Adaptively Secure Distributed PRFs from LWE

Benoît Libert[1,2(✉)], Damien Stehlé[2], and Radu Titiu[2,3]

[1] CNRS, Laboratoire LIP, Lyon, France
benoit.libert@ens-lyon.fr
[2] ENS de Lyon, Laboratoire LIP (U. Lyon, CNRS, ENSL, INRIA, UCBL), Lyon, France
[3] Bitdefender, Bucharest, Romania

Abstract. In distributed pseudorandom functions (DPRFs), a PRF secret key SK is secret shared among N servers so that each server can locally compute a partial evaluation of the PRF on some input X. A combiner that collects t partial evaluations can then reconstruct the evaluation $F(SK, X)$ of the PRF under the initial secret key. So far, all non-interactive constructions in the standard model are based on lattice assumptions. One caveat is that they are only known to be secure in the static corruption setting, where the adversary chooses the servers to corrupt at the very beginning of the game, before any evaluation query. In this work, we construct the first fully non-interactive adaptively secure DPRF in the standard model. Our construction is proved secure under the LWE assumption against adversaries that may adaptively decide which servers they want to corrupt. We also extend our construction in order to achieve robustness against malicious adversaries.

Keywords: LWE · Pseudorandom functions · Distributed PRFs
Threshold cryptography · Adaptive security

1 Introduction

A pseudorandom function (PRF) family [35] is a set $\mathcal{F}$ of keyed functions with common domain Dom and range Rng such that no ppt adversary can distinguish a real experiment, where it has oracle access to a random member $f \hookleftarrow \mathcal{F}$ of the PRF family, from an ideal experiment where it is interacting with a truly random function $R : \mathsf{Dom} \rightarrow \mathsf{Rng}$. To be useful, a PRF should be efficiently computable – meaning that $F_{\mathbf{s}}(x)$ must be deterministically computable in polynomial time given the key $\mathbf{s}$ and the input $x \in \mathsf{Dom}$ – and the key size must be polynomial.

Pseudorandom functions are fundamental objects in cryptography as most central tasks of symmetric cryptography (like secret-key encryption, message authentication or identification) can be efficiently realized from a secure PRF family. Beyond their use for cryptographic purposes, they can also be used to prove circuit lower bounds [56] and they are strongly connected to the hardness of certain tasks in learning theory [62].

A. Beimel and S. Dziembowski (Eds.): TCC 2018, LNCS 11240, pp. 391–421, 2018.
https://doi.org/10.1007/978-3-030-03810-6_15

Goldreich, Goldwasser and Micali (GGM) [35] showed how to build a PRF from any length-doubling pseudorandom generator (PRG). In turn, PRGs are known [39] to exist under the sole assumption that one-way functions exist. However, much more efficient constructions can be obtained by relying on specific number theoretic assumptions like the Decision Diffie-Hellman assumption [51] and related variants [17,21,28,44] or the hardness of factoring [51,52].

In the context of lattice-based cryptography, the noisy nature of hard-on-average problems, like Learning-With-Errors (LWE) [57], makes it challenging to design efficient PRF families. The LWE assumption for a modulus q states that, given a random matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ with $m > n$, the vector $\mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ is computationally indistinguishable from a uniform vector over $\mathbb{Z}_q^m$ when $\mathbf{s} \in \mathbb{Z}_q^n$ is uniformly chosen in $\mathbb{Z}_q^n$ and $\mathbf{e} \in \mathbb{Z}^m$ is a small-norm noise vector sampled from a Gaussian distribution. In order to design PRFs with small-depth evaluation circuits, several works [7,8,16] rely on the Learning-With-Rounding (LWR) technique [8], which is a "de-randomization" of LWE where noisy vectors $\mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ are replaced by rounded vectors $\lfloor (p/q) \cdot (\mathbf{A} \cdot \mathbf{s}) \rfloor \in \mathbb{Z}_p^m$ for a smaller modulus $p < q$.

An appealing advantage of lattice-based techniques is that they enable the design of *key-homomorphic* PRF families [7,16]. Namely, assuming that their range and key space form an additive group, for any input x and keys $\mathbf{s}, \mathbf{t}$, we have $F_{\mathbf{s}+\mathbf{t}}(x) \approx F_{\mathbf{s}}(x) + F_{\mathbf{t}}(x)$. In turn, key-homomorphic PRFs provide simple and non-interactive constructions of distributed pseudorandom functions [50]. In a (threshold) distributed PRF (DPRF), secret keys are broken into N shares $\mathbf{s}_1, \ldots, \mathbf{s}_N$, each of which is given to a different server. Using its secret key share $\mathbf{s}_i$, the i-th server can locally compute a partial evaluation $F_{\mathbf{s}_i}(x)$ of the function. A dedicated server can then gather at least $t \leq N$ correct partial evaluations $F_{\mathbf{s}_{i_1}}(x), \ldots, F_{\mathbf{s}_{i_t}}(x)$ and reconstruct the evaluation $F_{\mathbf{s}}(x)$ for the long-term key $\mathbf{s}$. As such, threshold PRFs inherit the usual benefits of threshold cryptography [25]. First, setting $t < N$ allows for fault-tolerant systems that can keep running when some server crashes. Second, the adversary is forced to break into t servers to compromise the security of the whole scheme. Ideally, servers should be able to generate their partial evaluations without interacting with one another.

Boneh *et al.* [16] gave a generic construction of non-interactive DPRF from any almost key homomorphic PRF (where "almost" means that $F_{\mathbf{s}+\mathbf{t}}(x)$ only needs to be sufficiently "close" to $F_{\mathbf{s}}(x) + F_{\mathbf{t}}(x)$). Their construction, however, is only proved to be secure under *static* corruptions. Namely, the adversary has to choose the corrupted servers all-at-once and before making any evaluation query.

Contribution. We consider the problem of proving security in the stronger *adaptive* corruption model, where the adversary chooses which servers it wants to corrupt based on the previously obtained information. In particular, an adaptive adversary is allowed to obtain partial evaluations before corrupting any server.

In this stronger adversarial model, we provide the first realization of non-interactive distributed pseudorandom function with a security proof under a polynomial reduction. We prove the security of our construction in the standard model under the Learning-With-Errors (LWE) assumption [57] with super-polynomial approximation factors.

In its basic version, our DPRF is only secure against passive adversaries. However, robustness against malicious adversaries can be readily achieved using leveled homomorphic signatures [37], as was suggested by earlier works on threshold lattice-based cryptography [14,15]. To our knowledge, we thus obtain the first DPRF candidate which is simultaneously: (i) secure under adaptive corruptions in the standard model under a well-studied assumption; (ii) robust against malicious adversaries; (iii) non-interactive (i.e., each server only sends one message to the combiner that reconstructs the final output of the PRF).

Techniques. For a polynomial N and when $t \approx N/2$, proving adaptive security is considerably more challenging as a trivial complexity leveraging argument (i.e., guessing the set of corrupted servers upfront) makes the reduction super-polynomial. Moreover, we show that allowing a *single* partial evaluation query before the first corruption query already results in a definition which is strictly stronger than that of static security. In the adaptive corruption setting, the difficulty is that, by making N partial evaluation queries before corrupting any server, the adversary basically commits the challenger to all secret key shares. Hence, a reduction that only knows $t - 1 \approx N/2$ shares is unlikely to work as it would have to make up its mind on which set of $t - 1$ shares it wants to know at the outset of the game. In particular, this hinders a generic reduction from the security of an underlying key-homomorphic PRF. This suggests to find a reduction that knows all shares of the secret key, making it easier to consistently answer adaptive corruption queries.

To this end, we turn to lossy trapdoor functions [54], which are function families that contain both injective and lossy functions with computationally indistinguishable evaluation keys. We rely on the fact that the LWE function and its deterministic LWR variant [8] are both lossy trapdoor functions (as shown in [6,9,36]). Namely, the function that maps $\mathbf{s} \in \mathbb{Z}^n$ to $\lfloor \mathbf{A} \cdot \mathbf{s} \rfloor_p$ is injective when $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ is a random matrix and becomes lossy when $\mathbf{A}$ is of the form $\bar{\mathbf{A}} \cdot \mathbf{C} + \mathbf{E}$, where $\bar{\mathbf{A}} \in \mathbb{Z}_q^{m \times n'}$, $\mathbf{C} \in \mathbb{Z}_q^{n' \times n}$ are uniformly random and $\mathbf{E} \in \mathbb{Z}^{m \times n}$ is a small-norm matrix. Our idea is to first construct a PRF which maps an input x to $\lfloor \mathbf{A}(x) \cdot \mathbf{s} \rfloor_p$, where $\mathbf{s} \in \mathbb{Z}^n$ is the secret key and $\mathbf{A}(x) \in \mathbb{Z}_q^{m \times n}$ is derived from public matrices. We thus evaluate a lossy trapdoor function on an input consisting of the secret key using a matrix that depends on the input. In the security proof, we use admissible hash functions [13] and techniques from fully homomorphic encryption [33] to "program" $\mathbf{A}(x)$ in such a way that, with non-negligible probability, it induces a lossy function in all evaluation queries and an injective function in the challenge phase.[1] (We note that this use of lossy trapdoor functions is somewhat unusual since their injective mode is usually used to handle adversarial queries while the lossy mode comes into play in the challenge phase.) By choosing a large enough ratio q/p, we can make sure that

[1] We use a "find-then-guess" security game where the adversary obtains correct evaluation for inputs of its choice before trying to distinguish a real function evaluation from a random element of the range.

evaluation queries always reveal the same information about the secret $\mathbf{s}$. Since $\lfloor \mathbf{A}(x^\star) \cdot \mathbf{s} \rfloor_p$ is an injective function in the challenge phase, we can argue that it has high min-entropy, even conditionally on responses to evaluation queries. At this point, we can extract statistically uniform bits from $\lfloor \mathbf{A}(x^\star) \cdot \mathbf{s} \rfloor_p$ using a deterministic randomness extractor: analogously to the deterministic encryption case [55], we need to handle a source that may be correlated with the seed.

We note that the above approach bears resemblance with key-homomorphic PRFs [7,16] which also evaluate functions of the form $\lfloor \mathbf{A}(x) \cdot \mathbf{s} \rfloor_p$. However, our proof method is very different in that it relies on the lossy mode of LWE and the homomorphic encryption scheme of [33]. The advantage of our approach is that the challenger knows the secret key $\mathbf{s}$ at all steps of the security proof. In the distributed setting, this makes it easier to handle adaptive adversaries because the reduction can always correctly answer corruption queries. In order to share the secret key $\mathbf{s}$ among N servers, we rely on the Linear Integer Secret Sharing (LISS) schemes of Damgård and Thorbek [24], which nicely fit the requirements of our security proof. Among other properties, they allow secret key shares to remain small with respect to the modulus, which helps us making sure that partial evaluations – as lossy functions of their share – always reveal the same information about uncorrupted shares. Moreover, they also enable small reconstruction constants: the secret $\mathbf{s}$ can be recovered as a linear combination of authorized shares with coefficients in $\{-1, 0, 1\}$, which is useful to avoid blowing up error terms when partial evaluations are combined together. A notable difference with [24] is that our DPRF uses a LISS scheme with Gaussian entries (instead of uniform ones), which makes it easier to analyze the remaining entropy of the key in the final step of the proof.

Related Work. Distributed pseudorandom functions were initially suggested by Micali and Sidney [47] and received a lot of attention since then [27,29,50,51,53]. They are motivated by the construction of distributed symmetric encryption schemes, distributed key distribution centers [50], or distributed coin tossing and asynchronous byzantine agreement protocols [18]. They also provide a distributed source of random coins that allows removing interaction from threshold decryption mechanisms, such as the one of Canetti and Goldwasser [20].

As mentioned in [16], the early DPRF realizations [47] were only efficient when the threshold t was very small or very large with respect to N. Before 2010, other solutions [27,29,50,51,53] either required random oracles [50] or multiple rounds of interaction [27,29,51,53]. Boneh, Lewi, Montgomery and Raghunathan [16] (BLMR) suggested a generic construction of non-interactive DPRF from key-homomorphic PRFs. They also put forth the first key-homomorphic PRF in the standard model assuming the hardness of LWE. Banerjee and Peikert [7] generalized the BLMR construction and obtained more efficient constructions under weaker LWE assumptions. Boneh *et al.* [14,15] described another generic DPRF construction from a general "universal thresholdizer" tool, which allows distributing many cryptographic functionalities. So far, none of these solutions is known to provide security under adaptive corruptions.

In the context of threshold cryptography, adaptive security has been addressed in a large body of work [1,5,19,30,42,46]. These techniques, however, require interaction (except in some cases when all players always correctly provide their contribution to the computation) and none of them is known to be compatible with existing non-interactive DPRFs. While lattice-based threshold protocols were studied by Bendlin *et al.* back in 2010 [11,12], they focused on distributing decryption operations or sharing lattice trapdoors and it is not clear how to apply them in our setting. Boneh *et al.* [14,15] showed how to generically compile cryptographic functionalities into threshold functionalities using distributed FHE. However, they do not consider adaptive corruptions and proceed by generically evaluating the circuit of the functionality at hand. While we follow their approach of using fully homomorphic signatures to acquire robustness, our basic PRF is a direct and more efficient construction.

To our knowledge, the approach of using lossy trapdoor functions to construct advanced PRFs was never considered before. In spirit, our construction is somewhat similar to a random-oracle-based threshold signature proposed in [45], which also relies on the idea of always revealing the same information about the key in all evaluation queries. This DDH-based threshold signature can be turned into an adaptively secure DPRF in the random oracle model (like a variant of the Naor-Pinkas-Reingold DPRF [50]) but it has no standard-model counterpart.

The idea of using randomness extraction as part of the security proof of a PRF appears in [38, Sect. 6.2], where the function only needs to be secure in a model without evaluation queries. Here, we have to handle a different setting which prevents us from using the standard Leftover Hash Lemma.

Organization. Sect. 2 recalls some relevant material about lattices, pseudorandom functions and integer secret sharing. A centralized version of our DPRF is presented in Sect. 3 as a warm-up. We describe its distributed variant in Sect. 4. In the full version of the paper, we explain how the techniques of [14,15] apply to obtain robustness without using interaction nor random oracles.

2 Background

For any $q \geq 2$, we let $\mathbb{Z}_q$ denote the ring of integers with addition and multiplication modulo q. We always set q as a prime integer. For $2 \leq p < q$ and $x \in \mathbb{Z}_q$, we define $\lfloor x \rfloor_p := \lfloor (p/q) \cdot x \rfloor \in \mathbb{Z}_p$. This notation is readily extended to vectors over $\mathbb{Z}_p$. If $\mathbf{x}$ is a vector over $\mathbb{R}$, then $\|\mathbf{x}\|$ denotes its Euclidean norm. If $\mathbf{M}$ is a matrix over $\mathbb{R}$, then $\|\mathbf{M}\|$ denotes its induced norm. We let $\sigma_n(\mathbf{M})$ denote the least singular value of $\mathbf{M}$, where n is the rank of $\mathbf{M}$. For a finite set S, we let $U(S)$ denote the uniform distribution over S. If X is a random variable over a countable domain, the min-entropy of X is defined as $H_\infty(X) = \min_x(-\log_2 \Pr[X = x])$. If X and Y are distributions over the same domain, then $\Delta(X, Y)$ denotes their statistical distance.

2.1 Lattices

Let $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ be a symmetric positive definite matrix, and $\mathbf{c} \in \mathbb{R}^n$. We define the Gaussian function on $\mathbb{R}^n$ by $\rho_{\mathbf{\Sigma},\mathbf{c}}(\mathbf{x}) = \exp(-\pi(\mathbf{x}-\mathbf{c})^\top \mathbf{\Sigma}^{-1}(\mathbf{x}-\mathbf{c}))$ and if $\mathbf{\Sigma} = \sigma^2 \cdot \mathbf{I}_n$ and $\mathbf{c} = \mathbf{0}$ we denote it by ρ_σ.

For a lattice Λ, we define $\eta_\varepsilon(\Lambda)$ as the smallest $r > 0$ such that $\rho_{1/r}(\widehat{\Lambda} \setminus \mathbf{0}) \leq \varepsilon$ with $\widehat{\Lambda}$ denoting the dual of Λ, for any $\varepsilon \in (0,1)$. In particular, we have $\eta_{2^{-n}}(\mathbb{Z}^n) \leq O(\sqrt{n})$. We define $\lambda_1^\infty(\Lambda) = \min(\|\mathbf{x}\|_\infty : \mathbf{x} \in \Lambda \setminus \mathbf{0})$.

For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, we define the lattices $\Lambda^\perp(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A} \cdot \mathbf{x} = \mathbf{0} \bmod q\}$ and $\Lambda(\mathbf{A}) = \mathbf{A}^\top \cdot \mathbb{Z}^n + q\mathbb{Z}^m$.

Lemma 2.1 ([32, Lemma 5.3]). *Let $m \geq 2n \cdot \log q$ and $q \geq 2$ prime and let $\mathbf{A} \hookleftarrow U(\mathbb{Z}_q^{n \times m})$. With probability $\geq 1 - 2^{-\Omega(n)}$, we have $\lambda_1^\infty(\Lambda(\mathbf{A})) \geq q/4$.*

Lemma 2.2 (Adapted from [49, Lemma 4.4]). *For any n-dimensional lattice Λ, $\mathbf{x}', \mathbf{c} \in \mathbb{R}^n$ and symmetric positive definite $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ satisfying $\sigma_n(\sqrt{\mathbf{\Sigma}}) \geq \eta_{2^{-n}}(\Lambda)$, we have*

$$\rho_{\mathbf{\Sigma},\mathbf{c}}(\Lambda + \mathbf{x}') \in [1 - 2^{-n}, 1 + 2^{-n}] \cdot \det(\mathbf{\Sigma})^{1/2} / \det(\Lambda).$$

Lemma 2.3. *For $c \in \mathbb{R}$ and $\sigma > 0$ such that $\sigma \geq \sqrt{\ln 2(1 + 1/\epsilon)/\pi}$, we have*

$$H_\infty(D_{\mathbb{Z},\sigma,c}) \geq \log(\sigma) + \log(1 + 2e^{-\pi\sigma^2}) - \log\left(1 + \frac{2\epsilon}{1-\epsilon}\right)$$

Proof. From [49, Lemma 3.3] we know that $\eta_\epsilon(\mathbb{Z}) \leq \sqrt{\ln 2(1 + 1/\epsilon)/\pi}$. So $\sigma \geq \eta_\epsilon(\mathbb{Z})$. By [48, Lemma 2.5], this implies that $\frac{1-\epsilon}{1+\epsilon} \cdot \rho_\sigma(\mathbb{Z}) \leq \rho_{\sigma,c}(\mathbb{Z})$, which translates into

$$H_\infty(D_{\mathbb{Z},\sigma,c}) \geq H_\infty(D_{\mathbb{Z},\sigma}) - \log\left(\frac{1+\epsilon}{1-\epsilon}\right)$$

From [58, Claim 8.1], we have $\rho_\sigma(\mathbb{Z}) \geq \sigma \cdot (1 + 2e^{-\pi\sigma^2})$, so

$$H_\infty(D_{\mathbb{Z},\sigma}) \geq \log \sigma + \log(1 + 2e^{-\pi\sigma^2})$$

□

Remark 2.4. For $\sigma = \Omega(\sqrt{n})$, we get $H_\infty(D_{\mathbb{Z},\sigma,c}) \geq \log(\sigma) - 2^{-n}$.

Definition 2.5 (LWE). *Let $m \geq n \geq 1$, $q \geq 2$ and $\alpha \in (0,1)$ be functions of a security parameter λ. The* LWE *problem consists in distinguishing between the distributions $(\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{e})$ and $U(\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m)$, where $\mathbf{A} \sim U(\mathbb{Z}_q^{m \times n})$, $\mathbf{s} \sim U(\mathbb{Z}_q^n)$ and $\mathbf{e} \sim D_{\mathbb{Z}^m,\alpha q}$. For an algorithm $\mathcal{A} : \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m \to \{0,1\}$, we define:*

$$\mathbf{Adv}_{q,m,n,\alpha}^{\mathsf{LWE}}(\mathcal{A}) = |\Pr[\mathcal{A}(\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{e}) = 1] - \Pr[\mathcal{A}(\mathbf{A}, \mathbf{u}) = 1|,$$

where the probabilities are over $\mathbf{A} \sim U(\mathbb{Z}_q^{m \times n})$, $\mathbf{s} \sim U(\mathbb{Z}_q^n)$, $\mathbf{u} \sim U(\mathbb{Z}_q^m)$ and $\mathbf{e} \sim D_{\mathbb{Z}^m,\alpha q}$ and the internal randomness of $\mathcal{A}$. We say that $\mathsf{LWE}_{q,m,n,\alpha}$ is hard if for all ppt *algorithm $\mathcal{A}$, the advantage $\mathbf{Adv}_{q,m,n,\alpha}^{\mathsf{LWE}}(\mathcal{A})$ is negligible.*

Micciancio and Peikert [48] described a trapdoor mechanism for LWE. Their technique uses a "gadget" matrix $\mathbf{G} \in \mathbb{Z}_q^{n\times m}$ for which anyone can publicly sample short vectors $\mathbf{x} \in \mathbb{Z}^m$ such that $\mathbf{G} \cdot \mathbf{x} = \mathbf{0}$. As in [48], we call $\mathbf{R} \in \mathbb{Z}^{m\times m}$ a $\mathbf{G}$-trapdoor for a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times 2m}$ if $\mathbf{A} \cdot [\mathbf{R}^\top \mid \mathbf{I}_m]^\top = \mathbf{H} \cdot \mathbf{G}$ for some invertible matrix $\mathbf{H} \in \mathbb{Z}_q^{n\times n}$ which is referred to as the trapdoor tag. If $\mathbf{H} = \mathbf{0}$, then $\mathbf{R}$ is called a "punctured" trapdoor for $\mathbf{A}$.

Lemma 2.6 ([48, Section 5]). *Assume that $m \geq 2n \log q$. There exists a* ppt *algorithm* GenTrap *that takes as inputs matrices $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{H} \in \mathbb{Z}_q^{n\times n}$ and outputs matrices $\mathbf{R} \in \{-1, 1\}^{m\times m}$ and*

$$\mathbf{A} = [\bar{\mathbf{A}} \mid -\bar{\mathbf{A}}\mathbf{R} + \mathbf{H}\mathbf{G}] \in \mathbb{Z}_q^{n\times 2m}$$

such that if $\mathbf{H} \in \mathbb{Z}_q^{n\times n}$ is invertible, then $\mathbf{R}$ is a $\mathbf{G}$-trapdoor for $\mathbf{A}$ with tag $\mathbf{H}$; and if $\mathbf{H} = \mathbf{0}$, then $\mathbf{R}$ is a punctured trapdoor.

Further, in case of a $\mathbf{G}$-trapdoor, one can efficiently compute from $\mathbf{A}, \mathbf{R}$ and $\mathbf{H}$ a basis $(\mathbf{b}_i)_{i\leq 2m}$ of $\Lambda^\perp(\mathbf{A})$ such that $\max_i \|\mathbf{b}_i\| \leq O(m^{3/2})$.

Micciancio and Peikert also showed that a $\mathbf{G}$-trapdoor for $\mathbf{A} \in \mathbb{Z}_q^{n\times 2m}$ can be used to invert the LWE function $(\mathbf{s}, \mathbf{e}) \mapsto \mathbf{A}^\top \cdot \mathbf{s} + \mathbf{e}$, for any $\mathbf{s} \in \mathbb{Z}_q^n$ and any sufficiently short $\mathbf{e} \in \mathbb{Z}^{2m}$.

2.2 Admissible Hash Functions

Admissible hash functions were introduced by Boneh and Boyen [13] as a combinatorial tool for partitioning-based security proofs for which Freire *et al.* [31] gave a simplified definition. Jager [41] considered the following generalization in order to simplify the analysis of reductions under decisional assumption.

Definition 2.7 ([41]). *Let $\ell(\lambda), L(\lambda) \in \mathbb{N}$ be functions of a security parameter $\lambda \in \mathbb{N}$. Let* AHF : $\{0,1\}^\ell \to \{0,1\}^L$ *be an efficiently computable function. For every $K \in \{0, 1, \perp\}^L$, let the partitioning function $P_K : \{0,1\}^\ell \to \{0,1\}$ be defined as*

$$P_K(X) := \begin{cases} 0 & \text{if} \quad \forall i \in [L] \quad (\mathsf{AHF}(X)_i = K_i) \ \vee \ (K_i = \perp) \\ 1 & \text{otherwise} \end{cases}$$

We say that AHF *is a* **balanced admissible hash function** *if there exists an efficient algorithm* $\mathsf{AdmSmp}(1^\lambda, Q, \delta)$ *that takes as input $Q \in \mathsf{poly}(\lambda)$ and a non-negligible $\delta(\lambda) \in (0, 1]$ and outputs a key $K \in \{0, 1, \perp\}^L$ such that, for all $X^{(1)}, \ldots, X^{(Q)}, X^\star \in \{0,1\}^\ell$ such that $X^\star \notin \{X^{(1)}, \ldots, X^{(Q)}\}$, we have*

$$\gamma_{\max}(\lambda) \geq \Pr_K \left[P_K(X^{(1)}) = \cdots = P_K(X^{(Q)}) = 1 \ \wedge \ P_K(X^\star) = 0 \right] \geq \gamma_{\min}(\lambda),$$

where $\gamma_{\max}(\lambda)$ and $\gamma_{\min}(\lambda)$ are functions such that

$$\tau(\lambda) = \gamma_{\min}(\lambda) \cdot \delta(\lambda) - \frac{\gamma_{\max}(\lambda) - \gamma_{\min}(\lambda)}{2}$$

is a non-negligible function of λ.

Intuitively, the condition that $\tau(\lambda)$ be non-negligible requires $\gamma_{\min}(\lambda)$ to be noticeable and the difference of $\gamma_{\max}(\lambda) - \gamma_{\min}(\lambda)$ to be small.

It is known [41] that balanced admissible hash functions exist for $\ell, L = \Theta(\lambda)$.

Theorem 2.8 ([41, Theorem 1]). *Let $(C_\ell)_{\ell \in \mathbb{N}}$ be a family of codes $C_\ell : \{0,1\}^\ell \to \{0,1\}^L$ with minimal distance $c \cdot L$ for some constant $c \in (0, 1/2)$. Then, $(C_\ell)_{\ell \in \mathbb{N}}$ is a family of balanced admissible hash functions. Furthermore, $\mathsf{AdmSmp}(1^\lambda, Q, \delta)$ outputs a key $K \in \{0,1,\perp\}^L$ for which $\eta = \lfloor \frac{\ln(2Q+Q/\delta)}{-\ln((1-c))} \rfloor$ components are not $\perp$ and*

$$\gamma_{\max} = 2^{-\eta}, \qquad \gamma_{\min} = \big(1 - Q(1-c)\big)^\eta \cdot 2^{-\eta},$$

so that $\tau = (2\delta - (2\delta + 1) \cdot Q \cdot (1-c)^\eta)/2^{\eta+1}$ is a non-negligible function of λ.

Lemma 2.9 ([43, Lemma 8], [2, Lemma 28]). *Let an input space $\mathcal{X}$ and consider a mapping γ that maps a $(Q+1)$-tuple of elements $(X^\star, X_1, \ldots, X_Q)$ in $\mathcal{X}$ to a probability value in $[0,1]$. We consider the following experiment where we first execute the PRF security game, in which the adversary eventually outputs a guess $\hat{b} \in \{0,1\}$ of the challenger's bit $b \in \{0,1\}$ and wins with advantage ε. We denote by $X^\star \in \mathcal{X}$ the challenge input and $X_1, \ldots, X_Q \in \mathcal{X}$ the evaluation queries. At the end of the game, we flip a fair random coin $b'' \hookleftarrow U(\{0,1\})$. With probability $\gamma = \gamma(X^\star, X_1, \ldots, X_Q)$, we define $b' = \hat{b}$ and, with probability, $1 - \gamma$, we define $b' = b''$. Then, we have*

$$|\Pr[b' = b] - 1/2| \geq \gamma_{\min} \cdot \varepsilon - \frac{\gamma_{\max} - \gamma_{\min}}{2},$$

where $\gamma_{\min}$ and $\gamma_{\max}$ are the maximum and minimum of $\gamma(\mathbb{X})$ for any $\mathbb{X} \in \mathcal{X}^{Q+1}$.

2.3 (Deterministic) Randomness Extractors

A consequence of the Leftover Hash Lemma was used by Agrawal *et al.* [2] to re-randomize matrices over $\mathbb{Z}_q$ by multiplying them with small-norm matrices. We also rely on the following generalization of [2, Lemma 13].

Lemma 2.10. *Let integers m, n, ℓ such that $m > 2(n + \ell) \cdot \log q$, for some prime $q > 2$. Let $\mathbf{B}, \widetilde{\mathbf{B}} \hookleftarrow U(\mathbb{Z}_q^{m \times \ell})$ and $\mathbf{R} \hookleftarrow U(\{-1,1\}^{m \times m})$. For any matrix $\mathbf{F} \in \mathbb{Z}_q^{m \times n}$, the distributions $(\mathbf{B}, \mathbf{R} \cdot \mathbf{B}, \mathbf{R} \cdot \mathbf{F})$ and $(\mathbf{B}, \widetilde{\mathbf{B}}, \mathbf{R} \cdot \mathbf{F})$ are within $2^{-\Omega(n)}$ statistical distance.*

In our security proof, we will need to extract statistically uniform bits from a high-entropy source. Here, we cannot just apply the Leftover Hash Lemma since the source may not be independent of the seed. For this reason, we will apply techniques from deterministic extraction [26,60] and seeded extractors with seed-dependent sources [55]. In particular, we will apply a result of Dodis [26] which extends techniques due to Trevisan and Vadhan [60] to show that, for a sufficiently large $\xi > 0$, a fixed ξ-wise-independent functions can be used to deterministically extract statistically uniform bits.

Lemma 2.11 ([26, Corollary 3]). *Fix any integers $\bar{n}$, m, M, any real $\epsilon < 1$ and any collection $\mathcal{X}$ of M distributions over $\{0,1\}^{\bar{m}}$ of min-entropy $\bar{n}$ each. Define*

$$\xi = \bar{n} + \log M, \qquad \bar{k} = \bar{n} - \Big(2\log\frac{1}{\epsilon} + \log\log M + \log\bar{n} + O(1)\Big),$$

and let $\mathcal{F}$ be any family of ξ-wise independent functions from $\bar{m}$ bits to $\bar{k}$ bits. With probability at least $(1-1/M)$, a random function $f \hookleftarrow U(\mathcal{F})$ is a good deterministic extractor for the collection $\mathcal{X}$. Namely, $f(X)$ is ϵ-close to $U(\{0,1\}^{\bar{k}})$ for any distribution $X \in \mathcal{X}$.

It is well-known that ξ-wise independent function can be obtained by choosing random polynomials of degree $\xi - 1$ over $GF(2^{\bar{m}})$ (which cost $O(\xi\bar{m})$ bits to describe) and truncating their evaluations to their first $\bar{k}$ bits.

2.4 Linear Integer Secret Sharing

This section recalls the concept of linear integer secret sharing (LISS), as defined by Damgård and Thorbek [24]. The definitions below are taken from [59] where the secret to be shared lives in an interval $[-2^l, 2^l]$ centered in 0, for some $l \in \mathbb{N}$.

Definition 2.12. *A* **monotone** *access structure on $[N]$ is a non-empty collection $\mathbb{A}$ of sets $A \subseteq [N]$ such that $\emptyset \notin \mathbb{A}$ and, for all $A \in \mathbb{A}$ and all sets B such that $A \subseteq B \subseteq [N]$, we have $B \in \mathbb{A}$. For an integer $t \in [N]$, the* **threshold**-*t access structure $T_{t,N}$ is the collection of sets $A \subseteq [N]$ such that $|A| \geq t$.*

Let $P = [N]$ be a set of shareholders. In a LISS scheme, a dealer D wants to share a secret s in a publicly known interval $[-2^l, 2^l]$. To this end, D uses a share generating matrix $M \in \mathbb{Z}^{d \times e}$ and a random vector $\boldsymbol{\rho} = (s, \rho_2, \ldots, \rho_e)^\top$, where s is the secret to be shared $\{\rho_i\}_{i=2}^e$ are chosen uniformly in $[-2^{l_0+\lambda}, 2^{l_0+\lambda}]^e$, for a large enough $l_0 \in \mathbb{N}$. The dealer D computes a vector $\boldsymbol{s} = (s_1, \ldots, s_d)^\top$ of share units as

$$\boldsymbol{s} = (s_1, \ldots, s_d)^\top = M \cdot \boldsymbol{\rho} \ \in \mathbb{Z}^d.$$

Each party in $P = \{1, \ldots, N\}$ is assigned a set of share units. Letting $\psi : \{1, \ldots, d\} \to P$ be a surjective function, the i-th share unit s_i is assigned to the shareholder $\psi(i) \in P$, in which case player $\psi(i)$ is said to own the i-th row of M. If $A \subseteq P$ is a set of shareholders, $M_A \in \mathbb{Z}^{d_A \times e}$ denotes the set of rows jointly owned by A. Likewise, $\boldsymbol{s}_A \in \mathbb{Z}^{d_A}$ denotes the restriction of $\boldsymbol{s} \in \mathbb{Z}^d$ to the coordinates jointly owned by the parties in A. The j-th shareholder's share consists of $\boldsymbol{s}_{\psi^{-1}(j)} \in \mathbb{Z}^{d_j}$, so that it receives $d_j = |\psi^{-1}(j)|$ out of the $d = \sum_{j=1}^n d_j$ share units. The *expansion rate* $\mu = d/N$ is defined to be the average number of share units per player. Sets $A \in \mathbb{A}$ are called *qualified* and $A \notin \mathbb{A}$ are called *forbidden*.

Definition 2.13. *A LISS scheme is private if, for any two secrets s, s', any independent random coins $\boldsymbol{\rho} = (s, \rho_2, \ldots, \rho_e)$, $\boldsymbol{\rho}' = (s', \rho'_2, \ldots, \rho'_e)$ and any forbidden set A of shareholders, the distributions $\{s_i(s, \boldsymbol{\rho}) = M_i \cdot \boldsymbol{\rho} \mid i \in A\}$ and $\{s_i(s', \boldsymbol{\rho}') = M_i \cdot \boldsymbol{\rho}' \mid i \in A\}$ are $2^{-\Omega(\lambda)}$ apart in terms of statistical distance.*

Damgård and Thorbek [24] showed how to build LISS scheme from integer span programs [23].

Definition 2.14 ([23])**.** *An integer span program (ISP) is a tuple* $\mathcal{M} = (M, \psi, \boldsymbol{\varepsilon})$*, where* $M \in \mathbb{Z}^{d \times e}$ *is an integer matrix whose rows are labeled by a surjective function* $\psi : \{1, \ldots, d\} \to \{1, \ldots, N\}$ *and* $\boldsymbol{\varepsilon} = (1, 0, \ldots, 0)$ *is called target vector. The size of* $\mathcal{M}$ *is the number of rows* d *in* M*.*

Definition 2.15. *Let* Γ *be a monotone access structure and let* $\mathcal{M} = (M, \psi, \boldsymbol{\varepsilon})$ *an integer span program. Then,* $\mathcal{M}$ *is an ISP for* Γ *if it computes* Γ*: namely, for all* $A \subseteq \{1, \ldots, N\}$*, the following conditions hold:*

1. *If* $A \in \Gamma$*, there exists a reconstruction vector* $\boldsymbol{\lambda} \in \mathbb{Z}^{d_A}$ *such that* $\boldsymbol{\lambda}^\top \cdot M_A = \boldsymbol{\varepsilon}^\top$*.*
2. *If* $A \notin \Gamma$*, there exists* $\boldsymbol{\kappa} = (\kappa_1, \ldots, \kappa_e)^\top \in \mathbb{Z}^e$ *such that* $M_A \cdot \boldsymbol{\kappa} = \mathbf{0} \in \mathbb{Z}^d$ *and* $\boldsymbol{\kappa}^\top \cdot \boldsymbol{\varepsilon} = 1$ *(i.e.,* $\kappa_1 = 1$*). In this case,* $\boldsymbol{\kappa}$ *is called a sweeping vector for* A*.*

We also define $\kappa_{\max} = \max\{|a| \mid a \text{ is an entry in some sweeping vector}\}$*.*

Damgård and Thorbek showed [24] that, if we have an ISP $\mathcal{M} = (M, \psi, \boldsymbol{\varepsilon})$ that computes the access structure Γ, a statistically private LISS scheme for Γ can be obtained by using M as the share generating matrix and setting $l_0 = l + \lceil \log_2(\kappa_{\max}(e-1)) \rceil + 1$, where l is the length of the secret.

A LISS scheme $\mathcal{L} = (\mathcal{M} = (M, \psi, \boldsymbol{\varepsilon}), \Gamma, \mathcal{R}, \mathcal{K})$ is thus specified by an ISP for the access structure Γ, a space $\mathcal{R}$ of reconstruction vectors satisfying Condition 1 of Definition 2.15, and a space $\mathcal{K}$ of sweeping vectors satisfying Condition 2.

Lemma 2.16 ([59, Lemma 3.1])**.** *Let* $l_0 = l + \lceil \log_2(\kappa_{\max}(e-1)) \rceil + 1$*. If* $s \in [-2^l, 2^l]$ *is the secret to be shared and* $\boldsymbol{\rho}$ *is randomly sampled from* $[-2^{l_0+\lambda}, 2^{l_0+\lambda}]^e$ *conditionally on* $\langle \boldsymbol{\rho}, \boldsymbol{\varepsilon} \rangle = s$*, the LISS scheme derived from* $\mathcal{M}$ *is private. For any arbitrary* $s, s' \in [-2^l, 2^l]$ *and any forbidden set of shareholders* $A \subset [N]$*, the two distributions* $\{s_A = M_A \cdot \boldsymbol{\rho} \mid \boldsymbol{\rho} \hookleftarrow U([-2^{l_0+\lambda}, 2^{l_0+\lambda}]^e) \text{ s.t. } \langle \boldsymbol{\rho}, \boldsymbol{\varepsilon} \rangle = s\}$*, and* $\{s'_A = M_A \cdot \boldsymbol{\rho} \mid \boldsymbol{\rho} \hookleftarrow U([-2^{l_0+\lambda}, 2^{l_0+\lambda}]^e) \text{ s.t. } \langle \boldsymbol{\rho}, \boldsymbol{\varepsilon} \rangle = s'\}$ *are within statistical distance* $2^{-\lambda}$*.*

In the following, we do not rely on the result of Lemma 2.16 as we will share vectors sampled from Gaussian (instead of uniform) distributions using Gaussian random coins. We also depart from Lemma 2.16 in that the random coins $(\rho_2, \ldots, \rho_e)$ are not sampled from a wider distribution than the secret: the standard deviation of $(\rho_2, \ldots, \rho_e)$ will be the same as that of s. While this choice does not guarantee the LISS to be private in general, we will show that it suffices in our setting because we only need the secret to have sufficient min-entropy conditionally on the shares observed by the adversary. Aside from the distribution of secrets and random coins, we rely on the technique of Damgård and Thorbek [24] for building share generating matrices.

It was shown in [24] that LISS schemes can be obtained from [10,23]. While the Benaloh-Leichter (BL) secret sharing [10] was initially designed to work over finite groups, Damgård and Thorbek generalized it [24] so as to share integers using access structures consisting of any monotone Boolean formula. In turn, this implies a LISS scheme for any threshold access structure by applying a result of Valiant [34,61]. Their LISS scheme built upon Benaloh-Leichter [10] comes in handy for our purposes because the reconstruction coefficients and the sweeping vectors are small: as can be observed from [24, Lemmas 4], the entries of $\boldsymbol{\lambda}$ live in $\{-1, 0, 1\}$ and [24, Lemma 5] shows that $\kappa_{\max} = 1$. For a monotone Boolean f, the BL-based technique allows binary share distribution matrices $M \in \{0,1\}^{d \times e}$ such that $d, e = O(\text{size}(f))$ and which have at most $\text{depth}(f) + 1$ non-zero entries, so that each share unit s_i has magnitude $O(2^{l_0+\lambda} \cdot \text{depth}(f))$.

Valiant's result [61] implies the existence of a monotone Boolean formula of the threshold-t function $T_{t,N}$, which has size $d = O(N^{5.3})$ and depth $O(\log N)$. Since each player receives d/N rows of M on average, the average share size is thus $O(N^{4.3} \cdot (l_0 + \lambda + \log\log N))$ bits. Valiant's construction was improved by Hoory *et al.* [40] who give a monotone formula of size $O(N^{1+\sqrt{2}})$ and depth $O(\log N)$ for the majority function.[2] This reduces the average share size to $O(N^{\sqrt{2}} \cdot (l_0 + \lambda + \log\log N))$ bits.

2.5 Some Useful Lemmas

Lemma 2.17 **([49, Lemma 4.4]).** *For $\sigma = \omega(\sqrt{\log n})$ there is a negligible function $\epsilon = \epsilon(n)$ such that:*

$$\Pr_{\mathbf{x} \sim D_{\mathbb{Z}^n,\sigma}} \left[\|\mathbf{x}\| > \sigma\sqrt{n}\right] \leq \frac{1+\epsilon}{1-\epsilon} \cdot 2^{-n}$$

Lemma 2.18 **([6, Lemma 2.7]).** *Let p, q be positive integers such that $p < q$. Given $R > 0$ an integer, the probability that there exists $e \in [-R, R]$ such that $\lfloor y \rfloor_p \neq \lfloor y + e \rfloor_p$, when $y \hookleftarrow U(\mathbb{Z}_q)$, is smaller than $\frac{2Rp}{q}$.*

Lemma 2.19. *If q is prime and $\mathcal{M}$ be a distribution over $\mathbb{Z}_q^{m \times n}$, and V a distribution over $\mathbb{Z}_q^n$ such that $\Delta\left(\mathcal{M}, U(\mathbb{Z}_q^{m \times n})\right) \leq \epsilon$. We have $\Delta\left(\mathcal{M} \cdot V, U(\mathbb{Z}_q^m)\right) \leq \epsilon + \alpha \cdot \left(1 - \frac{1}{q^m}\right)$, where $\alpha := \Pr[V = \mathbf{0}]$.*

2.6 (Distributed) Pseudorandom Functions

A pseudorandom function family is specified by efficient algorithms $(\mathsf{Keygen}, \mathsf{Eval})$, where Keygen a randomized key generation algorithm that takes in a security parameter 1^λ and outputs a random key $K \hookleftarrow \mathcal{K}$ from a key space $\mathcal{K}$. Eval is a deterministic evaluation algorithm, which takes in a key $K \in \mathcal{K}$ and an input X in a domain $\mathcal{D} = \{0,1\}^\ell$ and evaluates a function $F(K, X)$ in a range $\mathcal{R} = \{0,1\}^\mu$. The standard security definitions for PRFs are recalled in the full version of the paper.

[2] Note that a threshold-t function can be obtained from the majority function by fixing the desired number of input bits, so that we need a majority function of size $\leq 2N$ to construct a threshold function $T_{t,N}$.

A distributed pseudorandom function (DPRF) is a tuple of algorithms $(\mathsf{Setup}, \mathsf{Share}, \mathsf{PEval}, \mathsf{Eval}, \mathsf{Combine})$ of efficient algorithms with the following specification. Setup takes as input a security parameter 1^λ, a number of servers 1^N, a threshold 1^t and a desired input length 1^ℓ and outputs public parameters pp. The key sharing algorithm $\mathsf{Share} : \mathcal{K} \to \mathcal{K}^N$ inputs a random master secret key $SK_0 \in \mathcal{K}$ and outputs a tuple of shares $(SK_1, \ldots, SK_N) \in \mathcal{K}^N$, which form a (t, N)-threshold secret sharing of SK_0. The partial evaluation algorithm $\mathsf{Eval} : \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ takes as input a key share SK_i and an input X and outputs a partial evaluation $Y_i = \mathsf{PEval}(SK_i, X) \in \mathcal{R}$. Algorithm $\mathsf{Combine} : \mathcal{S} \times \mathcal{R}^t \to \mathcal{R}$ takes in a t-subset $\mathcal{S} \subset [N]$ together with t partial evaluations $\{Y_i\}_{i \in \mathcal{S}}$, where $Y_i \in \mathcal{R}$ for all $i \in \mathcal{S}$, and outputs a value $Y \in \mathcal{R}$. The centralized evaluation algorithm $\mathsf{Eval} : \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ operates as in a ordinary PRF and outputs a value $Y = \mathsf{Eval}(SK_0, X) \in \mathcal{R}$ on input of $X \in \mathcal{D}$ and a key $SK_0 \in \mathcal{K}$.

Consistency. A DPRF is consistent if, for any $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^t, 1^N)$, any master key $SK_0 \hookleftarrow \mathcal{K}$ shared according to $(SK_1, \ldots, SK_N) \leftarrow \mathsf{Share}(SK_0)$, any t-subset $\mathcal{S} = \{i_1, \ldots, i_t\} \subset [N]$ and any input $X \in \mathcal{D}$, if $Y_{i_j} = \mathsf{PEval}(SK_{i_j}, X)$ for each $j \in [t]$, then we have $\mathsf{Eval}(SK_0, X) = \mathsf{Combine}(\mathcal{S}, (Y_{i_1}, \ldots, Y_{i_t}))$ with overwhelming probability over the random coins of Setup and Share.

We say that a DPRF provides adaptive security if it remains secure against an adversary that can adaptively choose which servers it wants to corrupt. In particular, the adversary can arbitrarily interleave evaluation and corruption queries as long as they do not allow it to trivially win.

Definition 2.20 (Adaptive DPRF security). *Let λ be a security parameter and let integers $t, N \in \mathsf{poly}(\lambda)$. We say that a (t, N)-DPRF is pseudorandom under adaptive corruptions if no PPT adversary has non-negligible advantage in the following game:*

1. *The challenger generates $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^t, 1^N)$ and chooses a random key $SK_0 \hookleftarrow \mathcal{K}$, which is broken into N shares $(SK_1, \ldots, SK_N) \leftarrow \mathsf{Share}(SK_0)$. It also initializes an empty set $\mathcal{C} \leftarrow \emptyset$ and flips a random coin $b \hookleftarrow U(\{0,1\})$.*
2. *The adversary $\mathcal{A}$ adaptively interleaves the following kinds of queries.*
   Corruption: *$\mathcal{A}$ chooses an index $i \in [N] \backslash \mathcal{C}$. The challenger returns SK_i to $\mathcal{A}$ and sets $\mathcal{C} := \mathcal{C} \cup \{i\}$.*
   Evaluation: *$\mathcal{A}$ chooses a pair $(i, X) \in [N] \times \mathcal{D}$ and the challenger returns $Y_i = \mathsf{PEval}(SK_i, X)$.*
3. *The adversary chooses an input $X^\star$. At this point, the challenger randomly samples $Y_0 \hookleftarrow U(\{0,1\}^\mu)$ and computes $Y_1 = \mathsf{Eval}(SK_0, X^\star)$. Then, it returns Y_b to the adversary.*
4. *The adversary $\mathcal{A}$ adaptively makes more queries as in Stage 2 under the restriction that, at any time, we should have $|\mathcal{C} \cup \mathcal{E}| < t$, where $\mathcal{E} \subset [N]$ denotes the set of indexes for which an evaluation query of the form $(i, X^\star)$ was made in Stage 2 or in Stage 4.*
5. *$\mathcal{A}$ outputs a bit $\hat{b} \in \{0,1\}$ and wins if $\hat{b} = b$. Its advantage is defined to be $\mathbf{Adv}_{\mathcal{A}}^{\mathrm{DPRF}}(\lambda) := |\Pr[\hat{b} = b] - 1/2|$.*

Definition 2.20 is a game based definition, which may not imply security in the sense of simulation-based definitions. Still, we show it is strictly stronger than the definition of static security used in [16]. It is well-known that static security does not imply adaptive security in distributed threshold protocols (see, e.g., [22]). In the case of DPRFs, we show in the full version of the paper that allowing even a *single* evaluation query before any corruption query already gives a stronger game-based definition than the game-based security definition of static security.

Theorem 2.21. *For any $t, N \in \mathsf{poly}(\lambda)$ such that $t < N/2$, there is a DPRF family which is secure in the sense of Definition A.1 (in Appendix A) but insecure in the sense of Definition 2.20.*

Note that the above separation still holds for small non-constant values of t and N if we assume polynomial or slightly super-polynomial adversaries.

3 A Variant of the BLMR PRF

Before describing our distributed PRF, we present its centralized version which can be seen as a variant of the key-homomorphic PRFs described by Boneh *et al.* [16] and Banerjee-Peikert PRFs [7]. However, the security proof is very different in that it does not use a hybrid argument over the input bits. Instead, it applies the strategy of partitioning the input space into disjoint subspaces (analogously to proof techniques for, e.g., identity-based encryption [63]) and builds on the lossy mode of LWE [36].

In [7,16], a PRF evaluation of an input x is of the form $\mathbf{y} = \lfloor \mathbf{A}(x)^\top \cdot \mathbf{s} \rfloor_p \in \mathbb{Z}_p^m$, where $\mathbf{s} \in \mathbb{Z}_q^n$ is the secret key and $\mathbf{A}(X) \in \mathbb{Z}_q^{n \times m}$ is an input-dependent matrix obtained from public matrices $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times m}$. Our variant is similar at a high level, with two differences. First, we derive $\mathbf{A}(x)$ from a set of $2L$ public matrices $\{\mathbf{A}_{i,0}, \mathbf{A}_{i,1}\}_{i=1}^L$. Second, $\lfloor \mathbf{A}(x)^\top \cdot \mathbf{s} \rfloor_p$ is not quite our PRF evaluation. Instead, we obtain the PRF value by using $\lfloor \mathbf{A}(x)^\top \cdot \mathbf{s} \rfloor_p$ as a source of entropy for a deterministic randomness extractor.

The security proof departs from [7,16] by exploiting the connection between the schemes and the Gentry-Sahai-Waters FHE [33]. For each $i \in [L]$ and $b \in \{0,1\}$, we interpret $\mathbf{A}_{i,b} \in \mathbb{Z}_q^{n \times m}$ as a GSW ciphertext $\mathbf{A}_{i,b} = \mathbf{A} \cdot \mathbf{R}_{i,b} + \mu_{i,b} \cdot \mathbf{G}$, where $\mathbf{R}_{i,b} \in \{-1,1\}^{m \times m}$, $\mu_{i,b} \in \{0,1\}$ and $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ is the gadget matrix of [48]. Before evaluating the PRF on an input X, we encode $X \in \{0,1\}^\ell$ into $x \in \{0,1\}^L$ using an admissible hash function. Then, we homomorphically derive $\mathbf{A}(x)$ as a GSW ciphertext $\mathbf{A}(x) = \mathbf{A} \cdot \mathbf{R}_x + (\prod_{i=1}^L \mu_{i,x[i]}) \cdot \mathbf{G}$, for some small-norm $\mathbf{R}_x \in \mathbb{Z}^{m \times m}$. By carefully choosing $\{\mu_{i,b}\}_{i \in [L], b \in \{0,1\}}$, the properties of admissible hash functions ensure that the product $\prod_{i=1}^L \mu_{i,x[i]}$ cancels out in all evaluation queries but evaluates to 1 on the challenge input $X^\star$.

In the next step of the proof, we move to a modified experiment where the random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ is replaced by a lossy matrix $\mathbf{A}^\top = \bar{\mathbf{A}}^\top \cdot \mathbf{C} + \mathbf{E}$, where $\bar{\mathbf{A}} \hookleftarrow U(\mathbb{Z}_q^{n' \times m})$, $\mathbf{C} \hookleftarrow U(\mathbb{Z}_q^{n' \times n})$ and $\mathbf{E} \in \mathbb{Z}^{m \times n}$ is a short Gaussian matrix. This modification has the consequence of turning $\lfloor \mathbf{A}(x)^\top \cdot \mathbf{s} \rfloor_p$ into a lossy function of $\mathbf{s}$ on all inputs X for which $\prod_{i=1}^{L} \mu_{i,x[i]} = 0$. At the same time, the function remains injective whenever $\prod_{i=1}^{L} \mu_{i,x[i]} = 1$. Using the properties of admissible hash functions, we still have a noticeable probability that the function be lossy in all evaluation queries and injective in the challenge phase. Moreover, by using a small-norm secret $\mathbf{s} \in \mathbb{Z}^n$ and setting the ratio q/p large enough, we can actually make sure that evaluation queries always reveal the same information (namely, the product $\mathbf{C} \cdot \mathbf{s}$) about $\mathbf{s}$. As long as we have $\prod_{i=1}^{L} \mu_{i,x^\star[i]} = 1$ for the challenge input $X^\star$, the value $\tilde{\mathbf{z}} = \lfloor \mathbf{A}(x^\star)^\top \cdot \mathbf{s} \rfloor_p = \lfloor (\mathbf{A} \cdot \mathbf{R}_{x^\star} + \mathbf{G})^\top \cdot \mathbf{s} \rfloor_p$ is guaranteed to have a lot of entropy as an injective function of an unpredictable $\mathbf{s}$. At this point, we can extract statistically uniform bits from the source $\tilde{\mathbf{z}}$. Since the latter depends on $x^\star$ (which can be correlated with the seed included in public parameters), we need an extractor that can operate on seed-dependent sources. Fortunately, deterministic extractors come in handy for this purpose.

3.1 Decomposing Random Matrices into Invertible Binary Matrices

In the following, we set $k = n\lceil \log q \rceil$ and $m = 2k$ and define

$$\mathbf{G} = [\ \mathbf{I}_n \otimes (1, 2, 4, \ldots, 2^{\lceil \log q \rceil - 1}) \mid \ \mathbf{I}_n \otimes (1, 2, 4, \ldots, 2^{\lceil \log q \rceil - 1})] \ \in \mathbb{Z}_q^{n \times m}$$

which is a variant of the gadget matrix of [48]. We also define $\mathbf{G}^{-1} : \mathbb{Z}_q^{n \times m} \to \mathbb{Z}^{m \times m}$ to be a deterministic algorithm that inputs $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and outputs a binary matrix $\mathbf{G}^{-1}(\mathbf{A}) \in \{0,1\}^{m \times m}$ such that $\mathbf{G} \cdot \mathbf{G}^{-1}(\mathbf{A}) = \mathbf{A}$. We will require that, for any $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{G}^{-1}(\mathbf{A})$ be invertible over $\mathbb{Z}_q$ with sufficiently high probability. The next lemma shows a function $\mathbf{G}^{-1}(\cdot)$ satisfying this condition.

Lemma 3.1 (Adapted from [16, Lemma A.3]**).** *Let $k = n\lceil \log q \rceil$. If $q \geq 2^{k/n} \cdot (1 - \frac{1}{2n})$, there exists an efficient algorithm that samples a statistically uniform matrix $\mathbf{A} \hookleftarrow U(\mathbb{Z}_q^{n \times m})$ such that $\mathbf{G}^{-1}(\mathbf{A}) \in \{0,1\}^{m \times m}$ is $\mathbb{Z}_q$-invertible.*

Proof. We first show how to sample a sequence of $k = n\lceil \log q \rceil$ uniform vectors over $\mathbb{Z}_q^n$ whose binary decompositions form a full-rank binary matrix over $\mathbb{Z}_q$. In turn, this will allow us to sample a random $\mathbf{A} \hookleftarrow U(\mathbb{Z}_q^{n \times m})$, where $m = 2k$, such that $\mathbf{G}^{-1}(\mathbf{A}) \in \{0,1\}^{m \times m}$ is invertible. As in the proof of [16, Lemma A.3], we use the observation that, for any i linearly independent vectors $\mathbf{v}_1, \ldots, \mathbf{v}_i \in \mathbb{Z}_q^k$ over $\mathbb{Z}_q$, if $V = \mathrm{span}_{\mathbb{Z}_q}(\mathbf{v}_1, \ldots, \mathbf{v}_i)$, we have $|V \cap \{0,1\}^k| \leq 2^i$.

For an index $i \in [k-1]$, suppose that we have chosen $\mathbb{Z}_q$-independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_i \in \{0,1\}^k$ and that $\mathbf{b}_{i+1} \in \{0,1\}^k$ is obtained as the binary decomposition of a random $\mathbf{a}_{i+1} \hookleftarrow U(\mathbb{Z}_q^n)$. The probability that $\mathbf{b}_{i+1}$ is independent of $\mathbf{b}_1, \ldots, \mathbf{b}_i$ is $\geq (q^n - 2^i)/q^n$. If we sample $\mathbf{a}_1, \ldots, \mathbf{a}_k \hookleftarrow U(\mathbb{Z}_q^n)$, the probability that their binary decompositions are linearly independent over $\mathbb{Z}_q$ is

$$\prod_{i=0}^{k-1} \Pr[\mathbf{b}_{i+1} \notin \mathrm{span}_{\mathbb{Z}_q}(\mathbf{b}_1, \ldots, \mathbf{b}_i)] = \prod_{i=0}^{k-1} \frac{q^n - 2^i}{q^n} \tag{1}$$

Note that the factors the right-hand-side member of (1) are all positive: indeed, we have $2^{k/n} \cdot (1 - \frac{1}{2n}) \leq q \leq 2^{k/n}$. Since $(1 - \frac{1}{2n})^n \approx \exp(-1/2)$ for large values of n, this implies $2^k/\sqrt{\exp(1)} \leq q^n \leq 2^k$ and thus $2^{k-1}/q^n \leq \sqrt{\exp(1)}/2 < 1$.

We now proceed to bound (1) as

$$\prod_{i=0}^{k-1}\Big(1-\frac{2^i}{q^n}\Big) = \exp\Big(\sum_{i=0}^{k-1}\ln\big(1-\frac{2^i}{q^n}\big)\Big)$$
$$\geq \exp\Big(-\frac{3}{q^n}\cdot\sum_{i=0}^{k-1}2^i\Big) = \exp\Big(-\frac{3}{q^n}\cdot 2^k\Big) = \exp(-3\cdot\sqrt{\exp(1)}),$$

where the inequality holds because $\ln(1-x) \geq -3x$ for all $x \in \big(0, \sqrt{\exp(1)}/2\big)$.

Hence, if we sample $141 \cdot k > k/\exp(-3\cdot\sqrt{\exp(1)})$ vectors $\mathbf{a}_i \hookleftarrow U(\mathbb{Z}_q^n)$ and stack up the binary decompositions of $\mathbf{a}_i^\top$, the probability that the resulting matrix contains a $\mathbb{Z}_q$-invertible sub-matrix over $\{0,1\}^k$ is at least $1-2^{-\Omega(k)}$.

We can thus sample a random matrix $\mathbf{A} = [\mathbf{A}_L|\mathbf{A}_R] \hookleftarrow U(\mathbb{Z}_q^{n\times m})$ that satisfies the required conditions by defining $\mathbf{G}^{-1}(\mathbf{A}) \in \{0,1\}^{m\times m}$ so that it contains the binary decomposition $\mathsf{BD}(\mathbf{A}_L) \in \{0,1\}^{k\times k}$ in its upper-left corner and $\mathsf{BD}(\mathbf{A}_R) \in \{0,1\}^{k\times k}$ in its lower-right corner. □

3.2 A Centralized Construction

Let λ be a security parameter and let $\ell \in \Theta(\lambda)$, $L \in \Theta(\lambda)$. We use parameters consisting of prime moduli p and q such that $q/p > 2^{L+\lambda}\cdot r$, dimensions $n, m, k \in \mathsf{poly}(\lambda)$ such that $m \geq 2n\cdot\lceil\log q\rceil$, an integer $\beta > 0$, $\alpha > 0$ and $r = m^{L+2}\cdot n\cdot\beta\cdot\alpha q$. We rely on the following ingredients.

- A balanced admissible hash function $\mathsf{AHF} : \{0,1\}^\ell \to \{0,1\}^L$.
- A family Π_λ of ξ-wise independent hash functions $\pi_i : \mathbb{Z}_p^m \to \mathbb{Z}_p^k$ for a suitable $\xi > 0$ that will be determined later on. Let a random member π of Π_λ. For example, the function π can be a random polynomial $\pi(Z) \in GF(p^m)[Z]$ of degree $\xi - 1$ with outputs truncated to their k first coordinates.

We also choose a Gaussian parameter $\sigma > 0$, which will specify an interval $[-\beta, \beta] = [-\sigma\sqrt{n}, \sigma\sqrt{n}]$ where the coordinates of the secret will be confined (with probability exponentially close to 1). We also need a rounding parameter $r > 0$, set as indicated above.

The pseudorandom function family assumes the availability of public parameters

$$\mathsf{pp} := \Big(q,\ \pi,\ \mathbf{A}_0,\ \{\mathbf{A}_{i,0}, \mathbf{A}_{i,1}\ \in \mathbb{Z}_q^{n\times m}\ \}_{i=1}^L,\ \mathsf{AHF},\ r,\ \sigma\Big),$$

where $\mathbf{A}_0 \sim U(\mathbb{Z}_q^{n\times m})$ and $\mathbf{A}_{i,0}, \mathbf{A}_{i,1} \sim U(\mathbb{Z}_q^{n\times m})$ for each $i \in [L]$. Importantly, $\{\mathbf{A}_{i,0}, \mathbf{A}_{i,1}\}_{i=1}^L$ should be chosen in such a way that $\mathbf{G}^{-1}(\mathbf{A}_{i,b}) \in \mathbb{Z}^{m\times m}$ is $\mathbb{Z}_q$-invertible for all $i \in [L]$ and $b \in \{0,1\}$.

Keygen(pp): Given pp, sample a vector $\mathbf{s} \hookleftarrow D_{\mathbb{Z}^n,\sigma}$ so that $\|\mathbf{s}\|_\infty < \beta = \sigma\sqrt{n}$ with overwhelming probability. The secret key is $SK := \mathbf{s} \in [-\beta, \beta]^n$.

Eval(pp, SK, X)**:** Given $SK = \mathbf{s} \in \mathbb{Z}^n$ and an input $X \in \{0,1\}^\ell$,

1. Compute $x = \mathsf{AHF}(X) \in \{0,1\}^L$ and parse it as $x = x_1 \dots x_L$.
2. Compute

$$\mathbf{z} = \left\lfloor \big(\mathbf{A}(x)\big)^\top \cdot \mathbf{s} \right\rfloor_p \in \mathbb{Z}_p^m, \tag{2}$$

where

$$\mathbf{A}(x) = \mathbf{A}_0 \cdot \prod_{i=1}^{L} \mathbf{G}^{-1}\big(\mathbf{A}_{i,x_i}\big),$$

and output $\mathbf{y} = \pi(\mathbf{z}) \in \mathbb{Z}_p^k$.

We remark that the way to compute $\mathbf{z} \in \mathbb{Z}_p^m$ in (2) is reminiscent of the key-homomorphic PRFs of [7,16]. Unlike [7,16], our security proof requires the secret $\mathbf{s}$ to have small entries. Also, our PRF is not key-homomorphic as the output is $\mathbf{y} = \pi(\mathbf{z}) \in \mathbb{Z}_p^k$ instead of $\mathbf{z} \in \mathbb{Z}_p^m$. Fortunately, losing the key-homomorphic property does not prevent us from building a DPRF since the randomness extraction step is only applied to the result of combining t partial evaluations.

Theorem 3.2. *Set an entropy lower bound $\bar{n} = \lfloor n \cdot \log \sigma - n' \cdot \log q \rfloor - 1$ as $\Omega(\lambda)$. If we choose the output length $\bar{k} = k \cdot \log p$ in such a way that*

$$\xi = \bar{n} + \ell, \qquad \bar{k} = \bar{n} - 2 \cdot (\lambda + \log \ell + \log \bar{n}),$$

then the construction above is a secure PRF family under the $\mathsf{LWE}_{q,m,n',\alpha}$ assumption.

The proof is given in the full version of the paper. It may be inferred as a sub-proof of the security proof of the upcoming DPRF construction.

4 The DPRF Construction

We design the distributed PRF by using a LISS inside the PRF construction of Sect. 3. As mentioned earlier, the latter is well-suited to our purposes because, in the security proof, the secret key is known to the challenger at any time. When the secret key $\mathbf{s}$ is shared using a LISS, the challenger is always able to consistently answer corruption queries because it has all shares at disposal.

In the construction, we rely on the specific LISS construction of Damgård and Thorbek [24], which is based on the Benaloh-Leichter secret sharing [10]. This particular LISS scheme is well-suited to our needs for several reasons. First, it has binary share generating matrices, which allows obtaining relatively short shares of $\mathbf{s} \in \mathbb{Z}^n$: in the security proof, this is necessary to ensure that the adversary always obtains the same information about uncorrupted shares in partial evaluation queries. Another advantage of the Benaloh-Leichter-based LISS is that its reconstruction constants live in $\{-1, 0, 1\}$, which avoids blowing up the homomorphism errors when partial evaluations are combined together. Finally, its sweeping vectors also have their coordinates in $\{-1, 0, 1\}$ (whereas they may be exponentially large in the number N of servers in the construction based on Cramer-Fehr [23]) and we precisely need sweeping vectors $\boldsymbol{\kappa}$ to be small in the proof of our Lemma 4.4.

4.1 Description

Setup$(1^\lambda, 1^\ell, 1^t, 1^N)$**:** On input of a security parameter λ, a number of servers N, a threshold $t \in [1, N]$ and an input length $\ell \in \Theta(\lambda)$, set $d, e = O(N^{1+\sqrt{2}})$. Then, choose a real $\alpha > 0$, a Gaussian parameter $\sigma = \sqrt{e} \cdot \Omega(\sqrt{n})$, which will specify an interval $[-\beta, \beta] = [-\sigma\sqrt{n}, \sigma\sqrt{n}]$ where the coordinates of the secret will live (with probability exponentially close to 1). Next, do the following.

1. Choose prime moduli p, q and u such that $p/u > d \cdot 2^{\lambda+L}$ and $q/p > 2^{L+\lambda} \cdot r$, where dimensions $n, m, k \in \mathsf{poly}(\lambda)$ such that $m \geq 2n \cdot \lceil \log q \rceil$, and $r = m^{L+2} \cdot n \cdot \beta^* \cdot \alpha q$ with $\beta^* = O(\beta \cdot \log N)$.
2. Choose a balanced admissible hash function $\mathsf{AHF} : \{0,1\}^\ell \to \{0,1\}^L$, for a suitable $L \in \Theta(\lambda)$. Choose a family Π_λ of ξ-wise independent hash functions $\pi_i : \mathbb{Z}_u^m \to \mathbb{Z}_u^k$, for a suitable integer $\xi > 0$, with $\pi \hookleftarrow U(\Pi_\lambda)$.
3. Choose random matrices $\mathbf{A}_0 \hookleftarrow U(\mathbb{Z}_q^{n\times m})$ and $\mathbf{A}_{i,b} \hookleftarrow U(\mathbb{Z}_q^{n\times m})$, for each $i \in [L]$, $b \in \{0,1\}$, subject to the constraint that $\mathbf{G}^{-1}(\mathbf{A}_{i,b}) \in \mathbb{Z}^{m\times m}$ be $\mathbb{Z}_q$-invertible for all $i \in [L]$ and $b \in \{0,1\}$.

Output

$$\mathsf{pp} := \Big(q,\ p,\ u,\ \pi,\ \mathbf{A}_0,\ \{\mathbf{A}_{i,0}, \mathbf{A}_{i,1}\ \in \mathbb{Z}_q^{n\times m}\ \}_{i=1}^L,\ \mathsf{AHF}\Big),$$

Share(pp, SK_0)**:** Given pp and a key $SK_0 = \mathbf{s}$ consisting of an integer vector $\mathbf{s}$ sampled from the Gaussian distribution $D_{\mathbb{Z}^n,\sigma}$, return $\perp$ if $\mathbf{s} \notin [-\beta, \beta]^n$, where $\beta = \sigma\sqrt{n}$. Otherwise, generate a LISS of $\mathbf{s}$ as follows.

1. Using the BL-based LISS scheme, construct the matrix $M \in \{0,1\}^{d\times e}$ that computes the Boolean formula associated with the $T_{t,N}$ threshold function. By using [40], we obtain a matrix $M \in \{0,1\}^{d\times e}$, so that each row of M contains $O(\log N)$ non-zero entries.
2. For each $k \in [n]$, generate a LISS of the k-th coordinate s_k of $\mathbf{s} \in \mathbb{Z}^n$. To this end, define a vector $\boldsymbol{\rho}_k = (s_k, \rho_{k,2}, \ldots, \rho_{k,e})^\top$, with Gaussian entries $\rho_{k,2}, \ldots, \rho_{k,e} \hookleftarrow D_{\mathbb{Z},\sigma}$, and compute
$$\mathbf{s}_k = (s_{k,1}, \ldots, s_{k,d})^\top = M \cdot \boldsymbol{\rho}_k \quad \in \mathbb{Z}^d,$$
whose entries are smaller than $\|\mathbf{s}_k\|_\infty \leq \beta^* = O(\beta \cdot \log N)$.
3. Define the matrix $\mathbf{S} = [\mathbf{s}_1 \mid \ldots \mid \mathbf{s}_n] \in \mathbb{Z}^{d\times n}$. For each $j \in [N]$, define the share of server P_j to be the sub-matrix $\mathbf{S}_{I_j} = M_{I_j} \cdot [\boldsymbol{\rho}_1 \mid \ldots \mid \boldsymbol{\rho}_n] \in \mathbb{Z}^{d_j\times n}$, where $I_j = \psi^{-1}(j) \subset \{1, \ldots, d\}$ is the set of indexes such that P_j owns the sub-matrix $M_{I_j} \in \{0,1\}^{d_j\times e}$.

For each $j \in [N]$, the share $SK_j = \mathbf{S}_{I_j} \in \mathbb{Z}^{d_j \times n}$ is privately sent to P_j.

PEval(pp, SK_j, X)**:** Given $SK_j = \mathbf{S}_{I_j} \in \mathbb{Z}^{d_j \times n}$ and an input $X \in \{0,1\}^\ell$,

1. Compute $x = \mathsf{AHF}(X) \in \{0,1\}^L$ and parse it as $x = x_1 \ldots x_L$.
2. Parse $\mathbf{S}_{I_j}^\top = [\boldsymbol{\rho}_1 \mid \ldots \mid \boldsymbol{\rho}_n]^\top \cdot M_{I_j}^\top \in \mathbb{Z}^{n \times d_j}$ as $[\bar{\mathbf{s}}_{j,1} \mid \ldots \mid \bar{\mathbf{s}}_{j,d_j}]$. For each $\theta \in \{1, \ldots, d_j\}$, compute

$$\mathbf{z}_{j,\theta} = \left\lfloor \big(\mathbf{A}(x)\big)^\top \cdot \bar{\mathbf{s}}_{j,\theta} \right\rfloor_p \; \in \mathbb{Z}_p^m, \tag{3}$$

where

$$\mathbf{A}(x) = \mathbf{A}_0 \cdot \prod_{i=1}^{L} \mathbf{G}^{-1}\big(\mathbf{A}_{i,x_i}\big),$$

and output the partial evaluation $\mathbf{Y}_j = [\mathbf{z}_{j,1} \mid \ldots \mid \mathbf{z}_{j,d_j}] \in \mathbb{Z}_p^{m \times d_j}$.

Eval(pp, SK_0, X)**:** Given $SK_0 = \mathbf{s} \in \mathbb{Z}^n$ and an input $X \in \{0,1\}^\ell$,

1. Compute $x = \mathsf{AHF}(X) \in \{0,1\}^L$ and write it as $x = x_1 \ldots x_L$.
2. Compute

$$\tilde{\mathbf{z}} = \left\lfloor \big(\mathbf{A}(x)\big)^\top \cdot \mathbf{s} \right\rfloor_p \; \in \mathbb{Z}_p^m,$$

where $\mathbf{A}(x) = \mathbf{A}_0 \cdot \prod_{i=1}^{L} \mathbf{G}^{-1}\big(\mathbf{A}_{i,x_i}\big)$, and output $\mathbf{y} = \pi(\lfloor \tilde{\mathbf{z}} \rfloor_u) \in \mathbb{Z}_u^k$.

Combine$(\mathcal{S}, (\mathbf{Y}_{j_1}, \ldots, \mathbf{Y}_{j_t}))$**:** Write $\mathcal{S} = \{j_1, \ldots, j_t\}$ and parse each $\mathbf{Y}_{j_\kappa} \in \mathbb{Z}_p^{m \times d_{j_\kappa}}$ as $[\mathbf{z}_{j_\kappa,1} \mid \ldots \mid \mathbf{z}_{j_\kappa,d_{j_\kappa}}]$ for all $\kappa \in [t]$.

1. Determine the vector $\boldsymbol{\lambda}_{\mathcal{S}} \in \{-1,0,1\}^{d_{\mathcal{S}}}$ such that $\boldsymbol{\lambda}_{\mathcal{S}}^\top \cdot M_{\mathcal{S}} = (1,0,\ldots,0)^\top$, where $M_{\mathcal{S}} \in \{0,1\}^{d_{\mathcal{S}} \times e}$ is the sub-matrix of M owned by the parties in $\mathcal{S}$ and $d_{\mathcal{S}} = \sum_{\kappa=1}^{t} d_{j_\kappa}$ with $d_{j_\kappa} = |\psi^{-1}(j_\kappa)|$ for all $\kappa \in [t]$. Then, parse $\boldsymbol{\lambda}_{\mathcal{S}}$ as $[\boldsymbol{\lambda}_{j_1}^\top \mid \ldots \mid \boldsymbol{\lambda}_{j_t}^\top]^\top$, where $\boldsymbol{\lambda}_{j_\kappa} \in \{-1,0,1\}^{d_{j_\kappa}}$ for all $\kappa \in [t]$.
2. Compute $\tilde{\mathbf{z}} = \sum_{\kappa=1}^{t} \mathbf{Y}_{j_\kappa} \cdot \boldsymbol{\lambda}_{j_\kappa} \in \mathbb{Z}_p^m$, which equals

$$\tilde{\mathbf{z}} = \left\lfloor \big(\mathbf{A}(x)\big)^\top \cdot \mathbf{s} \right\rfloor_p + \mathbf{e}_z \; \in \mathbb{Z}_p^m,$$

for some $\mathbf{e}_z \in \{-2d_{\mathcal{S}}, \ldots, 2d_{\mathcal{S}}\}^m$.
3. Compute $\mathbf{z} = \lfloor \tilde{\mathbf{z}} \rfloor_u \in \mathbb{Z}_u^m$, which equals

$$\mathbf{z} = \left\lfloor \left\lfloor \big(\mathbf{A}(x)\big)^\top \cdot \mathbf{s} \right\rfloor_p \right\rfloor_u \; \in \mathbb{Z}_u^m$$

with overwhelming probability. Finally, output $\mathbf{y} = \pi(\mathbf{z}) \in \mathbb{Z}_u^k$.

By setting $\sigma = \sqrt{e \cdot n} = O(N^{\frac{1+\sqrt{2}}{2}}) \cdot \sqrt{n}$ as allowed by [40], we have share units of magnitude $\beta^* = \Theta(\sigma\sqrt{n}\log N) = O\left(N^{\frac{1+\sqrt{2}}{2}}\log N\right) \cdot n$. Since $d = O(N^{1+\sqrt{2}})$, the average share size amounts to $\frac{d \cdot n \cdot \log \beta^\star}{N} = n \cdot N^{\sqrt{2}} \cdot (\log n + O(\log N))$ bits.

Regarding the parameters, Theorem 4.2 allows us to rely on the presumed hardness of $\mathsf{LWE}_{q,m,n',\alpha}$ for n' which may be set as $\Theta(n(\log Nn)/(\log q))$ if $n(\log Nn) = \Omega(\lambda)$. To make sure that the best known attacks on LWE require 2^λ bit operations, it suffices that $\alpha q = \Omega(\sqrt{n'})$ and $n' \log q / \log^2 \alpha = \Omega(\lambda/\log\lambda)$. We may set $n = \mathsf{poly}(\lambda)$ (for a small degree polynomial) and $q = 2^{\Omega(\lambda \log \lambda)}$ since r contains a term $m^L = \mathsf{poly}(\lambda)^{\Theta(\lambda)} = 2^{O(\lambda \log \lambda)}$.

We remark that our modulus q is exponential in the input length L, but not in the number of servers N. In contrast, the DPRF of [16] requires an exponential modulus in N incurred by the use of Shamir's secret sharing and the technique of clearing out the denominators [3].

4.2 Security and Correctness

We now show that the construction provides statistical consistency.

Lemma 4.1. *Let* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^t, 1^N)$ *and let a secret key* $SK_0 = s \leftarrow D_{\mathbb{Z}^n,\sigma}$*, which is shared as* $(SK_1, \ldots, SK_N) \leftarrow \mathsf{Share}(\mathsf{pp}, SK_0)$*. For any* t*-subset* $\mathcal{S} = \{j_1, \ldots, j_t\} \subset [N]$ *and input* $X \in \{0,1\}^\ell$*, if* $Y_{j_k} = \mathsf{PEval}(\mathsf{pp}, SK_{j_k}, X)$ *for all* $\kappa \in [t]$*, we have*

$$\mathsf{Combine}(\mathcal{S}, (Y_{j_1}, \ldots, Y_{j_t})) = \mathsf{Eval}(\mathsf{pp}, SK_0, X)$$

with probability exponentially close to 1*.*

Proof. Let $\boldsymbol{\lambda}_\mathcal{S} \in \{-1,0,1\}^{d_\mathcal{S}}$ such that $\boldsymbol{\lambda}_\mathcal{S}^\top \cdot M_\mathcal{S} = (1,0,\ldots,0) \in \mathbb{Z}^e$. If we parse $\boldsymbol{\lambda}_\mathcal{S}$ as $[\boldsymbol{\lambda}_{j_1}^\top \mid \ldots \mid \boldsymbol{\lambda}_{j_t}^\top]^\top$ we have

$$\mathbf{s} = [\boldsymbol{\rho}_1 \mid \cdots \mid \boldsymbol{\rho}_n]^\top \cdot M_\mathcal{S}^\top \cdot \boldsymbol{\lambda}_\mathcal{S} = \sum_{k=1}^{t} \mathbf{S}_{I_{j_k}}^\top \cdot \boldsymbol{\lambda}_{j_k}.$$

In turn, this implies

$$\lfloor \mathbf{A}(x)^\top \cdot \mathbf{s} \rfloor_p = \left\lfloor \sum_{k=1}^{t} \mathbf{A}(x)^\top \cdot \mathbf{S}_{I_{j_k}}^\top \cdot \boldsymbol{\lambda}_{j_k} \right\rfloor_p = \sum_{k=1}^{t} \left\lfloor \mathbf{A}(x)^\top \cdot \mathbf{S}_{I_{j_k}}^\top \right\rfloor_p \cdot \boldsymbol{\lambda}_{j_k} + \boldsymbol{e}, \quad (4)$$

where the last equality of (4) stems from fact that, for any two vectors $\boldsymbol{v}_1, \boldsymbol{v}_2 \in \mathbb{Z}_q^m$, we have $\lfloor \boldsymbol{v}_1 + \boldsymbol{v}_2 \rfloor_p = \lfloor \boldsymbol{v}_1 \rfloor_p + \lfloor \boldsymbol{v}_2 \rfloor_p + \boldsymbol{e}^+$, for some vector $\boldsymbol{e}^+ \in \{0,1\}^m$, and $\lfloor \boldsymbol{v}_1 - \boldsymbol{v}_2 \rfloor_p = \lfloor \boldsymbol{v}_1 \rfloor_p - \lfloor \boldsymbol{v}_2 \rfloor_p + \boldsymbol{e}^-$, where $\boldsymbol{e}^- \in \{-1,0\}^m$. The error vector $\boldsymbol{e}$ of (4) thus lives in $\{-d_\mathcal{S}, \ldots, d_\mathcal{S}\}^m$. By the definition of $\mathbf{Y}_{j_k} = \lfloor \mathbf{A}(x)^\top \cdot \mathbf{S}_{I_{j_k}}^\top \rfloor_p$, if we define $\tilde{\mathbf{z}} := \sum_{k=1}^{t} \mathbf{Y}_{j_k} \cdot \boldsymbol{\lambda}_{j_k}$ and $\boldsymbol{e}_z := -\boldsymbol{e} \in \{-d_\mathcal{S}, \ldots, d_\mathcal{S}\}^m$, we have

$$\tilde{\mathbf{z}} = \left\lfloor \left(\mathbf{A}(x)\right)^\top \cdot \mathbf{s} \right\rfloor_p + \mathbf{e}_z \ \in \mathbb{Z}_p^m.$$

Then, we observe that $\mathbf{A}(x)^\top \cdot \mathbf{s}$ is of the form $\mathbf{T}_q \cdot \mathbf{A}_0^\top \cdot \boldsymbol{s}$, for some matrix $\mathbf{T}_q = (\prod_{i=1}^{L} \mathbf{G}^{-1}(\mathbf{A}_{i,x_i}))^\top$ which is a product of $\mathbb{Z}_q$-invertible matrices. By Lemma 2.19, $\mathbf{A}_0^\top \cdot \mathbf{s}$ is statistically close to the uniform distribution $U(\mathbb{Z}_q^m)$. Since $\mathbf{T}_q$ is invertible, the vector $\mathbf{A}(x)^\top \cdot \mathbf{s}$ is itself statistically close to $U(\mathbb{Z}_q^m)$. Hence, the vector $\lfloor \mathbf{A}(x)^\top \cdot \mathbf{s} \rfloor_p$ is statistically close to $U(\mathbb{Z}_p^m)$ since the statistical distance between $\lfloor U(\mathbb{Z}_q^m) \rfloor_p$ and $U(\mathbb{Z}_p^m)$ is at most $m \cdot (p/q)$. Therefore we can apply Lemma 2.18, which implies that

$$\big\lfloor \lfloor \mathbf{A}(x)^\top \cdot \boldsymbol{s} \rfloor_p \big\rfloor_u = \big\lfloor \lfloor \mathbf{A}(x)^\top \cdot \boldsymbol{s} \rfloor_p + \boldsymbol{e}_z \big\rfloor_u$$

except with probability $2^L \cdot m \cdot \frac{4 d_{\mathcal{S}} \cdot u}{p} \leq 2^L \cdot m \cdot \frac{4d \cdot u}{p} \leq m \cdot 2^{-\lambda}$.

This shows that the equality $\lfloor \tilde{\mathbf{z}} \rfloor_u = \big\lfloor \lfloor \mathbf{A}(x)^\top \cdot \boldsymbol{s} \rfloor_p \big\rfloor_u$ holds with overwhelming probability if the vector $\tilde{\mathbf{z}} := \sum_{k=1}^{t} \mathbf{Y}_{j_k} \cdot \boldsymbol{\lambda}_{j_k}$ in the left-hand-side member is computed by the Combine algorithm and the right-hand-side member is the $\lfloor \tilde{\mathbf{z}} \rfloor_u$ computed by Eval. ☐

Theorem 4.2. *Assume that an entropy lower bound* $\bar{n} = \lfloor n \cdot \log \sigma - \frac{n}{2} \cdot \log e - n' \cdot \log q \rfloor - 1$ *is* $\Omega(\lambda)$. *If we set the output length* $\bar{k} = k \cdot \log u$ *so as to have*

$$\xi = \bar{n} + \ell, \qquad \bar{k} = \bar{n} - 2 \cdot (\lambda + \log \ell + \log \bar{n}),$$

then the construction above is an adaptively secure DPRF family under the $\mathsf{LWE}_{q,m,n',\alpha}$ *assumption.*

Proof. The proof considers a sequence of hybrid games. In each game, we call W_i the event that $b' = b$.

Game$_0$: This is the experiment, as described by Definition 2.20. Namely, the challenger initially samples a secret Gaussian vector $SK_0 = \mathbf{s} \hookleftarrow D_{\mathbb{Z}^n,\sigma}$, which is shared by computing

$$\mathbf{S}_{I_j} = M_{I_j} \cdot [\boldsymbol{\rho}_1 \mid \ldots \mid \boldsymbol{\rho}_n] = M_{I_j} \cdot \mathbf{\Gamma} \quad \in \mathbb{Z}^{d_j \times n} \qquad \forall j \in [N],$$

where

$$\mathbf{\Gamma} = \big[\boldsymbol{\rho}_1 \mid \ldots \mid \boldsymbol{\rho}_n\big] = \begin{bmatrix} \mathbf{s}^\top \\ \hline \rho_{1,2} \cdots \rho_{n,2} \\ \vdots \quad \ddots \quad \vdots \\ \rho_{1,e} \cdots \rho_{n,e} \end{bmatrix} \in \mathbb{Z}^{e \times n},$$

with $\rho_{k,\nu} \hookleftarrow D_{\mathbb{Z},\sigma}$ for all $(k, \nu) \in [1, n] \times [2, e]$. At each partial evaluation query $(j, X^{(i)}) \in [N] \times \{0,1\}^\ell$, the adversary $\mathcal{A}$ obtains

$$\mathbf{Y}_j = \Big\lfloor (\mathbf{A}(x))^\top \cdot \mathbf{S}_{I_j}^\top \Big\rfloor_p \in \mathbb{Z}_p^{m \times d_j}. \tag{5}$$

In the challenge phase, the adversary chooses an input $X^\star \in \{0,1\}^\ell$. It obtains a random vector $\mathbf{y}^\star \hookleftarrow U(\mathbb{Z}_u^k)$ if the challenger's bit is $b = 0$. If $b = 1$, it obtains the real evaluation $\mathbf{y}^\star = \pi(\lfloor \tilde{\mathbf{z}}^\star \rfloor_u) \in \mathbb{Z}_u^k$, where

$$\tilde{\mathbf{z}}^\star = \Big\lfloor \big(\mathbf{A}(x^\star)\big)^\top \cdot \mathbf{s} \Big\rfloor_p \quad \in \mathbb{Z}_p^m,$$

with $\mathbf{A}(x^\star) = \mathbf{A}_0 \cdot \prod_{i=1}^{L} \mathbf{G}^{-1}(\mathbf{A}_{i,x_i^\star})$ and $x^\star = \mathsf{AHF}(X^\star) \in \{0,1\}^L$. At the end of the game, we define $\mathcal{C}^\star \subset [N]$ to the set of servers that were corrupted by $\mathcal{A}$ or such that an evaluation query of the form $(i, X^\star)$ was made. By hypothesis, we have $|\mathcal{C}^\star| < t$. When the adversary halts, it outputs $\hat{b} \in \{0,1\}$ and the challenger defines $b' := \hat{b}$. The adversary's advantage is $\mathbf{Adv}(\mathcal{A}) := |\Pr[W_0] - 1/2|$, where W_0 is event that $b' = b$.

Game$_1$: This game is identical to Game$_0$ with the following changes. First, the challenger runs $K \leftarrow \mathsf{AdmSmp}(1^\lambda, Q, \delta)$ to generate a key $K \in \{0,1,\perp\}^L$ for a balanced admissible hash function $\mathsf{AHF} : \{0,1\}^\ell \to \{0,1\}^L$, with $\delta := \mathbf{Adv}(\mathcal{A})$ and Q is an upper bound on the number of queries that the adversary makes. When the adversary halts and outputs $\hat{b} \in \{0,1\}$, the challenger checks if the conditions

$$P_K(X^{(1)}) = \cdots = P_K(X^{(Q)}) = 1 \quad \wedge \quad P_K(X^\star) = 0 \tag{6}$$

are satisfied, where $X^\star$ is the challenge input and $X^{(1)}, \ldots, X^{(Q)}$ are the adversarial queries. If these conditions do not hold, the challenger ignores $\mathcal{A}$'s output $\hat{b} \in \{0,1\}$ and overwrites it with a random bit $b'' \hookleftarrow \{0,1\}$ to define $b' = b''$. If conditions (6) are satisfied, the challenger sets $b' = \hat{b}$. By Lemma 2.9, we have

$$\begin{aligned} |\Pr[W_1] - 1/2| &= |\Pr[b' = b] - 1/2| \\ &\geq \gamma_{\min} \cdot \mathbf{Adv}(\mathcal{A}) - \frac{1}{2} \cdot (\gamma_{\max} - \gamma_{\min}) = \tau, \end{aligned}$$

where $\tau(\lambda)$ is a noticeable function.

Game$_2$: In this game, we modify the generation of pp in the following way. Initially, the challenger samples a uniformly random matrix $\mathbf{A} \hookleftarrow U(\mathbb{Z}_q^{n\times m})$. Next, for each $i \in [L]$, it samples $\mathbf{R}_{i,0}, \mathbf{R}_{i,1} \hookleftarrow U(\{-1,1\})^{m\times m}$ and defines $\{\mathbf{A}_{i,0}, \mathbf{A}_{i,1}\}_{i=1}^{L}$ as follows for all $i \in [L]$ and $j \in \{0,1\}$:

$$\mathbf{A}_{i,j} := \begin{cases} \mathbf{A} \cdot \mathbf{R}_{i,j} & \text{if} \quad (j \neq K_i) \ \wedge \ (K_i \neq \perp) \\ \mathbf{A} \cdot \mathbf{R}_{i,j} + \mathbf{G} & \text{if} \quad (j = K_i) \ \vee \ (K_i = \perp) \end{cases} \tag{7}$$

It also defines $\mathbf{A}_0 = \mathbf{A} \cdot \mathbf{R}_0 + \mathbf{G}$ for a randomly sampled $\mathbf{R}_0 \hookleftarrow U(\{-1,1\}^{m\times m})$. Since $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ was chosen uniformly, the Leftover Hash Lemma ensures that $\{\mathbf{A}_{i,0}, \mathbf{A}_{i,1}\}_{i=1}^{L}$ are statistically independent and uniformly distributed over $\mathbb{Z}_q^{n\times m}$. It follows that $|\Pr[W_2] - \Pr[W_1]| \leq L \cdot 2^{-\lambda}$ since the distribution of pp is statistically unchanged.

We note that, at each query X, we can view $\mathbf{A}(x)$ as a GSW encryption

$$\mathbf{A}(x) = \mathbf{A} \cdot \mathbf{R}_x + (\prod_{i=1}^{n} \mu_i) \cdot \mathbf{G},$$

for some small norm $\mathbf{R}_x \in \mathbb{Z}^{m\times m}$, where

$$\mu_i := \begin{cases} 0 & \text{if} \quad (\mathsf{AHF}(X)_i \neq K_i) \ \wedge \ (K_i \neq \perp) \\ 1 & \text{if} \quad (\mathsf{AHF}(X)_i = K_i) \ \vee \ (K_i = \perp) \end{cases}$$

If conditions (6) are satisfied, at each query $X^{(i)}$, the admissible hash function ensures that $x^{(i)} = \mathsf{AHF}(X^{(i)})$ satisfies

$$\mathbf{A}(x^{(i)}) = \mathbf{A} \cdot \mathbf{R}_{x^{(i)}}, \tag{8}$$

for some small norm $\mathbf{R}_{x^{(i)}} \in \mathbb{Z}^{m \times m}$. Moreover, the admissible hash function maps the challenge input $X^\star$ to an L-bit string $x^\star = \mathsf{AHF}(X^\star)$ such that

$$\mathbf{A}(x^\star) = \mathbf{A} \cdot \mathbf{R}_{x^\star} + \mathbf{G}. \tag{9}$$

Game$_3$**:** In this game, we modify the distribution of pp and replace the uniform matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ by a lossy matrix such that

$$\mathbf{A}^\top = \bar{\mathbf{A}}^\top \cdot \mathbf{C} + \mathbf{E} \ \in \mathbb{Z}_q^{m \times n}, \tag{10}$$

where $\bar{\mathbf{A}} \hookleftarrow U(\mathbb{Z}_q^{n' \times m})$, $\mathbf{C} \hookleftarrow U(\mathbb{Z}_q^{n' \times n})$ and $\mathbf{E} \hookleftarrow D_{\mathbb{Z}^{m \times n}, \alpha q}$, for n' significantly smaller than n. The matrix in (10) is thus "computationally close" to a matrix $\bar{\mathbf{A}}^\top \cdot \mathbf{C}$ of much lower rank than n. Under the LWE assumption with in dimension n', this change should not significantly alter $\mathcal{A}$'s behavior and a straightforward reduction $\mathcal{B}$ shows that $|\Pr[W_3] - \Pr[W_2]| \leq n \cdot \mathbf{Adv}_{\mathcal{B}}^{\mathsf{LWE}_{q,m,n',\alpha}}(\lambda)$, where the factor n comes from the use of an LWE assumption with n secrets.

The modification introduced in Game_3 has the following consequence. Assuming that conditions (6) are satisfied, for each partial evaluation query $X^{(i)}$ such that $X^{(i)} \neq X^\star$, the response is of the form $\mathbf{Y}_j = [\mathbf{z}_{j,1} \mid \ldots \mid \mathbf{z}_{j,d_j}] \in \mathbb{Z}_p^{m \times d_j}$, where

$$\begin{aligned} \mathbf{z}_{j,\theta} &= \lfloor \big(\mathbf{A} \cdot \mathbf{R}_{x^{(i)}}\big)^\top \cdot \bar{\mathbf{s}}_{j,\theta} \rfloor_p \\ &= \lfloor \big(\mathbf{R}_{x^{(i)}}^\top \cdot \bar{\mathbf{A}}^\top \cdot \mathbf{C} + \mathbf{R}_{x^{(i)}}^\top \cdot \mathbf{E}\big) \cdot \bar{\mathbf{s}}_{j,\theta} \rfloor_p \qquad \forall \theta \in [d_j]. \end{aligned}$$

Game$_4$**:** In this game, we modify the evaluation oracle and introduce a bad event. We define BAD to be the event that the adversary makes a partial evaluation query (j, X) such that the AHF-encoded input $x = \mathsf{AHF}(X) \in \{0,1\}^L$ corresponds to a matrix $\mathbf{A}(x) = \mathbf{A} \cdot \mathbf{R}_x$, for some small-norm $\mathbf{R}_x \in \mathbb{Z}^{m \times m}$, such that we have

$$\mathbf{z}_{j,\theta} = \lfloor \big(\mathbf{A} \cdot \mathbf{R}_x\big)^\top \cdot \bar{\mathbf{s}}_{j,\theta} \rfloor_p \neq \lfloor \big(\mathbf{R}_x^\top \cdot \bar{\mathbf{A}}^\top \cdot \mathbf{C}\big) \cdot \bar{\mathbf{s}}_{j,\theta} \rfloor_p. \tag{11}$$

for some $\theta \in [d_j]$. Note that the challenger can detect this event since it knows $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n' \times m}$, $\mathbf{C} \in \mathbb{Z}_q^{n' \times n}$ and $\mathbf{E} \in \mathbb{Z}^{m \times n}$ satisfying (10). If BAD occurs, the challenger overwrites $\mathcal{A}$'s output $\hat{b}$ with a random bit $b'' \hookleftarrow \{0,1\}$ and sets $b' = b''$ (otherwise, it sets $b' = \hat{b}$ as before). Lemma 4.3 shows that we have the inequality $|\Pr[W_4] - \Pr[W_3]| \leq \Pr[\mathsf{BAD}] \leq 2^{-\Omega(\lambda)}$.

We note that, if BAD does not occur, we have

$$\Big\lfloor \big(\mathbf{A} \cdot \mathbf{R}_{x^{(i)}}\big)^\top \cdot \bar{\mathbf{s}}_{j,\theta} \Big\rfloor_p = \big\lfloor \big(\mathbf{R}_{x^{(i)}}^\top \cdot \bar{\mathbf{A}}^\top \cdot \mathbf{C}\big) \cdot \bar{\mathbf{s}}_{j,\theta} \big\rfloor_p \qquad \forall (j,\theta) \in [N] \times [d_j] \tag{12}$$

at each query $(j, X^{(i)})$ for which $X^{(i)} \neq X^\star$. We note that the right-hand-side member of (12) is fully determined by $\mathbf{R}_{x^{(i)}}^\top \cdot \bar{\mathbf{A}}^\top$ and the product $\mathbf{C} \cdot \bar{\mathbf{s}}_{j,\theta} \in \mathbb{Z}_q^{n'}$. This means that partial evaluation queries $(j, X^{(i)})$ such that $X^{(i)} \neq X^\star$ always reveal the same information (namely, $\mathbf{C} \cdot \bar{\mathbf{s}}_{j,\theta} \in \mathbb{Z}_q^{n'}$) about $\bar{\mathbf{s}}_{j,\theta} \in \mathbb{Z}^n$.

Conversely, the right-hand-side member of (12) uniquely determines $\mathbf{C} \cdot \bar{\mathbf{s}}_{j,\theta}$ with high probability: observe that $\mathbf{R}_{x^{(i)}}^\top \cdot \bar{\mathbf{A}}^\top$ is statistically uniform over $\mathbb{Z}_q^{m \times n'}$, so by Lemma 2.1, the quantity $\lfloor \mathbf{R}_{x^{(i)}}^\top \cdot \bar{\mathbf{A}}^\top \cdot (\mathbf{C} \cdot \boldsymbol{s}) \rfloor_p$ is an injective function of $\mathbf{C} \cdot \boldsymbol{s} \bmod q$. It comes that partial evaluation queries information-theoretically reveal $\mathbf{C} \cdot \mathbf{s} \bmod q$, but we will show that $\mathbf{s}$ still retains high entropy in $\mathcal{A}$'s view.

Game$_5$: In this game, we modify the challenge value for which, if $b = 1$, the adversary is given a random $\mathbf{y}^\star \hookleftarrow U(\mathbb{Z}_u^k)$. Clearly, we have $\Pr[W_5] = 1/2$ since the distribution of the challenge value does not depend on $b \in \{0, 1\}$. Moreover, we will show that $|\Pr[W_5] - \Pr[W_4]| \leq 2^{-\Omega(\lambda)}$.

Indeed, we claim that, conditionally on $\mathcal{A}$'s view, the vector $\mathbf{y}^\star$ is already statistically uniform over $\mathbb{Z}_u^k$ in Game$_4$. Indeed, the source $\lfloor \tilde{\mathbf{z}}^\star \rfloor_u$ depends on an injective function $\mathbf{G}^\top \cdot \mathbf{s} \in \mathbb{Z}_q^m$ of the vector $\mathbf{s}$. In Lemma 4.4, we show that this vector has high min-entropy if BAD does not occur.

We observe that the source $\lfloor \tilde{\mathbf{z}}^\star \rfloor_u$ can be written

$$\begin{aligned}
\lfloor \tilde{\mathbf{z}}^\star \rfloor_u &= \Big\lfloor \Big\lfloor \big(\mathbf{A} \cdot \mathbf{R}_{x^\star} + \mathbf{G}\big)^\top \cdot \mathbf{s} \Big\rfloor_p \Big\rfloor_u && (13)\\
&= \Big\lfloor \big(\mathbf{A} \cdot \mathbf{R}_{x^\star} + \mathbf{G}\big)^\top \cdot \mathbf{s} \Big\rfloor_u + \mathbf{e}_{s,x,u} && \text{with } \mathbf{e}_{s,x,u} \in \{-1, 0\}^m \\
&= \lfloor \mathbf{R}_{x^\star}^\top \cdot \mathbf{A}^\top \cdot \mathbf{s} \rfloor_u + \lfloor \mathbf{G}^\top \cdot \mathbf{s} \rfloor_u + \mathbf{e}_{s,x,u} + \mathbf{e}_{s,x}, && \text{with } \mathbf{e}_{s,x} \in \{0, 1\}^m \\
&= \lfloor \mathbf{R}_{x^\star}^\top \cdot \mathbf{A}^\top \cdot \mathbf{s} \rfloor_u + \lfloor \mathbf{G}^\top \cdot \mathbf{s} \rfloor_u + \mathbf{e}'_{s,x}, && \text{with } \mathbf{e}'_{s,x} \in \{-1, 0, 1\}^m. \quad (14)
\end{aligned}$$

The proof of Lemma 4.3 (see also the proof of the claim in Game 4 in the proof of Theorem 3.2) implies that $\lfloor \mathbf{R}_{x^\star}^\top \cdot \mathbf{A}^\top \cdot \mathbf{s} \rfloor_p = \lfloor \mathbf{R}_{x^\star}^\top \cdot \bar{\mathbf{A}}^\top \cdot \mathbf{C} \cdot \mathbf{s} \rfloor_p$ with overwhelming probability. In turn, this implies $H_\infty\big(\lfloor \mathbf{R}_{x^\star}^\top \cdot \mathbf{A}^\top \cdot \mathbf{s} \rfloor_u \mid \mathbf{C} \cdot \mathbf{s}\big) = 0$ with high probability. In the expression of $\tilde{\mathbf{z}}^\star$ in (14), we also remark that $\lfloor \mathbf{G}^\top \cdot \mathbf{s} \rfloor_u + \mathbf{e}'_{s,x}$ is an injective function of $\mathbf{s} \in \mathbb{Z}^n$. To see this, observe that

$$\lfloor \mathbf{G}^\top \cdot \mathbf{s} \rfloor_u + \mathbf{e}'_{s,x} = (u/q) \cdot \mathbf{G}^\top \cdot \mathbf{s}' - \mathbf{t}_{s,x} + \mathbf{e}'_{s,x}$$

for some $\mathbf{t}_{s,x} \in (0, 1)^m$, so that

$$(q/u) \cdot (\lfloor \mathbf{G}^\top \cdot \mathbf{s} \rfloor_u + \mathbf{e}'_{s,x}) = \mathbf{G}^\top \cdot \mathbf{s} + \mathbf{e}''_{s,x} \quad (15)$$

for some $\mathbf{e}''_{s,x} \in (-q/u, 2 \cdot q/u)^m$. The vector $\mathbf{s}$ is thus uniquely determined by (15) using the public trapdoor of $\mathbf{G}$ so long as $q/u \ll q$.

Consider the entropy of $\tilde{\mathbf{z}}^\star$ conditionally on $\mathcal{A}$'s view. We have

$$\begin{aligned}
H_\infty\big(\lfloor \tilde{\mathbf{z}}^\star \rfloor_u &\mid \mathbf{C} \cdot \boldsymbol{\Gamma}^\top,\ \{\mathbf{S}_{I_j}\}_{j \in \mathcal{C}^\star}\big) \\
&= H_\infty\big(\lfloor \mathbf{R}_{x^\star}^\top \cdot \mathbf{A}^\top \cdot \mathbf{s} \rfloor_u + \lfloor \mathbf{G}^\top \cdot \mathbf{s} \rfloor_u + \mathbf{e}'_{s,x} \mid \mathbf{C} \cdot \boldsymbol{\Gamma}^\top,\ \{\mathbf{S}_{I_j}\}_{j \in \mathcal{C}^\star}\big) \\
&= H_\infty\big(\lfloor \mathbf{G}^\top \cdot \mathbf{s} \rfloor_u + \mathbf{e}'_{s,x} \mid \mathbf{C} \cdot \boldsymbol{\Gamma}^\top,\ \{\mathbf{S}_{I_j}\}_{j \in \mathcal{C}^\star}\big) \\
&= H_\infty\big(\mathbf{s} \mid \mathbf{C} \cdot \boldsymbol{\Gamma}^\top,\ \{\mathbf{S}_{I_j}\}_{j \in \mathcal{C}^\star}\big) \geq n \cdot \log \sigma - \frac{n}{2} \cdot \log e - n' \cdot \log q - 1.
\end{aligned}$$

Here, the last inequality is given by Lemma 4.4. The second equality follows from the fact that, for any random variables X, Y, Z defined over an additive group, we have $H_\infty(Y + Z \mid X) = H_\infty(Z|X)$ if $H_\infty(Y|X) = 0$.

In order to extract statistically random bits from $\tilde{\mathbf{z}}^\star$, we must take into account that it possibly depends on $x^\star$ which may depend on pp. As long as $P_K(X^\star) = 0$, the source $\tilde{\mathbf{z}}^\star$ is taken from a distribution determined by the challenge input $X^\star \in \{0,1\}^\ell$ within a collection of less than 2^ℓ distributions (namely, those inputs X for which $P_K(X) = 0$), which all have min-entropy $\bar{n} \geq n \log \sigma - \frac{n}{2} \cdot \log e - n' \log q - 1$. By applying Lemma 2.11 with $\epsilon = 2^{-\lambda}$ for a collection $\mathcal{X}$ of at most $M = 2^\ell$ distributions, we obtain that the distribution of $\pi(\lfloor \tilde{\mathbf{z}}^\star \rfloor_u)$ is $2^{-\Omega(\lambda)}$-close to the uniform distribution over $\mathbb{Z}_u^k$. □

Lemma 4.3. *Assume that $q/p > 2^{L+\lambda} \cdot r$, where $r = m^{L+2} \cdot n \cdot \beta^* \cdot \alpha q$ with $\beta^* = O(\beta \cdot \log N)$. Then, we have the inequality*

$$|\Pr[W_4] - \Pr[W_3]| \leq \Pr[\mathsf{BAD}] \leq 2^{-\Omega(\lambda)}.$$

(The proof is given in the full version of the paper.)

Lemma 4.4. *In Game_4, the min-entropy of $\mathbf{s}$ conditionally on $\mathcal{A}$'s view is at least $n \cdot \log \sigma - \frac{n}{2} \cdot \log e - n' \cdot \log q - \frac{n}{2^n}$.*

Proof. Let us assume that BAD does not occur in Game_4 since, if it does, the challenger replaces the adversary's output with a random bit, in which case both games have the same outcome. We show that, assuming $\neg\mathsf{BAD}$, the shared secret vector $\mathbf{s}$ retains high min-entropy conditionally on the adversary's view.

Let us first recap what the adversary can see in Game_4. For each partial evaluation query $(j, X^\star)$, the response $\lfloor (\mathbf{A} \cdot \mathbf{R}_{x^\star} + \mathbf{G})^\top \cdot \mathbf{S}_{I_j}^\top \rfloor_p$ consists of non-lossy functions of $\mathbf{S}_{I_j}^\top \in \mathbb{Z}^{n \times d_j}$. We thus consider partial evaluation queries of the form $(j, X^\star)$ as if they were corruption queries and assume that they information-theoretically reveal $\mathbf{S}_{I_j}$ (we thus merge the two sets $\mathcal{C}$ and $\mathcal{E}$ of Definition 2.20 into one set $\mathcal{C}^\star$). As for uncorrupted shares $\{\mathbf{S}_{I_j}\}_{j \in [N] \setminus \mathcal{C}^\star}$, partial evaluation queries $(j, X^{(i)})$ for which $X^{(i)} \neq X^\star$ only reveal the information $\{\mathbf{C} \cdot \mathbf{S}_{I_j}^\top\}_{j \in [N] \setminus \mathcal{C}^\star}$. More precisely, those partial evaluations $\{\mathbf{Y}_j\}_{j \in [N] \setminus \mathcal{C}^\star}$ can be written

$$\mathbf{Y}_j = \Big\lfloor (\mathbf{A}(x^{(i)}))^\top \cdot \mathbf{S}_{I_j}^\top \Big\rfloor_p = \Big\lfloor (\mathbf{R}_{x^{(i)}} \cdot \bar{\mathbf{A}}^\top \cdot \mathbf{C}) \cdot \mathbf{S}_{I_j}^\top \Big\rfloor_p \tag{16}$$

where

$$\mathbf{S}_{I_j}^\top = \begin{bmatrix} \boldsymbol{\rho}_1^\top \\ \vdots \\ \boldsymbol{\rho}_n^\top \end{bmatrix} \cdot M_{I_j}^\top \quad \in \mathbb{Z}^{n \times d_j}$$

is a product of $M_{I_j}^\top$ with the matrix $[\boldsymbol{\rho}_1 \mid \ldots \mid \boldsymbol{\rho}_n]^\top \in \mathbb{Z}^{n \times e}$ whose first column is the secret $SK_0 = \mathbf{s} \in \mathbb{Z}^n$. Hence, the information revealed by (16) for $j \in [N] \setminus \mathcal{C}^\star$ is only a lossy function $\mathbf{C} \cdot \mathbf{S}_{I_j}^\top$ of the share $\mathbf{S}_{I_j}$: namely,

$$\mathbf{C} \cdot \begin{bmatrix} \boldsymbol{\rho}_1^\top \\ \vdots \\ \boldsymbol{\rho}_n^\top \end{bmatrix} \cdot M_{I_j}^\top = \left[\begin{array}{c|c|c|c} \mathbf{C} \cdot \mathbf{s} & \mathbf{C} \cdot \begin{pmatrix} \rho_{2,2} \\ \vdots \\ \rho_{n,2} \end{pmatrix} & \dots & \mathbf{C} \cdot \begin{pmatrix} \rho_{2,e} \\ \vdots \\ \rho_{n,e} \end{pmatrix} \end{array} \right] \cdot M_{I_j}^\top, \quad (17)$$
$$= \mathbf{C} \cdot \mathbf{\Gamma}^\top \cdot M_{I_j}^\top,$$

where

$$\mathbf{\Gamma} = \left[\boldsymbol{\rho}_1 \mid \dots \mid \boldsymbol{\rho}_n\right] = \begin{bmatrix} \mathbf{s}^\top \\ \hline \rho_{2,2} & \cdots & \rho_{n,2} \\ \vdots & \ddots & \vdots \\ \rho_{2,e} & \cdots & \rho_{n,e} \end{bmatrix} \in \mathbb{Z}^{e \times n}$$

is the matrix of Gaussian entries which is used to compute secret key shares

$$\mathbf{S}_{I_j} = M_{I_j} \cdot \mathbf{\Gamma} \qquad\qquad \forall j \in [N].$$

The information revealed by exposed shares $\{\mathbf{S}_{I_j}\}_{j \in \mathcal{C}^\star}$ can thus be written

$$\mathbf{S}_{I_j} = [\mathbf{s}_{I_j,1} \mid \dots \mid \mathbf{s}_{I_j,n}] = M_{I_j} \cdot \mathbf{\Gamma} \quad \in \mathbb{Z}^{d_j \times n} \qquad\qquad \forall j \in \mathcal{C}^\star. \quad (18)$$

At this stage, we see that proving the following fact on distributions is sufficient to complete the proof of the lemma.

Fact. *Let $M_{\mathcal{C}^\star}$ to be the sub-matrix of M obtained by stacking up the rows assigned to corrupted parties $j \in \mathcal{C}^\star$. Conditionally on*

$$\big(\mathbf{C},\ \mathbf{C} \cdot \mathbf{\Gamma}^\top \cdot M^\top,\ M_{\mathcal{C}^\star},\ M_{\mathcal{C}^\star} \cdot \mathbf{\Gamma}\big), \quad (19)$$

the vector $\mathbf{s}^\top = (1, 0, \dots, 0)^\top \cdot \mathbf{\Gamma}$ has min-entropy at least

$$n \cdot \log \sigma - \frac{n}{2} \cdot \log e - n' \cdot \log q - \frac{n}{2^n}.$$

To prove this statement, we apply arguments inspired from [4, Lemma 1]. First, we observe that conditioning on (19) is the same as conditioning on $\big(\mathbf{C}, \mathbf{C} \cdot \mathbf{\Gamma}^\top \cdot M_{[N] \setminus \mathcal{C}^\star}^\top, M_{\mathcal{C}^\star}, M_{\mathcal{C}^\star} \cdot \mathbf{\Gamma}\big)$ since $M_{\mathcal{C}^\star} \cdot \mathbf{\Gamma}$ and $\mathbf{C}$ are given. In fact, it is sufficient to prove the result when conditioning on

$$\big(\mathbf{C}, \mathbf{C} \cdot \mathbf{\Gamma}^\top, M_{\mathcal{C}^\star}, M_{\mathcal{C}^\star} \cdot \mathbf{\Gamma}\big),$$

as $\mathbf{C} \cdot \mathbf{\Gamma}^\top \cdot M_{[N] \setminus \mathcal{C}^\star}^\top$ is computable from $\mathbf{C} \cdot \mathbf{\Gamma}^\top$. By the definition of an Integer Span Program, we know that there exists a sweeping vector $\boldsymbol{\kappa} \in \mathbb{Z}^e$ whose first coordinate is $\kappa_1 = 1$ and such that $M_{\mathcal{C}^\star} \cdot \boldsymbol{\kappa} = \mathbf{0}$. The rows of $M_{\mathcal{C}^\star}$ thus live in the lattice $\mathcal{L}_{\mathcal{C}^\star} = \{\boldsymbol{m} \in \mathbb{Z}^e : \langle \boldsymbol{m}, \boldsymbol{\kappa} \rangle = 0\}$. Hence, if we define a matrix $L_{\mathcal{C}^\star} \in \mathbb{Z}^{(e-1) \times e}$ whose rows form a basis of $\mathcal{L}_{\mathcal{C}^\star}$, we may prove the min-entropy lower bound conditioned on

$$\big(\mathbf{C}, \mathbf{C} \cdot \mathbf{\Gamma}^\top, L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star} \cdot \mathbf{\Gamma}\big).$$

This is because $L_{\mathcal{C}^\star} \cdot \mathbf{\Gamma}$ provides at least as much information as $M_{\mathcal{C}^\star} \cdot \mathbf{\Gamma}$.

We first consider the distribution of $\mathbf{\Gamma}$, conditioned on $(L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star} \cdot \mathbf{\Gamma})$. Since the columns of $\mathbf{\Gamma}$ are statistically independent, we may look at them individually. For each $i \in [n]$, we let $\boldsymbol{\rho}_i^* \in \mathbb{Z}^e$ be an arbitrary solution of $L_{\mathcal{C}^\star} \cdot \boldsymbol{\rho}_i^* = L_{\mathcal{C}^\star} \cdot \boldsymbol{\rho}_i \in \mathbb{Z}_q^{e-1}$. The distribution of $\boldsymbol{\rho}_i \in \mathbb{Z}^e$ conditionally on $\left(L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star} \cdot \boldsymbol{\rho}_i\right)$ is $\boldsymbol{\rho}_i^* + D_{\Lambda,\sigma,-\boldsymbol{\rho}_i^*}$, where $\Lambda = \{\boldsymbol{x} \in \mathbb{Z}^e \mid L_{\mathcal{C}^\star} \cdot \boldsymbol{x} = \mathbf{0}\}$ is the 1-dimensional lattice $\Lambda = \boldsymbol{\kappa} \cdot \mathbb{Z}$.

At this stage, we know that conditioned on $(L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star} \cdot \mathbf{\Gamma})$, each row $\boldsymbol{\rho}_i = (s_i, \rho_{i,2}, \ldots, \rho_{i,e})^\top$ of $\mathbf{\Gamma}^\top$ is Gaussian over an affine line. We use this observation to show that conditioning on $(\mathbf{C}, \mathbf{C} \cdot \mathbf{\Gamma}^\top, L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star} \cdot \mathbf{\Gamma})$ is the same as conditioning on $(\mathbf{C}, \mathbf{C} \cdot \mathbf{s}, L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star} \cdot \mathbf{\Gamma})$.[3] In fact, we claim that, conditioned on $(L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star} \cdot \mathbf{\Gamma})$, the last $e-1$ columns of $\mathbf{\Gamma}^\top$ do not reveal any more information than its first column. Indeed, conditioned on $(L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star} \cdot \mathbf{\Gamma})$, each $\boldsymbol{\rho}_i$ can be written $\boldsymbol{\rho}_i = \xi_i \cdot \boldsymbol{\kappa} + \boldsymbol{\rho}_i^*$ for some integer $\xi_i \in \mathbb{Z}$. We may assume that the shifting vector $\boldsymbol{\rho}_i^* = (\rho_{i,1}^*, \ldots, \rho_{i,e}^*)^\top \in \mathbb{Z}_q^e$ is known to $\mathcal{A}$ as it can be obtained from $L_{\mathcal{C}^\star} \cdot \boldsymbol{\rho}_i$ via de-randomized Gaussian elimination. Writing $\boldsymbol{\kappa} = (\kappa_1, \ldots, \kappa_e)$, the j-th column $(\mathbf{\Gamma}^\top)_j$ of $\mathbf{\Gamma}^\top$ is

$$(\mathbf{\Gamma}^\top)_j = \kappa_j \cdot \begin{pmatrix} \xi_1 \\ \vdots \\ \xi_n \end{pmatrix} + \begin{pmatrix} \rho_{1,j}^* \\ \vdots \\ \rho_{n,j}^* \end{pmatrix} \qquad \forall j \in [e].$$

As $\kappa_1 = 1$, we have

$$(\mathbf{\Gamma}^\top)_j = \kappa_j \cdot (\mathbf{\Gamma}^\top)_1 - \kappa_j \cdot \begin{pmatrix} \rho_{1,1}^* \\ \vdots \\ \rho_{n,1}^* \end{pmatrix} + \begin{pmatrix} \rho_{1,j}^* \\ \vdots \\ \rho_{n,j}^* \end{pmatrix} \qquad \forall j \in [e].$$

In the latter, the last two terms are information-theoretically known to $\mathcal{A}$ (once we have conditioned on $(L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star} \cdot \mathbf{\Gamma})$) and so is κ_j.

We now study the distribution of $\mathbf{s} = (\mathbf{\Gamma}^\top)_1$ conditioned on $(L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star} \cdot \mathbf{\Gamma})$. By statistical independence, we may consider each coordinate $s_i = (1, 0, \ldots, 0)^\top \cdot \boldsymbol{\rho}_i$ of $\mathbf{s}$ individually. Recall that, conditioned on $(L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star} \cdot \mathbf{\Gamma})$, each $\boldsymbol{\rho}_i$ is distributed as $\boldsymbol{\rho}_i^* + D_{\boldsymbol{\kappa}\mathbb{Z},\sigma,-\boldsymbol{\rho}_i^*}$. Write $\boldsymbol{\rho}_i^* = y \cdot \boldsymbol{\kappa} + (\boldsymbol{\rho}_i^*)^\perp$, with $y \in \mathbb{R}$ and $(\boldsymbol{\rho}_i^*)^\perp$ orthogonal to $\boldsymbol{\kappa}$. Then,

$$\begin{aligned} \boldsymbol{\rho}_i^* + D_{\boldsymbol{\kappa}\mathbb{Z},\sigma,-\boldsymbol{\rho}_i^*} &= (\boldsymbol{\rho}_i^*)^\perp + y \cdot \boldsymbol{\kappa} + D_{\boldsymbol{\kappa}\mathbb{Z},\sigma,-y\cdot\boldsymbol{\kappa}-(\boldsymbol{\rho}_i^*)^\perp} \\ &= (\boldsymbol{\rho}_i^*)^\perp + y \cdot \boldsymbol{\kappa} + \boldsymbol{\kappa} \cdot D_{\mathbb{Z},\sigma/\|\boldsymbol{\kappa}\|,-y}. \end{aligned}$$

We now take the inner product with $(1, 0, \ldots, 0)$ and use the fact that $\kappa_1 = 1$ to obtain that, conditioned on $(L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star} \cdot \mathbf{\Gamma})$, the coordinate s_i is distributed as $(\boldsymbol{\rho}_i^*)_1^\perp + y + D_{\mathbb{Z},\sigma/\|\boldsymbol{\kappa}\|,-y}$. As $\boldsymbol{\kappa} \in \{-1, 0, 1\}^e$ with the Benaloh-Leichter-based LISS scheme of [24], and by our choice of σ, we have that $\sigma/\|\boldsymbol{\kappa}\| = \Omega(\sqrt{n})$. Using Lemma 2.3 (Remark 2.4), this implies that each s_i has min-entropy $\geq$

[3] Note that conditioned on $(L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star} \cdot \mathbf{\Gamma})$, the *rows* of $\mathbf{\Gamma}^\top$ are Gaussian on affine lines, but a *column* of $\mathbf{\Gamma}^\top$ is an inner product of unit vector with all these rows.

$\log(\sigma/\|\boldsymbol{\kappa}\|) - 2^{-n} \geq \log\sigma - \frac{1}{2}\log e - 2^{-n}$. Overall, we obtain

$$H_\infty\big(\mathbf{s} \mid L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star}\cdot\boldsymbol{\Gamma}\big) \geq n\cdot\log\sigma - \frac{n}{2}\cdot\log e - \frac{n}{2^n}.$$

We are now ready to conclude. By the above, to prove the fact (and hence the lemma), it suffices to obtain a lower bound on the min-entropy of $\mathbf{s}$ conditioned on $(\mathbf{C}, \mathbf{C}\cdot\mathbf{s}, L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star}\cdot\boldsymbol{\Gamma})$. We then use the above min-entropy lower bound on $\mathbf{s}$ conditioned on $(L_{\mathcal{C}^\star}, L_{\mathcal{C}^\star}\cdot\boldsymbol{\Gamma})$ and the fact that given $\mathbf{C}$, the quantity $\mathbf{C}\cdot\mathbf{s}\in\mathbb{Z}_q^{n'}$ reveals at most $n'\log q$ bits. □

Acknowledgements. We thank Javier Herranz for his suggestion to use linear integer secret sharing schemes. Part of this research was funded by the French ANR ALAMBIC project (ANR-16-CE39-0006) and by BPI-France in the context of the national project RISQ (P141580). This work was also supported in part by the European Union PROMETHEUS project (Horizon 2020 Research and Innovation Program, grant 780701). The second author was supported by ERC Starting Grant ERC-2013-StG-335086-LATTAC.

A Definition of Static DPRF Security

In this section, we recall the definition of static security used in [16].

Definition A.1. *Let λ be a security parameter and let integers $t, N \in \mathsf{poly}(\lambda)$. A (t,N)-DPRF is pseudorandom under static corruptions if no PPT adversary has non-negligible advantage in the following game:*

1. *The challenger generates $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^t, 1^N)$ and chooses a random key $SK_0 \hookleftarrow \mathcal{K}$, which is broken into N shares $(SK_1,\ldots,SK_N) \leftarrow \mathsf{Share}(SK_0)$. It also initializes empty sets $\mathcal{C},\mathcal{V}\leftarrow\emptyset$ and flip a random coin $b \hookleftarrow U(\{0,1\})$.*
2. *The adversary $\mathcal{A}$ chooses a set $S^\star = \{i_1,\ldots,i_{t-1}\}$ and the challenger returns the secret key shares $\{SK_{i_1},\ldots,SK_{i_{t-1}}\}$.*
3. *The adversary $\mathcal{A}$ adaptively interleaves the following kinds of queries.*

   Evaluation: *$\mathcal{A}$ chooses an input $X\in\mathcal{D}$. The challenger replies by returning $\{Y_i = \mathsf{PEval}(SK_i, X)\}_{i\in[N]\setminus S^\star}$ and updating $\mathcal{V} := \mathcal{V}\cup\{X\}$.*

   Challenge: *$\mathcal{A}$ chooses an input $X\in\mathcal{D}$. If X previously occurred in a challenge query, the challenger returns the same output as before. Otherwise, it randomly chooses $Y_{X,0} \hookleftarrow U(\{0,1\}^\mu)$ and computes $Y_{X,1} = \mathsf{Eval}(SK_0, X)$. It returns $Y_{X,b}$ and updates $\mathcal{C} := \mathcal{C}\cup\{X\}$.*

   It is required that $\mathcal{C}\cap\mathcal{V}=\emptyset$ at any time.
4. *The adversary $\mathcal{A}$ outputs a bit $\hat{b}\in\{0,1\}$ and wins if $\hat{b}=b$. Its advantage is defined to be $\mathbf{Adv}_{\mathcal{A}}^{\mathrm{DPRF}}(\lambda) := |\Pr[\hat{b}=b]-1/2|$.*

We may assume w.l.o.g. that the adversary only makes one challenge query in the experiment of Definition A.1. Indeed, a standard hybrid argument allows showing that security in the single-challenge sense implies security when polynomially-many queries are allowed.

References

1. Abe, M., Fehr, S.: Adaptively secure Feldman VSS and applications to universally-composable threshold cryptography. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 317–334. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28628-8_20
2. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_28
3. Agrawal, S., Boyen, X., Vaikuntanathan, V., Voulgaris, P., Wee, H.: Functional encryption for threshold functions (or Fuzzy IBE) from lattices. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 280–297. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30057-8_17
4. Agrawal, S., Libert, B., Stehlé, D.: Fully Secure Functional Encryption for Inner Products, from Standard Assumptions. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9816, pp. 333–362. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53015-3_12
5. Almansa, J.F., Damgård, I., Nielsen, J.B.: Simplified threshold RSA with adaptive and proactive security. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 593–611. Springer, Heidelberg (2006). https://doi.org/10.1007/11761679_35
6. Alwen, J., Krenn, S., Pietrzak, K., Wichs, D.: Learning with rounding, revisited. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 57–74. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_4
7. Banerjee, A., Peikert, C.: New and improved key-homomorphic pseudorandom functions. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 353–370. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_20
8. 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 (2012). https://doi.org/10.1007/978-3-642-29011-4_42
9. Bellare, M., Kiltz, E., Peikert, C., Waters, B.: Identity-based (Lossy) trapdoor functions and applications. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 228–245. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_15
10. Benaloh, J., Leichter, J.: Generalized secret sharing and monotone functions. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 27–35. Springer, New York (1990). https://doi.org/10.1007/0-387-34799-2_3
11. Bendlin, R., Damgård, I.: Threshold decryption and zero-knowledge proofs for lattice-based cryptosystems. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 201–218. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-11799-2_13
12. Bendlin, R., Krehbiel, S., Peikert, C.: How to share a lattice trapdoor: threshold protocols for signatures and (H)IBE. In: Jacobson, M., Locasto, M., Mohassel, P., Safavi-Naini, R. (eds.) ACNS 2013. LNCS, vol. 7954, pp. 218–236. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38980-1_14
13. Boneh, D., Boyen, X.: Secure identity based encryption without random oracles. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 443–459. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28628-8_27

14. Boneh, D., et al.: Threshold cryptosystems from threshold fully homomorphic encryption. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10991, pp. 565–596. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_19
15. Boneh, D., Gennaro, R., Goldfeder, S., Kim, S.: A lattice-based universal thresholdizer for cryptographic systems. Cryptology ePrint Archive: Report 2017/251, September 2017
16. Boneh, D., Lewi, K., Montgomery, H., Raghunathan, A.: Key homomorphic PRFs and their applications. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 410–428. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_23
17. Boneh, D., Montogomery, H., Raghunathan, A.: Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In: ACM-CCS (2010)
18. Cachin, C., Kursawe, K., Shoup, V.: Random oracles in constantinople: practical asynchronous byzantine agreement using cryptography. In: PODC (2000)
19. Canetti, R., Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Adaptive security for threshold cryptosystems. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 98–116. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_7
20. Canetti, R., Goldwasser, S.: An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack (Extended Abstract). In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 90–106. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_7
21. Chase, M., Meiklejohn, S.: Déjà Q: using dual systems to revisit q-type assumptions. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 622–639. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_34
22. Cramer, R., Damgård, I., Dziembowski, S., Hirt, M., Rabin, T.: Efficient multiparty computations secure against an adaptive adversary. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 311–326. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_22
23. Cramer, R., Fehr, S.: Optimal black-box secret sharing over arbitrary abelian groups. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 272–287. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_18
24. Damgård, I., Thorbek, R.: Linear integer secret sharing and distributed exponentiation. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 75–90. Springer, Heidelberg (2006). https://doi.org/10.1007/11745853_6
25. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 307–315. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_28
26. Dodis, Y.: Exposure-Resilient Cryptography. Ph.D. thesis, MIT (2000)
27. Dodis, Y.: Efficient construction of (distributed) verifiable random functions. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 1–17. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36288-6_1
28. Dodis, Y., Yampolskiy, A.: A verifiable random function with short proofs and keys. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 416–431. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-30580-4_28
29. Dodis, Y., Yampolskiy, A., Yung, M.: Threshold and proactive pseudo-random permutations. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 542–560. Springer, Heidelberg (2006). https://doi.org/10.1007/11681878_28

30. Frankel, Y., MacKenzie, P., Yung, M.: Adaptively-secure distributed public-key systems. In: Nešetřil, J. (ed.) ESA 1999. LNCS, vol. 1643, pp. 4–27. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48481-7_2
31. Freire, E.S.V., Hofheinz, D., Paterson, K.G., Striecks, C.: Programmable hash functions in the multilinear setting. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 513–530. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_28
32. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of STOC, pp. 197–206. ACM (2008)
33. 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. LNCS, vol. 8042, pp. 75–92. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_5
34. Goldreich, O.: Valiant's polynomial-size monotone formula for majority (2014)
35. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. J. ACM **33**, 792 (1986)
36. Goldwasser, S., Kalai, Y., Peikert, C., Vaikuntanathan, V.: Robustness of the learning with errors assumption. In: ICS (2010)
37. Gorbunov, S., Vaikuntanathan, V., Wichs, D.: Leveled fully homomorphic signatures from standard lattices. In: STOC (2015)
38. Goyal, R., Hohenberger, S., Koppula, V., Waters, B.: A generic approach to constructing and proving verifiable random functions. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017. LNCS, vol. 10678, pp. 537–566. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70503-3_18
39. Hastad, J., Impagliazzo, R., Levin, L., Luby, M.: A pseudorandom generator from any one-way function. SIAM J. Comput. **8**(4), 1364–1396 (1999)
40. Hoory, S., Magen, A., Pitassi, T.: Monotone circuits for the majority function. In: Díaz, J., Jansen, K., Rolim, J.D.P., Zwick, U. (eds.) APPROX/RANDOM -2006. LNCS, vol. 4110, pp. 410–425. Springer, Heidelberg (2006). https://doi.org/10.1007/11830924_38
41. Jager, T.: Verifiable random functions from weaker assumptions. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015. LNCS, vol. 9015, pp. 121–143. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_5
42. Jarecki, S., Lysyanskaya, A.: Adaptively secure threshold cryptography: introducing concurrency, removing erasures. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 221–242. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45539-6_16
43. Katsumata, S., Yamada, S.: Partitioning via non-linear polynomial functions: more compact IBEs from ideal lattices and bilinear maps. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016. LNCS, vol. 10032, pp. 682–712. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53890-6_23
44. Lewko, A., Waters, B.: Efficient pseudorandom functions from the decisional linear assumption and weaker variants. In: ACM-CCS (2009)
45. Libert, B., Joye, M., Yung, M.: Born and raised distributively: fully distributed non-interactive adaptively secure threshold signatures with short shares. In: PODC (2014)
46. Lysyanskaya, A., Peikert, C.: Adaptive security in the threshold setting: from cryptosystems to signature schemes. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 331–350. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_20

47. Micali, S., Sidney, R.: A simple method for generating and sharing pseudo-random functions, with applications to clipper-like key escrow systems. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 185–196. Springer, Heidelberg (1995). https://doi.org/10.1007/3-540-44750-4_15
48. Micciancio, D., Peikert, C.: Trapdoors for lattices: simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_41
49. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on Gaussian measures. SIAM J. Comput. **37**(1), 267–302 (2007)
50. Naor, M., Pinkas, B., Reingold, O.: Distributed pseudo-random functions and KDCs. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 327–346. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_23
51. Naor, M., Reingold, O.: Number-theoretic constructions of efficient pseudo-random functions. In: FOCS (1997)
52. Naor, M., Reingold, O., Rosen, A.: Pseudo-random functions and factoring. In: STOC (2000)
53. Nielsen, J.B.: A threshold pseudorandom function construction and its applications. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 401–416. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_26
54. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: STOC, pp. 187–196. ACM (2008)
55. Raghunathan, A., Segev, G., Vadhan, S.: Deterministic public-key encryption for adaptively chosen plaintext distributions. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 93–110. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38348-9_6
56. Razborov, A., Rudich, S.: Natural proofs. J. Comput. Syst. Sci. **55**(1), 24–35 (1987)
57. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC (2005)
58. Regev, O., Stephens-Davidowitz, S.: A reverse Minkowski theorem. In: STOC (2017)
59. Thorbek, R.: Linear integer secret sharing. Ph.D. thesis, Department of Computer Science - University of Arhus (2009)
60. Trevisan, L., Vadhan, S.: Extracting randomness from samplable distributions. In: FOCS (2000)
61. Valiant, L.: Short monotone formulae for the majority function. J. Alg. **5**(3), 363 (1984)
62. Valiant, L.: A theory of the learnable. Commun. ACM 27(11), 1134 (1984)
63. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_7