# Better Zero-Knowledge Proofs for Lattice Encryption and Their Application to Group Signatures*

Fabrice Benhamouda[1], Jan Camenisch[2], Stephan Krenn[2], Vadim Lyubashevsky[3,1], Gregory Neven[2]

[1] Département d'Informatique, École Normale Supérieure, Paris, France
fabrice.ben.hamouda@ens.fr, lyubash@di.ens.fr
[2] IBM Research Zurich – Rüschlikon, Switzerland
{jca,skr,nev}@zurich.ibm.com
[3] INRIA France

**Abstract.** Lattice problems are an attractive basis for cryptographic systems because they seem to offer better security than discrete logarithm and factoring based problems. Efficient lattice-based constructions are known for signature and encryption schemes. However, the constructions known for more sophisticated schemes such as group signatures are still far from being practical. In this paper we make a number of steps towards efficient lattice-based constructions of more complex cryptographic protocols. First, we provide a more efficient way to prove knowledge of plaintexts for lattice-based encryption schemes. We then show how our new protocol can be combined with a proof of knowledge for Pedersen commitments in order to prove that the committed value is the same as the encrypted one. Finally, we make use of this to construct a new group signature scheme that is a "hybrid" in the sense that privacy holds under a lattice-based assumption while security is discrete-logarithm-based.

**Keywords:** Verifiable Encryption, Group Signatures, Zero-Knowledge Proofs for Lattices.

## 1 Introduction

There has been a remarkable increase of research in the field of lattice-based cryptography over the past few years. This renewed attention is largely due to a number of exciting results showing how cryptographic primitives such as fully homomorphic encryption [21] and multi-linear maps [20] can be built from lattices, while no such instantiations are known based on more traditional problems such as factoring or discrete logarithms. Lattice problems are also attractive to build standard primitives such as encryption and signature schemes, however, because of their strong security properties. In particular, their worst-case to average-case reductions as well as their apparent resistance against quantum computers set them apart from traditional cryptographic assumptions such as factoring or computing discrete logarithms, in particular in situations that require security many years or even decades into the future.

* The research leading to these results has received partial funding from the European Commission under the Seventh Framework Programme (CryptoCloud #339563, PERCY #321310, FutureID #318424) and from the French ANR-13-JS02-0003 CLE Project.

Long-term integrity requirements, e.g., for digital signatures, can usually be fulfilled by re-signing documents when new, more secure signature schemes are proposed. The same approach does not work, however, for privacy requirements, e.g., for encryption or commitment schemes, because the adversary may capture ciphertexts or commitments now and store them until efficient attacks on the schemes are found.

Several lattice-based encryption schemes have been proposed in the literature, e.g., [22, 24, 30, 35], but many of their applications in more complex primitives require efficient zero-knowledge proofs of the encrypted plaintext. Examples include optimistic fair exchange [2], non-interactive zero-knowledge proofs [34], multiparty computation [18], and group signatures [12]. In this paper, we present a more efficient zero-knowledge proof for lattice-based encryption schemes. We then combine it with a non-lattice-based signature scheme to build a group signature scheme with privacy under lattice assumptions in the random-oracle model.

## 1.1 Improved Proofs of Plaintext Knowledge for Lattice Schemes

In a zero-knowledge proof of plaintext knowledge, the encryptor wants to prove in zero-knowledge that the ciphertext is of the correct form and that he knows the message. Efficient constructions of these primitives are known based on number-theoretic hardness assumptions such as discrete log, strong RSA, etc.

Encryptions in lattice-based schemes generally have the form $t = Ae \bmod q$, where $A$ is some public matrix and $e$ is the unique vector with small coefficients that satisfies the equation (in this general example, we are lumping the message with $e$). A proof that $t$ is a valid ciphertext (and also a proof of plaintext knowledge), therefore, involves proving that one knows a short $e$ such that $Ae = t$. It is currently known how to accomplish this task in two ways. The first uses a "Stern-type" protocol [37] in which every run has soundness error $2/3$ [26]. It does not seem possible to improve this protocol since some steps in it are inherently combinatorial and non-algebraic.

A second possible approach is to use the Fiat-Shamir approach for lattices using rejection sampling introduced in [27, 29]. But while the latter leads to fairly efficient Fiat-Shamir signatures, there are some barriers to obtaining a proof of knowledge. What one is able to extract from a prover are short vectors $r', z'$ such that $Ae' = tc$ for some integer $c$, which implies that $Ae'c^{-1} = t$. Unfortunately, this does not imply that $e'c^{-1}$ is short unless $c = \pm 1$. This is the main way in which lattice-based Fiat-Shamir proofs differ from traditional schemes like the discrete-log based Schnorr protocol. In the latter, it is enough to extract any discrete log, whereas in lattice protocols, one must extract a *short* vector. Thus, the obvious approach at Fiat-Shamir proofs of knowledge for lattices (i.e., using binary challenge vectors) also leads to protocols with soundness error $1/2$.

Things do not improve for lattice-based proofs of knowledge even if one considers ideal lattices. Even if $A$ and $e$ are a matrix and a vector of polynomials in the ring $\mathbb{Z}_q[X]/(X^n + 1)$, and $c$ is now a polynomial (as in the Ring-LWE based Fiat-Shamir schemes in [28, 29]), then one can again extract only a short $e'$ such that $Ae'c^{-1} = t$. In this case, not only is $c^{-1}$ not necessarily short, but it does not even necessarily exist (since the polynomial $X^n + 1$ can factor into up to $n$ terms). At this point, we are not aware of any techniques that reduce the soundness error of protocols that prove plaintext knowledge of lattice encryptions, except by (parallel) repetition.

In a recent work, Damgård et al. [18] gave an improved *amortized* proof of plaintext knowledge for LWE encryptions. Their protocol allows one to prove knowledge of $O(k)$ plaintexts (where $k$ is the security parameter) in essentially the same time as just one plaintext. The ideas behind their protocol seem do not reduce the time requirement for proving just one plaintext, nor do they apply to Ring-LWE based encryption schemes. In particular, Ring-LWE based schemes are able to encrypt $O(n)$ plaintext bits into one (or two) polynomial, which is often all that is needed. Yet, the techniques in [18] do not seem to be helpful here. The reason is that the challenge matrix required in [18] needs to be of a particular form and cannot simply be a ring element in $\mathbb{Z}_q[X]/(X^n+1)$.

In this paper, we show that one can reduce the soundness error of lattice-based zero-knowledge proofs of knowledge for ciphertext validity from $1/2$ to $1/(2n)$, which in practice decreases the number of required iterations of the protocol by a factor of approximately 10. Interestingly, our techniques only work for ideal lattices, and we do not know how to adapt them to general ones. The key observation is that, when working over the ring $\mathbb{Z}[X]/(X^n+1)$, the quantity $2/(X^i - X^j)$ for all $0 \leq i \neq j < n$ is a polynomial with coefficients in $\{-1, 0, 1\}$, cf. Section 3.1.

This immediately allows us to prove that, given $A$ and $t$, we know a vector of short polynomials $e$ such that $Ae = 2t$. While this is not quite the same as proving that $Ae = t$, it is good enough for most applications, since it still allows us to prove knowledge of the plaintext. This result immediately gives improvements in all schemes that require such a proof of knowledge for Ring-LWE based encryption schemes such as the ring-version of the "dual" encryption scheme [22], the "two element" scheme of Lyubashevsky et al. [30], and NTRU [24,35].

## 1.2 Linking Lattice-based and Classical Primitives

A main step in applying our new proof protocol to construct a "hybrid" group signature scheme is to prove that two primitives, one based on classical cryptography and the other one on lattices, are committing to the same message (and that the prover knows that message). In our application, we will use the perfectly hiding Pedersen commitment scheme as the classical primitive, and a Ring-LWE encryption scheme as the lattice-based primitive.

While the Pedersen commitment and the lattice-based encryption scheme work over different rings, we show that we can still perform operations "in parallel" on the two. For example, if the message is $\mu_0, \mu_1, \ldots, \mu_{n-1}$, then it is encrypted in Ring-LWE schemes by encrypting the polynomial $\mu = \mu_0 + \mu_1 X + \ldots + \mu_{n-1} X^{n-1}$, and each $\mu_i$ is committed to individually using a Pedersen commitment. We will then want to prove that a Ring-LWE encryption of $\mu$ encrypts the same thing as $n$ Pedersen commitments of the $\mu_i$'s. Even though the two computations are performed over different rings, we show that by mimicking polynomial multiplications over a polynomial ring by appropriate additions and multiplications of coefficients in exponents, we can use our previously mentioned proof of plaintext knowledge to both prove knowledge of $\mu$ and show that the Pedersen commitments are committing to the coefficients of the same $\mu$. One reason enabling such a proof is that the terms dealing with $\mu$ (and $\mu_i$) in the proof of knowledge are done "over the integers"—that is, no modular reduction needs to be performed on these terms. We describe this protocol in detail in Section 4.

### 1.3 Applications to Group Signatures and Credentials

Group signatures [12] are schemes that allow members of a group to sign messages on behalf of the group without revealing their identity. In case of a dispute, the group manager can lift a signer's anonymity and reveal his identity. Currently known group signatures based on lattice assumptions are mainly proofs of concepts, rather than practically useful schemes. The schemes by Gordon et al. [23] and Camenisch et al. [10] have signature size linear in the number of group members. The scheme due to Laguillaumie et al. [25] performs much better asymptotically with signature sizes logarithmic in the number of group members, but, as the authors admit, instantiating it with practical parameters would lead to very large keys and signatures. This is in contrast to classical number-theoretic solutions, where both the key and the signature size are constant for arbitrarily many group members.

One can argue that the privacy requirement for group signatures is a concern that is more long-term than traceability (i.e., unforgeability), because when traceability turns out to be broken, verifiers can simply stop accepting signatures for the broken scheme. When privacy is broken, however, an adversary can suddenly reveal the signers behind all previous signatures. Users may only be willing to use a group signature scheme if their anonymity is guaranteed for, say, fifty years in the future. It therefore makes sense to provide anonymity under lattice-based assumptions, while this is less crucial for traceability.

Following this observation, we propose a "hybrid" group signature scheme, where unforgeability holds under classical assumptions, while privacy is proved under lattice-based ones. This allows us to combine the flexible tools that are available in the classical framework with the strong privacy guarantees of lattice problems. Our group signature scheme has keys and signatures of size logarithmic in the number of group members; for practical choices of parameters and realistic numbers of group members, the sizes will even be independent of the number of users. Furthermore, by basing our scheme on ring-LWE and not standard LWE, we partially solve an open problem stated in [25].

Our construction follows a variant of a generic approach that we believe is folklore, as it underlies several direct constructions in the literature [3, 8] and was described explicitly by Chase and Lysyanskaya [11]. When joining the group, a user obtains a certificate from the group manager that is a signature on his identity under the group manager's public key. To sign a message, the user now encrypts his identity under the manager's public encryption key, and then issues a signature proof of knowledge that he possesses a valid signature on the encrypted plaintext. Our construction follows a variant of this general paradigm, with some modifications to better fit the specifics of our proof of plaintext knowledge for lattice encryption. To the best of our knowledge, however, the construction was never proved secure, so our proof can be seen as a contribution of independent interest.

## 2 Preliminaries

In this section, we informally introduce several notions. Formal definitions and proofs can be found in the full version.

## 2.1 Notation

We denote algorithms by sans-serif letters such as $\mathsf{A}, \mathsf{B}$. If $\mathcal{S}$ is a set, we write $s \xleftarrow{\$} \mathcal{S}$ to denote that $s$ was drawn uniformly at random from $\mathcal{S}$. Similarly, we write $y \xleftarrow{\$} \mathsf{A}(x)$ if $y$ was computed by a randomized algorithm $\mathsf{A}$ on input $x$, and $d \xleftarrow{\$} D$ for a probability distribution $D$, if $d$ was drawn according to $D$. When we make the random coins $\rho$ of $\mathsf{A}$ explicit, we write $y \leftarrow \mathsf{A}(x; \rho)$.

We write $\Pr[\mathcal{E} : \Omega]$ to denote the probability of event $\mathcal{E}$ over the probability space $\Omega$. For instance, $\Pr[x = y : x, y \xleftarrow{\$} D]$ denotes the probability that $x = y$ if $x, y$ were drawn according to a distribution $D$.

We identify the vectors $(a_0, \ldots, a_{n-1})$ with the polynomial $a_0 + a_1 X + \cdots + a_{n-1} X^{n-1}$. If $v$ is a vector, we denote by $\|v\|$ its Euclidean norm, by $\|v\|_\infty$ its infinity norm, and by $v_{\ll l}$ an anti-cyclic shift of a vector $v$ by $l$ positions, corresponding to a multiplication by $X^l$ in $R_q = \mathbb{Z}_q[X]/(X^n + 1)$. That is, $v_{\ll l} = (v_0, \ldots, v_{n-1})_{\ll l} = (-v_{n-l}, \ldots, -v_{n-1}, v_0, \ldots, v_{n-l-1})$.

Throughout the paper, $\lambda$ denotes the main security parameter and $\varepsilon$ denotes the empty string.

## 2.2 Commitment Schemes and Pedersen Commitments

Informally, a commitment scheme is a tuple $(\mathsf{CSetup}, \mathsf{Commit}, \mathsf{COpen})$, where $\mathsf{CSetup}$ generates commitment parameters, which are then used to commit to a message $m$ using $\mathsf{Commit}$. A commitment $cmt$ can then be verified using $\mathsf{COpen}$. Informally, a commitment scheme needs to be binding and hiding. The former means that no $cmt$ can be opened to two different messages, while the latter guarantees that $cmt$ does not leak any information about the contained $m$.

The following commitment scheme was introduced by Pedersen [32]. Let be given a family of prime order groups $\{\mathbb{G}(\lambda)\}_{\lambda \in \mathbb{N}}$ such that the discrete logarithm problem is hard in $\mathbb{G}(\lambda)$ for security parameter $\lambda$, and let $\tilde{q} = \tilde{q}(\lambda)$ be the order of $\mathbb{G} = \mathbb{G}(\lambda)$.

To avoid confusion, all elements with order $\tilde{q}$ are denoted with a tilde in the following. To ease the presentation of our main result, we will write the group $\mathbb{G}(\lambda)$ additively.

$\mathsf{CSetup}$**.** This algorithm chooses $\tilde{h} \xleftarrow{\$} \mathbb{G}$, $\tilde{g} \xleftarrow{\$} \langle \tilde{h} \rangle$, and outputs $cpars = (\tilde{g}, \tilde{h})$.

$\mathsf{Commit}$**.** To commit to a message $m \in \mathcal{M} = \mathbb{Z}_{\tilde{q}}$, this algorithm first chooses $r \xleftarrow{\$} \mathbb{Z}_{\tilde{q}}$. It then outputs the pair $(\widetilde{cmt}, o) = (m\tilde{g} + r\tilde{h}, r)$.

$\mathsf{COpen}$**.** Given a commitment $\widetilde{cmt}$, an opening $o$, a public key $cpars$ and a message $m$, this algorithm outputs `accept` if and only if $\widetilde{cmt} \stackrel{?}{=} m\tilde{g} + o\tilde{h}$.

**Theorem 2.1.** *Under the discrete logarithm assumption for $\mathbb{G}$, the given commitment scheme is perfectly hiding and computationally binding.*

## 2.3 Semantically Secure Encryption and NTRU

A semantically secure (or IND-CPA secure) encryption scheme is a tuple $(\mathsf{EncKG}, \mathsf{Enc}, \mathsf{Dec})$ of algorithms, where $\mathsf{EncKG}$ generates public/private key pair, $\mathsf{Enc}$ can be used to encrypt a message $m$ under the public key, and the message can be recovered from the ciphertext by $\mathsf{Dec}$ using the secret key. Informally, while $\mathsf{Dec}(\mathsf{Enc}(m)) = m$ should always hold, only knowing the ciphertext and the public key should not leak any information about the contained message.

In this paper we present improved zero-knowledge proofs of plaintext knowledge for lattice-based encryption schemes, and show how to link messages being encrypted by these schemes to Pedersen commitments. Our improved proof of knowledge protocol will work for any Ring-LWE based scheme where the basic encryption operation consists of taking public key polynomial(s) $a_i$ and computing the ciphertext(s) $b_i = a_i s + e_i$ where $s$ and $e_i$ are polynomials with small norms. Examples of such schemes include the ring-version of the "dual" encryption scheme [22], the "two element" scheme of Lyubashevsky et al. [30], and the NTRU encryption scheme [24, 35].

In this paper we will for simplicity only be working over the rings $R = \mathbb{Z}[X]/(X^n + 1)$ and $R_q = R/qR$, for some prime $q$. Also for simplicity, we will use NTRU as our encryption scheme because its ciphertext has only one element and is therefore simpler to describe in protocols. The NTRU scheme was first proposed by Hoffstein et al. [24], and we will be using a modification of it due to Stehlé and Steinfeld [35].

**Definition 2.2.** *The* discrete normal distribution *on $\mathbb{Z}^m$ centered at $v$ with standard deviation $\sigma$ is defined by the density function $D^m_{v,\sigma}(x) = \rho^m_{v,\sigma}(x)/\rho_\sigma(\mathbb{Z}^m)$, with $\rho^m_{v,\sigma}(x) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^m e^{-\frac{\|x-v\|^2}{2\sigma^2}}$ being the continuous normal distribution on $\mathbb{R}^m$ and $\rho_\sigma(\mathbb{Z}^m) = \sum_{z\in\mathbb{Z}^m} \rho^m_{0,\sigma}(z)$ being the scaling factor required to obtain a probability distribution. When $v = 0$, we also write $D^m_\sigma = D^m_{0,\sigma}$.*

We will sometimes write $u \xleftarrow{\$} D_{v,\sigma}$ instead of $u \xleftarrow{\$} D^n_{v,\sigma}$ for a polynomial $u \in R_q$ if there is no risk of confusion.

In the following, let $p$ be a prime less than $q$ and $\sigma, \alpha \in \mathbb{R}$.

**Message space.** The message space $\mathcal{M}$ is any subset of $\{y \in R : \|y\|_\infty < p\}$.

**KeyGen.** Sample $f', g$ from $D_\sigma$, set $f = pf' + 1$, and resample, if $f \bmod q$ or $g \bmod q$ are not invertible. Output the public key $h = pg/f$ and the secret key $f$. Note here that $h$ is invertible.

**Encrypt.** To encrypt a message $m \in \mathcal{M}$, set $s, e \xleftarrow{\$} D_\alpha$ and return the ciphertext $y = hs + pe + m \in R_q$.

**Decrypt.** To decrypt $y$ with secret key $f$, compute $y' = fy \in R_q$ and output $m' = y' \bmod p$.

If the value of $\sigma$ is large enough (approximately $\tilde{\mathcal{O}}(n\sqrt{q})$), then $g/f$ is uniformly random in $R_q$ [35], and the security of the above scheme is based on the Ring-LWE problem. For smaller values of $\sigma$, however, the scheme is more efficient and can be based on the assumption that $h = g/f$ is indistinguishable from uniform. This type of assumption, while not based on any worst-case lattice problem, has been around since the introduction of the original NTRU scheme over fifteen years ago. Our protocol works for either instantiation.

To obtain group signatures, we will need our encryption scheme to additionally be a commitment scheme. In other words, there should not be more than one way to obtain the same ciphertext. For the NTRU encryption scheme, this will require that we work over a modulus $q$ such that the polynomial $X^n + 1$ splits into two irreducible polynomials of degree $n/2$, which can be shown to be always the case when $n$ is a power of 2 and $q = 3 \bmod 8$ [36, Lemma 3].

**Lemma 2.3.** *Suppose that $q = 3 \bmod 8$ and let $\#S$,$\#E$, and $\#\mathcal{M}$ be the domain sizes of the parameters $s, e$, and $m$ in the ciphertext $y = hs + pe + m$. Additionally suppose that for all $m \in \mathcal{M}$, $\|m\|_\infty < p/2$. Then the probability that for a random $h$, there exists a ciphertext that can be obtained in two ways is at most $\frac{(2\#\mathcal{M}+1)\cdot(2\#S+1)\cdot(2\#E+1)}{q^{n/2}}$.*

Note that the above lemma applies to NTRU public keys $h$ that are uniformly random. If $h = pg/f$ is not random, then the ability to come up with two plaintexts for the same ciphertext would constitute a distinguisher for the assumed pseudorandomness of $h$.

## 2.4 Rejection Sampling

For a protocol to be zero-knowledge, the prover's responses must not depend on its secret inputs.However, in our protocols, the prover's response will be from a discrete normal distribution which is shifted depending on the secret key. To correct for this, we employ rejection sampling [28, 29], where a potential response is only output with a certain probability, and otherwise the protocol is aborted.

Informally, the following theorem states that for sufficiently large $\sigma$ the rejection sampling procedure outputs results that are independent of the secret. The technique only requires a constant number of iterations before a value is output, and furthermore the output is also statistically close for every secret $v$ with norm at most $T$. For concrete parameters we refer to the original work of Lyubashevsky [29, Theorem 4.6].

**Theorem 2.4.** *Let $V$ be a subset of $\mathbb{Z}^\ell$ in which all elements have norms less than $T$, and let $H$ be a probability distribution over $V$. Then, for any constant $M$, there exists a $\sigma = \tilde{\Theta}(T)$ such that the output distributions of the following algorithms $\mathsf{A}, \mathsf{F}$ are statistically close:*

$$\mathsf{A}: v \xleftarrow{\$} H;\ z \xleftarrow{\$} D^\ell_{v,\sigma};\ \textit{output } (z, v) \textit{ with probability } \min\left(D^\ell_\sigma(z)/(MD^\ell_{v,\sigma}(z)), 1\right)$$

$$\mathsf{F}: v \xleftarrow{\$} H;\ z \xleftarrow{\$} D^\ell_{0,\sigma};\ \textit{output } (z, v) \textit{ with probability } 1/M$$

*The probability that $\mathsf{A}$ outputs something is exponentially close to that of $\mathsf{F}$, i.e., $1/M$.*

## 2.5 Zero-Knowledge Proofs and $\Sigma'$-Protocols

On a high level, a zero-knowledge proof of knowledge (ZKPoK) is a two party protocol between a *prover* and a *verifier*, which allows the former to convince the latter that it knows some secret piece of information, without revealing anything about the secret apart from what the claim itself already reveals. For a formal definition we refer to Bellare and Goldreich [4].

A language $\mathcal{L} \subseteq \{0,1\}^*$ has witness relationship $R \subseteq \{0,1\}^* \times \{0,1\}^*$ if $x \in \mathcal{L} \Leftrightarrow \exists (x, w) \in R$. We call $w$ a witness for $x \in \mathcal{L}$. The ZKPoKs constructed in this paper will be instantiations of the following definition, which is a straightforward generalization of $\Sigma$-protocols [13, 15]:

**Definition 2.5.** *Let $(\mathsf{P}, \mathsf{V})$ be a two-party protocol, where $\mathsf{V}$ is PPT, and let $\mathcal{L}, \mathcal{L}' \subseteq \{0,1\}^*$ be languages with witness relations $\mathcal{R}, \mathcal{R}'$ such that $\mathcal{R} \subseteq \mathcal{R}'$. Then $(\mathsf{P}, \mathsf{V})$ is called a* $\Sigma'$-protocol *for $\mathcal{L}, \mathcal{L}'$ with completeness error $\alpha$, challenge set $\mathcal{C}$, public input $x$ and private input $w$, if and only if it satisfies the following conditions:*

- Three-move form*: The protocol is of the following form: The prover* $\mathsf{P}$*, on input* $(x, w)$*, computes a commitment* $\mathtt{t}$ *and sends it to* $\mathsf{V}$*. The verifier* $\mathsf{V}$*, on input* $x$*, then draws a challenge* $\mathtt{c} \xleftarrow{\$} \mathcal{C}$ *and sends it to* $\mathsf{P}$*. The prover sends a response* $\mathtt{s}$ *to the verifier. Depending on the protocol transcript* $(\mathtt{t}, \mathtt{c}, \mathtt{s})$*, the verifier finally accepts or rejects the proof. The protocol transcript* $(\mathtt{t}, \mathtt{c}, \mathtt{s})$ *is called* accepting*, if the verifier accepts the protocol run.*
- Completeness*: Whenever* $(x, w) \in \mathcal{R}$*, the verifier* $\mathsf{V}$ *accepts with probability at least* $1 - \alpha$*.*
- Special soundness*: There exists a PPT algorithm* $\mathsf{E}$ *(the* knowledge extractor*) which takes two accepting transcripts* $(\mathtt{t}, \mathtt{c}', \mathtt{s}'), (t, \mathtt{c}'', \mathtt{s}'')$ *satisfying* $\mathtt{c}' \neq \mathtt{c}''$ *as inputs, and outputs* $w'$ *such that* $(x, w') \in \mathcal{R}'$*.*
- Special honest-verifier zero-knowledge (HVZK)*: There exists a PPT algorithm* $\mathsf{S}$ *(the* simulator*) taking* $x \in \mathcal{L}$ *and* $\mathtt{c} \in \mathcal{C}$ *as inputs, that outputs* $(\mathtt{t}, \mathtt{s})$ *so that the triple* $(\mathtt{t}, \mathtt{c}, \mathtt{s})$ *is indistinguishable from an accepting protocol transcript generated by a real protocol run.*

This definition differs from the standard definition of $\Sigma$-protocols in two ways. First, we allow the honest prover to fail in at most an $\alpha$-fraction of all protocol runs, whereas the standard definition requires perfect completeness, i.e., $\alpha = 0$. However, this relaxation is crucial in our construction that is based on rejection sampling [28,29], where the honest prover sometimes has to abort the protocol to achieve zero-knowledge. Second, we introduce a second language $\mathcal{L}'$ with witness relation $\mathcal{R}' \supseteq \mathcal{R}$, such that provers knowing a witness in $\mathcal{R}$ are guaranteed privacy, but the verifier is only ensured that the prover knows a witness for $\mathcal{R}'$. This has already been used in [1] and informally also in, e.g., [16, 19]. If the *soundness gap* between $\mathcal{R}$ and $\mathcal{R}'$ is sufficiently small, the implied security guarantees are often enough for higher-level applications. Note that the original definition of $\Sigma$-protocols is the special case that $\alpha = 0$ and $R = R'$.

We want to stress that previous results showing that a $\Sigma$-protocol is always also an honest-verifier ZKPoK with knowledge error $1/|\mathcal{C}|$ directly carry over to the modified definition whenever $1 - \alpha > 1/|\mathcal{C}|$. Zero-knowledge against arbitrary verifiers can be achieved by applying standard techniques such as Damgård et al. [14, 17].

Finally, it is a well known result that negligible knowledge and completeness errors in $\lambda$ can be achieved, e.g., by running the protocol $\lambda$ times in parallel and accepting if and only if at least $\lambda(1 - \alpha)/2$ transcripts were valid, if there exists a constant $c$ such that $(1 - \alpha)/2 > 1/|\mathcal{C}| + c$ .

Some of the $\Sigma'$-protocols presented in this paper will further satisfy the following useful properties:

- *Quasi-unique responses*: No PPT adversary $\mathsf{A}$ can output $(y, \mathtt{t}, \mathtt{c}, \mathtt{s}, \mathtt{s}')$ with $\mathtt{s} \neq \mathtt{s}'$ such that $\mathsf{V}(y, \mathtt{t}, \mathtt{c}, \mathtt{s}) = \mathsf{V}(y, \mathtt{t}, \mathtt{c}, \mathtt{s}') = \mathtt{accept}$.
- *High-entropy commitments*: For all $(y, w) \in \mathcal{R}$ and for all $\mathtt{t}$, the probability that an honestly generated commitment by $\mathsf{P}$ takes on the value $\mathtt{t}$ is negligible.

# 3 Proving Knowledge of Ring-LWE Secrets

In the following we show how to efficiently prove knowledge of short $2s, 2e$ such that $2y = 2as + 2e$. This basic protocol can easily be adapted for proving more complex relations including more than one public image or more than two secret witnesses. Before

presenting the protocol, we prove a technical lemma that is at the heart of the knowledge extractor thereof.

## 3.1 A Technical Lemma

The following lemma guarantees that certain binomials in $\mathbb{Z}[X]/(X^n+1)$ can be inverted, and their inverses have only small coefficients.

**Lemma 3.1.** *Let $n$ be a power of 2 and let $0 < i, j < 2n-1$. Then $2(X^i - X^j)^{-1} \bmod (X^n+1)$ only has coefficients in $\{-1, 0, 1\}$.*

*Proof.* Without loss of generality, assume that $j > i$. Using that $X^n = -1 \bmod (X^n+1)$, we have that $2(X^i - X^j)^{-1} = -2X^{n-i}(1-X^{j-i})^{-1}$. It is therefore sufficient to prove the claim for $i = 0$ only.

Now remark that, for every $k \geq 1$ it holds that: $(1-X^j)(1+X^j+X^{2j}+\ldots+X^{(k-1)j}) = 1 - X^{kj}$.

Let us write $j = 2^{j'} j''$, with $j''$ a positive odd integer and $0 \leq j' \leq \log_2(n)$, and let us choose $k = 2^{\log_2(n)-j'}$ (recall that $n$ is a power of 2). We then have $jk = nj''$, and $X^{kj} = (-1)^{j''} = -1 \bmod (X^n+1)$, hence $1 - X^{kj} = 2 \bmod (X^n+1)$. Therefore, we have

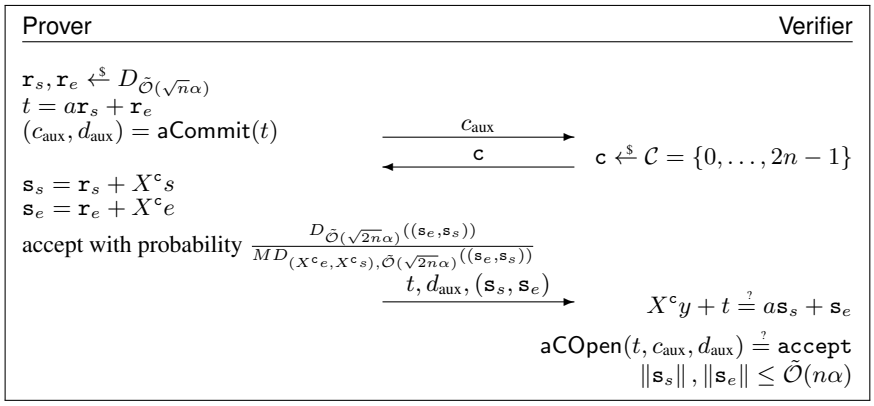
$$\begin{aligned} 2(1-X^j)^{-1} &= 1 + X^j + X^{2j} + \ldots + X^{(k-1)j} \bmod (X^n+1) \\ &= 1 \pm X^{j \bmod n} \pm X^{2j \bmod n} \pm \ldots \pm X^{(k-1)j \bmod n} \bmod (X^n+1)\,. \end{aligned}$$

Finally, in this equation, no two exponents are equal, since otherwise that would mean that $n$ divides $jk'$ with $1 \leq k' < k$, which is impossible by definition of $k$. □

## 3.2 The Protocol

We next present our basic protocol. Let therefore be $y = as + e$, where the LWE-secrets $s, e \xleftarrow{\$} D_\alpha$ are chosen from a discrete Gaussian distribution with standard deviation $\alpha$. Protocol 3.2 now allows a prover to convince a verifier that it knows $s'$ and $e'$ such that $2y = 2as' + 2e'$ with $2s'$ and $2e'$ being short (after reduction modulo $q$), i.e., the verifier is ensured that the prover knows short secrets for twice the public input. Here, by short we mean the following: An honest prover will always be able to convince the verifier whenever $\|s\|, \|e\| \leq \tilde{\mathcal{O}}(\sqrt{n}\alpha)$, which is the case with overwhelming probability if they were generated honestly. On the other hand, the verifier is guaranteed that the prover knows LWE-secrets with norm at most $\tilde{\mathcal{O}}(n^2\alpha)$. This soundness gap on the size of the witnesses is akin to those in, e.g., [1, 16].

To be able to simulate aborts when proving the zero-knowledge property of the protocol, we must not send the prover's first message in the plain, but commit to it and later open it in the last round of the $\Sigma'$-protocol. We therefore make use of an auxiliary commitment scheme $(\mathsf{aCSetup}, \mathsf{aCommit}, \mathsf{aCOpen})$, and assume that honestly generated commitment parameters are given as common input to both parties. We do not make any assumptions on the auxiliary commitment scheme. However, if it is computationally binding, the resulting protocol is only sound under the respective assumption, and similarly if it is computationally hiding. For simplicity, the reader may just think of the scheme as a random oracle.




**Protocol 3.2:** Proof of knowledge of LWE-secrets $s, e$ such that $y = as + e$.

**Theorem 3.3.** *Protocol 3.2 is an HVZK $\Sigma'$-protocol for the following relations:*

$$\mathcal{R} = \{((a, y), (s, e)) : y = as + e \quad \wedge \quad \|s\|, \|e\| \leq \tilde{\mathcal{O}}(\sqrt{n}\alpha)\}$$
$$\mathcal{R}' = \{((a, y), (s, e)) : 2y = 2as + 2e \quad \wedge \quad \|2s\|, \|2e\| \leq \tilde{\mathcal{O}}(n^2\alpha)\}$$

*where $2s$ and $2e$ are reduced modulo $q$. The protocol has a knowledge error of $1/(2n)$, a completeness error of $1 - 1/M$, and high-entropy commitments.*

We remark that in Protocol 3.2, the rejection sampling is applied on the whole vector $(\mathbf{s}_e, \mathbf{s}_s)$ instead of applying it twice on $\mathbf{s}_e$ and on $\mathbf{s}_s$. This yields better parameters ($M$ or the "$\sigma = \tilde{\mathcal{O}}(T)$" in Theorem 2.4) by a factor of about $\sqrt{2}$, because of the use of the Euclidean norm.

*Proof.* We need to prove the properties from Definition 2.5.
*Completeness.* First note that by Theorem 2.4, the prover will respond with probability $1/M$. If the prover does not abort, we have that:

$$a\mathbf{s}_s + \mathbf{s}_e = a(\mathbf{r}_s + X^{\mathsf{c}} s) + (\mathbf{r}_e + X^{\mathsf{c}} e) = X^{\mathsf{c}}(as + e) + (a\mathbf{r}_s + \mathbf{r}_e) = X^{\mathsf{c}} y + \mathbf{t}\,.$$

For the norms we have that $\|\mathbf{s}_s\| \leq \|\mathbf{r}_s\| + \|s\| \leq \tilde{\mathcal{O}}(n\alpha)$ with overwhelming probability, as the standard deviations of $\mathbf{r}_s$ is $\tilde{\mathcal{O}}(\sqrt{n}\alpha)$, and similarly for $\mathbf{s}_e$.
*Honest-verifier zero-knowledge.* Given a challenge value $\mathsf{c}$, the simulator outputs the tuple $(\mathsf{aCommit}(0), \mathsf{c}, \perp)$ with probability $1 - 1/M$. With probability $1/M$, the simulator $\mathsf{S}$ proceeds as follows: It chooses $\mathbf{s}_s, \mathbf{s}_e \xleftarrow{\$} D_{\tilde{\mathcal{O}}(\sqrt{n}\alpha)}$, and computes $\mathbf{t} = a\mathbf{s}_s + \mathbf{s}_e - X^{\mathsf{c}} y$, and $(c_{\text{aux}}, d_{\text{aux}}) \xleftarrow{\$} \mathsf{aCommit}(\mathbf{t})$. Finally, $\mathsf{S}$ outputs $(c_{\text{aux}}, \mathsf{c}, (\mathbf{t}, d_{\text{aux}}, (\mathbf{s}_s, \mathbf{s}_e)))$.

It follows from Theorem 2.4 that if no abort occurs the distribution of $\mathbf{s}_e, \mathbf{s}_s$ does not depend on $s, e$, and thus simulated and real protocol transcripts are indistinguishable. In case that an abort occurs, the indistinguishability follows from the hiding property of $\mathsf{aCommit}$ and the fact that aborts are equally likely for every $\mathsf{c}$.
*Special soundness.* Assume that we are given $(c_{\text{aux}}, \mathsf{c}', (\mathbf{t}', d'_{\text{aux}}, (\mathbf{s}'_s, \mathbf{s}'_e)))$ and $(c_{\text{aux}}, \mathsf{c}'', (\mathbf{t}'', d''_{\text{aux}}, (\mathbf{s}''_s, \mathbf{s}''_e)))$ passing the checks performed by the verifier. From the binding property of the auxiliary commitment scheme we get that $\mathbf{t}' = \mathbf{t}'' =: \mathbf{t}$. Now, by

subtracting the verification equations we get: $(X^{\mathtt{c}'} - X^{\mathtt{c}''})y = a(\mathtt{s}'_s - \mathtt{s}''_s) + (\mathtt{s}'_e - \mathtt{s}''_e)$. Multiplying by $2(X^{\mathtt{c}'} - X^{\mathtt{c}''})^{-1}$ yields:

$$2y = a\frac{2(\mathtt{s}'_s - \mathtt{s}''_s)}{X^{\mathtt{c}'} - X^{\mathtt{c}''}} + \frac{2(\mathtt{s}'_e - \mathtt{s}''_e)}{X^{\mathtt{c}'} - X^{\mathtt{c}''}} =: 2a\hat{s} + 2\hat{e}\,.$$

Furthermore, we get that $\|2\hat{s}\| \leq \|\mathtt{s}'_e - \mathtt{s}''_e\|\sqrt{n}\left\|\frac{2}{X^{\mathtt{c}'} - X^{\mathtt{c}''}}\right\| \leq \tilde{\mathcal{O}}(n^2\alpha)$, where in the second inequality we used Lemma 3.1, and similarly for $\hat{e}$.

*High-entropy commitments.* This directly follows from the security of the auxiliary commitment scheme. □

By Section 2.5, both the completeness and the knowledge error can be made negligible if $n > M^2$.

# 4 Proving Equality among Classical and Lattice-Based Primitives

In the following we show how our basic protocol from Section 3 can be used to link number-theory and lattice-based primitives via zero-knowledge proofs of knowledge. We exemplify this by showing how to prove that the messages contained in Pedersen commitments correspond to the plaintext in an encryption under the secure version of NTRU. We want to stress that in particular the choice of the encryption scheme is arbitrary, and it is easy to exchange it against other schemes, including standard NTRU [24] or Ring-LWE encryption [30].

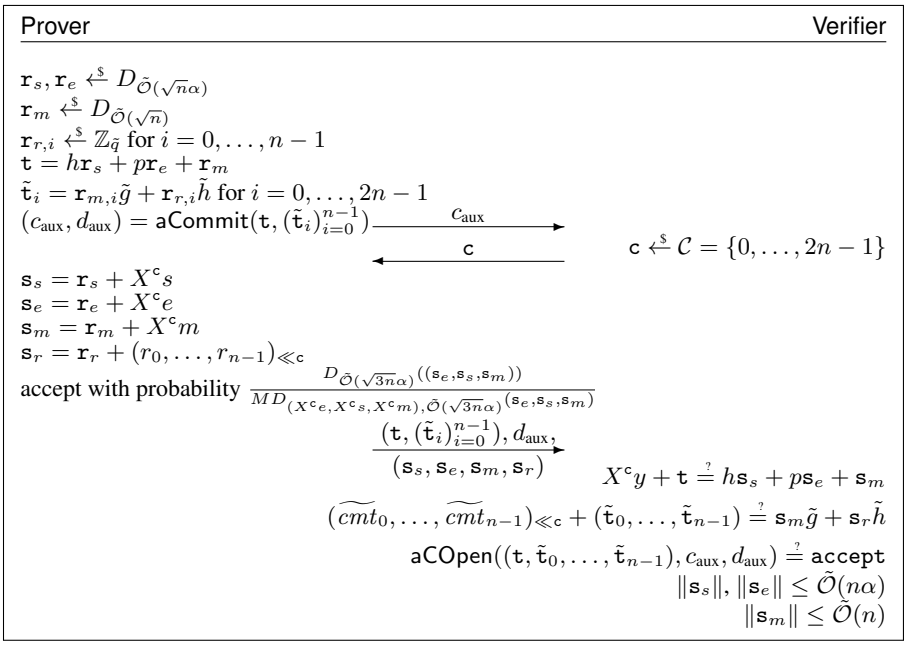
Let $y = hs + pe + m \in R_q$ be the NTRU encryption of a message $m \in \{0,1\}^n$, and let $p > 2n^2$ be coprime with $q$. Let further $\tilde{g}, \tilde{h}$ be a Pedersen commitment parameters, cf. Section 2.2, and let $\widetilde{cmt}_i = m_i\tilde{g} + r_i\tilde{h}$ for $i = 0, \ldots, n-1$ be commitments to coefficients of $m$, where the order of $\tilde{g}$ and $\tilde{h}$ is $\tilde{q} > 2n^2$.

Then Protocol 4.1 can be used to prove, in zero-knowledge, that the commitments and the ciphertext are broadly well-formed and consistent, i.e., contain the same message. More precisely, the protocol guarantees the verifier that the prover knows the plaintext encrypted in $2y$, and that the coefficients of the respective message are all smaller than $p$. Furthermore, it shows that the messages are the same that are contained in $2\widetilde{cmt}_i$, i.e., $2y$ and the $2\widetilde{cmt}_i$ are consistent.

**Theorem 4.2.** *Protocol 4.1 is an HVZK $\Sigma'$-protocol for the following relations:*

$$\mathcal{R} = \Bigg\{((\tilde{g},\tilde{h},(\widetilde{cmt}_i)_{i=0}^{n-1},h,p,y),(m,s,e,(r_i)_{i=0}^{n-1})) : y = hs + pe + m \\ \wedge \bigwedge_{i=0}^{n-1} \widetilde{cmt}_i = m_i\tilde{g} + r_i\tilde{h} \wedge \|m\|_\infty \leq 1 \wedge \|s\|, \|e\| \leq \tilde{\mathcal{O}}(\sqrt{n}\alpha)\Bigg\},$$

$$\mathcal{R}' = \Bigg\{((\tilde{g},\tilde{h},(\widetilde{cmt}_i)_{i=0}^{n-1},h,p,y),(m,s,e,(r_i)_{i=0}^{n-1})) : 2y = 2hs + 2pe + 2m \\ \wedge \bigwedge_{i=0}^{n-1} 2\widetilde{cmt}_i = (2m \bmod q)_i\tilde{g} + 2r_i\tilde{h} \\ \wedge \|2m\|_\infty \leq 2n^2 \wedge \|2s\|, \|2e\| \leq \tilde{\mathcal{O}}(n^2\alpha)\Bigg\}.$$




**Protocol 4.1:** Proof that Pedersen commitments and NTRU encryption contain the same plaintext.

*where* $(2m \bmod q)_i$ *is the* $i$*-coefficient of* $2m \in R_q$*. The protocol has a knowledge error of* $1/(2n)$*, and a completeness error of* $1 - 1/M$*.*

*Furthermore, if for the auxiliary commitment scheme a commitment does not only bind the user to the message, but also to the opening information, the protocol has quasi-unique responses and high-entropy commitments.*

A detailed proof is given in the full version. By the remark in Section 2.5, both the completeness and the knowledge error can be made negligible if $n > M$.

# 5 Application to Group Signatures

We next show how Protocol 4.1 can be used to construct a group signature scheme with signature size logarithmic in the number of group members. The scheme is private under lattice assumptions, but traceable/unforgeable under non-lattice assumptions. As argued in the introduction, this may be realistic in applications where privacy needs to be guaranteed on the long term. For example, if group signatures are used to sign votes in electronic elections, unforgeability is mainly important when the votes are counted, but privacy needs to be preserved long after that.

Before presenting the actual signature scheme, we will prove secure a variation of a generic construction that we believe is folklore, as it underlies several direct schemes in the literature [3,8] and was explicitly described by Chase and Lysyanskaya [11]. The resulting construction satisfies the following definition of group signatures providing full (CCA) anonymity put forth by Bellare et al. [5].

Experiment $\mathbf{Exp}^{\text{anon}-b}_{\mathsf{A}}(\lambda)$:
$(gpk, gok, \mathbf{gsk}) \xleftarrow{\$} \mathsf{GKG}(1^\lambda, 1^N)$
$(st, i_0^*, i_1^*, m^*) \xleftarrow{\$} \mathsf{A}^{\mathsf{GOpen}(gok,\cdot,\cdot)}((gpk, \mathbf{gsk}), \varepsilon)$
$\sigma^* \xleftarrow{\$} \mathsf{GSign}(\mathbf{gsk}[i_b], m^*)$
$b' \xleftarrow{\$} \mathsf{A}^{\mathsf{GOpen}(gok,\cdot,\cdot)}(\sigma^*, st)$
If $(m^*, \sigma^*) \notin \mathcal{Q}_{\mathsf{GOpen}}$
then return $b'$ else return 0

Experiment $\mathbf{Exp}^{\text{trace}}_{\mathsf{A}}(\lambda)$:
$(gpk, gok, \mathbf{gsk}) \xleftarrow{\$} \mathsf{GKG}(1^\lambda, 1^N)$
$(m, \sigma) \xleftarrow{\$} \mathsf{A}^{\mathsf{GSign}(\mathbf{gsk}[\cdot],\cdot),\, \mathbf{gsk}[\cdot]}(gpk, gok)$
$i \xleftarrow{\$} \mathsf{GOpen}(gok, m, \sigma)$
If $\mathsf{GVerify}(gpk, m, \sigma) = 1 \ \wedge \ i \notin \mathcal{Q}_{\mathbf{gsk}}$
$\wedge\ (i, m) \notin \mathcal{Q}_{\mathsf{GSign}}$
then return 1 else return 0

**Fig. 1.** The anonymity (left) and traceability (right) experiments for group signatures. The sets $\mathcal{Q}_{\mathsf{GOpen}}, \mathcal{Q}_{\mathsf{GSign}}, \mathcal{Q}_{\mathbf{gsk}}$ contain all queries $(m, \sigma)$, $(i, m)$, and $i$ that $\mathsf{A}$ submitted to its $\mathsf{GOpen}$, $\mathsf{GSign}$, and $\mathbf{gsk}$ oracles, respectively.

**Definition 5.1.** *A* group signature scheme *is a tuple* $(\mathsf{GKG}, \mathsf{GSign}, \mathsf{GVerify}, \mathsf{GOpen})$ *where:*

- *On input* $1^\lambda, 1^N$*, the key generation algorithm* $\mathsf{GKG}$ *outputs a group public key* $gpk$*, an opening key* $gok$*, and a vector of* $N$ *signing keys* $\mathbf{gsk}$ *where* $\mathbf{gsk}[i]$ *is given to user* $i \in \{1, \ldots, N\}$*.*
- *On input* $gsk = \mathbf{gsk}[i]$ *and message* $m \in \mathcal{M}$*, the signing algorithm* $\mathsf{GSign}$ *outputs a group signature* $\sigma$*.*
- *On input* $gpk, m, \sigma$*, the verification algorithm* $\mathsf{GVerify}$ *outputs* `accept` *or* `reject`*.*
- *On input* $gok, m, \sigma$*, the opening algorithm* $\mathsf{GOpen}$ *outputs the identity of the purported signer* $i \in \{1, \ldots, N\}$ *or* $\perp$ *to indicate failure.*

*The algorithms satisfy the following properties:*

- Correctness*: Verification accepts whenever keys and signatures are honestly generated, i.e., for all* $\lambda, N \in \mathbb{N}$*, all* $i \in \{1, \ldots, N\}$*, and all* $m \in \mathcal{M}$

$$\Pr\left[\begin{array}{c}\mathsf{GVerify}(gpk, m, \sigma) = \texttt{accept} : \\ (gpk, gok, \mathbf{gsk}) \xleftarrow{\$} \mathsf{GKG}(1^\lambda, 1^N), \sigma \xleftarrow{\$} \mathsf{GSign}(\mathbf{gsk}[i], m)\end{array}\right] = 1 \ .$$

- Anonymity*: One cannot tell which signer generated a particular signature, even when given access to an opening oracle. Referring to Figure 1, for all PPT* $\mathsf{A}$ *there exists a negligible function* $\mathsf{negl}$ *such that*

$$\left|\Pr[\mathbf{Exp}^{\text{anon}-0}_{\mathsf{A}}(\lambda) = 1] - \Pr[\mathbf{Exp}^{\text{anon}-1}_{\mathsf{A}}(\lambda) = 1]\right| \leq \mathsf{negl}(\lambda) \ .$$

- Traceability*: One cannot generate a signature that cannot be opened or that opens to an honest user. Referring to Figure 1, for all PPT* $\mathsf{A}$ *there exists a negligible function* $\mathsf{negl}$ *such that*

$$\Pr[\mathbf{Exp}^{\text{trace}}_{\mathsf{A}}(\lambda)] \leq \mathsf{negl}(\lambda) \ .$$

## 5.1 Building Blocks

The construction is based on weakly unforgeable standard signatures, and signature proofs of knowledge. In the following, we recap the respective definitions.

Informally, a signature scheme is a triple $(\mathsf{SKG}, \mathsf{SSign}, \mathsf{SVerify})$, where $\mathsf{SKG}$ generates a signing/verification key pair $(ssk, spk)$, $\mathsf{SSign}$ can be used to sign a message $m$ using the signing key, and $\mathsf{SVerify}$ can be used to check the validity of a signature only using the public verification key. It should hold that honestly computed signatures are always valid, and that no adversary can come up with a valid signature on a new message after having received signatures on messages that he chose before obtaining $spk$. A formal definition can be found in the full version.

Concerning signature proofs of knowledge, we adapt the definitions of Chase and Lysyanskaya [11] to allow for signatures in the random-oracle model (ROM) that are simulated by programming the random oracle $H$ and extracted through rewinding. We also generalize the definition to allow for a soundness gap: signing is performed using a witness from $R$ for a language $\mathcal{L}$, while extraction only guarantees that the signer knows a witness from $R' \supseteq R$ for relation $\mathcal{L}'$. Finally, we add a definition of simulation soundness, meaning that an adversary cannot produce new signatures for false statements even after seeing simulated signatures on arbitrary statements.

**Definition 5.2.** *A* signature of knowledge scheme *for languages $\mathcal{L}, \mathcal{L}'$ with respective witness relations $R, R'$ is a tuple* $(\mathsf{SoKSetup}, \mathsf{SoKSign}, \mathsf{SoKVerify}, \mathsf{SoKSim})$ *where:*

- *On input $1^\lambda$, the setup algorithm* $\mathsf{SoKSetup}$ *outputs common parameters $sokp$.*
- *On input $sokp, x, w$ such that $(x, w) \in R$ and message $m \in \mathcal{M}$, the signing algorithm* $\mathsf{SoKSign}$ *outputs a signature of knowledge $sok$.*
- *On input $sokp, x, m, sok$, the verification algorithm* $\mathsf{SoKVerify}$ *outputs* `accept` *or* `reject`.
- *The stateful simulation algorithm* $\mathsf{SoKSim}$ *can be called in three modes. When called as $(sokp, st) \xleftarrow{\$} \mathsf{SoKSim}(\mathtt{setup}, 1^\lambda, \varepsilon)$, it produces simulated parameters $sokp$, possibly keeping a trapdoor in its internal state $st$. When run as $(h, st') \xleftarrow{\$} \mathsf{SoKSim}(\mathtt{ro}, Q, st)$, it produces a response $h$ for a random oracle query $Q$. When run as $(sok, st') \xleftarrow{\$} \mathsf{SoKSim}(\mathtt{sign}, x, m, st)$, it produces a simulated signature of knowledge $sok$ without using a witness.*
  *For ease of notation, let $\mathsf{StpSim}(1^\lambda)$ be the algorithm that returns the first part of $\mathsf{SoKSim}(\mathtt{setup}, 1^\lambda, st)$, let $\mathsf{ROSim}(Q)$ be the algorithm that returns the first part of $\mathsf{SoKSim}(\mathtt{ro}, Q, st)$, let $\mathsf{SSim}(x, w, m)$ be the algorithm that returns the first part of $\mathsf{SoKSim}(\mathtt{sign}, x, m, st)$ if $(x, w) \in R$ and returns $\perp$ otherwise, and let $\mathsf{SSim}'(x, m)$ be the algorithm that returns the first part of $\mathsf{SoKSim}(\mathtt{sign}, x, m, st)$ without checking language membership. The experiment keeps a single synchronized state for* $\mathsf{SoKSim}$ *across all invocations of these derived algorithms.*

*The algorithms satisfy the following properties:*

- Correctness*: Verification accepts whenever parameters and signatures are correctly generated, i.e., for all $\lambda \in \mathbb{N}$, all $(x, w) \in R$, and all $m \in \mathcal{M}$, there exists a negligible function* $\mathsf{negl}$ *such that*

$$\Pr\left[\begin{array}{c} \mathsf{SoKVerify}(sokp, x, m, sok) = \mathtt{reject} : \\ sokp \xleftarrow{\$} \mathsf{SoKSetup}(1^\lambda),\ sok \xleftarrow{\$} \mathsf{SoKSign}(sokp, x, w, m) \end{array}\right] \leq \mathsf{negl}(\lambda)\ .$$

– Simulatability*: No adversary can distinguish whether it is interacting with a real random oracle and signing oracle, or with their simulated versions. Formally, for all PPT* $\mathsf{A}$ *there exists a negligible function* $\mathsf{negl}$ *such that*

$$\Big|\Pr[b = 1 : sokp \xleftarrow{\$} \mathsf{SoKSetup}(1^\lambda), b \xleftarrow{\$} \mathsf{A}^{H(\cdot),\mathsf{SoKSign}(sokp,\cdot,\cdot,\cdot)}(sokp)] - \Pr[b = 1 : sokp \xleftarrow{\$} \mathsf{StpSim}(1^\lambda), b \xleftarrow{\$} \mathsf{A}^{\mathsf{ROSim}(\cdot),\mathsf{SSim}(\cdot,\cdot,\cdot)}(sokp)]\Big| \leq \mathsf{negl}(\lambda)\,.$$

– Extractability*: The only way to produce a valid signature of knowledge is by knowing a witness from* $R'$*. Formally, for all PPT* $\mathsf{A}$ *there exists an extractor* $\mathsf{SoKExt}_\mathsf{A}$ *and a negligible function* $\mathsf{negl}$ *such that*

$$\Pr\left[\begin{array}{c} \mathsf{SoKVerify}(sokp, x, m, sok) = \mathtt{accept} \\ \wedge (x, w, m) \notin \mathcal{Q} \wedge (x, w) \notin R' : \\ sokp \xleftarrow{\$} \mathsf{StpSim}(1^\lambda; \rho_\mathsf{S}), \\ (x, m, sok) \xleftarrow{\$} \mathsf{A}^{\mathsf{ROSim}(\cdot),\mathsf{SSim}(\cdot,\cdot,\cdot)}(sokp; \rho_\mathsf{A}), \\ w \xleftarrow{\$} \mathsf{SoKExt}_\mathsf{A}(sokp, x, m, sok, \rho_\mathsf{S}, \rho_\mathsf{A}) \end{array}\right] \leq \mathsf{negl}(\lambda)\,,$$

*where* $\mathcal{Q}$ *is the set of queries* $(x, w, m)$ *that* $\mathsf{A}$ *submitted to its* $\mathsf{SSim}$ *oracle.*

– Simulation-soundness*: No adversary can produce a new signature on a false statement for* $\mathcal{L}'$*, even after seeing a signature on an arbitrary statement. Formally, for all PPT* $\mathsf{A}$ *there exists a negligible function* $\mathsf{negl}$ *such that*

$$\Pr\left[\begin{array}{c} \mathsf{SoKVerify}(sokp, x, m, sok) = \mathtt{accept} \\ \wedge (x', m', sok') \neq (x, m, sok) \wedge x \notin \mathcal{L}' : \\ sokp \xleftarrow{\$} \mathsf{StpSim}(1^\lambda), (x, m, st) \xleftarrow{\$} \mathsf{A}^{\mathsf{ROSim}(\cdot)}(sokp), \\ sok \xleftarrow{\$} \mathsf{SSim}'(x, m), (x', m', sok') \xleftarrow{\$} \mathsf{A}^{\mathsf{ROSim}(\cdot)}(sok, st) \end{array}\right] \leq \mathsf{negl}(\lambda)\,.$$

## 5.2 Generic Construction

A folklore construction of group signatures is to have a user's signing key be a standard signature on his identity $i$, and to have a group signature on message $m$ be an encryption of his identity together with a signature of knowledge on $m$ that the encrypted identity is equal to the identity in his signing key. The construction appeared implicitly [3, 8] or explicitly [11] in the literature, but was never proved secure.

To obtain full anonymity, this generic construction would probably require CCA security from encryption scheme, but our NTRU variant is only semantically secure. We could apply a generic CCA-yielding transformation using random oracles or non-interactive zero-knowledge proofs of knowledge (NIZK), but this would make the signature of knowledge hopelessly inefficient. Instead, we take inspiration from the Naor-Yung construction [31, 33] by using a semantically secure scheme to encrypt the user's identity twice under two different public keys and letting the signature of knowledge prove that both ciphertexts encrypt the same plaintext. Moreover, our proof systems have a soundness gap: the adversary for the soundness game may use more noise in the ciphertexts than what the encryption algorithm $\mathsf{Enc}$ does, and may also encrypt plaintexts outside $\{0, 1\}^n$. We therefore give a generic construction that deviates slightly from the general idea, but that is sufficient and that we can efficiently instantiate with our protocol from Section 3.

Let $(\mathsf{EncKG}, \mathsf{Enc}, \mathsf{Dec})$ be an encryption scheme with message space $\mathcal{ID}$, let $\mathcal{ID}' \supseteq \mathcal{ID}$, and let $\mathsf{Enc}'$ be an algorithm such that for all key pairs $(epk, esk) \xleftarrow{\$} \mathsf{EncKG}(1^\lambda)$, for all $i \in \mathcal{ID}$ and for all random tapes[4] $\rho, \rho'$ and all $i \in \mathcal{ID}$, $i' \in \mathcal{ID}'$:

$$\mathsf{Enc}(epk, i; \rho) = \mathsf{Enc}'(epk, i; \rho) \text{ and } \mathsf{Dec}(esk, \mathsf{Enc}'(epk, i'; \rho')) = i' .$$

The algorithm $\mathsf{Enc}'$ represents the way the adversary can generate the ciphertexts and still prove them valid. The above property ensures that completeness holds perfectly even with $\mathsf{Enc}'$. The IND-CPA property still has to hold with $\mathsf{Enc}$.

For our instantiation with the NTRU encryption scheme from Theorem 4.2, $\mathcal{ID} = \{0,1\}^\ell$ which is identified with $\{0, \dots, 2^\ell - 1\}$ (with $\ell \leq n, \tilde{q}$), $\mathcal{ID}' = \mathbb{Z}_{\tilde{q}}$ and $\rho = (s, e)$. The algorithm $\mathsf{Enc}'((h,p), i'; \rho')$ with $i' \in \mathcal{ID}'$ checks that either $\rho'$ is a triple $(s, e, i'')$, or $i' \in \mathcal{ID}$ and $\rho'$ is a pair of vectors $(s, e)$. In the latter case, $i''$ is just the binary vector in $\{0,1\}^\ell$ corresponding to $i'$. In both cases, $s$ and $e$ must be such that $\|2s\|, \|2e\| \leq \tilde{\mathcal{O}}(n^2\alpha)$, $i'' \in R_q$, $2i' = \sum_{j=0}^{n-1} 2^j (2i'' \bmod q)_j \bmod \tilde{q}$, and $\|i''\|_\infty \leq 2n^2 < p, \tilde{q}$. If all these requirements are met, $\mathsf{Enc}'$ outputs $y \leftarrow hs + pe + i'$. We need to slightly change the algorithms $\mathsf{EncKG}$ and $\mathsf{Enc}$ to truncate the distribution of $g$, $s$ and $e$, to ensure that $\|s\|, \|e\| \leq \tilde{\mathcal{O}}(\sqrt{n}\alpha)$ and $\|g\|$ is small enough for the decryption below. We also change the algorithm $\mathsf{Dec}$: to decrypt $C = y$ with secret key $f$, it computes $y' = 2fy \in R_q$, and outputs $i' = (\sum_{j=0}^{n-1} 2^j (y' \bmod p)_j)/2 \bmod \tilde{q}$. In other words, it decrypts $2C = 2y$ into $y' \bmod p$, and then recover the corresponding identity in $\mathcal{ID}' = \mathbb{Z}_{\tilde{q}}$. This does not touch security.

Let $(\mathsf{SKG}, \mathsf{SSign}, \mathsf{SVerify})$ be a signature scheme and let $(\mathsf{SoKSetup}, \mathsf{SoKSign}, \mathsf{SoKVerify}, \mathsf{SoKSim})$ be a signature of knowledge scheme for the languages $\mathcal{L}, \mathcal{L}'$ with witness relationships

$$R = \{((spk, epk_1, epk_2, C_1, C_2), (i, sig, \rho_1, \rho_2)) : \mathsf{SVerify}(spk, i, sig) = \mathtt{accept} \\ \wedge\ C_1 = \mathsf{Enc}(epk_1, i; \rho_1) \wedge C_2 = \mathsf{Enc}(epk_2, i; \rho_2)\} ,$$

$$R' = \{((spk, epk_1, epk_2, C_1, C_2), (i', sig, \rho'_1, \rho'_2)) : \mathsf{SVerify}(spk, i', sig) = \mathtt{accept} \\ \wedge\ C_1 = \mathsf{Enc}'(epk_1, i'; \rho'_1) \wedge C_2 = \mathsf{Enc}'(epk_2, i'; \rho'_2)\} .$$

Consider the following group signature scheme with user identities $i \in \mathcal{ID}$:

- $\mathsf{GKG}(1^\lambda, 1^N)$: The group manager generates signing keys $(spk, ssk) \xleftarrow{\$} \mathsf{SKG}(1^\lambda)$, encryption keys $(epk_1, esk_1) \xleftarrow{\$} \mathsf{EncKG}(1^\lambda)$, $(epk_2, esk_2) \xleftarrow{\$} \mathsf{EncKG}(1^\lambda)$, and parameters $sokp \xleftarrow{\$} \mathsf{SoKSetup}(1^\lambda)$. He computes $\mathbf{gsk}[i] \xleftarrow{\$} \mathsf{SSign}(ssk, i)$ for $i \in \mathcal{ID}$ and outputs $gpk = (spk, epk_1, epk_2, sokp)$, $gok = (gpk, esk_1)$, and $\mathbf{gsk}$.
- $\mathsf{GSign}(gsk, m)$: Signer $i$ computes two ciphertexts $C_1 \leftarrow \mathsf{Enc}(epk_1, i; \rho_1)$ and $C_2 \leftarrow \mathsf{Enc}(epk_2, i; \rho_2)$, computes a signature of knowledge $sok \xleftarrow{\$} \mathsf{SoKSign}(sokp, (spk, epk_1, epk_2, C_1, C_2), (i, sig, \rho_1, \rho_2), m)$ and outputs the group signature $\sigma = (C_1, C_2, sok)$.
- $\mathsf{GVerify}(gpk, m, \sigma)$: To verify a group signature, one checks that $\mathsf{SoKVerify}(sokp, (spk, epk_1, epk_2, C_1, C_2), m, sok) = \mathtt{accept}$.

[4] To simplify notation in this section, we assume that the random tapes $\rho, \rho'$ are not necessarily a uniform binary bitstrings as usual. Rather, we see $\rho$ as the list of random values that $\mathsf{Enc}$ directly derives from the random tape, while $\rho'$ can be seen as an auxiliary adversarial input to the $\mathsf{Enc}'$ algorithm.

– $\mathsf{GOpen}(gok, m, \sigma)$: The opener checks that $\mathsf{GVerify}(gpk, m, \sigma) = \mathtt{accept}$, and returns $i \leftarrow \mathsf{Dec}(esk_1, C_1)$.

**Theorem 5.3.** *The group signature scheme sketched above is anonymous in the ROM if the encryption scheme is semantically secure and the signature of knowledge scheme is simulatable and simulation-sound.*

**Theorem 5.4.** *The group signature scheme is traceable in the ROM if the underlying signature scheme is weakly unforgeable and the signature of knowledge scheme is simulatable and extractable.*

The proofs of the last two theorems are omitted here and are given in the full version.

## 5.3 Signatures of Knowledge from $\Sigma'$-Protocols

We now show a construction of the required signatures of knowledge in the random-oracle model from a signature scheme and an encryption scheme with $\Sigma'$-protocol proofs. More particularly, we require that for the signature scheme one can prove knowledge of a signature on a committed message, while for the encryption scheme one can prove that an encrypted plaintext is equal to a committed message.

Let $(\mathsf{CSetup}, \mathsf{Commit}, \mathsf{COpen})$ be a commitment scheme, let $(\mathsf{EncKG}, \mathsf{Enc}, \mathsf{Dec})$ be an encryption scheme with message space $\mathcal{M}$ and let $\mathsf{Enc}'$ be an associated algorithm as described earlier. Let $(\mathsf{P}_\mathrm{s}, \mathsf{V}_\mathrm{s}, \mathsf{S}_\mathrm{s})$ be a $\Sigma$-protocol for the language $\mathcal{L}_\mathrm{s}$ with

$$R_\mathrm{s} = \{((spk, cpars, cmt), (sig, m, o)) : \mathsf{SVerify}(spk, m, sig) = \mathtt{accept} \wedge \mathsf{COpen}(cpars, m, cmt, o) = \mathtt{accept}\} .$$

Let also $(\mathsf{P}_\mathrm{e}, \mathsf{V}_\mathrm{e}, \mathsf{S}_\mathrm{e})$ be a $\Sigma'$-protocol for the languages $\mathcal{L}_\mathrm{e}, \mathcal{L}'_\mathrm{e}$ with

$$R_\mathrm{e} = \{((epk, C, cpars, cmt), (m, \rho, o)) : C = \mathsf{Enc}(epk, m; \rho) \wedge \mathsf{COpen}(cpars, m, cmt, o) = \mathtt{accept}\} ,$$

$$R'_\mathrm{e} = \{((epk, C, cpars, cmt), (m, \rho', o)) : C = \mathsf{Enc}'(epk, m; \rho') \wedge \mathsf{COpen}(cpars, m, cmt, o) = \mathtt{accept}\} .$$

Let $\mathcal{C}_\mathrm{s}$ and $\mathcal{C}_\mathrm{e}$ be the challenge spaces for these respective protocols, and let $H : \{0,1\}^* \to \mathcal{C}_\mathrm{s} \times \mathcal{C}_\mathrm{e}$. Consider the following construction of a signature of knowledge scheme for the languages $\mathcal{L}$ and $\mathcal{L}'$:

– $\mathsf{SoKSetup}(1^\lambda)$: Return $sokp = cpars \xleftarrow{\$} \mathsf{CSetup}(1^\lambda)$.
– $\mathsf{SoKSign}(sokp, x, w, m)$: Create a commitment $(cmt, o) \xleftarrow{\$} \mathsf{Commit}(cpars, m)$. Compute the first round of the $\Sigma'$-protocols for a signature and two encryptions $(\mathtt{t}_\mathrm{s}, st_\mathrm{s}) \xleftarrow{\$} \mathsf{P}_\mathrm{s}((spk, cpars, cmt), (sig, m, o))$ and $(\mathtt{t}_j, st_j) \xleftarrow{\$} \mathsf{P}_\mathrm{e}((epk_j, C_j, cpars, cmt), (m, \rho_j, o))$ for $j = 1, 2$. Generate the challenges $(\mathtt{c}_\mathrm{s}, \mathtt{c}_\mathrm{e}) \leftarrow H(spk, cpars, cmt, epk_1, C_1, epk_2, C_2, \mathtt{t}_\mathrm{s}, \mathtt{t}_1, \mathtt{t}_2, m)$. Compute responses $\mathtt{s}_\mathrm{s} \leftarrow \mathsf{P}_\mathrm{s}(\mathtt{c}_\mathrm{s}, st_\mathrm{s})$ and $\mathtt{s}_j \leftarrow \mathsf{P}_\mathrm{e}(\mathtt{c}_\mathrm{e}, st_j)$ for $j = 1, 2$ and output the signature of knowledge $sok = (\mathtt{t}_\mathrm{s}, \mathtt{t}_1, \mathtt{t}_2, \mathtt{s}_\mathrm{s}, \mathtt{s}_1, \mathtt{s}_2)$.
– $\mathsf{SoKVerify}(sokp, x, m, sok)$: Recompute the challenges $(\mathtt{c}_\mathrm{s}, \mathtt{c}_\mathrm{e}) \leftarrow H(spk, cpars, cmt, epk_1, C_1, epk_2, C_2, \mathtt{t}_\mathrm{s}, \mathtt{t}_1, \mathtt{t}_2, m)$. Return $\mathtt{accept}$ if $\mathsf{V}_\mathrm{s}((spk, cpars, cmt), \mathtt{t}_\mathrm{s}, \mathtt{c}_\mathrm{s}, \mathtt{s}_\mathrm{s}) = \mathtt{accept}$ and $\mathsf{V}_\mathrm{e}((epk_j, C_j, cpars, cmt), \mathtt{t}_j, \mathtt{c}_\mathrm{e}, \mathtt{s}_j) = \mathtt{accept}$ for $j = 1, 2$. Otherwise, return $\mathtt{reject}$.

– SoKSim: The simulation algorithm keeps in its state its random tape, an initially empty table $HT$ to keep track of previous random-oracle queries, and a counter $ctr$ initialized to zero. The simulator's random tape $\rho$ includes random-oracle responses $h_1, \ldots, h_{q_\mathrm{H}+q_\mathrm{S}} \xleftarrow{\$} \mathcal{C}_\mathrm{s} \times \mathcal{C}_\mathrm{e}$, where $q_\mathrm{H}$ and $q_\mathrm{S}$ are upper bounds on the number of random-oracle and signing queries that an adversary can make. When called as $\mathsf{SoKSim}(\mathtt{setup}, 1^\lambda, \varepsilon)$, it generates commitment parameters $cpars \xleftarrow{\$} \mathsf{CSetup}(1^\lambda)$ and returns $(cpars, st = (\rho, HT, ctr, cpars))$. When run as $\mathsf{SoKSim}(\mathtt{ro}, Q, st)$, it checks whether the query $Q$ was made before. If so, it returns $h_{HT[q]}$. Otherwise, it increases the counter $ctr$, sets $HT[Q] \leftarrow ctr$, and returns $h_{ctr}$. When run as $\mathsf{SoKSim}(\mathtt{sign}, (spk, epk_1, epk_2, C_1, C_2), m, st)$, the simulator first creates a commitment $(cmt, o) \xleftarrow{\$} \mathsf{Commit}(1, cpars)$. It then increases the counter $ctr$ and parses $h_{ctr}$ as $(\mathtt{c}_\mathrm{s}, \mathtt{c}_\mathrm{e})$. It runs the simulators $\mathsf{S}_\mathrm{s}, \mathsf{S}_\mathrm{e}$ to obtain simulated protocol transcripts $(\mathtt{t}_\mathrm{s}, \mathtt{s}_\mathrm{s}) \xleftarrow{\$} \mathsf{S}_\mathrm{s}((spk, cpars, cmt), \mathtt{c}_\mathrm{s})$ and $(\mathtt{t}_j, \mathtt{s}_j) \xleftarrow{\$} \mathsf{S}_\mathrm{s}((epk_j, C_j, cpars, cmt), \mathtt{c}_\mathrm{e})$ for $j = 1, 2$. If $HT[spk, cpars, cmt, epk_1, C_1, epk_2, C_2, \mathtt{t}_\mathrm{s}, \mathtt{t}_1, \mathtt{t}_2, m]$ is not defined, then set it to $h_{ctr}$, else abort.

**Theorem 5.5.** *The above scheme is correct if the proof systems* $(\mathsf{P}_\mathrm{s}, \mathsf{V}_\mathrm{s})$ *and* $(\mathsf{P}_\mathrm{e}, \mathsf{V}_\mathrm{e})$ *have negligible completeness error.*

**Theorem 5.6.** *The above scheme is simulatable in the random-oracle model if the commitment scheme is hiding and the proof systems* $(\mathsf{P}_\mathrm{s}, \mathsf{V}_\mathrm{s})$ *and* $(\mathsf{P}_\mathrm{e}, \mathsf{V}_\mathrm{e})$ *are special HVZK and have high-entropy commitments.*

**Theorem 5.7.** *The above scheme is extractable in the random-oracle model if the commitment scheme is binding and the proof systems* $(\mathsf{P}_\mathrm{s}, \mathsf{V}_\mathrm{s})$ *and* $(\mathsf{P}_\mathrm{e}, \mathsf{V}_\mathrm{e})$ *are special-sound and have super-polynomial challenge spaces and negligible knowledge error.*

**Theorem 5.8.** *The above scheme is simulation-sound if the underlying commitment scheme is binding and the underlying* $\Sigma'$*-protocols* $(\mathsf{P}_\mathrm{s}, \mathsf{V}_\mathrm{s}, \mathsf{S}_\mathrm{s})$ *and* $(\mathsf{P}_\mathrm{e}, \mathsf{V}_\mathrm{e}, \mathsf{S}_\mathrm{e})$ *are special-sound, have quasi-unique responses, super-polynomial challenge spaces, and negligible knowledge error.*

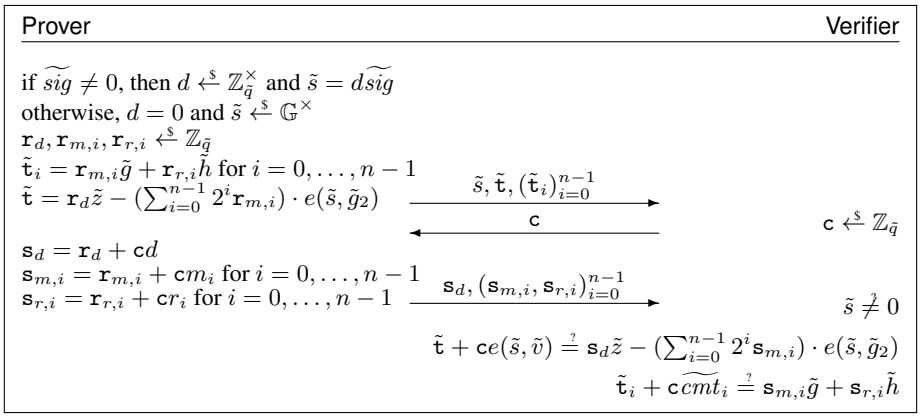
Due to length limitations, the proofs of the previous theorems can be found in the full version.

## 5.4 $\Sigma'$-Protocols for Boneh-Boyen Signatures and the Group Signature Scheme

In the following we briefly recap the weakly unforgeable version of the Boneh-Boyen signature scheme [6, 7]. We assume that the reader is familiar with bilinear pairings and the strong Diffie-Hellman (SDH) assumption.The Boneh-Boyen signature scheme is defined as follows for a bilinear group generator BGGen:

SKG. This algorithm first computes $(\tilde{q}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \xleftarrow{\$} \mathsf{BGGen}(1^\lambda)$. It chooses $\tilde{g}_1 \xleftarrow{\$} \mathbb{G}_1^\times$, $\tilde{g}_2 \xleftarrow{\$} \mathbb{G}_2^\times$, $x \xleftarrow{\$} \mathbb{Z}_{\tilde{q}}^\times$, and defines $\tilde{v} = x\tilde{g}_2$ and $\tilde{z} = e(\tilde{g}_1, \tilde{g}_2)$. It outputs $spk = ((\tilde{q}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e), \tilde{g}_1, \tilde{g}_2, \tilde{v}, \tilde{z})$ and $ssk = x$.

SSign. To sign a message $m \in \mathbb{Z}_{\tilde{q}} \setminus \{-ssk\}$ with secret key $ssk = x$, this algorithm outputs the signature $\widetilde{sig} = \frac{1}{x+m}\tilde{g}_1$ if $x + m \neq 0$, and 0 otherwise.




**Protocol 5.10:** Proof of possession of a signature on $m$, which is also contained in a set of Pedersen commitments.

SVerify**.** Given a signature public key $spk$, a message $m \in \mathbb{Z}_{\tilde{q}}$ and a signature $\widetilde{sig}$, this algorithm outputs `accept` if $\tilde{v} + m\tilde{g}_2 = 0$ in case $\widetilde{sig} = 0$, and if $e(\widetilde{sig}, \tilde{v} + m\tilde{g}_2) = \tilde{z}$ in case $\widetilde{sig} \neq 0$. In all other cases, it outputs `reject`.

**Lemma 5.9.** *If the SDH assumption holds for* BGGen*, then the above scheme is a weakly unforgeable signature scheme.*

We next show how a user can prove possession of a Boneh-Boyen signature on a message $m$, while keeping both, the message and the signature, private. In addition, the proof will additionally show that the $m$ is also contained in a set of Pedersen commitments $\widetilde{cmt}_i = m_i\tilde{g} + r_i\tilde{h}$ such that $m = \sum_{i=0}^{n-1} 2^i m_i$, cf. Section 2.2.

The idea underlying Protocol 5.10 is similar to that in Camenisch et al. [9]: The prover first re-randomizes the signature to obtain a value $s$, which it sends to the verifier. Subsequently, the prover and the verifier run a standard Schnorr proof for the resulting statement.

**Theorem 5.11.** *Protocol 5.10 is a perfectly HVZK $\Sigma$-proof of knowledge for the following relation:*

$$\mathcal{R} = \left\{ ((spk, (\widetilde{cmt}_i)_{i=0}^{n-1}), (\widetilde{sig}, m, r, (m_i, r_i)_{i=0}^{n-1})) : m = \sum_{i=0}^{n-1} 2^i m_i \wedge \right.$$
$$\left. \widetilde{cmt}_i = m_i\tilde{g} + r_i\tilde{h} \ \wedge \ \mathsf{SVerify}(spk, m, \widetilde{sig}) = \mathtt{accept} \right\}.$$

*The protocol is perfectly complete, and has a knowledge error of $1/\tilde{q}$. Furthermore, the protocol has quasi unique responses (under the discrete logarithm assumption in $\mathbb{G}$) and high-entropy commitments.*

The proof of this theorem is straightforward and can be found in the full version.

**The Group Signature Scheme.** Combining Protocols 4.1 and 5.10 now directly gives a group signature by the construction from Section 5.3. The $\mathcal{ID}$ is given by $\{0,1\}^\ell$ (which can be identified with $\{0,\ldots,2^\ell-1\}$), where $\ell \leq n$ and $n/\tilde{q}$ is negligible,$n$ is the dimension of the ring being used, and $\tilde{q}$ is the order of the groups of the commitment- and the signature schemes. The condition $n/\tilde{q}$ is just to ensure that with overwhelming probability, $ssk \notin \mathcal{ID}$, so that all signatures of an identity $i \in \mathcal{ID}$ is non-zero and can be used as a witness in Protocol 5.10. The commitment $(\mathsf{CSetup}, \mathsf{Commit}, \mathsf{COpen})$ scheme from Section 5.3, corresponds to the bit-by-bit Pedersen commitments $\widetilde{cmt}_i$.

# References


1. Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., Wichs, D.: Multiparty computation with low communication, computation and interaction via threshold FHE. In: EUROCRYPT 2012. LNCS, vol. 7237, pp. 483–501. Springer (Apr 2012)
2. Asokan, N., Shoup, V., Waidner, M.: Optimistic fair exchange of digital signatures (extended abstract). In: EUROCRYPT. vol. 1403, pp. 591–606. Springer (1998)
3. Ateniese, G., Camenisch, J., Joye, M., Tsudik, G.: A practical and provably secure coalition-resistant group signature scheme. In: CRYPTO 2000. LNCS, vol. 1880, pp. 255–270. Springer (Aug 2000)
4. Bellare, M., Goldreich, O.: On defining proofs of knowledge. In: CRYPTO. pp. 390–420 (1992)
5. Bellare, M., Micciancio, D., Warinschi, B.: Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In: EUROCRYPT 2003. LNCS, vol. 2656, pp. 614–629. Springer (May 2003)
6. Boneh, D., Boyen, X.: Short signatures without random oracles. In: EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer (May 2004)
7. Boneh, D., Boyen, X.: Short signatures without random oracles and the SDH assumption in bilinear groups. Journal of Cryptology 21(2), 149–177 (Apr 2008)
8. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer (Aug 2004)
9. Camenisch, J., Dubovitskaya, M., Neven, G.: Oblivious transfer with access control. In: ACM Conference on Computer and Communications Security. pp. 131–140 (2009)
10. Camenisch, J., Neven, G., Rückert, M.: Fully anonymous attribute tokens from lattices. In: SCN 12. LNCS, vol. 7485, pp. 57–75. Springer (Sep 2012)
11. Chase, M., Lysyanskaya, A.: On signatures of knowledge. In: CRYPTO 2006. LNCS, vol. 4117, pp. 78–96. Springer (Aug 2006)
12. Chaum, D., van Heyst, E.: Group signatures. In: EUROCRYPT'91. LNCS, vol. 547, pp. 257–265. Springer (Apr 1991)
13. Cramer, R.: Modular Design of Secure yet Practical Cryptographic Protocols. Ph.D. thesis, CWI and University of Amsterdam (1997)
14. Damgård, I.: Efficient concurrent zero-knowledge in the auxiliary string model. In: EUROCRYPT. pp. 418–430 (2000)
15. Damgård, I.: On $\Sigma$-Protocols. Lecture on Cryptologic Protocol Theory; Faculty of Science, University of Aarhus (2010)
16. Damgård, I., Fujisaki, E.: A statistically-hiding integer commitment scheme based on groups with hidden order. In: ASIACRYPT. pp. 125–142 (2002)
17. Damgård, I., Goldreich, O., Okamoto, T., Wigderson, A.: Honest verifier vs dishonest verifier in public coin zero-knowledge proofs. In: CRYPTO. pp. 325–338 (1995)
18. Damgård, I., Pastro, V., Smart, N.P., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: CRYPTO. pp. 643–662 (2012)

19. Fujisaki, E., Okamoto, T.: Statistical zero knowledge protocols to prove modular polynomial relations. In: CRYPTO. pp. 16–30 (1997)
20. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: EUROCRYPT. pp. 1–17 (2013)
21. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC. pp. 169–178 (2009)
22. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC. pp. 197–206 (2008)
23. Gordon, S.D., Katz, J., Vaikuntanathan, V.: A group signature scheme from lattice assumptions. In: ASIACRYPT 2010. LNCS, vol. 6477, pp. 395–412. Springer (Dec 2010)
24. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A ring-based public key cryptosystem. In: ANTS. pp. 267–288 (1998)
25. Laguillaumie, F., Langlois, A., Libert, B., Stehlé, D.: Lattice-based group signatures with logarithmic signature size. In: ASIACRYPT 2013, Part II. LNCS, vol. 8270, pp. 41–61. Springer (Dec 2013)
26. Ling, S., Nguyen, K., Stehlé, D., Wang, H.: Improved zero-knowledge proofs of knowledge for the isis problem, and applications. In: Public Key Cryptography. pp. 107–124 (2013)
27. Lyubashevsky, V.: Lattice-based identification schemes secure under active attacks. In: Public Key Cryptography. pp. 162–179 (2008)
28. Lyubashevsky, V.: Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In: ASIACRYPT. pp. 598–616 (2009)
29. Lyubashevsky, V.: Lattice signatures without trapdoors. In: EUROCRYPT. pp. 738–755 (2012)
30. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. J. ACM 60(6), 43 (2013), preliminary version appeared in EUROCRYPT 2010
31. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: 22nd ACM STOC. pp. 427–437. ACM Press (May 1990)
32. Pedersen, T.P.: Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In: CRYPTO. pp. 129–140 (1991)
33. Sahai, A.: Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In: 40th FOCS. pp. 543–553. IEEE Computer Society Press (Oct 1999)
34. Santis, A.D., Persiano, G.: Zero-knowledge proofs of knowledge without interaction (extended abstract). In: 33rd FOCS. pp. 427–436. IEEE Computer Society Press (Oct.)
35. Stehlé, D., Steinfeld, R.: Making NTRU as secure as worst-case problems over ideal lattices. In: EUROCRYPT. pp. 27–47 (2011)
36. Stehlé, D., Steinfeld, R., Tanaka, K., Xagawa, K.: Efficient public key encryption based on ideal lattices. In: ASIACRYPT. pp. 617–635 (2009)
37. Stern, J.: A new identification scheme based on syndrome decoding. In: CRYPTO. pp. 13–21 (1993)