Round-optimal Verifiable Oblivious Pseudorandom Functions from Ideal Lattices

Martin R. Albrecht[1], Alex Davidson[2], Amit Deo[1], and Nigel P. Smart[3,4]

[1] Information Security Group, Royal Holloway, University of London
[2] Cloudflare Portugal
[3] COSIC-imec, KU Leuven, Belgium
[4] Dept. Computer Science, University of Bristol, UK
martin.albrecht@royalholloway.ac.uk, adavidson@cloudflare.com,
amit.deo.2015@rhul.ac.uk, nigel.smart@kuleuven.be

Abstract. Verifiable Oblivious Pseudorandom Functions (VOPRFs) are protocols that allow a client to learn verifiable pseudorandom function (PRF) evaluations on inputs of their choice. The PRF evaluations are computed by a server using their own secret key. The security of the protocol prevents both the server from learning anything about the client's input, and likewise the client from learning anything about the server's key. VOPRFs have many applications including password-based authentication, secret-sharing, anonymous authentication and efficient private set intersection. In this work, we construct the first round-optimal (online) VOPRF protocol that retains security from well-known lattice hardness assumptions. Our protocol requires constructions of non-interactive zero-knowledge arguments of knowledge (NIZKAoK). For analogues of Stern-type proofs in the lattice setting, we show that our VOPRF may be securely instantiated in the quantum random oracle model. We construct such arguments as extensions of prior work in the area of lattice-based zero-knowledge proof systems.

1 Introduction

A verifiable oblivious pseudorandom function (VOPRF) is an interactive protocol between two parties; a client and a server. Intuitively, this protocol allows a server to provide a client with an evaluation of a pseudorandom function (PRF) on an input x chosen by the client using the server's key k. Informally, the security of a VOPRF, from the server's perspective, guarantees that the client learns nothing more than the PRF evaluated at x using k as the key. Security from the perspective of the client guarantees the two conditions below:

1. the server learns nothing about the input x;

2. the client's output in the protocol is indeed the evaluation on input x and key k.

The second property makes the protocol a *verifiable* oblivious PRF. If we were to remove this second requirement, the protocol would be an *oblivious* pseudorandom function (OPRF). From a multi-party computation perspective, an OPRF can be seen as a protocol that securely achieves the functionality $g(x, k) = (F_k(x), \perp)$ where F is a PRF and $\perp$ indicates that the server receives no output. (V)OPRFs have numerous applications including secure keyword search [25], private set intersection [34], secure data de-duplication [35], password-protected secret sharing [31,32], password-authenticated key exchange (PAKE) [33] and privacy-preserving lightweight authentication mechanisms [19].

Many applications of (V)OPRFs have had recent and considerable real-world impact. The work of Jarecki at al. [33] constructs a PAKE protocol, known as OPAQUE, that can be integrated with TLS 1.3; the work of Davidson et al. [19] constructs an authentication mechanism (known as Privacy Pass) for anonymously bypassing Internet reverse Turing tests. The Privacy Pass protocol is currently used at scale by the web performance company Cloudflare [53]. Both schemes use discrete-log (DL) based (V)OPRF constructions that produce notably performant protocols. In addition, there is an ongoing (V)OPRF standardisation effort being carried forward by the Crypto Forum Research Group (CFRG) at the Internet Engineering Task Force (IETF) [20]. The aim of this effort is to crystallise the design of performant DL-based OPRF constructions for usage as primitives in wider protocols. The OPAQUE protocol is also up for standardisation as a candidate in the CFRG PAKE selection process [36].

Unfortunately, and in spite of the practical value of VOPRFs, all of the available constructions in the literature to date are based on classical assumptions such as decisional Diffie-Hellman (DDH) and RSA. This means that all current VOPRFs would be insecure when confronted with an adversary that can run quantum computations. Therefore, the design of a post-quantum secure (V)OPRF is required to ensure that the applications above remain secure in these future adversarial conditions. In fact, for full post-quantum security, both the PRF and the VOPRF protocol itself must be secure in the quantum adversarial model. While PRF constructions with claimed post-quantum security do exist, it remains an open problem to translate these into secure (V)OPRF protocols.

Constructions of PRFs arising from lattice-based cryptography have been known since the original work of Banerjee, Peikert and Rosen [4]. These constructions are post-quantum secure assuming the hardness of the learning with errors (LWE) problem against quantum adversaries. To get around the fact that the LWE problem involves the addition of random small errors, carefully chosen rounding is used to obtain *deterministic* outputs for PRFs based on the LWE assumption [4,10,3]. These earlier works on LWE-based PRFs were followed by constructions of more advanced variants of PRFs [16,14,48]. Despite this, there

is yet to be an OPRF protocol for any LWE-based PRF. The same is true for variants of these constructions that are based on the ring LWE (RLWE) problem [3].

Contributions. In this work we instantiate a round-optimal[5] VOPRF whose security relies on hardness assumptions over lattices. Our basic VOPRF design and proof assumes certain non-interactive zero knowledge arguments of knowledge (NIZKAoKs). With the goal of creating an example instantiation of the required NIZKAoKs, we adapt the usage of Stern's protocol to argue knowledge of the input and (small) key to a PRF evaluation from the Banerjee and Peikert design [3] (henceforth BP14) in the ring setting. These same techniques may also be used to prove in zero knowledge that a batch of RLWE samples all share the same secret. Note that using a post-quantum secure commitment scheme within Stern's protocol implies that the zero-knowledge arguments are also post-quantum. To obtain non-interactive arguments, we rely on the validity of the Fiat-Shamir transform in the quantum random oracle model QROM [22,41].

We stress that our results show the *feasibility* of round-optimal VOPRF protocols based on lattice assumptions rather than practicality. Our construction is unlikely to represent a practically performant VOPRF due to the required size of parameters. These parameter sizes are necessary for instantiating our construction whilst ensuring that the underlying lattice assumptions are reasonable – a consequence of using the BP14 PRF construction with the proof technique we employ. In addition, we require heavy zero-knowledge proof computations for ensuring that neither participant deviates from the protocol. As future work, there may be potential for adaptations of more efficient zero knowledge techniques [11,54] to replace our Stern proofs. One major difficulty in doing this is the fact that these more efficient techniques assume a particular form of q that appears incompatible with our VOPRF security proof [42,23,12].

Technical Overview. We design a VOPRF for a particular instantiation of the BP14 PRF in the ring setting. Specifically, for a particular *function* $\boldsymbol{a}^F : \{0,1\}^L \to R_q^{1\times\ell}$ where $R_q := \mathbb{Z}_q[X]/\langle X^n+1\rangle$, we want to design a VOPRF for the PRF

$$F_k(x) = \left\lfloor \frac{p}{q} \cdot \boldsymbol{a}^F(x) \cdot k \right\rceil$$

where the key $k \in R_q$ has small coefficients when represented in $\{-q/2, \ldots, q/2\}$. As mentioned above, the security of this construction can be reduced to the hardness of RLWE. Consider the PRF for 2-bit inputs: then $\boldsymbol{a}^F(x) = \boldsymbol{a}_1 \cdot G^{-1}(\boldsymbol{a}_2)$ where $\boldsymbol{a}_1, \boldsymbol{a}_2 \in R_q^{1\times\ell}$ are uniform, $G = (1, 2, \ldots, 2^{\ell-1})$ and $G^{-1}(\boldsymbol{a}_2) \in R_2^{\ell\times\ell}$ is binary. Informally, for small $\boldsymbol{e}, \boldsymbol{e}'' \in R_q^{1\times\ell}$, uniform $\boldsymbol{e}' \in R_q^{1\times\ell}/(R_q \cdot G)$ and q

[5] Meaning that only two messages are sent in the online (query) phase.

much larger than p, we can write

$$\begin{aligned}
\left\lfloor \frac{p}{q} \cdot \boldsymbol{a}^F(x) \cdot k \right\rceil &= \left\lfloor \frac{p}{q} k \cdot \boldsymbol{a}_1 \cdot G^{-1}(\boldsymbol{a}_2) \right\rceil = \left\lfloor \frac{p}{q} (k \cdot \boldsymbol{a}_1 + \boldsymbol{e}) \cdot G^{-1}(\boldsymbol{a}_2) \right\rceil \\
&\approx_c \left\lfloor \frac{p}{q} (\boldsymbol{u}) \cdot G^{-1}(\boldsymbol{a}_2) \right\rceil \text{ (RLWE)} \\
&= \left\lfloor \frac{p}{q} (u' G + \boldsymbol{e}') \cdot G^{-1}(\boldsymbol{a}_2) \right\rceil = \left\lfloor \frac{p}{q} (u' \boldsymbol{a}_2 + \boldsymbol{e}'') + \frac{p}{q} \boldsymbol{e}' \cdot G^{-1}(\boldsymbol{a}_2) \right\rceil \\
&\approx_c \left\lfloor \frac{p}{q} \boldsymbol{u}'' + \frac{p}{q} \boldsymbol{e}' \cdot G^{-1}(\boldsymbol{a}_2) \right\rceil \text{ (RLWE)} \\
&= \left\lfloor \frac{p}{q} \widetilde{\boldsymbol{u}} \right\rceil
\end{aligned}$$

where $\boldsymbol{u}, \boldsymbol{u}'', \widetilde{\boldsymbol{u}}$ are uniform in $R_q^{1 \times \ell}$ and u' is uniform in R_q.

To provide intuition for our VOPRF design, we describe a basic protocol below that serves as a *starting point*. We assume that zero knowledge proofs of each message are implicitly provided to ensure that the protocol is followed.

1. The server publishes some commitment to a small key $k \in R_q$.
2. On input x, the client picks *invertible* $s \in R_q$, small $\boldsymbol{e} \in R_q^{1 \times \ell}$ and sends $\boldsymbol{c}_x = \boldsymbol{a}^F(x) \cdot s + \boldsymbol{e}$.
3. On input small $k \in R_q$, the server sends $\boldsymbol{d}_x = \boldsymbol{c}_x \cdot k + \boldsymbol{e}$ for small $\boldsymbol{e}' \in R_q^{1 \times \ell}$.
4. The client outputs $\boldsymbol{y} = \left\lfloor \frac{p}{q} \cdot \boldsymbol{d}_x \cdot s^{-1} \right\rceil$.

For server security, note that $\boldsymbol{d}_x = \boldsymbol{a}^F(x) \cdot s \cdot k + \boldsymbol{e} \cdot k + \boldsymbol{e}'$. Suppose that we choose $\boldsymbol{e}'$ from a distribution that hides addition of terms $\boldsymbol{e} \cdot k$ and $\boldsymbol{e}_s \cdot s$ (where $\boldsymbol{e}_s$ is identically distributed to $\boldsymbol{e}$). Then, from the perspective of the client, the server might as well have sent $\boldsymbol{d}_x = (\boldsymbol{a}^F(x) \cdot k + \boldsymbol{e}_s) \cdot s + \boldsymbol{e}'$. Picking $\boldsymbol{e}_s$ (and $\boldsymbol{e}$) from an appropriate distribution [3] makes the term in brackets i.e. $\boldsymbol{a}^F(x) \cdot k + \boldsymbol{e}_s$ computationally indistinguishable from uniform random under a RLWE assumption. This implies that the message $\boldsymbol{d}_x$ leaks nothing about the server's key k.

For client security in the first message, we pick s from a valid RLWE secret distribution and $\boldsymbol{e}$ from the same distribution as that of $\boldsymbol{e}_s$. Similarly to the above, this implies that $\boldsymbol{c}_x = \boldsymbol{a}^F(x) \cdot s + \boldsymbol{e}$ is indistinguishable from uniform. Finally, we must show that the client does indeed recover $F_k(x)$ as its output $\boldsymbol{y}$. For correctness, we *would like to say* that

$$\left\lfloor \frac{p}{q} \cdot \boldsymbol{d}_x \cdot s^{-1} \right\rceil = \left\lfloor \frac{p}{q} \cdot \boldsymbol{a}^F(x) \cdot k + \frac{p}{q} (\boldsymbol{e} \cdot k \cdot s^{-1} + \boldsymbol{e}' \cdot s^{-1}) \right\rceil = \left\lfloor \frac{p}{q} \cdot \boldsymbol{a}^F(x) \cdot k \right\rceil.$$

Thus, we guarantee correctness if all coefficients of $\frac{p}{q} \cdot \boldsymbol{a}^F(x) \cdot k$ are at least $\left| \frac{p}{q} \left(\boldsymbol{e} \cdot k \cdot s^{-1} + \boldsymbol{e}' \cdot s^{-1} \right) \right|_\infty$ away from $\mathbb{Z} + \frac{1}{2}$. It turns out that if all coefficients

of s^{-1} are small, then this condition is satisfied with extremely high probability due to the 1-dimensional short integer solution (1D-SIS) assumption [15]. The form of $\boldsymbol{a}^F(x)$ is crucial to the connection with the 1D-SIS problem. In particular, we rely on the fact that we can decompose $\boldsymbol{a}^F(x)$ as $\boldsymbol{a}'_1 \cdot \boldsymbol{a}'_2$ where $\boldsymbol{a}'_1 \in R_q^{1\times\ell}$ is uniform random and $\boldsymbol{a}'_2 \in R_q^{\ell\times\ell}$ has entries that are polynomials with *binary* coefficients.

Unfortunately, this simplified protocol cannot quite be realised using standard RLWE secret distributions. The problem is that (to our knowledge) there is no standard RLWE secret distribution where samples from the distribution are guaranteed to have *small inverses* in R_q. To overcome this issue, we apply a technique for sampling "full" NTRU keys [30,50]. Firstly, we sample small ring elements s and t from a Gaussian distribution. Secondly, we use the extended GCD algorithm – in combination with Babai's rounding algorithm – to recover small u and v, such that $u \cdot s + v \cdot t = 1 \bmod R_q$. To adapt the basic protocol to our actual protocol, the client sends $\boldsymbol{c}_x^1 = \boldsymbol{a}^F(x) \cdot s + \boldsymbol{e}_1$, $\boldsymbol{c}_x^2 = \boldsymbol{a}^F(x) \cdot t + \boldsymbol{e}_2$ and receives back

$$\boldsymbol{d}_x^1 = \boldsymbol{c}_x^1 \cdot k + \boldsymbol{e}'_1, \quad \boldsymbol{d}_x^2 = \boldsymbol{c}_x^2 \cdot k + \boldsymbol{e}'_2.$$

The final output is then $\left\lfloor \frac{p}{q} \left(u \cdot \boldsymbol{d}_x^1 + v \cdot \boldsymbol{d}_x^2\right) \right\rceil$. In addition, the real protocol incorporates zero knowledge arguments of knowledge (that we show are instantiable based on adaptations of Stern's protocol) to prevent malicious parties from deviating from the protocol description.

Ultimately, the security of our VOPRF construction (using the Stern-style proofs) holds in the QROM and relies on the hardness of RLWE and 1D-SIS which are both at least as hard as certain lattice problems using appropriate parameters. We discuss asymptotic parameter settings for which our protocol relies directly on assumed hard lattice problems in Section 5.

Road map. We begin with preliminaries in Section 2. We draw attention to Definition 1 which deviates from the usual MPC definition. In particular, we argue security against malicious clients when k is sampled from a key distribution for which the PRF is pseudorandom, rather than arguing security for arbitrary fixed k. Next is the VOPRF construction (Section 3) followed by a high-level description of the zero knowledge proof instantiations (Section 4). Finally we give the security proof for our VOPRF protocol in Section 5.

Appendices. Our appendices consist of a more detailed account of our computational hardness assumptions (Appendix A) followed by a collection of miscellaneous results (Appendix B) and more details of our zero knowledge instantiations (Appendix C).

2 Preliminaries

All algorithms will be considered to be randomised algorithms unless explicitly stated otherwise. A PPT algorithm is a randomised (i.e. probabilistic) algorithm with polynomial running time in the security parameter κ. We consider the probability distribution of outputs of algorithms as being over all possible choices of the internal coins of the algorithm. For a distribution $\mathcal{D}$, we denote the sampling of x according to distribution $\mathcal{D}$ by $x \leftarrow \mathcal{D}$. We write $x \leftarrow S$ for a finite set S to indicate sampling uniformly at random from S. We use the notation $\mathcal{D}_1 \approx_c \mathcal{D}_2$ to mean the distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ are computationally indistinguishable and $\approx_s$ to denote statistical indistinguishability. We use the standard asymptotic notations. We let $\mathsf{negl}(\kappa)$ denote a negligible function (i.e. a function that is $\kappa^{-\omega(1)}$) and write $r_1 \gg r_2$ as short-hand for $r_1 \geq \kappa^{\omega(1)} \cdot r_2$. We say a distribution $\mathcal{D}$ is (B, δ)-bounded if $\Pr[\|x\| \geq B \mid x \leftarrow \mathcal{D}] < \delta$. If a distribution is (B, δ)-bounded for a negligible δ, then we say that distribution is simply B-bounded.

In this work we will use power of two cyclotomic rings. In particular, for some integer q, we will be considering polynomials in the power-of-two cyclotomic ring $R = \mathbb{Z}[X]/\langle X^n + 1\rangle$ and $R_q := R/qR$ where n is a power-of-two. $R_{\leq c}$ is the set of elements of R where all coefficients have an absolute value at most c. We also use a rounding operation from $\mathbb{Z}_q$ to $\mathbb{Z}_{q'}$ where $q' < q$. For $x \in \mathbb{Z}_q$, this rounding operation is defined as

$$\lceil x \rfloor_{q'} := \lceil (q'/q) \cdot x \rfloor$$

where $\lceil \cdot \rfloor$ denotes rounding to the nearest integer (rounding down in the case of a tie). If q' divides q, we can lift rounded integers back up to $\mathbb{Z}_q$ by simply multiplying by q/q'. Note that lifting the result of a rounding takes an $x \in \mathbb{Z}_q$ to the nearest multiple of q/q'. Therefore, the difference between x and the result of this rounding then lifting is at most $q/(2 \cdot q')$. Polynomials and vectors are rounded component-wise. We write $\|\cdot\|$ for the Euclidean norm and $\|\cdot\|_\infty$ for the infinity norm. We define the norms of ring elements by considering the norms of their *coefficient* vectors. Vectors whose entries are ring elements will be denoted using bold characters and integer vectors will be indicated by an over-arrow e.g. $\boldsymbol{v}$ has ring entries and $\vec{w}$ has integer entries. Suppose $\boldsymbol{v} = (v_1, \ldots, v_n)$. A norm of $\boldsymbol{v}$ is the norm of the vector obtained by concatenating the coefficient vectors of $v_1, \ldots, v_n$.

Gaussian distributions. For any $\sigma > 0$, define the *Gaussian function* on $\mathbb{R}^n$ centred at $\boldsymbol{c} \in \mathbb{R}^n$ with parameter σ to be:

$$\forall\, \boldsymbol{x} \in \mathbb{R}^n,\ \rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x}) = e^{-\pi \cdot \|\boldsymbol{x}-\boldsymbol{c}\|^2/\sigma^2}.$$

Define $\rho_\sigma(\mathbb{Z}) := \sum_{i \in \mathbb{Z}} \rho_\sigma(i)$. The *discrete Gaussian distribution over* $\mathbb{Z}$, denoted χ_σ assigns probability $\rho_\sigma(i)/\rho_\sigma(\mathbb{Z})$ to each $i \in \mathbb{Z}$ and probability 0 to each non-integer point. The *discrete Gaussian distribution over* R, denoted as $R(\chi_\sigma)$, is the distribution over R where each coefficient is distributed according to χ_σ.

Using the results of [26,13], χ_σ can be sampled in polynomial time. Moreover the Euclidean norm of a sample from $R(\chi_\sigma)$ can be bounded using an instantiation of Lemma 1.5 of [2]. We state this lemma next.

Lemma 1. *Let $\sigma > 0$ and $n = \mathsf{poly}(\kappa)$. Then*

$$\Pr\left[\|x\| \geq \sigma\sqrt{n} \;\middle|\; x \leftarrow R(\chi_\sigma)\right] < \mathsf{negl}(\kappa)\,.$$

In addition, following the same reasoning as in [21] we have the following "drowning/smudging" lemma.

Lemma 2. *Let $\sigma > 0$ and $y \in \mathbb{Z}$. The statistical distance between χ_σ and $\chi_\sigma + y$ is at most $|y|/\sigma$.*

2.1 Verifiable Oblivious Pseudorandom Functions

Recall that the main goal of our work is to build a verifiable oblivious pseudorandom function (VOPRF). A VOPRF is a protocol between two parties: a server $\mathbb{S}$ and a client $\mathbb{C}$, securely realising the ideal functionality in Figure 1. The functionality consists of two phases, the initialisation phase and the query phase. In the event that the functionality $\mathcal{F}_{\mathsf{VOPRF}}$ receives an input k from party $\mathbb{S}$ (i.e. the server) during the initialisation phase, it stores the key for use during the query phase. This models a server in a real protocol committing to a PRF key k. Next comes the query phase, where the client $\mathbb{C}$ sends some value x to $\mathcal{F}_{\mathsf{VOPRF}}$. Once this value x has been received, the server $\mathbb{S}$ either sends the functionality an instruction to abort or to deliver the value $y = F_k(x)$ to $\mathbb{C}$. Finally, the functionality carries out this instruction. Importantly, (assuming that no abort is triggered) the client has the guarantee that its output is indeed $F_k(x)$ i.e. the output of the client is *verifiably* correct when interacting with $\mathcal{F}_{\mathsf{VOPRF}}$.

This is a two party functionality between a server $\mathbb{S}$ and a client $\mathbb{C}$. We assume there is a fixed PRF function defined by $F_k(x)$.

Init: On input of init from both parties the functionality waits for an input k from party $\mathbb{S}$. If $\mathbb{S}$ returns abort then the functionality aborts. Otherwise the functionality stores the value k.

Query: On input of (query, x) from the client $\mathbb{C}$, if $x \neq \perp$ then functionality waits for an input from party $\mathbb{S}$. If $\mathbb{S}$ returns $\mathsf{deliver}$ then the functionality sends $y = F_k(x)$ to party $\mathbb{C}$. If $\mathbb{S}$ returns abort then the functionality aborts.

Figure 1. The Ideal Functionality $\mathcal{F}_{\mathsf{VOPRF}}$

We now describe the distributions that arise in the security requirement. We consider malicious adversaries throughout that behave arbitrarily. We begin with the distributions of interest when a server has been corrupted. First, we consider a "real" world protocol Π between $\mathbb{C}(x)$ and $\mathbb{S}(k)$ along with an adversary $\mathcal{A}$. We denote $\mathsf{real}_{\Pi,\mathcal{A},\mathbb{S}}(x,k,1^\kappa)$ to be the joint output distribution of $\mathcal{A}(k)$ when corrupting $\mathbb{S}(k)$ and $\mathbb{C}(x)$ where $\mathbb{C}(x)$ behaves as specified by Π. In this setting, $\mathcal{A}$ interacts directly with $\mathbb{C}$. Now we introduce a simulator denoted Sim that lives in the "ideal" world. Specifically, still assuming $\mathcal{A}$ corrupts a server, Sim interacts with $\mathcal{A}$ on one hand and with $\mathbb{C}(x)$ via $\mathcal{F}_{\mathsf{VOPRF}}$ on the other hand. Considering this setting, for any client/server input pair (x,k), we define $\mathsf{ideal}_{\mathcal{F}_{\mathsf{VOPRF}},\mathsf{Sim},\mathcal{A},\mathbb{S}}(x,k,1^\kappa)$ to be the joint output distribution of $\mathcal{A}(k)$ and the honest client $\mathbb{C}(x)$ when $\mathcal{A}(k)$ interacts via Sim. Informally, one may interpret Sim as an attacker-in-the-middle between $\mathcal{A}$ and the outside world that interacts with $\mathcal{F}_{\mathsf{VOPRF}}$ external to the view of $\mathcal{A}$. Security will argue that whatever $\mathcal{A}$ can learn/affect in the real protocol can be emulated via Sim in the ideal world setting.

Next, we describe the distributions of interest when a client has been corrupted by an adversary $\mathcal{A}$. We let $\mathcal{K}$ denote the key distribution under which PRF security of F holds. First, consider a "real" world case where $\mathcal{A}$ corrupts $\mathbb{C}(x)$ and directly interacts with honest $\mathbb{S}(k)$ which follows the specification of protocol Π. In this case, we use $\mathsf{real}_{\Pi,\mathcal{A},\mathbb{C}}(x,\mathcal{K},1^\kappa)$ to denote the joint output distribution of $\mathcal{A}(x)$ and $\mathbb{S}(k)$ where $k \leftarrow \mathcal{K}$. Now consider an alternative "ideal" world case where we introduce a simulator Sim interacting with $\mathcal{A}$ on one hand and with $\mathbb{S}(x)$ via $\mathcal{F}_{\mathsf{VOPRF}}$ on the other hand. Once again, one may wish to interpret the simulator as an attacker-in-the-middle interacting with $\mathcal{F}_{\mathsf{VOPRF}}$ external to the view of $\mathcal{A}$. In this alternative case, we denote the joint output distribution of $\mathcal{A}(x)$ and $\mathbb{S}(k)$ where $\mathcal{A}$ interacts via Sim and $k \leftarrow \mathcal{K}$ as $\mathsf{ideal}_{\mathcal{F}_{\mathsf{VOPRF}},\mathsf{Sim},\mathcal{A},\mathbb{C}}(x,\mathcal{K},1^\kappa)$.

Finally, for protocol Π, let $\mathsf{output}(\Pi,x,k)$ denote the output distribution of a *client* with input x running protocol Π with a server whose input key is k. Using the notation established above, we are ready to present our definition of a VOPRF.

Definition 1. *A protocol Π is a verifiable oblivious pseudorandom function if all of the following hold:*

1. ***Correctness:*** *For every pair of inputs (x,k),*

$$\Pr[\mathsf{output}(\Pi,x,k) \neq F_k(x)] \leq \mathsf{negl}(\kappa)\,.$$

2. ***Malicious server security:*** *For any PPT adversary $\mathcal{A}$ corrupting a server, there exists a PPT simulator Sim such that for every pair of inputs (x,k):*

$$\mathsf{ideal}_{\mathcal{F}_{\mathsf{VOPRF}},\mathsf{Sim},\mathcal{A},\mathbb{S}}(x,k,1^\kappa) \approx_c \mathsf{real}_{\Pi,\mathcal{A},\mathbb{S}}(x,k,1^\kappa).$$

3. ***Average case malicious client security:*** *For any PPT adversary $\mathcal{A}$ corrupting a client, there exists a PPT simulator* Sim *such that for all client inputs x:*
   - $\mathsf{ideal}_{\mathcal{F}_{\mathsf{VOPRF}},\mathsf{Sim},\mathcal{A},\mathbb{C}}(x,\mathcal{K},1^\kappa) \approx_c \mathsf{real}_{\Pi,\mathcal{A},\mathbb{C}}(x,\mathcal{K},1^\kappa)$.
   - *If $\mathcal{A}$ correctly outputs $F_k(x)$ with all but negligible probability over the choice $k \leftarrow \mathcal{K}$ when interacting directly with $\mathbb{S}(k)$ using protocol Π, then $\mathcal{A}$ also outputs $F_k(x)$ with all but negligible probability when interacting via* Sim.

We now discuss this definition. Note that the correctness and malicious server security requirements are the standard ones used in MPC. Therefore, we restrict this discussion to the condition that we call average case malicious client security. The motivation for this non-standard property is that an honest server will always sample a key from distribution $\mathcal{K}$ as it wishes to provide pseudorandom function evaluations. In particular, PRF security holds with respect to this key distribution $\mathcal{K}$. Therefore, it makes sense to ask what a malicious client may learn/affect only in the case where $k \leftarrow \mathcal{K}$ which leads to the first point of our average case malicious client security requirement. The second point of the requirement captures the fact that adversaries may have access to an oracle that checks whether the PRF was evaluated correctly or not. Suppose that we give the adversary $\mathcal{A}$ access to an oracle which can check an input/output pair to the PRF is valid or not. Then $\mathcal{A}$ should not be able to distinguish whether it is interacting with a real server $\mathbb{S}$ or a simulation Sim. Note that our proof structure relies heavily on our alternative malicious client security definition. In particular, the definition above allows us to argue over the entropy of secret keys when making indistinguishability claims.

2.2 Computational assumptions

Here we present the presumed quantum hard computational problems that will be used in our security proofs. Evidence that these problems are indeed quantum hard follows via reductions from standard lattice problems (see Appendix A). These reductions from lattice problems will be used when setting parameters for our VOPRF. The first is the standard decisional RLWE problem [45].

Definition 2. (RLWE problem) *Let $q, m, n, \sigma > 0$ depend on κ (q, m, n are integers). The* decision-RLWE problem ($\mathsf{dRLWE}_{q,n,m,\sigma}$) *is to distinguish between:*

$$(a_i,\ a_i \cdot s + e_i)_{i\in[m]} \in (R_q)^2 \quad \text{and} \quad (a_i, u_i)_{i\in[m]} \in (R_q)^2$$

for $a_i, u_i \leftarrow R_q$; $s, e_i \leftarrow R(\chi_\sigma)$.

We sometimes write $\mathsf{dRLWE}_{q,n,\sigma}$, leaving the parameter m (representing the number of samples) implicit.

The second problem is slightly less standard. It is the short integer solution problem in *dimension 1* (1D-SIS). The following formulation of the problem was used in [15] in conjunction with a lemma attesting to its hardness. See Appendix A for more details.

Definition 3. (1D-SIS, [15, Definition 3.4]) *Let q, m, t depend on κ. The one-dimensional SIS problem, denoted $\mathsf{1D\text{-}SIS}_{q,m,t}$, is the following: Given a uniform $\boldsymbol{v} \leftarrow \mathbb{Z}_q^m$, find $\boldsymbol{z} \in \mathbb{Z}_{p'}^m$ such that $||\boldsymbol{z}||_\infty \leq t$ and $\langle \boldsymbol{v}, \boldsymbol{z} \rangle \in [-t, t] + q\mathbb{Z}$.*

2.3 Non-interactive zero-knowledge arguments of knowledge

The foundations of zero-knowledge (ZK) proof systems were established in a number of works [24,29,28,9]. At a high level, a ZK proof system for language $\mathcal{L}$ allows a prover $\mathbb{P}$ to convince a verifier $\mathbb{V}$ that some instance x is in $\mathcal{L}$, without revealing anything beyond this statement. Further, a ZK argument of knowledge (ZKAoK) system allows $\mathbb{P}$ to convince $\mathbb{V}$ that they hold a witness w attesting to the fact that x is in $\mathcal{L}$ (where the $\mathcal{L}$ is defined by a relation predicate $\mathsf{R}_\mathcal{L}(x, w)$).

Definition 4. (NIZKAoK) *Let $\mathbb{P}$ be a* prover*, let $\mathbb{V}$ be a* verifier*, let $\mathcal{L}$ be a language with accompanying relation predicate $\mathsf{R}_\mathcal{L}(\cdot, \cdot)$. Let $\mathcal{W}_\mathcal{L}(x)$ be a generic set of witnesses attesting to the fact that $x \in \mathcal{L}$, i.e. $\forall x \in \mathcal{L}$, and $w \in \mathcal{W}_\mathcal{L}(x)$ we have $\mathsf{R}_\mathcal{L}(x, w) = 1$. Let $\mathsf{nizk} = (\mathsf{Setup}, \mathbb{P}, \mathbb{V})$ be a tuple of algorithms defined as follows:*

- $\mathsf{crs} \leftarrow \mathsf{nizk.Setup}(1^\kappa)$*: outputs a common random string* crs
- $\pi \leftarrow \mathsf{nizk}.\mathbb{P}(\mathsf{crs}, x, w)$*: on input* crs*, a word $x \in \mathcal{L}$ and a witness $w \in \mathcal{W}_\mathcal{L}(x)$; outputs a proof $\pi \in \{0,1\}^{\mathsf{poly}(\kappa)}$.*
- $b \leftarrow \mathsf{nizk}.\mathbb{V}(\mathsf{crs}, x, \pi)$*: on input* crs*, a word $x \in \mathcal{L}$ and a proof $\pi \in \{0,1\}^{\mathsf{poly}(\kappa)}$; outputs $b \in \{0,1\}$.*

Definition 5. (NIZKAoK Security) *We say that* nizk *is a* non-interactive zero-knowledge argument of knowledge *(NIZKAoK) for $\mathcal{L}$ if the following holds.*

1. (Completeness)*: Consider $x \in \mathcal{L}$ and $w \in \mathcal{W}_\mathcal{L}(x)$, where $\mathsf{R}_\mathcal{L}(x, w) = 1$. Then:*

$$\Pr\left[1 \leftarrow \mathsf{nizk}.\mathbb{V}(\mathsf{crs}, x, \pi) \middle| \begin{smallmatrix} \mathsf{crs} \leftarrow \mathsf{nizk.Setup}(1^\kappa) \\ \pi \leftarrow \mathsf{nizk}.\mathbb{P}(\mathsf{crs}, x, w) \end{smallmatrix}\right] \geq 1 - \mathsf{negl}(\kappa)\,.$$

2. (Computational knowledge extraction)*: The proof system satisfies computational knowledge extraction with knowledge error $\bar{\kappa}$ if, for any PPT prover*

$\mathbb{P}^*$ *with auxiliary information* aux*, the following holds. There exists a PPT algorithm* $\mathsf{nizk.Extract}$ *and a polynomial* p *such that, for any input* x*, then:*

$$\Pr[1 \leftarrow \mathsf{P}_{\mathcal{L}}(x, w') | w' \leftarrow \mathsf{nizk.Extract}(\mathbb{P}^*(\mathsf{crs}, x, \mathsf{aux}))] \geq \frac{\nu - \bar{\kappa}}{p(|x|)}$$

is satisfied, where ν *is the probability that* $\mathsf{nizk}.\mathbb{V}(\mathsf{crs}, x, \mathbb{P}^*(\mathsf{crs}, x, \mathsf{aux}))$ *outputs 1.*

3. (Computational zero-knowledge)*: There exists a simulated setup algorithm* $\mathsf{nizk.SimSetup}(1^\kappa)$ *outputting* $\mathsf{crs}_{\mathsf{Sim}}$ *and a trapdoor* $\mathcal{T}$ *along with a PPT algorithm* $\mathsf{nizk.Sim}(\mathsf{crs}_{\mathsf{Sim}}, \mathcal{T}, x)$ *satisfying*

$$\left\{ \begin{matrix} \mathsf{crs} \leftarrow \mathsf{nizk.Setup}(1^\kappa) \\ \pi \leftarrow \mathsf{nizk}.\mathbb{P}(\mathsf{crs}, x, w) \end{matrix} \right\} \approx_c \left\{ \pi_{\mathsf{Sim}} \leftarrow \mathsf{nizk.Sim}(\mathsf{crs}_{\mathsf{Sim}}, \mathcal{T}, x) \middle| (\mathsf{crs}_{\mathsf{Sim}}, \mathcal{T}) \leftarrow \mathsf{nizk.SimSetup}(1^\kappa) \right\}$$

$\forall x \in \mathcal{L}$ *and* $w \in \mathcal{W}_{\mathcal{L}}(x)$.

Interactive proof systems. An interactive proof system is one where the proving algorithm ($\mathbb{P}$) requires interaction with the verifier. Such an interaction could be an arbitrary protocol, with many message exchanges, but a typical (in the honest verifier case) scenario is a three-move protocol consisting of a commitment (from the prover), a uniformly chosen challenge (from the verifier) and then a response (from the prover).

Fiat and Shamir [24] established a mechanism of switching a (constant-round) honest verifier zero-knowledge interactive proof of knowledge into a *non-interactive* zero-knowledge proof of knowledge in the random oracle model (ROM). In particular, the random challenge provided by the verifier is replaced with the output of a random oracle evaluation taking as input the statement x and the provers initial commitment. It was recently shown that the standard Fiat-Shamir transform is also secure in the *quantum* ROM (QROM) [22,41].

2.4 Lattice PRF

We will use an instantiation of the lattice PRF from [3]. Below, we present relevant definitions/results, all of which are particular cases of definitions/results from [3]. We set $\ell = \lceil \log_2 q \rceil$ throughout. The construction from [3] makes use of *gadget matrices* that can be found in many previous works [46,3,15,27].

Gadgets G, G^{-1}. Define $G : R_q^{\ell \times \ell} \rightarrow R_q^{1 \times \ell}$ to be the linear operation corresponding to left multiplication by $(1, 2, \ldots, 2^{\ell-1})$. Further, define $G^{-1} : R_q^{1 \times \ell} \rightarrow R_q^{\ell \times \ell}$ to be the bit decomposition operation that essentially inverts G i.e. the i^{th} column of $G^{-1}(\boldsymbol{a})$ is the bit decomposition of $a_i \in R_q$ into binary polynomials.

The PRF from [3] is defined as $F_k(x) = \lfloor \boldsymbol{a}_x \cdot k \rceil_p$ for $\boldsymbol{a}_x \in R_q^{1 \times \ell}$ as defined below.

Definition 6. *Fix some* $\boldsymbol{a}_0, \boldsymbol{a}_1 \leftarrow R_q^{1\times\ell}$. *For any* $x = (x_1, \ldots, x_L) \in \{0,1\}^L$. *We define* $\boldsymbol{a}_x \in R_q^{1\times\ell}$ *as*

$$\boldsymbol{a}_x := \boldsymbol{a}_{x_1} \cdot G^{-1}\left(\boldsymbol{a}_{x_2} \cdot G^{-1}\left(\boldsymbol{a}_{x_3} \cdot G^{-1}\left(\ldots\left(\boldsymbol{a}_{x_{L-1}} \cdot G^{-1}\left(\boldsymbol{a}_{x_L}\right)\right)\right)\right)\right) \in R_q^{1\times\ell}.$$

The pseudorandomness of this construction follows from the ring learning with errors assumption.

Theorem 1 ([3]). *Sample* $k \leftarrow R(\chi_\sigma)$. *If* $q \gg p \cdot \sigma \cdot \sqrt{L} \cdot n \cdot \ell$, *then the function* $F_k(x) = \lfloor \boldsymbol{a}_x \cdot k \rceil_p$ *is a PRF under the* $\mathsf{dRLWE}_{q,n,\sigma}$ *assumption.*

When we eventually prove security of our VOPRF, it will be useful to define a special error distribution such that $\boldsymbol{a}_x \cdot k + \boldsymbol{e}$ remains indistinguishable from uniform (under RLWE) when $\boldsymbol{e}$ is sampled from this special error distribution. To this end, we introduce the distributions $\mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$ followed by a lemma that is implicit in the pseudorandomness of the PRF from [3].

Definition 7. *For* $\boldsymbol{a}_0, \boldsymbol{a}_1 \in R_q^{1\times\ell}$, *define*

$$\boldsymbol{a}_{x\setminus i} := G^{-1}\left(\boldsymbol{a}_{x_{i+1}} \cdot G^{-1}\left(\boldsymbol{a}_{x_{i+2}} \cdot G^{-1}\left(\cdots\left(\boldsymbol{a}_{x_{L-1}} \cdot G^{-1}\left(\boldsymbol{a}_{x_L}\right)\right)\cdots\right)\right)\right) \in R_q^{\ell\times\ell}.$$

Furthermore, let $\mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$ *be the distribution that is sampled by choosing* $\boldsymbol{e}_i \leftarrow R(\chi_\sigma)^{1\times\ell}$ *for* $i = 1, \ldots, L$ *and outputting*

$$\boldsymbol{e} = \sum_{i=1}^{L-1} \boldsymbol{e}_i \cdot a_{x\setminus i} + \boldsymbol{e}_L.$$

Lemma 3 (Implicit in [3]). *If* $\boldsymbol{a}_0, \boldsymbol{a}_1 \leftarrow R_q^{1\times\ell}, \boldsymbol{e} \leftarrow \mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$ *and* $s \leftarrow R(\chi_\sigma)$, *then for any fixed* $x \in \{0,1\}^L$,

$$(\boldsymbol{a}_0,\ \boldsymbol{a}_1,\ \boldsymbol{a}_x \cdot s + \boldsymbol{e})$$

is indistinguishable from uniform random by the $\mathsf{dRLWE}_{q,n,\sigma}$ *assumption.*

In addition to introducing $\mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$, it will be useful to write down an upper bound on the infinity norm on errors drawn from this distribution. The following lemma follows from the fact that for $y \leftarrow \chi_\sigma$, $\|y\|_\infty \leq \sigma\sqrt{n}$ with all but negligible probability by Lemma 1. In fact, we could use the result that $\|y\|_\infty \leq \sigma n^{c'}$ with probability at least $1 - c \cdot \exp(-\pi n^{2c'})$ for any constant $c' > 0$ and some universal constant c to reduce the upper bound, but we choose not to for simplicity.

Lemma 4 (Bound on errors). *Let* $x \in \{0,1\}^L$, $\ell = \lceil \log_2 q \rceil$ *and* $n = \mathsf{poly}(\kappa)$. *Samples from* $\mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$ *have infinity norm at most* $L \cdot \ell \cdot \sigma \cdot n^{3/2}$ *with all but negligible probability.*

3 A VOPRF Construction From Lattices

In this section, we provide a construction emulating the DH blinding construction $g^a = ((g^r)^a)^{1/r}$. In what follows, we will initially ignore the zero-knowledge proofs establishing that all computations are performed honestly. A detailed description of the protocol is in Figure 2 but the main high-level idea follows.

Recall that we are working with power-of-two cyclotomic rings. Informally, suppose a *client* wants to obtain $a \cdot k + e \in R_q$ (where e is relatively small) from a server holding a *short* k without revealing $a \in R_q$. One way to achieve this is for the client to sample $s, t, e_0, e_1 \leftarrow R(\chi_\sigma)$. The client then also samples *short* u, v such that $u \cdot s + v \cdot t = 1 \in R_q$ (we discuss how below). The client submits $a \cdot s + e_0$ and $a \cdot t + e_1$ and obtains $(a \cdot s + e_0) \cdot k + e'_0$ and $(a \cdot t + e_1) \cdot k + e'_1$ from the server where e_1, e'_1 are small. Finally the client can compute:

$$\begin{aligned} r &= u \cdot \big((a \cdot s + e_0) \cdot k + e'_0\big) + v \cdot \big((a \cdot t + e_1) \cdot k + e'_1\big) \\ &= a \cdot (u \cdot s + v \cdot t) \cdot k + u \cdot e_0 \cdot k + u \cdot e'_0 + v \cdot e_1 \cdot k + v \cdot e'_1 \\ &= a \cdot k + u \cdot e_0 \cdot k + u \cdot e'_0 + v \cdot e_1 \cdot k + v \cdot e'_1 \\ &\approx a \cdot k. \end{aligned}$$

To compute tuples s, t, u, v such that $u \cdot s + v \cdot t = 1$ and all elements are short, we may use known techniques for sampling "full" NTRU private keys [30,50] on input of $(s, t) \in R^2$. From now on we use: $\mathrm{res}(\cdot, \cdot)$ to refer to the computation of the resultant of two polynomials; $\mathrm{xgcd}(\cdot, \cdot)$ to refer to the computation of the extended GCD of two integers; and $s^\star$ to refer to the conjugate of s in R. In particular, $\mathsf{fullNTRU}(s, t)$ runs the following steps.

1. Compute $r_s = \mathrm{res}(s, X^n + 1) \in \mathbb{Z}$ and $u' \in R$ s.t. $u' \cdot s = r_s$.
2. Compute $r_t = \mathrm{res}(t, X^n + 1) \in \mathbb{Z}$ and $v' \in R$ s.t. $v' \cdot t = r_t$.
3. Compute $r, u'', v'' = \mathrm{xgcd}(r_s, r_t)$. If $r \neq 1$: abort
4. Set $u = u'' \cdot u' \in R$ and $v = v'' \cdot v' \in R$.
5. Run Babai's inverting and rounding algorithm [1]:
   (a) Compute
   $$r = \left\lfloor \frac{v \cdot s^\star - u \cdot t^\star}{s \cdot s^\star + t \cdot t^\star} \right\rceil.$$
   (b) Update $(u,\ v) = (u + r \cdot t,\ v - r \cdot s) \in R^2$.
6. Return u, v.

Note that it might be significantly more efficient to implement rational arithmetic using floating point arithmetic as in [50] which might entail repeatedly computing r. Finally, using the same heuristic arguments as in [30, Appendix A], we may expect the norm of u, v to satisfy $\|(u, v)\| \approx \sqrt{n/12} \cdot \|(s, t)\|$. However, for the

CRS SetUp: To set up the CRS execute the following steps:

- Pick $\boldsymbol{a}_0, \boldsymbol{a}_1 \leftarrow R_q^{1\times\ell}$
- crs_0 contains $a \in R_q$
- crs_1 and crs_2 are for proof systems $\mathbb{P}_1$ and $\mathbb{P}_2$ respectively

Init: The initialization procedure is executed by the server $\mathbb{S}$ and the client $\mathbb{C}$ both with initial input crs_0.

1. The server $\mathbb{S}$ executes the following steps
   - $k, e \leftarrow R(\chi_\sigma)$.
   - $c \leftarrow a \cdot k + e \bmod q$.
   - $\pi_0 \leftarrow \mathbb{P}_0(k, e : \mathsf{crs}_0)$.

   and sends (c, π_0) to the client $\mathbb{C}$.
2. On receipt of (c, π_0) the client executes
   - $b \leftarrow \mathbb{V}_0(\mathsf{crs}_0, c, \pi_0)$.
   - Output abort if $b = 0$, otherwise store c.

Query: This is a two round protocol between the client and the server, with the client going first.

1. On input of $(x \in \{0,1\}^L, \mathsf{crs}_1, \mathsf{crs}_2)$ the client $\mathbb{C}$ executes the following steps
   - $s, t \leftarrow R(\chi_\sigma)$.
   - If $\mathsf{fullNTRU}(s, t)$ aborts: go back to previous step
     else: $(u, v) \leftarrow \mathsf{fullNTRU}(s, t)$.
   - $\boldsymbol{a}_x = \boldsymbol{a}_{x_1} \cdot G^{-1}\left(\cdots\left(\boldsymbol{a}_{x_{L-1}} \cdot G^{-1}\left(\boldsymbol{a}_{x_L}\right)\right)\cdots\right) \bmod q$.
   - $\boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow \mathcal{E}_{\boldsymbol{a}_0, \boldsymbol{a}_1, x, \sigma}$.
   - $\boldsymbol{c}_x^1 \leftarrow \boldsymbol{a}_x \cdot s + \boldsymbol{e}_1 \bmod q$.
   - $\boldsymbol{c}_x^2 \leftarrow \boldsymbol{a}_x \cdot t + \boldsymbol{e}_2 \bmod q$.
   - $\pi_1 \leftarrow \mathbb{P}_1(x, s, t, \boldsymbol{e}_1, \boldsymbol{e}_2 : \mathsf{crs}_1, \boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \boldsymbol{a}_0, \boldsymbol{a}_1)$.

   and sends $(\boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \pi_1)$ to the server $\mathbb{S}$.
2. On receipt of $(\boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \pi_1)$ the server $\mathbb{S}$ executes the following steps
   - $b \leftarrow \mathbb{V}_1(\mathsf{crs}_1, \boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \boldsymbol{a}_0, \boldsymbol{a}_1, \pi_1)$.
   - Output abort if $b = 0$
   - $\boldsymbol{e}_1', \boldsymbol{e}_2' \leftarrow R(\chi_{\sigma'})^{1\times\ell}$.
   - $\boldsymbol{d}_x^1 = \boldsymbol{c}_x^1 \cdot k + \boldsymbol{e}_1' \bmod q$.
   - $\boldsymbol{d}_x^2 = \boldsymbol{c}_x^2 \cdot k + \boldsymbol{e}_2' \bmod q$.
   - $\pi_2 \leftarrow \mathbb{P}_2(k, \boldsymbol{e}_1', \boldsymbol{e}_2', e : \mathsf{crs}_0, \mathsf{crs}_2, c, \boldsymbol{d}_x^1, \boldsymbol{d}_x^2, \boldsymbol{c}_x^1, \boldsymbol{c}_x^2)$.

   and sends $(\boldsymbol{d}_x^1, \boldsymbol{d}_x^2, \pi_2)$ to the client $\mathbb{C}$.
3. On receipt of $(\boldsymbol{d}_x^1, \boldsymbol{d}_x^2, \pi_2)$ the client $\mathbb{C}$ executes
   - $b \leftarrow \mathbb{V}_2(\mathsf{crs}_0, \mathsf{crs}_2, c, \boldsymbol{d}_x^1, \boldsymbol{d}_x^2, \boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \pi_2)$.
   - Output abort if $b = 0$.
   - $\boldsymbol{y}_x = \lfloor u \cdot \boldsymbol{d}_x^1 + v \cdot \boldsymbol{d}_x^2 \rceil_p$.
   - Output $\boldsymbol{y}_x$.

Figure 2. VOPRF construction

purposes of our security proofs, we use the upper bound $||(u,v)||_\infty \leq n\sigma$ (see Appendix B.2 for details).

Suppose we sample $(s,t) \leftarrow R(\chi_\sigma)^2$. Then there is a chance that s and t are not co-prime, causing the above algorithm to abort. However, it is shown in Lemma 4.4 in the full version[6] of [51] that discrete Gaussian s and t will be co-prime with non-negligible probability as long as $\sigma \geq 7 \cdot n^{3/2} \cdot \ln^{3/2}(n)$. In addition, an algorithm solving RLWE with *two* discrete Gaussian coprime secrets (s,t) with non-negligible advantage would also solve RLWE where the two secrets are sampled independently from Gaussian distributions, with non-negligible probability. Therefore, the sampling algorithm above results in a secret distribution for which RLWE is believed to be hard if $(s,t) \leftarrow R(\chi_\sigma)^2$.

As mentioned above, a more detailed formulation of our construction is given in Figure 2. In this description, $\mathbb{P}_i$ and $\mathbb{V}_i$ prover and verifier algorithms for three different zero-knowledge proof systems indexed by $i \in \{0,1,2\}$.

3.1 Zero Knowledge Argument of Knowledge Statements

The arguments of $\mathbb{P}_i$ algorithms fall into two groups separated by a colon. Arguments before a colon are intended as "secret" information pertaining to a witness for a statement. Arguments after a colon should be interpreted as "public" information describing the statement that is being proved.

Client Proof. The client proof denoted $\mathbb{P}_1(x, s, t, \boldsymbol{e}_1, \boldsymbol{e}_2 : \mathsf{crs}_1, \boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \boldsymbol{a}_0, \boldsymbol{a}_1)$ should prove knowledge of

- $x \in \{0,1\}^L$
- $s, t \in R$ where $\|s\|_\infty, \|t\|_\infty \leq \sigma \cdot \sqrt{n}$
- $\boldsymbol{e}_1, \boldsymbol{e}_2 \in R^{1\times\ell}$ where $\|\boldsymbol{e}_1\|_\infty, \|\boldsymbol{e}_2\|_\infty \leq L \cdot \ell \cdot \sigma \cdot n^{3/2}$

such that

$$\boldsymbol{c}_x^1 = \boldsymbol{a}_x \cdot s + \boldsymbol{e}_1 \bmod q,$$
$$\boldsymbol{c}_x^2 = \boldsymbol{a}_x \cdot t + \boldsymbol{e}_2 \bmod q.$$

Server Proofs. The server proof in the *initialisation phase* denoted $\mathbb{P}_0(k, e : \mathsf{crs}_0)$ has the purpose of proving knowledge of $k, e \in R$ where $\|k\|_\infty$, $\|e\|_\infty \leq \sigma \cdot \sqrt{n}$ such that

$$c = a \cdot k + e \bmod q,$$

[6] available at http://perso.ens-lyon.fr/damien.stehle/NTRU.html

where crs_0 contains (a, b).

The server proof in the *query phase* denoted by

$$\mathbb{P}_2(k, \boldsymbol{e}'_1, \boldsymbol{e}'_2, e : \mathsf{crs}_0, \mathsf{crs}_2, c, \boldsymbol{d}^1_x, \boldsymbol{d}^2_x, \boldsymbol{c}^1_x, \boldsymbol{c}^2_x)$$

has the purpose of proving that there is some

- $k, e \in R$ where $\|k\|_\infty, \|e\|_\infty \leq \sigma \cdot \sqrt{n}$
- $\boldsymbol{e}'_1, \boldsymbol{e}'_2 \in R^{1\times\ell}$ where $\|\boldsymbol{e}'_1\|_\infty, \|\boldsymbol{e}'_2\|_\infty \leq \sigma' \cdot \sqrt{n}$

such that

$$\begin{aligned} c &= a \cdot k + e \bmod q, \\ \boldsymbol{d}^1_x &= \boldsymbol{c}^1_x \cdot k + \boldsymbol{e}'_1 \bmod q, \\ \boldsymbol{d}^2_x &= \boldsymbol{c}^2_x \cdot k + \boldsymbol{e}'_2 \bmod q. \end{aligned} \tag{1}$$

It is important to note that both $\boldsymbol{d}^1_x$ and $\boldsymbol{d}^2_x$ each consist of ℓ ring elements. Therefore, the above system consists of a total of $1 + 2\ell$ noisy products of public ring elements and k. Note that the well-definedness of normal form RLWE (where the secret is drawn from the error distribution) implies that the witnesses used by the prover in π_0 and π_2 share the same value k.

The security proof of our VOPRF construction can be found in Section 5, as in the next section we turn to discussing possible instantiations of the required zero-knowledge proofs.

3.2 Correctness

Before proving correctness, we present a lemma that we will apply below. The proof of this lemma is in Appendix B.1.

Lemma 5. *Fix any $x \in \{0,1\}^L$. Suppose there exists a PPT algorithm $\mathcal{D}(x, \boldsymbol{a}_0, \boldsymbol{a}_1)$ that outputs $r \in R$ such that $\|r\| \leq B$ and at least one coefficient of $\boldsymbol{a}_x \cdot r$ is in the set $(q/p) \cdot \mathbb{Z} + [-T, T]$ with non-negligible probability (over a uniform choice of $\boldsymbol{a}_0, \boldsymbol{a}_1 \leftarrow R^\ell_q$ and its random coins). Then there exists an efficient algorithm solving $\mathsf{1D\text{-}SIS}_{q/p, n\ell, \max\{n\ell B, T\}}$.*

Lemma 6 (Correctness). *Adopt the notation of Figure 2, assuming an honest client and server. Define $T := \sigma n^2(L\ell\sigma^2 n^{5/2} + \sigma')$. For any $x \in \{0,1\}^L, k \in R_q$ such that $\|k\|_\infty \leq \sigma \cdot \sqrt{n}$, we have that*

$$\Pr[\boldsymbol{y}_x \neq F_k(x)] \leq \mathsf{negl}(\kappa)$$

over the choice of PRF parameters $\boldsymbol{a}_0, \boldsymbol{a}_1 \leftarrow R^{1\times\ell}_q$ assuming the hardness of $\mathsf{1D\text{-}SIS}_{q/p, n\ell, T}$.

Proof. Fix an arbitrary x. Assume that there exists a k such that $\|k\| \leq \sigma \cdot \sqrt{n}$ and $\Pr[\boldsymbol{y}_x \neq F_k(x)]$ is non-negligible over the choice of $\boldsymbol{a}_0, \boldsymbol{a}_1 \leftarrow R_q^{1\times\ell}$. Expanding $\boldsymbol{d}_x^1$ and $\boldsymbol{d}_x^2$, we have that

$$\boldsymbol{y}_x = \lfloor \boldsymbol{a}_x \cdot k + u \cdot (\boldsymbol{e}_1 \cdot k + \boldsymbol{e}_1') + v \cdot (\boldsymbol{e}_2 \cdot k + \boldsymbol{e}_2') \rceil_p .$$

Note that $\boldsymbol{e} := u \cdot (\boldsymbol{e}_1 \cdot k + \boldsymbol{e}_1') + v \cdot (\boldsymbol{e}_2 \cdot k + \boldsymbol{e}_2')$ has infinity norm less than T with all but negligible probability. Therefore, it must be that at least one coefficient of $\boldsymbol{a}_x \cdot k$ is within T is in the set $(q/p) \cdot \mathbb{Z} + [T, T]$ with non-negligible probability, otherwise $\boldsymbol{y}_x = \lfloor \boldsymbol{a}_x \cdot k \rceil_p$. Applying Lemma 5 to the algorithm $\mathcal{D}(x)$ that ignores $\boldsymbol{a}_0, \boldsymbol{a}_1$ and simply outputs k implies an efficient algorithm solving $\mathsf{1D\text{-}SIS}_{q/p, n\ell, \max\{n^{3/2}\ell\sigma, T\}}$. □

4 Lattice-based NIZKAoK Instantiations

We now describe instantiations of our zero knowledge proofs of knowledge. At a high level, we may use Stern-based proofs for all proof systems (although there may be other alternatives). In particular, we use the Fiat-Shamir transform on parallel repetitions of Stern-based proofs as in [39]. We recall that the Fiat-Shamir transform has recently been shown to be secure in the QROM [22,41]. We place most of our attention on discussing how to instantiate Proof System 1, as the other proof systems may be derived straight-forwardly using a subset of the techniques arising in Proof System 1. For more precise details on how to instantiate Proof System 1 using Stern's protocol, see Appendix C.

Proof System 0: Small secret RLWE sample

Let $A \in \mathbb{Z}_q^{n\times n}$ be the negacyclic matrices associated to multiplication by $a \in R_q$ respectively. Further, let $\vec{c} \in \mathbb{Z}_q^n$ be the coefficient vectors of $c \in R_q$ respectively. The first proof aims to prove in zero knowledge, knowledge of a short solution $\vec{x} := (\vec{x}_1, \vec{x}_2)$, where $\|\vec{x}\|_\infty \leq \sigma \cdot \sqrt{n}$ to the system

$$\vec{c} = A \cdot \vec{x}_1 + \vec{x}_2.$$

The security of our VOPRF uses a very special form of q for security due to the use of the 1D-SIS assumption (see Appendix A). In particular, q is neither an integer permitting an NTT, nor a prime power. This is unfortunate because the state-of-the-art for proving zero knowledge of short solutions to linear equations use the fact that $x(x-1) = 0 \bmod q$ if and only if $x \in \{0, 1\}$ to prove that witness vectors have binary entries [54] (or utilise NTTs and similar algebraic relations for ternary entries [11]). Since our composite q is not amenable to these techniques, we can either use Stern's protocol as described in [40], or rejection sampling techniques [43,44] to perform this zero knowledge proof. However, due

to the soundness gap suffered when using rejection sampling (i.e. the fact that the infinity norm of the extractable witness may be a small constant factor times larger than intended), one can imagine the use of the less efficient Stern's protocol for the sake of keeping our VOPRF security proofs conceptually simpler. In addition, attempting to use the protocol of Beullens [7] for our choice of q leads to superpolynomially sized proofs.

Proof System 1: Non-interactive proofs of PRF evaluations

At a high level, we will run Stern's protocol [52] $\mathcal{O}(\kappa)$ times in parallel and apply the Fiat-Shamir heuristic in the QROM. We do not actually present Stern's protocol itself in this work, but we do highlight the sufficient requirements that are required for the use of an abstraction of Stern's protocol. This abstraction is both presented and proven to be a ZKAoK in [37] with respect to a computationally binding commitment scheme. If a statistically binding commitment scheme is used, the Stern protocol is a ZKPoK. For simplicity, we use the abstraction of Stern's protocol in a $\mathcal{F}_{\text{Com}}$-hybrid model where the functionality of a perfectly binding commitment scheme is provided, rather than using any post-quantum perfectly binding commitment scheme explicitly. In this model, security holds in the QROM. Note that perfectly binding lattice-based commitment schemes do exist [6,5]. For some set VALID and a matrix M representing a set of linear equations over the *integers* modulo a natural number, the abstraction of Stern's protocol allows a prover to argue knowledge of a solution $\vec{w} \in \mathsf{VALID}$ to a system $M \cdot \vec{w} = \vec{y}$ in zero knowledge. In order to apply Stern's protocol, there must be a set of permutations $\Gamma = \{\Gamma_\phi : \phi \in \mathcal{S}\}$ acting on the *entries* of $\vec{w}$ such that both of the following key properties hold.

Key properties:

1. For every $\phi \in \mathcal{S}, \vec{w} \in \mathsf{VALID} \iff \Gamma_\phi(\vec{w}) \in \mathsf{VALID}$.
2. For every $\vec{w} \in \mathsf{VALID}$, the distribution of $\Gamma_\phi(\vec{w})$ (for $\phi \leftarrow \mathcal{S}$) is uniform over the set VALID.

Therefore, in order to apply the abstract Stern's protocol, we must rewrite our problem as a linear system of equations and describe a set VALID alongside a set of permutations Γ possessing the key properties above. The details of how this is done are presented in Appendix C, but we now give a short high-level summary of the technique.

First note that we can compute $\boldsymbol{a}_x$ recursively by setting variables $B_i \in R_q^{\ell \times \ell}$ for $i = L-1, \ldots, 0$ via $B_{L-1} = G^{-1}(\boldsymbol{a}_{x_{L-1}})$, and $B_i = G^{-1}(\boldsymbol{a}_{x_i} \cdot B_{i+1})$ for $i = L-2, \ldots, 0$. Using this, we have $\boldsymbol{a}_x = G \cdot B_0$. We can therefore use the system $G \cdot B_i = \boldsymbol{a}_{x_i} \cdot B_{i-1}$ to facilitate computation of $\boldsymbol{a}_x$ along with the equation

$\boldsymbol{y}_x = G \cdot B_0 \cdot k + \boldsymbol{e}$ (where $\boldsymbol{e}$ represents a rounding error) to fully describe a PRF evaluation. However, the resulting system is over ring elements and is not linear in unknowns. To solve these issues, we simply replace ring multiplication by integer matrix-vector products and then linearise the resulting system using known techniques [38,39]. At this point, we carefully describe the set VALID, noting the structure that linearisation/ring structure introduces. We also make use of bit-decompositions to bound the infinity norms of valid solutions. From this, we use known techniques [38,39] (extended to the ring setting) to describe Γ satisfying the key properties above.

Proof System 2: Non-interactive proofs of secret equivalence

Recall that we wish to prove existence of a solution to Equations (1). Note that $\boldsymbol{d}_x^1, \boldsymbol{d}_x^2$ from the protocol in Section 3 are vectors holding ℓ ring elements. Therefore, Equations (1) can be expressed as a system

$$c_i = a_i k + e_i, \quad i = 1, \ldots, 1 + 2\ell$$

where $\|e_1\|_\infty, \|k\|_\infty \leq \sigma \cdot \sqrt{n}$, $\|e_2\|_\infty, \ldots, \|e_{1+2\ell}\|_\infty \leq \sigma' \cdot \sqrt{n}$. In order to instantiate this proof system, we may use the abstract Stern protocol again. Note that in Appendix 4, we implicitly show how to prove knowledge of RLWE secrets. Therefore, using the same techniques, we can straight-forwardly obtain abstract Stern proofs for Proof System 2.

5 Security Proof

In this section, we show that the protocol in Figure 2 is a VOPRF achieving security against malicious adversaries. In particular, corrupted clients and servers that attempt to subvert the protocol learn/affect only as much as in an ideal world, where they interact via the functionality $\mathcal{F}_{\mathsf{VOPRF}}$.

Theorem 2. (Security) *Assume $p|q$. The protocol in Figure 2 is a secure VOPRF protocol (according to Definition 1) if the following conditions hold:*

- $\mathsf{dRLWE}_{q,n,\sigma}$ *is hard,*
- $\frac{q}{2p} \gg \sigma' \gg L \cdot \ell \cdot \sigma^2 \cdot n^3$,
- $\mathsf{1D\text{-}SIS}_{q/(2p),n\cdot\ell,4\cdot\sigma'\cdot\sigma\cdot n^{5/2}}$ *is hard.*

Note that correctness of our protocol with respect to honest clients and servers is shown in Section 3.2. Therefore, what remains is to show average malicious client security and malicious server security.

Correctness of non-aborting malicious protocol runs. During the malicious client proof, it will be useful to call upon the fact that any non-aborting protocol transcript allows the computation of $F_k(x)$ (with all but negligible probability).

Lemma 7. *Assume that* $\mathsf{dRLWE}_{q,n,\sigma}$ *is hard,* σ *and* n *are* $\mathsf{poly}(\kappa)$*, and* $\frac{q}{2p} \gg \sigma' \gg L \cdot \ell \cdot \sigma^2 n^3$*. For any* $x \in \{0,1\}^L$*, consider a non-aborting run of the protocol in Figure 2 between a (potentially malicious) efficient client* $\mathbb{C}^*$ *and honest server* $\mathbb{S}$*. Consider any* $u, v \in R_q$*, such that* $\|u\|_\infty, \|v\|_\infty \leq \sigma \cdot n$ *and* $u \cdot s + v \cdot t = 1$*, where* s, t *are extracted from* $\mathbb{C}^*$*'s proof in its message to* $\mathbb{S}$*. Then, the value of* $\left\lfloor u \cdot \boldsymbol{d}_x^1 + v \cdot \boldsymbol{d}_x^2 \right\rceil_p$ *is equal to* $\lfloor \boldsymbol{a}_x \cdot k \rceil_p$ *with all but negligible probability.*

Proof. We use the notation from Figure 2. First note that for a *non-aborting* protocol run, any *efficient* client $\mathbb{C}^*$ must have produced $\boldsymbol{c}_x^1$ and $\boldsymbol{c}_x^2$ correctly using some $x \in \{0,1\}^L, s, t, \boldsymbol{e}_1, \boldsymbol{e}_2$ where $\|s\|_\infty, \|t\|_\infty \leq \sigma \cdot \sqrt{n}$ and $\|\boldsymbol{e}_1\|_\infty, \|\boldsymbol{e}_2\|_\infty \leq L \cdot \ell \cdot \sigma \cdot n^{3/2}$. To complete the proof, we will use the fact that $\frac{p}{q}(\boldsymbol{a}_x \cdot k + \boldsymbol{e})$ is computationally indistinguishable from uniform random over $\frac{p}{q} R_q^{1 \times \ell}$ when $\boldsymbol{e} \leftarrow \mathcal{E}_{a_0, a_1, x, \sigma}$ assuming the hardness of $\mathsf{dRLWE}_{q,n,\sigma}$ (Lemma 3). This implies that every coefficient in $\frac{p}{q}(\boldsymbol{a}_x k + \boldsymbol{e})$ is at least T' away from $\mathbb{Z} + 1/2$ with all but negligible probability for any $T' \ll 1$. We will use this fact twice to complete the proof. With this in mind, a client computing the output as prescribed in Figure 2 obtains

$$\left\lfloor \frac{p}{q}(u \cdot \boldsymbol{d}_x^1 + v \cdot \boldsymbol{d}_x^2) \right\rceil = \left\lfloor \frac{p}{q} \boldsymbol{a}_x \cdot k + \frac{p}{q} u \cdot (\boldsymbol{e}_1 \cdot k + \boldsymbol{e}_1') + \frac{p}{q} v \cdot (\boldsymbol{e}_2 \cdot k + \boldsymbol{e}_2') \right\rceil. \tag{2}$$

The quantity $\left\lfloor \frac{p}{q}(\boldsymbol{a}_x \cdot k + \boldsymbol{e}) \right\rceil$ can be shown to be equal to Equation (2) (with all but negligible probability) using the negligible value of

$$T_0' = \frac{3p}{q} \sigma n^2 (L \cdot \ell \cdot \sigma^2 \cdot n^{5/2} + \sigma') \geq \left\| \frac{p}{q} u(\boldsymbol{e}_1 \cdot k + \boldsymbol{e}_1') + \frac{p}{q} v(\boldsymbol{e}_2 \cdot k + \boldsymbol{e}_2') - \frac{p}{q} \boldsymbol{e} \right\|_\infty.$$

Furthermore, $\left\lfloor \frac{p}{q}(\boldsymbol{a}_x \cdot k + \boldsymbol{e}) \right\rceil$ is equal to $\left\lfloor \frac{p}{q} \boldsymbol{a}_x \cdot k \right\rceil$ with all but negligible probability, using the negligible value of

$$T_1' = \frac{p}{q} \cdot L \cdot \ell \cdot \sigma \cdot n^{3/2} \geq \left\| \frac{p}{q} \boldsymbol{e} \right\|_\infty.$$

. □

5.1 Malicious Client Proof

Lemma 8 (Average-case malicious client security). *Assume that σ and n are $\mathsf{poly}(\kappa)$, and $p|q$, and let conditions (i) and (ii) be as follows:*

(i) $\mathsf{dRLWE}_{q,n,\sigma}$ *is hard,*
(ii) $\frac{q}{2p} \gg \sigma' \gg L \cdot \ell \cdot \sigma^2 \cdot n^3$.

If the above conditions hold, then the protocol in Figure 2 has average-case security against malicious clients according to Definition 1.

Proof. We describe a simulation $\mathcal{S}$ that communicates with the functionality $\mathcal{F}_{\mathsf{VOPRF}}$ (environment) on one hand, and the malicious client $\mathbb{C}^*$ on the other. $\mathcal{S}$ carries out the following steps:

1. During CRS.SetUp, publish honest $\boldsymbol{a}_0, \boldsymbol{a}_1, \mathsf{crs}_1$ and (dishonest) simulated versions of crs_0 and crs_2. Denote the simulated CRS elements by crs'_0 and crs'_2.
2. During the **Init** phase, send $\mathbb{C}^*$ a uniform $c \leftarrow R_q$ with a simulated proof $\pi_{0,\mathsf{Sim}}$ and pass the init message onto $\mathcal{F}_{\mathsf{VOPRF}}$. Initialise an empty list $\mathsf{received}$.
3. During the **Query** stage, for each message $(\boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \pi_1)$ from $\mathbb{C}^*$, do the following:
   (a) $b \leftarrow \mathbb{V}_1(\mathsf{crs}_1, \boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \boldsymbol{a}_0, \boldsymbol{a}_1, \pi_1)$. If $b = 0$ send abort to the functionality and abort the protocol with the malicious client. If $b = 1$ continue to the next step.
   (b) Extract the values x, s, t from π_1 using the ZKAoK extractor and send (query, x) to the functionality.
   (c)
   - If $\mathcal{F}_{\mathsf{VOPRF}}$ aborts:
     $\mathcal{S}$ aborts.
   - If $\mathcal{F}_{\mathsf{VOPRF}}$ returns $\boldsymbol{y} \in R_p^{1\times\ell}$ **and** $\forall \boldsymbol{y}^*, (x, \boldsymbol{y}^*) \notin \mathsf{received}$:
     (i.e. if this is the first time x is queried) uniformly sample

$$\boldsymbol{y}_q \leftarrow R_q^{1\times\ell} \cap \left(\frac{q}{p}\boldsymbol{y} + R_{\leq \frac{q}{2p}}^{1\times\ell}\right)$$

     and do $\mathsf{received.add}(x, \boldsymbol{y}_q)$.
   - If $\mathcal{F}_{\mathsf{VOPRF}}$ returns $\boldsymbol{y} \in R_p^{\ell}$ **and** $\exists \boldsymbol{y}^*$s.t.$(x, \boldsymbol{y}^*) \in \mathsf{received}$:
     (i.e. x was previously queried) Then set $\boldsymbol{y}_q = \boldsymbol{y}^*$.

   (d) Next pick $\bar{\boldsymbol{e}}'_1, \bar{\boldsymbol{e}}'_2 \leftarrow \chi_{\sigma'}$ and set

$$\bar{\boldsymbol{d}}_x^1 = \boldsymbol{y}_q \cdot s + \bar{\boldsymbol{e}}'_1 \bmod q,$$
$$\bar{\boldsymbol{d}}_x^2 = \boldsymbol{y}_q \cdot t + \bar{\boldsymbol{e}}'_2 \bmod q.$$

   Finally, produce a simulated proof $\pi_{2,\mathsf{Sim}}$ using crs'_2 and send $(\bar{\boldsymbol{d}}_x^1, \bar{\boldsymbol{d}}_x^2, \pi_{2,\mathsf{Sim}})$ to $\mathbb{C}^*$.

We now argue that $\mathbb{C}^*$ cannot decide whether it is interacting with $\mathcal{S}$ or with a genuine server. Firstly, recognise that $\mathsf{crs}'_0, \mathsf{crs}'_2$ is indistinguishable from honestly created $\mathsf{crs}_0, \mathsf{crs}_2$. Secondly, the malicious client cannot distinguish the simulator's uniform c_1, c_2 that it sends during the **Init** phase from the real protocol by the $\mathsf{dRLWE}_{q,n,\sigma}$ assumption (condition (i)). This implies that both the SetUp and **Init** phases that $\mathcal{S}$ performs are indistinguishable from the real protocol.

The most challenging step is arguing that the simulator's behaviour in the **Query** phase is indistinguishable from the real protocol from the malicious client's point of view. We will analyse the behaviour of the simulator assuming that no abort is triggered. We begin by arguing that the server message in the real protocol with respect to any triple (x, s, t) can be replaced by a related message $(\boldsymbol{a}_x \cdot k + \boldsymbol{e}^x) \cdot s + \widehat{\boldsymbol{e}}'_1$ where $\boldsymbol{e}^x \leftarrow \mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$ and $\widehat{\boldsymbol{e}}'_1 \leftarrow R(\chi_{\sigma'})^{1\times\ell}$ (and similarly for the message depending on t) without detection by the following statistical argument. For brevity, consider the quantities that depend on s, i.e. $\boldsymbol{c}_x^1$ and $\boldsymbol{d}_x^1$ (a similar argument holds for the quantities depending on t). We have that the server response in the *real* protocol has $\boldsymbol{d}_x^1$ of the form

$$(\boldsymbol{a}_x \cdot s + \boldsymbol{e}_1) \cdot k + \boldsymbol{e}'_1 \tag{3}$$

where $\boldsymbol{e}_1 \leftarrow \mathcal{E}_{a_0,a_1,x,\sigma}$ and $\boldsymbol{e}'_1 \leftarrow R(\chi_{\sigma'})^{1\times\ell}$. By Lemma 2, the message distribution in Equation (3) is statistically indistinguishable (condition (ii)) from

$$\boldsymbol{a}_x \cdot k \cdot s + \boldsymbol{e}''_1 \tag{4}$$

where $\boldsymbol{e}''_1 \leftarrow R(\chi_{\sigma'})^{1\times\ell}$ due to the fact that $\sigma' \gg L \cdot \ell \cdot \sigma^2 \cdot n^3$. By a similar argument, the quantity given in Equation (4) is statistically close in distribution to

$$(\boldsymbol{a}_x \cdot k + \boldsymbol{e}^x) \cdot s + \boldsymbol{e}'''_1. \tag{5}$$

where $\boldsymbol{e}^x \leftarrow \mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$ and $\boldsymbol{e}'''_1 \leftarrow R(\chi_{\sigma'})^{1\times\ell}$.

Using Lemma 3 and condition (i), we have that the term in front of s in Equation (5) is indistinguishable from random by the hardness of $\mathsf{dRLWE}_{q,n,\sigma}$ (Lemma 3). In particular, from an efficient $\mathbb{C}^*$'s point of view, $\boldsymbol{d}_x^1$ cannot be distinguished from

$$\boldsymbol{u}_x \cdot s + \boldsymbol{e}_1$$

where $\boldsymbol{u}_x \leftarrow R_q^{1\times\ell}$ and $\boldsymbol{e}_1 \leftarrow R(\chi_{\sigma'})^{1\times\ell}$. Similarly, $\boldsymbol{d}_x^2$ cannot be distinguished from $\boldsymbol{u}_x \cdot t + \boldsymbol{e}_2$ for the same $\boldsymbol{u}_x$ as above and $\boldsymbol{e}_2 \leftarrow R(\chi_{\sigma'})^{1\times\ell}$. Note that on repeated queries, the errors sampled from $R(\chi_\sigma)^{1\times\ell}$ are fresh. The fact that $\mathcal{S}$ samples $\boldsymbol{y}_q$ as a uniformly chosen element of a uniformly chosen interval implies the indistinguishability part of average-case malicious client security.

Next, we show that if the malicious client can indeed compute the correct value from the messages it receives from the honest server (in the real protocol), then it can do the same with the messages that it receives from the simulator. In Lemma 7, we show that a malicious client which does not cause an abort can

compute $\lfloor \boldsymbol{a}_x \cdot k \rceil_p$ from the messages it receives from the honest server with all but negligible probability. We now show that this is also the case with the messages it receives from $\mathcal{S}$. Consider $\boldsymbol{y}_q$ sampled by $\mathcal{S}$ and also the corresponding values $\bar{\boldsymbol{d}}_x^1$ and $\bar{\boldsymbol{d}}_x^2$. In addition, define $\boldsymbol{e} := \boldsymbol{y}_q - (q/p) \cdot \boldsymbol{y} \in R^{1\times\ell}_{\leq \frac{q}{2p}}$ so that $\boldsymbol{e}$ follows the uniform distribution over $R^{1\times\ell}_{\leq \frac{q}{2p}}$. We have that

$$\left\lfloor \frac{p}{q}(u \cdot \bar{\boldsymbol{d}}_x^1 + v \cdot \bar{\boldsymbol{d}}_x^2) \right\rceil = \left\lfloor \boldsymbol{y} + \frac{p}{q}\left(\boldsymbol{e} + u \cdot \bar{\boldsymbol{e}}_1' + v \cdot \bar{\boldsymbol{e}}_2'\right) \right\rceil . \tag{6}$$

We also know that with all but negligible probability, $\|u\cdot\bar{\boldsymbol{e}}_1' + v\cdot\bar{\boldsymbol{e}}_2'\|_\infty \leq \sigma\cdot\sigma'\cdot n^{5/2}$ (since no abort occurred) and that $\|\boldsymbol{e}\|_\infty$ is less than $q/(2p) - T$ with all but negligible probability as long as $T \ll (q/2p)$. Taking $T = \sigma \cdot \sigma' \cdot n^{5/2}$, we get that with all but negligible probability,

$$\left\| \frac{p}{q} \cdot \left(\boldsymbol{e} + u \cdot \bar{\boldsymbol{e}}_1' + v \cdot \bar{\boldsymbol{e}}_2'\right) \right\|_\infty \leq \frac{1}{2},$$

implying that the quantity in Equation (6) rounds correctly to $\boldsymbol{y}$ with all but negligible probability. Therefore, both the real protocol and simulator enable correct evaluation of the PRF. □

5.2 Malicious Server Proof

Lemma 9. *Let conditions (i),(ii) and (iii) be as follows:*

(i) $\mathsf{dRLWE}_{q,n,\sigma}$ *is hard,*
(ii) $\sigma' \gg L \cdot \ell \cdot \sigma^2 \cdot n^{5/2}$,
(iii) $\mathsf{1D\text{-}SIS}_{q/(2p),n\cdot\ell,4\cdot\sigma'\cdot\sigma\cdot n^{5/2}}$ *is hard.*

If the above conditions hold, then the protocol in Figure 2 is secure in the presence of malicious servers.

Proof. We construct a simulator $\mathcal{S}$ interacting with the malicious server $\mathbb{S}^*$ on one hand and with the functionality $\mathcal{F}_{\mathsf{VOPRF}}$ on the other. The simulator $\mathcal{S}$ behaves as follows:

1. During the CRS.SetUp phase, publish honest $a_0, a_1, \mathsf{crs}_0, \mathsf{crs}_2$ and (dishonest) simulated crs_1' to use with the proof systems.
2. During the **Init** phase, if $\mathbb{S}^*$ sends $c \in R_q$ and an accepting proof π_0, then use the zero knowledge extractor to obtain a key k' from π_0 and forward this on to the functionality. If the message is not of the correct format, or the proof does not verify, then abort.

3. During the **Query** phase, select two uniform random values $\boldsymbol{u}_1, \boldsymbol{u}_2 \leftarrow R_q^{1\times\ell}$, and using the ZK simulator, produce a simulated proof $\pi_{1,\mathsf{Sim}}$ using crs_1'. Send the message $(\boldsymbol{u}_1, \boldsymbol{u}_2, \pi_{1,\mathsf{Sim}})$. Wait for a response of the form $(\widetilde{\boldsymbol{d}}_x^1, \widetilde{\boldsymbol{d}}_x^2, \widetilde{\pi}_2)$ from $\mathbb{S}^*$. If the proof $\widetilde{\pi}_2$ verifies[7], forward on $\mathsf{deliver}$ to $\mathcal{F}_{\mathsf{VOPRF}}$. Otherwise, forward abort to $\mathcal{F}_{\mathsf{VOPRF}}$.

We will show that the joint output of an honest client $\mathbb{C}$ and $\mathbb{S}^*$ in the real world (where they interact directly) and the ideal world (where they interact via $\mathcal{F}_{\mathsf{VOPRF}}$ and $\mathcal{S}$) are computationally indistinguishable. We begin by arguing that the malicious server $\mathbb{S}^*$ cannot distinguish whether it is interacting with a real client or $\mathcal{S}$, as described above. Firstly, replacing crs_1 by crs_1' is indistinguishable from the point of view of $\mathbb{S}^*$ by definition of a simulated CRS. Importantly, if $\mathbb{S}^*$ can produce valid proofs in the **Init** phase, the key k' obtained by the simulator is the *unique* ring element consistent with c by the uniqueness of normal form RLWE solutions.

All that is left to consider is the **Query** phase. Note that in the real protocol, the client produces two values $\boldsymbol{c}_x^1, \boldsymbol{c}_x^2$ that are pseudorandom under the hardness of $\mathsf{dRLWE}_{q,n,\sigma}$ by Lemma 3. Therefore, the malicious server $\mathbb{S}^*$ cannot distinguish a real $(\boldsymbol{c}_x^1, \boldsymbol{c}_x^2)$ from the pair $(\boldsymbol{u}_1, \boldsymbol{u}_2)$ that $\mathcal{S}$ uses. By the properties of a ZK simulator, it follows that a real client message $(\boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \pi_1)$ and crs_1 is indistinguishable from $(\boldsymbol{u}_1, \boldsymbol{u}_2, \pi_{1,\mathsf{Sim}})$ and crs_1'. Next, if the response from $\mathbb{S}^*$ has a valid proof, then $\mathcal{S}$ forwards on $\mathsf{deliver}$. This means that the ideal functionality passes a PRF evaluation to the client using the server key k'. We now argue that this emulates the output on the client side when running the real protocol with malicious server $\mathbb{S}^*$.

The case where the proof verification fails is trivial since the client aborts in the real and ideal worlds. As a result, we focus on the case where the zero knowledge proof produced by $\mathbb{S}^*$ verifies correctly. Let $\boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow \mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$ be sampled by the honest client. For this honest client interacting with malicious $\mathbb{S}^*$ in the real protocol, observe that

$$\frac{p}{q}\left(u \cdot \boldsymbol{d}_x^1 + v \cdot \boldsymbol{d}_x^2\right) = \frac{p}{q}\boldsymbol{a}_x k' + \frac{p}{q}(\boldsymbol{e}_1 k' + \boldsymbol{e}_1')u + \frac{p}{q}(\boldsymbol{e}_2 k' + \boldsymbol{e}_2')v \tag{7}$$

for $k', \boldsymbol{e}_1', \boldsymbol{e}_2'$ chosen by $\mathbb{S}^*$ where $\|k'\|_\infty \leq \sigma \cdot \sqrt{n}$ and $\|\boldsymbol{e}_1'\|_\infty, \|\boldsymbol{e}_2'\|_\infty \leq \sigma' \cdot \sqrt{n}$. Therefore, rounding the quantity in Equation (7) is guaranteed to result in the correct value if every coefficient of $\frac{p}{q} \cdot \boldsymbol{a}_x \cdot k'$ is further than

$$\left\| \frac{p}{q}(\boldsymbol{e}_1 k' + \boldsymbol{e}_1')u + \frac{p}{q}(\boldsymbol{e}_2 k' + \boldsymbol{e}_2')v \right\|_\infty$$

[7] Alternatively, if $\widetilde{\boldsymbol{d}}_x^1, \widetilde{\boldsymbol{d}}_x^2$ is consistent with k'

away from $\mathbb{Z} + 1/2$. In other words if $\mathbb{S}^*$ can force incorrect evaluation, it has found $k' \leq \sigma \cdot \sqrt{n}$ such that a coefficient of $\boldsymbol{a}_x k'$ is within a distance

$$\begin{aligned}\left\|(\boldsymbol{e}_1 \cdot k' + \boldsymbol{e}'_1) \cdot u + (\boldsymbol{e}_2 \cdot k' + \boldsymbol{e}'_2) \cdot v\right\|_\infty \\ &\leq 2\left(L \cdot \ell \cdot \sigma \cdot n^{5/2} \cdot \sigma \cdot \sqrt{n} + \sigma' \cdot \sqrt{n}\right) \cdot \sigma \cdot n^2 \\ &\overset{condition(ii)}{\leq} 4 \cdot \sigma' \cdot \sigma \cdot n^{5/2}\end{aligned}$$

of $\frac{q}{p}\mathbb{Z} + \frac{q}{2p} \subset \frac{q}{2p}\mathbb{Z}$. At this point we apply Lemma 5 with $2 \cdot p$, $T = 4 \cdot \sigma' \cdot \sigma \cdot n^{5/2}$ to show that $\mathbb{S}^*$ forcing incorrect evaluation with non-negligible probability violates the assumption that

$$\mathsf{1D\text{-}SIS}_{q/2p, n\cdot\ell, \max\{n^{3/2}\cdot\ell\cdot\sigma, 4\cdot\sigma'\cdot\sigma\cdot n^{5/2}\}}$$

is hard. Therefore, condition (iii) enforces correct evaluation with all but negligible probability when the parameters satisfy condition (ii). □

5.3 Setting the parameters

Let κ be the security parameter. Theorem 2 requires the following conditions:

- $\frac{q}{2p} \gg \sigma' \gg L \cdot \ell \cdot \sigma^2 \cdot n^3$
- $\mathsf{dRLWE}_{q,n,\sigma}$ is hard
- $\mathsf{1D\text{-}SIS}_{q/(2p), n\ell, 4\cdot\sigma'\cdot\sigma\cdot n^{5/2}}$ is hard.

We will be using the presumed hardness of SIVP_γ for approximation factors $\gamma = 2^{o(\sqrt{n})}$. The SIVP_γ lattice dimension associated to RLWE will be $n = \kappa^c$ (for some constant c); the dimension associated to 1D-SIS hardness will be $n' := \kappa$. We first choose $\sigma = \mathsf{poly}(n)$ and $\sigma' = \sigma \cdot \kappa^{\omega(1)}$, and then set $q = p \cdot \prod_{i=1}^{n'} p_i$ by picking coprime $p, p_1, \ldots, p_{n'} = 4\sigma'\sigma n^{5/2} \cdot \omega(\sqrt{nn' \log q \log n'})$. Having made these choices, it should be clear that the first of the three conditions is satisfied. We can apply Theorem 3 to argue RLWE hardness via SIVP for sub-exponential approximation factors $2^{\widetilde{\mathcal{O}}(n^{1/c})}$ (for $c > 2$), noting that $\sigma = \mathsf{poly}(n)$ and

$$\begin{aligned}q &= (4 \cdot \sigma' \cdot \sigma \cdot n^{5/2})^{n'} \omega((n \cdot n' \cdot \log q \cdot \log n')^{n'/2}) \\ &= 2^{(2+2\log\sigma+\omega(1)\log\kappa+(5/2)\log n)\cdot n^{1/c}} \cdot \omega((n \cdot n' \cdot \log q \cdot \log n')^{n'/2}) \\ &= 2^{\omega(1)\cdot n^{1/c}\cdot\log n} \cdot \omega((n^{1+\frac{1}{c}} \cdot \log q \cdot \log n)^{n^{1/c}/2}) \\ &= 2^{\widetilde{\mathcal{O}}(n^{1/c})}.\end{aligned}$$

Finally for the 1D-SIS condition, we note that $q/p = \prod_{i=1}^{n'} p_i$ and

$$\begin{aligned} p_1 &= 4 \cdot \sigma' \cdot \sigma \cdot n \cdot \omega(\sqrt{n \cdot n' \log q \cdot \log n'}) \\ &= 4 \cdot \sigma^2 \cdot \kappa^{\omega(1)} n \cdot \omega(\sqrt{n \cdot n' \cdot \log q \cdot \log n'}) \\ &= (n')^{\omega(1)} \cdot \omega(\sqrt{n'^{1+c} \cdot \log q \cdot \log n'}). \end{aligned}$$

So applying Lemma 10, we get hardness of our 1D-SIS instance via the presumed hardness of SIVP on n'-dimensional lattices for $(n')^{\omega(1)} \cdot \mathsf{poly}(n')$ approximation factors. We summarise the parameters of our construction in Table 1.

Parameter	Description	Requirement	Asymptotic
n	ring dimension	$n = \mathsf{poly}(\kappa)$	$\mathsf{poly}(\kappa)$
q	original modulus	$q = p \cdot \sigma' \cdot \kappa^{\omega(1)}$	$\kappa^{\omega(1)}$
p	rounding modulus	—	$\mathsf{poly}(\kappa)$
ℓ	$\log_2(q)$	—	$\log_2(\kappa^{\omega(1)})$
σ	secret/error distribution	$q/\sigma = 2^{o(\sqrt{n})}$	$\mathsf{poly}(\kappa)$
σ'	drowning distribution	$\sigma' = L\ell\sigma^2 n \cdot \kappa^{\omega(1)}$	$\kappa^{\omega(1)}$
L	bit-length of PRF input	—	—

Table 1. Parameters of our VOPRF

Acknowledgements

The research of Albrecht was supported by EPSRC grants EP/S020330/1 and EP/S02087X/1, and by the European Union Horizon 2020 Research and Innovation Program Grant 780701; the research of Deo was supported by the EPSRC and the UK government as part of the Centre for Doctoral Training in Cyber Security at Royal Holloway, University of London (EP/K035584/1); the research fof Smart was supported by ERC Advanced Grant ERC-2015-AdG-IMPaCT and by the FWO under an Odysseus project GOH9718N.

References

1. Babai, L.: On Lovász' lattice reduction and the nearest lattice point problem. Combinatorica 6(1), 1–13 (1986), https://doi.org/10.1007/BF02579403 13
2. Banaszczyk, W.: New bounds in some transference theorems in the geometry of numbers. Mathematische Annalen 296(1) (Dec 1993) 7
3. Banerjee, A., Peikert, C.: New and improved key-homomorphic pseudorandom functions. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 353–370. Springer, Heidelberg (Aug 2014) 2, 3, 4, 11, 12

4. Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 719–737. Springer, Heidelberg (Apr 2012) 2
5. Baum, C., Damgård, I., Lyubashevsky, V., Oechsner, S., Peikert, C.: More efficient commitments from structured lattice assumptions. In: Catalano, D., De Prisco, R. (eds.) SCN 18. LNCS, vol. 11035, pp. 368–385. Springer, Heidelberg (Sep 2018) 18
6. Benhamouda, F., Krenn, S., Lyubashevsky, V., Pietrzak, K.: Efficient zero-knowledge proofs for commitments from learning with errors over rings. In: Pernul, G., Ryan, P.Y.A., Weippl, E.R. (eds.) ESORICS 2015, Part I. LNCS, vol. 9326, pp. 305–325. Springer, Heidelberg (Sep 2015) 18
7. Beullens, W.: On sigma protocols with helper for mq and pkp, fishy signature schemes and more. Cryptology ePrint Archive, Report 2019/490 (2019), https://eprint.iacr.org/2019/490 18
8. Biasse, J.F., Espitau, T., Fouque, P.A., Gélin, A., Kirchner, P.: Computing generator in cyclotomic integer rings - A subfield algorithm for the principal ideal problem in $L_{|\Delta_{\mathbb{K}}|}(\frac{1}{2})$ and application to the cryptanalysis of a FHE scheme. In: Coron, J., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part I. LNCS, vol. 10210, pp. 60–88. Springer, Heidelberg (Apr / May 2017) 31
9. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: 20th ACM STOC. pp. 103–112. ACM Press (May 1988) 10
10. Boneh, D., Lewi, K., Montgomery, H.W., Raghunathan, A.: Key homomorphic PRFs and their applications. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 410–428. Springer, Heidelberg (Aug 2013) 2
11. Bootle, J., Lyubashevsky, V., Seiler, G.: Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. Cryptology ePrint Archive, Report 2019/642 (2019), https://eprint.iacr.org/2019/642 3, 17
12. Bootle, J., Lyubashevsky, V., Seiler, G.: Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 176–202. Springer, Heidelberg (Aug 2019) 3
13. Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) 45th ACM STOC. pp. 575–584. ACM Press (Jun 2013) 7
14. Brakerski, Z., Tsabary, R., Vaikuntanathan, V., Wee, H.: Private constrained PRFs (and more) from LWE. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part I. LNCS, vol. 10677, pp. 264–302. Springer, Heidelberg (Nov 2017) 2
15. Brakerski, Z., Vaikuntanathan, V.: Constrained key-homomorphic PRFs from standard lattice assumptions - or: How to secretly embed a circuit in your PRF. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 1–30. Springer, Heidelberg (Mar 2015) 5, 10, 11, 30, 31
16. Canetti, R., Chen, Y.: Constraint-hiding constrained PRFs for NC^1 from LWE. In: Coron, J., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part I. LNCS, vol. 10210, pp. 446–476. Springer, Heidelberg (Apr / May 2017) 2
17. Cramer, R., Ducas, L., Peikert, C., Regev, O.: Recovering short generators of principal ideals in cyclotomic rings. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 559–585. Springer, Heidelberg (May 2016) 31
18. Cramer, R., Ducas, L., Wesolowski, B.: Short stickelberger class relations and application to ideal-SVP. In: Coron, J., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part I. LNCS, vol. 10210, pp. 324–348. Springer, Heidelberg (Apr / May 2017) 31

19. Davidson, A., Goldberg, I., Sullivan, N., Tankersley, G., Valsorda, F.: Privacy pass: Bypassing internet challenges anonymously. PoPETs 2018(3), 164–180 (2018) 2
20. Davidson, A., Sullivan, N., Wood, C.: Oblivious pseudorandom functions (oprfs) using prime-order groups. Internet-Draft draft-irtf-cfrg-voprf-01, IETF Secretariat (July 2019), http://www.ietf.org/internet-drafts/draft-irtf-cfrg-voprf-01.txt 2
21. Dodis, Y., Goldwasser, S., Kalai, Y.T., Peikert, C., Vaikuntanathan, V.: Public-key encryption schemes with auxiliary inputs. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 361–381. Springer, Heidelberg (Feb 2010) 7
22. Don, J., Fehr, S., Majenz, C., Schaffner, C.: Security of the Fiat-Shamir transformation in the quantum random-oracle model. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part II. LNCS, vol. 11693, pp. 356–383. Springer, Heidelberg (Aug 2019) 3, 11, 17
23. Esgin, M.F., Steinfeld, R., Liu, J.K., Liu, D.: Lattice-based zero-knowledge proofs: New techniques for shorter and faster constructions and applications. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 115–146. Springer, Heidelberg (Aug 2019) 3
24. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (Aug 1987) 10, 11
25. Freedman, M.J., Ishai, Y., Pinkas, B., Reingold, O.: Keyword search and oblivious pseudorandom functions. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 303–324. Springer, Heidelberg (Feb 2005) 2
26. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC. pp. 197–206. ACM Press (May 2008) 7
27. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 75–92. Springer, Heidelberg (Aug 2013) 11
28. Goldreich, O., Micali, S., Wigderson, A.: How to prove all NP-statements in zero-knowledge, and a methodology of cryptographic protocol design. In: Odlyzko, A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 171–185. Springer, Heidelberg (Aug 1987) 10
29. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems (extended abstract). In: 17th ACM STOC. pp. 291–304. ACM Press (May 1985) 10
30. Hoffstein, J., Howgrave-Graham, N., Pipher, J., Silverman, J.H., Whyte, W.: NTRUSIGN: Digital signatures using the NTRU lattice. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 122–140. Springer, Heidelberg (Apr 2003) 5, 13
31. Jarecki, S., Kiayias, A., Krawczyk, H.: Round-optimal password-protected secret sharing and T-PAKE in the password-only model. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014, Part II. LNCS, vol. 8874, pp. 233–253. Springer, Heidelberg (Dec 2014) 2
32. Jarecki, S., Kiayias, A., Krawczyk, H., Xu, J.: Highly-efficient and composable password-protected secret sharing (or: How to protect your bitcoin wallet online). In: EuroS&P. pp. 276–291. IEEE (2016) 2
33. Jarecki, S., Krawczyk, H., Xu, J.: OPAQUE: An asymmetric PAKE protocol secure against pre-computation attacks. In: Nielsen, J.B., Rijmen, V. (eds.) EU-

ROCRYPT 2018, Part III. LNCS, vol. 10822, pp. 456–486. Springer, Heidelberg (Apr / May 2018) 2
34. Jarecki, S., Liu, X.: Efficient oblivious pseudorandom function with applications to adaptive OT and secure computation of set intersection. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 577–594. Springer, Heidelberg (Mar 2009) 2
35. Keelveedhi, S., Bellare, M., Ristenpart, T.: Dupless: Server-aided encryption for deduplicated storage. In: Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13). pp. 179–194. USENIX, Washington, D.C. (2013) 2
36. Krawczyk, H.: The opaque asymmetric pake protocol. Internet-Draft draft-krawczyk-cfrg-opaque-02, IETF Secretariat (July 2019), http://www.ietf.org/internet-drafts/draft-krawczyk-cfrg-opaque-02.txt, http://www.ietf.org/internet-drafts/draft-krawczyk-cfrg-opaque-02.txt 2
37. Libert, B., Ling, S., Mouhartem, F., Nguyen, K., Wang, H.: Signature schemes with efficient protocols and dynamic group signatures from lattice assumptions. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part II. LNCS, vol. 10032, pp. 373–403. Springer, Heidelberg (Dec 2016) 18
38. Libert, B., Ling, S., Mouhartem, F., Nguyen, K., Wang, H.: Zero-knowledge arguments for matrix-vector relations and lattice-based group encryption. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part II. LNCS, vol. 10032, pp. 101–131. Springer, Heidelberg (Dec 2016) 19, 36, 37, 39
39. Libert, B., Ling, S., Nguyen, K., Wang, H.: Zero-knowledge arguments for lattice-based PRFs and applications to E-cash. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017, Part III. LNCS, vol. 10626, pp. 304–335. Springer, Heidelberg (Dec 2017) 17, 19, 34, 36, 37
40. Ling, S., Nguyen, K., Stehlé, D., Wang, H.: Improved zero-knowledge proofs of knowledge for the ISIS problem, and applications. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 107–124. Springer, Heidelberg (Feb / Mar 2013) 17, 40
41. Liu, Q., Zhandry, M.: Revisiting post-quantum Fiat-Shamir. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part II. LNCS, vol. 11693, pp. 326–355. Springer, Heidelberg (Aug 2019) 3, 11, 17
42. Lyubashevsky, V.: Lattice-based identification schemes secure under active attacks. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 162–179. Springer, Heidelberg (Mar 2008) 3
43. Lyubashevsky, V.: Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 598–616. Springer, Heidelberg (Dec 2009) 17
44. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 738–755. Springer, Heidelberg (Apr 2012) 17
45. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (May / Jun 2010) 9
46. Peikert, C.: A decade of lattice cryptography. Cryptology ePrint Archive, Report 2015/939 (2015), http://eprint.iacr.org/2015/939 11
47. Peikert, C., Regev, O., Stephens-Davidowitz, N.: Pseudorandomness of ring-LWE for any ring and modulus. In: Hatami, H., McKenzie, P., King, V. (eds.) 49th ACM STOC. pp. 461–473. ACM Press (Jun 2017) 31

48. Peikert, C., Shiehian, S.: Privately constraining and programming PRFs, the LWE way. In: Abdalla, M., Dahab, R. (eds.) PKC 2018, Part II. LNCS, vol. 10770, pp. 675–701. Springer, Heidelberg (Mar 2018) 2
49. Pellet-Mary, A., Hanrot, G., Stehlé, D.: Approx-SVP in ideal lattices with pre-processing. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part II. LNCS, vol. 11477, pp. 685–716. Springer, Heidelberg (May 2019) 31
50. Pornin, T., Prest, T.: More efficient algorithms for the ntru key generation using the field norm. Cryptology ePrint Archive, Report 2019/015 (2019), https://eprint.iacr.org/2019/015 5, 13
51. Stehlé, D., Steinfeld, R.: Making NTRU as secure as worst-case problems over ideal lattices. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 27–47. Springer, Heidelberg (May 2011) 15
52. Stern, J.: A new identification scheme based on syndrome decoding. In: Stinson, D.R. (ed.) CRYPTO'93. LNCS, vol. 773, pp. 13–21. Springer, Heidelberg (Aug 1994) 18
53. Sullivan, N.: Cloudflare supports privacy pass. Cloudflare Blog (November 09 2017), https://blog.cloudflare.com/cloudflare-supports-privacy-pass/. Accessed Aug 2019 2
54. Yang, R., Au, M.H., Zhang, Z., Xu, Q., Yu, Z., Whyte, W.: Efficient lattice-based zero-knowledge arguments with standard soundness: Construction and applications. Cryptology ePrint Archive, Report 2019/747 (2019), https://eprint.iacr.org/2019/747 3, 17

A Computational Lattice Problems

An n-dimensional lattice Λ is a discrete subgroup of $\mathbb{R}^n$. The i^{th} successive minimum of a lattice Λ, denoted by $\lambda_i(\Lambda)$, is the radius of the smallest ball centred at the origin containing at least i linearly independent lattice vectors. Note that $\lambda_1(\Lambda)$ is the the length of the shortest non-zero lattice vector. In addition to the 1D-SIS and RLWE problems, it is useful to define the 1D-SISR problem:

Definition 8. ([15, Definition 3.6]) *Let $q = p \cdot \prod_{i\in[n]} p_i$ where $p_1 < \cdots < p_n$ and p are all co-prime. Further, let $m \in \mathbb{N}$. The $\mathsf{1D\text{-}SIS\text{-}R}_{q,p,m,t}$ problem is the following: Given $\boldsymbol{v} \leftarrow \mathbb{Z}_q^m$, find $\boldsymbol{z} \in \mathbb{Z}^m$ with $||\boldsymbol{z}||_\infty \leq t$ such that $\langle \boldsymbol{v}, \boldsymbol{z} \rangle \in [-t, t] + (q/p)\mathbb{Z}$.*

Next we recall some standard lattice problems.

Definition 9. (SVP_γ) *The* γ-approximate shortest vector problem*, denoted SVP_γ, asks that given any basis B of an n-dimensional lattice Λ, and $\gamma = \gamma(n) \geq 1$ that one finds a $\boldsymbol{v} \in \Lambda\backslash\{\mathbf{0}\}$ such that $\|\boldsymbol{v}\|_\infty \leq \gamma \cdot \lambda_1(\Lambda)$.*

Definition 10. (GapSVP_γ) *The* γ-gap shortest vector problem*, denoted GapSVP_γ is the following: Given any basis B of an n-dimensional lattice Λ, $\gamma = \gamma(n) \geq 1$,*

and $r \in \mathbb{R}_+$*, output* 1 *if* $\lambda_1(\Lambda) \leq r$*, and* 0 *if* $\gamma \cdot r \leq \lambda_1(\Lambda)$*. If* $\gamma \cdot r \leq \lambda_1(\Lambda) \leq r$*, then any output is acceptable.*

Definition 11. (SIVP_γ) *The* γ-shortest independent vectors problem*, denoted* SIVP_γ *is the following: Given any basis* B *of an* n*-dimensional lattice* Λ*, find* n *linearly independent vectors* $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ *such that* $\max(\|\boldsymbol{v}_i\|_2) \leq \gamma \cdot \lambda_n(\Lambda)$*.*

Writing $A \geq B$ to denote that there is a polynomial time reduction from B to A, we rely on the following reductions: $\mathsf{1D\text{-}SIS\text{-}R}_{q,p,m,t} \geq \mathsf{1D\text{-}SIS}_{q,m,t} \geq (\mathsf{GapSVP}_\gamma, \mathsf{SIVP}_\gamma)$ and $\mathsf{dRLWE}_{q,n,m,\sigma} \geq \mathsf{SIVP}_\gamma$. This is formalised in the following lemma statements.

Lemma 10. ([15, Corollary 3.5]) *Let* $n \in \mathbb{N}$ *and* $q = \prod_{i \in [n]} p_i$ *where all* $p_1 < \ldots, < p_n$ *are co-prime. Let* $m \geq cn \log q$ *(for some universal constant* c*). Assuming that* $p_1 \geq t \cdot \omega(\sqrt{mn \log n})$*,* $\mathsf{1D\text{-}SIS}_{q,m,t}$ *is at least as hard as* $\mathsf{SIVP}_{t \cdot \widetilde{\mathcal{O}}(\sqrt{mn})}$ *and* $\mathsf{GapSVP}_{t \cdot \widetilde{\mathcal{O}}(\sqrt{mn})}$*.*

Lemma 11. ([15, Corollary 3.7]) *Let* q, p, t, m *be as in Definition 8. Then the* $\mathsf{1D\text{-}SIS\text{-}R}_{q,p,m,t}$ *problem is at least as hard as* $\mathsf{1D\text{-}SIS}_{q/p,m,t}$*. Further, if* $p_1 \geq t \cdot \omega(\sqrt{mn \log n})$*, then* $\mathsf{1D\text{-}SIS\text{-}R}_{q,p,m,t}$ *is at least as hard as* $\mathsf{SIVP}_{t \cdot \widetilde{\mathcal{O}}(\sqrt{mn})}$ *and* $\mathsf{GapSVP}_{t \cdot \widetilde{\mathcal{O}}(\sqrt{mn})}$*.*

For hardness, we require that the approximation factors $t \cdot \widetilde{\mathcal{O}}(\sqrt{mn})$ be sub-exponential (in the lattice dimension) for the general lattice problems in the corollary above (see below the next lemma). We also recall a reduction from lattice problems to the RLWE problem:

Theorem 3. ([47], Corollary 6.3) *Let* $q = q(n) \geq 2$ *and* $\sigma < q$ *be such that* $\sigma \geq \omega(1)$*. Then* $\mathsf{RLWE}_{q,n,\sigma}$ *is at least as hard as* SIVP_γ *over ideal lattices where* $\gamma \leq \max\{\omega(\sqrt{n \log n} \cdot q/\sigma), 2\sqrt{n}\}$*.*

Previous work [8,17,18,49] shows that the best known algorithms solving SVP_γ for $\gamma = 2^{o(\sqrt{n})}$ have a superpolynomial cost in both the classical and quantum computing models. Therefore, we make the *assumption* that SIVP_γ for $\gamma = 2^{o(\sqrt{n})}$ cannot be solved efficiently.

B Various Results

B.1 Proof of Lemma 5

Proof. We will explicate a reduction from the related 1D-SIS-R$_{,,,}$ problem and then use Lemma 11 to complete the proof. Consider the following algorithm $\mathcal{A}$ using $\mathcal{D}$ as a sub-routine that attempts to solve 1D-SIS-R$_{q,p,n\ell,\max\{n\cdot\ell\cdot B,T\}}$ on input $\boldsymbol{v} \in \mathbb{Z}_q^{n\ell}$:

1. Let $j \in \{0,1\}$ denote the first bit of x and set $\boldsymbol{w}^j := \boldsymbol{v} \in \mathbb{Z}_q^{n\ell}$.
2. Sample $\boldsymbol{w}^{\bar{j}} \leftarrow \mathbb{Z}_q^{n\ell}$
3. For $i = 0, \ldots, \ell - 1$:
   $(a_j)_i = \sum_{k=0}^{n-1} w_{in+k}^j X^k$
   $(a_{\bar{j}})_i = \sum_{k=0}^{n-1} w_{in+k}^{\bar{j}} X^k$
4. Run $r \leftarrow \mathcal{D}(x, \boldsymbol{a}_0, \boldsymbol{a}_1)$.
5. If there is no coefficient of $\boldsymbol{a}_x \cdot r$ in the set $(q/p)\cdot\mathbb{Z} + [-T', T']$, then abort.
6. Otherwise let x' be the input x with the first bit removed. There is a coefficient of $\boldsymbol{a}_x \cdot r = \boldsymbol{a}_j \cdot G^{-1}(\boldsymbol{a}_{x'}) \cdot r$ in $(q/p)\cdot\mathbb{Z} + [-T, T]$ meaning that for some k^*, there is a column of $G^{-1}(\boldsymbol{a}_{x'}) \cdot r$, say $\boldsymbol{y} \in R_q^\ell$ such that the X^{k^*} coefficient of $\langle \boldsymbol{a}_j, \boldsymbol{y}\rangle$ is in $(q/p)\cdot\mathbb{Z} + [-T, T]$.
7. Let $\mathbb{1}_{(\cdot)}$ be an indicator function. Noting that the coefficient of X^{k^*} of $\langle \boldsymbol{a}_j, \boldsymbol{y}\rangle$ is equal to
$$\sum_{i=0}^{\ell-1}\sum_{k=0}^{n-1} v_{in+k} \cdot (-1)^{\mathbb{1}_{k>k^*}} (y_i)_{k^*-k \bmod n},$$
output $\boldsymbol{z} \in \mathbb{Z}_q^{n\ell}$ where $z_{in+k} = (-1)^{\mathbb{1}_{k>k^*}} (y_i)_{k^*-k \bmod n}$ for $i = 0, \ldots, \ell-1$, $k = 0, \ldots, n-1$.

It is clear that if $\mathcal{A}$ does not abort, it outputs a vector $\boldsymbol{z} \in \mathbb{Z}_q^{n\ell}$ such that $\langle \boldsymbol{v}, \boldsymbol{z}\rangle \in (q/p)\cdot\mathbb{Z} + [-T, T]$. Furthermore, if no abort occurs, then the entries of $\boldsymbol{z}$ (up to a sign) correspond to the coefficients of a column of $r \cdot G^{-1}(\boldsymbol{a}_{x'})$ where $\|r\|_\infty \leq B$ with non-negligible probability. Recalling that $G^{-1}(\boldsymbol{a}_{x'}) \in R_q^{\ell\times\ell}$ is a binary decomposition of polynomials, we can see that,

$$||\boldsymbol{z}||_\infty \leq \ell \cdot n \cdot B$$

with non-negligible probability. In other words, $\mathcal{A}$ solves the 1D-SIS-R$_{q,p,n\ell,\max\{n\ell B,T\}}$ problem in polynomial time with non-negligible probability. To complete the proof, we use Lemma 11. □

B.2 Upper Bound on $\boldsymbol{u}, \boldsymbol{v}$

Babai's rounding technique is an efficient way of obtaining a candidate solution to CVP. Given a target point $\boldsymbol{t} \in \mathbb{R}^n$ and a lattice Λ with basis B (which need

not be an invertible square matrix), Babai's rounding technique outputs the lattice vector $\boldsymbol{w} = B\lfloor(B^TB)^{-1}B^T\boldsymbol{t}\rceil$. The offset vector obtained can therefore be written as

$$\boldsymbol{t} - \boldsymbol{w} = B \cdot \left((B^TB)^{-1}B^T\boldsymbol{t} - \lfloor(B^TB)^{-1}B^T\boldsymbol{t}\rceil\right) \in B \cdot \left[-\frac{1}{2}, \frac{1}{2}\right]^n. \tag{8}$$

Let $\boldsymbol{b}'_i$ denote the i^{th} *row* of B. From Equation (8), we have that

$$||\boldsymbol{t} - \boldsymbol{w}||_\infty \leq \frac{1}{2} \cdot \max_i ||\boldsymbol{b}'_i||_1 \leq \frac{\sqrt{n}}{2} \cdot \max_i ||\boldsymbol{b}'_i||_2. \tag{9}$$

We now use this analysis to give an upper bound on $||(u, v)||_\infty$ that is computed in the algorithm from Section 3. At a high level, the first four steps find a (potentially very long) pair $(u, v) \in R$ such that $u \cdot s + v \cdot t = 1 \mod R_q$ and the final two steps update this (u, v) using Babai's rounding technique. In particular, suppose we define $S, T \in \mathbb{Z}_q^{n\times n}$ to be the negacyclic matrices denoting multiplication by s and t respectively. Then the final two steps run Babai's rounding technique on the lattice $\Lambda = \{\boldsymbol{z} \in \mathbb{Z}^{2n} : [S|T] \cdot \boldsymbol{z} = 0 \mod q\}$ and target point $\boldsymbol{t} = (u, v)$ (using the coefficient embedding), and update (u, v) to be the resulting offset. The basis for Λ used is $B = [T| - S]^T \in \mathbb{Z}^{2n\times n}$ (which has linearly independent columns by invertibility of s, t in the field $\mathbb{Q}(X)/\langle X^n + 1\rangle$). Therefore, bounding the infinity norm of the offset (via Equation (9)) gives us a bound for the final value of $||(u, v)||_\infty$. Noting that each row of our basis consists of the coefficients of $s, t \leftarrow \chi_\sigma$, we obtain the bound

$$||(u, v)||_\infty \leq \frac{\sqrt{n}}{2} \cdot ||(s, t)||_2 \leq \frac{\sqrt{n}}{2} \cdot 2\sigma\sqrt{n} = n\sigma$$

that holds with all but negligible probability over the choice of s and t.

C Our Stern Protocol for Proof System 1

In this section we outline how we rewrite our problem in the Stern proof as a linear system of equations and describe a set VALID alongside a set of permutations Γ possessing the relevant properties given in Section 4. We recall the **key properties**:

1. For every $\phi \in \mathcal{S}, \vec{w} \in \mathsf{VALID} \iff \Gamma_\phi(\vec{w}) \in \mathsf{VALID}$.
2. For every $\vec{w} \in \mathsf{VALID}$, the distribution of $\Gamma_\phi(\vec{w})$ (for $\phi \leftarrow \mathcal{S}$) is uniform over the set VALID.

(Randomised) PRF Evaluation and the ZK Relation. Recall that G^{-1} is the non-linear binary decomposition operation, and G is the powers of two matrix that undoes G^{-1}. Also recall that in the query phase, the client computes the function $F'_{k;\boldsymbol{e}} : \{0,1\}^L \to R_q^{1\times\ell}$ where

$$F'_{k;\boldsymbol{e}}(x) = \boldsymbol{a}_{x_0} \cdot G^{-1}\left(\boldsymbol{a}_{x_1} \cdot G^{-1}\left(\boldsymbol{a}_{x_2} G^{-1}\left(\ldots\right)\right)\right) \cdot k + \boldsymbol{e} \bmod q. \tag{10}$$

Note that this function is similar to, but not exactly the same as the PRF F from Section 2.4. In particular, the function F' is a *randomised* version of F where the error $\boldsymbol{e}$ is not obtained in a deterministic fashion. Note, however, that the techniques of this section can be straight-forwardly adapted to prove the analogous relation that uses F instead of F' (see [39]). In terms of the function F', the relation we are interested in providing a ZKPoK for is

$$\begin{aligned}\mathcal{R} = \{(\boldsymbol{a}_0, \boldsymbol{a}_1, \boldsymbol{y}_1, \boldsymbol{y}_2), (s, t, x, \boldsymbol{e}_1, \boldsymbol{e}_2) \in (R_q^{1\times\ell})^4 \times \left((R)^2, \{0,1\}^L, (R^{1\times\ell})^2\right) \\ : \boldsymbol{y}_1 = F'_{k;\boldsymbol{e}_1}(x), \boldsymbol{y}_2 = F'_{k;\boldsymbol{e}_2}(x) \\ \|s\|_\infty, \|t\|_\infty \leq \beta_1, \\ \|\boldsymbol{e}_1\|_\infty, \|\boldsymbol{e}_2\|_\infty \leq \beta_2\}.\end{aligned}$$

To begin with, we can describe the computation of $F'_{s,\boldsymbol{e}_1}(x)$ and $F'_{t,\boldsymbol{e}_2}(x)$ recursively using

$$\begin{aligned} B_{L-1} &= G^{-1}(\boldsymbol{a}_{x_{L-1}}) \\ B_{L-2} &= G^{-1}(\boldsymbol{a}_{x_{L-2}} \cdot B_{L-1}) \\ B_{L-3} &= G^{-1}(\boldsymbol{a}_{x_{L-3}} \cdot B_{L-2}) \\ &\vdots \\ B_0 &= G^{-1}(\boldsymbol{a}_{x_0} \cdot B_1) \\ F_{s;\boldsymbol{e}_1}(x) &= G \cdot B_0 \cdot s + \boldsymbol{e}_1 \\ F_{t,\boldsymbol{e}_2}(x) &= G \cdot B_0 \cdot t + \boldsymbol{e}_2 \end{aligned}$$

where each equation is considered over the ring R_q. Importantly, $B_i \in R_2^{\ell\times\ell}$ represent binary decompositions and $\boldsymbol{a} \in R_q^{1\times\ell}$.

Evaluation of $\boldsymbol{F'}$ as a System of Linear Equations. However, the system of equations above is not linear since G^{-1} is not a linear operator. In the hope of deriving a linear system of equations that we can use Stern's protocol on, we first multiply by the linear operator $G \in R_q^{1\times\ell}$ or equivalently $\boldsymbol{g}^T = (1, 2, \ldots, 2^{\ell-1}) \in$

$R_q^{1\times\ell}$. In doing so, we can set $\boldsymbol{b}_0 = (\boldsymbol{g}^T \cdot B_0) \in R_q^\ell$ to obtain

$$\begin{aligned}
\boldsymbol{g}^T \cdot B_{L-1} &= \boldsymbol{a}_{x_{L-1}} && (11)\\
\boldsymbol{g}^T \cdot B_{L-2} &= \boldsymbol{a}_{x_{L-2}} \cdot B_{L-1}\\
\boldsymbol{g}^T \cdot B_{L-3} &= \boldsymbol{a}_{x_{L-3}} \cdot B_{L-2}\\
&\vdots\\
\boldsymbol{b}_0^T &= \boldsymbol{a}_{x_0} \cdot B_1 && (12)\\
F'_{s;\boldsymbol{e}_1}(x) &= \boldsymbol{b}_0 \cdot s + \boldsymbol{e}_1\\
F'_{t;\boldsymbol{e}_2}(x) &= \boldsymbol{b}_0 \cdot t + \boldsymbol{e}_2.
\end{aligned}$$

We now wish to come up with a ZKPoK allowing to prove knowledge of $\{(B_i)_{i=1}^{L-1}, \boldsymbol{b}_0, s, t, \boldsymbol{e}_1, \boldsymbol{e}_2\}$ (where $s, t, \boldsymbol{e}_1, \boldsymbol{e}_2$ are short, and $B_i \in R_2^{\ell\times\ell}$) satisfying the above system of linear equations.

Three Problems with the Linear System. In order to use Stern's protocol, the witness must be a vector with entries $\mathbb{Z}_q$ that solves some publicly known linear system. Considering the current formulation, we have three initial problems to solve:

1. The B_i's are matrices, rather than vectors,
2. The "witness" $\{(B_i)_{i=1}^{L-1}, \boldsymbol{b}_0, s, t, \boldsymbol{e}_1, \boldsymbol{e}_2\}$ consists of vectors/matrices with entries in R_q rather than $\mathbb{Z}_q$.
3. The system is quadratic in unknowns, rather than linear.

Solving the First Problem. To get the unknowns $B_i \in R_2^{\ell\times\ell}$ in vector-form rather than matrix-form, we can introduce some tensor products. For $i = 1, \ldots, L-1$, define $\boldsymbol{b}_i \in R_2^{\ell^2}$ to be the vector consisting of the columns of B_i stacked on top of each other. Inserting the appropriate tensor products, Equations (11)-(12) end up being of the form

$$\begin{aligned}
(I_\ell \otimes \boldsymbol{g}^T) \cdot \boldsymbol{b}_i &= (I_\ell \otimes \boldsymbol{a}_{x_i}) \cdot \boldsymbol{b}_{i+1},\\
\boldsymbol{b}_0 &= (I_\ell \otimes \boldsymbol{a}_{x_0}) \cdot \boldsymbol{b}_1.
\end{aligned}$$

Solving the Second Problem. We would like to replace all multiplications in R_q by a matrix-vector multiplication over $\mathbb{Z}_q$. To do so we simply use the well known negacyclic matrices over $\mathbb{Z}_q$ that represent multiplication in R_q. We define $\boldsymbol{A}_0 \in \mathbb{Z}_q^{n\times n\ell}$ (and $\boldsymbol{A}_1$) to be the *horizontal* concatenation of the negacyclic matrices corresponding to the entries of $\boldsymbol{a}_0 \in R_q^{1\times\ell}$ (resp. $\boldsymbol{a}_1$). Furthermore, we define $\boldsymbol{S} \in \mathbb{Z}_q^{n\times n}$ (and $\boldsymbol{T}$) to be the negacyclic matrices representing $s \in R_q$ (resp.

t). Note that this turns part of our witness back into a matrices, but we will show how to deal with this using the techniques of [38] later. Also, for $i = 0, \ldots, L-1$, let $\vec{b}_i$ be the vertical concatenation of the coefficients in the ring entries of $\boldsymbol{b}_i$ and let $\vec{e}_1$ (resp. $\vec{e}_2$) be the vertical concatenation of coefficients in the entries of $\boldsymbol{e}_1$ (resp. $\boldsymbol{e}_2$). Further, let $\vec{y}_1$ and $\vec{y}_2$ be the vertical concatenation of the coefficients in $F'_{s,\boldsymbol{e}_1}(x)$ and $F'_{t,\boldsymbol{e}_2}(x)$ respectively. Setting $\boldsymbol{G}^{\otimes} = I_\ell \otimes (\boldsymbol{g}^T \otimes I_n)$ for $\boldsymbol{g}^T = (1, \ldots, 2^\ell) \in \mathbb{Z}_q^{1\times\ell}$, $\boldsymbol{A}^{\otimes}_{x_i} = I_\ell \otimes \boldsymbol{A}_{x_i} \in \mathbb{Z}_q^{n\ell\times n\ell^2}$ and $\vec{b}_L$ to be the binary vector with a 1 in each block of $n\ell$ entries (at the $(i-1)n+1^{th}$ position in the i^{th} block), we end up with the following system of equations mod q:

$$\boldsymbol{G}^{\otimes} \cdot \vec{b}_{L-1} = \boldsymbol{A}^{\otimes}_{x_{L-1}} \vec{b}_L \tag{13}$$

$$\boldsymbol{G}^{\otimes} \cdot \vec{b}_{L-2} = \boldsymbol{A}^{\otimes}_{x_{L-2}} \cdot \vec{b}_{L-1}$$

$$\boldsymbol{G}^{\otimes} \cdot \vec{b}_{L-3} = \boldsymbol{A}^{\otimes}_{x_{L-3}} \cdot \vec{b}_{L-2} \tag{14}$$

$$\vdots$$

$$\vec{b}_0 = \boldsymbol{A}^{\otimes}_{x_0} \cdot \vec{b}_1 \tag{15}$$

$$\vec{y}_1 = (I_\ell \otimes \boldsymbol{S}) \cdot \vec{b}_0 + \vec{e}_1 \tag{16}$$

$$\vec{y}_2 = (I_\ell \otimes \boldsymbol{T}) \cdot \vec{b}_0 + \vec{e}_2. \tag{17}$$

where $\vec{b}_0 \in \mathbb{Z}_q^{n\ell}$, $\vec{b}_i \in \{0,1\}^{n\ell^2}$ for $i \neq 0$ and $\boldsymbol{S}, \boldsymbol{T}$ have small entries.

Solving the Third Problem. We very briefly overview the techniques of [39] to indicate how one can linearise Equations (13) to (15). The idea is to represent $\boldsymbol{A}^{\otimes}_{x_i} \cdot \vec{b}_{i+1}$ by writing

$$\boldsymbol{A}^{\otimes}_{x_i} \cdot \vec{b}_{i+1} = [\boldsymbol{A}^{\otimes}_0 | \boldsymbol{A}^{\otimes}_1] \cdot \begin{bmatrix} \bar{x_i} \vec{b}_{i+1} \\ x_i \vec{b}_{i+1}. \end{bmatrix}$$

In order to make use of this, we treat unknowns x_i and $\vec{b}_{i+1}$ together by considering the single unknown

$$\vec{\tilde{b}}_i = \begin{bmatrix} \bar{x_i} \vec{b}_{i+1} \\ x_i \vec{b}_{i+1} \end{bmatrix} \tag{18}$$

In doing so, Equations (13) - (14) end up being of the form

$$[\boldsymbol{G}^{\otimes} | \boldsymbol{G}^{\otimes}] \cdot \vec{\tilde{b}}_{i-1} = [\boldsymbol{A}^{\otimes}_0 | \boldsymbol{A}^{\otimes}_1] \cdot \vec{\tilde{b}}_i.$$

where for $i = 1, \ldots, L-1$, valid solutions $\vec{\tilde{b}}_i$ are of the form given in (18) i.e. a binary vector where the top half or bottom half of entries are 0. Equation (15) becomes

$$\vec{b}_0 = [\boldsymbol{A}^{\otimes}_0 | \boldsymbol{A}^{\otimes}_1] \cdot \vec{\tilde{b}}_1. \tag{19}$$

Now we turn our attention to Equations (16) and (17). The high level idea for obtaining equations linear in unknowns is the same. We essentially rewrite the equations in terms of a new single unknown that depends quadratically in the old unknowns and then take note of the structure that this induces on valid solutions. We only consider the term $(I_\ell \otimes \boldsymbol{S}) \cdot \vec{b}_0$ from Equation (16) since the quadratic term in Equation (17) can be dealt with in exactly the same way. For the ring element $s = \sum_{i=0}^{n-1} s_i x^i \in R_q$ corresponding to $\boldsymbol{S} \in \mathbb{Z}_q^{n \times n}$, it is clear that if we know the products $s_i \cdot (\vec{b}_0)_j$ for every $(i,j) \in \{0,\dots,n-1\} \times \{1,\dots,n\ell\}$, then we can calculate $(I_\ell \otimes \boldsymbol{S}) \cdot \vec{b}_0$ since every entry will be a linear combination of these products. Therefore, letting $\vec{s} \in \mathbb{Z}_q^n$ be the coefficient vector of s, we can write $\vec{z}_s = \vec{b}_0 \otimes \vec{s}$ so that

$$(I_\ell \otimes \boldsymbol{S}) \cdot \vec{b}_0 = \boldsymbol{Q} \cdot \vec{z}_s \bmod q$$

where $\boldsymbol{Q} \in \mathbb{Z}_2^{n\ell \times n^2 \ell}$ is some known constant matrix. Note that this methodology is the same as in [38] apart from the fact that $\boldsymbol{Q}$ is defined using the structure of R_q in our case. It is also useful here to express $\vec{b}_0$ in terms of its binary decomposition vector. In particular, we define $\vec{\tilde{b}}_0 \in \{0,1\}^{n\ell^2}$ to be the vertical concatenation of the binary decomposition of entries in $\vec{b}_0$. We can also rewrite $\vec{s}$ using a special binary decomposition. In particular, set $\delta_j = \lfloor (\beta_1 + 2^{j-1})/2^j \rfloor$ for $j = 1,\dots,\lfloor \log \beta_1 \rfloor + 1$, i.e. powers of two in reverse order plus β_1, and set $\boldsymbol{D}_\beta = \boldsymbol{I}_n \otimes (\delta_1,\dots,\delta_{\lfloor \log \beta_1 \rfloor + 1})$. As in [39], we can efficiently find a vector $\vec{s}' \in \{-1,0,1\}^{n(\lfloor \log \beta_1 \rfloor + 1)}$ such that $\boldsymbol{D}_{\beta_1} \vec{s}' = \vec{s}$ for any $\vec{s} \in \{-\beta_1,\dots,\beta_1\}^n$. In addition, $\sum_{i=1}^{\lfloor \log \beta_1 \rfloor + 1} \delta_i = \beta_1$ so for any $\vec{s}' \in \{-1,0,1\}^{n(\lfloor \log \beta_1 \rfloor + 1)}$, $\|D_{\beta_1} \cdot \vec{s}'\|_\infty \le \beta_1$. Defining $\boldsymbol{H}_q := (1,2,\dots,2^{\ell-1}) \otimes \boldsymbol{I}_{n\ell}$, we have that $\boldsymbol{H}_q \vec{\tilde{b}}_0 = \vec{b}_0$. Similarly, defining $\boldsymbol{R} := \boldsymbol{Q} \cdot (\boldsymbol{H}_q \otimes \boldsymbol{D}_\beta)$ and $\vec{z}'_s := \vec{\tilde{b}}_0 \otimes \vec{s}'$ we can write

$$(I_\ell \otimes \boldsymbol{S}) \cdot \vec{b}_0 = \boldsymbol{R} \cdot \vec{z}'_s \bmod q.$$

Note that we can derive a similar equation for t. We can also define $\boldsymbol{D}_{\beta_2}$ similarly to $\boldsymbol{D}_{\beta_2}$ to decompose $\vec{e}_1, \vec{e}_2$ into trinary $\vec{e}'_1, \vec{e}'_2$.

The Final Linear System. Finally, we arrive at the following system modulo q:

$$\begin{aligned}
[\boldsymbol{G}^\otimes | \boldsymbol{G}^\otimes] \cdot \vec{\tilde{b}}_{L-1} &= [\boldsymbol{A}_0^\otimes | \boldsymbol{A}_1^\otimes] \cdot \vec{\tilde{b}}_L \\
[\boldsymbol{G}^\otimes | \boldsymbol{G}^\otimes] \cdot \vec{\tilde{b}}_{L-2} &= [\boldsymbol{A}_0^\otimes | \boldsymbol{A}_1^\otimes] \cdot \vec{\tilde{b}}_{L-1} \\
&\vdots \\
\boldsymbol{H}_q \cdot \vec{\tilde{b}}_0 &= [\boldsymbol{A}_0^\otimes | \boldsymbol{A}_1^\otimes] \cdot \vec{\tilde{b}}_1 \\
\vec{y}_1 &= \boldsymbol{R} \cdot \vec{z}'_s + \boldsymbol{D}_{\beta_2} \cdot \vec{e}'_1 \\
\vec{y}_2 &= \boldsymbol{R} \cdot \vec{z}'_t + \boldsymbol{D}_{\beta_2} \cdot \vec{e}'_2
\end{aligned} \tag{20}$$

(21)

where valid solutions are such that:

- $\vec{\tilde{b}}_L = (1,0)^T \otimes \vec{c}$ or $(0,1)^T \otimes \vec{c}$ where for $\widehat{c}_i = \overbrace{(0,\ldots,0}^{i-1},1,\overbrace{0,\ldots,0)}^{\ell-i}$,

$$\vec{c} = (\widehat{c}_1||\widehat{c}_2,||\ldots||\widehat{c}_\ell)^T \otimes \overbrace{(1,0,\ldots,0)^T}^{n} \tag{22}$$

- for $i = 1,\ldots,L-1$, $\vec{\tilde{b}}_i \in \{0,1\}^{2n\ell^2}$ and either the first or second $n\ell^2$ entries are 0
- $\vec{\tilde{b}}_0 \in \{0,1\}^{n\ell^2}$
- $\vec{z}'_s = \vec{\tilde{b}}_0 \otimes \vec{s}'$ for some $\vec{s}' \in \{-1,0,1\}^{n(\lfloor \log \beta_1 \rfloor + 1)}$
- $\vec{z}'_t = \vec{\tilde{b}}_0 \otimes \vec{t}'$ for some $\vec{t}' \in \{-1,0,1\}^{n(\lfloor \log \beta_1 \rfloor + 1)}$
- $\vec{e}'_1, \vec{e}'_2 \in \{-1,0,1\}^{n\ell\cdot(\lfloor \log \beta_2 \rfloor + 1)}$

The Building Block Extensions and Permutations. Now we will show how to use Stern's protocol to prove knowledge of a *valid* solution/witness

$$\vec{\psi} = \begin{bmatrix} \vec{\tilde{b}}_L \\ \vdots \\ \vec{\tilde{b}}_0 \\ \vec{\tilde{b}}_0 \otimes \vec{s}' \\ \vec{\tilde{b}}_0 \otimes \vec{t}' \\ \vec{e}'_1 \\ \vec{e}'_2 \end{bmatrix} \tag{23}$$

to the system $\boldsymbol{M} \cdot \vec{\psi} = \vec{y}$ implicit in Equations (20)-(21). We do this in the standard way by extending the witness vector (and updating the system), and then defining a set VALID along with a set of permutations Γ such that the two key properties from Section 4 hold. We begin by describing an extension and permutation for each section of the witness.

Extension for $\vec{\tilde{b}}_L$.
It turns out that we do not need to extend the part of the witness comprising $\vec{\tilde{b}}_L$. All we need to do is define the permutations indexed by bit b, $\pi_b : \{0,1\}^{2n\ell^2} \to \{0,1\}^{2n\ell^2}$. Writing $\vec{v} = (\vec{v}_0, \vec{v}_1)$ where $\vec{v}_0, \vec{v}_1 \in \{0,1\}^{n\ell^2}$, we define π_b via the equation $\pi_b(\vec{v}) = (\vec{v}_b, \vec{v}_{\bar{b}})$. In words, this permutation either does nothing or switches a valid $\vec{\tilde{b}}_L$ to the other valid option according to the value of $b \in \{0,1\}$.

Extension for $\vec{\tilde{b}}_1, \ldots, \vec{\tilde{b}}_{L-1}$.

Recalling that either the second or first half of entries in each of $\vec{\tilde{b}}_1, \ldots, \vec{\tilde{b}}_{L-1}$ is 0, we define the extension Ext_0 to act as follows on a vector $\vec{v} \in \{0,1\}^{2n'}$. Writing $\vec{v} = (\vec{v}_1, \vec{v}_2)$ where $\vec{v}_1, \vec{v}_2 \in \{0,1\}^{n'}$ and letting h be the hamming weight of $\vec{v}$, we define

$$\mathsf{Ext}_1(\vec{v}) = \begin{cases} (\vec{v}_1, \overbrace{1, \ldots, 1}^{n'-h}, \overbrace{0, \ldots, 0}^{h}, \vec{v}_2, \vec{v}_2) \text{ if } \vec{v}_2 = \vec{0}, \\ (\vec{v}_1, \vec{v}_1, \vec{v}_2, \underbrace{1, \ldots, 1}_{n'-h}, \underbrace{0, \ldots, 0}_{h}) \text{ if } \vec{v}_1 = \vec{0}. \end{cases}$$

The corresponding permutations are given by $\pi_b \circ \tau_\sigma$ where τ_σ is indexed by $\sigma \in \mathcal{S}_{n'}$ where $\mathcal{S}_{n'}$ is the symmetric group on n' elements. Writing $\vec{v}_1 = (v_{1,1}, \ldots, v_{1,n'})$ and $\vec{v}_2 = (v_{2,1}, \ldots, v_{2,n'})$, we define $\tau_\sigma(\vec{v}) := (v_{1,\sigma(1)}, \ldots, v_{1,\sigma(n')}, v_{2,\sigma(1)}, \ldots, v_{2,\sigma(n')})$.

Extension for $\vec{\tilde{b}}_0$.

For a vector $\vec{v} = (v_1, \ldots, v_{n'}) \in \{0,1\}^{n'}$, we define

$$\mathsf{Ext}_2(\vec{v}) = (v_1, \bar{v}_1, v_2, \bar{v}_2, \ldots, v_{n'}, \bar{v}_{n'}) \in \{0,1\}^{2n'}.$$

The corresponding permutations $\rho_{\vec{d}} : \{0,1\}^{2n'} \to \{0,1\}^{2n'}$ are indexed by $\vec{d} \in \{0,1\}^{n'}$. For $\vec{w} = (w_{1,0}, w_{1,1}, w_{2,0}, w_{2,1}, \ldots, w_{n',0}, w_{n',1})$, we define $\rho_{\vec{d}}(\vec{w}) := (w_{1,d_1}, w_{1,\bar{d}_1}, \ldots, w_{n',d_{n'}}, w_{n',\bar{d}_{n'}})$. The crucial observation is that

$$\vec{w} = \mathsf{Ext}_2(\vec{v}) \iff \rho_{\vec{d}}(\vec{w}) = \mathsf{Ext}_2(\vec{v} \oplus \vec{d}).$$

Product extensions for $\vec{\tilde{b}}_0 \otimes \vec{s}'$, $\vec{\tilde{b}}_0 \otimes \vec{t}'$.

In [38], an extension and permutation for products of two bits compatible with Stern's protocol is presented. Inspired by this, we first show an extension and permutation that can handle products between $c_1 \in \{0,1\}$ and $c_2 \in \{-1,0,1\}$. For $c \in \{1,2\}$, we use the notation $c_2^{+c} = c_2 + c \bmod 3$ and define

$$\mathsf{Ext}_3(c_1, c_2) := (c_1 c_2, c_1 c_2^{+1}, c_1 c_2^{+2}, \bar{c}_1 c_2, \bar{c}_1 c_2^{+1}, \bar{c}_1 c_2^{+2}) \in \{-1,0,1\}^6. \tag{24}$$

Let cyc denote the clockwise cyclic permutation on entries of a 3 dimensional vector. The corresponding building-block permutations are indexed by $b_1 \in \{0,1\}, b_2 \in \{-1,0,1\}$, and are defined by

$$T^3_{b_1,b_2} : (\vec{v}_0, \vec{v}_1) \longmapsto (\mathsf{cyc}^{b_2}(\vec{v}_{b_1}), \mathsf{cyc}^{b_2}(\vec{v}_{\bar{b}_1}))$$

where $\vec{v}_i \in \{-1,0,1\}^3$ for $i = 0, 1$. This ensures that

$$\vec{v} = \mathsf{Ext}_3(c_1, c_2) \iff T^3_{b_1,b_2}(\vec{v}) = \mathsf{Ext}_3(c_1 \oplus b_1, c_2 \oplus_3 b_2) \tag{25}$$

where $\oplus_3$ denotes addition modulo 3 and $\vec{v} = (\vec{v}_0, \vec{v}_1) \in \{-1,0,1\}^6$. This permutation essentially one time pads both c_1 and c_2. We can generalise Ext_3 and $T^3_{(,)}$ to act on vectors $\vec{a} = (a_1, \ldots, a_{n'}) \in \{0,1\}^{n'}$ and $\vec{b} = (b_1, \ldots, b_{n''}) \in \{-1,0,1\}^{n''}$ as follows. Informally, the generalised extension $\mathsf{Ext}_3^{\otimes}$ is the vertical concatenation of the $\mathsf{Ext}_3((a_i, b_j))$ ranging over $i = 1, \ldots, n'$ and $j = 1, \ldots, n''$. More precisely,

$$\begin{aligned}\mathsf{Ext}_{3,\otimes}(\vec{a}, \vec{b}) =& \mathsf{Ext}_3(a_1, b_1)||\mathsf{Ext}_3(a_1, b_2)|| \ldots ||\mathsf{Ext}_3(a_1, b_{n''})|| \ldots || \ldots || \ldots \\ & \ldots \quad \mathsf{Ext}_3(a_{n'}, b_1)||\mathsf{Ext}_3(a_{n'}, b_2)|| \ldots ||\mathsf{Ext}_3(a_{n'}, b_{n''}).\end{aligned}$$

Importantly, this generalised extension contains all entries arising in the tensor product $\vec{a} \otimes \vec{b}$, so can be considered as an extension of $\vec{a} \otimes \vec{b}$. The generalised permutations are indexed by $\vec{c} = (c_1, \ldots, c_{n'}) \in \{0,1\}^{n'}$, $\vec{d} = (d_1, \ldots, d_{n''}) \in \{-1,0,1\}^{n''}$ and are denoted $T^{3,\otimes}_{\vec{c},\vec{d}}$. Writing $\vec{v} = (\vec{v}_{1,1}, \ldots, \vec{v}_{1,n''}, \ldots, \vec{v}_{n',1}, \ldots, \vec{v}_{n',n''})$ where each $\vec{v}_{i,j} \in \{-1,0,1\}^6$, we define

$$\begin{aligned}T^{3,\otimes}_{\vec{c},\vec{d}}(\vec{v}) :=& (T^3_{c_1,d_1}(\vec{v}_{1,1})||T^3_{c_1,d_2}(\vec{v}_{1,2})|| \ldots ||T^3_{c_1,d_{n''}}(\vec{v}_{1,n''})|| \ldots || \ldots || \ldots \\ & \ldots \quad ||T^3_{c_{n'},d_1}(\vec{v}_{n',1})||T^3_{c_{n'},d_2}(\vec{v}_{n',2})|| \ldots ||T^3_{c_{n'},d_{n''}}(\vec{v}_{n',n''})).\end{aligned}$$

Using these definitions, we have

$$\vec{v} = \mathsf{Ext}_{3,\otimes}(\vec{a}, \vec{b}) \iff T^{3,\otimes}_{\vec{c},\vec{d}}(\vec{v}) = \mathsf{Ext}_{3,\otimes}(\vec{a} \oplus \vec{c}, \vec{b} \oplus_3 \vec{d}). \tag{26}$$

Extension for $\vec{e}'_1, \vec{e}'_2$.
Here we use the technique from [40]. For any $\vec{v} \in \{-1,0,1\}^{n'}$ with h_{-1}, h_0, h_1 entries equal to $-1, 0, 1$ respectively, we define the extension

$$\mathsf{Ext}'(\vec{v}) = (\vec{v}||\overbrace{-1, \ldots, -1}^{n'-h_{-1}}, \overbrace{0, \ldots, 0}^{n'-h_0}, \overbrace{1, \ldots, 1}^{n'-h_1}).$$

Note that this outputs a vector in $\{-1,0,1\}^{3n'}$ with exactly n' entries that take each of the values $-1, 0, 1$. The corresponding permutations τ'_σ are indexed by $\sigma \in \mathcal{S}_{3n'}$ where $\mathcal{S}_{3n'}$ is the symmetric group over $3n'$ elements. For $\vec{w} = (w_1, \ldots, w_{3n'})$, the permutation τ'_σ is defined via $\tau'_\sigma(\vec{w}) = (w_{\sigma(1)}, \ldots, w_{\sigma(3n')})$.

The Full Extension, Permutation and Valid Set. Using the extensions in the previous section, we extend the witness of the form (23) to the following:

$$\vec{\psi}' = \begin{bmatrix} \vec{\tilde{b}}_L \\ \mathsf{Ext}_1(\vec{\tilde{b}}_{L-1}) \\ \vdots \\ \mathsf{Ext}_1(\vec{\tilde{b}}_1) \\ \mathsf{Ext}_2(\vec{\tilde{b}}_0) \\ \mathsf{Ext}_{3,\otimes}(\vec{\tilde{b}}_0 \otimes \vec{s}') \\ \mathsf{Ext}_{3,\otimes}(\vec{\tilde{b}}_0 \otimes \vec{t}') \\ \mathsf{Ext}'(\vec{e}'_1) \\ \mathsf{Ext}'(\vec{e}'_2) \end{bmatrix} \tag{27}$$

This forces us to update the system of equations to $\boldsymbol{M}' \cdot \vec{\psi}' = \vec{y}$. We now conclude by defining the set VALID and set of permutations Γ satisfying the key properties from Section 4.

We will say that $\vec{v} = (\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_L, \vec{v}_{L+1}, \vec{v}_{L+2}, \vec{v}_{L+3}, \vec{v}_{L+4}, \vec{v}_{L+5}) \in \mathsf{VALID}$ if and only if:

- $\vec{v}_1 \in \{0,1\}^{2n\ell^2}$ is either $(1,0)^T \otimes \vec{c}$ or $(0,1)^T \otimes \vec{c}$ where $\vec{c}$ is defined in Equation (22).
- $\vec{v}_2, \ldots, \vec{v}_L \in \{0,1\}^{4n\ell^2}$ have Hamming weight $n\ell^2$ with either the first half or second half of entries all 0.
- $\vec{v}_{L+1} \in \{0,1\}^{2n\ell^2}$ and is consistent with the form of vector output by Ext_2.
- Letting $\vec{w}$ be the valid preimage of $\vec{v}_{L+1}$ under Ext_2, $\vec{v}_{L+2}$ and $\vec{v}_{L+3} \in \{-1,0,1\}^{6n^2\ell^2(\log\lfloor\beta_1\rfloor+1)}$ are of the form $\mathsf{Ext}_{3,\otimes}(\vec{w}, \vec{s}'')$ and $\mathsf{Ext}_{3,\otimes}(\vec{w}, \vec{t}'')$ respectively for some $s'', t'' \in \{-1,0,1\}^{n(\lfloor\log\beta_1\rfloor+1)}$
- $\vec{v}_{L+4}, \vec{v}_{L+5} \in \{-1,0,1\}^{3n\ell(\lfloor\log\beta_2\rfloor+1)}$ have an equal number of $-1, 0, 1$ entries

The permutation set Γ is

$$\left\{ \Gamma_{\boldsymbol{\phi}} \;\middle|\; \begin{array}{l} \boldsymbol{\phi} = (\phi_1, \phi_{2,1}, \phi_{2,2}, \ldots, \phi_{L,1}, \phi_{L,2}, \phi_{L+1}, \phi_{L+2}, \ldots, \phi_{L+5}), \\ \phi_1, \phi_{2,1}, \phi_{3,1}, \ldots, \phi_{L,1} \in \{0,1\} \\ \phi_{2,2}, \ldots, \phi_{L,2} \in \mathcal{S}_{n\ell^2} \\ \phi_{L+1} \in \{0,1\}^{n\ell^2} \\ \phi_{L+2}, \phi_{L+3} \in \{-1,0,1\}^{n(\lfloor\log\beta_1\rfloor+1)} \\ \phi_{L+4}, \phi_{L+5} \in \mathcal{S}_{3n\ell(\lfloor\log\beta_2\rfloor+1)} \end{array} \right\} \tag{28}$$

where

$$\Gamma_{\boldsymbol{\phi}} : \begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \\ \vdots \\ \vec{v}_{L+1} \\ \vec{v}_{L+2} \\ \vec{v}_{L+3} \\ \vec{v}_{L+4} \\ \vec{v}_{L+5} \end{bmatrix} \longmapsto \begin{bmatrix} \pi_{\phi_1}(\vec{v}_1) \\ \pi_{\phi_{2,1}} \circ \tau_{\phi_{2,2}}(\vec{v}_2) \\ \vdots \\ \pi_{\phi_{L,1}} \circ \tau_{\phi_{L,2}}(\vec{v}_L) \\ \rho_{\phi_{L+1}}(\vec{v}_{L+1}) \\ T^{3,\otimes}_{\phi_{L+1},\phi_{L+2}}(\vec{v}_{L+2}) \\ T^{3,\otimes}_{\phi_{L+1},\phi_{L+3}}(\vec{v}_{L+3}) \\ \tau'_{\phi_{L+4}}(\vec{v}_{L+4}) \\ \tau'_{\phi_{L+5}}(\vec{v}_{L+5}) \end{bmatrix} \tag{29}$$

Finally, we note that $\Gamma_{\boldsymbol{\phi}}$ acts on elements of VALID in the following way:

$$\mathsf{VALID} \ni \begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \\ \vdots \\ \vec{v}_L \\ \mathsf{Ext}_2(\vec{\nu}_{L+1}) \\ \mathsf{Ext}_{3,\otimes}(\vec{\nu}_{L+1} \otimes \vec{s}'') \\ \mathsf{Ext}_{3,\otimes}(\vec{\nu}_{L+1} \otimes \vec{t}'') \\ \vec{v}_{L+4} \\ \vec{v}_{L+5} \end{bmatrix} \mapsto \begin{bmatrix} \pi_{\phi_1}(\vec{v}_1) \\ \pi_{\phi_{2,1}} \circ \tau_{\phi_{2,2}}(\vec{v}_2) \\ \vdots \\ \pi_{\phi_{L,1}} \circ \tau_{\phi_{L,2}}(\vec{v}_L) \\ \mathsf{Ext}_2(\vec{\nu}_{L+1} \oplus \phi_{L+1}) \\ \mathsf{Ext}_{3,\otimes}((\vec{\nu}_{L+1} \oplus \phi_{L+1}) \otimes (\vec{s}'' \oplus_3 \phi_{L+2}) \\ \mathsf{Ext}_{3,\otimes}((\vec{\nu}_{L+1} \oplus \phi_{L+1}) \otimes (\vec{t}'' \oplus_3 \phi_{L+3}) \\ \tau'_{\phi_{L+4}}(\vec{v}_{L+4}) \\ \tau'_{\phi_{L+5}}(\vec{v}_{L+5}) \end{bmatrix}.$$

The above implies both the first and second key properties required for the abstract version of Stern's protocol. In particular, a random permutation from Γ one-time pads the arguments of the Ext sections of the witness and randomly permutes the remaining sections of the witness in a structure preserving way. This observation implies that the second key property holds.