Post-Quantum Single Secret Leader Election (SSLE) from Publicly Re-Randomizable Commitments

Dan Boneh
Stanford University, CA, USA

Aditi Partap
Stanford University, CA, USA

Lior Rotem
Stanford University, CA, USA

Abstract

A *Single Secret Leader Election (SSLE)* enables a group of parties to randomly choose exactly one leader from the group with the restriction that the identity of the leader will be known to the chosen leader and nobody else. At a later time, the elected leader should be able to publicly reveal her identity and prove that she is the elected leader. The election process itself should work properly even if many registered users are passive and do not send any messages. SSLE is used to strengthen the security of proof-of-stake consensus protocols by ensuring that the identity of the block proposer remains unknown until the proposer publishes a block. Boneh, Eskandarian, Hanzlik, and Greco (AFT'20) defined the concept of an SSLE and gave several constructions. Their most efficient construction is based on the difficulty of the Decision Diffie-Hellman problem in a cyclic group.

In this work we construct the first efficient SSLE protocols based on the standard Learning With Errors (LWE) problem on integer lattices, as well as the Ring-LWE problem. Both are believed to be post-quantum secure. Our constructions generalize the paradigm of Boneh et al. by introducing the concept of a re-randomizable commitment (RRC). We then construct several post-quantum RRC schemes from lattice assumptions and prove the security of the derived SSLE protocols. Constructing a lattice-based RRC scheme is non-trivial, and may be of independent interest.

2012 ACM Subject Classification Security and privacy → Cryptography

Keywords and phrases Consensus, Leader Election, Post-Quantum, Lattice Cryptography, Blockchain

Digital Object Identifier 10.4230/LIPIcs.AFT.2023.26

Related Version *Full Version*: https://eprint.iacr.org/2023/1241.pdf

Funding This work was funded by NSF, DARPA, the Simons Foundation, UBRI, and NTT Research. Opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA.

1 Introduction

Leader election is a core component of many consensus protocols used in practice. In proof-of-work systems such as [34], the identity of the leader remains hidden until the moment that the leader publishes a proposed block. In contrast, in many proof-of-stake systems, the identity of the leader is known in advance, long before the leader publishes a proposed block. This opens up the leader to certain attacks, including denial of service, that may prevent the chosen leader from publishing the newly created block. This in turn, can lead to a liveness failure for the chain.

In response, several works have studied *secret* leader election, where the identity of a randomly chosen leader remains secret until she publishes the new block and reveals herself as the leader [25, 31, 27, 8]. The added secrecy protects the leader from attacks that may

5th Conference on Advances in Financial Technologies (AFT 2023).
Editors: Joseph Bonneau and S. Matthew Weinberg; Article No. 26; pp. 26:1–26:23
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

prevent her from publishing the new block. However, existing proposals for secret leader election work by electing a few potential leaders *in expectation*, and describing a run-off procedure so that exactly one of the potential leaders is recognized as the final leader once all potential leaders have revealed themselves. The possibility of several potential leaders, however, can lead to wasted effort and may even cause a safety violation in case of an attack on the run-off procedure.

This issue motivates the need for a different type of leader election protocol, called a *Single Secret Leader Election*, or SSLE [16] (see also [21]). An SSLE protocol is comprised of two phases.

- In the first phase, parties may register to participate in leader elections. This step involves publishing some public information on a public bulletin board, while keeping some secret information associated with it private.
- In the second phase, elections are held using a protocol that is executed by the participating parties. The election protocol uses a randomness beacon and the public information on the bulletin to choose a leader among the parties. At a later time, the leader can declare themselves as such by providing a proof that they were selected as the leader.

Informally, an SSLE protocol needs to satisfy three security properties. **Uniqueness** asserts that at most a single party can prove they were elected as leader. **Fairness** requires that all participating parties have the same probability of being elected as leader, even if some parties are malicious. **Unpredictability** means that until the leader reveals itself, its identity should remain essentially hidden from the other parties, even if a subset of them colludes. It was recently shown that relying on SSLE leads to more efficient consensus protocols than relying on a secret leader election protocol that elects few leaders in expectation [7].

The concept of SSLE was formalized by Boneh, Eskandarian, Hanzlik, and Greco [16] who also presented a number of constructions. Their most efficient construction is based on the Decision Diffie-Hellman problem (DDH) in cyclic groups. We refer to this SSLE protocol as the **BEHG protocol**. The Ethereum Foundation optimized BEHG to obtain *Whisk* [30], which is the current proposal for SSLE to be used in Ethereum consensus. Since then, additional works have suggested alternative SSLE constructions with various security and efficiency tradeoffs (see, for example, [23, 40, 18, 9, 19, 24]).

Due to the potential long-term risk of a large scale quantum computer [41] there is a desire to also develop a *post-quantum* secure SSLE. One approach, already in [16], is an SSLE protocol based on fully homomorphic encryption (FHE). A further optimized FHE-based construction was recently proposed by Freitas et al. [24]. However, the complexity of these proposals is far greater than the simple DDH-based scheme. Another elegant approach to post-quantum SSLE was proposed by Sanso [40], who showed how to adapt Whisk to use an isogeny-based assumption, which is believed to be post-quantum secure. Finally, Drake [23] proposed an SSLE protocol that can be made post-quantum secure, but the proposal inherently relies on the availability of an anonymous broadcast channel (e.g., ToR).

Our results. In this paper we construct the first practical post-quantum SSLE protocols based on the Learning With Errors (LWE) problem [38] and Ring-LWE problem [33]. We do so by generalizing the BEHG protocol using a new concept we call a re-randomizable commitment (RRC). We show that an RRC together with a shuffle protocol gives an SSLE. We then construct a number of RRC schemes from lattices. The next section gives a detailed overview of the construction and explains the technical challenges in building an RRC from lattices.

1.1 Technical Overview

We briefly sketch the main ideas behind our construction. We begin with an abstract view of the BEHG protocol. Then, we present the notion of re-randomizable commitments (RRC) used by this protocol. Finally, we present our new lattice-based post-quantum RRCs for instantiating the abstract BEHG protocol.

The BEHG approach. The BEHG protocol employs a commit-and-shuffle approach. The following is a generalized and abstract view of the protocol.

- When party i registers for elections, it chooses some secret key k_i, computes a commitment c_i to k_i, and publishes c_i. We will define what is needed of this commitment in a minute. To avoid duplicity of secrets, each party also publishes a deterministic hash of k_i.
- At election time, participating parties run a protocol to shuffle and rerandomize the commitments. For simplicity of presentation in this overview, let us assume that the shuffle protocol works as follows: in each round, one of the parties locally permutes the entire list of commitments and then rerandomizes each of the commitments. It then publishes the new list of commitments, and proves in zero-knowledge that this new list is well-formed (i.e., it is obtained from the previous list by permuting and rerandomizing the commitments). Once the shuffle protocol is done, the parties obtain a list of commitments $\tilde{c}_1, \ldots, \tilde{c}_n$, where each $\tilde{c}_i$ is a rerandomization of $c_{\pi(i)}$ for some unknown permutation π on $\{1, \ldots, n\}$. They then let the randomness beacon choose an index $i^* \xleftarrow{\$} \{1, \ldots, n\}$, and party $j^* = \pi(i^*)$ is the chosen leader. In due time, party j^* can prove that it was elected by publishing k_{j^*} and the other parties can check this value against the commitment $\tilde{c}_{i^*}$.

Re-randomizable commitments. We identify several properties that the commitment scheme being used must satisfy for the resulting SSLE protocol to be correct and secure. First, the commitments have to be **re-randomizable** in a very specific sense. Given a commitment c to some value k, one should be able to re-randomize c without knowledge of k or the randomness used to generate c. Moreover, given a value k and a (potentially re-randomized) commitment $\tilde{c}$, one should be able to efficiently test whether $\tilde{c}$ is a commitment to k. In particular, this test should not require the randomness used for re-randomization. In the BEHG protocol, this means that the original committer to $\tilde{c}_{i^*}$ can: (i) recognize itself as the winner of the elections (by checking if $\tilde{c}_{i^*}$ is a commitment to k_{j^*}); and (ii) prove that it won by publishing k_{j^*}.

The commitment scheme should also satisfy the standard notion of **binding**. This means that it should be infeasible to produce a commitment c alongside two *distinct* values k and k', such that c passes both as a commitment to k and as a commitment to k'. In the context of the BEHG protocol, this means that there is only a single party that can prove ownership of the chosen commitment $\tilde{c}_{i^*}$ by publishing k_{j^*}.

Finally, commitments should also be **unlinkable**. This means that given two commitments c_0 and c_1 to two random values, and a re-randomization $\tilde{c}$ for one of them, it should be infeasible to determine if $\tilde{c}$ is a re-randomization of c_0 or of c_1. This is essential for the BEHG protocol to achieve unpredictability: an adversary should not be able to link the chosen commitment $\tilde{c}_{i^*}$ to the original commitment c_{j^*} and therefore identify party j^* as the leader. Looking ahead, the use of re-randomizable commitments in the generalized BEHG SSLE protocol actually requires a stronger notion of unlinkability. We postpone the discussion on this matter and will revisit it shortly.

The DDH-based construction of re-randomizable commitments (RRCs) suggested by BEHG is as follows. Let $\mathbb{G}$ be a cyclic group of order p generated by $g \in \mathbb{G}$. A commitment c to a random value $k \xleftarrow{\$} \mathbb{Z}_p$ is a pair (g^r, g^{rk}) where $r \xleftarrow{\$} \mathbb{Z}_p$. To check if a commitment

$c = (c_1, c_2)$ is a consistent with a value k, once can simply check if $c_2 = c_1^k$. To re-randomize, one chooses a random $r' \xleftarrow{\$} \mathbb{Z}_p$ and outputs $\tilde{c} = (c_1^{r'}, c_2^{r'})$. The scheme is perfectly binding, and unlinkability easily follows from the DDH assumption.

It should be noted that previous works also considered other variants of re-randomizable commitments (see, for example, [5, 20]). However, in these works, opening a re-randomized commitment requires knowledge of the randomness used for re-randomization (or a function thereof). Such commitments are much simpler to construct, and indeed, many long-standing algebraic and lattice-based constructions can be easily re-randomized according to this weaker definition. Unfortunately, as discussed above, such commitments are insufficient for instantiating the BEHG protocol.

RRCs from LWE: A first attempt. Consider the following (flawed) RRC scheme. The secret key space is $\mathbb{Z}_q^n$, where q is a prime and $n \approx \lambda$ is the LWE hardness parameter. To commit to a random $\boldsymbol{k} \in \mathbb{Z}_q^n$ the *Commit* algorithm samples a uniformly random $\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$ and outputs $(\boldsymbol{A}, \boldsymbol{u}) = (\boldsymbol{A}, \boldsymbol{A} \cdot \boldsymbol{k} + \boldsymbol{e})$, where $\boldsymbol{e}$ is an LWE noise vector and $m > n$. To test whether a key $\boldsymbol{k}$ is tied to a commitment $c = (\boldsymbol{A}, \boldsymbol{u})$, we can check whether $\boldsymbol{A} \cdot \boldsymbol{k}$ is close (say, in Euclidean distance) to $\boldsymbol{u}$. We accept $\boldsymbol{k}$ if this is the case and reject otherwise. If $\boldsymbol{A}$ is chosen randomly and $m \approx n \log n$ ($\boldsymbol{A}$ is a "tall" matrix), a standard argument shows that with high probability over the choice of $\boldsymbol{A}$, there are no $\boldsymbol{k}, \boldsymbol{k}'$ and $\boldsymbol{u}$ such that $\boldsymbol{A} \cdot \boldsymbol{k} \approx \boldsymbol{u}$ and $\boldsymbol{A} \cdot \boldsymbol{k}' \approx \boldsymbol{u}$.

To re-randomize, the rerandomization algorithm samples a low-norm m-by-m matrix $\boldsymbol{R}$ and computes $c' = (\boldsymbol{A}', \boldsymbol{u}') = (\boldsymbol{R} \cdot \boldsymbol{A}, \boldsymbol{R} \cdot \boldsymbol{u})$. Since $\boldsymbol{R}$ is of low norm $\boldsymbol{Re}$ may only be slightly longer than $\boldsymbol{e}$. Hence, $\boldsymbol{Re}$ is also short and we have

$$\boldsymbol{A}' \cdot \boldsymbol{k} = \boldsymbol{R} \cdot \boldsymbol{A} \cdot \boldsymbol{k} \approx \boldsymbol{R} \cdot \boldsymbol{A} \cdot \boldsymbol{k} + \boldsymbol{R} \cdot \boldsymbol{e} \approx \boldsymbol{R} \cdot \boldsymbol{u} = \boldsymbol{u}'.$$

The noise does grow a bit with each re-randomization, which is why the scheme only supports a bounded number of re-randomizations (the LWE parameters can be chosen according to the number of re-randomizations required by the SSLE shuffle protocol). In terms of unlinkability, note that assuming LWE is hard, a fresh commitment $c = (\boldsymbol{A}, \boldsymbol{u})$ is just a pseudorandom matrix-vector pair. Moreover, if m is sufficiently greater than n and each row of $\boldsymbol{R}$ has high min-entropy, the leftover hash lemma [26, 29] shows that c' is also pseudorandom, which implies that the scheme is unlinkable.

The problem. Unfortunately, the above analysis is flawed. It is true that the scheme is binding when the matrix $\boldsymbol{A}$ is chosen uniformly at random from $\mathbb{Z}_q^{m \times n}$. But since $\boldsymbol{A}$ is part of the commitment c, the adversary may choose it from some other skewed distribution, thus breaking the binding argument. This is not just an issue of reworking the proof. The scheme is in fact insecure: fix any $\boldsymbol{k}$ and $\boldsymbol{k}'$ and it is easy to come up with a matrix $\boldsymbol{A}$ for which $\boldsymbol{A} \cdot \boldsymbol{k} \approx \boldsymbol{A} \cdot \boldsymbol{k}'$. To fix this issue, one might be tempted to choose the matrix $\boldsymbol{A}$ as part of the public parameters, or to force committers to choose $\boldsymbol{A}$ as the output of a hash function modeled as a random oracle. Indeed, this would make the scheme binding, but then it becomes unclear how to re-randomize the commitments.

The key observation. Let us revisit the naive "proof" of binding for the above construction. If $\boldsymbol{A}$ is indeed chosen uniformly at random, then with overwhelming probability there are no $\boldsymbol{k}$ and $\boldsymbol{k}'$ such that $\boldsymbol{A} \cdot \boldsymbol{k} \approx \boldsymbol{A} \cdot \boldsymbol{k}'$. In particular, this would suggest that for random $\boldsymbol{A}$, $\boldsymbol{k}$ and $\boldsymbol{k}'$ it holds that $\boldsymbol{A} \cdot \boldsymbol{k}$ and $\boldsymbol{A} \cdot \boldsymbol{k}'$ are almost surely far apart. Put differently, for a uniform $\boldsymbol{k}$ and $\boldsymbol{k}'$, there are *very few* matrices $\boldsymbol{A}$ for which $\boldsymbol{Ak} \approx \boldsymbol{Ak}'$. So what if instead of choosing a single $\boldsymbol{k}$, we make the *Commit* algorithm sample the commitment key k as a pair $(\boldsymbol{k}_1, \boldsymbol{k}_2)$ of independent and uniformly-random vectors? One could expect that for two such random pairs $(\boldsymbol{k}_1, \boldsymbol{k}_2)$ and $(\boldsymbol{k}'_1, \boldsymbol{k}'_2)$, the set of matrices $\boldsymbol{A}$ for which $\boldsymbol{A} \cdot \boldsymbol{k}_1 \approx \boldsymbol{A} \cdot \boldsymbol{k}'_1$ *and* $\boldsymbol{A} \cdot \boldsymbol{k}_2 \approx \boldsymbol{A} \cdot \boldsymbol{k}'_2$ is *even smaller*. Indeed, we show that for $\ell \approx n$, if one samples two ℓ-tuples $(\boldsymbol{k}_1, \ldots, \boldsymbol{k}_\ell)$ and $(\boldsymbol{k}'_1, \ldots, \boldsymbol{k}'_\ell)$ of vectors uniformly at random, then with very high probability a matrix $\boldsymbol{A}$ for which $\boldsymbol{A} \cdot \boldsymbol{k}_i \approx \boldsymbol{A} \cdot \boldsymbol{k}'_i$ for every i *simply does not exist.*

Alas, the proposed commitment scheme is binding for keys that are *random* tuples of vectors, but the binding security game allows the adversary to choose the "colliding" keys $(\boldsymbol{k}_1, \ldots, \boldsymbol{k}_\ell)$ and $(\boldsymbol{k}'_1, \ldots, \boldsymbol{k}'_\ell)$ as it pleases – they need not be uniformly random. On the face of it, it might seem that we are back to square one. Fortunately, this is not the case. The final observation is that for this construction, we *can* make the commitment algorithm choose the vectors $\boldsymbol{k}_1, \ldots, \boldsymbol{k}_\ell$ as the output of a cryptographic hash function H, without hampering re-randomization. That is, to commit, one samples a matrix $\boldsymbol{A}$ and a key $k \xleftarrow{\$} \{0,1\}^\lambda$, computes $\boldsymbol{k}_1, \ldots, \boldsymbol{k}_\ell \leftarrow \mathsf{H}(k)$ and outputs the commitment $c \leftarrow (\boldsymbol{A}, \{\boldsymbol{A} \cdot \boldsymbol{k}_i + \boldsymbol{e}_i\}_i)$ where all the $\boldsymbol{e}_i$s are independent LWE noise vectors. To test a key k against a commitment $c = (\boldsymbol{A}, \{\boldsymbol{u}_i\}_i)$, the *Test* algorithm simply recomputes $\boldsymbol{k}_1, \ldots, \boldsymbol{k}_\ell$ from k and checks that $\boldsymbol{A} \cdot \boldsymbol{k}_i \approx \boldsymbol{u}_i$ for every $i = 1, \ldots, \ell$.

Adversarial re-randomizations. The construction that we just saw indeed satisfies the notion of unlinkability sketched above. Unfortunately, as we already mentioned, this notion is insufficient for the resulting SSLE protocol to achieve unpredictability. This reason is this: unlinkability only guarantees that if honestly-generated commitments $c_1, \ldots, c_n$ are honestly re-randomized and shuffled, an adversary cannot trace the re-randomized commitments to the original ones. In the SSLE protocol above, an honest re-randomization might follow an adversarial one. So we need to require unlinkability of commitments even after adversarial re-randomizations. We call this *strong unlinkability.*

In the DDH-based construction of BEHG strong unlinkability comes "for free". Unfortunately, this is not the case with our LWE-based RRC scheme. For example, consider an adversary that given a commitment $c = (\boldsymbol{A}, \{\boldsymbol{A} \cdot \boldsymbol{k}_i + \boldsymbol{e}_i\}_i)$, finds a matrix $\boldsymbol{R}$ such that $\boldsymbol{R} \cdot \boldsymbol{A}$ has short columns. The adversary then uses this $\boldsymbol{R}$ to re-randomize c into $\tilde{c} \leftarrow (\boldsymbol{R} \cdot \boldsymbol{A}, \{\boldsymbol{R} \cdot \boldsymbol{A} \cdot \boldsymbol{k}_i + \boldsymbol{e}_i\}_i)$. Now, even if we honestly re-randomize $\tilde{c}$, we will almost surely end up with a commitment $\hat{c}$ whose first coordinate is still a short-columns matrix. Hence, the adversary can easily trace $\hat{c}$ back to c.

We present several methods to thwart such attacks. In this overview, we focus on what we view as the simplest and most practical one. Ahead of time, all parties commit to the matrices $\boldsymbol{R}_1, \boldsymbol{R}_2, \ldots$ they are going to use for re-randomization using a standard additively homomorphic commitment scheme. When a party now has to carry out its ith re-randomization, it does so using the matrix $\boldsymbol{R}_i + \boldsymbol{R}'_i$, where $\boldsymbol{R}'_i$ is a low norm matrix outputted by a public randomness beacon. Such a beacon can be external or implemented in various standard ways. Using the homomorphic properties of the commitment scheme, everyone can now compute a commitment to $\boldsymbol{R}_i + \boldsymbol{R}'_i$. The re-randomizer can hence prove that this is the matrix it used. Informally, since $\boldsymbol{R}_i$ was committed to ahead of time, it is independent of $\boldsymbol{R}'_i$. Hence, the re-randomizer is forced to use a high-entropy matrix for re-randomization, which guarantees the resulting commitment is from the appropriate distribution. Since $\boldsymbol{R}_i$ is always hidden, $\boldsymbol{R}_i + \boldsymbol{R}'_i$ has high min-entropy even given $\boldsymbol{R}'_i$, and we can still rely on the leftover hash lemma to argue that subsequent honest re-randomizations provide unlinkability.

Extending the scheme to Ring LWE. We extend our LWE-based RRC scheme to the ring setting, relying on the Ring Learning with Errors (Ring-LWE) assumption. As we discuss in Section 5 in detail, moving to the ring setting offers several gains in efficiency. Specifically, we work in a polynomial ring $\mathcal{R}$ modulo a cyclotomic polynomial f, which factors into a constant number of irreducible polynomials over $\mathbb{Z}_q$. Concretely, we choose $q = 3 \bmod 8$ so that f has exactly two irreducible factors f_1, f_2 over $\mathbb{Z}_q$ (but other choices are possible).

The construction follows the same template as our LWE-based construction, but the matrix $\boldsymbol{A}$ is now replaced with a vector of ring elements. To commit, one samples $\boldsymbol{a} \xleftarrow{\$} \mathcal{R}_q^m$, and a key $k \xleftarrow{\$} \{0,1\}^\lambda$, computes ℓ ring elements as $k_1, \ldots, k_\ell \leftarrow \mathsf{H}(k)$ and the commitment is given by $c \leftarrow (\boldsymbol{a}, \{\boldsymbol{a} \cdot k_i + \boldsymbol{e}_i\}_i)$ where all $\boldsymbol{e}_i$s are independent RLWE noise vectors in $\mathcal{R}_q^m$. Re-randomization is done by sampling a low-norm matrix $\boldsymbol{R} \xleftarrow{\$} \mathcal{R}^{m \times m}$, and computing $c' = (\boldsymbol{R} \cdot \boldsymbol{a}, \{\boldsymbol{R} \cdot \boldsymbol{u}_i\}_i)$. To test a commitment $c = (\boldsymbol{a}, \boldsymbol{u})$ against a key k, one computes $k_1, \ldots, k_\ell \leftarrow \mathsf{H}(k)$ and check that $\boldsymbol{a} \cdot k_i \approx \boldsymbol{u}_i$ for all i. Correctness and unlinkability are proven similarly to the integer case, with one exception: instead of relying on the leftover hash lemma, we rely on the regularity lemma of [42].

Two main observations make our ring-based scheme more efficient than our integer-based one:

- We can choose ℓ to be smaller than in the integer case, and still make the binding argument go through. Intuitively, the reason is that each entry of $\boldsymbol{a} \cdot k_i$ is now a polynomial in the ring $\mathcal{R}$ and not an integer. Thus, we may hope that it has more than $\log q$ bits of min-entropy (roughly the entropy of a random integer in $\mathbb{Z}_q$). If this is indeed the case, then the probability that $\boldsymbol{a} \cdot k \approx \boldsymbol{a} \cdot k'$, over the choice of random $\boldsymbol{a}, k, k'$, is much smaller than the probability that $\boldsymbol{a}^T \cdot \boldsymbol{k} \approx \boldsymbol{a}^T \cdot \boldsymbol{k}'$ in the integer case for random $\boldsymbol{a}, \boldsymbol{k}, \boldsymbol{k}' \xleftarrow{\$} \mathbb{Z}_q^n$. This would imply that we can choose ℓ to be smaller, resulting in smaller commitments. To argue that $\boldsymbol{a} \cdot k_i$ indeed has high min-entropy, we rely on the particular structure of the ring $\mathcal{R}$. If $k \neq k'$, it means that the polynomials must be distinct modulo f_1 or modulo f_2. Assume with loss of generality that they are distinct modulo f_1. Since f_1 is irreducible mod q, $a \cdot (k - k')$ is uniformly random in $\mathbb{Z}_q[x]/f_1$, and hence it has at least $\approx \mathsf{deg}(f_1) \cdot \log q$ bits of min entropy. This analysis is inspired by the statistically-binding commitments of Benhamouda, Krenn, Lyubashevsky, and Pietrzak [14].
- The second observation is that our use of the leftover hash lemma in the LWE setting incurred an overhead that can be avoided in the Ring LWE setting. To explain this point, we need to revisit our LWE unlinkability argument in more detail. Recall that we wanted to argue that if we have a commitment $c = (\boldsymbol{A}, \boldsymbol{U})$ and we re-randomize it to $c' = (\boldsymbol{R} \cdot \boldsymbol{A}, \boldsymbol{R} \cdot \boldsymbol{U})$, then the commitment c' we end up with is pseudorandom. The first step was to argue that c is pseudorandom, thanks to the LWE assumption. This step remains essentially unchanged here, relying on the Ring-LWE assumption instead. The second step was to rely on the leftover has lemma; this step required each row of $\boldsymbol{R}$ to have more than $\Omega((n + \ell) \cdot \log q)$ bits of min-entropy. This implied that m had to be set to be at least $(n + \ell) \cdot \log q$. In the ring setting, however, since each coordinate of $\boldsymbol{R}$ can have $\Omega(n)$ bits of min-entropy, m can be reduced to roughly $\log q$. This results in much "shorter" matrices $\boldsymbol{A}, \boldsymbol{U}$ making up the commitment.

Reducing communication. Catalano, Fiore, and Giuta [19] observed that when instantiating the BEHG protocol with a DDH-based RRC of the form $c = (g^r, g^{rk})$, the commitments of all parties can share the same first coordinate $h = g^r$, which is part of the public parameters. Then, to re-randomize N commitments $(h^r, g^{rk_1}, \ldots, g^{rk_N})$, a shuffler can sample a single $r' \xleftarrow{\$} \mathbb{Z}_q$ and raise all the elements to the r'. This optimization cuts storage and communication by about half. It is tempting to implement this optimization using our lattice-based commitments; have all commitments share the first coordinate $\boldsymbol{A}$ (or $\boldsymbol{a}$ in the ring setting) and use a single re-randomization matrix $\boldsymbol{R}$ to re-randomize all commitments. The problem is that to retain unlinkability, the dimensions of $\boldsymbol{R}$ need to grow as a function of the number of commitments N, which may eliminate the gains of sharing $\boldsymbol{A}$ across all commitments. We discuss this further in the full version where we consider settings where this can still lead to some savings.

Post-quantum proof of shuffle. Recall that in the BEHG protocol, after each shuffle, the shuffling party has to prove that it indeed performed a valid shuffle; that is, it applied the *Randomize* algorithm of the RRC scheme to each commitment and then permuted the resulting commitments. This can be done by using any general-purpose non-interactive argument of knowledge, proving that the shuffler knows random coins for *Randomize* and a permutation that together yield the resulting list of re-randomized commitments (for such argument systems based on post-quantum secure assumptions, see for example [13, 11, 12, 4, 15, 28, 10, 6, 32, 2, 35] and the references therein).

When using our RLWE-based RRC commitments, we also show how we can change the recent lattice-based proof-of-shuffle protocol of [22] to work with our commitments. This is a simple protocol that may provide better concrete efficiency.

1.2 Paper Organization

The remainder of the paper is organized as follows. In Section 2 we present basic notation and computational assumptions used in the paper. In section 3, we define RRC schemes, and in Sections 4 and 5, we present our constructions from LWE and Ring-LWE, respectively. In section 6 we strengthen our security notion and constructions for RRC schemes. In the full version of this paper, we present the generalized BEHG protocol, discuss and construct proofs of shuffle for our RRC schemes, and present additional ways to obtain our stronger security notion. The full version also contains proofs that are omitted from this version.

2 Preliminaries

In this section, we present the basic notions and cryptographic primitives that are used in this work. For an integer $n \in \mathbb{N}$ we denote by $[n]$ the set $\{1, \ldots, n\}$. For a distribution X we denote by $x \leftarrow\!\!\$\ X$ the process of sampling a value x from the distribution X. Similarly, for a set $\mathcal{X}$, we denote by $x \leftarrow\!\!\$\ \mathcal{X}$ the process of sampling a value x from the uniform distribution over $\mathcal{X}$. For a pair X, Y of distributions defined over the same domain Ω, we denote by $\mathsf{SD}(X, Y)$ the *statistical distance* between them, defined as $\mathsf{SD}(X, Y) = \frac{1}{2} \sum_{\omega \in \Omega} |\Pr[X = \omega] - \Pr[Y = \omega]|$.

We denote matrices by boldface capital letters, e.g. $\boldsymbol{A}$, and vectors in boldface lower-case letters, e.g. $\boldsymbol{v}$. We may use a non-bold capital letter, e.g. A or V, to describe a matrix or a vector, when we wish to emphasize that this matrix or vector is being treated as a random variable. As standard, we identify $\mathbb{Z}_q$ for a prime q with the set $(-q/2, \ldots, q/2]$, and we define the absolute value of an element $x \in \mathbb{Z}_q$ as $|x| = \{\min |y| : y \in \mathbb{Z}, y = x \pmod{q}\}$.

For $n, p \in \mathbb{N}$ where p is prime, we define the rings $\mathcal{R} = \mathbb{Z}[x]/f(x)$ and $\mathcal{R}_p = \mathcal{R}/\langle p \rangle$, where $f(x)$ is monic and of degree n. That is, $\mathcal{R}_p$ is the ring of polynomials modulo $f(x)$ with integer coefficients in $\mathbb{Z}_p$. We define the norm of elements in these rings to be the norm of their coefficient vector in $\mathbb{Z}^n$, which is also called the coefficient embedding. For any $g(x) = \sum_{i \in 0 \cup [n-1]} \alpha_i x^i \in \mathcal{R}$, we use $\mathsf{coeff}(g)$ to denote the vector $\{\alpha_0, \ldots, \alpha_{n-1}\}$, i.e. the coefficient embedding of $g(x)$, and the norm is defined as follows:

$$||g||_1 = \sum \alpha_i \quad ||g||_2 = (\sum \alpha_i^2)^{1/2} \quad ||g||_\infty = \max|\alpha_i|$$

For a vector $\boldsymbol{v}$ over $\mathcal{R}$, we define $||\boldsymbol{v}|| = (\sum_i ||\boldsymbol{v}_i||^2)^{1/2}$.

2.1 Lattice Assumption

The paper makes use of two basic and standard lattice-based assumptions, the learning with errors (LWE) assumption and the short integer solution (SIS) assumption, both of which over integer lattices. We briefly recall these assumptions here. For a more detailed survey of these assumptions and their hardness, see, for example, [36] and the many references therein.

The LWE assumption. We rely on the following formulation of the learning with errors (LWE) problem, introduced by Regev [39]. The problem is parameterized by a prime modulus q, a vector length n which typically corresponds to the security parameter λ, and a noise distribution χ. For our needs, the important thing is that χ is highly concentrated on low-norm vectors such that with overwhelming probability $\|\boldsymbol{x}\|_2 \leq \delta$ for $\boldsymbol{x} \xleftarrow{\$} \chi$ for some $\delta = \delta(\lambda)$ (one typically takes χ to be a discrete Gaussian with appropriate parameters)

▶ **Definition 1.** *Let $q = q(\lambda)$ be a prime, $n = n(\lambda)$ be an integer, and $\chi = \chi(\lambda)$ be a distribution over $\mathbb{Z}_q$, all public functions of the security parameter $\lambda \in \mathbb{N}$. The (q, n, χ)-LWE assumption states that for every probabilistic polynomial time algorithm $\mathcal{A}$ and for all polynomially-bounded functions $m = m(\lambda)$ there exists a negligible function $\nu(\cdot)$ such that*

$$\mathsf{Adv}^{\mathrm{lwe}}_{\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(\boldsymbol{A}, \boldsymbol{A} \cdot \boldsymbol{s} + \boldsymbol{e}) = 1] - \Pr[\mathcal{A}(\boldsymbol{A}, \boldsymbol{v}) = 1]| \leq \nu(\lambda)$$

for all sufficiently large $\lambda \in \mathbb{N}$, where $\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}, \boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n, e \xleftarrow{\$} \chi^m$, and $\boldsymbol{v} \xleftarrow{\$} \mathbb{Z}_q^m$.

2.2 Ring Lattice Assumption

We will also use the ring-based variant of the LWE assumption, introduced by [33].

The RLWE assumption. This problem is also parameterized by the prime modulus q, degree of the modulus polynomial n, and a noise distribution χ. We focus on a special case of the Ring-LWE problem where $f(x) = x^n + 1$, and n is a power of two. Similar to LWE, χ is highly concentrated on low-norm polynomials such that with overwhelming probability $||x||_2 \leq \delta$ for $x \leftarrow\$ \chi$ for some $\delta = \delta(\lambda)$. χ is usually taken to be a discrete gaussian in the coefficient embedding of $\mathcal{R}$.

▶ **Definition 2.** *Let $q = q(\lambda)$ be a prime, $n = n(\lambda)$ be an integer, and $\chi = \chi(\lambda)$ be a distribution over $\mathcal{R}$, all public functions of the security parameter $\lambda \in \mathbb{N}$. The (q, n, χ)-RLWE assumption states that for every probabilistic polynomial time algorithm $\mathcal{A}$ and for all polynomially-bounded functions $m = m(\lambda)$, there exists a negligible function $\nu(\cdot)$ such that*

$$\mathsf{Adv}^{\mathrm{rlwe}}_{\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(\boldsymbol{a}, \boldsymbol{b}) = 1] - \Pr[\mathcal{A}(\boldsymbol{a}, \boldsymbol{v}) = 1]| \leq \nu(\lambda)$$

for all sufficiently large $\lambda \in \mathbb{N}$, where $\boldsymbol{a} \xleftarrow{\$} \mathcal{R}_q^m, s \xleftarrow{\$} \mathcal{R}_q, \boldsymbol{e} \xleftarrow{\$} \chi^m, \boldsymbol{b}_i = \boldsymbol{a}_i \cdot s + \boldsymbol{e}_i \, \forall\, i \in [m]$, and $\boldsymbol{v} \xleftarrow{\$} \mathcal{R}_q^m$.

2.3 Randomness Extraction

We will use the following lemma from the work of Gentry, Peikert, and Vaikuntanathan [26]. The lemma follows from the leftover hash lemma [29].

▶ **Lemma 3** ([26, 29]). *Let q be a prime and let m, n be integers. Let R, A and B be random variables distributed uniformly in $\{-1,1\}^{m\times m}$, $\mathbb{Z}_q^{m\times n}$, and $\mathbb{Z}_q^{m\times n}$, respectively. Then, it holds that*

$$\mathsf{SD}\left((A, R\cdot A), (A, B)\right) \leq \frac{m}{2}\cdot\sqrt{2^{-m+n\log q}}.$$

When working over polynomial rings, we will not be able to use the leftover hash lemma. Instead, we will use the regularity lemma defined over rings [42].

▶ **Lemma 4** (Generalization of Theorem 3.2, [42]). *Let $\mathbb{F}$ be a finite field and $f \in \mathbb{F}[x]$ be monic and of degree $n > 0$. Let $\mathcal{R}$ be the ring $\mathbb{F}[x]/f$ and $m > 0$. For every $i, j \in [m]$ and $k \in [n]$, let $D_{i,j,k} \subseteq \mathbb{F}$, with $|D_{i,j,k}| = d$. Let A, B be random variables distributed uniformly in $\mathcal{R}^{m\times\ell}$. Let $R \in \mathcal{R}^{m\times m}$ be a matrix of polynomials, wherein the kth coefficient of $R_{i,j}$ is chosen uniformly randomly and independently from $D_{i,j,k}$, for all $i, j \in [m]$ and $k \in [n]$. Then, it holds that,*

$$\mathsf{SD}\left((A, RA), (A, B)\right) \leq \frac{m}{2}\sqrt{\prod_{i\in[t]}\left(1+\left(\frac{|\mathbb{F}|}{d^m}\right)^{\mathsf{deg}(f_i)}\right)^{\ell} - 1}$$

where $f = \prod_{i\in[t]} f_i$ is the factorization of f over $\mathbb{F}[x]$, and $\mathsf{deg}(f_i)$ is the degree of the polynomial f_i.

Specifically, we will choose $\mathbb{F} = \mathbb{Z}_q$ and $D_{i,j,k} = \{-1,1\}\,\forall\, i, j \in [m], k \in [n]$.

We will also rely on the following definition for the norm of a matrix and a related lemma from Agrawal, Boneh, and Boyen [1] (a similar lemma appears in [3]), which states that a random Bernoulli matrix has low norm with overwhelming probability.

▶ **Definition 5.** *Let $\boldsymbol{R}$ be an $m\times m$ matrix over $\mathbb{Z}$. Let $\boldsymbol{B}_m := \{\boldsymbol{x} \in \mathbb{R}^m : \|\boldsymbol{x}\|_2 = 1\}$ be the unit ball in $\mathbb{R}^m$. Define the norm of the matrix $\boldsymbol{R} \in \mathbb{Z}^{m\times m}$ as*

$$\|\boldsymbol{R}\| := \max_{\boldsymbol{x}\in\boldsymbol{B}_m} \|\boldsymbol{R}\cdot x\|_2 .$$

The norm for a matrix in $\mathcal{R}_q^{m\times m}$ is defined similarly. The following two lemmas bound the norm of random matrices where all entries are sampled i.i.d. from a distribution concentrated around 0.

▶ **Lemma 6** ([1, 3]). *Let q be a prime and let m be an integer. Let R be a random variable uniformly sampled from $\{-1,1\}^{m\times m}$. Then, there is a universal constant $C > 0$ such that*

$$\Pr\left[\|R\| \geq C\cdot\sqrt{m}\right] < e^{-2m}.$$

A proof for the following lemma can be found in the full version.

▶ **Lemma 7.** *Let q be a prime and let m, n be integers. Let $R \in \mathcal{R}_q^{m\times m}$ be a random variable, such that for all $i, j \in [m]$, the coefficient vector of $R_{i,j}$ is sampled uniformly at random from $\{-1,1\}^n$. Then,*

$$\Pr\left[\|R\| \geq m\sqrt{mn}\cdot\omega(\sqrt{logn})\right] < negl(n)$$

3 Re-randomizable Commitments

Informally, a re-randomizable commitment (RRC, for short) is a scheme that allows one to commit to random keys.[1] Moreover, an RRC scheme supports re-randomizations of commitments: given a commitment c to a key k, one should be able to re-randomize to commitment to produce a new commitment c' for k. Importantly, knowledge of c suffices for such re-randomization, and no additional secrets are needed. In particular, the re-randomizing entity is not required to know the key k nor the randomness used to create c.

We first present the syntax for RRC schemes and the associated correctness requirement. Then, we discuss two security notions that such schemes should satisfy.

3.1 Syntax & Correctness

An RRC scheme R is a tuple of four algorithms:

- $\mathit{Setup}(1^\lambda) \to pp$: outputs public parameters pp,
- $\mathit{Commit}(pp) \to (c, k)$: outputs a commitment string c and a key k,
- $\mathit{Randomize}(pp, c) \to c'$: randomize the commitment,
- $\mathit{Test}(pp, c, k) \to \{0, 1\}$: outputs 1 if k is a valid key for c.

The first three are probabilistic polynomial time (PPT) and the fourth is deterministic polynomial time.

In terms of correctness, we require that $\mathit{Test}(pp, c, k)$ outputs 1 for (c, k) output by $\mathit{Commit}(pp)$. Moreover, $\mathit{Test}(pp, c', k)$ should output 1 if c' was obtained from c via at most B consecutive re-randomizations, where B is a parameter. We call this correctness requirement *B-randomizability*. If a scheme is B-randomizable for all B, we call it fully-randomizable.

▶ **Definition 8.** *An RRC scheme is* **B-randomizable** *if there exists a negligible function $\nu(\cdot)$ such that the following holds for every $\lambda \in \mathbb{N}$:*
let $pp \xleftarrow{\$} \mathit{Setup}(1^\lambda)$, $(c_0, k) \xleftarrow{\$} \mathit{Commit}(pp)$, and $c_i \xleftarrow{\$} \mathit{Randomize}(pp, c_{i-1})$ for $i = 1, \ldots,$ then

$$\Pr\Big[\mathit{Test}(pp, c_i, k) = 1 \quad \text{for } i = 0, 1, 2, \ldots, B\Big] \geq 1 - \nu(\lambda).$$

An RRC scheme that is B-randomizable for all $B \in \mathbb{N}$ is said to be **fully randomizable***.* ⌟

For the notion of RRC schemes to be non-trivial, we require that the key k generated by *Commit* to have high min-entropy.

▶ **Definition 9.** *An RRC scheme is* **B-randomizable** *is* **non-trivial** *if there exists a negligible function $\nu(\cdot)$ such that the following holds for every $\lambda \in \mathbb{N}$: let $pp \xleftarrow{\$} \mathit{Setup}(1^\lambda)$, $(c_0, k_0) \xleftarrow{\$} \mathit{Commit}(pp)$ and $(c_1, k_1) \xleftarrow{\$} \mathit{Commit}(pp)$, then* $\Pr[k_0 = k_1] \leq \nu(\lambda)$. ⌟

3.2 Notions of Security

An RRC scheme should satisfy two security properties: Binding and Unlinkability.

[1] Committing to random keys is sufficient for the main application we consider, which is SSLE protocols. Observe, however, that such a scheme can be easily converted into a scheme that allows one to commit to arbitrary messages via a one-time pad.

```
Game $\mathbf{G}_{\mathcal{A},\mathsf{R}}(\lambda, B)$
$b \xleftarrow{\$} \{0,1\}$
$pp \xleftarrow{\$} \mathsf{R}.\mathit{Setup}(1^\lambda)$
$(c_0, k_0) \xleftarrow{\$} \mathsf{R}.\mathit{Commit}(pp), \quad (c_1, k_1) \xleftarrow{\$} \mathsf{R}.\mathit{Commit}(pp)$
$(\mathsf{state}, i_0, i_1) \xleftarrow{\$} \mathcal{A}(pp, c_0, c_1)$
$\textbf{if } (i_0 > B) \text{ OR } (i_1 > B) : \textbf{ abort}$
$\textbf{if } (i_0 = 0) \text{ OR } (i_1 = 0) : \textbf{ abort}$
$c \leftarrow c_b$
$\textbf{for } t \textbf{ in } \{1, \ldots, i_b\} : \quad c \xleftarrow{\$} \mathsf{R}.\mathit{Randomize}(pp, c)$
$b' \xleftarrow{\$} \mathcal{A}(c, \mathsf{state})$
$\textbf{return } b = b'$
```

Figure 1 The security game for an adversary $\mathcal{A}$ attacking the unlinkability of an RRC scheme R.

Binding. Similarly to standard commitment schemes, we require that a commitment can be tied to at most one key.

▶ **Definition 10.** *An RRC scheme is* **perfectly binding** *if for every $\lambda \in \mathbb{N}$ and for all c, k, k' we have*

$$\Pr_{pp \xleftarrow{\$} \mathit{Setup}(1^\lambda)}[k \neq k' \quad \text{AND} \quad \mathit{Test}(pp, c, k) = \mathit{Test}(pp, c, k') = 1] = 0. \tag{1}$$

⌟

Condition (1) ensures that a commitment c will *never* be accepted by two distinct keys. As we will later discuss, this is satisfied by the previous DDH-based construction of Boneh et al. [16]. For our lattice-based construction, we need to weaken this condition a bit and only require that (1) holds computationally. This leads to the following definition.

▶ **Definition 11.** *We say that an RRC scheme is* **computationally binding** *if for all* PPT*adversaries $\mathcal{A}$ the following function is negligible.*

$$\Pr\left[k \neq k' \quad \text{AND} \quad \mathit{Test}(pp, c, k) = \mathit{Test}(pp, c, k') = 1 \; : \; \begin{array}{l} pp \xleftarrow{\$} \mathit{Setup}(1^\lambda) \\ (c, k, k') \xleftarrow{\$} \mathcal{A}(pp) \end{array}\right] \tag{2}$$

⌟

Unlinkability. An RRC scheme R is unlinkable if a PPT adversary is unable to distinguish the i-th re-randomization of a commitment c_0 from the j-th re-randomization of another commitment c_1. This is captured in the security game in Figure 1. As usual, we define the adversary's advantage as

$$\mathsf{Adv}^{\mathrm{rrc}}_{\mathcal{A},\mathsf{R},B}(\lambda) := \left|2\Pr[\mathbf{G}_{\mathcal{A},\mathsf{R}}(\lambda, B) = 1] - 1\right|.$$

▶ **Definition 12.** *A B-randomizable RRC scheme is* **unlinkable** *if for all* PPT*adversaries $\mathcal{A}$ the function $\mathsf{Adv}^{\mathrm{rrc}}_{\mathcal{A},\mathsf{R},B}(\lambda)$ is negligible.*

We make two remarks on the unlinkability definition:

- Looking ahead, for some applications, we might want the scheme to remain unlinkable even if adversarial re-randomizations were applied to it at some point. We present such a definition in Section 6. We also discuss ways to augment our basic LWE-based and Ring-LWE-based constructions to accommodate this stronger security definition. Since the stronger unlinkability definition is much more complicated than the one in Fig. 1, we first focus on this weaker notion.
- An unlinkable RRC scheme is, in particular, *hiding*. Meaning, that a commitment c leaks no information (in a computational sense) regarding the committed key k. Intuitively, an adversary that can distinguish between a commitment to a key k and a commitment to a different key k' can trivially link a commitment c to either a commitment c_0 to k_0 or to a commitment c_1 to k_1 by outputting the bit b such that c is a commitment to k_b.

3.3 An RRC scheme based on DDH

Equipped with the above definitions, we can briefly recall the DDH-based RRC scheme used in [16]. The scheme, called $\mathsf{R}_{\mathrm{ddh}}$, is defined by:

- $\mathit{Setup}(1^\lambda)$: choose a finite cyclic group $\mathbb{G}$ with generator $g \in \mathbb{G}$ and output $pp := (\mathbb{G}, g)$.
- $\mathit{Commit}(pp)$: choose random $u \xleftarrow{\$} \mathbb{G}$ and $k \xleftarrow{\$} \mathbb{Z}_q$, set $c \leftarrow (u, u^k)$, and output (c, k).
- $\mathit{Randomize}(pp, c)$: parse $c = (u, v)$, choose a random $\rho \xleftarrow{\$} \mathbb{Z}_q$, and output $c' := (u^\rho, v^\rho)$.
- $\mathit{Test}(pp, c, k)$: parse $c = (u, v)$ and output 1 iff $u^k = v$, otherwise output 0.

▶ **Theorem 13** ([16]). *If the DDH assumption holds in $\mathbb{G}$ then R_{ddh} is a perfectly-binding, unlinkable, and fully randomizable RRC.*

The fact that the scheme is full-randomizable and perfectly binding is easy to observe. The proof of unlinkability is a direct application of DDH. In the next section, we construct an RRC scheme that is post-quantum secure based on the LWE assumption.

4 A Construction from Learning with Errors

In this section, we present a construction of an RRC scheme from the LWE assumption [39] (see Section 2). An informal overview of the construction is presented in Section 1.1.

4.1 The Construction

Our construction of an RRC scheme from LWE, denoted $\mathsf{R}_{\mathsf{lwe}}$ is presented in Fig. 2. The construction is parameterized by an integer B, which serves as a bound on the number of re-randomizations that can be applied to a commitment. In the construction, we use Δ to denote $(C \cdot \sqrt{m})^B \cdot \delta$, where m is a parameter of the scheme determined by the analysis (think of $m = O(\lambda)$), C is the universal constant from Lemma 6 and δ is a bound on the ℓ_2 norm of the LWE noise vectors used in the construction.

Correctness. First, note that prior to any randomization being preformed, for an honestly-generated commitment $c = (\boldsymbol{A}, \boldsymbol{U})$ it holds that $\boldsymbol{A} \cdot \mathsf{H}(k) - \boldsymbol{U}$ is equal to the noise matrix $\boldsymbol{E}$ sampled according to $\chi^{m \times \ell}$ during the generation of the commitment. Hence, the matrix computed by the *Test* algorithm is simply $\boldsymbol{E}$, and each of its columns has norm at most δ. Now, after $t \leq B$ applications of *Randomize* to c using matrices $\boldsymbol{R}_1, \ldots, \boldsymbol{R}_t$, the commitment we get is of the form

$$(\boldsymbol{R}_t \cdots \boldsymbol{R}_1 \cdot \boldsymbol{A}, \boldsymbol{R}_t \cdots \boldsymbol{R}_1 \cdot \boldsymbol{U}) = (\boldsymbol{R}_t \cdots \boldsymbol{R}_1 \cdot \boldsymbol{A}, \boldsymbol{R}_t \cdots \boldsymbol{R}_1 \cdot \boldsymbol{A} \cdot \mathsf{H}(k) + \boldsymbol{R}_t \cdots \boldsymbol{R}_1 \cdot \boldsymbol{E}).$$

```
- Setup($1^\lambda$):
    Let $n := \lambda$, choose a prime $q$, and choose $m = m(n, q)$ and $\ell = \ell(n, q)$.
        // we will explain how to choose $m$ and $\ell$ in the analysis
    Let $\chi$ be the LWE noise distribution over $\mathbb{Z}_q$.
        // if $\boldsymbol{e} \xleftarrow{\$} \chi^m$, and we lift $\boldsymbol{e}$ to $\mathbb{Z}^m$, then with high probability, $\|\boldsymbol{e}\|_2 \leq \delta$ for some $\delta \ll q$
    return $pp \leftarrow (\lambda, q, n, m, \ell, \chi)$
- Commit($pp$):
    $\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$   // choose a random matrix $\boldsymbol{A}$
    $k \xleftarrow{\$} \{0, 1\}^{1^\lambda}$   // choose a random $\lambda$-bit string
    $\boldsymbol{V} \leftarrow \mathsf{H}(k) \in \mathbb{Z}_q^{n \times \ell}$   // hash $k$ to an $n$-by-$\ell$ matrix
    sample $\boldsymbol{E} \in \mathbb{Z}_q^{m \times \ell}$ from the LWE noise distribution $\chi^{m \times \ell}$
        // then for each column $\boldsymbol{e}$ of $\boldsymbol{E}$, $\|\boldsymbol{e}\|_2 \leq \delta$ w.h.p when $\boldsymbol{e}$ is lifted to $\mathbb{Z}^m$
    $\boldsymbol{U} \leftarrow \boldsymbol{A} \cdot \boldsymbol{V} + \boldsymbol{E} \in \mathbb{Z}_q^{m \times \ell}$
    $c \leftarrow (\boldsymbol{A}, \boldsymbol{U})$
    return $(c, k)$
- Randomize($pp, c$): parse $c = (\boldsymbol{A}, \boldsymbol{U})$ and do
    sample a random matrix $\boldsymbol{R} \xleftarrow{\$} \{-1, 1\}^{m \times m}$   // $\boldsymbol{R}$ is a low-norm matrix
    $c' \leftarrow (\boldsymbol{R} \cdot \boldsymbol{A},\ \boldsymbol{R} \cdot \boldsymbol{U}) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{m \times \ell}$
    return $c'$
- Test($pp, c, k$): parse $c = (\boldsymbol{A}, \boldsymbol{U})$ and do
    $\boldsymbol{V} \leftarrow \boldsymbol{A} \cdot \mathsf{H}(k) - \boldsymbol{U} \in \mathbb{Z}_q^{m \times \ell}$
    return 1 iff for each column $\boldsymbol{v}$ of $\boldsymbol{V}$, $\|\boldsymbol{v}\|_2 \leq \Delta$ when $\boldsymbol{v}$ is lifted to $\mathbb{Z}^m$
    Otherwise, return 0
```

Figure 2 $\mathsf{R}_{\mathsf{lwe}}$ – A B-randomizable RRC scheme based on the learning with errors (LWE) problem.

Hence, the matrix computed by the *Test* algorithm is $\boldsymbol{E}' = \boldsymbol{R}_t \cdots \boldsymbol{R}_1 \cdot \boldsymbol{E}$. Since $\boldsymbol{R}_1, \ldots, \boldsymbol{R}_t$ are sampled independently from $\{-1, 1\}^{m \times m}$, Lemma 6 guarantees that with overwhelming probability, each column of $\boldsymbol{E}'$ has norm at most $(C \cdot \sqrt{m})^t \cdot \delta \leq (C \cdot \sqrt{m})^B \cdot \delta = \Delta$.

4.2 Binding

▶ **Theorem 14.** *The above scheme is computationally binding when* H *is modeled as a random oracle. Concretely, for every adversary* $\mathcal{A}$ *making at most* Q *queries to* H *it holds that*

$$\Pr\left[\begin{array}{c} k \neq k' \text{ AND} \\ \mathit{Test}(pp, c, k) = \mathit{Test}(pp, c, k') = 1 \end{array} : \begin{array}{c} pp \xleftarrow{\$} \mathit{Setup}(1^\lambda) \\ (c, k, k') \xleftarrow{\$} \mathcal{A}(pp) \end{array}\right] \leq Q^2 \cdot q^n \cdot \left(\frac{4\Delta + 1}{q}\right)^\ell$$

The proof of Theorem 14 can be found in the full version.

4.3 Unlinkability

▶ **Theorem 15.** *The above construction is unlinkable. In particular, for every* PPT *adversary* $\mathcal{A}$ *making at most* $Q = Q(\lambda)$ *queries to* H*, there exists a* PPT *adversary* $\mathcal{B}$ *such that for all* $\lambda \in \mathbb{N}$ *it holds that:*

$$\mathsf{Adv}^{\mathrm{rrc}}_{\mathcal{A},\mathsf{R}_{\mathsf{LWE}},B}(\lambda) \leq \frac{2Q}{2^\lambda - Q} + 2\ell \cdot \mathsf{Adv}^{\mathrm{lwe}}_{\mathcal{B}}(\lambda) + \frac{B \cdot m}{2} \cdot \sqrt{2^{-m+(n+\ell)\cdot \log q}}.$$

The proof of the theorem is in the full version.

5 A Construction from Ring LWE

In this section we present our RRC construction from the Ring LWE assumption [33] (see Section 2). Our construction, denoted $\mathsf{R}_{\mathsf{rlwe}}$ is presented in Fig. 3. The construction works in a polynomial ring $\mathcal{R}$ modulo a cyclotomic polynomial f that has exactly two irreducible factors f_1, f_2 over $\mathbb{Z}_q$.

Improvements over $\mathsf{R}_{\mathsf{LWE}}$. Compared to the integer-based scheme, the ring-based scheme accommodates more efficient parameter choices. For concreteness, the ensuing discussion focuses on the regime in which $q = \Omega(\Delta^2)$. In this regime, for the ring-based scheme to be binding, we only need ℓ to be $\Omega(\log(q) + \lambda/n)$, where λ is the security parameter. This is a factor of $\Omega(n)$ smaller than the LWE case. Secondly, m only needs to be of order $\Omega(\log(q) + (\ell + \kappa)/n)$ where κ is a statistical security parameter (we want the re-randomized commitments to be distributed $1/2^\kappa$ close to a uniform distribution). This also turns out to be a factor of $\Omega(n)$ smaller than the integer case. Combining these together, each ring-based RRC commitment and each re-randomization matrix is $\Omega(n)$-times smaller than the integer-based commitment and matrix, respectively (this already takes into account the fact that representing each ring element takes n-times the representation length of a $\mathbb{Z}_q$ element).

Additionally, if we are re-randomizing a list of t commitments, then we consider the possibility of using a single, larger matrix to re-randomize all the commitments. In the ring case, we would only need to scale m by a factor of t/n, but in the integer case, m grows by a factor of t. In particular, for $t = \Omega(n)$, the ring-based RRC commitments and the re-randomization matrix only grow by a constant factor, while in the integer case, the commitments and the re-randomization matrix still grow linearly in t (making it essentially infeasible to use a single re-randomization matrix in this setting)[2].

We now prove the correctness, binding, and unlinkability for our ring-based RRC scheme.

Correctness. We first note that, prior to any rerandomization, for an honestly generated commitment $c = (\boldsymbol{a}, \boldsymbol{U})$, it holds that $\boldsymbol{U} - \boldsymbol{a} \cdot \mathsf{H}(k)^T$ is equal to the noise matrix $\boldsymbol{E}$ sampled at the generation of the commitments. Hence, the matrix $\boldsymbol{V}$ computed by the *Test* algorithm is just $\boldsymbol{E}$, and each of its columns has norm at most δ, since $\boldsymbol{E}$ was sampled according to $\chi^{m\times\ell}$. Now, after $t \leq B$ applications of *Randomize* to the commitment c using matrices $\boldsymbol{R}_1, \ldots, \boldsymbol{R}_t$, the commitment we get is of the form

$$(\boldsymbol{R}_t \cdots \boldsymbol{R}_1 \cdot \boldsymbol{a}, \boldsymbol{R}_t \cdots \boldsymbol{R}_1 \cdot \boldsymbol{U}) = (\boldsymbol{R}_t \cdots \boldsymbol{R}_1 \cdot \boldsymbol{a}, \boldsymbol{R}_t \cdots \boldsymbol{R}_1 \cdot \boldsymbol{a} \cdot \mathsf{H}(k)^T + \boldsymbol{R}_t \cdots \boldsymbol{R}_1 \cdot \boldsymbol{E}).$$

Hence, the matrix computed by the $Test$ algorithm is $\boldsymbol{E}' = \boldsymbol{R}_t \cdots \boldsymbol{R}_1 \cdot \boldsymbol{E}$. Lemma 7 guarantees that with a high probability, each column of $\boldsymbol{E}'$ has norm at most

$$(m\sqrt{mn} \cdot \omega(\sqrt{\log n}))^t \cdot \delta \leq (m\sqrt{mn} \cdot \omega(\sqrt{\log n}))^B \cdot \delta = \Delta$$

where Δ is an upper bound on the expression $\delta \cdot (m\sqrt{mn} \cdot \omega(\sqrt{\log n}))^B$.

[2] While the re-randomization matrix would typically not be transmitted in the clear, its size does affect the complexity of the proof of shuffle (recall our discussion in the introduction).

```
- Setup($1^\lambda$):
    Let $n := 2^r$ where $r = r(\lambda)$, and let $f(x) = x^n + 1$ and $\mathcal{R} = \mathbb{Z}[x]/f(x)$.
    Choose a prime $q$ and let $\mathcal{R}_q = \mathbb{Z}_q[x]/f(x)$.
      // $r, q$ are chosen such that $f$ factors into two irreducible polynomials over $\mathbb{Z}_q$
    Let $m = m(n, q)$ and $\ell = \ell(n, q)$.
      // we will explain how to choose $m$ and $\ell$ in the analysis
    Let $\chi$ be the RLWE noise distribution over $\mathcal{R}_q$.
      // if we sample a vector $\boldsymbol{e} \xleftarrow{\$} \chi^m$, then with high probability, $\|\boldsymbol{e}\|_2 \leq \delta$ for some $\delta \ll q$
    return $pp \leftarrow (\lambda, q, n, m, \ell, \chi)$
- Commit($pp$):
    $\boldsymbol{a} \xleftarrow{\$} \mathcal{R}_q^m$   // choose a random vector $\boldsymbol{a}$
    $k \xleftarrow{\$} \{0,1\}^\lambda$   // choose a random $1^\lambda$-bit string
    $\boldsymbol{v} \leftarrow \mathsf{H}(k) \in \mathcal{R}_q^\ell$   // hash $k$ to a vector of length $\ell$
    Sample $\boldsymbol{E} \in \mathcal{R}_q^{m\times\ell}$ from the RLWE noise distribution $\chi^{m\times\ell}$
      // then for each column $\boldsymbol{e}$ of $\boldsymbol{E}$, $\|\boldsymbol{e}\|_2 \leq \delta$ w.h.p
    $\boldsymbol{U} \leftarrow \boldsymbol{a} \cdot \boldsymbol{v}^T + \boldsymbol{E} \in \mathcal{R}_q^{m\times\ell}$
    $c \leftarrow (\boldsymbol{a}, \boldsymbol{U})$
    return $(c, k)$
- Randomize($pp, c$): parse $c = (\boldsymbol{a}, \boldsymbol{U})$ and do
    Sample $\boldsymbol{R} \in \mathcal{R}^{m\times m}$:
    $\forall i, j \in [m]$, sample the coefficients of $\boldsymbol{R}_{i,j}$ uniformly and independently from $\{-1, 1\}$
      // $\boldsymbol{R}$ is a low-norm matrix
    $c' \leftarrow (\boldsymbol{R}\cdot\boldsymbol{a},\ \boldsymbol{R}\cdot\boldsymbol{U}) \in \mathcal{R}_q^m \times \mathcal{R}_q^{m\times\ell}$
    return $c'$
- Test($pp, c, k$): parse $c = (\boldsymbol{a}, \boldsymbol{U})$ and do
    $\boldsymbol{V} \leftarrow \boldsymbol{U} - \boldsymbol{a}\cdot\mathsf{H}(k)^T \in \mathcal{R}_q^{m\times\ell}$
    return 1 iff for each column $\boldsymbol{v}$ of $\boldsymbol{V}$, $\|\boldsymbol{v}\|_2 \leq \Delta$, and return 0 otherwise
```

Figure 3 $\mathsf{R}_{\mathsf{rlwe}}$ – A B-randomizable RRC scheme based on the learning with errors over rings (RLWE) problem.

5.1 Binding

▶ **Theorem 16.** *The above scheme is computationally binding when* H *is modeled as a random oracle. Concretely, for every adversary* $\mathcal{A}$ *making at most* Q *queries to* H *it holds that*

$$\Pr\left[\begin{array}{c} k \neq k' \text{ AND} \\ \mathit{Test}(pp, c, k) = \mathit{Test}(pp, c, k') = 1 \end{array} : \begin{array}{c} pp \xleftarrow{\$} \mathit{Setup}(1^\lambda) \\ (c, k, k') \xleftarrow{\$} \mathcal{A}(pp) \end{array}\right] \leq Q^2 \cdot q^n \cdot \left(\frac{4\Delta + 1}{\sqrt{q}}\right)^{n\ell}$$

The proof of Theorem 16 is in the full version.

5.2 Unlinkability

▶ **Theorem 17.** *The above construction is unlinkable. In particular, for every* PPT *adversary* $\mathcal{A}$ *making at most* $Q = Q(\lambda)$ *queries to* H*, there exists a* PPT *adversary* $\mathcal{B}$ *such that for all* $\lambda \in \mathbb{N}$ *it holds that:*

$$\mathsf{Adv}^{\mathrm{rrc}}_{\mathcal{A},\mathsf{R}_{\mathsf{RLWE}},B}(\lambda) \leq \frac{2Q}{2^{\lambda}-Q} + (2\ell)\cdot\mathsf{Adv}^{\mathrm{rlwe}}_{\mathcal{B}}(\lambda) + B\cdot\frac{m}{2}\sqrt{\left(1+\left(\frac{q}{2^m}\right)^{n/2}\right)^{2(\ell+1)} - 1}.$$

The proof of Theorem 17 can be found in the full version.

6 Handling Adversarially-Randomized Commitments

In this section, we present a stronger notion of unlinkability, called *strong unlinkability* for RRC schemes, and then present different approaches to augment our basic schemes from Sections 4 and 5 to satisfy this definition.

Loosely speaking, strong unlinkability requires that re-randomization should result in unlinkable commitments, even if they were previously re-randomized by the adversary. This trivially holds for the DDH-based construction of Boneh et al. [16] thanks to two properties of the scheme:

- Suppose the adversary receives a commitment c for which k is a valid key, and outputs a randomized commitment c'. As long as $Test(pp, c', k) = 1$, there exists some randomness r such that $c' = Randomize(pp, c; r)$.
- Re-randomization using *Randomize* is a commutative operation. Hence, in conjunction with the observation above, any knowledge the adversary could gain by re-randomizing a commitment before an honest re-randomization, it could also gain by re-randomizing it afterwards (which the adversary can already do in the security game from Fig. 1).

Alas, this is not the case for our lattice-based constructions. The main issue is that matrix multiplication is not commutative. Hence a "bad" re-randomization (even one that does not invalidate the honest commitment key) can have a long lasting effect on a commitment even after many subsequent honest re-randomizations have taken place. Concretely, on input $c = (\boldsymbol{A}, \boldsymbol{U})$, the adversary may output $c' = (\boldsymbol{A}', \boldsymbol{U}')$, such that $\boldsymbol{A}'$ is "bad" in the sense that the distribution $R \cdot \boldsymbol{A}'$ for a random $R \xleftarrow{\$} \{-1, 1, \}^{m\times m}$ is very far from the uniform distribution over $\mathbb{Z}_q^{m\times n}$. As a hypothetical example, suppose that the adversary can find a matrix $\boldsymbol{R} \in \{-1, 1, \}^{m\times m}$ such that $\boldsymbol{A}' = \boldsymbol{R} \cdot \boldsymbol{A}$ is a low-norm matrix. Then, the distribution $R \cdot \boldsymbol{A}'$ will be concentrated on low-norm matrices as well, enabling the adversary to distinguish between this distribution and the uniform distribution over $\mathbb{Z}_q^{m\times n}$, which is concentrated on high-norm matrices.

6.1 A Stronger Unlinkability Definition

We first need to define what it means for an RRC scheme to be unlinkable in the face of adversarial re-randomizations. To do this, we augment the security game of RRC schemes by letting the adversary re-randomize the commitments at points in time of its choosing. To avoid trivial attacks, we require that the adversary justifies its outputs by providing the randomness it used for re-randomization.

To this end, and to facilitate our constructions, we introduce several new notions for RRC schemes:

- We augment an RRC scheme with a corresponding *beacon distribution* D. This distribution is used to model a randomness beacon, and will be used by one of our constructions of a strongly-unlinkable RRC scheme. In practice, the beacon may be assumed as an outside resource or implemented in various ways using known techniques [37].
- We introduce two new algorithms $\mathsf{R}.\mathit{Precommit}$ and $\mathsf{R}.\mathit{Extract}$ to an RRC scheme R. $\mathsf{R}.\mathit{Precommit}$ is a randomized algorithm that takes in the public parameters pp an outputs some "precommitment" pcom, whose role will become apparent in a minute. $\mathsf{R}.\mathit{Extract}$

is a (potentially randomized) algorithm that takes in pp, the randomness $r \in \{0,1\}^*$ used by $\mathsf{R}.\mathit{Precommit}$ to generate pcom, and a sample rand from D, and outputs some randomness r' to be used by $\mathsf{R}.\mathit{Randomize}$. Throughout this section, we will denote the number of random coins used by $\mathsf{R}.\mathit{Precommit}$ by $\rho = \rho(\lambda)$.

- An RRC scheme R is now also parameterized by a class $\mathcal{G}$ of *admissible random strings* , and only members of $\mathcal{G}$ can be used as randomness for $\mathsf{R}.\mathit{Randomize}$. This is checked by the security game for randomness used by the adversary. A natural selection for $\mathcal{G}$ is the entire support of the randomness used by the honest *Randomize* algorithm; for example, in our (integer) LWE-based construction, this corresponds to $\mathcal{G} = \{-1,1\}^{m\times m}$, but one might also consider strict supersets or subsets of this set.

  We allow $\mathcal{G}$ to depend on a precommitment pcom, the randomness $r \in \{0,1\}^*$ used by $\mathsf{R}.\mathit{Precommit}$ to generate pcom, and a sample rand from D. We denote this by $\mathcal{A}(\mathsf{pcom}, r, \mathsf{rand})$. The set $\mathcal{G}$ may also depend on the public parameters pp, but we do not note this explicitly, since the public parameters typically remain fixed.

To recap, an RRC scheme R now consists of six algorithms ($\mathsf{R}.\mathit{Setup}$, $\mathsf{R}.\mathit{Commit}$, $\mathsf{R}.\mathit{Randomize}$, $\mathsf{R}.\mathit{Test}$, and now also $\mathsf{R}.\mathit{Precommit}$ and $\mathsf{R}.\mathit{Extract}$), a distribution D, and a set $\mathcal{G} = \mathcal{G}(\mathsf{pcom}, r, \mathsf{rand})$.

Correctness and unlinkability. For correctness, we now require that the scheme is B-rerandomizable (Definition 8), where the randomness for rerandomization is generated by *Precommit*, D, and *Extract*. We additionally require that honestly generated randomness for *Randomize* is indeed admissible.

▶ **Definition 18.** *Let* R *be an RRC scheme such that* $\mathsf{R}.\mathit{Precommit}$ *takes* $\rho = \rho(\lambda)$ *random coins. We say* R*is* B**-randomizable** *if there exists a negligible function* $\nu(\cdot)$ *such that the following conditions hold for every* $\lambda \in \mathbb{N}$*:*

1. *Let* $pp \xleftarrow{\$} \mathsf{R}.\mathit{Setup}(1^\lambda)$, $(c_0, k) \xleftarrow{\$} \mathsf{R}.\mathit{Commit}(pp)$, $r_i \xleftarrow{\$} \{0,1\}^\rho$, $\mathsf{rand}_i \xleftarrow{\$} D$, $r'_i \xleftarrow{\$} \mathsf{R}.\mathit{Extract}(pp, r_i, \mathsf{rand}_i)$, $c_i \xleftarrow{\$} \mathsf{R}.\mathit{Randomize}(pp, c_{i-1}; r'_i)$ *for* $i \in [B]$*, then*

$$\Pr\Big[\mathsf{R}.\mathit{Test}(pp, c_i, k) = 1 \quad \text{for } i = 0,1,2,\ldots,B\Big] \geq 1 - \nu(\lambda).$$

2. *Let* $pp \xleftarrow{\$} \mathsf{R}.\mathit{Setup}(1^\lambda)$, $r \xleftarrow{\$} \{0,1\}^\rho$, $\mathsf{pcom} \leftarrow \mathsf{R}.\mathit{Precommit}(pp; r)$, $\mathsf{rand} \xleftarrow{\$} D$*, and* $r' \xleftarrow{\$} \mathsf{R}.\mathit{Extract}(pp, r, \mathsf{rand})$*, then*

$$\Pr\left[r' \in \mathcal{G}(\mathsf{pcom}, r, \mathsf{rand})\right] \geq 1 - \nu(\lambda).$$

An RRC scheme that is B*-randomizable for all* $B \in \mathbb{N}$ *is said to be* **fully randomizable***.* ⌟

The new strong-unlinkability game is defined in Figure 4. It uses the following abbreviated writing: we write $(\mathsf{rand}, c') \xleftarrow{\$} \mathsf{R}.\mathit{Randomize}(pp, r, c)$ as a shorthand for the process of (1) sampling $\mathsf{rand} \xleftarrow{\$} D$, (3) sampling $r' \xleftarrow{\$} \mathsf{R}.\mathit{Extract}(pp, r, \mathsf{rand})$, (4) computing $c' \leftarrow \mathsf{R}.\mathit{Randomize}(pp, c; r')$, and (5) outputting (rand, c'). The new game is obtained from the old unlinkability security game (Figure 1) by the following modifications:

1. At the onset of the game, the challenger samples precommitments $\{\mathsf{pcom}\}$ to be used for the honest re-randomizations it performs. The adversary then also outputs a precommitment $\{\mathsf{pcom}\}$ for its own future re-randomizations. For each re-randomization, the set $\mathcal{G}$ will depend on the corresponding precommitment. Looking ahead, in a couple of our constructions, the precommitments will serve as commitments for randomness to be used in the future re-randomizations.

2. The challenger samples $\{\mathsf{rand}\}$ values from the beacon distribution D. These serve as the beacon values for each re-randomization (adversarial or honest). The adversary receives the corresponding rand value before each adversarial re-randomization, and together with each honest re-randomization.
3. Each time the adversary $\mathcal{A}$ outputs re-randomized commitments, it also outputs the associated randomness used to generate the associated precommitment and the randomness used for re-randomization. The challenger then checks that this randomness is indeed admissible.

   As before, we define the adversary's advantage as

$$\mathsf{Adv}^{\text{strong-rrc}}_{\mathcal{A},\mathsf{R}}(\lambda) := \left|2\Pr[\mathbf{G}^{\text{strong}}_{\mathcal{A},\mathsf{R}}(\lambda) = 1] - 1\right|.$$

```
Game $\boldsymbol{G}^{\text{strong}}_{\mathcal{A},\mathsf{R}}(\lambda)$
$b \xleftarrow{\$} \{0,1\}$
$pp \xleftarrow{\$} \mathsf{R}.Setup(1^\lambda)$
$(T, i_0^{(1)}, i_1^{(1)} \ldots, i_0^{(T)}, i_1^{(T)}, \mathsf{state}) \xleftarrow{\$} \mathcal{A}(pp)$
$\overrightarrow{\mathsf{pcom}} \leftarrow ()$   // initialize an empty vector
for $t$ in $\{1, \ldots, T\}$ :
    for $j$ in $\{1, \ldots, i_0^{(t)}\}$ :  $r_{t,0,j} \xleftarrow{\$} \{0,1\}^\rho$,  $\mathsf{pcom}_{t,0,j} \leftarrow \mathsf{R}.Precommit(pp; r_{t,0,j})$,  $\overrightarrow{\mathsf{pcom}} \leftarrow \overrightarrow{\mathsf{pcom}} \| \mathsf{pcom}_{t,0,j}$
    for $j$ in $\{1, \ldots, i_1^{(t)}\}$ :  $r_{t,1,j} \xleftarrow{\$} \{0,1\}^\rho$,  $\mathsf{pcom}_{t,1,j} \leftarrow \mathsf{R}.Precommit(pp; r_{t,1,j})$,  $\overrightarrow{\mathsf{pcom}} \leftarrow \overrightarrow{\mathsf{pcom}} \| \mathsf{pcom}_{t,1,j}$
        // $\rho$ denotes the number of random coins used by $\mathsf{R}.Precommit$
$(\mathsf{pcom}'_{1,0}, \mathsf{pcom}'_{1,1}, \ldots, \mathsf{pcom}'_{T,0}, \mathsf{pcom}'_{T,1}, \mathsf{state}) \xleftarrow{\$} \mathcal{A}(\mathsf{state}, \overrightarrow{\mathsf{pcom}})$
$\mathsf{rand}_{1,0}, \mathsf{rand}_{1,1} \ldots, \mathsf{rand}_{T,0}, \mathsf{rand}_{T,1} \xleftarrow{\$} D$
$(c_0^{(0)}, k_0) \xleftarrow{\$} \mathsf{R}.Commit(pp)$,   $(c_1^{(0)}, k_1) \xleftarrow{\$} \mathsf{R}.Commit(pp)$
$\mathsf{aux}_0 \leftarrow c_0^{(0)} \| c_1^{(0)}$
for $t$ in $\{1, \ldots, T\}$ :
    $(\mathsf{state}, c_0^{(t)}, c_1^{(t)}, r_0, r_1, r'_0, r'_1) \xleftarrow{\$} \mathcal{A}(\mathsf{state}, \mathsf{aux}_{t-1}, \mathsf{rand}_{t,1}, \mathsf{rand}_{t,0})$
        // $r_0$ and $r_1$ are the random coins $\mathcal{A}$ claims to have used to generate $\mathsf{pcom}_{t,0}$ and $\mathsf{pcom}_{t,1}$
        // $r'_0$ and $r'_1$ are the random coins $\mathcal{A}$ claims to have used for re-randomization
    if $(c_0^{(t)} \neq \mathsf{R}.Randomize(pp, c_0^{(t-1)}; r'_0))$ OR $(c_1^{(t)} \neq \mathsf{R}.Randomize(pp, c_1^{(t-1)}; r'_1))$ :  abort
        // check that $r'_0$ and $r'_1$ were used by $\mathcal{A}$ for re-randomization
    for $d$ in $\{0,1\}$  :  $\mathcal{G}_d \leftarrow \mathcal{G}(\mathsf{pcom}'_{t,d}, r_d, \mathsf{rand}_{t,d})$
    $\mathsf{aux}_t \leftarrow ()$   // initialize an empty vector
    for $j$ in $\{1, \ldots, i_0^{(t)}\}$ :  $(\mathsf{rand}^t_{0,j}, c_0^{(t)}) \xleftarrow{\$} \mathsf{R}.Randomize(pp, r_{t,0,j}, c_0^{(t)})$,  $\mathsf{aux}_t \leftarrow \mathsf{aux}_t \| (\mathsf{rand}^t_{0,j}, c_0^{(t)})$
    for $j$ in $\{1, \ldots, i_1^{(t)}\}$ :  $(\mathsf{rand}^t_{1,j}, c_1^{(t)}) \xleftarrow{\$} \mathsf{R}.Randomize(pp, r_{t,1,j}, c_1^{(t)})$,  $\mathsf{aux}_t \leftarrow \mathsf{aux}_t \| (\mathsf{rand}^t_{1,j}, c_1^{(t)})$
        // the notation $(\mathsf{rand}, c') \xleftarrow{\$} \mathsf{R}.Randomize(pp, r, c)$ is defined above
    if $r'_0 \notin \mathcal{G}_0$ OR $r'_1 \notin \mathcal{G}_1$ :  $c_0^{(t)} \leftarrow c_0^{(t-1)}$,  $c_1^{(t)} \leftarrow c_1^{(t-1)}$
$(\mathsf{rand}_0, c_0) \xleftarrow{\$} \mathsf{R}.Randomize(pp, c_0^{(T)})$,  $(\mathsf{rand}_1, c_1) \xleftarrow{\$} \mathsf{R}.Randomize(pp, c_1^{(T)})$
$b' \xleftarrow{\$} \mathcal{A}(c_b, c_{1-b}, \mathsf{rand}_0, \mathsf{rand}_1, \mathsf{state})$
return $b = b'$
```

Figure 4 The strong unlinkability security game for an adversary $\mathcal{A}$ and an RRC scheme R.

▶ **Definition 19.** *An RRC scheme* R *is* **strongly-unlinkable** *if for all* PPT *adversaries* $\mathcal{A}$ *the function* $\mathsf{Adv}^{\text{strong-rrc}}_{\mathcal{A},\mathsf{R}}(\lambda)$ *is negligible.*

```
Game $\mathbf{G}^{\mathrm{pr}}_{\mathcal{A},\mathsf{R}}(\lambda)$
$b \xleftarrow{\$} \{0,1\}$
$pp \xleftarrow{\$} \mathsf{R}.\mathit{Setup}(1^{\lambda})$
$r \xleftarrow{\$} \{0,1\}^{\rho}$, $\mathsf{pcom} \leftarrow \mathsf{R}.\mathit{Precommit}(pp; r)$
$(\mathsf{pcom}', \mathsf{state}) \xleftarrow{\$} \mathcal{A}(pp, \mathsf{pcom})$
$\mathsf{rand} \xleftarrow{\$} D$
$(c, k) \xleftarrow{\$} \mathsf{R}.\mathit{Commit}(pp)$, $(c_0', k') \xleftarrow{\$} \mathsf{R}.\mathit{Commit}(pp)$
$(c'', r', r'', \mathsf{state}) \xleftarrow{\$} \mathcal{A}(\mathsf{state}, c, \mathsf{rand})$
if $c'' \neq \mathsf{R}.\mathit{Randomize}(pp, c; r'')$ : abort
if $r'' \notin \mathcal{G}(\mathsf{pcom}', r', \mathsf{rand})$ : abort
$(\mathsf{rand}', c_0) \xleftarrow{\$} \mathsf{R}.\mathit{Randomize}(pp, r, c'')$
$c_1, c_1' \xleftarrow{\$} \mathcal{C}_{\lambda}$
$b' \xleftarrow{\$} \mathcal{A}(c_b, c_b', \mathsf{rand}', \mathsf{state})$
return $b = b'$
```

Figure 5 The security game for an adversary $\mathcal{A}$ attacking the strong pseudorandomness of R.

How to put the strong unlinkability definition to use. In the strengthened security game from Fig. 4, whenever the adversary re-randomizes, it also sends to the challenger the randomness that went into this re-randomization process (that is, the randomness that went into *Precommit* and into *Randomize*). This means that whenever using a strongly-unlinkable RRC scheme within a larger protocol, one should require that re-randomizers provide a argument of knowledge for such randomness (and potentially of additional secrets that are related to the larger super-protocol). Then, a security reduction that tries to break the security of the RRC scheme can use the knowledge extractor of the proof system to extract the randomness and output it in the RRC security game. Our SSLE protocol, detailed in the full version, provides an example of how to use RRC schemes within a larger protocol. In the full version, we discuss specific ways to construct the necessary arguments of knowledge for our lattice-based RRC schemes.

Strongly-pseudorandom RRC schemes. We present the notion of strong pseudorandomness for RRC schemes. Roughly speaking, an RRC scheme enjoys strong pseudorandomness, if honestly re-randomized commitments are pseudorandom. That is, it is indistinguishable from a uniformly-random member of the domain $\mathcal{C} = \{\mathcal{C}_{\lambda}\}_{\lambda}$ of commitments. Moreover, honest re-randomization should output pseudorandom commitments even on commitments that were previously re-randomized by the adversary (using admissible randomness). This is captured by the security game in Fig. 5.

As before, we define the adversary's advantage as

$$\mathsf{Adv}^{\text{pr-rrc}}_{\mathcal{A},\mathsf{R}}(\lambda) := \left|2\Pr[\mathbf{G}^{\mathrm{pr}}_{\mathcal{A},\mathsf{R}}(\lambda) = 1] - 1\right|.$$

▶ **Definition 20.** *A B-randomizable* R *scheme is* **strongly-pseudorandom** *if for all* PPT *adversaries $\mathcal{A}$ the function* $\mathsf{Adv}^{\text{pr-rrc}}_{\mathcal{A},\mathsf{R}}(\lambda)$ *is negligible.*

A simple hybrid argument shows that an RRC scheme that is strongly-pseudorandom is also strongly-unlinkable.

▶ **Proposition 21.** *If an RRC scheme* R *is strongly-pseudorandom then it is also strongly-unlinkable.*

We now turn to present several ways to augment our basic RRC schemes so that they achieve strong-pseudorandomness, and hence strong unlinkability.

6.2 Constructing Strongly-Pseudorandom RRCs

We now present a way to turn our lattice-based constructions of RRC schemes to ones that provide strong pseudorandomness, and hence strong unlinkability. We start by describing such a mechanism for our LWE-based scheme, and then discuss how the same ideas can also be applied to our Ring-LWE-based scheme.

Immunizing our LWE-based RRC scheme $\mathsf{R}_{\mathsf{LWE}}$ against adversarial re-randomizations per the Definition 19 amounts to defining the beacon distribution D, the algorithms $\mathsf{R}_{\mathsf{LWE}}.\mathit{Precommit}$ and $\mathsf{R}_{\mathsf{LWE}}.\mathit{Extract}$, and the set $\mathcal{G}$ of admissible random strings. We do so as follows:

- D is the uniform distribution over $\{-1,1\}^{m\times m}$.
- $\mathsf{R}_{\mathsf{LWE}}.\mathit{Precommit}(pp;r)$: the randomness r to the algorithm is parsed as a tuple $(\boldsymbol{R},r')$ of a uniformly-random matrix $\boldsymbol{R}$ in $\{-1,1\}^{m\times m}$ and randomness r' to a standard (not re-randomizable) statistically-binding non-interactive commitment scheme $\mathsf{C}=(\mathsf{C}.\mathit{Setup},\mathsf{C}.\mathit{Commit})$ (for definitions of standard commitment schemes, see for example [17]). The algorithm then commits to $\boldsymbol{R}$ using C: it computes $\mathsf{pcom}\leftarrow\mathsf{C}.\mathit{Commit}(pp_{\mathsf{C}},\boldsymbol{R};r')$ and outputs pcom (the public parameters pp_{C} for C are sampled by the $\mathsf{C}.\mathit{Setup}$ algorithm during the operation of $\mathsf{R}_{\mathsf{LWE}}.\mathit{Setup}$ and are included as part of the public parameters of $\mathsf{R}_{\mathsf{LWE}}$).
- $\mathsf{R}_{\mathsf{LWE}}.\mathit{Extract}(pp,r,\mathsf{rand})$ parses r as $(\boldsymbol{R},r')$ and treats rand as a matrix $\boldsymbol{R}'$ in $\{-1,1\}^{m\times m}$. It outputs $\boldsymbol{R}''\leftarrow\boldsymbol{R}+\boldsymbol{R}'\in\mathbb{Z}_q^{m\times m}$.
- The set $\mathcal{G}=\mathcal{G}(\mathsf{pcom},r=(\boldsymbol{R},r'),\mathsf{rand}=\boldsymbol{R}')$ is then the singleton set $\{\boldsymbol{R}+\boldsymbol{R}'\}$ if $\mathsf{pcom}=\mathsf{C}.\mathit{Commit}(pp_{\mathsf{C}},\boldsymbol{R};r')$. Otherwise, if $\mathsf{pcom}\neq\mathsf{C}.\mathit{Commit}(pp_{\mathsf{C}},\boldsymbol{R};r')$ then $\mathcal{G}=\emptyset$ and there is no admissible randomness. That is, $\mathcal{G}$ "checks" if pcom is a valid commitment to $\boldsymbol{R}$ given the randomness used to generate it, and if so, the only admissible randomness for $\mathsf{R}_{\mathsf{LWE}}.\mathit{Randomize}$ is the sum of $\boldsymbol{R}+\boldsymbol{R}'$.

We denote the RRC scheme obtained by these augmentations by $\mathsf{R}^{+}_{\mathsf{LWE}}$. We first argue that the scheme is correct per Definition 18. Condition 2 of the definition holds trivially. To see why Condition 1 holds, observe that honest r_is used for re-randomization are now m-by-m matrices, whose coordinates are independently sampled from a distribution which attains 0 with probability 1/2, and −2 or 2 with probability 1/4 each. A straightforward adaptation of the proof of Lemma 6 shows that it still applies (with a slightly worse constant C) and hence the previous proof of correctness still goes through.

As for security, the following theorem, proved in the full version, proves that $\mathsf{R}^{+}_{\mathsf{LWE}}$ satisfies strong pseudorandomness. In conjunction with Proposition 21, this implies that it is also strongly-unlinkable.

▶ **Theorem 22.** *The scheme* $\mathsf{R}^{+}_{\mathsf{LWE}}$ *is a strongly-pseudorandom RRC scheme.*

Strong unlinkability from the Ring LWE assumption. We can use a similar technique in order to augment our Ring-LWE-based RRC scheme with strong unlinkability. The only difference is that now $\boldsymbol{R}$ and $\boldsymbol{R}'$ are sampled as matrices of "short" polynomials. That is, the distribution D samples a matrix $\boldsymbol{R}'$ as follows: Each coordinate is an independent polynomial, whose coefficients are sampled independently and uniformly from $\{-1,1\}$. *Precommit* samples a commitment to a matrix $\boldsymbol{R}$ sampled from the same distribution, and *Extract* outputs $\boldsymbol{R}+\boldsymbol{R}'$. Finally, the set $\mathcal{G}(\mathsf{pcom}, r, \mathsf{rand}) = \{\boldsymbol{R}+\boldsymbol{R}'\}$ as before if the precommitment pcom is consistent with r and $\emptyset$ otherwise. Correctness follows similarly as in the LWE case, replacing the use of Lemma 6 with Lemma 7. For strong pseudorandomness, we replace the use of the leftover hash lemma [29] with Lemma 4.[3]

6.3 Strong Pseudorandomness without A Randomness Beacon

The above approach requires a randomness beacon, which is a very reasonable assumption in the context of SSLE protocols. However, there might be other scenarios in which one might want to use RRCs without assuming the availability of such a beacon. This is formally captured by the above definitions by fixing D to be the constant distribution outputting $\perp$ with probability 1. In the full version, we present three different approaches to augment our schemes to provide strong unlinkability without assuming a randomness beacon.

References

1 Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (h)ibe in the standard model. In *Advances in Cryptology – EUROCRYPT 2010*, pages 553–572, 2010.

2 Martin R. Albrecht, Valerio Cini, Russell W. F. Lai, Giulio Malavolta, and Sri AravindaKrishnan Thyagarajan. Lattice-based snarks: Publicly verifiable, preprocessing, and recursively composable. In *Advances in Cryptology – CRYPTO 2022*, pages 102–132, 2022.

3 Joël Alwen and Chris Peikert. Generating shorter bases for hard random lattices. *Theory of Computing Systems*, 48:535–553, 2011.

4 Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 2087–2104, Dallas, TX, USA, October 31 – November 2 2017. ACM Press. doi:10.1145/3133956.3134104.

5 Prabhanjan Ananth, Apoorvaa Deshpande, Yael Tauman Kalai, and Anna Lysyanskaya. Fully homomorphic NIZK and NIWI proofs. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part II*, volume 11892 of *LNCS*, pages 356–385, Nuremberg, Germany, December 1–5 2019. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-36033-7_14.

6 Thomas Attema, Ronald Cramer, and Lisa Kohl. A compressed Σ-protocol theory for lattices. In *Advances in Cryptology – CRYPTO 2021*, pages 549–579, 2021.

7 Sarah Azouvi and Daniele Cappelletti. Private attacks in longest chain proof-of-stake protocols with single secret leader elections. In *Proceedings of the 3rd ACM Conference on Advances in Financial Technologies*, AFT '21, pages 170–182, 2021. doi:10.1145/3479722.3480996.

8 Sarah Azouvi, Patrick McCorry, and Sarah Meiklejohn. Betting on blockchain consensus with fantomette, 2018. arXiv:1805.06786.

9 Michael Backes, Pascal Berrang, Lucjan Hanzlik, and Ivan Pryvalov. A framework for constructing single secret leader election from MPC. In Vijayalakshmi Atluri, Roberto Di Pietro, Christian Damsgaard Jensen, and Weizhi Meng, editors, *ESORICS 2022, Part II*, volume 13555 of *LNCS*, pages 672–691, Copenhagen, Denmark, September 26–30 2022. Springer, Heidelberg, Germany. doi:10.1007/978-3-031-17146-8_33.

10 Carsten Baum, Jonathan Bootle, Andrea Cerulli, Rafael del Pino, Jens Groth, and Vadim Lyubashevsky. Sub-linear lattice-based zero-knowledge arguments for arithmetic circuits. In *Advances in Cryptology – CRYPTO 2018*, pages 669–699, 2018.

[3] Technically speaking, we require a generalization of Lemma 7, in which the coefficients of each entry of $\boldsymbol{R}$ may be chosen from different (but small) sets. Fortunately, the proof of 7 readily extends to this setting.

11 Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, , and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Paper 2018/046, 2018. URL: `https://eprint.iacr.org/2018/046`.

12 Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. Aurora: Transparent succinct arguments for R1CS. In *Advances in Cryptology – EUROCRYPT 2019*, pages 103–128, 2019.

13 Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In *Theory of Cryptography*, pages 31–60, 2016.

14 Fabrice Benhamouda, Stephan Krenn, Vadim Lyubashevsky, and Krzysztof Pietrzak. Efficient zero-knowledge proofs for commitments from learning with errors over rings. Cryptology ePrint Archive, Report 2014/889, 2014. URL: `https://eprint.iacr.org/2014/889`.

15 Rishabh Bhadauria, Zhiyong Fang, Carmit Hazay, Muthuramakrishnan Venkitasubramaniam, Tiancheng Xie, and Yupeng Zhang. Ligero++: A new optimized sublinear IOP. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 2025–2038, Virtual Event, USA, November 9–13 2020. ACM Press. `doi:10.1145/3372297.3417893`.

16 Dan Boneh, Saba Eskandarian, Lucjan Hanzlik, and Nicola Greco. Single secret leader election. In *AFT '20*, pages 12–24. ACM, 2020. Available online at eprint/2020/025.

17 Dan Boneh and Victor Shoup. *A Graduate Course in Applied Cryptography, Draft 0.6*. Cambridge University Press, 2023.

18 Dario Catalano, Dario Fiore, and Emanuele Giunta. Efficient and universally composable single secret leader election from pairings. Cryptology ePrint Archive, Report 2021/344, 2021. URL: `https://eprint.iacr.org/2021/344`.

19 Dario Catalano, Dario Fiore, and Emanuele Giunta. Adaptively secure single secret leader election from ddh. In *Proceedings of the 2022 ACM Symposium on Principles of Distributed Computing*, PODC'22, pages 430–439, 2022. `doi:10.1145/3519270.3538424`.

20 Rutchathon Chairattana-Apirom and Anna Lysyanskaya. Compact cut-and-choose: Boosting the security of blind signature schemes, compactly. Cryptology ePrint Archive, Paper 2022/003, 2022. URL: `https://eprint.iacr.org/2022/003`.

21 Miranda Christ, Valeria Nikolaenko, and Joseph Bonneau. Leader election from randomness beacons and other strategies, 2022. URL: `https://a16zcrypto.com/posts/article/leader-election-from-randomness-beacons-and-other-strategies`.

22 Nuria Costa, Ramiro Martínez, and Paz Morillo. Lattice-based proof of a shuffle. In *FC 2019: Financial Cryptography and Data Security*, pages 330–346, 2019.

23 Justin Drake. Low-overhead secret single-leader election, 2019. URL: `https://ethresear.ch/t/low-overhead-secret-single-leader-election/5994`.

24 Luciano Freitas, Andrei Tonkikh, Adda-Akram Bendoukha, Sara Tucci-Piergiovanni, Renaud Sirdey, Oana Stan, and Petr Kuznetsov. Homomorphic sortition – single secret leader election for pos blockchains. Cryptology ePrint Archive, Paper 2023/113, 2023. URL: `https://eprint.iacr.org/2023/113`.

25 Chaya Ganesh, Claudio Orlandi, and Daniel Tschudi. Proof-of-stake protocols for privacy-aware blockchains. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 690–719, Darmstadt, Germany, May 19–23 2019. Springer, Heidelberg, Germany. `doi:10.1007/978-3-030-17653-2_23`.

26 Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, pages 197–206, 2008. `doi:10.1145/1374376.1374407`.

27 Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. Cryptology ePrint Archive, Report 2017/454, 2017. URL: `https://eprint.iacr.org/2017/454`.

28 Alexander Golovnev, Jonathan Lee, Srinath Setty, Justin Thaler, , and Riad S. Wahby. Brakedown: Linear-time and post-quantum snarks for R1CS. Cryptology ePrint Archive, Paper 2021/1043, 2021. URL: `https://eprint.iacr.org/2021/1043`.

29 Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

30 George Kadianakis. Whisk: A practical shuffle-based ssle protocol for ethereum, 2022. X.

31 Thomas Kerber, Aggelos Kiayias, Markulf Kohlweiss, and Vassilis Zikas. Ouroboros crypsinous: Privacy-preserving proof-of-stake. In *2019 IEEE Symposium on Security and Privacy*, pages 157–174, San Francisco, CA, USA, May 19–23 2019. IEEE Computer Society Press. doi:10.1109/SP.2019.00063.

32 Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In *Advances in Cryptology – CRYPTO 2022*, pages 71–101, 2022.

33 Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23, French Riviera, May 30 – June 3 2010. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-13190-5_1.

34 Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. URL: http://bitcoin.org/bitcoin.pdf.

35 Ngoc Khanh Nguyen and Gregor Seiler. Practical sublinear proofs for r1cs from lattices. In *Advances in Cryptology – CRYPTO 2022*, pages 133–162, 2022.

36 Chris Peikert. A decade of lattice cryptography. *Foundations and Trends in Theoretical Computer Science*, 10(4):283–424, 2016. Available online at eprint/2015/939.

37 Mayank Raikwar and Danilo Gligoroski. Sok: Decentralized randomness beacon protocols. In *Australasian Conference on Information Security and Privacy*, pages 420–446. Springer, 2022. available here.

38 Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93, Baltimore, MA, USA, May 22–24 2005. ACM Press. doi:10.1145/1060590.1060603.

39 Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):34:1–34:40, 2009. Available online here.

40 Antonio Sanso. Towards practical post quantum single secret leader election (ssle) - part 1, 2022. X.

41 Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th FOCS*, pages 124–134, Santa Fe, NM, USA, November 20–22 1994. IEEE Computer Society Press. doi:10.1109/SFCS.1994.365700.

42 Damien Stehlé, Ron Steinfeld, Keisuke Tanaka, and Keita Xagawa. Efficient public key encryption based on ideal lattices. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 617–635, Tokyo, Japan, December 6–10 2009. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-10366-7_36.