How to Avoid Repetitions in Lattice-based Deniable Zero-Knowledge Proofs

Xavier Arnal[1], Abraham Cano[1], Tamara Finogina[1,2], and Javier Herranz[1]

[1] Dept. Matemàtiques, Universitat Politècnica de Catalunya
Barcelona, Spain
xavier.arnal@upc.edu,abraham.cano@upc.edu,javier.herranz@upc.edu
[2] Scytl Election Technologies, S.L.U.,
Barcelona, Spain
tamara.finogina@scytl.com

Abstract. Interactive zero-knowledge systems are a very important cryptographic primitive, used in many applications, especially when deniability (also known as non-transferability) is desired. In the lattice-based setting, the currently most efficient interactive zero-knowledge systems employ the technique of rejection sampling, which implies that the interaction does not always finish correctly in the first execution; the whole interaction must be re-run until abort does not happen.

While repetitions due to aborts are acceptable in theory, in some practical applications it is desirable to avoid re-runs for usability reasons. In this work we present a generic technique that departs from an interactive zero-knowledge system (that might require multiple re-runs to complete the protocol) and obtains a 3-moves zero-knowledge system (without re-runs). The transformation combines the well-known Fiat-Shamir technique with a couple of initially exchanged messages. The resulting 3-moves system enjoys honest-verifier zero-knowledge and can be easily turned into a fully deniable proof using standard techniques. We show some practical scenarios where our transformation can be beneficial and we also discuss the results of an implementation of our transformation.

1 Introduction

In some applications, it is mandatory to ensure that the result of an interaction between a prover and a verifier is non-transferable, i.e., deniable. For example, electronic voting requires ballot content correctness proofs (i.e., proof that the ballot indeed contains the option voter selected) to be non-transferable to prevent vote selling. It means no one except the voter should be able to realize that valid content correctness proof is coming from the voting device and not simulated. Similarly, a deniable authentication should produce a proof that convinces only protocol participants and no one else.

Informally deniability means that only the verifier who interacted with the prover can be convinced that some statement is true. No one else looking at the same transcript should be able to say if it comes from a real interaction or from one simulated by the verifier. At first glance, it might seem that a standard notion of zero-knowledge (ZK) definition, which requires the existence of a simulator algorithm that can produce transcripts indistinguishable from the ones produced in a real execution, suffices for achieving deniability. However, it is not always the case. For deniability, it is crucial that the verifier can run such a simulator algorithm in the real life; definitions of the ZK notion where the simulator controls a random oracle or a common reference string are thus not satisfactory.

Let us consider for instance non-interactive proofs that are ZK in the random oracle model (ROM). To simulate the transcript, one needs to have control over the hash oracle, which is impossible in real-life settings. Hence, ZK in ROM results in only theoretical deniability: any valid non-interactive proof is, without a doubt, coming from the prover.

The impossibility of achieving deniability with a one-round protocol (in ROM or CRS model) was proven already by Rafael Pass [24]. Also, he argued that two rounds are necessary and sufficient for achieving deniability in ROM. His solution for proving that some public x belongs to some language $\mathcal{L}_{\mathcal{R}}$ (borrowing notation from Section 2) was to use a two-rounds proof of the following form: in the first round, the verifier generates a commitment to a trapdoor and a non-interactive zero-knowledge proof of knowledge (NIZKPoK) of the trapdoor. In the second round, the prover verifies the received NIZKPoK and, if valid, creates an OR proof for the statement "$x \in \mathcal{L}_{\mathcal{R}}$ OR I know the verifier's trapdoor". We, however, claim that this solution is non-deniable. A malicious verifier can easily copy an existing commitment and corresponding NIZKPoK from some other party/execution and, therefore, engage in a protocol without knowing the trapdoor. Later such a verifier can somehow demonstrate that it could not have known the trapdoor, which makes the origin of any valid proof undeniable.

Therefore, it seems natural that, for deniability, we need a protocol with full zero-knowledge (ZK)[3] in the plain model (with no extra assumptions like a random oracle, a common reference string, a public key infrastructure, etc.). Achieving ZK in the plain model requires at least three rounds of interaction. Furthermore, existing 3-rounds ZK systems

[3] Honest-verifier zero-knowledge (HVZK) is not enough, since the notion of deniability is intrinsically related to a dishonest verifier who could be interested in transferring its conviction to somebody else.

are quite theoretical and inefficient; we will thus focus on achieving efficient and 4-rounds ZK systems.

Two classical ways of doing so (and thus to obtain a deniable proof system), assuming the existence of an interactive proof system Π for proving $x \in \mathcal{L}_{\mathcal{R}}$, are:

(a) If Π is a 3-rounds (Sigma) protocol, where transcripts have the form (a, c, z), then add an initial round where the verifier commits to the challenge c. In round 3 now the verifier opens the commitment; the prover sends z only if the opening is correct. Soundness and ZK hold if the commitment is perfectly hiding and trapdoor.

(b) The verifier sends Y in the first round, where $Y = F(X)$ for some homomorphic one-way function F and input X. It also engages a Sigma protocol for proving knowledge of X, which is finished in the two next rounds. In round 4, if the Sigma protocol has been successful, the prover computes a non-interactive (via Fiat-Shamir) proof for the statement "$x \in \mathcal{L}_{\mathcal{R}}$ OR I know X".

The lattice-based setting. Both solutions (a) and (b) above require execution of a Sigma protocol for some language. In the lattice-based setting (the most promising one for achieving security in front of quantum computers, through the hardness of some shortest vector problems in lattices [2] such as the learning with errors (LWE) problem and the short integer solution (SIS) problem) such Sigma protocols are not trivial to construct: the security of LWE and SIS requires that the solution not only has a specific structure but also is small. Thus, a masking term has to be small, but that unavoidably leaks parts of the secret.

A solution to this problem was proposed by Lyubashevsky in [18]. He proposed the smart idea of a (possibly) aborting prover, using rejection sampling to ensure that the answer's distribution is independent of the secret. The rejection sampling allowed to ensure correctness and security and led to many fundamental cryptographic constructions: canonical identification (CID) and signatures (e.g. [18]), zero-knowledge proofs (e.g. [20]), blind signatures (e.g. **BLAZE**+ [4]), and others.

The main downside of the idea is the possibility of multiple protocol repetitions that may be very undesirable in practical applications. Real people expect to interact with a system only once and always receive the (correct) result at the end. Hence, an unpredictable number of repeats due to rejection sampling makes deniable protocols unpractical.

Unfortunately, eliminating aborts in lattice-based ZK proofs is quite challenging. Behnia et al. [8] studied rejection conditions in Lyubashevsky's

CID scheme and found a way to remove one of the two conditions. However, they concluded that full elimination of rejection sampling is problematic.

An alternative is to reduce the occurrence of protocol re-runs. One of the simplest methods to ensure the protocol terminates after a fixed number of repetitions $M \geq 2$ (with a high probability) is to use a large enough distribution over $\mathbb{Z}$. However, this comes at the cost of increased execution time and proof size.

Another method to decrease the occurrence of aborts is to run several protocols in parallel. Usually, parallel repetition aims to reduce the knowledge error rather than the completeness error. However, we can use it for decreasing rejects as well. Prover starts N independent instances of the protocol and sends N commitments to the verifier, replying only with the first proof that did not cause an abort. While this increases the probability of successful protocol termination, it significantly increases communication and computational complexity (by a factor of N) and does not eliminate aborts completely.

Another work by Attema et al. [5] proposes an s-out-of-t threshold parallel repetition, where the verifier accepts if s out of t of the parallel instances are accepting. Unfortunately, the completeness error is still $\geq \rho^t$, where ρ is the completeness error of a single protocol run. Therefore, achieving a negligible probability of aborts would require a substantial t and would result in an increased proof size and proving time.

An improvement over parallel repetition — a generic construction for reducing aborts in 3-moves protocols — was proposed in [4]. This construction builds on top of the idea of ℓ parallel repetitions and uses (unbalanced) binary hash trees to reduce the size of the first answer, from ℓ-commitments to a tree root.

An alternative to the aborts approach is probabilistically checkable proofs (PCPs) and interactive oracle proofs (IOPs) cleverly combined with lattice-based algebraic techniques. For example, [10] presents a zero-knowledge system for proving knowledge of Learning With Errors (LWE) pre-images, which does not involve aborts. Unfortunately, this solution is more efficient than a general lattice-based system (with aborts) only for some specific settings, for instance, when proving at the same time knowledge of a lot of LWE pre-images with the same matrix $\boldsymbol{A}$. In other cases, the initial commitment and Merkle paths result in bigger proofs than possible alternatives, especially considering that we need several iterations to achieve negligible soundness.

All in all, the currently most efficient and compact interactive zero-knowledge systems in the lattice-based setting are those with aborts and, so far, there is no efficient way to eliminate them.

1.1 Our Contribution

In this work, we show a simple way to construct a deniable lattice-based proof system that always requires just a single execution for completeness. We depart from any multi-round interactive lattice-based zero-knowledge systems (possibly with re-runs due to the presence of aborts). We demonstrate the security and effectiveness of our construction and its applicability to a wide class of protocols. In particular, we have implemented the system that results from applying our construction to the system proposed in [11].

The general idea of the transformation is to apply the Fiat-Shamir transformation to the original system Π, combined with an initial message by the prover; the challenges of the non-interactive Fiat-Shamir version of Π will not be simply the outputs of the hash functions (as in Fiat-Shamir), but a combination (a sort of trapdoor commitment) of these outputs with the values sent by the prover in the initial message. This allows us to prove that the resulting 3-rounds protocol enjoys the honest-verifier zero-knowledge (HVZK) property.

At this point, we can use both (a) and (b) solutions depicted above: if the system Π was for proving the desired statement, then we can use (a); otherwise, if Π is for proving the knowledge of a pre-image for a lattice-based homomorphic one-way function, we can use (b).

1.2 Illustrating Our Technique

We show how to eliminate the necessity of the protocol re-runs using the lattice-based CID scheme [18] as an example (see Fig. 1). First, we briefly recall the CID scheme and then show how to apply our transformation.

Let $\boldsymbol{A}$ be a public matrix selected uniformly at random from $\mathbb{Z}_q^{n\times m}$. The prover $\mathcal{P}$ would like to prove the knowledge of a secret matrix $\boldsymbol{S} \in \mathbb{Z}^{m\times n}$ with small entries such that $\boldsymbol{B} = \boldsymbol{A}\cdot\boldsymbol{S} (\bmod\ q)$, where $\boldsymbol{B} \in \mathbb{Z}_q^{n\times n}$ is also public. To do so, $\mathcal{P}$ samples a fresh masking vector $\boldsymbol{y}$ from χ^m, where χ is some distribution over $\mathbb{Z}$ (the discrete Gaussian over $\mathbb{Z}$ or the uniform over a small subset of $\mathbb{Z}$). Then it sends commitment $\boldsymbol{v} = \boldsymbol{A}\cdot\boldsymbol{y} (\bmod\ q)$ to the verifier $\mathcal{V}$. The $\mathcal{V}$ picks a random challenge from the challenge space $\mathcal{C} = \{(c_1,\ldots,c_n) \in \mathbb{Z}^n : c_i \in \{-1,0,1\}, \sum_{i=1}^n |c_i| = \kappa\}$. The $\mathcal{P}$ returns response $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{c}\cdot\boldsymbol{S}$ to the challenge only if rejection sampling algorithm $\mathsf{RejSampl}(\boldsymbol{z})$ does not abort. The protocol is repeated by sampling a fresh $\boldsymbol{y}$ until $\mathsf{RejSampl}$ accepts. The verifier $\mathcal{V}$ accepts if and only if $\boldsymbol{v} = \boldsymbol{A} \cdot \boldsymbol{z} - \boldsymbol{B} \cdot \boldsymbol{c} (\text{mod } q)$ and $||\boldsymbol{z}||_p$ is smaller than a pre-defined bound B, where $p \in \{2, \infty\}$ depending on the distribution χ.

In our construction, a proof is generated non-interactively (via the Fiat-Shamir transformation) and turned back into an interactive one with the help of a very simple trapdoor commitment:

1. To prove a statement st, $\mathcal{P}$ samples a value $\mathsf{r} \in \{0,1\}^\ell$ at random and sends it to the verifier $\mathcal{V}$.
2. $\mathcal{V}$ sends to $\mathcal{P}$ a random challenge $\gamma \in \{0,1\}^\ell$.
3. $\mathcal{P}$ runs a non-interactive version of the CID scheme (if necessary, re-running it until abort does not happen) to get a typical proof $(\mathsf{com}, \mathbf{e}, \mathsf{z})$; but the challenge $\mathbf{e}$ is defined as $\mathbf{e} = H_1\big(\mathsf{r} \oplus \mathsf{H}(\mathsf{st}, \mathsf{com}, \gamma)\big)$ instead of the usual $\mathsf{H}(\mathsf{st}, \mathsf{com})$ in the Fiat-Shamir transformation.

Fig. 1 gives an example of a singe-run (without re-runs) CID scheme. Note that in case of aborting, a fresh $\boldsymbol{y}$ is sampled and the process is repeated until $\mathsf{RejSampl}(\boldsymbol{z})$ accepts, ensuring $\boldsymbol{z}$ is statistically indistinguishable from $\mathbf{e} \cdot \boldsymbol{S}$.

Intuitively, we see that security is inherited from the non-interactive version of the protocol. On the one hand, $\mathcal{P}$ commits to r prior to receiving the verifier's challenge γ, thus it cannot manipulate the non-interactive challenge $\mathbf{e}$. Therefore the resulting proof behaves as a standard non-interactive version of the initial (interactive) protocol. On the other hand, thanks to the use of the simple trapdoor commitment, anyone can generate a simulated transcript that is indistinguishable from the real one.

In the protocol described in Fig. 1, $H : \{0,1\}^* \to \{0,1\}^\ell$ and $H_1 : \{0,1\}^\ell \to \mathcal{C}$ denote two hash functions and $\oplus$ denotes component-wise XOR operation between two strings of ℓ bits.

2 Preliminaries: (Public Coin) Interactive Proofs

Let $\mathcal{R} \subset \{0,1\}^* \times \{0,1\}^*$ be a binary relation. If a pair $(x, w) \in \mathcal{R}$, we call x an statement and w a witness for x. The relation is an NP-relation if, given (x, w), one can decide in polynomial time if $(x, w) \in \mathcal{R}$ or not. Such a relation $\mathcal{R}$ gives rise to the set of "yes"-instances defined as $\mathcal{L}_\mathcal{R} = \{x \in \mathcal{X} \mid \exists w \in \mathcal{W} \; s.t. \; (x, w) \in \mathcal{R}\}$, known as the language of $\mathcal{R}$. The set of witnesses for a valid statement $x \in \mathcal{L}_\mathcal{R}$ is denoted as $R(x)$.

$$
\begin{array}{lcr}
\mathcal{P}(\boldsymbol{A},\boldsymbol{B},\boldsymbol{S}) & & \mathcal{V}(\boldsymbol{A},\boldsymbol{B}) \\
\mathsf{r} \xleftarrow{\$} \{0,1\}^{\ell} & & \\
 & \xrightarrow{\quad \mathsf{r} \quad} & \\
 & & \boldsymbol{\gamma} \xleftarrow{\$} \{0,1\}^{\ell} \\
 & \xleftarrow{\quad \boldsymbol{\gamma} \quad} & \\
\left\{ \begin{array}{l}
\boldsymbol{y} \xleftarrow{\$} \chi^{m} \\
\boldsymbol{v} = \boldsymbol{A} \cdot \boldsymbol{y} \\
\mathbf{e} = H_1\big(\mathsf{r} \oplus \mathsf{H}\big(\boldsymbol{A},\boldsymbol{B},\boldsymbol{v},\mathsf{aux},\gamma\big)\big) \\
\boldsymbol{z} = \boldsymbol{y} + \mathbf{e} \cdot \boldsymbol{S} \\
\text{If } \mathsf{RejSampl}(\boldsymbol{z}) \text{ does not accept, repeat.}
\end{array} \right. & & \\
 & \xrightarrow{\quad \boldsymbol{z}, \mathbf{e} \quad} & \\
 & & \boldsymbol{v} = \boldsymbol{A} \cdot \boldsymbol{z} - \boldsymbol{B} \cdot \mathbf{e} \\
 & & \mathbf{e}' = H_1\big(\mathsf{r} \oplus \mathsf{H}\big(\boldsymbol{A},\boldsymbol{B},\boldsymbol{v},\mathsf{aux},\gamma\big)\big) \\
 & & \text{Accept if:} \\
 & & \mathbf{e}' = \mathbf{e} \\
 & & ||\boldsymbol{z}||_p \leq B
\end{array}
$$

Fig. 1. Interactive CID scheme that always terminates after a single run.

Interactive Proofs. An interactive proof system Π for relation $\mathcal{R}$ is an interactive protocol between two probabilistic polynomial-time (PPT) algorithms, the prover $\mathcal{P}$ and the verifier $\mathcal{V}$. The common input of the two parties is a statement x, whereas $\mathcal{P}$ has as an additional input a witness $w \in R(x)$. We thus denote an execution of such a protocol as $\langle \mathcal{P}(y), \mathcal{V}(y) \rangle_{\Pi}$. The final output of the protocol is a bit — 1 if $\mathcal{V}$ accepts, 0 otherwise. The set of messages exchanged during the execution of Π is called an (accepting or rejecting) transcript.

We will consider in this work a specific but very common type of interactive proof systems: those where the first and last messages are sent by $\mathcal{P}$, leading to $(2\mu+1)$ rounds of communication, for some integer $\mu \geq 1$. We will be considering *public coin* systems: all the random choices of $\mathcal{V}$ are made public during the execution of Π. This is equivalent to say that the $2i$-th message of the protocol, sent by $\mathcal{V}$ to $\mathcal{P}$, is a random element $c_i \leftarrow_R \mathcal{C}_i$, called a challenge, taken from some challenge space(s) $\mathcal{C}_i$.

The first property that must be required to such an interactive system is δ-completeness: if $(x,w) \in \mathcal{R}$ then it holds $Pr[\langle \mathcal{P}(x,w), \mathcal{V}(x) \rangle_{\Pi} = 1] = 1 - \delta$.

Zero-Knowledge. A public coin interactive protocol Π as above enjoys the *honest-verifier zero-knowledge* (HVZK) property if there exists a PPT algorithm $\mathtt{M}_{\Pi}$ such that, for any $(x,w) \in \mathcal{R}$, on input x and μ challenge values $c_1, \ldots, c_\mu$ with $c_i \in \mathcal{C}_i$, outputs an accepting transcript with the same distribution as the one produced by an execution of $\langle \mathcal{P}(x,w), \mathcal{V}(x)\rangle_\Pi$ run with a honest verifier $\mathcal{V}$ that has chosen those challenges $c_i \leftarrow_R \mathcal{C}_i$, for $i = 1, \ldots, \mu$.

A stronger notion is *full zero-knowledge* (ZK), which clearly implies deniability; it requires that, for every verifier $\mathcal{V}^*$ there exists a PPT simulator $\mathtt{M}_{\mathcal{V}^*,\Pi}$ such that for every $(x,w) \in \mathcal{R}$ the output $\langle \mathcal{P}(x,w), \mathcal{V}^*(x)\rangle_\Pi$ is identically distributed to the output $\mathtt{M}_{\mathcal{V}^*}(x)$. This property can be relaxed requiring that the outputs only be statistically or computationally indistinguishable.

(Knowledge) Soundness. A protocol Π has the ϵ-soundness property if, for any $x \notin \mathcal{L}_\mathcal{R}$, it holds $Pr[\langle \mathcal{P}(x), \mathcal{V}(y)\rangle_\Pi = 1] \leq \epsilon$.

There is a stronger version of soundness — that of knowledge soundness. A protocol Π enjoys *knowledge soundness* with knowledge error $\kappa : \mathbb{N} \to [0,1]$ if there exist a positive polynomial $q(\cdot)$ and algorithm $\mathtt{K}$, such that for every prover $\mathcal{P}^*$ and $x \in \mathcal{L}_\mathcal{R}$, the extractor $\mathtt{K}$, on input x, with black-box oracle access to $\mathcal{P}^*$ and within an expected number of steps polynomial in $|x|$, outputs a witness $w \in R(x)$ with probability at least

$$\frac{Pr[\langle \mathcal{P}(x,w), \mathcal{V}(x)\rangle_\Pi = 1] \;-\; \kappa(|x|)}{q(|x|)}$$

3 The Transformation

Let $\Pi = \langle \mathcal{P}(x,\omega), \mathcal{V}(x)\rangle_\Pi$ be a public coin $(2\mu+1)$-rounds interactive proof system for language $\mathcal{L}_\mathcal{R}$. We denote as a_i the message sent by $\mathcal{P}$ to $\mathcal{V}$ in round $2i-1$, for $i = 1, \ldots, \mu$, and as z the last message sent by $\mathcal{P}$ in round $2\mu+1$. The message sent by $\mathcal{V}$ in round $2i$ is a random challenge $c_i \in \mathcal{C}_i$, for some challenge space $\mathcal{C}_i$, for $i = 1, \ldots, \mu$.

Let us consider $1+\mu$ hash functions: on the one hand $H : \{0,1\}^* \to \{0,1\}^\ell$ and on the other hand $H_i : \{0,1\}^\ell \to \mathcal{C}_i$, for $i = 1, \ldots, \mu$.

We construct a 3-rounds interactive proof system $\Sigma = \langle \mathcal{P}(x,w), \mathcal{V}(x)\rangle_\Sigma$ for the same language $\mathcal{L}_\mathcal{R}$, as follows.

1. For $i = 1, \ldots, \mu$, $\mathcal{P}$ chooses $r_i \in \{0,1\}^\ell$ uniformly at random. These values $r_1, \ldots, r_\mu$ are sent to $\mathcal{V}$.
2. $\mathcal{V}$ chooses a challenge $\gamma \in \{0,1\}^\ell$ uniformly at random and sends it to $\mathcal{P}$.
3. $\mathcal{P}$ runs an execution of the system Π by using inputs (x,ω), and playing also the role of the verifier, by defining the challenges as $c_i =$

$H_i(r_i \oplus h_i)$, where $h_i = H(x, a_1, \ldots, a_i, c_1, \ldots, c_{i-1}, \gamma)$, for $i = 1, \ldots, \mu$. The resulting transcript $(a_1, a_2, \ldots, a_\mu, z)$ is sent by $\mathcal{P}$ to $\mathcal{V}$.

$\mathcal{V}$ accepts the interaction as valid if $(a_1, c_1, a_2, c_2, \ldots, a_\mu, c_\mu, z)$ is an accepting transcript for Π with input x, where $c_i = H_i(r_i \oplus h_i)$ and $h_i = H(x, a_1, \ldots, a_i, c_1, \ldots, c_{i-1}, \gamma)$, for $i = 1, \ldots, \mu$.

3.1 Security Analysis

The completeness property of Σ is trivially satisfied, assuming the interactive system Π enjoys completeness. In the next sections we show how the zero-knowledge and soundness properties of Π are also inherited by Σ.

Zero-Knowledge

Proposition 1. *Assuming Π enjoys the honest-verifier zero-knowledge (HVZK) property, then the new interactive system Σ also enjoys the HVZK property.*

Proof. The goal is to show that, for any $(x, w) \in \mathcal{L}_\mathcal{R}$, a simulator algorithm $\mathtt{M}_\Sigma$ can, on input x and any (honest) random challenge $\gamma \in \{0,1\}^\ell$, produce transcripts $(r_1, \ldots, r_\mu, \gamma, a_1, \ldots, a_\mu, z)$ indistinguishable from those produced by an execution of $\langle \mathcal{P}(x, w), \mathcal{V}(x) \rangle_\Sigma$ with a honest verifier $\mathcal{V}$ which takes that γ uniformly at random in $\{0,1\}^\ell$.

By hypothesis, there is a simulator $\mathtt{M}_\Pi$ for Π. What the simulator $\mathtt{M}_\Sigma$ does first is to choose uniformly at random μ values $v_1, \ldots, v_\mu \xleftarrow{\$} \{0,1\}^\ell$ and to compute $c_i = H_i(v_i)$ for $i = 1, \ldots, \mu$. Then $\mathtt{M}_\Pi$ runs simulator $\mathtt{M}_\Pi$ with input x and challenges $c_1, \ldots, c_\mu$, which results in an accepting transcript $(a_1, c_1, a_2, c_2, \ldots, a_\mu, c_\mu, z)$, indistinguishable from those produced by $\langle \mathcal{P}(x, w), \mathcal{V}(x) \rangle_\Pi$. After that $\mathtt{M}_\Sigma$ computes the values $h_i = H_i(x, a_1, \ldots, a_i, c_1, \ldots, c_{i-1}, \gamma)$ and $r_i = v_i \oplus h_i$, for $i = 1, \ldots, \mu$.

It is easy to check that the transcript has the same distribution as those produced in a real execution of $\langle \mathcal{P}(x, w), \mathcal{V}(x) \rangle_\Sigma$ where γ is the challenge chosen by the honest verifier.

Assuming hash functions H_i are pseudo-random functions, the values $c_i = H_i(v_i)$ generated by $\mathtt{M}_\Sigma$ and given as inputs to $\mathtt{M}_\Pi$ are random and uniform elements in $\mathcal{C}_i$. $\square$

If we combine this 3-rounds and HVZK protocol Σ with a trapdoor and perfectly hiding commitment scheme (as scheduled in item (a) in the

Introduction), we can obtain full zero-knowledge in the plain model and thus, deniability.

Alternatively, we can apply our construction to a protocol Π for proving knowledge of a pre-image X for some value $Y = F(X)$, where F is a lattice-based homomorphic one-way function. The resulting protocol Σ can then be used in the solution depicted in item (b) in the Introduction.

Soundness

Proposition 2. *Assuming Π has ϵ-soundness and if ℓ is big enough, then the new interactive system Σ has the ϵ'-soundness, in the (classical) Random Oracle Model, where $\epsilon' \leq \epsilon \cdot Q^{\mu}$ and Q is an upper bound on the number of hash queries that a prover of Σ can make.*

Proof. The proof of this result works in a similar way as the well-known (in its naive, non-optimized version) proof that the Fiat-Shamir transformation of a public-coin interactive system with soundness results in a secure non-interactive system: the idea is to rewind the adversary several (in our case, μ times), by fixing the randomness and the answers to the hash queries up to a specific point, and then to use the Forking Lemma [25] to ensure that, with non-negligible probability, all the instances of the adversary will lead to forgeries with the desired outputs (that have been fixed in the rewinds).

First of all, if ℓ is big enough, then the probability $2^{-\ell}$ of breaking soundness by guessing the challenge $\gamma \in \{0,1\}^{\ell}$ is negligible. In that setting, let us assume that Σ still does not have ϵ'-soundness. Thus, there exists a prover $\mathcal{P}_{\Sigma}$ that is accepted with probability $> \epsilon'$, when run with some instance $x' \notin \mathcal{L}_{\mathcal{R}}$. We are going to construct a prover $\mathcal{P}_{\Pi}$ against the soundness of Π, running thus with the same $x' \notin \mathcal{L}_{\mathcal{R}}$.

As its first instruction, $\mathcal{P}_{\Pi}$ starts running $\mathcal{P}_{\Sigma}$, which sends its first message $(r_1, \ldots, r_{\mu})$. Now $\mathcal{P}_{\Pi}$ chooses at random $\gamma \in \{0,1\}^{\ell}$ and sends it to $\mathcal{P}_{\Sigma}$. We remark that $(r_1, \ldots, r_{\mu})$ and γ are going to be fixed for all the calls that $\mathcal{P}_{\Pi}$ makes to $\mathcal{P}_{\Sigma}$. In this first call, $\mathcal{P}_{\Sigma}$ gives its final answer $(a_1^{(1)}, \ldots, a_{\mu}^{(1)}, z^{(1)})$, which is valid with probability $\geq \epsilon'$.

During this and the other executions of $\mathcal{P}_{\Sigma}$, our new prover $\mathcal{P}_{\Pi}$ has to answer the hash queries made by $\mathcal{P}_{\Sigma}$. This is done in the usual way, by keeping track of all previous queries, selecting a random output for new queries, storing the (input,output) relations in a table, etc. With overwhelming probability, a successful prover $\mathcal{P}_{\Sigma}$ will have made all the key queries $h_i \leftarrow H(x', a_1^{(1)}, \ldots, a_i^{(1)}, c_1^{(1)}, \ldots, c_{i-1}^{(1)}, \gamma)$ and $H_i(r_i + h_i)$, for $i = 1, \ldots, \mu$.

After the first execution, $\mathcal{P}_\Pi$ sends the value $a_1^{(1)}$ to its verifier $\mathcal{V}_\Pi$, which then sends a challenge c_1. With overwhelming probability, it will be the case that $c_1 \neq H_1(r_1 \oplus h_1)$. What $\mathcal{P}_\Pi$ does now is to rewind: it starts a new running of $\mathcal{P}_\Sigma$, with the same random tape and the same answers to the hash queries, up to the point where the query $H_1(r_1 \oplus h_1)$ is made; this time, the answer to this query is defined as c_1. The Forking Lemma ensures that, with non-negligible probability, this second execution of $\mathcal{P}_\Sigma$ will produce a valid transcript $(a_1^{(1)}, a_2^{(2)}, \dots, a_\mu^{(2)}, z^{(2)})$ with the same value $a_1^{(1)}$ as in the first execution (because, with overwhelming probability, the value $a_1^{(1)}$ had been queried to hash oracle H to produce h_1, before the key query $H_1(r_1 \oplus h_1)$ was made). At this point, $\mathcal{P}_\Pi$ sends the value $a_2^{(2)}$ to its verifier $\mathcal{V}_\Pi$, which then sends a challenge c_2.

The same rewind argument is done again, with the same random tape and hash answers as in the second execution, but now defining $H_2(r_2 \oplus h_2)$ to be c_2. Again with overwhelming probability this query, which depends on h_2 which depends on c_1, must have been made after the query $H_1(r_1 \oplus h_1)$, which is again answered as c_1. With non-negligible probability, this third execution of $\mathcal{P}_\Sigma$ produces a valid transcript $(a_1^{(1)}, a_2^{(2)}, a_3^{(3)}, \dots, a_\mu^{(3)}, z^{(3)})$.

Repeating this argument μ times, letting $\mathcal{P}_\Pi$ send $a_i^{(i)}$ to its verifier $\mathcal{V}_\Pi$ in round i, getting c_i as answer and rewinding $\mathcal{P}_\Sigma$ accordingly, at the end we eventually finish, after $\mu + 1$ executions of $\mathcal{P}_\Sigma$, with a valid transcript $(a_1^{(1)}, a_2^{(2)}, a_3^{(3)}, \dots, a_\mu^{(\mu)}, z^{(\mu+1)})$ satisfying $c_i = H_i(r_i \oplus h_i)$, where $h_i = H_i(x', a_1^{(1)}, \dots, a_i^{(i)}, c_1, \dots, c_{i-1}, \gamma)$. Thus, our $\mathcal{P}_\Pi$ has convinced its verifier $\mathcal{V}_\Pi$ with non-negligible probability ϵ. By the iterated use of the Forking Lemma, the relation between ϵ and ϵ' is essentially $\epsilon \approx \frac{\epsilon'}{Q^\mu}$. $\square$

3.2 Extensions

- The same idea as in the proof for soundness can be applied to prove that knowledge soundness of Π implies knowledge soundness of Σ.
- The soundness property of Σ is obtained in the classical Random Oracle Model. If one wants to achieve soundness in the Quantum Random Oracle Model, then one can use alternative transformations to Fiat-Shamir, either generic [29,12] or specific for lattice-based systems [17], that have been proposed in the last years.
- The naive reduction in our proof for the soundness property implies a loss factor Q^μ which is exponential in the number of rounds of Π. This problem can be solved by using the results in [6], whenever the starting protocol Π enjoys $(k_1, \dots, k_\mu)$-special soundness. We stress that

most (if not all) popular interactive systems Π enjoy this property, including lattice-based ones.

– If the challenge spaces $\mathcal{C}_i$ of the interactive protocol Π are closed spaces for some mathematical operation (that we denote for simplicity as $+$), then a small modification to our construction is possible, basically choosing $r_i \leftarrow_R \mathcal{C}_i$ and then defining $c_i = r_i + h_i$, where $h_i = H_i(x, a_1, \ldots, a_i, h_1, \ldots, h_{i-1}, \gamma)$, being now $H_i : \{0,1\}^* \to \mathcal{C}_i$. This situation happens for instance when Π is the protocol in [30]: the challenge space contains integers modulo a prime p.

4 Applications

The transformation proposed in the previous section is useful in settings where a lattice-based interactive zero-knowledge protocol is mandatory (for instance, if deniability is required), or for some reason preferable to a non-interactive protocol. In such a situation, the most efficient existing protocols Π involve rejection sampling and thus aborts [11,14,30,13,21,20]. Our transformation results in a 3-rounds protocol Σ without mandatory protocol restarts, at the cost of relying on the Random Oracle Model to achieve provable security.

We give below three specific examples of situations where such interactive protocols are used. After that, we discuss other situations where our result in the previous section does not seem applicable.

4.1 Canonical Identification Schemes

Canonical Identification (CID) schemes are three round public coin protocols in which a prover (who sends the first and third messages) proves knowledge of the secret key matching a specific public key. The second message, sent by the verifier, is a random challenge.

Although these schemes are often used as building blocks to design other cryptographic protocols (in particular, signature schemes, with no interaction between the signer and the verifier), they can be used on their own: for instance, in access control systems where the user trying to get access proves to the access entity (the verifier, in this context) that he owns the secret key which matches a public key of some authorized user. If the users want their access to remain private, a solution can be to run a CID scheme, so that the transcript is non-transferable and the (possibly dishonest) access entity cannot prove to someone else that a user got access to the system. An example of the use of such non-transferable identification schemes can be found in [9].

CID schemes are one of the examples considered in the work [4] to motivate their use of trees of commitments, in order to reduce the abort probability of lattice-based interactive zero-knowledge systems. They use Lyubashevsky's identification protocol [18] (recalled in the Section 1.2 of this work) as an illustrative lattice-based CID scheme. Therein, the probability of aborting in a single execution of the protocol is $\approx 1 - \frac{1}{M}$, where $M = \exp\left(\frac{12}{\alpha} + \frac{1}{2\alpha^2}\right)$, being α a lattice parameter that affects the size of the standard deviation σ used to sample the underlying Gaussian distribution: $\alpha = T\alpha$.

There are basically four options[4] if one wants to be sure that the identification protocol will finish with overwhelming probability p_{sc} in three rounds of communication (that is, without forcing the verifier to send more than one message):

1. keep the typically proposed parameters for α, σ, and repeat the protocol, in parallel, at least M times. Here the choice of α will depend on the desired probability p_{sc}. The M repetitions imply that the global communication contains M vectors in the ring $(R_q)^k$;
2. run a single execution of the protocol, but with highly increased parameters α, σ so that M is very close to one;
3. keep the typical values for α, σ and apply the tree of commitments technique introduced in [4], which increases the computational complexity of the prover by a factor ℓ and the communication complexity by $\log(\ell)$ hash values, where ℓ (the number of leaves in the tree) depends on α, p_{sc};
4. apply our transformation to Lyubashevsky's CID scheme, which results in a protocol that always succeeds; the communication complexity of the protocol is almost the same as in the original CID scheme, whereas the computational cost for the prover is essentially the same as in options 1, 2, and 3 above.

Note that option 4 is the only one that ensures that the protocol will always finish successfully. The other advantage of option 4 over the three first options is, of course, its communication complexity. On the negative side, option 4 is the only one that needs the heuristic Random Oracle Model to have provable security.

Some specific values given in Section 3 of [4] are as follows, for $p_{sc} = 1 - 2^{-10}$: option 1 could have $\alpha = 11$ and $M \approx 3$, option 2 should have

[4] We stress that the abort-free protocol in [10] is not really suitable for this setting, in terms of efficiency.

$\alpha > 2^{13.6}$ and option 3 could have $\alpha = 23$ and $\ell = 8$ or alternatively $\alpha = 12$ and $\ell = 16$.

For higher values of p_{sc}, parameters in options 1,2,3 must be increased even more. As an example, authors of [4] show that $\alpha = 42$ and $\ell = 64$ are needed for $p_{sc} = 1 - 2^{-128}$.

4.2 Non-Transferable Signatures

In some kinds of signature schemes that have been introduced in the last decades, the validity of the signature is not universally verifiable as it happens in standard signatures. In contrast, the signer puts some limit on the user(s) who can verify a signature, and also on the capability to transfer this conviction to other users. Examples of this kind of signature schemes are designated verifier signatures, directed signatures, nominative signatures, undeniable signatures, and designated confirmer signatures.

Some of them are aggregated under the name of on-line non-transferable signatures [27]. In such schemes, the signing algorithm is run by the signer, but then there are interactive protocols, Confirm and Disavow, run by both the signer and the verifier, which confirm the verifier of the validity or invalidity of a signature. The verifier cannot convince anybody else of any of these facts. Applications of these kinds of signatures include machine-readable travel documents and identity documents like e-Passports [23,7].

The interaction between signer and verifier typically involves a 3-rounds zero-knowledge system. If one intends to design such schemes in a lattice-based setting, thus, our result in this paper can be directly used as an ingredient of such designs, so that the interaction between signer and verifier needs to be run only once, without the verifier noticing the presence of aborts and without (parallel) repetitions.

As a particular example, the first (and maybe only) secure lattice-based undeniable signature scheme is the one in [26]. The confirmation and disavowal protocols of the scheme are designed by using Stern's techniques [28]: a dishonest prover is accepted with probability 2/3 (soundness error), which means the protocols must be run a large number of times to achieve real soundness. Our techniques, combined with some suitable and efficient lattice-based zero-knowledge system Π for the languages involved in those confirmation / disavowal protocols, would result in protocols Σ with overwhelming soundness, without repetitions due to aborts. There are many options today (see for instance [20] and references therein) to find a suitable and efficient Π for the specific lattice-based languages appearing in the confirmation / disavowal protocols of [26].

4.3 eVoting with CAI and CR Properties

Two important properties of an electronic voting system are cast-as-intended (CAI) verifiability and coercion-resistance (CR). CAI verifiability means that the voter is convinced that the option inside a ciphertext that goes to the ballot box is the one that he/she has chosen, when the ciphertext has been created by an external (possibly dishonest) voting device. Coercion-resistance is achieved if a voter has means of deceiving a coercer who tries to force the voter to act in a specific way during the voting protocol.

In scenarios where voters do not receive secret information (such as credentials) from the election authorities, it has been recently shown [15] that at least three rounds of interaction between the voter and voting device are necessary in order to achieve CAI and CR at the same time. The authors of that paper propose two generic constructions involving four rounds of communication. For instance, in one of the constructions, the interaction is essentially a combination of a commitment scheme (where the voter commits to the challenge that will be used later) and a zero-knowledge system, with honest-verifier zero-knowledge, where the voting device proves knowledge of randomness r such that $\mathsf{Enc}_{pk}(m; r) = c$, for some public parameters pk, m, c: the public key pk of the encryption scheme Enc, the plaintext m which the voting option chosen by the voter and the ciphertext c that will go to the ballot box.

If one wants to instantiate this construction with post-quantum secure tools, one can choose a lattice-based encryption scheme, for instance, one based on the hardness of the Ring Learning With Errors (RLWE) problem [22] and combine it with some of the recent efficient zero-knowledge systems for lattice-based relations [11,14,30,13,21,20]. Since the interactive versions of all these zero-knowledge systems Π involve rejection sampling and aborts, we can apply our transformation to get a 3-rounds system Σ, with honest-verifier zero-knowledge as desired, and without any repetitions. This means that the voter does not need to run many executions of the system (in parallel or not) in order to get convinced that the ciphertext contains the voting option m.

4.4 Settings where Our Result Is Not Useful

We insist once again that the "abort problem" of zero-knowledge systems based on lattices is not an issue if these systems are to be used in the non-interactive version resulting from applying Fiat-Shamir or a similar transformation. In these cases, the party acting as the prover will eventually abort and start the process again, without the final verifier noticing. This happens in a lot of practical uses of these protocols — including standard/group/ring/attribute-based signatures.

A kind of signature that requires interaction is blind signature, where a user wants to obtain a signature by a signer on some message m, without the signer obtaining any information about the message m. Currently, in the setting of lattice-based blind signatures, the tree of commitments technique introduced in [4] to reduce the abort probability has been successfully used a couple of times, first in the same paper [4] as an improvement of the signature scheme BLAZE [3] and then in [16] to construct a provably-secure (in contrast to BLAZE and BLAZE+) but inefficient scheme which involves three rounds of communication.

A natural question is thus: can our $\Pi \to \Sigma$ transformation be applied in the setting of (lattice-based) blind signatures, as it happened with the tree of commitments technique? The answer seems to be no, as a blind signature scheme where the signer proves something using Σ appears to be very far from achieving the blindness property. In any case, a positive answer to the question would result in a blind signature scheme with at least three rounds of communication, which would not improve the state-of-the-art: recently, a couple of schemes involving only two rounds of communication have been proposed in the lattice setting [1,19].

5 Implementation

In this section we present our experimental results of the implementation of our transformation. We applied our transformation to the 5 round protocol of Bootle et al. [11], using a custom-built library for polynomial operations over $Z_q/\langle x^n + 1\rangle$, along with the RustCrypto library for computing SHA2 hashes in Rust.

The tests were performed on an Intel Core i7-10750H CPU. We have performed 1,000 tests over the protocol with and without the transformation. We have found that, when using the parameters proposed by Bootle et al. [11], the mean execution time increases from 20.6 to 21.5 seconds ($\sigma < 0.3$), amounting to an increase of about 5% in execution time. In Fig. 2, we can see the time distribution of the protocol over the executions with (orange) and without (blue) the transformation.

While we have obtained an expected decrease in the performance of a single run of the protocol, we have been able to avoid the need for re-runs and thus achieve an improvement over the whole testing. For completeness, Fig. 3 gives the distribution of the number of aborts produced in another 10,000 tests of the non-transformed protocol with faster parameters ($n = 16$). For instance, more than 6% of the executions required 10 repetitions or more of the protocol; this may be very undesirable in some real-life interactive protocols.

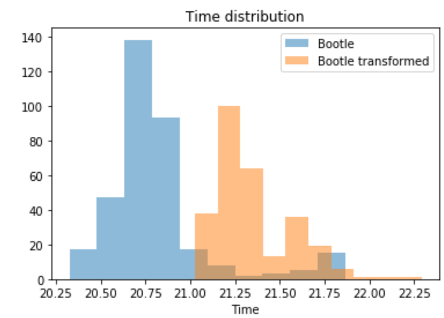

Fig. 2. Time distribution (in seconds) of the 1,000 executions of the first sampling test.

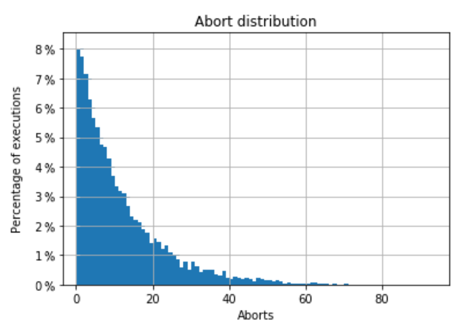

Fig. 3. Percentage of executions in the second sampling test, with 10,000 executions, that suffered i aborts, for each i in the x-axis. For instance, 1.5% of the executions had 20 repetitions with aborts.

6 Conclusion

This work presents a theoretical result related to the cryptographic primitive of interactive zero-knowledge systems: a transformation from any public coin $(2\mu + 1)$-rounds interactive proof system to a public coin 3-rounds proof system for the same language. This result, by itself, may seem not original nor useful at all, because the well-known Fiat-Shamir transformation can transform the same $(2\mu + 1)$-rounds interactive proof system into a non-interactive (one-round) proof system.

But in some practical settings, when proofs must be deniable, non-interactive systems are not a valid solution; in these cases, a 3-rounds solution is essentially optimal. When all this happens in the post-quantum secure setting of lattice-based cryptography, such $(2\mu+1)$-rounds interactive proof systems use to employ the rejection sampling technique, which leads to a non-negligeable number of repetitions of the protocol, before the proof is securely produced. This means that the verifier must be on-line for a while and, in case of required repetitions, choose again different random challenges. All in all, this may lead to an undesirable user-oriented experience, for instance in electronic voting applications.

The most relevant property of our transformation, in the lattice-based setting, is that possible repetitions due to rejection sampling do not affect the verifier: they are done internally by the prover, in the 3rd (and last) round of the proof system. The verifier can choose a single challenge and disconnect; the proof will be produced with 100% probability in any case.

References

1. Shweta Agrawal, Elena Kirshanova, Damien Stehlé, and Anshu Yadav. Can round-optimal lattice-based blind signatures be practical? Cryptology ePrint Archive, Report 2021/1565, 2021. `https://ia.cr/2021/1565`.
2. M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, pages 99–108, New York, NY, USA, 1996. Association for Computing Machinery.
3. Nabil Alkeilani Alkadri, Rachid El Bansarkhani, and Johannes Buchmann. BLAZE: practical lattice-based blind signatures for privacy-preserving applications. In *Financial Cryptography and Data Security, FC 2020*, volume 12059 of *Lecture Notes in Computer Science*, pages 484–502. Springer, 2020.
4. Nabil Alkeilani Alkadri, Rachid El Bansarkhani, and Johannes Buchmann. On lattice-based interactive protocols: An approach with less or no aborts. In *Information Security and Privacy*, pages 41–61, Cham, 2020. Springer International Publishing.
5. Thomas Attema and Serge Fehr. Parallel repetition of $(k_1, \ldots, k_\mu)$-special-sound multi-round interactive proofs. Cryptology ePrint Archive, Paper 2021/1259, 2021. `https://eprint.iacr.org/2021/1259`.
6. Thomas Attema, Serge Fehr, and Michael Klooß. Fiat-shamir transformation of multi-round interactive proofs. *IACR Cryptol. ePrint Arch.*, page 1377, 2021.

7. Fatih Balli, F. Betül Durak, and Serge Vaudenay. Bioid: A privacy-friendly identity document. In Sjouke Mauw and Mauro Conti, editors, *Security and Trust Management, STM 2019*, volume 11738 of *Lecture Notes in Computer Science*, pages 53–70. Springer, 2019.
8. Rouzbeh Behnia, Yilei Chen, and Daniel Masny. On removing rejection conditions in practical lattice-based signatures. In *Post-Quantum Cryptography*, pages 380–398, Cham, 2021. Springer International Publishing.
9. Carlo Blundo, Giuseppe Persiano, Ahmad-Reza Sadeghi, and Ivan Visconti. Improved security notions and protocols for non-transferable identification. In *ESORICS 2008*, volume 5283 of *Lecture Notes in Computer Science*, pages 364–378. Springer, 2008.
10. Jonathan Bootle, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. More efficient amortization of exact zero-knowledge proofs for LWE. In *ESORICS 202*, volume 12973 of *Lecture Notes in Computer Science*, pages 608–627. Springer, 2021.
11. Jonathan Bootle, Vadim Lyubashevsky, and Gregor Seiler. Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. In *Advances in Cryptology - CRYPTO 2019*, volume 11692 of *Lecture Notes in Computer Science*, pages 176–202. Springer, 2019.
12. Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Security of the fiat-shamir transformation in the quantum random-oracle model. In *Advances in Cryptology - CRYPTO 2019*, volume 11693 of *Lecture Notes in Computer Science*, pages 356–383. Springer, 2019.
13. Muhammed F. Esgin, Ngoc Khanh Nguyen, and Gregor Seiler. Practical exact proofs from lattices: New techniques to exploit fully-splitting rings. In *Advances in Cryptology - ASIACRYPT 2020*, volume 12492 of *Lecture Notes in Computer Science*, pages 259–288. Springer, 2020.
14. Muhammed F. Esgin, Ron Steinfeld, Joseph K. Liu, and Dongxi Liu. Lattice-based zero-knowledge proofs: New techniques for shorter and faster constructions and applications. In *Advances in Cryptology - CRYPTO 2019*, volume 11692 of *Lecture Notes in Computer Science*, pages 115–146. Springer, 2019.
15. Tamara Finogina, Javier Herranz, and Enrique Larraia. How (not) to achieve both coercion resistance and cast as intended verifiability in remote evoting. In *CANS 2021*, volume 13099 of *Lecture Notes in Computer Science*, pages 483–491. Springer, 2021.
16. Eduard Hauck, Eike Kiltz, Julian Loss, and Ngoc Khanh Nguyen. Lattice-based blind signatures, revisited. In *Advances in Cryptology - CRYPTO 2020*, volume 12171 of *Lecture Notes in Computer Science*, pages 500–529. Springer, 2020.
17. Shuichi Katsumata. A new simple technique to bootstrap various lattice zero-knowledge proofs to QROM secure nizks. In *Advances in Cryptology - CRYPTO 2021*, volume 12826 of *Lecture Notes in Computer Science*, pages 580–610. Springer, 2021.
18. Vadim Lyubashevsky. Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In *Advances in Cryptology – ASIACRYPT 2009*, pages 598–616, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.
19. Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Efficient lattice-based blind signatures via gaussian one-time signatures. In *PKC 2022*, volume 13178 of *Lecture Notes in Computer Science*, pages 498–527. Springer, 2022.
20. Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. Cryptology ePrint Archive, Report 2022/284, 2022. `https://ia.cr/2022/284`.

21. Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. Shorter lattice-based zero-knowledge proofs via one-time commitments. In *PKC 2021*, volume 12710 of *Lecture Notes in Computer Science*, pages 215–241. Springer, 2021.
22. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *J. ACM*, 60(6):43:1–43:35, 2013.
23. Jean Monnerat, Sylvain Pasini, and Serge Vaudenay. Efficient deniable authentication for signatures. In *ACNS 2009*, volume 5536 of *Lecture Notes in Computer Science*, pages 272–291, 2009.
24. Rafael Pass. On deniability in the common reference string and random oracle model. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, pages 316–337, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.
25. David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *J. Cryptol.*, 13(3):361–396, 2000.
26. Swati Rawal, Sahadeo Padhye, and Debiao He. Lattice-based undeniable signature scheme. *Ann. des Télécommunications*, 77(3-4):119–126, 2022.
27. Jacob C. N. Schuldt and Kanta Matsuura. On-line non-transferable signatures revisited. In *PKC 2011*, volume 6571 of *Lecture Notes in Computer Science*, pages 369–386. Springer, 2011.
28. Jacques Stern. A new identification scheme based on syndrome decoding. In *Advances in Cryptology - CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 13–21. Springer, 1993.
29. Dominique Unruh. Post-quantum security of fiat-shamir. In *Advances in Cryptology - ASIACRYPT 2017*, volume 10624 of *Lecture Notes in Computer Science*, pages 65–95. Springer, 2017.
30. Rupeng Yang, Man Ho Au, Zhenfei Zhang, Qiuliang Xu, Zuoxia Yu, and William Whyte. Efficient lattice-based zero-knowledge arguments with standard soundness: Construction and applications. In *Advances in Cryptology - CRYPTO 2019*, volume 11692 of *Lecture Notes in Computer Science*, pages 147–175. Springer, 2019.