm}_{\mathsf{ISIS}} & \xrightarrow{cmt_{\mathsf{ISIS}}} & \\ & \xleftarrow{ch_{\mathsf{ISIS}}} & ch_{\mathsf{ISIS}} \leftarrow_{\$} \{0,1\}^t \\ \mathsf{Resp}_{\mathsf{ISIS}} & \xrightarrow{rsp_{\mathsf{ISIS}}} & \mathsf{Verify}_{\mathsf{ISIS}} \end{array}$$

```
$\mathsf{Comm}_{\mathsf{ISIS}}(\mathbf{A})$:
For $i = 1, \ldots, t$ do
    $\mathbf{r}_i \leftarrow_{\$} \{-5(m+n), \ldots, 5(m+n)\}^{m+n}$
    $\mathbf{s}_i \leftarrow \mathbf{A}\mathbf{r}_i \bmod q$
$cmt_{\mathsf{ISIS}} \leftarrow (\mathbf{s}_1, \ldots, \mathbf{s}_t)$ ; $st \leftarrow (\mathbf{r}_1, \ldots, \mathbf{r}_t)$

$\mathsf{Resp}_{\mathsf{ISIS}}(\mathbf{x}, st, ch_{\mathsf{ISIS}})$:
For $i = 1, \ldots, t$ do
    $\mathbf{z}_i \leftarrow \mathbf{r}_i + ch_{\mathsf{ISIS},i}\mathbf{x}$
    If $ch_{\mathsf{ISIS},i} = 1 \ \wedge\ \|\mathbf{z}_i\|_\infty \leq 5(m+n) - d$
    then $\mathbf{z}_i \leftarrow \perp$
$rsp_{\mathsf{ISIS}} \leftarrow (\mathbf{z}_1, \ldots, \mathbf{z}_t)$
```

```
$\mathsf{Verify}_{\mathsf{ISIS}}(\mathbf{A}, \mathbf{y}, cmt_{\mathsf{ISIS}}, ch_{\mathsf{ISIS}}, rsp_{\mathsf{ISIS}})$:
$d \leftarrow 0$
For $i = 1, \ldots, t$ do
    If $\|\mathbf{z}_i\|_2 \leq 5d(m+n)^{1.5} - d\sqrt{m+n}$
        $\wedge\ \mathbf{A}\mathbf{z}_i \equiv \mathbf{s}_i + ch_{\mathsf{ISIS},i}\mathbf{y}$
    then $d \leftarrow d + 1$
If $d \geq 0.65t$ then return 1 else return 0 .
```

Fig. 1. Generalization of Lyubashevsky's identification scheme with parameter $t = \omega(\log n)$, public key $(\mathbf{A}, \mathbf{y})$, and secret key $\mathbf{x}$.

Completeness. Observe that for all $\mathbf{z}_i \neq \perp$, we always have $\mathbf{A}\mathbf{z}_i \equiv \mathbf{A}(\mathbf{r}_i + ch_{\mathsf{ISIS},i}\mathbf{x}) = \mathbf{s}_i + ch_{\mathsf{ISIS},i}\mathbf{y}$ as desired. For reasonable $k = m + n$ and for $(\phi, A, B) = (5, d, 5kA)$, the following simple lemma from [Rüc10b] establishes $\text{Prob}[\, \|\mathbf{z}_i\|_\infty \geq 0.81]$ as in [Lyu08a].

Lemma 1. *Let $k = \Omega(n)$, $\mathbf{a}, \mathbf{b} \in \mathbb{Z}^k$ with* arbitrary *$\mathbf{a} \in \{\mathbf{v} \in \mathbb{Z}^k : \|\mathbf{v}\|_\infty \leq A\}$ and* random *$\mathbf{b} \leftarrow_{\$} \{\mathbf{v} \in \mathbb{Z}^k : \|\mathbf{v}\|_\infty \leq B\}$. Given $B \geq \phi k A$ for $\phi \in \mathbb{N}_{>0}$, we have $\underset{\mathbf{b}}{\text{Prob}}[\, \|\mathbf{a} + \mathbf{b}\|_\infty \leq B - A] > \frac{1}{e^{1/\phi}} - o(1)$.*

Since all $\mathbf{z}_i$ are independent, the Chernoff bound gives $\text{Prob}[\, d < 0.65t] = \text{Prob}[\, d < (0.81 - 0.16)t] \leq e^{-2t(0.16^2) < 2^{-t/14}}$. Hence, for $t = \omega(\log(n))$, we achieve a negligible completeness error.

Witness-indistinguishability First of all, we need that for essentially all secret keys $\mathbf{x}$ there is a second (valid) key $\mathbf{x}'$ such that $\mathbf{A}\mathbf{x} \equiv \mathbf{A}\mathbf{x}'$. A simple counting argument shows that all but at most q^n secret keys admit such a collision, but there are $(2d+1)^{m+n}$ possible secret keys. Hence, the probability of choosing a non-colliding key is $2^{-\Omega(n\log(n))}$.

Let $\mathbf{x}$ and $\mathbf{x}'$ be such a pair of colliding keys with public key $\mathbf{y}$. The particular choice the secret key is statistically hidden by $\mathbf{y}$ and the adversary's challenge $ch_{\mathsf{ISIS}} = (ch_{\mathsf{ISIS},i})_{i=1}^t$ is independent of that choice. The same holds for $cmt_{\mathsf{ISIS}} = (\mathbf{s}_i)_{i=}^t$; each $\mathbf{s}_i$ statistically hides the underlying randomness $\mathbf{r}_i$. However, the response $rsp_{\mathsf{ISIS}} = (\mathbf{z}_i)_{i=1}^t$ is sensitive to the secret key, as $\mathbf{z}_i \leftarrow \mathbf{r}_i + ch_{\mathsf{ISIS},i}\mathbf{x}$.

To show that the protocol is statistically witness-indistinguishable, we need to be able to switch from $\mathbf{x}$ to $\mathbf{x}'$ without changing the output behavior. To this end, we define $\mathbf{r}_i' \leftarrow \mathbf{z}_i - ch_{\mathsf{ISIS},i}\mathbf{x}'$ and $\mathbf{z}_i' \leftarrow \mathbf{r}_i' + ch_{\mathsf{ISIS},i}\mathbf{x}'$ for all i. Observe that $\mathbf{z}_i = \mathbf{z}_i'$ and that $\mathbf{A}\mathbf{r}_i' \equiv \mathbf{A}(\mathbf{z}_i - ch_{\mathsf{ISIS},i}\mathbf{x}') \equiv \mathbf{s}_i$. Moreover, $\mathbf{r}_i'$ is chosen from the correct set as $\|\mathbf{r}_i'\|_\infty \leq 5(m+n) - d + d = 5(m+n)$.

Soundness. We sketch how a short lattice vector is extracted from a cheating adversary. We explicitly exploit witness-indistinguishability to (honestly) answer the adversary's prover queries. Hence, we pick a secret key $\mathbf{x}$, run the adversary on input $(\mathbf{A}, \mathbf{y})$ with $\mathbf{y} \leftarrow \mathbf{A}\mathbf{x} \bmod q$, and answer all queries honestly. During the adversary's impersonation attempt, we choose a uniformly random challenge vector $ch_{\mathsf{ISIS}}^{(1)} = \left(ch_{\mathsf{ISIS},i}^{(1)}\right)_{i=1}^t$ and receive the response $rsp_{\mathsf{ISIS}}^{(1)} = \left(\mathbf{z}_i^{(1)}\right)_{i=1}^t$.

Then, we rewind the adversary to the challenge phase of the protocol, send a fresh set $ch_{\mathsf{ISIS}}^{(2)} = \left(ch_{\mathsf{ISIS},i}^{(2)}\right)_{i=1}^t$ uniformly at random, and receive $rsp_{\mathsf{ISIS}}^{(2)} = \left(\mathbf{z}_i^{(2)}\right)_{i=1}^t$. We need that there is an index j such that $ch_{\mathsf{ISIS},j}^{(1)} \neq ch_{\mathsf{ISIS},j}^{(2)}$ as well as $\mathbf{A}\mathbf{z}_j^{(1)} \equiv \mathbf{s}_j + ch_{\mathsf{ISIS},j}^{(1)}\mathbf{y}$ and $\mathbf{A}\mathbf{z}_j^{(2)} \equiv \mathbf{s}_j + ch_{\mathsf{ISIS},j}^{(2)}\mathbf{y}$. Let us assume that $ch_{\mathsf{ISIS},j}^{(1)} = 1$ and $ch_{\mathsf{ISIS},j}^{(2)} = 0$. Hence, we can extract $\mathbf{x}' \leftarrow \mathbf{z}_j^{(1)} - \mathbf{z}_j^{(2)}$ with norm at most $10d(m+n)^{1.5} - 2d\sqrt{m+n}$ and $\mathbf{A}\mathbf{x}' \equiv 0$ as desired.

Let us assume that the adversary has a non-negligible success probability ϵ. As shown in [Lyu08a, Theorem 13], such an index j exists with non-negligible probability at least $\epsilon^2 - 2^{-t/18}$

because with probability at least $1 - 2^{-t/18}$, the hamming distance of $ch^{(1)}_{\mathsf{ISIS}}$ and $ch^{(2)}_{\mathsf{ISIS}}$ is large enough to guarantee this "overlap" of both executions by the Chernoff bound.

D Full Proof of Theorem 4

We define a sequence of games with Game-1 being the anonymity experiment for $b = 0$ and Game-8 being the same experiment with $b = 1$. We prove the theorem by showing indistinguishability between each pair of subsequent games.

In Game-1, the experiment chooses $\mathbf{B} \leftarrow_{\$} \mathbb{Z}_q^{n\times(m+n)}$ and generates $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{OrthoSamp}(\mathbf{B})$. It then runs the adversary $\mathcal{A}$ on input $mpk = \mathbf{A}$, $msk = (\mathbf{A}, \mathbf{T})$, and common reference string $\mathbf{B}$. At the end of the first phase, $\mathcal{A}$ outputs user indices $u_0, u_1 \in [u_{\max}]$, credentials $cred_{u_0}, cred_{u_1}$, attribute values $(a_{u_0,i})_{i=1}^{\ell_0}, (a_{u_1,i})_{i=1}^{\ell_1}$, set $R \subseteq [\min(\ell_0, \ell_1)]$, and message M. If $\mathsf{VCred}(mpk, cred_b, (a_{u_b,i})_{i=1}^{\ell_b}) = 1$ for $b \in \{0, 1\}$ and $a_{u_0,i} = a_{u_1,i}$ for $i \in R$, then the experiment generates $\tau^* \leftarrow_{\$} \mathsf{GenToken}(mpk, cred_{u_0}, (a_{u_0,i})_{i=1}^{\ell_0}, R, M)$ and hands it to $\mathcal{A}$. All random oracle queries, including the internal queries spawned by generating τ^*, are responded with truly random values. The latter outputs a bit b' and wins the game if $b' = 0$.

In Game-2, the experiment generates $\boldsymbol{\tau}^*$ by choosing a random vector $\mathbf{b} \leftarrow_{\$} \mathbb{Z}_q^{m+n}$, setting $\mathbf{y} \leftarrow \mathbf{A}\mathbf{b} \bmod q$, computing $\mathbf{x} \leftarrow \mathsf{GPVInvert}(\mathbf{A}, \mathbf{T}, \mathbf{y}, \eta)$, computing a random (long) vector $\tilde{\boldsymbol{\alpha}}_{u_1}$ such that $\mathbf{A}\tilde{\boldsymbol{\alpha}}_{u_1} \equiv \sum_{i\in R} \mathsf{H}(u_1\|i\|a_{u_1,i})$, and setting $\boldsymbol{\tau}_{u_1} \leftarrow \mathbf{b} + \tilde{\boldsymbol{\alpha}}_{u_1} \bmod q$. Otherwise, the signature generation proceeds as prescribed by the $\mathsf{GenToken}$ algorithm using the vector $\mathbf{x}$ computed earlier, i.e., using the short vectors $\boldsymbol{\sigma}_{u_0,i}$ included in $cred_{u_0}$ to compute $\boldsymbol{\tau}_{u_0}$ and using long vectors $\tilde{\boldsymbol{\alpha}}_v$ for $v \in [u_{\max}] \setminus \{u_0, u_1\}$.

Claim. If the decision LWE problem with noise distribution Ψ is hard, then Game-1 and Game-2 are computationally indistinguishable.

Proof. Given an adversary $\mathcal{A}$ that distinguishes Game-1 from Game-2, we build a distinguisher $\mathcal{D}$ that, on input $\mathbf{B}, \mathbf{b}$, decides whether $\mathbf{b}$ is a random vector or was generated as $\mathbf{b} \leftarrow \mathbf{B}^t\mathbf{s}+\mathbf{e} \bmod q$ with $\mathbf{s} \leftarrow_{\$} \mathbb{Z}_q^n$ and $\mathbf{e} \sim \Psi$.

On inputs $\mathbf{B}, \mathbf{b}$, algorithm $\mathcal{D}$ runs $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{OrthoSamp}(\mathbf{B})$. It then runs $\mathcal{A}$ on inputs $\mathbf{A}, (\mathbf{A}, \mathbf{T})$ with common reference string $\mathbf{B}$. When $\mathcal{A}$ outputs u_0, u_1, $cred_{u_0}, cred_{u_1}$, $(a_{u_0,i})_{i=1}^{\ell_0}$, $(a_{u_1,i})_{i=1}^{\ell_1}$, R, and M, algorithm $\mathcal{D}$ generates the token τ^* as done in Game-2 using the vector $\mathbf{b}$. It is clear that when $\mathbf{b}$ is random, then $\mathcal{A}$'s view is identical to Game-2, while if $\mathbf{b}$ was generated as $\mathbf{B}^t\mathbf{s} + \mathbf{e}$, then it is identical to Game-1. □

Game-3 is identical to Game-2, except that the short vector $\boldsymbol{\alpha}_{u_1} \leftarrow \sum_{i\in R} \boldsymbol{\sigma}_{u_1,i}$ taken from $cred_{u_1}$ is used to generate $\boldsymbol{\tau}_{u_1} \leftarrow \mathbf{b}+\boldsymbol{\alpha}_{u_1} \bmod q$. Token generation remains unchanged otherwise. Since $\mathbf{b}$ is a uniformly random vector, Game-3 is information-theoretically indistinguishable from Game-2.

In Game-4, the challenge token is generated using a random $\mathbf{x} \sim \Psi$ and $\mathbf{y} \leftarrow \mathbf{A}\mathbf{x} \bmod q$. Moreover, $\boldsymbol{\tau}_{u_1}$ is computed as $\boldsymbol{\tau}_{u_1} \leftarrow \mathbf{B}^t\mathbf{s}_{u_1} + \boldsymbol{\alpha}_{u_1} + \mathbf{x} \bmod q$. Notice that now short vectors $\boldsymbol{\sigma}_{u_0}, \boldsymbol{\sigma}_{u_1}$ are being used for both of $\boldsymbol{\tau}_{u_0}, \boldsymbol{\tau} u_1$, but that $cmt_{\vee}, rsp_{\vee}$ are still generated with $(u_0, \mathbf{s}_{u_0})$ as a witness. Game-4 is computationally indistinguishable from Game-3 under the assumption that the LWE problem w.r.t. Ψ is hard by a similar argument as made for Game-1 and Game-2.

Game-5 is identical to Game-4, except that the BDD proof is generated as $(cmt_{\vee}, rsp_{\vee}) \leftarrow \mathcal{P}_{\mathsf{BDD}}((\mathbf{B}, \boldsymbol{\tau}_v)_{v=1}^{u_{\max}}, u_1, \mathbf{s}_{u_1})$. This game hop is indistinguishable by the witness-indistinguishability of $(cmt_{\vee}, rsp_{\vee})$.

The rest of the game hops are essentially the same as the previous ones, but in reverse order. In Game-6, a random vector $\mathbf{b} \leftarrow_{\$} \mathbb{Z}^{m+n}$ is chosen to set $\mathbf{y} \leftarrow \mathbf{A}\mathbf{b} \bmod q$ and $\mathbf{x} \leftarrow \mathsf{GPVInvert}(\mathbf{A}, \mathbf{T}, \mathbf{y}, \eta)$. Token generation proceeds as in Game-5, but $\boldsymbol{\tau}_{u_0}$ is generated as $\boldsymbol{\tau}_{u_0} \leftarrow \mathbf{b} + \boldsymbol{\alpha}_{u_0}$. This game hop is indistinguishable under the decisional LWE assumption. In Game-7, $\boldsymbol{\tau}_{u_0}$ is generated using a random long vector $\tilde{\boldsymbol{\alpha}}_{u_0}$ instead of the short vector $\boldsymbol{\alpha}_{u_0}$. The game hop is information-theoretically indistinguishable due to the randomness of $\mathbf{b}$. Finally, Game-8 is a real attack with $\boldsymbol{\tau}^*$ generated via cred_{u_1}. It is indistinguishable from Game-7 under the LWE assumption by a similar argument as made for Game-1 and Game-2.

E Full Proof of Theorem 5

We show how a traceability adversary $\mathcal{A}$ gives rise to a solver $\mathcal{B}$ for the SIS problem. We do so by converting $\mathcal{A}$ into an algorithm $\mathcal{C}$ that can be rewound in Bellare-Neven's generalization [BN06] of Pointcheval-Stern's forking lemma [PS00]. We then show that the forking algorithm $\mathcal{F}_{\mathcal{C}}$ can be converted into the SIS solver $\mathcal{B}$.

Consider algorithm $\mathcal{C}$ that on input $\mathbf{A} \in \mathbb{Z}_q^{n\times(m+n)}$ and random values $g_1, \dots, g_{q_\mathsf{G}} \in \{0,1\}^t$ runs $(\mathbf{B}, \mathbf{S}) \leftarrow \mathsf{OrthoSamp}(\mathbf{A})$. Algorithm $\mathcal{C}$ then runs $\mathcal{A}$ on input $\mathit{mpk} = \mathbf{A}$ and with common reference string $\mathbf{B}$, responding to its $\mathsf{H}(H_i)$ queries by selecting $\boldsymbol{\sigma}_i \sim D_{\mathbb{Z}^{m+n},\eta}$ and responding with $\mathbf{A}\boldsymbol{\sigma}_i \bmod q$, keeping a list $(H_i, \boldsymbol{\sigma}_i)$. When responding to $\mathcal{A}$'s i-th query $\mathsf{G}(G_i)$, the algorithm answers with g_i while maintaining a list of tuples (i, G_i, g_i). Queries $\mathsf{F}(F_i)$ are simply responded to using random strings. (Without loss of generality, we assume that $\mathcal{A}$ never makes the same random oracle query twice.) When $\mathcal{A}$ makes an initialization query for u with attributes $(a_i)_{i=1}^{\ell}$, $\mathcal{C}$ simulates queries $\mathsf{H}(u\|i\|a_i)$, causing the random oracle to be programmed as described above. When $\mathcal{A}$ makes an issue query for user u, it looks up tuples $(u\|i\|a_i, \boldsymbol{\sigma}_{u,i})$ in the list for the attributes a_i with which u was initialized and returns $\mathit{cred}_u = (u, (\boldsymbol{\sigma}_{u,i})_{i=1}^{\ell})$.

When $\mathcal{A}$ requests a token by user u, $\mathcal{C}$ chooses $\mathbf{x} \sim D_{\mathbb{Z}^{m+n},\eta}$ and computes $\mathbf{y}$, $\mathit{cmt}_\mathsf{ISIS}$, $\mathit{rsp}_\mathsf{ISIS}$ as in the real $\mathsf{GenToken}$ algorithm. It then generates $\boldsymbol{\tau}_v$ using a long vector $\tilde{\boldsymbol{\alpha}}_v$ for *all* $v \in [u_{\max}]$, i.e., it chooses a random $\tilde{\boldsymbol{\alpha}}_v \in \mathbb{Z}_q^{m+n}$ such that $\mathbf{A}\tilde{\boldsymbol{\alpha}}_v \equiv \sum_i \mathsf{H}(v\|i\|a_i)$ using Gauss elimination, chooses $\mathbf{s}_v \leftarrow_\$ \mathbb{Z}_q^n$ and computes $\boldsymbol{\tau}_v \leftarrow \mathbf{B}^t\mathbf{s}_v + \tilde{\boldsymbol{\alpha}}_v + \mathbf{x} \bmod q$. It then simulates a proof $(\mathit{cmt}_\vee, \mathit{ch}_\vee, \mathit{rsp}_\vee) \leftarrow \mathcal{S}_{\vee\text{-}\mathsf{pBDD}}((\mathbf{B}, \boldsymbol{\tau}_v)_{v=1}^{u_{\max}})$, programs random oracle F so that $\mathsf{F}(\mathit{cmt}_\vee) = \mathit{ch}_\vee$, and returns the token $\tau \leftarrow (\boldsymbol{\tau}_1, \dots, \boldsymbol{\tau}_{u_{\max}}, \mathbf{y}, \mathit{cmt}_\mathsf{ISIS}, \mathit{rsp}_\mathsf{ISIS}, \mathit{cmt}_\vee, \mathit{rsp}_\vee)$.

The environment that $\mathcal{C}$ provides to $\mathcal{A}$ is indistinguishable from a real attack environment. Namely, mpk and the common reference string are correctly distributed by Proposition 2 and credentials returned by the issuing oracle are correctly distributed as per the properties of $\mathsf{GPVInvert}$. The simulated tokens are computationally indistinguishable from the ones generated by $\mathsf{GenToken}$ if the decision LWE problem with noise distribution Ψ is hard by a similar argument as used in the claim showing indistinguishability between Game-1 and Game-2 in the proof of Theorem 4 above.

Eventually, $\mathcal{A}$ outputs its forgery $\tau^* = (\boldsymbol{\tau}_1^*, \dots, \boldsymbol{\tau}_{u_{\max}}^*, \mathbf{y}^*, \mathit{cmt}_\mathsf{ISIS}^*, \mathit{rsp}_\mathsf{ISIS}^*, \mathit{cmt}_\vee^*, \mathit{rsp}_\vee^*)$ on message M^* and attributes $(a_i^*)_{i\in R^*}$ for R^*. If the token is invalid, $\mathcal{C}$ returns $(0, \perp)$. Algorithm $\mathcal{C}$ then computes $(\mathbf{s}'_v, \boldsymbol{\rho}'_v) \leftarrow \mathsf{LWEInvert}(\mathbf{B}, \mathbf{S}, \boldsymbol{\tau}_v^*)$ for $v \in [u_{\max}]$ until it finds an index u^* for which $\|\boldsymbol{\rho}_{u^*}\|_2 \le (\#R^* + 1)\eta\sqrt{m+n}$. By the soundness of $\mathcal{P}_{\vee\text{-}\mathsf{pBDD}}$, at least one such index must exist. (Note that the adversary cannot have reused a simulated proof $\mathit{cmt}_\vee, \mathit{rsp}_\vee$ from one of the responses of the token generation oracle, since the forgery has to be on a different set of attributes or a different message than any of the requested tokens, and therefore $\mathit{ch}_\vee$ as generated through F will be a different random value.) Next, $\mathcal{C}$ looks up the index i^* of $\mathcal{A}$'s query $\mathsf{G}(\mathbf{y}^*\|\mathit{cmt}_\mathsf{ISIS}^*\|M^*) = \mathit{ch}_\mathsf{ISIS}^*$, which, without loss of generality, can be assumed to exist. It also looks up entries $(u^*\|i\|a_i^*, \boldsymbol{\sigma}_i)$ in the list associated to H for $i \in R^*$ and computes $\boldsymbol{\alpha} \leftarrow \sum_{i\in R^*} \boldsymbol{\sigma}_i$. Algorithm $\mathcal{C}$ returns $(i^*, (\boldsymbol{\alpha}, \boldsymbol{\rho}'_{u^*}, \mathbf{y}^*, \mathit{cmt}_\mathsf{ISIS}^*, \mathit{ch}_\mathsf{ISIS}^*, \mathit{rsp}_\mathsf{ISIS}^*))$.

The forking lemma of [BN06] says that if $\mathcal{C}$ returns (i^*, τ^*) with $i^* \neq 0$ with non-negligible probability, then the forking algorithm $\mathcal{F}_{\mathcal{C}}$ will with non-negligible probability return $(1, \tau^{(1)}, \tau^{(2)})$ based on two executions of $\mathcal{C}$ that are identical up to the i^*-th query to G, and where the responses to the i^*-th query in both executions are different.

Consider now the SIS solving algorithm $\mathcal{B}$ that, on input $\mathbf{A}$, runs $\mathcal{F}_{\mathcal{C}}(\mathbf{A})$ to obtain $(1, \tau^{(1)}, \tau^{(2)})$. Let $\tau^{(1)} = (\boldsymbol{\alpha}^{(1)}, \boldsymbol{\rho}^{(1)}, \mathbf{y}^{(1)}, \mathit{cmt}_\mathsf{ISIS}^{(1)}, \mathit{ch}_\mathsf{ISIS}^{(1)}, \mathit{rsp}_\mathsf{ISIS}^{(1)})$ and $\tau^{(2)} = (\boldsymbol{\alpha}^{(2)}, \boldsymbol{\rho}^{(2)}, \mathbf{y}^{(2)}, \mathit{cmt}_\mathsf{ISIS}^{(2)}, \mathit{ch}_\mathsf{ISIS}^{(2)}, \mathit{rsp}_\mathsf{ISIS}^{(2)})$. Since the two executions of $\mathcal{C}$ are identical up to the i^*-th query to G, the arguments to the i^*-th query to G are also identical in both executions, and hence we have that $\mathbf{y}^{(1)} = \mathbf{y}^{(2)} =: \mathbf{y}$ and $cmt^{(1)}_{\mathsf{ISIS}} = cmt^{(2)}_{\mathsf{ISIS}}$. Since the responses to the i^*-th query are different, we have that $ch^{(1)}_{\mathsf{ISIS}} \neq ch^{(2)}_{\mathsf{ISIS}}$. By the rewinding argument of Lemma 1 in Appendix C, which is a generalization of [Lyu08a][Theorem 13], we can extract a vector $\mathbf{x}' \leftarrow \mathsf{Ext}_{\mathsf{ISIS}}(cmt^{(1)}_{\mathsf{ISIS}}, ch^{(1)}_{\mathsf{ISIS}}, rsp^{(1)}_{\mathsf{ISIS}}, ch^{(2)}_{\mathsf{ISIS}}, rsp^{(2)}_{\mathsf{ISIS}})$ such that $\mathbf{Ax}' \equiv \mathbf{y}$ and $\|\mathbf{x}'\|_2 \leq \widetilde{\mathcal{O}}(n^{1.5})$. Algorithm $\mathcal{B}$ then computes $\boldsymbol{\alpha}^{(1)} \leftarrow \boldsymbol{\rho}^{(1)} - \mathbf{x}'$.

By the definition of a successful adversary, we know that at $\mathcal{A}$'s view must be independent from at least one of the vectors $\boldsymbol{\sigma}_{u^*,i}$ underlying random oracle queries $\mathsf{H}(u^*\|i\|a_i^*)$, either because user u^* was never corrupted, or because u^* was initialized with a different attribute a_i'. Remember that token generation queries are responded without using $\boldsymbol{\sigma}_{u^*,i}$, so these do not leak any information about $\boldsymbol{\sigma}_{u^*,i}$. Since also $\mathcal{P}_{\mathsf{ISIS}}$ is witness-indistinguishable, we can apply the same technique as in the proof of Theorem 3 and conclude that with overwhelming probability $\mathbf{z} = \boldsymbol{\alpha}^{(1)} - \boldsymbol{\alpha}$ is a non-zero vector with $\mathbf{Az} \equiv \mathbf{0}$, where $\boldsymbol{\alpha}$ is an appropriate aggregate of the $\boldsymbol{\sigma}_{u^*,i}$ vectors. Furthermore, we can conclude that $\|\mathbf{z}\|_2 \leq (\ell_{max}+1)\eta\sqrt{m+n} + \widetilde{\mathcal{O}}(n^{1.5}) + \ell_{max}\eta\sqrt{m+n} \leq (2\ell_{max}+1)\eta\ sqrtm+n + \widetilde{\mathcal{O}}(n^{1.5})$.

F Proof of Theorem 6

Proof. We define a sequence of games with the first game being the anonymity experiment for $b = 0$ and the last being the same experiment with $b = 1$. We prove the theorem by showing indistinguishability between each pair of subsequent games.

Game-1 is the real game with $b = 0$.

In Game-2, the experiment generates the key pair $otvk^*, otsk^*$ to be used in the challenge token τ^* at the very beginning of the game. If at any point during the game the opening oracle is queried on a token τ with $otvk = otvk^*$, then it returns $\perp$. It is straightforward to show that any difference between Game-1 and Game-2 gives rise to a forger for $\mathcal{OTS}$.

In Game-3, the component $\boldsymbol{\tau}^*_{u_1}$ in the challenge token is replaced with a random element $\mathbf{b} = [\mathbf{b}_0, \ldots, \mathbf{b}_\kappa]$ from $\mathbb{Z}_q^{(\kappa+1)(m+n)}$. The vector $\mathbf{y}^*_0$ is set to $\mathbf{y}^*_0 \leftarrow \mathbf{Ab}_0 - \sum \mathsf{H}(u_1\|i\|a_i)$, and the proof $(cmt^*_{\mathsf{ISIS}}, rsp^*_{\mathsf{ISIS}})$ is computed using $\mathbf{x}^* \leftarrow \mathsf{GPVInvert}(\mathbf{A}, \mathbf{T}, \mathbf{y}, \eta)$. Otherwise, τ^* is generated as in the real $\mathsf{GenToken}$ algorithm with u_1 as a signer.

Claim. If the decision LWE problem with noise distribution Ψ is hard, then Game-2 and Game-3 are computationally indistinguishable.

Proof. Given an adversary $\mathcal{A}$ that distinguishes Game-2 from Game-3, we build a distinguisher $\mathcal{D}$ that, on input matrix $\mathbf{B} \in \mathbb{Z}_q^{n\times(\kappa+1)(m+n)}$ and vector $\mathbf{b} \in \mathbb{Z}_q^{(\kappa+1)(m+n)}$, decides whether $\mathbf{b}$ is a random vector or was generated as $\mathbf{b} \leftarrow \mathbf{Bs} + \mathbf{e} \bmod q$ with $\mathbf{s} \leftarrow_\$ \mathbb{Z}_q^n$ and $\mathbf{e} \sim \Psi$.

On inputs $\mathbf{B}, \mathbf{b}$, algorithm $\mathcal{D}$ splits $\mathbf{B}$ in $\kappa+1$ blocks $[\mathbf{B}_0\|\ldots\|\mathbf{B}_\kappa]$ of dimension $n \times (m+n)$. It generates $(otvk^*, otsk^*) \leftarrow_\$ \mathsf{OTKeygen}(1^n)$. For all $i \in [\kappa]$ it sets $\mathbf{B}_{i,otvk_i^*} \leftarrow \mathbf{B}_i$ and generates $(\mathbf{B}_{i,1-otvk_i^*}, \mathbf{S}_{i,1-otvk_i^*}) \leftarrow_\$ \mathsf{GPVGen}(1^n)$. It also generates $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{OrthoSamp}(\mathbf{B}_0)$. It then runs $\mathcal{A}$ on input $mpk = (\mathbf{A}, \mathbf{B}_0, \mathbf{B}_{1,0}, \mathbf{B}_{1,1}, \ldots, \mathbf{B}_{\kappa,0}, \mathbf{B}_{\kappa,1})$. It responds to random oracle queries with random elements from appropriate domains, and responds to issuing queries via the real Issue algorithm using trapdoor $\mathbf{T}$.

It responds to opening queries for a token τ as follows. If the token is invalid or $otvk = otvk^*$, it returns $\perp$. Else, let i be such that $otvk_i \neq otvk_i^*$. For each $v \in [u_{\max}]$ it computes $(\mathbf{s}_v, \mathbf{x}_{v,i}) \leftarrow \mathsf{LWEInvert}(\mathbf{B}_{i,otvk_i}, \mathbf{S}_{i,otvk_i}, \boldsymbol{\tau}_{v,i})$. For each $v \in [u_{\max}]$ and $j \in [\kappa]\backslash\{i\}$ compute $\mathbf{x}_{v,j} \leftarrow \boldsymbol{\tau}_{v,j} - \mathbf{B}^t_{j,otvk_j}\mathbf{s}_v \bmod q$ and $\boldsymbol{\rho}_v \leftarrow \boldsymbol{\tau}_{v,0} - \mathbf{B}_0^t\mathbf{s}_v \bmod q$. If $\|\mathbf{e}_{v,j}\|_2 \leq \eta\sqrt{m+n}$ for all $v \in [u_{\max}]$ and all $j \in \{0, \ldots, \kappa\}$ then it returns the smallest index v for which $\|\boldsymbol{\rho}_v\|_2 \leq (\#R+1)\eta\sqrt{m+n}$, which by the soundness of $\mathcal{P}_{\mathsf{V\text{-}pBDD}}$ must exist. Else, it returns $\perp$.

The challenge token τ^* is generated as in Game-3 with vector $\mathbf{b}$ taking the place of $\boldsymbol{\tau}^*_{u_1}$.

One can verify that $\mathcal{A}$'s environment is exactly as in Game-2 if $\mathbf{b}$ was generated as $\mathbf{b} \leftarrow \mathbf{Bs}+\mathbf{e} \bmod q$, and exactly as in Game-3 if it is random. Any adversary $\mathcal{A}$ distinguishing between both games will therefore allow $\mathcal{D}$ to win the LWE game. □

The rest of the proof is very similar to that of Theorem 4. Namely, in Game-4, $\boldsymbol{\tau}_{u_1,0}$ is generated as $\mathbf{b}_0 + \boldsymbol{\alpha}_{u_1} \bmod q$ for a short vector $\boldsymbol{\alpha}_{u_1}$. By the randomness of $\mathbf{b}$, Game-4 is information-theoretically indistinguishable from Game-3. In Game-5, $\boldsymbol{\tau}_{u_1}$ is generated as $\boldsymbol{\tau}_{u_1,0} \leftarrow \mathbf{B}_0^t \mathbf{s}_{u_1} + \boldsymbol{\alpha} u_1 + \mathbf{x}_0 \bmod q$ using a random $\mathbf{x}_0 \sim \Psi$ and $\mathbf{y}_0 \leftarrow \mathbf{A}\mathbf{x}_0 \bmod q$. It is indistinguishable from Game-6 by a similar argument as made in the above claim. In Game-6, $\mathbf{s}_{u_1}$ is used a witness in the generation of $cmt^*_\vee, rsp^*_\vee$; the witness-indistinguishability of $\mathcal{P}_{\vee\text{-pBDD}}$ guarantees the indistinguishability of this game hop. In the subsequent games, the same steps are taken in reverse order as for the previous games, gradually moving to having τ^* entirely generated by u_1, as done in the proof of Theorem refthm:aat:anonymity. □

G Proof of Theorem 7

Proof. We show how a traceability adversary $\mathcal{A}$ gives rise to a solver $\mathcal{B}$ for the SIS problem. We do so by converting $\mathcal{A}$ into an algorithm $\mathcal{C}$ that can be rewound in Bellare-Neven's generalization [BN06] of Pointcheval-Stern's forking lemma [PS00]. We then show that the forking algorithm $\mathcal{F}_\mathcal{C}$ can be converted into the SIS solver $\mathcal{B}$.

Consider algorithm $\mathcal{C}$ that on input $\mathbf{A} \in \mathbb{Z}_q^{n\times(m+n)}$ and random values $g_1, \ldots, g_{q_\mathsf{G}} \in \{0,1\}^t$ runs $(\mathbf{B}_0, \mathbf{S}_0) \leftarrow \mathsf{OrthoSamp}(\mathbf{A})$ and chooses uniform matrices $\mathbf{B}_{i,b}$ for $i \in [\kappa]$ and $b \in \{0,1\}$. Algorithm $\mathcal{C}$ then runs $\mathcal{A}$ on input $mpk = (\mathbf{A}, \mathbf{B}_0, \mathbf{B}_{1,0}, \mathbf{B}_{1,1}, \ldots, \mathbf{B}_{\kappa,0}, \mathbf{B}_{\kappa,1})$, responding to its $\mathsf{H}(H_i)$ queries by selecting $\boldsymbol{\sigma}_i \sim D_{\mathbb{Z}^{m+n},\eta}$ and responding with $\mathbf{A}\boldsymbol{\sigma}_i \bmod q$, keeping a list $(H_i, \boldsymbol{\sigma}_i)$. When responding to $\mathcal{A}$'s i-th query $\mathsf{G}(G_i)$, the algorithm answers with g_i while maintaining a list of tuples (i, G_i, g_i). Queries $\mathsf{F}(F_i)$ are simply responded to using random strings. (Without loss of generality, we assume that $\mathcal{A}$ never makes the same random oracle query twice.) When $\mathcal{A}$ makes an initialization query for u with attributes $(a_i)_{i=1}^{\ell}$, $\mathcal{C}$ simulates queries $\mathsf{H}(u\|i\|a_i)$, causing the random oracle to be programmed as described above. When $\mathcal{A}$ makes an issue query for user u, it looks up tuples $(u\|i\|a_i, \boldsymbol{\sigma}_{u,i})$ in the list for the attributes a_i with which u was initialized and returns $cred_u = (u, (\boldsymbol{\sigma}_{u,i})_{i=1}^{\ell})$.

When $\mathcal{A}$ requests a token by user u, $\mathcal{C}$ chooses $\mathbf{x}_0 \sim D_{\mathbb{Z}^{m+n},\eta}$ and computes $\mathbf{y}_0, cmt_\mathsf{ISIS}, rsp_\mathsf{ISIS}$ as in the real $\mathsf{GenToken}$ algorithm. It then generates $\boldsymbol{\tau}_{v,0}$ using a long vector $\tilde{\boldsymbol{\alpha}}_v$ for *all* $v \in [u_{\max}]$, i.e., it chooses a random $\tilde{\boldsymbol{\alpha}}_v \in \mathbb{Z}_q^{m+n}$ such that $\mathbf{A}\tilde{\boldsymbol{\alpha}}_v \equiv \sum_i \mathsf{H}(v\|i\|a_i)$ using Gauss elimination, chooses $\mathbf{s}_v \leftarrow_\$ \mathbb{Z}_q^n$ and computes $\boldsymbol{\tau}_{v,0} \leftarrow \mathbf{B}_0^t \mathbf{s}_v + \tilde{\boldsymbol{\alpha}}_v + \mathbf{x}_0 \bmod q$. It also generates $(otvk, otsk) \leftarrow \mathsf{OTKeygen}(1^n)$, chooses $\mathbf{x}_{v,i} \sim \Psi$ and computes $\boldsymbol{\tau}_{v,i} \leftarrow \mathbf{B}_{i,otvk_i}^t \mathbf{s}_v + \mathbf{x}_{v,i} \bmod q$ as in the real token generation algorithm. It then simulates a proof $(cmt_\vee, ch_\vee, rsp_\vee) \leftarrow \mathcal{S}_{\vee\text{-pBDD}}((\mathbf{B}, \boldsymbol{\tau}_v)_{v=1}^{u_{\max}})$, by programming the random oracle F so that $\mathsf{F}(\mathbf{B}_{otvk}\|\,(\boldsymbol{\tau}_v)_{v=1}^{u_{\max}}\,\|cmt_\vee\|\,(a_i)_{i=1}^{\ell}\,\|R\|M) = ch_\vee$. Finally, she computes the one-time signature $otsig$ using $otsk$ and returns the token $\tau \leftarrow (\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_{u_{\max}}, \mathbf{y}_0, cmt_\mathsf{ISIS}, rsp_\mathsf{ISIS}, cmt_\vee, rsp_\vee, otvk, otsig)$.

When $\mathcal{A}$ submits an opening query $(\tau, R, (a_i)_{i\in R}, M)$, algorithm $\mathcal{C}$ first verifies τ using VToken. If it is invalid, $\mathcal{C}$ returns $\perp$. If the non-interactive proof cmt, rsp in τ was recycled from a previous response to a token generation request, then $\mathcal{C}$ returns the user index u of that token generation request. Otherwise, it returns the result of the real Open algorithm using $\mathbf{S}_0$.

The environment that $\mathcal{C}$ provides to $\mathcal{A}$ is indistinguishable from a real attack environment. Namely, $\mathbf{A}$ and $\mathbf{B}_0$ are correctly distributed by Proposition 2 and credentials returned by the issuing oracle are correctly distributed as per the properties of $\mathsf{GPVInvert}$. The simulated tokens are computationally indistinguishable from generated by $\mathsf{GenToken}$ if the decision LWE problem with noise distribution Ψ is hard by a similar argument as used in the claim showing indistinguishability between Game-1 and Game-2 in the proof of Theorem 4. The responses to opening queries are correctly distributed since the only responses that differ from those of the real Open algorithm are tokens involving a simulated proof (cmt, rsp), for which the user index for which they were created is returned.

Eventually, $\mathcal{A}$ outputs its forgery $\tau^* = (\boldsymbol{\tau}_1^*, \ldots, \boldsymbol{\tau}_{u_{\max}}^*, \mathbf{y}_0^*, cmt^*_\mathsf{ISIS}, rsp^*_\mathsf{ISIS}, cmt^*_\vee, rsp^*_\vee, otvk^*, otsig^*)$ on message M^* and attributes $(a_i^*)_{i\in R^*}$ for R^*. If the token is invalid, $\mathcal{C}$ returns $(0, \perp)$.

If the non-interactive proof $cmt_\vee^*, rsp_\vee^*$ was recycled from a previously generated token, then this cannot be a valid forgery, since the same vectors $\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_{u_{\max}}$ will always open to the same user u, and since the revealed attributes $(a_i^*)_{i \in R^*}$ and the message M^* must also be the same as when the token was generated as these are included in the random oracle argument of $ch_\vee = \mathsf{F}(\mathbf{B}_{otvk} \| \, (\boldsymbol{\tau}_v)_{v=1}^{u_{\max}} \, \| cmt_\vee \| \, (a_i)_{i=1}^{\ell} \, \| R \| M), rsp_\vee)$. Otherwise, let $u^* = \mathsf{Open}(msk, \tau^*, R^*, (a_i^*)_{i \in R^*}, M^*)$. Since the token is valid and $cmt_\vee, rsp_\vee$ is not a recycled simulated proof, by the soundness of $\mathcal{P}_{\vee\text{-}\mathsf{pBDD}}$ we have that $u^* \neq \perp$. Algorithm $\mathcal{C}$ then computes $(\mathbf{s}'_{u^*}, \boldsymbol{\rho}'_{u^*}) \leftarrow \mathsf{LWEInvert}(\mathbf{B}_0, \mathbf{S}_0, \boldsymbol{\tau}^*_{u^*,0})$. Next, $\mathcal{C}$ looks up the index i^* of $\mathcal{A}$'s query $\mathsf{G}(\mathbf{y}^* \| cmt^*_{\mathsf{ISIS}} \| M^*) = ch^*_{\mathsf{ISIS}}$, which, without loss of generality, can be assumed to exist. It also looks up entries $(u^* \| i \| \, a_i^*, \boldsymbol{\sigma}_i)$ in the list associated to H for $i \in R^*$ and computes $\boldsymbol{\alpha} \leftarrow \sum_{i \in R^*} \boldsymbol{\sigma}_i$. Algorithm $\mathcal{C}$ returns $(i^*, (\boldsymbol{\alpha}, \boldsymbol{\rho}'_{u^*}, \mathbf{y}^*, cmt^*_{\mathsf{ISIS}}, ch^*_{\mathsf{ISIS}}, rsp^*_{\mathsf{ISIS}}))$.

The forking lemma of [BN06] says that if $\mathcal{C}$ returns (i^*, τ^*) with $i^* \neq 0$ with non-negligible probability, then the forking algorithm $\mathcal{F}_{\mathcal{C}}$ will with non-negligible probability return $(1, \tau^{(1)}, \tau^{(2)})$ based on two executions of $\mathcal{C}$ that are identical up to the i^*-th query to G, and where the responses to the i^*-th query in both executions are different.

Consider now the SIS solving algorithm $\mathcal{B}$ that, on input $\mathbf{A}$, runs $\mathcal{F}_{\mathcal{C}}(\mathbf{A})$ to obtain $(1, \tau^{(1)}, \tau^{(2)})$. Let $\tau^{(1)} = (\boldsymbol{\alpha}^{(1)}, \boldsymbol{\rho}^{(1)}, \mathbf{y}^{(1)}, cmt^{(1)}_{\mathsf{ISIS}}, ch^{(1)}_{\mathsf{ISIS}}, rsp^{(1)}_{\mathsf{ISIS}})$ and $\tau^{(2)} = (\boldsymbol{\alpha}^{(2)}, \boldsymbol{\rho}^{(2)}, \mathbf{y}^{(2)}, cmt^{(2)}_{\mathsf{ISIS}}, ch^{(2)}_{\mathsf{ISIS}}, rsp^{(2)}_{\mathsf{ISIS}})$. Since the two executions of $\mathcal{C}$ are identical up to the i^*-th query to G, the arguments to the i^*-th query to G are also identical in both executions, and hence we have that $\mathbf{y}^{(1)} = \mathbf{y}^{(2)} =: \mathbf{y}$ and $cmt^{(1)}_{\mathsf{ISIS}} = cmt^{(2)}_{\mathsf{ISIS}}$. Since the responses to the i^*-th query are different, we have that $ch^{(1)}_{\mathsf{ISIS}} \neq ch^{(2)}_{\mathsf{ISIS}}$. By the rewinding argument of Lemma 1 in Appendix C, which is a generalization of [Lyu08a][Theorem 13], we can extract a vector $\mathbf{x}' \leftarrow \mathsf{Ext}_{\mathsf{ISIS}}(cmt^{(1)}_{\mathsf{ISIS}}, ch^{(1)}_{\mathsf{ISIS}}, rsp^{(1)}_{\mathsf{ISIS}}, ch^{(2)}_{\mathsf{ISIS}}, rsp^{(2)}_{\mathsf{ISIS}})$ such that $\mathbf{A}\mathbf{x}' \equiv \mathbf{y}$ and $\|\mathbf{x}'\|_2 \leq \widetilde{\mathcal{O}}(n^{1.5})$. Algorithm $\mathcal{B}$ then computes $\boldsymbol{\alpha}' \leftarrow \boldsymbol{\rho}^{(1)} - \mathbf{x}'$.

By the definition of a successful adversary, we know that at $\mathcal{A}$'s view must be independent from at least one of the vectors $\boldsymbol{\sigma}_{u^*,i}$ underlying random oracle queries $\mathsf{H}(u^* \| i \| \, a_i^*)$, either because user u^* was never corrupted, or because u^* was initialized with a different attribute a'_i. Remember that token generation queries are responded without using $\boldsymbol{\sigma}_{u^*,i}$, so these do not leak any information about $\boldsymbol{\sigma}_{u^*,i}$. Since also $\mathcal{P}_{\mathsf{ISIS}}$ is witness-indistinguishable, we can apply the same technique as in the proof of Theorem 3 and conclude that with overwhelming probability $\mathbf{z} = \boldsymbol{\alpha}^{(1)} - \boldsymbol{\alpha}$ is a non-zero vector with $\mathbf{A}\mathbf{z} \equiv \mathbf{0}$ and $\|\mathbf{z}\|_2 \leq (\ell_{max} + 1)\eta\sqrt{m+n} + \widetilde{\mathcal{O}}(n^{1.5}) + \ell_{max}\eta\sqrt{m} \leq (2\ell_{max} + 1)\eta\sqrt{m+n} + \widetilde{\mathcal{O}}(n^{1.5})$. □