# Bounded-Equivocable Pseudorandom Functions

Paul Gerhart, Davide Li Calsi, Luigi Russo, and Dominique Schröder

TU Wien, Vienna, Austria
{paul.gerhart, davide.li.calsi, luigi.russo, dominique.schroeder}@tuwien.ac.at

**Abstract.** We introduce Bounded-Equivocable PRFs, a new variant of pseudorandom functions. They combine standard pseudorandomness with a bounded form of programmability. In our model, an adversary may issue an arbitrary number of queries that remain indistinguishable from random. Bounded equivocability ensures that responses can be programmed consistently with a later-revealed key, up to a fixed bound $q$. This relaxation avoids known impossibility results, which preclude polynomial unbounded equivocability in the standard model, while preserving the programmability required for applications.

We present standard-model constructions of bounded-equivocable PRFs under the DDH and LWE assumptions, and we show how to make these constructions verifiable. Prior SIM-AC style primitives could not achieve verifiability since their programmability relied on embedding the secret key into the random oracle.

We demonstrate applications to (i) adaptively secure private-key encryption, (ii) two-round threshold Schnorr signatures secure against adaptive corruptions, and (iii) leader election in Proof of Stake blockchains. Together, these results establish bounded-equivocable PRFs as a practical primitive that achieves programmability with verifiability in the standard model, and enables applications previously out of reach.

## 1 Introduction

A pseudorandom function (PRF) family $F = \{F_k\}_{k\in\mathcal{K}}$ is one of the most fundamental primitives in modern cryptography. Intuitively, a PRF is a deterministic and efficiently computable function that nevertheless "looks random" to any efficient adversary. Formally, for every probabilistic polynomial-time distinguisher $\mathcal{D}$, the advantage in distinguishing oracle access to $F_k$, for a uniformly chosen key $k$, from oracle access to a uniformly random function of the same domain and range, is negligible. This simple yet powerful guarantee has made PRFs indispensable in the design of symmetric encryption, message authentication, key derivation, and numerous advanced cryptographic protocols [KL21,BKR94,BCK96,BLMR13].

Recent work has emphasized security under adaptive corruptions, a notoriously difficult problem for most of the primitives of interest. For PRFs, corruption of an evaluator trivially reveals the secret key, undermining any direct claim of pseudorandomness. This limitation complicates the use of PRFs as building blocks in larger protocols. A standard proof step is to replace PRF outputs with uniform random values, but this becomes infeasible in the adaptive setting. If the adversary later corrupts a party, the simulator may be unable to explain past outputs with a consistent key.

At CRYPTO 2020, Jaeger and Tyagi [JT20a] introduced the notion of Simulatable under Adaptive Corruptions (SIM-AC) PRFs. Such PRFs allow the reduction to *program* answers consistently with later-revealed keys. While a trivial construction exists in the Random Oracle Model (ROM), strong impossibility results [JT20a] rule out standard-model instantiations. Due to the uninstantiatability results of Canetti, Goldreich and Halevi [CGH98] and Goldwasser and Kalai [GK03] this leaves a major gap. Programmability is essential for adaptive security proofs, yet current techniques either rely on idealized random oracles or run into impossibility barriers.

*Bounded-Equivocable PRFs.* We propose a new variant of pseudorandom functions called *Bounded-Equivocable PRFs*. This notion retains the indistinguishability from randomness of standard PRFs, but strengthens them by being simulatable under a limited number of adaptive corruptions. More precisely, the adversary's interaction with the function proceeds in two phases. In the first phase, the adversary may query the PRF up to

$q$ times, and as in the classical setting, the responses are computationally indistinguishable from random. In the second phase, the adversary may trigger a corruption and obtain the secret key $k$, which certifies that the earlier function values were computed correctly. The key feature is that even under this stronger form of exposure, the primitive supports a controlled form of *equivocation*: during the query phase, the challenger may program the answers arbitrarily, yet later still provide a key that validates them as genuine function evaluations.

Furthermore, in the absence of corruptions our primitive remains pseudorandom for a polynomially many queries. This combination of properties circumvents the impossibility results, and is achievable in the standard model under standard assumption. With this, we overcome the verifiablity-related limits of previous ROM-based SIM-AC PRFs. Specifically, it is unfeasible to formally capture the random oracle inside a proof of evaluation over secret inputs due to its exponential-size description, so soundness of the resulting proof no longer holds. At the same time, programmability of the oracle is essential within such proofs: the simulator must be able to adapt answers consistently with later information. By imposing an a priori bound on the number of queries, we obtain a primitive that admits programmability while still having a concise description suitable for use inside zero-knowledge proofs systems, thus enabling bounded but sound standard-model instantiations of random oracles.

The distinctive properties of our primitive extend beyond mere theoretical interest and enable a wide range of applications, most notably in the fully adaptive setting. Adaptive security formalizes a powerful scenario in which an adversary can corrupt parties during the execution of a protocol, with whom to corrupt potentially depending on previously observed transcripts. While this captures realistic threats, it is also notoriously difficult to achieve and in fact, it is impossible for some tasks. Conceptually, the model reflects systems with many users, each holding private state, where an adversary can interact with the system and then later decide whose secrets to compromise. While this abstraction is broad, it captures concrete scenarios. For instance, in the searchable encryption scheme of Cash et al. [CJJ+14], keywords act as "users," each associated with a secret key. To perform a search, the protocol must reveal the key for the corresponding keyword, thereby creating an avenue for adaptive compromise. Variants of this setting have been explored under different terminology, including adaptive corruption [CFGN96,JT20a], non-committing encryption [CFGN96,CPR17,BBD+20], and selective-opening attacks [BDWY12,BHY09,BHK12,HPW15,HRW16]. Despite differences in formulation, these lines of work all address the same core difficulty: maintaining meaningful security even when secrets are disclosed *after* interaction with the adversary.

Recently, the issue of adaptive security has become a key topic in the digital signature community, particularly in regard to threshold Schnorr signatures. Schnorr signatures are one of the most widely studied and deployed digital signature schemes, valued for their simplicity, efficiency, and provable security in the ROM. In addition to being already in use in Bitcoin (via the Taproot upgrade), they are currently under consideration for standardization by the IETF. The threshold variant of Schnorr signatures is highly attractive in practice, as it distributes trust, mitigates single points of failure, and provides robustness against the compromise of individual parties. Proving adaptive security for threshold Schnorr signatures, however, is highly non-trivial. In the adaptive setting, an adversary may corrupt parties during the signing protocol, obtaining their secret shares *after* having observed partial transcripts. Ensuring that signatures remain unforgeable even under such exposures requires strong simulation techniques, since the simulator must be able to explain past protocol messages consistently with the subsequently revealed keys. Traditional proof techniques for Schnorr signatures, while sufficient in the static-corruption model, fall short in the adaptive case. Our bounded-equivocable PRFs provide exactly the needed flexibility: they allow partial transcripts to be *programmed* during the protocol execution, while still admitting a secret key that validates them in retrospect. This property makes it possible, for the first time, to obtain adaptively secure two-round threshold Schnorr signatures in the standard model.

### 1.1 Our Contributions

Our work introduces and explores the notion of *Bounded-Equivocable PRFs* and demonstrates its wide applicability across multiple domains. The main contributions are as follows:

– **A Relaxed Notion of SIM-AC.** We introduce a relaxation of SIM-AC-security that preserves *unbounded pseudorandomness* while imposing *bounded equivocability*. Concretely, adversaries may issue an arbitrary number of queries that remain pseudorandom, but equivocation is guaranteed only up to a fixed bound $q$. This relaxation preserves the essential programmability property while avoiding known impossibility results.
– **Standard-Model Constructions.** We show that the relaxed definition is achievable in the standard model. In particular, we present two concrete instantiations based on well-studied hardness assumptions: one from the Decisional Diffie–Hellman (DDH) assumption and one from the Learning With Errors (LWE) assumption. These constructions establish that bounded-equivocable PRFs are both conceptually and practically realizable.
– **Verifiability of Constructions.** We extend our constructions to achieve verifiability, meaning that outputs of the PRF can be efficiently checked for correctness relative to the provided key. Previous fully SIM-AC approaches could not achieve this property, since their programmability relied on embedding the secret key into the random oracle, precluding efficient verification. Our bounded notion overcomes this limitation.
– **Applications Across Domains.** We demonstrate the utility of bounded-equivocable PRFs through three distinct applications: (i) *Threshold Cryptography*, where we construct the first adaptively secure two-round threshold Schnorr signatures. (ii) *Private-Key Encryption*, where we obtain adaptively secure encryption schemes in the standard model. (iii) *Proof of Stake*, where we leverage the pseudorandomness and verifiability of bounded-equivocable PRFs for leader election. These case studies highlight the breadth of the primitive and its potential as a foundational tool.

Together, these results establish bounded-equivocable PRFs as a versatile and realizable primitive that combines programmability with verifiability, opening the door to new applications both in theory and in practice.

### 1.2 Related Work

PRFs were introduced in the seminal work of Goldreich, Goldwasser, and Micali [GGM84], who showed how to construct them from any secure pseudorandom generator. Subsequent research explored more efficient and structure-rich instantiations. Naor and Reingold [NRR00,NR97] introduced number-theoretic PRFs based on modular exponentiation, while lattice-based alternatives have emerged more recently via the Learning With Rounding problem [BPR12,AKPW13,CS19]. These developments establish PRFs as both a theoretical cornerstone and a practical building block for post-quantum secure cryptography.

The concept of SIM-AC PRF was introduced in the seminal work of Jaeger and Tyagi [JT20a], who also gave constructions in idealized models such as the Random Oracle Model or the Ideal Cipher Model. Few years later Jaeger [Jae23] upgraded the definition to SIM$^*$-AC, by giving more power to the attacker and reducing that of the simulator. This is motivated by the difficulty of proving interesting results through the SIM-AC definition alone, in particular the seemingly natural implication from single-user security to multi-user security. Recent work by Cheng and Jaeger [CJ25] leveraged the aforementioned definitions to achieve adaptive security for constrained PRFs. Both [JT20a] and [Jae23] extended the respective definitions beyond PRFs, e.g. by considering SIM-AC and SIM$^*$-AC notions for CPA-secure encryption. In concurrent work, Jarecki, Krawczyk, and Xu proposed a similar notion of equivocal encryption [JKX18], which Jaeger and Tyagi later demonstrated to be implied by their SIM-AC-CPA definition. Both SIM-AC and SIM$^*$-AC PRFs are not achievable in the standard model. Jaeger and Tyagi [JT20b] provide a proof in the extended version of their CRYPTO paper, based on the ideas put forth by Nielsen [Nie02]. This entails that the programmability of the ROM is required to achieve them, which in turn hinders sound proof of correct evaluations over their instances.

## 2 Technical Overview

Adaptive security is of central practical importance. Real-world systems are inherently multi-user and it is infeasible to predict in advance which parties an adversary may corrupt. Security proofs in this setting

typically rely on programmability. The key difficulty is that an adversary may corrupt a party *after* the reduction has already programmed some primitive. In that case all previously given answers must remain consistent, or the proof immediately fails.

The challenge is even greater when programmable primitives are combined with zero-knowledge proofs, where consistency requirements are particularly stringent. Modeling a hash function as a random oracle permits adaptive programming of $H$ within proofs, but the random oracle model cannot itself be part of the statement to be proven. This motivates the search for standard-model realizations that provide programmability while supporting sound proofs.

### 2.1 Equivocable PRFs

Our starting point is a PRF augmented with a programmability interface. The standard evaluation returns $y \leftarrow \mathsf{PRF}(k, x)$. A second, keyless interface

$$(y', \mathsf{st}') \leftarrow \mathsf{EqEval}(\mathsf{st}, x)$$

allows the reduction to program the mapping $(x, y')$ without knowing or specifying $k$, while evolving some internal state $\mathsf{st}'$. Upon corruption the reduction must output a key $k'$ that is consistent with all previously programmed pairs, that is,

$$k' \leftarrow \mathsf{EqKey}\big((x_i, y_i)_i, \mathsf{st}\big) \quad \text{such that} \quad \forall i : \; y_i = \mathsf{PRF}(k', x_i).$$

Unbounded programmability is impossible to realize in the standard model: allowing an arbitrary number of programmed queries recovers the simulation-based adaptive compromise (SIM-AC) barrier of Jaeger and Tyagi [JT20a], which is provably achievable in idealized models only. This motivates our relaxation to *bounded-equivocable PRFs*, where pseudorandomness remains unbounded but equivocation is required only for a fixed number $q$ of programmed queries.

Our formalization differs from [JT20a] in scope. We define security in the single-user setting, whereas SIM-AC is formulated for multiple users. Jaeger and Tyagi noted that lifting single-user guarantees to the multi-user is delicate in their setting due to a shared ideal primitive that simulators must control. Subsequent work [Jae23] strengthened the adversary to address these tensions. In our standard model treatment, there is no shared programmable ideal primitive and therefore, multi-user security follows from the single-user case via a standard hybrid argument.

*Why Existing Approaches Fail.* Before turning to our constructions, it is helpful to understand why natural candidates do not meet our goals. A first attempt is to rely on the ROM, as in the simple construction of Jaeger and Tyagi [JT20a], where $F(k, x) = \mathsf{H}(k, x, ol)$ and $ol$ is an output-length term depending on the security parameter. This trivially provides both pseudorandomness and unbounded equivocation, but only in the ROM, and therefore falls outside our standard-model focus.

A second approach is to use $t$-wise independent functions, which offer bounded pseudorandomness and equivocation by evaluating a random polynomial of sufficiently high degree. While this yields limited programmability, the number of queries is inherently tied to the polynomial degree, and hence such functions cannot serve as fully pseudorandom PRFs.

Finally, one might hope that existing standard-model PRFs could suffice. Indeed, many known PRFs provide unbounded pseudorandomness, but they only admit very limited equivocation. For example, the ROM-based construction $F(k, x) = \mathsf{H}(x)^k$ is a secure PRF under DDH [NPR99], yet it has equivocability threshold one: the first output already binds the key. Even increasing the key length is not enough. As we illustrate with the Naor–Reingold PRF, adversaries can adaptively choose inputs so that outputs depend on only a small portion of the key, forcing inconsistencies after only a handful of queries.

These examples highlight the core difficulty: achieving higher equivocability is not a matter of simply enlarging the key, but of carefully managing how key material contributes to each evaluation. Designing a PRF that is both fully pseudorandom and equivocable beyond trivial thresholds requires fundamentally new ideas.

**Bounded-Equivocable PRFs from DDH** Our starting point is the Naor–Reingold PRF [NRR00,NR97], because the scheme has $\lambda$ independent key components that give us enough degrees of freedom to leverage for equivocability. Concretely, the function is defined as

$$\mathsf{PRF}(\boldsymbol{k}, \boldsymbol{x}) = g^{k_0 \prod_{i=1}^{\ell} k_i^{x_i}}$$

where the key is $\boldsymbol{k} = (k_0, k_1, \ldots, k_\ell) \in \mathbb{Z}_q^{\ell+1}$ and the input $x \in \{0,1\}^\ell$ is interpreted bitwise.

*Programming strategy.* To return a programmed output on some input $\boldsymbol{x}$, the simulator samples a random exponent $r \leftarrow\!\!\$\, \mathbb{Z}_q$ and outputs $g^r$. Since $r$ is uniform, so is $g^r$. At the same time, the simulator records the constraint

$$k_0 \cdot \prod_{j \in I_1(\boldsymbol{x})} k_j \equiv r \pmod{q},$$

where $I_1(\boldsymbol{x}) = \{i : x_i = 1\}$. Across multiple queries, the simulator thus accumulates equations relating the key components to the programmed exponents.

*Equivocation intuition.* If the adversary later corrupts the key, the simulator must output a key vector $\boldsymbol{k}'$ that satisfies all recorded constraints. As long as the number of equations is strictly less than the number of unknowns, one would expect that such a system is satisfiable with high probability. Intuitively, this means that the $\ell + 1$ key components provide enough degrees of freedom to reconcile up to $\ell$ programmed queries, giving bounded but nontrivial equivocability.

*Attacking the Naor–Reingold PRF.* The above intuition fails under adaptive queries: answering with uniformly random outputs does not guarantee that a consistent key exists later. To see this, consider the naïve Naor–Reingold PRF, and let the simulator program the answers as suggested above. An adversary needs only four queries—$(0,0,0,\ldots,0), (1,0,0,\ldots,0)$, $(0,1,0,\ldots,0)$, and $(1,1,0,\ldots,0)$—to force the constraints

$$r_0 = k_0, \quad r_1 = k_0 k_1, \quad r_2 = k_0 k_2, \quad r_3 = k_0 k_1 k_2 \pmod{q},$$

where $r_i$ are the programmed exponents. Consistency requires

$$r_0 \cdot r_3 \;\equiv\; r_1 \cdot r_2 \pmod{q}.$$

For uniformly random $r_i$, this relation holds with probability about $1/q$, so with overwhelming probability the simulator cannot find a consistent key.

*Adapting the Naor–Reingold PRF.* The root cause is that the adversary can deterministically shape the constraint system by adaptively choosing inputs that ignore portions of the key. This collapses the number of effective degrees of freedom and induces contradictions after only a handful of queries. To prevent such manipulations, we must randomize the constraint system itself. Our approach is to preprocess each input $\boldsymbol{x}$ through an independent length-preserving PRF, so that the indices of the key involved in each constraint appear pseudorandom. Intuitively, this ensures that the resulting system resembles one generated by random incidence patterns, which is satisfiable with high probability as long as the number of equations remains below the number of unknowns.

*Proving Security.* Assume that the inputs supplied to the construction are uniformly random. Each adversarial query then introduces a constraint of the form $k_0 \prod_{j \in I_1(\boldsymbol{x})} k_j = z$. Although the constraint is multiplicative, it can be converted into an additive form over $\mathbb{Z}_{q-1}$. This is possible because the multiplicative group $(\mathbb{Z}_q^*, \cdot)$ is cyclic when $q$ is prime, and therefore admits a generator $h$ such that every $k_i \in \mathbb{Z}_q^*$ can be written as $k_i = h^{y_i}$. Although the PRF is naturally defined over $\mathbb{Z}_q$, we can safely work in $\mathbb{Z}_q^*$ because when the key elements are chosen uniformly from $\mathbb{Z}_q$ the probability of obtaining 0 is $1/q$, which is negligible. Under this encoding, the original constraint becomes

$$y_0 + \sum_{j \in I_1(x)} y_j = \log_h(z) \mod (q-1),$$

where $y_i = \log_h(k_i)$ and $\log_h(z)$ denotes the discrete logarithm of $z$ with respect to $h$. That is, $y_0 + \sum_{j|x_j=1} y_j = r \mod (q-1)$, where $h^r = z \in \mathbb{Z}_q^*$. Considering $L$ such constraints and $\lambda + 1$ unknowns, we obtain a linear system over $\mathbb{Z}_{q-1}$ of the form $Ay \equiv R \mod (q-1)$, where $A = (\mathbf{1}|D)$, where $\mathbf{1}$ denotes a column of $L$ 1s and $D$ is a random $L \times \lambda$ binary matrix, while $R$ stacks the corresponding $r$-values. Because $q$ is prime, we have that $q-1$ is composite. Consequently, $\mathbb{Z}_{q-1}$ is in general *not* a field, and hence not all nonzero elements are invertible. This makes it more difficult to characterise the structure of the solution space and prevents the use of standard linear algebraic techniques.

To simplify our proof, we now make the assumption that $q = 2q' + 1$ is a safe prime. Then, by the Chinese Remainder Theorem (CRT), we can break down the system into two independent systems modulo 2 and $q'$, and analyze them separately. Through a combinatorial analysis, we show that both systems are consistent with overwhelming probability for appropriately chosen parameters.

At a high level, this is achieved by lower-bounding the number of full-rank random binary matrices modulo 2 and $q'$, and showing that they represent the vast majority of all such matrices. We present the counting argument modulo 2 and defer the more complex (though similar in spirit) argument modulo $q'$ to the main body of the paper.

Imagine, for a moment, that we generate a full-rank binary matrix of size $m \times n$ row-by-row. The first row can be any non-zero vector in $\mathbb{F}_2^n$, for a total of $2^n - 1$ possibilities. When generating the second row, we must avoid any linear combination of the first row. Over $\mathbb{F}_2$, there are only two such combinations (the zero vector and the row itself), leaving $2^n - 2$ valid choices. In general, when generating the $i$-th row, we must avoid any binary linear combination of the first $i-1$ rows, leaving $2^n - 2^{i-1}$ possibilities.

By multiplying the number of choices for each of the $m$ rows, we obtain the total number of full-rank binary matrices. Dividing this by $2^{mn}$ yields the probability that a random binary matrix is full-rank:

$$\Pr[\text{full rank}] = \prod_{i=0}^{m-1} \left(1 - \frac{1}{2^{n-i}}\right).$$

The argument modulo $q'$ is similar in foundation, but considerably more involved, due to the fact that the linear combinations to avoid are no longer binary, but involve coefficients in $\mathbb{Z}_{q'}$. Since we still use binary vectors to generate each row, it becomes intractable to determine exactly which (or how many) $q'$-ary linear combinations of binary vectors remain binary. We therefore resort to a different approach: Specifically, instead of exact counting, we argue that the measure of linearly dependent cases is vanishingly small for chosen parameters, and adjust the relevant parameters to ensure a suitable (overwhelming) lower bound on the probability that the matrix is full-rank modulo $q'$.

Once both systems are known to be consistent, we can solve them efficiently using standard Gaussian elimination, and combine their solutions via CRT. We show that each of these steps is computationally efficient and, in particular, does not require solving discrete logarithms. Moreover, we demonstrate that the resulting key is statistically indistinguishable from an honestly generated one.

The indistinguishability argument again proceeds separately modulo 2 and modulo $q'$. In both cases, it focuses on the joint distribution of the equivocated key and the supplied evaluations, and shows that it is statistically close to that of an honestly generated key and evaluations. The proof modulo 2 leverages the well-known Leftover Hash Lemma [BDK+11,HILL99] and is feasible because the aforementioned random binary matrix acts as a uniformly random universal hash function. The same argument does not directly carry over modulo $q'$, but we still exploit known results from linear algebra to argue statistical indistinguishability in that case. Finally, a simple CRT-based lemma allows us to combine the two indistinguishability results modulo 2 and $q'$ to conclude indistinguishability modulo $q$, as desired.

*From Randomness to Pseudorandomness.* The assumption of uniformity simplifies the proof. In practice, we realize this by employing a $t$-wise independent function (or equivalently an internal length-preserving PRF) to randomize the input $x$ before feeding it into the Naor–Reingold PRF. The pseudorandomness of this inner function ensures that no PPT adversary can trigger inconsistencies with non-negligible probability, as implied by the statistical analysis on the rank of random matrices together with the efficient computability of

ranks. Consequently, any efficient adversary causing inconsistencies could be transformed into a distinguisher between PRFs and random functions.

*Instantiating Groups.* We can instantiate the PRF over elliptic-curve groups, resulting in outputs that are elliptic curve points. This yields clear advantages in terms of computational efficiency, practicality, and confidence in the hardness of the Decisional Diffie-Hellman (DDH) problem in the target group.

If instead a pseudorandom scalar output is desired, we can instantiate the PRF in the group of quadratic residues modulo $2q+1$, provided that $2q+1$ is prime. Specifically, this group consists of the $q$ distinct elements $\{1^2, 2^2, \ldots, q^2\} \mod 2q+1$, equipped with multiplication modulo $2q+1$. Once the PRF outputs a quadratic residue, it can be converted into a scalar by computing its modular square root, which is efficiently computable modulo primes.

This approach works as long as $(q', q, 2q+1)$ form a Cunningham chain of length three.[1] Alternatively, one can leverage known results on randomness extraction from elliptic curves defined over quadratic extension fields $\mathbb{F}_{p^2}$. In particular, given the $x$-coordinate $x_0 + x_1 Z$ (interpreted as an element of $\mathbb{F}_p[Z] \cong \mathbb{F}_{p^2}$) of a random point on such a curve, it is known that $x_0$ is statistically close to uniform over $\mathbb{F}_p$. This method is often more practical than the approach based on Cunningham chains, especially since the latter requires working modulo large integers to retain cryptographic security.

**Bounded-Equivocable PRFs from LWR.** A similar idea leads to a post-quantum construction based on the Learning With Rounding (LWR) problem, which itself inherits hardness from the Learning With Errors (LWE) assumption. Given a key $k \in \mathbb{Z}_q^\ell$, the LWR function maps an input $x \in \mathbb{Z}_q^\ell$ to $\lfloor \langle k, x \rangle \rfloor_p = \left\lfloor \frac{\langle k,x \rangle \cdot p}{q} \right\rfloor$, where the inner product is computed over $\mathbb{Z}_q$, and the result is rounded to a smaller modulus $p$.

We again assume that the input $x$ is randomized via e.g., an internal, length-preserving PRF mapping $x$ to $y \leftarrow \mathsf{H}(\boldsymbol{k}_{in}, x)$. Then, if the LWR outputs are defined as $\lfloor y \rfloor_p$ with $y \leftarrow\!\!\$\ \mathbb{Z}_q$, the problem of finding a consistent key again reduces to solving a linear system $A \cdot K \equiv Y \mod q$, where $A$ is a random matrix over $\mathbb{Z}_q$ and $Y$ is obtained by stacking the corresponding $y$ values.

Similar to the DDH-based construction, we start from the simplified case of uniformly random inputs (ignoring $\mathsf{H}$). Then $A$ can be shown to be full-rank with overwhelming probability by a combinatorial argument, as long as the number of rows does not exceed the number of columns [LN97, Sec 3.3]. The argument once again counts the number of full-rank matrices by generating them row-by-row, ensuring at each step that we do not get a linear combination of the previous rows. With this strategy, one can show that a random $n \times n$ matrix is invertible with probability at least $1 - n/q$. Furthermore, similar to the DDH instance, we can efficiently solve the system and guarantee that the joint distribution of the key and the PRF evaluation is statistically close to uniformly. Finally, by leveraging the pseudorandomness of $\mathsf{H}$, we can show that the same arguments would carry over if the latter is used as a preprocessing block. Furthermore, in this case $\mathsf{H}$ randomizes $x$, and ensures that the outer *weak* PRF is always evaluated on uniformly random inputs, thus ensuring the overall pseudorandomness of the construction.

We can slightly increase the number of allowed evaluations by leveraging tools from lattice theory. Intuitively, the previous requirement of exact consistency is overly restrictive: to ensure compatibility with the randomized evaluations, it suffices that the scalar products $\langle A_i, K \rangle$ are *close enough* to the values $y_i$, equality is unnecessary.Recall that the LWR function conceptually partitions the interval $[0, q)$ into $p$ contiguous subintervals, each of approximate length $q/p$, and labels them from 0 to $p-1$. The operation $\lfloor x \rfloor_p$ then assigns to $x$ the label of the subinterval it belongs to. As a consequence, it suffices for $\langle A_i, K \rangle$ and $y_i$ to fall within the same subinterval, i.e., to differ by at most $q/2p$.

Let $\Lambda_q(A)$ denote the lattice generated by the rows of $A$ modulo $q$, that is, the set of all integer linear combinations of the rows of $A$ (over $\mathbb{Z}$), reduced modulo $q$. Our goal then reduces to finding a lattice point $v \in \Lambda_q(A)$ such that $\|v - y\| \leq q/2p$. Due to the randomness of $A$, it suffices to show that such a point exists with high probability.

[1] Cunningham chains are sequences of primes where $p_i = 2p_{i-1} + 1$. No theoretical guarantees on their existence are known, but empirical evidence and dedicated algorithms suggest that such chains of cryptographic size can be found in practice [YY98].

To this end, we appeal to known results on the *covering radius* $\mu$ [Rog64] of random lattices $\Lambda_q(A)$. The covering radius is defined as the largest distance between any point in $\mathbb{R}^\ell$ and the closest lattice point. Our proof leverages probabilistic bounds on $\mu$ to show that, with overwhelming probability, the covering radius does not exceed $q/2p$. Since $\mu$ bounds the distance between any target point $y$ and the lattice $\Lambda_q(A)$, we conclude that a consistent key exists.

However, finding such a key is no longer trivial. As established above, the problem reduces to decoding in the lattice $\Lambda_q(A)$. The closest decoding problem in the available litterature is the *Bounded Distance Decoding* (BDD) problem [MR07] with target $t$ and lattice $\Lambda_q(A)$. This problem is central in lattice-based cryptography, as it underpins several worst-case to average-case reductions. BDD is only well-defined as long as the ratio between the target-to-lattice distance and the lattice's minimum distance does not exceed $1/2$, as otherwise the uniqueness of the solution is not guarantees. It turns out that in the parameter regime of our construction this is not the case; thus, we defer a thorough investigation of the feasibility of decoding in our setting to future work.

### 2.2 Bounded Equivocable Exponent VRFs

We apply our paradigm to the domain of *exponent verifiable random functions (eVRFs)*. Exponent VRFs, introduced by Nick *et al.* [NRSW20] and later formalized by Boneh *et al.* [BHLS25], serve as a bridge between pseudorandom functions (PRFs) and verifiable random functions (VRFs). An eVRF is associated with a public key $K = g^k$, and on input $\hat{m}$ enables the computation

$$(\hat{R}, \hat{r}, \pi) \leftarrow \mathsf{eVRF}(k, \hat{m}), \quad \text{where } \hat{r} \leftarrow \mathsf{PRF}(k, \hat{m}), \quad (\hat{R}, \pi) \leftarrow \mathsf{VRF}(k, \hat{m}),$$

such that $\hat{R} = g^{\hat{r}}$. The proof $\pi$ certifies that $\hat{r}$ was correctly derived under the secret key $k$, and verification is performed using the public key $K$.

Any PRF can be transformed into an eVRF through three steps: (1) evaluate the PRF on input $\hat{m}$ to obtain a pseudorandom scalar $y \in \mathbb{Z}_p$; (2) map this scalar into the target group via exponentiation, yielding $Y = g^y$; and (3) produce a proof $\pi$ certifying both the correctness of the PRF evaluation and the consistency of the exponentiation, i.e., that $Y$ indeed equals $g^y$ for the hidden PRF output $y$.

Using our paradigm in this context yields a notable advantage: it eliminates the need for hash function evaluations. This makes the construction particularly appealing as a building block in larger protocols, such as, for example, two-round Schnorr threshold signature schemes. In fact, our instantiation can directly replace the eVRF component in the recent protocol of Gerhart *et al.* [GLCRS25], resulting in a cleaner design with comparable computational overhead.

Finally, we demonstrate that the transformation admits efficient instantiations in both classical and post-quantum settings. In the DDH-based setting, a standard combination with Bulletproofs [BBB+18] suffices to efficiently prove the correctness of exponentiation. In the lattice-based setting, we introduce a tailored encoding mechanism to handle rational values prior to rounding, thereby enabling compact and efficient proofs of correctness.

## 3 Preliminaries

*Notation.* We denote by $1^\lambda$ the security parameter encoded in unary, and by $x \leftarrow\!\!\$\ \mathcal{X}$ the uniform sampling of $x$ from a finite set $\mathcal{X}$. The notation $y \leftarrow \mathcal{A}(x)$ indicates that $y$ is the output returned by the (possibly randomized) algorithm $\mathcal{A}$ on input $x$. An algorithm $\mathcal{A}$ is called *probabilistic polynomial-time (PPT)* if it is a probabilistic Turing machine that runs in time polynomial in the security parameter $1^\lambda$. Throughout this work, we assume that all adversaries are PPT unless stated otherwise. For any $q \in \mathbb{N}$, we denote by $\mathbb{Z}_q$ the ring of integers modulo $q$. For any prime power $q$, we use $\mathbb{F}_q$ to denote the finite field of order $q$. For any real $x \in \mathbb{R}$, $\lfloor x \rfloor$ denotes the largest integer $x'$ such that $x' \leq x$. For lattice-based applications, we define the rounding function $\lfloor \cdot \rceil_p : \mathbb{Z}_q \to \mathbb{Z}_p$, where $p$ is a modulus such that $p < q$, as $\lfloor x \rceil_p = \left\lfloor x \cdot \frac{p}{q} \right\rceil$. We extend this function to vectors over $\mathbb{Z}_q$ by applying it component-wise.

*Auxiliary Lemmas and Definitions.* In the following we report standard definitions and results in the literature.

**Theorem 1 (Chinese Remainder Theorem).** *Let $m_1, \ldots, m_k$ be pairwise coprime positive integers and set*

$$M := \prod_{i=1}^{k} m_i.$$

*For any integers $a_1, \ldots, a_k$ the system of congruences*

$$x \equiv a_i \pmod{m_i} \qquad (i = 1, \ldots, k)$$

*has a solution $x$. Moreover the solution is unique modulo $M$: if $x$ and $x'$ are two solutions then $x \equiv x' \pmod{M}$.*

**Definition 1 (Universal Hash).** *Let $H$ be finite family of hash functions $h : \{0,1\}^n \to \{0,1\}^r$. $H$ is universal if for all $x, x' \in \{0,1\}^n$ such that $x \neq x'$*

$$\Pr[h(x) = h(x') | h \leftarrow\!\!\$\ H] = \frac{1}{2^r}.$$

**Lemma 1 (Leftover Hash Lemma [BDK+11,HILL99]).** *Let $H$ be finite family of universal hash functions $h : \{0,1\}^n \to \{0,1\}^r$. Let $X$ be a random variable supported over $\{0,1\}^n$ whose min-entropy is $H_\infty(X) = k$ and $G \sim \mathcal{U}(H)$, where $\mathcal{U}(\cdot)$ denotes the uniformly random distribution over its argument. Then*

$$\Delta\left[(G, G(X)), (G, \mathcal{U}(\{0,1\}^r))\right] \leq 2^{(r-k)/2}$$

## 4 Bounded Equivocable Pseudorandom Functions

In this section, we introduce the notion of *equivocable pseudorandom functions*. The central idea is to enhance the classical definition of pseudorandom functions with a form of *programmability*, reminiscent of how random oracles are programmed in cryptographic proofs. To this end, we define an additional interface $\mathsf{EqEval}$, which enables the explicit programming of specific input-output pairs. Our focus is on multi-party settings, where parties may be eventually corrupted, and their secret keys might be exposed. To address this, we further extend the PRF with an interface $\mathsf{EqKey}$ that allows computing a key consistent with prior evaluations. That is, even after $m$ evaluations or programmed queries, there exists an efficient algorithm that outputs a secret key such that the standard PRF evaluation on those $m$ inputs yields the previously defined outputs.

### 4.1 Definition of Bounded Equivocable Pseudorandom Functions

Pseudorandom functions (PRFs) [GGM84] are fundamental cryptographic primitives that aim to efficiently simulate a uniformly random function using a keyed, deterministic algorithm. Let $\mathcal{K}$ be the key space, $\mathcal{X}$ the input space, and $\mathcal{Y}$ the output space. We assume that all three are finite and efficiently samplable. Let $\mathsf{Func}_{\mathcal{X}\to\mathcal{Y}}$ denote the set of all functions $f : \mathcal{X} \to \mathcal{Y}$, which has cardinality $|\mathcal{Y}|^{|\mathcal{X}|}$ and includes functions that are not efficiently computable.

**Definition 2 (Pseudorandom Function).** *A* pseudorandom function (PRF) *is a pair of PPT algorithms* $\mathsf{PRF} = (\mathsf{KeyGen}, \mathsf{Eval})$ *with the following syntax:*

$k \leftarrow \mathsf{KeyGen}(1^\lambda)$**:** *The* key generation *algorithm outputs a key $k \in \mathcal{K}$ given a security parameter $\lambda$.*
$y \leftarrow \mathsf{Eval}(k, x)$**:** *The* evaluation *algorithm takes a key $k \in \mathcal{K}$ and input $x \in \mathcal{X}$, and returns an output $y \in \mathcal{Y}$.*

*The function is said to be* pseudorandom *if for all PPT adversaries $\mathcal{A}$, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that:*

$$\left| \Pr_{k \leftarrow \mathsf{KeyGen}(1^\lambda)} [\mathcal{A}^{\mathsf{Eval}(k,\cdot)}(1^\lambda) = 1] - \Pr_{f \leftarrow \$ \mathsf{Func}_{\mathcal{X} \to \mathcal{Y}}} [\mathcal{A}^{f}(1^\lambda) = 1] \right| \leq \mathsf{negl}(\lambda).$$

Equivocable PRFs capture the idea of "programming a function on demand" while still maintaining the façade of a genuine PRF. In practice, the need for such programmability arises when parties may later become corrupted and their long-term keys revealed. For example, in a secure multi-party protocol, one may wish to commit to a small number of outputs in advance, while still ensuring that an eventual exposure of the PRF key remains consistent with those outputs. Crucially, most protocols only require programming at a limited number of inputs, such as those that correspond to challenge values or committed labels, rather than across the entire domain. This motivates the notion of *bounded equivocability*: we allow the adversary to program at most $m$ outputs of its choice, after which a consistent key can be generated. By restricting the number of equivocable points to a polynomial bound, we avoid the impossibility results associated with full programmability, yet retain enough flexibility for protocol design.

**Definition 3 (Equivocable PRF).** *An* equivocable PRF *is a tuple* $\mathsf{EqPRF} = (\mathsf{KeyGen}, \mathsf{Eval}, \mathsf{EqEval}, \mathsf{EqKey})$*, where* KeyGen *and* Eval *are defined as in Definition 2, and the remaining algorithms are as follows:*

$(y, \mathsf{st}') \leftarrow \mathsf{EqEval}(x, \mathsf{st})$**:** *The* equivocable evaluation *algorithm returns a programmable output $y$ on input $x$ and updates the internal state from* st *to* $\mathsf{st}'$*.*

$k \leftarrow \mathsf{EqKey}(\{(x_i, y_i)\}_{i=1}^m, \mathsf{st})$**:** *The* equivocation *algorithm outputs a key $k \in \mathcal{K}$ consistent with the given input-output pairs, or $\perp$ on failure.*

*On the internal state* st*.* The equivocable evaluation algorithm EqEval maintains an auxiliary state st that is updated after each query. This state records information necessary to ensure consistency across multiple programmed outputs. In particular, st may contain randomness used by EqEval as well as a log of previously programmed pairs $(x, y)$. The state does not depend on any secret key, and its size is polynomial in the security parameter. At the end of the game, the procedure EqKey takes as input both the set $Q$ of all programmed pairs and the final state st, and must output a key that is consistent with these pairs. Hence, st serves only as bookkeeping for equivocation and cannot be used to smuggle additional information beyond what is obtainable through the oracle queries.

For conceptual completeness, we also define a natural correctness property for equivocable PRFs. In essence, correctness ensures that the key produced by EqKey is consistent with the previously programmed outputs. As the reader may observe, this property is implicitly required by the security definition in Definition 5. Indeed, if correctness fails, an adversary could trivially distinguish the real from the ideal world by checking whether the outputs match under the returned key.

**Definition 4 (Correctness of Equivocable PRFs).** *An equivocable pseudorandom function* $\mathsf{EqPRF} = (\mathsf{KeyGen}, \mathsf{Eval}, \mathsf{EqEval}, \mathsf{EqKey})$ *is* $m$-correct *if, for any sequence of $m$ inputs $x_1, \ldots, x_m \in \mathcal{X}$, the following holds:*

1. *Initialize $Q \leftarrow \{\}$ and $\mathsf{st} \leftarrow \perp$.*
2. *For each $i \in [m]$, compute $(y_i, \mathsf{st}) \leftarrow \mathsf{EqEval}(x_i, \mathsf{st})$ and update $Q \leftarrow Q \cup \{(x_i, y_i)\}$.*
3. *Compute $k \leftarrow \mathsf{EqKey}(Q, \mathsf{st})$.*
4. *Then, with overwhelming probability in $\lambda$, it holds that $y_i = \mathsf{Eval}(k, x_i)$ for all $i \in [m]$.*

### 4.2 Security of Bounded Equivocable Pseudorandom Functions

The security game for $m$-equivocability formalizes the idea that the PRF's outputs can be programmed for up to $m$ distinct inputs and remain indistinguishable from real evaluations under a randomly sampled key. In

$\mathsf{Game}^{\mathsf{Eqv},m,b}_{\mathcal{A},\mathsf{EqPRF}}(\lambda)$

$Q \leftarrow \{\}, C \leftarrow 0, \mathsf{st} \leftarrow \bot$
**if** $b = 0$ :
  $k \leftarrow \mathsf{EqPRF.KeyGen}(1^\lambda)$
  $\mathcal{A}^{\mathsf{ORealEval}(k,\cdot)}(1^\lambda)$
**else** :
  $\mathcal{A}^{\mathsf{OIdealEval}(\cdot)}(1^\lambda)$
  $k \leftarrow \mathsf{EqPRF.EqKey}(Q, \mathsf{st})$
$b' \leftarrow \mathcal{A}(k)$
**return** $b == b' \wedge C \leq m$

Oracle $\mathsf{ORealEval}(k, x)$

$C \leftarrow C + 1$
$y \leftarrow \mathsf{EqPRF.Eval}(k, x)$
$Q \leftarrow Q \cup \{(x, y)\}$
**return** $y$

Oracle $\mathsf{OIdealEval}(x)$

// If $x$ is re-supplied
// don't increment $C$
**if** $\exists y$ s.t. $(x, y) \in Q$ :
  **return** $y$
$C \leftarrow C + 1$
$(y, \mathsf{st}) \leftarrow \mathsf{EqPRF.EqEval}(x, \mathsf{st})$
$Q \leftarrow Q \cup \{(x, y)\}$
**return** $y$

**Fig. 1.** Security game for bounded equivocability.

the ideal world ($b = 1$), the adversary interacts with a programmable interface that simulates PRF outputs using $\mathsf{EqEval}$, without knowledge of a key. After answering at most $m$ distinct queries, a key $k$ is computed via $\mathsf{EqKey}$ to match the simulated outputs. In the real world ($b = 0$), the adversary instead interacts with the genuine PRF using a secret key $k$. The adversary's goal is to distinguish the two worlds. If the advantage is negligible, we conclude that the equivocation procedure convincingly simulates the PRF up to $m$ inputs, and the computed key is computationally indistinguishable from a real key. This captures the core notion of equivocable programmability.

**Definition 5 ($m$-Equivocability).** *Let* $\mathsf{EqPRF} = (\mathsf{KeyGen}, \mathsf{Eval}, \mathsf{EqEval}, \mathsf{EqKey})$ *be an equivocable PRF, and let* $m \in \mathbb{N}$ *be a bound on the number of programmable queries. We say that* $\mathsf{EqPRF}$ *is* $m$-equivocable *if it is pseudorandom according to Definition 2 and for all PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that:*

$$\left|\Pr\left[\mathsf{Game}^{\mathsf{Eqv},m,0}_{\mathcal{A},\mathsf{EqPRF}}(\lambda) = 1\right] - \Pr\left[\mathsf{Game}^{\mathsf{Eqv},m,1}_{\mathcal{A},\mathsf{EqPRF}}(\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda).$$

*The game* $\mathsf{Game}^{\mathsf{Eqv},m,b}_{\mathcal{A},\mathsf{EqPRF}}$ *is defined in Figure 1.*

The game in Figure 1 mirrors the SIM-AC definition, except that the number of programmable queries is bounded by $m$and it is defined as distinguishing game between an ideal and real world. In the real world, a key is sampled and the PRF is evaluated honestly on all inputs from $\mathcal{A}$. Upon corruption, the key is revealed to $\mathcal{A}$. In the ideal world, outputs are generated via $\mathsf{EqEval}$ without any key. At corruption, the procedure $\mathsf{EqKey}$ takes the set $Q$ of recorded input-output pairs and the state $\mathsf{st}$, and returns a key that is consistent with $Q$ and indistinguishable from one sampled by $\mathsf{KeyGen}$.

In the original SIM-AC definition, impossibility in the standard model was circumvented by giving $\mathcal{A}$ access to an ideal primitive oracle $\mathsf{Prim}$, which was also used internally in the PRF evaluation. Our constructions work in the standard model and thus do not require $\mathsf{Prim}$. Some instantiations use a random oracle $\mathsf{H}$, but only for randomization of public inputs. The model in Figure 1 easily extends to give $\mathcal{A}$ oracle access to $\mathsf{H}$, and since $\mathsf{H}$ is only evaluated on public inputs, sound verifiability is preserved.

Finally, the most striking difference between our definition and that of [JT20a] is the fact that we consider single-user security, while the SIM-AC definition is inherently multiuser. Jaeger and Tyagi adopted this design choice due to some shortcomings of SIM-AC, including the difficulty of proving multi-user security as an implication of single-user. This was addressed in the definition of SIM$^*$-AC PRFs [Jae23] by giving $\mathcal{A}$

more power, such as the ability to program $\mathsf{Prim}$[2]. As elaborated in [Jae23], the difficulty arises because each single-user simulator expects to have full control (including programming) over $\mathsf{Prim}$, possibly leading to conflicts in the multi-user to single-user reduction. Because our definition of equivocability targets the standard model, such problem does not arise. Our standard-model constructions need not share an ideal primitive, thus dodging conflicts entirely. Furthermore, even in the aforementioned ROM-based instances, we only use a significantly smaller subset of the ROM's properties, namely its ability to randomize inputs. In particular, we do need to program $\mathsf{H}$ on any input. Therefore, for our setting, multi-user security naturally stems from single-user security.

# 5 Equivocable PRF from DDH

We first present a construction of bounded-equivocable PRFs from the Decisional Diffie-Hellman ($\mathsf{DDH}$) assumption. The intuition behind our design is provided in the technical outline (Section 2); here we focus on the formal construction. First we recall the DDH assumption in a cyclic group $\mathbb{G} = \langle g \rangle$ of prime order $q$ with bit length $\lambda$. For $\alpha, \beta, \gamma \leftarrow\!\!\$\, \mathbb{Z}_q$, define the advantage of distinguisher $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathcal{A}}_{\mathsf{DDH}}(\lambda) = \Big| \Pr\left[1 \leftarrow \mathcal{A}(g, g^\alpha, g^\beta, g^\gamma)\right] - \Pr\left[1 \leftarrow \mathcal{A}(g, g^\alpha, g^\beta, g^{\alpha\beta})\right] \Big|.$$

The $\mathsf{DDH}$ assumption states that $\mathsf{Adv}^{\mathcal{A}}_{\mathsf{DDH}}(\lambda)$ is negligible for all PPT $\mathcal{A}$.

## 5.1 Construction

In this section, we show that a tweaked version of the Naor and Reingold PRF [NRR00,NR97] is equivocable.

*Setup.* Let $\ell \in \mathbb{N}$ be a parameter that determines the input length of the PRF, and $\mathcal{K}$ a finite samplable set. We work in a finite cyclic group $\mathbb{G}$ of prime order $q$ with generator $g$, where $q = 2q' + 1$ is assumed to be a safe prime. We also fix a function $\mathsf{H}\colon \{0,1\}^\ell \times \mathcal{K} \to \{0,1\}^\ell$.

$\mathsf{KeyGen}(1^\lambda, \ell)$**:** On input the security parameter $1^\lambda$ and length parameter $\ell$, sample a random key

$$\boldsymbol{k}_{in} \leftarrow\!\!\$\, \mathcal{K}, \quad \boldsymbol{k}_{out} \leftarrow\!\!\$\, \mathbb{Z}_q^{\ell+1}.$$

Output $\boldsymbol{k} =: (\boldsymbol{k}_{in}, \boldsymbol{k}_{out})$.

$\mathsf{Eval}(\boldsymbol{k}, \boldsymbol{x})$**:** On input a key $\boldsymbol{k}_{in} \in \mathcal{K}, \boldsymbol{k}_{out} = (k_0, k_1, \ldots, k_\ell) \in \mathbb{Z}_q^{\ell+1}$ and $\boldsymbol{x} \in \{0,1\}^\ell$:

1. Compute $z \leftarrow \mathsf{H}(\boldsymbol{k}_{in}, \boldsymbol{x}) = (z_1, \ldots, z_\ell) \in \{0,1\}^\ell$.
2. Evaluate the Naor–Reingold PRF as

$$F(\boldsymbol{k}_{out}, z) \;=\; g^{\,k_0 \prod_{i=1}^{\ell} k_i^{z_i}}.$$

3. Output $F(\boldsymbol{k}_{out}, z)$.

$\mathsf{EqEval}(\boldsymbol{x}, \mathsf{st})$**:** On input $\boldsymbol{x} \in \{0,1\}^\ell$:

1. Sample $s \leftarrow\!\!\$\, \mathbb{Z}_{q-1}$.
2. Let $h$ be a generator of $\mathbb{Z}_q^*$. Compute $r \leftarrow h^s$.
3. Record the pair $(\boldsymbol{x}, g^r)$ in the query list $Q$.
4. Store $s$ in the state $\mathsf{st}$.
5. Output $r$.

[2] Note that, unlike the aforementioned impossibility results, neither [JT20a] not [Jae23] present a formal theorem showing that single-user $\mathsf{SIM\text{-}AC}$ PRFs cannot imply multi-user $\mathsf{SIM\text{-}AC}$ PRFs, but give convincing informal arguments that show where the proofs would fail.

$\mathsf{EqKey}(Q, \mathsf{st})$**:** Let $Q$ be a list of $L$ pairs $(\boldsymbol{x}_i, g^{r_i})$:

1. Sample $\boldsymbol{k}_{in} \leftarrow \mathcal{K}$.
2. Construct the matrix $A \in \{0,1\}^{L \times (\ell+1)}$ whose $i$-th row is
$$(1, H(\boldsymbol{k}_{in}, \boldsymbol{x}_i)_1, H(\boldsymbol{k}_{in}, \boldsymbol{x}_i)_2, \ldots, H(\boldsymbol{k}_{in}, \boldsymbol{x}_i)_\ell)\,.$$
3. For each $i \in [L]$, find $s_i$ such that $r_i = h^{s_i}$ and set $\boldsymbol{s} = (s_1, \ldots, s_L)$.
4. Solve the linear system
$$A\boldsymbol{y} \equiv \boldsymbol{s} \pmod{q-1}.$$
If the system has no solution, output $\perp$.
5. Otherwise, return $\boldsymbol{k}_{out} = g^{\boldsymbol{y}}$, where $g^{\boldsymbol{y}}$ denotes component-wise exponentiation.

Regarding the security, we prove the following theorem.

**Theorem 2.** *Let $q = 2q' + 1$ be a safe prime and $\mathbb{G}$ be a cyclic group of order $q$ in which the DDH problem is hard, and $t = 1 + \ell \cdot \frac{\log(3/2)}{\log((q-1)/2)}$. If $\mathsf{H}$ is $t$-wise independent function, then the construction above is a $t$-equivocable pseudorandom function. The same is true if $\mathsf{H}$ is a pseudorandom function.*

### 5.2 Proof of Security

*Proof Overview.* Definition 5 imposes two orthogonal requirements, namely pseudorandomness and bounded equivocability. Pseudorandomness immediately follows from the fact that the outer block is a PRF, and we only perform preprocessing before feeding the inputs to it.

For the proof of equivocability, the central difficulty is that adversarial queries induce multiplicative constraints over $\mathbb{Z}_q$, which are cumbersome to analyze directly. To gain algebraic leverage, we move to the exponent domain: for prime $q$, the group $\mathbb{Z}_q^*$ is cyclic of order $q-1$ and isomorphic to $(\mathbb{Z}_{q-1}, +)$. While in principle we work over the full $\mathbb{Z}_q$ space, we can safely ignore 0 since the latter only has a negligible probability mass. Writing each key component as $k_i = h^{z_i}$ for a generator $h$, the multiplicative constraints translate into a linear system $A\boldsymbol{z} \equiv R \pmod{q-1}$, where $A = (\mathbf{1} \mid D)$ and $D$ encodes the PRF inputs. Since $q-1$ is composite, we assume $q = 2q' + 1$ is a safe prime, which allows us to split the system into two linear systems modulo 2 and $q'$, solve them separately over the fields $\mathbb{F}_2$ and $\mathbb{F}_{q'}$, and recombine solutions via the Chinese Remainder Theorem. At first we idealize inputs as uniformly random and independent bitstrings; in this setting $D$ is a random binary matrix. Standard rank arguments over finite fields imply that such matrices are full-rank with overwhelming probability, ensuring that both systems admit solutions, and hence so does the original one. Moreover, solving for $\boldsymbol{z}$ yields a key vector $\boldsymbol{k}_{out}$ whose distribution is indistinguishable from a genuinely sampled key, by a standard hybrid argument.

We then remove the simplifying assumptions instantiating $\mathsf{H}$ with a $t$-wise independent function or, equivalently, a pseudorandom function. In this setting, for any set of at most $t$ distinct queries the rows of $D$ are distributed exactly as independent uniform binary vectors. Hence the distribution matches the uniform-input analysis, and the previous rank argument applies without change. Equivocation remains efficient: instead of computing discrete logarithms, we sample exponents directly and record them in the state.

*Existence of a Consistent Key.* For each input–output pair $(\boldsymbol{x}, g^r) \in Q$ we can associate the constraint
$$k_0 \prod_{j \in I_1(\boldsymbol{x})} k_j \equiv r \pmod{q},$$
where $I_1(\boldsymbol{x}) \subseteq [\ell]$ denotes the set of indices $j$ for which $\mathsf{H}(\boldsymbol{k}_{in}, \boldsymbol{x})_j = 1$. Finding a key consistent with all pairs in $Q$ thus amounts to solving the resulting system of constraints. We work over $\mathbb{Z}_q^*$ instead of $\mathbb{Z}_q$, which is harmless: when sampling keys and outputs at random, the probability that any of the $\ell$ key coordinates or any of the $L$ output exponents equals 0 is at most $(\ell + L)/q$, negligible in the security parameter. Since

$(\mathbb{Z}_q^*, \cdot)$ is cyclic of order $q-1$, it is isomorphic to $(\mathbb{Z}_{q-1}, +)$. Writing each key element as $k_i = h^{y_i}$ for a generator $h$, the constraints translate into a linear system

$$A\boldsymbol{y} \equiv R \pmod{q-1},$$

where $A \in \{0,1\}^{L \times (\ell+1)}$ and $R \in \mathbb{Z}_{q-1}^L$. The matrix $A$ has the form $A = (\mathbf{1} \mid D)$: the first column $\mathbf{1}$ reflects the fact that $k_0$ always contributes to the exponent, while $D$ is the $L \times \ell$ binary matrix obtained by stacking the values $\mathsf{H}(\boldsymbol{k}_{in}, \boldsymbol{x})$. Concretely, the requirement $F_{\boldsymbol{k}}(\mathsf{H}(\boldsymbol{k}_{in}, \boldsymbol{x})) = g^r$ becomes

$$y_0 + \sum_{i=1}^{\ell} y_i \, \mathsf{H}(\boldsymbol{k}_{in}, \boldsymbol{x})_i \equiv s \pmod{q-1},$$

where $k_i = h^{y_i}$, $r = h^s$, and the exponents $s$ are chosen at random. Importantly, we avoid any discrete-logarithm computation: instead of sampling $r \in \mathbb{Z}_q^*$ and computing $\log_h(r)$, we sample $s \leftarrow\!\!\$\ \mathbb{Z}_{q-1}$, set $r \leftarrow h^s$, and store $s$ in the state $\mathsf{st}$ for use at equivocation time. Because $q-1$ is composite, direct analysis is inconvenient. We therefore assume $q = 2q' + 1$ is a safe prime, which allows us to reduce the system to

$$A\boldsymbol{y} \equiv R \pmod{2} \qquad \text{and} \qquad A\boldsymbol{y} \equiv R \pmod{q'},$$

and then combine the two solutions using the Chinese Remainder Theorem. This is always possible since $\gcd(2, q') = 1$.

It remains to argue that the systems admit solutions. To this end, we first consider an idealized setting where the inputs $\boldsymbol{x}$ are uniformly random and independent bitstrings, so that $D$ is a uniformly random binary matrix. In this regime, rank properties of random matrices over finite fields (see Lemmas below) imply that with overwhelming probability both systems modulo 2 and modulo $q'$ are consistent. Consequently, the original system modulo $q-1$ also admits a solution. Formally, we rely on two key technical ingredients. The first one states that for a random $m \times n$ binary matrix over $\mathbb{F}_2$ is full rank with overwhelming probability whenever $m/n$ is bounded away from 1 by a small constant.

**Lemma 2.** *A random binary matrix $A$ of dimension $m \times n$ with $m = (1-\delta)n$ for some constant $\delta \in (0,1)$ has rank $m$ over $\mathbb{F}_2$ with probability at least $1 - \frac{(1-\delta)n}{2^{\delta n+1}}$ for sufficiently large $n$.*

The second one resembles the first in spirit, as it also attempts to lower-bound the probability that a rectangular $m \times n$ random binary matrix over $\mathbb{F}_q$ is full-rank. However, due to the inhomogeneity between the binary matrix entries and the prime modulus $q$, the formal statement is slightly different, and requires stronger conditions on $m$, $n$, and $q$.

**Lemma 3.** *Let $q$ be an odd prime. Consider a random binary matrix $A$ of dimension $m \times n$ with $m = 1 + \frac{n \log(3/2)}{\log(q)}$. Furthermore, suppose $n^{1/2} = \log(q)$. Then $A$ has rank $m$ over $\mathbb{F}_q$ with probability at least $1 - (\sqrt{n}\log(3/2) + 1)(3/4)^n$ for sufficiently large $n$.*

*Proof (of Lemma 2).* Let us explicitly count the matrices with this property. Imagine we generate a binary matrix row-by-row until completion in such a way that the final rank is $m$. The first row can be any non-zero binary vector of length $n$, thus we have $2^n - 1$ possible choices. When selecting the second row, we have to exclude all multiples of the first row, yielding a total of $2^n - 2$ possibilities. The third row can be any binary vector that is not a linear combination of the first two, thus giving us $2^n - 4$ possibilities. It is straightforward to go on with this until $m$ rows are generated. The total number of admissible matrices is

$$\prod_{i=0}^{m-1} (2^n - 2^i)$$

Because the total number of binary $m \times n$ matrices is $2^{nm}$, and taking into account the ansatz $m = (1-\delta)n$ we get the probability

$$p = \prod_{i=0}^{m-1} (1 - 2^{i-n})$$

Note that $p^{1/m}$ is the geometric mean of values $(1-2^{i-n})_{0\le i\le m-1}$. Through the properties of the geometric mean

$$p \ge \min_{0\le i\le m-1}(1-2^{i-n})^m = (1-2^{m-1-n})^m = (1-2^{-\delta n-1})^{(1-\delta)n}$$

Now, let us apply Bernoulli's inequality. The latter states that for real $r \ge 1$, $x \ge -1$,

$$(1+x)^r \ge 1+rx$$

In our case, $r = (1-\delta)n$, which is above 1 for sufficiently large $n$, and $x = -2^{-\delta n-1} \ge -1$ (for any $n \ge 0$). Thus, by Bernoulli's inequality

$$p \ge 1 - \frac{(1-\delta)n}{2^{\delta n+1}}$$

□

*Proof (of Lemma 3).* We again count the number of matrices with rank $m$, using the same row-by-row generation policy. When generating row $i+1$ we must make sure to avoid linear combinations over $\mathbb{F}_q$ of the previous $i$ rows. In total, there are $q^i$ possible such combinations, not all of them being 0-1 vectors. Nevertheless, the number of possible choices for the $i$-th row is at least $2^n - q^i$. The reason is that there are at most $2^n$ possible binary rows of length $n$, and in the worst case all the $q^i$ linear combinations of the previous rows yield a binary result. Then, the probability of hitting a rank-$m$ matrix is at least

$$p = \prod_{i=0}^{m-1}(1-\frac{q^i}{2^n})$$

Through previous techniques, we can again derive bound

$$p \ge \left(1-\frac{q^{m-1}}{2^n}\right)^m$$

Recalling that $m-1 = n\log(3/2)/\log(q)$, the ratio $-q^{m-1}/2^n > -(3/4)^n > -1$ for $n > 0$. Furthermore, the exponent $m = n\log(3/2)/\log(q)+1$ is always guaranteed to exceed 1. Therefore, by Bernoulli's inequality

$$p \ge 1 - (n\log(3/2)/\log(q)+1)(3/4)^n$$

Finally, since $\log(q) = \sqrt{n}$, we derive the claimed bound. □

In the following, we show how to combine Lemma 2 and Lemma 3 to prove the existence of a consistent and efficiently computable key. The theorem below essentially shows that the system of constraints that arises during the EqKey procedure admits a solution with overwhelming probability (for appropriately chosen parameters), and said solution is efficiently computable too. Once we solve for $\boldsymbol{y}$, we can set $\boldsymbol{k}_{out} = g^{\boldsymbol{y}}$ as per EqKey's description.

**Theorem 3.** *Let $q = 2q'+1$ be a safe prime and $\lambda$ a security parameter. Assume $A \in \{0,1\}^{L\times(\ell+1)}$, with $\ell = \lambda^2$, $L = 1 + \frac{\ell\log(3/2)}{\log(q')}$, $\lfloor\log(q')\rfloor = \lambda$ and $A = (\mathbf{1}|D)$ where $D$ is a binary random $L\times\ell$ matrix and $\mathbf{1}$ denotes a column consisting of $L$ 1s. Then, for any $R \in \mathbb{Z}_{q-1}^L$, the system $A\boldsymbol{y} = R \mod q-1$ in the unknown $y$ has solutions over $\mathbb{Z}_{q-1}$ with overwhelming probability in $\lambda$. Furthermore, when a solution exists it is efficiently computable.*

*Proof (of Theorem 3).* Consider the following algorithm

1. Choose arbitrary $y_0' \in \mathbb{Z}_{q-1}$.
2. Consider the linear system

$$S : D\begin{pmatrix} y_1 \\ \vdots \\ y_l \end{pmatrix} = \begin{pmatrix} R_1 - y_0' \\ \vdots \\ R_L - y_0' \end{pmatrix}$$

3. Then if $(y_1, \ldots, y_l)$ is a solution to system $S$, $(y'_0, y_1, \ldots, y_l)$ is a solution to the initial system.

In our case, $D$ is random over $\{0,1\}^{L\times\ell}$. Furthermore, for sufficiently large $\lambda$

$$\ell = \lambda^2 \geq 1 + \lambda \log(3/2) = L$$

Consider the events

$$E_2 : r_2(D) = L$$
$$E_{q'} : r_q(D) = L$$

whew $r_t$ denotes the matrix rank over $\mathbb{F}_t$. By the previous inequality, $L$ is the maximum possible rank for $D$. If $E_2$ occurs, then $r_2(D|R) = L$ as well, since we are just adding to a column to $D$. The same holds when reasoning on $E_{q'}$ and $r_{q'}$. The lower bound

$$\Pr[E_2 \cap E_q] \geq \Pr[E_2] + \Pr[E_{q'}] - 1 \tag{1}$$

is guaranteed by Frechet inequality.

Let us focus on $E_{q'}$. Note that $D$ is a $L \times \ell$ matrix with entries modulo $q'$, and $\ell = \lambda^2$, $L = 1 + \frac{\ell \log(3/2)}{\log(q')}$, and $\log(q') = \lambda = \sqrt{\ell}$. By Lemma 3, whose preconditions are all fulfilled

$$\Pr[E_{q'}] \geq 1 - \left(\frac{3}{4}\right)^{\lambda^2} \cdot (1 + \lambda \log(3/2))$$

so the lower bound reduces to $\Pr[E_2] - negl_1(\lambda)$. Now, note that $\Pr E_2$ is a decreasing function of $L$. More precisely, for $L' > L$ we get

$$\Pr[E_2|L'] = \prod_{i=0}^{L'-1}(1-2^{i-n}) = \prod_{i=0}^{L-1}(1-2^{i-n}) \cdot \prod_{i=L}^{L'-1}(1-2^{i-n}) \leq \prod_{i=0}^{L-1}(1-2^{i-n}) = \Pr[E_2|L]$$

So, let us set $L' = (1-\delta)\lambda^2$, and recall that in our setting $L = 1 + \log(3/2)\lambda$, and for any constant $\delta$ and sufficiently large $\lambda$, $L' > L$. Thus, using Lemma

$$\Pr[E_2|L] \geq \Pr[E_2|L'] \geq 1 - negl_2(\lambda).$$

Hence, the overall bound, which also bounds the probability that a solution exists, is overwhelming in $\lambda$. Furthermore, thanks to aforementioned CRT strategy, we can efficiently compute a solution when it exists. □

*Deterministic Generation of D in the Standard Model* Having established these preliminaries, we now drop the simplifying assumption on the input distribution and move to the construction with an internal keyed function $\mathsf{H}$. The key difference is that the $d_i$ values that compose the matrix $D$ are no longer uniformly random, but instead derived from $\mathsf{H}$. We first consider the case where $\mathsf{H}$ is a pseudorandom function. Note that the rank of a matrix is an efficiently computable property. Consequently, the rank of $A$ should remain comparably high when $D$ is generated uniformly at random or by $\mathsf{H}$. Otherwise, one could devise an efficient distinguisher. We formalize this intuition in the following statement.

**Theorem 4.** *Let $L$, $\ell$ and $q$ be defined as in Theorem 3. Let $A = (\mathbf{1}|D)$ be obtained as in Theorem 3, but with the difference that its rows are generated by evaluating a PRF $\mathsf{H}(\boldsymbol{k}_{in}, \cdot)$. Then, for any PPT adversary $\mathcal{A}$ interacting with the PRF the associated linear system is consistent with overwhelming probability.*

The previous theorem shows that PPT adversaries are unable to enforce an inconsistent system of constraints. We note that this requirement is, in principle, distinct from pseudorandomness. Identifying cryptographic primitives that satisfy this weaker notion is an intriguing research direction, and it is conceivable that more lightweight constructions than PRFs could suffice. Exploring this possibility remains an open problem for future work.

*Proof (of Theorem 4).* Let $L = (1-\delta)\ell$ for $\delta \in (0,1)$. Consider the following distinguishing algorithm obtained by running $\mathcal{A}$ with oracle access to $\mathsf{O}f$:

1. Run $\mathcal{A}$, letting it generate adversarial queries $\boldsymbol{x} = x_1, \ldots, x_{L+1}$.
2. Feed $x_1, \ldots, x_{L+1}$ to $\mathsf{O}f$, each at point of its generation by $\mathcal{A}$. Obtain $\mathbf{d}_1, \ldots, \mathbf{d}_{L+1}$.
3. Arrange the latter in a matrix $D$ where the $j$-th column equals $\mathbf{d}_j$.
4. Obtain a matrix $A = (\mathbf{1}|D)$ as before.
5. Check if $A$ has rank less than $L$. Output 0 if yes, and 1 otherwise.

Let us now assume that $\mathsf{O}f$ can either be $\mathsf{H}(\boldsymbol{k}_{in}, \cdot)$ with uniformly random key $\boldsymbol{k}_{in}$ or a uniformly random function drawn from the set of functions from. By the pseudorandomness of $\mathsf{H}$, this distinguisher must succeed with negligible probability. More precisely, we already showed that if $\mathsf{O}f$ is a uniformly random function then $A$'s rank is $L$ with probability $1 - \mathsf{negl}(\lambda)$. If $\mathsf{O}f$ is a PRF, then by assumption $\mathcal{A}$ should return 1 with a probability $\epsilon_{\mathsf{PRF}}$ such that $|1 - \mathsf{negl}(\lambda) - \epsilon_{\mathsf{PRF}}| \leq \mathsf{negl}(\lambda)'$. It immediately follows that $\epsilon_{\mathsf{PRF}}$ is overwhelming in $\lambda$. By construction, we get that the $A$ matrix obtained from a PRF has the desired rank with equal probability. □

We now turn to the case where $\mathsf{H}$ is instantiated as a $t$-wise independent hash function. This admits a considerably simpler analysis: by $t$-wise independence, for any $t$ (or fewer) distinct inputs $\{x_i\}_{i\in[t]}$, the corresponding evaluations $\mathsf{H}(\boldsymbol{k}_{in}, x_i)i \in [t]$ are uniformly random and mutually independent. Therefore, the previous lemmas on random binary matrices and Theorem 2 continue to apply.

*Proof of Indistinguishability.* For our argument to be fully sound, we must address one obstacle. Specifically, we need to show that the output of $\mathsf{EqKey}$ is indistinguishable from a honestly generated key. We first focus on the "outer" key $\boldsymbol{k}_{out}$. We present the argument for uniformly random $D$ first, and then adapt it to the case of pseudorandom $D$ in the next section. We build upon a well-known result in linear algebra.

**Lemma 4.** *A* consistent *linear system $AX = B$ over $\mathbb{Z}_q$ with prime $q$ and $A \in \mathbb{Z}_q^{m\times n}, B \in \mathbb{Z}_q^m$ has $q^{n-r_q(A|B)}$ distinct solutions.*

We now sketch how to build upon this lemma to show that the output of $\mathsf{EqKey}$ is indistinguishable from an honestly generated key. The full argument is provided in Section B.1. We give two arguments, each concerning the indistinguishability of the system's solution modulo 2 and $q'$ respectively. We start with the latter.

The main idea is to partition the $\mathbb{Z}_q^\ell$ space into disjoint subsets $\mathcal{I}_R(D)$, where the latter contains exactly those vectors $y$ such that $Dy = R$. Conditioning on the event that $D$ is full-rank, which, by our earlier argument, occurs with overwhelming probability, each $\mathcal{I}_R(D)$ has cardinality $q^{\ell-L}$. When equivocating, we essentially return a random element of $\mathcal{I}_R(D)$. In the real world, on the other hand, we sample a random vector in $\mathbb{Z}_q^\ell$ (the key), and multiply by $D$ to obtain the corresponding image $R$. We can show that the joint distribution of $D$, the image, and the preimage is statistically close in both cases. The argument holds for any prime $q$and in particular we can set $q = q'$.

We now switch to the modulo 2 case. Here, the same indistinguishability conclusion follows from Lemma 1 (the Leftover Hash Lemma), since the random $D$ acts a universal hash function (Definition 1 in our setting). Finally, we can combine the two arguments modulo 2 and $q'$ into an indistinguishability statement modulo $2q'$. This is enabled by applying Lemma 12, which itself relies on Theorem 1 i.e., the standard Chinese Remainder Theorem. At this point, we conclude the proof by noting that $\boldsymbol{k}_{out}$ is derived as a function the system's solution that is identical in both the ideal and real world.

It remains to treat the case of the $\boldsymbol{k}_{in}$. Recall that this intermediate key only serves the purpose of randomizing the $D$ matrix, as explained in Section 5.2. Therefore, $\boldsymbol{k}_{in}$ can be generated on-the-fly by sampling uniformly at random from $\mathcal{K}$, and returned alongside $\boldsymbol{k}_{out}$ and the evaluations. We formally prove in Section B.1 that this modification does not affect indistinguishability.

### 5.3 Mapping to Scalars

For some applications, it may be necessary to get an output in $\mathbb{Z}_p$ for prime $p$. This is natively not supported by our construction if instantiated in an elliptic curve group, since it would return an elliptic curve point. We envision two possible fixes.

The first one instantiates the Naor-Reingold PRF in a different group. Indeed, DDH is believed to hold in the prime-order subgroup of $(\mathbb{Z}_p^*, \cdot)$ with $p$ a safe prime, sometimes called Schnorr group. More precisely:

- Pick safe prime $p = 2p' + 1$.
- The aforementioned group has order $2p'$ and is cyclic.
- Therefore it must have a subgroup of prime order $p'$. Let's call it $\mathbb{G}_{p'}$.
- DDH is believed to hold in $\mathbb{G}_{p'}$.

$\mathbb{G}_{p'}$ is a scalar group, and it conveniently holds that $\mathbb{G}_{p'}$ is the subgroup of square residues of $\mathbb{Z}_p^*$. In particular, this group consists of the $p'$ distinct elements $1^2, 2^2, \ldots, (p')^2 \mod p$. This suggests a very nice conversion function: the modular square root. That is, given an element $\alpha \in \mathbb{G}_{p'}$ we convert it into an element of $\mathbb{Z}_{p'}$ by:

1. Compute the square root $\beta$ of $\alpha$ modulo $p$ efficiently, using Cipolla's algorithm.
2. If $\beta < p'$, return $\beta$. Otherwise, return $\beta \mod p'$.

To enable our construction in $\mathbb{G}_{p'}$ some number-theoretic requirements must be satisfied, namely the fact that both $p'$ and $2p' + 1$ are safe primes. This property is captured by the notion of Cunningham chains [For99], i.e. sequences of primes $p_1, \ldots, p_k$ such that every $p_i$ is prime and $p_i = 2p_{i-1} + 1$. The parameter $k$ is called *length* of the chain. For our purposes, it suffices to find a chain of length $k = 3$ with a large enough $p_1$. Fortunately, such chains are known to exist, with the record[3] for the largest known $p_1$ to date being $1128330746865 \cdot 2^{66439} - 1$. Furthermore, previous work by Young and Yung [YY98] showed the feasibility of algorithmically finding chains of length three for cryptographic purposes.

On the other, this strategy requires working modulo rather large integers, which in turn yields significantly larger runtime than what the same construction would achieve if it was run in an elliptic curve group. To unlock this computational speedup, we can resort to a suitable conversion function $f : \mathbb{G} \to \mathbb{Z}_p$ such that if $g$ is a uniformly random element of $\mathbb{G}$ then $f(g)$ is uniformly random over $\mathbb{Z}_p$. For this purpose, we set $\mathbb{G}$ to be an elliptic curve over $\mathbb{F}_{p^2}$. Recall that any $\alpha \in \mathbb{F}_{p^2}$ can be represented as a linear polynomial $\alpha_0 + \alpha_1 x$ with coefficients in $\mathbb{F}_p$. Then, define the function $f$ mapping $0_{\mathbb{G}}$ to 0 and any non-zero $(X_0 + xX_1, Y_0 + xY_1) \in \mathbb{G}$ to $X_0$. A lemma by Farashi and Pellikan [FP07] states that $f$ acts as a good randomness extractor from $\mathbb{G}$ to $\mathbb{F}_p$. The statement is made quantitative as follows.

**Lemma 5 ([FP07]).** *If $p > 18$ is prime, $\mathbb{G}$ is an elliptic curve over $\mathbb{F}_{p^2}$, $g$ is a uniformly random element of $\mathbb{G}$, and $r$ is a uniformly random element of $\mathbb{F}_p$ then $\Delta(f(g), r) \leq \frac{3}{p}$.*

## 6 Equivocable PRF from LWR

We now turn to the second main construction of this work, which instantiates a (plausibly) post-quantum equivocable PRF from the Learning With Rounding (LWR) problem. Since our interest lies in pseudorandomness, we consider its decisional version, formulated as a distinguishing problem between two distributions. Let $A \leftarrow\!\$\ \mathbb{Z}_q^{m \times n}, s \leftarrow\!\$\ \mathbb{Z}_q^n$, and $b \leftarrow\!\$\ \mathbb{Z}_p^m$. We define the distributions $E_0 : (A, \lfloor As \rfloor_p)$ and $E_0 : (A, b)$, and ask the solver to distinguish a sample from $E_0$ from one from $E_1$. The corresponding $\mathsf{LWR}$ assumption with parameters $(q, p, n, m)$ states that no PPT adversary $\mathcal{A}$ has a non-negligible advantage in distinguishing the two distributions, with the latter defined as

$$\mathsf{Adv}^{\mathcal{A}}_{\mathsf{LWR}}(\lambda) = |\Pr[\mathcal{A}(t) = 1 | t \leftarrow\!\$\ E_0] - \Pr[\mathcal{A}(t) = 1 | t \leftarrow\!\$\ E_1]|$$

For appropriately chosen parameters, $\mathsf{LWR}$ inherits its hardness [BPR12,AKPW13,BGM+16,AA16] from that of Learning With Errors and, in turn, from the worst-case hardness of lattice problems such as Gap-SVP [Reg05,Pei09,MM11,MP12]. Unlike the aforementioned $\mathsf{DDH}$ assumption, no efficient quantum algorithm is known, which leads to the belief that $\mathsf{LWR}$ is quantum-resistant.

[3] A regularly updated list of world records is available here https://www.pzktupel.de/CC/cc.php.

### 6.1 Construction

*Setup.* Let $\ell \in \mathbb{N}$ denote the input length, and let $\mathcal{K}'$ be another finite samplable key space. Furthermore, let $\mathcal{K} = \mathcal{X} = \mathbb{Z}_q^\ell$ for prime $q$. The function $F : \mathcal{K} \times \mathcal{X} \to \mathbb{Z}_p$ defined as $F_{\boldsymbol{k}_{out}}(\boldsymbol{x}) = \lfloor \langle \boldsymbol{x}, \boldsymbol{k}_{out} \rangle \rfloor_p$ for another prime $p < q$ and we fix a keyed function $\mathsf{H} : \mathcal{X} \times \mathcal{K}' \to \mathcal{X}$.

$\mathsf{KeyGen}(1^\lambda, \ell)$**:** On input the security parameter $1^\lambda$ and length parameter $\ell$, sample a random key

$$\boldsymbol{k}_{in} \leftarrow\!\!\$\, \mathcal{K}', \quad \boldsymbol{k}_{out} \leftarrow\!\!\$\, \mathbb{Z}_q^\ell.$$

Output $\boldsymbol{k} =: (\boldsymbol{k}_{in}, \boldsymbol{k}_{out})$.

$\mathsf{Eval}(\boldsymbol{k}, \boldsymbol{x})$**:** on input $\boldsymbol{x} \in \mathbb{Z}_q^\ell$ and key $\boldsymbol{k}_{in} \in \mathcal{K}', \boldsymbol{k}_{out} \in \mathbb{Z}_q^\ell$:

1. Compute $z \leftarrow \mathsf{H}(\boldsymbol{k}_{in}, \boldsymbol{x})$.
2. Output $F_{\boldsymbol{k}_{out}}(z)$

$\mathsf{EqEval}(\boldsymbol{x}, \mathsf{st})$**:** On input $\boldsymbol{x} \in \mathbb{Z}^\ell$:

1. Sample a random $R \leftarrow\!\!\$\, \mathbb{Z}_q$.
2. Record pair $(\boldsymbol{x}, \lfloor R \rfloor_p)$ in $Q$.
3. Store $(\boldsymbol{x}, R)$ in $\mathsf{st}$.
4. Output $\lfloor R \rfloor_p$.

$\mathsf{EqPRF}(Q, \mathsf{st})$**:** Parse $Q$ as a list of $L$ pairs $(\boldsymbol{x}_i, \lfloor z_i \rfloor_p))$. Let $\{x_i\}_{i \in L}$ denote the inputs of the $L$ incoming queries.

1. Sample $\boldsymbol{k}_{in} \leftarrow\!\!\$\, \mathcal{K}$.
2. Compute $\{d_i\} i \in L$ where $d_i = \mathsf{H}(\boldsymbol{k}_{in}, \boldsymbol{x}_i)$ for $i \in [L]$.
3. Solve the linear system $\begin{pmatrix} d_1 \dots d_L \end{pmatrix}^T \cdot \boldsymbol{k}_{out} \equiv \begin{pmatrix} R_1 \dots R_L \end{pmatrix}^T$. If the system is inconsistent, output $\perp$.
4. Output $(\boldsymbol{k}_{in}, \boldsymbol{k}_{out})$.

The following theorem captures the security of our construction.

**Theorem 5.** *Let $q$ be a prime of bit length $\lambda$, and $p$ another prime s.t. $p < q$ and the LWR with parameters $(q, p, \lambda, m(n))$ is hard for any polynomial-bounded function $m$. If $\mathsf{H}$ is a pseudorandom function, then the construction above is a $\lambda$-equivocable pseudorandom function.*

### 6.2 Proof of Security

The proof strategy closely follows that of the DDH-based construction. Therefore, we omit a separate proof overview and proceed directly to the technical details.

We begin with the requirement of pseudorandomness. Since $\mathsf{H}$ is a PRF, it randomizes the inputs before passing them to the outer weak PRF $F_{\boldsymbol{k}_{out}}$. As a consequence, $F_{\boldsymbol{k}_{out}}$ is always evaluated on uniformly random inputs and thus remains pseudorandom. Importantly, evaluating a PRF as a preprocessing step does not trivialize the problem, since the PRF itself need not be equivocable. We remark that pseudorandomness would not necessarily hold if $\mathsf{H}$ was a $t$-wise independent function, as pseudorandomness would only hold up to $t$ queries. As in the DDH-based setting, we first analyze a simplified variant in which there is no preprocessing $\mathsf{H}$, and the inputs $\boldsymbol{x}$ are uniformly random and directly fed to $F$. We can show that the target system is consistent with high probability. More precisely, as shown in Section C:

**Lemma 6.** *Let $D \leftarrow\!\!\$\, \mathbb{Z}_q^{n \times n}$ and $R \leftarrow\!\!\$\, \mathbb{Z}_q^n$. If $\lambda = \lfloor \log(q) \rfloor$ and $n = \lambda$, the probability that $D$ is invertible is $Q \geq 1 - nq^{-1}$, which is overwhelming in $\lambda$.*

We remark that in the lattice-based setting it is customary to work with a modulus that is polynomial in the security parameter. In this case the bound $Q \geq 1 - nq^{-1}$ still holds, but we can no longer conclude that $Q$ is overwhelming in $\lambda$. Hence our construction is inherently restricted to super-polynomial $q$ moduli.

Since matrices over finite prime fields are efficiently invertible, the above lemma directly implies that the equivocation procedure is efficiently computable. It remains to argue that the joint distribution of inputs, outputs, and the key in the real world is indistinguishable from that in the ideal world. This follows via a standard hybrid argument—analogous to the proof in the DDH-based setting. For completeness we include it hereafter.

*Proof (of Lemma 6).* Using a counting argument similar to the one of Lemma 2, albeit modulo $q$, we get the probability of getting an invertible $D$ is

$$Q = \prod_{i=1}^{n}(1 - q^{i-1-n}) \geq (1 - q^{-1})^n.$$

We can leverage the inequality $a^n - b^n \leq n \cdot \max(a, b)^{n-1}|a - b|$ to upper bound the probability of $D$ being NOT invertible

$$1 - Q \leq 1 - (1 - q^{-1})^n \leq n/q.$$

By our ansatz on $n$ and $\log(q)$ the thesis immediately follows. □

Finally, consider the case where $\mathsf{H}$ is instantiated by a PRF. As in the DDH-based construction, if an efficient adversary can force $D$ to be non-invertible with non-negligible probability, then one could use it to distinguish $\mathsf{H}$ from a uniformly random function. This follows by a minor adaptation of Theorem 4. Likewise, the indistinguishability proof of Section B.1 carries over with the same adjustment, showing that instantiating $\mathsf{H}$ with a PRF does not weaken our claim.

### 6.3 Achieving Higher Equivocability

In principle, the equivocation bound can be pushed further. The key intuition is that requiring an *exact* solution to system $D\boldsymbol{k}_{out} = R$ is unnecessarily strict. Consider the case where $D\boldsymbol{k}_{out} = R$ is inconsistent, but there exists a short error vector $e$ (in the infinity norm) such that $D\boldsymbol{k}_{out} = R + e$ is consistent and, moreover, $\lfloor R + e \rceil_p = \lfloor R \rceil_p$. A direct application of the previous equivocation construction would fail, yet any solution to $D\boldsymbol{k}_{out} = R + e$ would still yield a consistent key.

*Overview.* We leverage some results in the theory of random $q$-ary lattices to show that for suitably chosen parameters we can push the $L$ threshold to about $O(\lambda^{3/2})$. For a matrix $A \in \mathbb{Z}_q^{n \times m}$, let $\Lambda_q(A)$ denote its row-generated lattice, i.e., the set of $m$-dimensional integer vectors $x$ such that $x \equiv A^T y \pmod q$ for some $y \in \mathbb{Z}^n$. If $A$ is sampled uniformly at random, we write $\Lambda \leftarrow\!\!\$\ \Lambda_q(m, n)$ for the distribution of such lattices. By combining standard results such as the transference theorem and Minkowski's bound, one can upper-bound the covering radius $\mu(\Lambda)$ with high probability. Specifically, with overwhelming probability

$$\mu(\Lambda) \leq \tfrac{1}{2\delta}\sqrt{m} \cdot q^{1-n/m}$$

for some constant $\delta$.

Intuitively, the covering radius measures how far any point in $\mathbb{R}^m$ can be from the nearest lattice vector. A small covering radius guarantees that every target $R$ lies close to some lattice point, meaning we can tolerate small perturbations $e$ without changing the rounded outputs. In other words, the lattice structure ensures that "almost" consistent systems can still be reconciled. Applied to $\Lambda_q(D)$, this ensures that the inconsistency of $D\boldsymbol{k}_{out} = R$ can be absorbed by a short error vector $e$ that does not affect rounding. This reasoning shows that, with appropriately chosen parameters, one can push the admissible number of queries $L$ up to about $O(\lambda^{3/2})$. We emphasize, however, that while existence of a consistent key can be established with overwhelming probability, actually *computing* it appears to require solving a decoding problem in a random $q$-ary lattice—an algorithmic task that is non-trivial.

*Extended Argument.* To lay the foundations of our argument, we introduce standard preliminary definitions from lattice theory. Given a matrix $A \in \mathbb{Z}_q^{n \times m}$ its *row-generated lattice* is

$$\Lambda_q(A) = \{x \in \mathbb{Z}^m | x \equiv A^T y \mod q \quad \text{for some } y \in \mathbb{Z}^n\}$$

It is also useful to introduce the concept of *parity-check lattice*, namely

$$\Lambda_q^{\perp}(A) = \{x \in \mathbb{Z}^m | Ax \equiv 0 \mod q\}$$

Both types of lattices are subject to well-known determinant bounds, namely $\det(\Lambda_q^\perp(A)) \leq q^n$ and $\det(\Lambda_q(A)) \geq q^{m-n}$ [MR09]. With $\Lambda_q(m,n)$ we denote the distribution of a row-generated lattice with uniformly random $A$, and $\Lambda_q^\perp(m,n)$ is analogously defined. We rely on the following statistical lemma.

**Lemma 7.** *Let $m > n$. $\Lambda_q(m, m-n)$ and $\Lambda_q^\perp(m,n)$ are statistically indistinguishable. More precisely*

$$\Delta\left(\Lambda_q(m,m-n), \Lambda_q^\perp(m,n)\right) \leq \frac{2n}{q^{m-n}}.$$

*Proof (sketch).* We only provide an overview of the main building blocks here, and refer the interested reader to [Ajt96,Mic21] for further details. Let us introduce the notation $C(\Lambda_q^\perp(m,n))$ to denote the distribution $\Lambda_q^\perp(m,n)$ but conditioned on the random lattice having maximum determinant. Similarly, we let $C(\Lambda_q(m,n))$ denote the $\Lambda_q(m,n)$ distribution, but conditioned on the lattice being with minimum determinant. The following statistical bounds

$$\Delta\left(C(\Lambda_q^\perp(m,n)), \Lambda_q^\perp(m,n)\right) \leq \epsilon_1 \quad \text{and} \quad \Delta\left(C(\Lambda_q(m,m-n)), \Lambda_q(m,m-n)\right) \leq \epsilon_2,$$

with $\epsilon_1 = \frac{n}{q^{m-n}} = \epsilon_2$. Furthermore, $C(\Lambda_q^\perp(m,n))$ and $C(\Lambda_q(m,m-n)), \Lambda_q(m,m-n))$ can be shown to be identical distributions. The claim follows by triangular inequality. □

We combine this result with a probabilistic bound on the covering radius of a parity-check lattice. The latter can be viewed as an application of the more popular Minkowski bound.

**Lemma 8.** *There exists a positive constant $\delta$ such that with overwhelming probability over the sampling of $A \leftarrow\!\$\, \mathbb{Z}_q^{n\times m}$*

$$\mu(\Lambda_q^\perp(A)) \leq \frac{1}{2\delta}\sqrt{m} \cdot q^{n/m}.$$

By the aforementioned statistical indistinguishability, then the same bound must hold for $\Lambda \sim \Lambda_q(m, m-n)$. With a simple substitution, we conclude that for $\Lambda \sim \Lambda_q(m,n)$ the bound

$$\mu(\Lambda) \leq \frac{1}{2\delta}\sqrt{m} \cdot q^{1-n/m}$$

holds.

We now apply these results to our LWR-based equivocable PRF. As pointed out, we do not really need to fix the $R_i$ values a priori. It suffices to pick random $S_i \in \mathbb{Z}_p$ when generating the responses, and then find a corresponding $R_i \in \mathbb{Z}_q$ such that $\lfloor R_i \rceil_p = S_i$. Note that the $\lfloor\cdot\rceil_p$ functions conceptually splits the interval $[0,q]$ into $p$ adjacent subintervals. Each of them is delimited by $jq/p$ and $(j+1)q/p$, for $j \in \{0,\ldots,p-1\}$, and $\lfloor x \rceil_p$ merely returns the interval to which $x$ belongs. This suggests an equivocation strategy:

1. Let $S_1,\ldots,S_L \in \mathbb{Z}_p$ represent the chosen intervals.
2. For each of them, consider the midpoints $M_1,\ldots,M_L$ with $M_i = \frac{j+1}{2}q/p$.
3. Suppose that there exists a lattice point $t \in \Lambda_q(D^T)$ such that $\|t - M\|_\infty \leq q/2p$.
4. Then $\lfloor t \rceil_p = S$, and there must exists a $y$ vector such that $t \equiv Ay \mod q$.
5. Then, we can return $y \mod q$ as a candidate key.

For this strategy to be feasible, we need guarantees on the existence of the aforementioned $t$. We define

$$\mathsf{dist}(t,\Lambda)_\ell = \min_{x\in\Lambda} \|t - x\|_\ell$$

for any lattice $\Lambda$ and point $t$. The previous task reduces to showing that there exists a point $t$ such that $\mathsf{dist}(t, \Lambda_q(D^T))_\infty \leq q/2p$. Furthermore, due to the infinity norm not exceeding the L2 norm, it suffices to find $t$ such that $\mathsf{dist}(t, \Lambda_q(D^T))_2 \leq q/2p$. This is trivially guaranteed if the covering radius of $\Lambda_q(D^T)$ does not exceed $q/2p$. Since $D$ is an $L \times \lambda$ matrix, by combining this intuition with Lemma 8 we impose

$$\frac{1}{2\delta}\sqrt{L} \cdot q^{1-\lambda/L} \leq q/2p \tag{2}$$

to have probabilistic guarantees on the existence of a solution. At the same time, we should choose parameters to ensure the pseudorandomness of LWR. We leverage the results of [AKPW13], which imply that LWR inherits pseudorandomness from LWE if $\frac{q}{p} > \lambda LB$ and $\lambda \geq \log(q) \cdot n'$. Here $B$ denotes the LWE error bound, and $n'$ is the number of columns of the LWE matrix. We in turn need to ensure LWE's hardness, and can do so via the following well-established result.

**Theorem 6 (GAP-SVP to LWE [Reg05,Pei09,MM11,MP12]).** *Set $n = \lambda$. Let $q > \tilde{O}(2^{N/2})$ be either a prime power or the product of small distinct primes, and let $B \geq \omega(\log n)\sqrt{n}$ be a norm bound. Suppose that for an efficiently samplable and $B$-bounded distribution $\chi$ there exists a PPT solver for LWE. Then there exists a polytime solver for Gap-SVP with approximation factor $\gamma = \tilde{O}(nq/B)$.*

Consider the following possible instantiation:

- we set $\lambda^{1/2} = \lceil \log(q) \rceil = n'$. Then $\lambda = \lceil \log(q) \rceil \cdot n' \geq \log(q) \cdot n'$.
- next, set $B = q/\lambda^{\log \lambda}$. This ensures that $n' \cdot \frac{q}{B} = \lambda^{\log \lambda + 1/2}$ is subexponential in $\lambda$. This allows in turn to claim pseudorandomness for LWE assuming the hardness of Gap-SVP with subexponential approximation factors, as previously done by [BGG+18,Bra12].
- at this point, we should ensure that Equation 2 and $\frac{q}{p} > \lambda LB$ both hold. You can verify that they reduce to
$$p < \frac{\lambda^{\log \lambda - 1}}{L} \quad \text{and} \quad p \leq \delta \frac{2^{\lambda^{3/2}/L}}{\sqrt{L}}$$
- these criteria can be met even if $L \geq \lambda$ (for instance, for $L = \lambda^{1.4}$) while still allowing for $p$ superpolynomial in $\lambda$. For instance, we can set $p = \lambda^{\sqrt{\log \lambda}}$, which satisfies the requirements above, and ensures that LWR is pseudorandom for polynomial number of queries $L$.

The previous instantiation is meant to target superpolynomial $p$, which is a desirable requirement to prevent random guessing with non-negligible probability.

While the previous instantiation guarantees the existence of a consistent key, it remains to show that said key is efficiently computable. Note that said computation essentially amounts to solving the following problem: given a random lattice $\Lambda \leftarrow \Lambda_q(m, n)$ and a random point $t \in \mathbb{Z}_q^m$ that is guaranteed to be within distance $D = \frac{q}{2p}$ from $\Lambda$, return the closest lattice point to $t$. This task is closely related to the well-known Bounded Distance Decoding (BDD) problem [MR07]. The concrete BDD hardness is determined by the ratio $\alpha = \frac{D}{\lambda_1(\Lambda)}$, where $\lambda_1(\Lambda)$ is the shortest non-zero vector of $\Lambda$. More precisely, define for $\ell \in \mathbb{N} \cup \{+\infty\}$

$$\lambda_1^{(\ell)}(\Lambda) = \min_{v \in \Lambda \setminus \{0\}} \|v\|_\ell.$$

Furthermore, the problem is only well-defined for $\alpha \in (0, 1/2)$, as for $\alpha > 1/2$ there is no guarantee that the solution is unique. Let us analyze $\lambda_1$ in our setting. For $\Lambda \leftarrow\$ \Lambda_q^\perp(m, m-n)$ we know that $\det(\Lambda) \leq q^{m-n}$ with overwhelming probability, and by Minkowski's bound

$$\lambda_1^{(2)}(\Lambda) \leq \sqrt{m} \cdot q^{1-n/m}.$$

By the statistical distinguishability of $\Lambda_q^\perp(m, m-n)$ and $\Lambda_q(m, n)$, the same bound must hold for the latter too. Furthermore, $\lambda_1^{(2)}(\Lambda)$ upper bounds $\lambda_1^{(\infty)}(\Lambda)$ by the standard properties of $\ell$-norms. Thus, in our setting

$$\alpha = \frac{D}{\lambda_1(\Lambda)^{(\infty)}} \geq \frac{q^{n/m}}{2p \cdot \sqrt{m}} = \frac{2^{\lambda^{3/2}/L}}{2\sqrt{L}\lambda^{\sqrt{\lambda}}}.$$

Note that the parameter exceeds the $1/2$ threshold for sufficiently large $\lambda$, making the BDD problem ill-defined. Hence, we defer a thorough investigation of the hardness of the above decoding problem to future work.

# 7 Verifiable Equivocable PRFs

Our instantiations are defined in the plain model, without relying on the random oracle. This enables us to enhance the constructions with *public verifiability*, and to lift them to the setting of verifiable PRFs and eVRFs. To this end, we introduce a formal definition of this property in Section 7.1. We further explore heuristics aimed at improving practical efficiency in Section 7.2, and present concrete constructions in Section 7.3. Finally, we illustrate some applications of our primitive in Section 7.4.

## 7.1 Definition of Verifiable Equivocable Pseudorandom Functions

As discussed in Section 2, one of our main motivations for obtaining standard-model instantiations is that it enables us to enhance bounded-equivocable PRFs with *verifiability*. We now formalize this notion. We begin with the standard concept of a Verifiable Random Function (VRF) [MRV99], which can be seen as a PRF augmented with public verifiability.

**Definition 6 (Verifiable Random Function (VRF) [MRV99]).** *A* verifiable random function *(VRF) is a triple of efficient algorithms* $\mathsf{VRF} = (\mathsf{KeyGen}, \mathsf{Eval}, \mathsf{Verify})$ *with the following syntax:*

- $\mathsf{KeyGen}(1^\lambda) \to (\mathsf{sk}, \mathsf{vk})$*: on input the security parameter* $\lambda$*, outputs a key pair* $(sk, \mathsf{vk})$*.*
- $\mathsf{Eval}(\mathsf{sk}, x) \to (y, \pi)$*: on input the secret key* $sk$ *and input* $x$*, outputs a pseudorandom value* $y$ *and a proof* $\pi$ *of correctness.*
- $\mathsf{Verify}(\mathsf{vk}, x, y, \pi) \to \{0,1\}$*: on input the public verification key* $\mathsf{vk}$*, an input* $x$*, a value* $y$*, and a proof* $\pi$*, outputs a bit* $b \in \{0,1\}$

*A VRF must satisfy the following properties that we briefly summarize:*

**Correctness:** *For all* $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and all* $x$*, if* $(y, \pi) \leftarrow \mathsf{Eval}(\mathsf{sk}, x)$*, then* $\mathsf{Verify}(\mathsf{vk}, x, y, \pi) = 1$*.*

**Unique Provability:** *For every verification key* $\mathsf{vk}$ *and input* $x$*, the probability that there exist proofs* $\pi, \pi'$ *and distinct values* $y \neq y'$ *such that* $\mathsf{Verify}(\mathsf{vk}, x, y, \pi) = 1$ *and* $\mathsf{Verify}(\mathsf{vk}, x, y', \pi') = 1$ *is negligible.*

**Pseudorandomness:** *For all PPT adversaries* $\mathcal{A}$ *the value* $y$ *on a fresh input* $x$ *is computationally indistinguishable from uniform over the output space, even given* $\mathsf{vk}$ *and access to* $\mathsf{Eval}(\mathsf{sk}, \cdot)$ *on other inputs.*

We extend this definition by combining it with the bounded-equivocability property, thereby obtaining *verifiable equivocable PRFs*. Intuitively, these inherit the indistinguishability guarantees of bounded-equivocable PRFs, while additionally allowing any evaluation to be publicly verified relative to a commitment to the secret key.

**Definition 7 (Verifiable Equivocable PRF).** *Let* $\mathcal{P}$ *denote a finite proof space. Consider a tuple of PPT algorithms* $\mathsf{EqPRF} = (\mathsf{KeyGen}, \mathsf{KeyGen}_0, \mathsf{Eval}, \mathsf{Eval}_0, \mathsf{EqEval}, \mathsf{EqKey}, \mathsf{Verify})$ *such that:*

- $\mathsf{EqEval}$ *and* $\mathsf{EqKey}$ *have interfaces identical to those of Definition 3.*
- $\mathsf{KeyGen}(1^\lambda)$ *returns a keypair* $(\mathsf{sk}, \mathsf{vk})$*.*
- $\mathsf{KeyGen}_0(1^\lambda)$ *is the algorithm that runs* $\mathsf{KeyGen}(1^\lambda)$ *and returns* $\mathsf{sk}$*.*
- $\mathsf{Eval}(k, x)$ *returns a pair* $(y, \pi) \in \mathcal{Y} \times \mathcal{P}$*.*
- $\mathsf{Eval}_0(k, x)$ *is the algorithm that evaluates* $(y, \pi) \leftarrow \mathsf{Eval}(k, x)$ *and returns* $y$*.*
- $\mathsf{Verify}(\mathsf{vk}, x, y, \pi)$ *returns* $b \in \{0,1\}$

$\mathsf{EqPRF}$ *is a verifiable equivocable pseudorandom function if* $(\mathsf{KeyGen}_0, \mathsf{Eval}_0, \mathsf{EqEval}, \mathsf{EqKey})$ *is an equivocable PRF, and* $(\mathsf{KeyGen}, \mathsf{Eval}, \mathsf{Verify})$ *is a VRF according to Definition 6.*

### 7.2 A Heuristic for Practical Verifiability

Being fully instantiated in the standard model, the constructions of Sections 5 and 6 already satisfy the requirements of Definition 7. From a practical standpoint, however, their structure makes proofs of correct evaluation challenging, primarily because the input must first be processed through the keyed function $\mathsf{H}(\boldsymbol{k}_{in}, \cdot)$. Since the value $\mathsf{H}(\boldsymbol{k}_{in}, x)$ is not available to the verifier, any proof system must account for this hidden computation, resulting in considerably higher cost. To overcome this, we replace the internal *keyed* function $\mathsf{H}(\boldsymbol{k}_{in}, \cdot)$ with a *keyless* hash function $\mathsf{H}$. This modification not only reduces prover-side space complexity, but also resolves the verification bottleneck: the verifier can compute $\mathsf{H}(x)$ directly, thereby enabling compatibility with much more practical proof systems. The keyless nature of $\mathsf{H}$ does, however, introduce a new challenge. Because computing $\mathsf{H}(x)$ requires no secret key, an adversary could query $\mathsf{H}$ on arbitrary inputs $x_i$ until it finds a small set $S$ of inputs whose outputs are linearly dependent over $\mathbb{Z}_{q'}$ and $\mathbb{Z}_2$. Crucially, this strategy would not consume any query to $\mathsf{EqPRF}$. Fortunately, this concern can be addressed rigorously. We restrict our proof to the case of the DDH-based construction of Section 5.1, but the same techniques are straightforwardly extended to the construction of Section 6. In particular, we can show the following.

**Theorem 7.** *Let $c \in \mathbb{N}$ be a constant, $\ell$ defined as in Theorem 2, and $\mathcal{A}$ be a PPT adversary with oracle access to $\mathsf{H} : \{0,1\}^{\ell} \to \{0,1\}^{\ell}$, which we model as a random oracle. Consider the following simple game:*

1. *Get $x_1, \ldots, x_{c\lambda} \leftarrow \mathcal{A}^{\mathsf{H}}(1^{\lambda})$.*
2. *Compute $v_i \leftarrow \mathsf{H}(x_i)$ for all $i \in [\lambda]$.*
3. *Return 1 if $v_1, \ldots, v_{c\lambda}$ are linearly dependent over $\mathbb{Z}_2$ or $\mathbb{Z}_{q'}$, otherwise 0.*

*Then, if $\ell = \lambda^2$ the game's outcome is 0 with probability $1 - \mathsf{negl}(\lambda)$.*

*Proof Overview.* Using counting arguments akin to the previous ones, we can upper bound the probability that $\lambda$ random independent 0-1 vectors are linearly dependent over $\mathbb{Z}_2$. Because of the randomness embedded in the ROM, the adversary can only query $\mathsf{H}$ until it finds a group of vectors with this property. However, for any $\lambda^d$ number of random oracle calls there are a total of $\binom{\lambda^d}{\lambda} \leq e^{\lambda}\lambda^{\lambda(d-1)}$ distinct groups of $\lambda$ vectors. If $B_2$ is the bound found at the previous point, a union bound implies that $B_2 e^{\lambda}\lambda^{\lambda(d-1)}$ upper bounds the probability that $\lambda$ linearly dependent vectors exist. With a similar strategy we can get an upper bound of the type $B_{q'} e^{\lambda}\lambda^{\lambda(d-1)}$ for linear dependency modulo $q'$. One can show that both $B_2$ and $B_{q'}$'s shrinkage far exceeds the growth of $e^{\lambda}\lambda^{\lambda(d-1)}$ in the parameter regime that is of interest to us. This implies the claimed result.

*Extended Argument.* We start with an auxiliary statistical lemma concerning the existence of small subset of linearly dependent vectors inside a larger set of uniformly random $q$-ary vectors. In essence, the lemma says that, in the parameter regime that is of interest for us, such a subset does not exist with overwhelming probability.

**Lemma 9.** *Let $c$ and $d$ denote positive integer constants. Suppose you are given $N = \lambda^d$ random vectors $\{v_i\}_{i\in[N]}$, taking values in $\mathbb{F}_2^{\lambda^2}$. Let $E$ denote the event that $c\lambda$ linearly dependent vectors exist within the supplied set*

$$\exists F \subseteq [N] : |F| = c\lambda \wedge \{v_i\}_{i\in F} \text{ are linearly dependent.}$$

*Then $\Pr[E] \leq \mathsf{negl}(\lambda)$.*

*Proof.* We immediately rule out the case $\lambda^d < c\lambda$, as in that case $\Pr[E] = 0$. Via a simple row-by-row counting argument, the probability that $\lambda$ random vectors sampled independently from $\mathbb{F}_2^{\lambda^2}$ are linearly independent is

$$\prod_{i=0}^{c\lambda-1} (1 - 2^{i-\lambda^2}) \geq (1 - 2^{c\lambda-1-\lambda^2})^{c\lambda}.$$

Then by Bernoulli inequality the probability that said vectors are linearly dependent is upper bounded by

$$1 - (1 - 2^{c\lambda-1-\lambda^2})^{c\lambda} \leq c\lambda/2^{\lambda^2-c\lambda+1}.$$

Given our $N$ input vectors, there are exactly $\binom{N}{c\lambda}$ possible groups of $\lambda$ vectors, each subject to the upper bound above. By a union bound

$$\Pr[E] \leq \binom{\lambda^d}{c\lambda} \frac{\lambda}{2^{\lambda^2-c\lambda+1}}.$$

Using standard upper bounds to binomial coefficients we get

$$\Pr[E] \leq \frac{\lambda^{(d-1)c\lambda} e^{\lambda}}{c^{c\lambda} 2^{\lambda^2-c\lambda+1}}.$$

The denominator vastly outgrows the numerator, which concludes our proof. □

**Lemma 10.** *Let $c = \log(3/2)$, and let $d$ denote a generic positive integer constants. Suppose you are given $N = \lambda^d$ random vectors $\{v_i\}_{i\in[N]}$, taking values in $\mathbb{F}_p^{\lambda^2}$ for prime $p$ with bitlength $\lambda$. Let $E$ denote the event that $c\lambda$ linearly dependent vectors exist within the supplied set*

$$\exists F \subseteq [N] : |F| = c\lambda \wedge \{v_i\}_{i\in F} \text{ are linearly dependent.}$$

*Then* $\Pr[E] \leq \mathsf{negl}(\lambda)$.

*Proof.* We again rule out the case $\lambda^d < c\lambda$ since then $\Pr[E] = 0$. By Lemma 3, considering a group of $c\lambda$ random such vectors the probability that they are linearly dependent is no more than $\lambda(3/4)^{\lambda^2}$. By a union bound over the $\binom{\lambda^d}{c\lambda}$ possible groups, we get an upper bound

$$\lambda \cdot 3^{\lambda^2} \cdot \frac{e^{\lambda}\lambda^{(d-1)c\lambda}}{c^{c\lambda}4^{\lambda^2}}.$$

This is again negligible in $\lambda$. □

The previous lemmas enable the main result about the unfeasibility of triggering inconsistencies when using the random oracle, which is formally captured by Theorem 7. We formally prove the latter for the sake of completeness.

*Proof ((Sketch) of Theorem 7).* Let $q = \mathsf{poly}(\lambda)$ be the number of (distinct) queries by $\mathcal{A}$ to $\mathsf{H}$, which we can assume to be upper bounded by $\lambda^d$ for constant $d$. The corresponding outputs are $q$ uniformly random bitstrings. Furthermore, by the winning condition, the game returns 1 iff then the output $F \subseteq [q]$ has cardinality $c\lambda$ and contains 0-1 vectors that are linearly dependent over either $\mathbb{Z}_2$ or $\mathbb{Z}_q$. The probability that such a set exists is, by a union bound and the previous two lemmas, no more than

$$\lambda \cdot 3^{\lambda^2} \cdot \frac{e^{\lambda}\lambda^{(d-1)c\lambda}}{c^{c\lambda}4^{\lambda^2}} + \frac{\lambda^{(d-1)c\lambda} e^{\lambda}}{c^{c\lambda} 2^{\lambda^2-c\lambda+1}}.$$

The latter is the sum of two negligible functions, and is therefore negligible itself. □

### 7.3 Verifiability through zkSNARKs

Concretely, verifiability requires a proof that a given PRF evaluation was computed correctly with respect to a public commitment to the secret key $k$. To preserve pseudorandomness, such a proof must not leak any information about $k$, naturally calling for a *zero-knowledge* proof [BFM88].

To ensure practicality, we consider succinct non-interactive arguments of knowledge (zkSNARKs) [Mic94], which offer short proofs and fast verification. Generic zkSNARK frameworks such as Groth16 [Gro16], Plonk [GWC19], Bulletproofs [BBB+18], and others [MBKM19,CFF+21,Set20] are expressive enough to

capture essentially arbitrary computations, including the evaluation of a PRF. Importantly, these systems have been the focus of extensive recent study [FFK+23,KPT23,DG23,FFR24,CFR25,LPS25], and it is by now well understood that many such SNARKs (sometimes with minor modifications or via ad-hoc compilers [GKO+23,CF24,BCC+25]) achieve stronger guarantees such as *non-malleability* [DDO+01] and universal composability [Can01]. This further reinforces their suitability for applications requiring public verifiability.

However, this generality comes at a cost. Generic SNARKs require *arithmetization* of the computation to be proved—translating the PRF into an equivalent arithmetic circuit. This process can introduce substantial overhead, especially when the PRF relies on algebraic operations such as group exponentiations or elliptic-curve computations. The resulting circuit depth and size directly affect the core efficiency metrics of the proof system: the size of the common reference string (CRS), the proof length, and the prover and verifier running time. Consequently, while general-purpose SNARKs provide a versatile tool, applying them in this context may entail unnecessarily heavy machinery.

To mitigate this, we present concrete instantiations tailored to our PRF constructions. In particular, we are interested in instantiating a zkSNARK for the following commit-and-prove relation:

$$\mathcal{R}_{\mathsf{PRF}} := \left\{ \begin{matrix} \mathbb{x} = (\mathsf{ck}, \mathbf{com}_k, x, y) \\ \mathbb{w} = (k, \rho_k) \end{matrix} : \begin{matrix} \mathbf{com}_k = \mathsf{Comm}(\mathsf{ck}, k; \rho_k) \\ f(k, x) = y \end{matrix} \right\}$$

that, roughly speaking, allows the prover to convince the verifier that a public output $y$ is the correct evaluation of a (bounded-equivocable) PRF $f$ at a public input $x$, using a secret key $k$ held by the prover and committed as $\mathbf{com}_k$.

*DDH-based.* We start by showing how to (semi-generically) instantiate $\mathcal{R}_{\mathsf{PRF}}$ when the PRF is the DDH-based construction described in Section 5. Our approach is based on the following observations. The expression $\mathsf{PRF}(k, x)$ can be interpreted as computing a multiplicative circuit over the subset $\{k_i \mid x_i = 1\}$, with the result multiplied by $k_0$. Thus, the exponent of $g$ is the product $z = k_0 \cdot \prod_{i:x_i=1} k_i$. The computation of $z$ can be modeled as a binary tree of multiplication gates, where the leaves are selected $k_i$ values, and the tree computes their product (combined with $k_0$). This circuit can be expressed efficiently and has logarithmic depth. In particular, it is amenable to optimized applications of the GKR protocol [GKR15], such as those in [Tha13], which provides efficient interactive public-coin arguments (and can be compiled into zkSNARKs, via Fiat-Shamir transform) for such structured computations. To enable the verifier to efficiently derive the exact committed values for this *grand product* argument, from a known input $x$, we encode the secret key $k = (k_0, \ldots, k_n)$ via individual Pedersen commitments to each $k_i$, i.e., the prover commits to each scalar $k_i$ separately using randomness $r_i$, producing commitments $\mathbf{com}_i = \mathsf{Comm}(k_i; r_i)$.

We highlight a few important aspects of our construction. First, it requires no trusted setup: both the Pedersen commitments and the underlying zkSNARK are transparent. Security relies on the DDH assumption (and the ROM for applying Fiat-Shamir). Shrinking the commitment to the PRF key (e.g., using vector commitments [CF13]) is possible, but we avoid doing so since the tradeoff space depends heavily on application-specific requirements. Moreover, since the key is committed only once and reused across multiple proofs, we do not view this as a significant drawback in practical scenarios.

*LWR-based.* We now describe an efficient instantiation of a zkSNARK for the relation $\mathcal{R}_{\mathsf{PRF}}$ when the underlying pseudorandom function is based on the LWR assumption. This construction can be interpreted as an inner-product argument (IPA) between the secret key and the input value, followed by a rounding operation. More precisely, the output of the PRF corresponds to the inner product between the key and input vector, scaled by a factor of $p/q$, and then rounded down (i.e., the floor function is applied).

The main idea is as follows. Once the key has been committed, the prover can demonstrate correctness of the PRF evaluation via a commit-and-prove IPA, followed by an appropriate implementation of the rounding operation, which can then be expressed either generically (e.g., using a bit-decomposition argument) or more efficiently via a lookup operation [BCG+18,ZBK+22,CFF+24,STW24]. Importantly, the lookup table used in this context is *structured* in the sense defined by [STW24], which enables efficient realization; however, the latter approach fails at achieving post-quantum security due to the reliance on the hardness

of discrete log. To preserve post-quantum security, it is nevertheless possible to encode the operation as an arithmetic constraint system and then rely on general-purpose zkSNARKs that are compiled using IOP-based polynomial commitment schemes such as FRI [BBHR18], Ligero [AHIV23], Brakedown [GLS+23] and WHIR [ACFY25].

### 7.4 Applications

*Encryption.* Due to its similarity with SIM-AC PRFs, our primitive inherits many of the applications to symmetric encryption outlined in [JT20a], except that if a property crucially relies on equivocability it holds up to the equivocation threshold. As an example, we outline the application of our primitive to *bounded-equivocable* encryption, the bounded version of equivocable encryption as defined in [JKX18]. From a high-level viewpoint, the definition can be viewed as a symmetric-key variant of non-committing encryption [CFGN96,CPR17,BBD+20], wherein a sender can encrypt arbitrary plaintexts while producing ciphertexts that are not binding to them. In [JKX18] the authors give a game-based definition of equivocable encryption where a PPT adversary should distinguish between the real world, where ciphertexts are generated by honestly evaluating the PRF, and the ideal world, where a simulator generates ciphertexts with no information about the key and the plaintexts. Furthermore, at the end of the experiment (but before outputting the decision bit) the adversary receives the honestly generated and used key (in the real world) or a simulated key (in the ideal world)[4]. Equivocable PRFs enable a bounded version of equivocable encryption by mimicking the standard construction of IND-CPA encryption from PRFs. That is, in the ideal world you encrypt $m$ by sampling uniformly random $r$, computing $c \leftarrow \mathsf{EqPRF.Eval}(k, r) \oplus m$, and returning $y = (c, r)$. After issuing up to $q$ ciphertexts $(\boldsymbol{c}, \boldsymbol{r})$ with a $q$-secure equivocable PRF, the game simply returns the key $k$. In the ideal world, the simulator essentially returns simulated PRF outputs through the $\mathsf{EqEval}$ interface, and finally derives the secret key $k'$ through the $\mathsf{EqKey}$ procedure by equivocating the PRF on input $r_i$ to output $m'_i \oplus c_i$ for $i \in [q]$. We envision an application of these constructions to threat scenarios akin to that of anamorphic encryption [PPY22,BGH+24], where encryptors struggle against a malicious dictator who forces them to share their secret key with them. Specifically, consider a dictator who seizes the power in an initially free country, where users are free to encrypt as they like. If users are forced to share their secret keys at the beginning of their regime, they can just return the fake $k'$, and shift the content of their messages to something that would not be compromising under the new regime. If instead no *coup d'etat* ever takes place, the unbounded pseudorandomness of the bounded-equivocable PRF guarantees that all present and future encryptions remain IND-CPA secure.

*Threshold Schnorr Signatures.* A recent work by Gerhart et al. [GLCRS25] proposed a new adaptively secure threshold Schnorr signature protocol. Their two-round scheme requires that in the first round every active signer $i$ evaluates a secret polynomial to obtain the exponent $r_i$, and outputs a group element of the form $R_i = g^{r_i} \cdot \mathsf{H}_0(\mathbf{d})^{w_i} \cdot \mathsf{H}_1(\mathbf{d})^{u_i}$[5]. Furthermore, signers must issue a non-interactive zero-knowledge proof of well-formedness of $R_i$ with respect to the committed polynomial and the public inputs. The secret polynomials are drawn uniformly at random at setup time, and ensure verifiability as well as a uniformly random output for a limited number of evaluations, after which a collective update phase is required. Our primitive comes with properties that make it fit for replacing the random polynomial in said scheme. Being pseudorandom, the derived $r_i$ are indistinguishable from uniformly random independent exponents to any computationally bounded adversary. Its instantiatability in the standard model (or, in the ROM with the latter only evaluated on public inputs) make it publicly verifiable, as outlined in the previous section. Finally, the property of equivocability is inherently compatible with adaptive corruptions. Specifically, following the reduction of [GLCRS25], we can at some point replace the equivocable PRF with a uniformly random, and consistently answer an adversarial corruption request by running the $\mathsf{EqKey}$ function and returning a

[4] Jaeger and Tyagi later found a gap [JT20a, Sec 4.1] in the definition of [JKX18], and provide a corrected version of the definition in the same work. However, they point out that the gap only emerges when considering ideal-models, and the original definition remains sound in the standard model.

[5] Here, $\mathsf{H}_0$ and $\mathsf{H}_1$ are two random oracles, $\mathbf{d}$ is a public session id, and $w_i$ and $u_i$ are two secret exponents.

consistent key. This is feasible as long as the corruption request takes place before the equivocation threshold is exceeded; afterwards, one can simply run an update phase following the blueprint of [GLCRS25].

*Proof of Stake.* We illustrate how our primitive can be used to implement a *lottery*, a fundamental component in proof-of-stake (PoS) blockchains. PoS blockchains rely on lottery-based leader election to randomly select one or more participants from the network to propose new blocks and drive consensus. The probability of being elected is proportional to the participant's stake, i.e., the amount of money/assets they have committed to the system.

From a security perspective, a well-designed lottery must ensure fairness, such that corrupt parties are not disproportionately favored. Additionally, it should preserve the privacy of elected leaders until they are ready to act, preventing adversaries from targeting them before they can contribute. Leaders should privately learn whether they have been elected and receive a publicly verifiable *winning ticket*, which they can attach to their messages to prove eligibility.

A simple and elegant construction was introduced in Algorand [GHM+17]. In this scheme, each participant is identified by a VRF public key, which acts as a commitment to a secret function $f_i$. Given a randomness seed $r$, party $i$ computes $f_i(r)$ and checks whether the output falls below a threshold $k$, indicating a win. Due to the pseudorandomness of the VRF, the outcome is unpredictable, and the verifiability property ensures that the winner can prove their legitimacy.

Following the same approach as in previous cases, we can instantiate the VRF using an equivocable PRF to handle adaptive corruptions. Specifically, we can simulate the equivocable PRF with a uniformly random function and consistently respond to adversarial corruption queries by equivocating the key. A crucial observation is that such equivocation remains feasible as long as it occurs before the equivocation threshold is surpassed.

## References


AA16. Jacob Alperin-Sheriff and Daniel Apon. Dimension-preserving reductions from LWE to LWR. Cryptology ePrint Archive, Report 2016/589, 2016.

ACFY25. Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. WHIR: Reed-solomon proximity testing with super-fast verification. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part IV*, volume 15604 of *LNCS*, pages 214–243. Springer, Cham, May 2025.

AHIV23. Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: lightweight sublinear arguments without a trusted setup. *DCC*, 91(11):3379–3424, 2023.

Ajt96. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *28th ACM STOC*, pages 99–108. ACM Press, May 1996.

AKPW13. Joël Alwen, Stephan Krenn, Krzysztof Pietrzak, and Daniel Wichs. Learning with rounding, revisited - new reduction, properties and applications. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 57–74. Springer, Berlin, Heidelberg, August 2013.

BBB+18. Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy*, pages 315–334. IEEE Computer Society Press, May 2018.

BBD+20. Zvika Brakerski, Pedro Branco, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Constant ciphertext-rate non-committing encryption from standard assumptions. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 58–87. Springer, Cham, November 2020.

BBHR18. Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast reed-solomon interactive oracle proofs of proximity. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *ICALP 2018*, volume 107 of *LIPIcs*, pages 14:1–14:17. Schloss Dagstuhl, July 2018.

BCC+25. Christian Badertscher, Matteo Campanelli, Michele Ciampi, Luigi Russo, and Luisa Siniscalchi. Universally composable SNARKs with transparent setup without programmable random oracle. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part VII*, volume 16006 of *LNCS*, pages 225–258. Springer, Cham, August 2025.

BCG+18. Jonathan Bootle, Andrea Cerulli, Jens Groth, Sune K. Jakobsen, and Mary Maller. Arya: Nearly linear-time zero-knowledge proofs for correct program execution. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part I*, volume 11272 of *LNCS*, pages 595–626. Springer, Cham, December 2018.

BCK96. Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In Neal Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 1–15. Springer, Berlin, Heidelberg, August 1996.

BDK+11. Boaz Barak, Yevgeniy Dodis, Hugo Krawczyk, Olivier Pereira, Krzysztof Pietrzak, François-Xavier Standaert, and Yu Yu. Leftover hash lemma, revisited. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 1–20. Springer, Berlin, Heidelberg, August 2011.

BDWY12. Mihir Bellare, Rafael Dowsley, Brent Waters, and Scott Yilek. Standard security does not imply security against selective-opening. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 645–662. Springer, Berlin, Heidelberg, April 2012.

BFM88. Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *20th ACM STOC*, pages 103–112. ACM Press, May 1988.

BGG+18. Dan Boneh, Rosario Gennaro, Steven Goldfeder, Aayush Jain, Sam Kim, Peter M. R. Rasmussen, and Amit Sahai. Threshold cryptosystems from threshold fully homomorphic encryption. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 565–596. Springer, Cham, August 2018.

BGH+24. Fabio Banfi, Konstantin Gegier, Martin Hirt, Ueli Maurer, and Guilherme Rito. Anamorphic encryption, revisited. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part II*, volume 14652 of *LNCS*, pages 3–32. Springer, Cham, May 2024.

BGM+16. Andrej Bogdanov, Siyao Guo, Daniel Masny, Silas Richelson, and Alon Rosen. On the hardness of learning with rounding over small modulus. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A, Part I*, volume 9562 of *LNCS*, pages 209–224. Springer, Berlin, Heidelberg, January 2016.

BHK12. Florian Böhl, Dennis Hofheinz, and Daniel Kraschewski. On definitions of selective opening security. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 522–539. Springer, Berlin, Heidelberg, May 2012.

BHLS25. Dan Boneh, Iftach Haitner, Yehuda Lindell, and Gil Segev. Exponent-VRFs and their applications. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part VII*, volume 15607 of *LNCS*, pages 195–224. Springer, Cham, May 2025.

BHY09. Mihir Bellare, Dennis Hofheinz, and Scott Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 1–35. Springer, Berlin, Heidelberg, April 2009.

BKR94. Mihir Bellare, Joe Kilian, and Phillip Rogaway. The security of cipher block chaining. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 341–358. Springer, Berlin, Heidelberg, August 1994.

BLMR13. Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key homomorphic PRFs and their applications. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 410–428. Springer, Berlin, Heidelberg, August 2013.

BPR12. Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 719–737. Springer, Berlin, Heidelberg, April 2012.

Bra12. Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical GapSVP. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 868–886. Springer, Berlin, Heidelberg, August 2012.

Can01. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001.

CF13. Dario Catalano and Dario Fiore. Vector commitments and their applications. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 55–72. Springer, Berlin, Heidelberg, February / March 2013.

CF24. Alessandro Chiesa and Giacomo Fenzi. zkSNARKs in the ROM with unconditional UC-security. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024, Part I*, volume 15364 of *LNCS*, pages 67–89. Springer, Cham, December 2024.

CFF+21. Matteo Campanelli, Antonio Faonio, Dario Fiore, Anaïs Querol, and Hadrián Rodríguez. Lunar: A toolbox for more efficient universal and updatable zkSNARKs and commit-and-prove extensions. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part III*, volume 13092 of *LNCS*, pages 3–33. Springer, Cham, December 2021.

CFF+24. Matteo Campanelli, Antonio Faonio, Dario Fiore, Tianyu Li, and Helger Lipmaa. Lookup arguments: Improvements, extensions and applications to zero-knowledge decision trees. In Qiang Tang and Vanessa Teague, editors, *PKC 2024, Part II*, volume 14602 of *LNCS*, pages 337–369. Springer, Cham, April 2024.

CFGN96. Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *28th ACM STOC*, pages 639–648. ACM Press, May 1996.

CFR25. Matteo Campanelli, Antonio Faonio, and Luigi Russo. SNARKs for virtual machines are non-malleable. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part IV*, volume 15604 of *LNCS*, pages 153–183. Springer, Cham, May 2025.

CGH98. Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *30th ACM STOC*, pages 209–218. ACM Press, May 1998.

CJ25. Kaishuo Cheng and Joseph Jaeger. Adaptive security for constrained PRFs. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part V*, volume 16004 of *LNCS*, pages 598–627. Springer, Cham, August 2025.

CJJ+14. David Cash, Joseph Jaeger, Stanislaw Jarecki, Charanjit S. Jutla, Hugo Krawczyk, Marcel-Catalin Rosu, and Michael Steiner. Dynamic searchable encryption in very-large databases: Data structures and implementation. In *NDSS 2014*. The Internet Society, February 2014.

CPR17. Ran Canetti, Oxana Poburinnaya, and Mariana Raykova. Optimal-rate non-committing encryption. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 212–241. Springer, Cham, December 2017.

CS19. Chitchanok Chuengsatiansup and Damien Stehlé. Towards practical GGM-based PRF from (module-)learning-with-rounding. In Kenneth G. Paterson and Douglas Stebila, editors, *SAC 2019*, volume 11959 of *LNCS*, pages 693–713. Springer, Cham, August 2019.

DDO+01. Alfredo De Santis, Giovanni Di Crescenzo, Rafail Ostrovsky, Giuseppe Persiano, and Amit Sahai. Robust non-interactive zero knowledge. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 566–598. Springer, Berlin, Heidelberg, August 2001.

DG23. Quang Dao and Paul Grubbs. Spartan and bulletproofs are simulation-extractable (for free!). In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part II*, volume 14005 of *LNCS*, pages 531–562. Springer, Cham, April 2023.

FFK+23. Antonio Faonio, Dario Fiore, Markulf Kohlweiss, Luigi Russo, and Michal Zajac. From polynomial IOP and commitments to non-malleable zkSNARKs. In Guy N. Rothblum and Hoeteck Wee, editors, *TCC 2023, Part III*, volume 14371 of *LNCS*, pages 455–485. Springer, Cham, November / December 2023.

FFR24. Antonio Faonio, Dario Fiore, and Luigi Russo. Real-world universal zkSNARKs are non-malleable. In Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie, editors, *ACM CCS 2024*, pages 3138–3151. ACM Press, October 2024.

For99. Tony Forbes. Prime clusters and cunningham chains. *Mathematics of computation*, 68(228):1739–1747, 1999.

FP07. Reza Rezaeian Farashahi and Ruud Pellikaan. The quadratic extension extractor for (hyper) elliptic curves in odd characteristic. In *International Workshop on the Arithmetic of Finite Fields*, pages 219–236. Springer, 2007.

GGM84. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions (extended abstract). In *25th FOCS*, pages 464–479. IEEE Computer Society Press, October 1984.

GHM+17. Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. Cryptology ePrint Archive, Report 2017/454, 2017.

GK03. Shafi Goldwasser and Yael Tauman Kalai. On the (in)security of the Fiat-Shamir paradigm. In *44th FOCS*, pages 102–115. IEEE Computer Society Press, October 2003.

GKO+23. Chaya Ganesh, Yashvanth Kondi, Claudio Orlandi, Mahak Pancholi, Akira Takahashi, and Daniel Tschudi. Witness-succinct universally-composable SNARKs. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part II*, volume 14005 of *LNCS*, pages 315–346. Springer, Cham, April 2023.

GKR15. Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for muggles. *Journal of the ACM*, 62(4):27:1–27:64, 2015.

GLCRS25. Paul Gerhart, Davide Li Calsi, Luigi Russo, and Dominique Schröder. Fully-adaptive two-round threshold schnorr signatures from DDH. Cryptology ePrint Archive, Paper 2025/1478, 2025.

GLS+23. Alexander Golovnev, Jonathan Lee, Srinath T. V. Setty, Justin Thaler, and Riad S. Wahby. Brakedown: Linear-time and field-agnostic SNARKs for R1CS. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part II*, volume 14082 of *LNCS*, pages 193–226. Springer, Cham, August 2023.

Gro16. Jens Groth. On the size of pairing-based non-interactive arguments. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 305–326. Springer, Berlin, Heidelberg, May 2016.

GWC19. Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: Permutations over Lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Report 2019/953, 2019.

HILL99. Johan HÅstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

HPW15. Carmit Hazay, Arpita Patra, and Bogdan Warinschi. Selective opening security for receivers. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 443–469. Springer, Berlin, Heidelberg, November / December 2015.

HRW16. Dennis Hofheinz, Vanishree Rao, and Daniel Wichs. Standard security does not imply indistinguishability under selective opening. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 121–145. Springer, Berlin, Heidelberg, October / November 2016.

Jae23. Joseph Jaeger. Let attackers program ideal models: Modularity and composability for adaptive compromise. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part III*, volume 14006 of *LNCS*, pages 101–131. Springer, Cham, April 2023.

JKX18. Stanislaw Jarecki, Hugo Krawczyk, and Jiayu Xu. OPAQUE: An asymmetric PAKE protocol secure against pre-computation attacks. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 456–486. Springer, Cham, April / May 2018.

JT20a. Joseph Jaeger and Nirvan Tyagi. Handling adaptive compromise for practical encryption schemes. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 3–32. Springer, Cham, August 2020.

JT20b. Joseph Jaeger and Nirvan Tyagi. Handling adaptive compromise for practical encryption schemes. Cryptology ePrint Archive, Paper 2020/765, 2020.

KL21. Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. CRC Press, 3rd edition, 2021.

KPT23. Markulf Kohlweiss, Mahak Pancholi, and Akira Takahashi. How to compile polynomial IOP into simulation-extractable SNARKs: A modular approach. In Guy N. Rothblum and Hoeteck Wee, editors, *TCC 2023, Part III*, volume 14371 of *LNCS*, pages 486–512. Springer, Cham, November / December 2023.

LN97. Rudolf Lidl and Harald Niederreiter. *Finite Fields*, volume 20 of *Encyclopedia of Mathematics and Its Applications*. Cambridge University Press, 2nd edition, 1997.

LPS25. Helger Lipmaa, Roberto Parisella, and Janno Siim. On knowledge-soundness of plonk in ROM from falsifiable assumptions. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part VII*, volume 16006 of *LNCS*, pages 362–395. Springer, Cham, August 2025.

MBKM19. Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 2111–2128. ACM Press, November 2019.

Mic94. Silvio Micali. CS proofs (extended abstracts). In *35th FOCS*, pages 436–453. IEEE Computer Society Press, November 1994.

Mic21. Daniele Micciancio. Lecture 4: Random lattices and lattice-based cryptography. Lecture notes for CSE 206A, UC San Diego, 2021. Accessed: 2025-10-02.

MM11. Daniele Micciancio and Petros Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 465–484. Springer, Berlin, Heidelberg, August 2011.

MP12. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Berlin, Heidelberg, April 2012.

MR07. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM Journal on Computing*, 37(1):267–302, 2007.

MR09. Daniele Micciancio and Oded Regev. *Lattice-based Cryptography*, pages 147–191. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

MRV99. Silvio Micali, Michael O. Rabin, and Salil P. Vadhan. Verifiable random functions. In *40th FOCS*, pages 120–130. IEEE Computer Society Press, October 1999.

Nie02. Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 111–126. Springer, Berlin, Heidelberg, August 2002.

NPR99. Moni Naor, Benny Pinkas, and Omer Reingold. Distributed pseudo-random functions and KDCs. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 327–346. Springer, Berlin, Heidelberg, May 1999.

NR97. Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th FOCS*, pages 458–467. IEEE Computer Society Press, October 1997.

NRR00. Moni Naor, Omer Reingold, and Alon Rosen. Pseudo-random functions and factoring (extended abstract). In *32nd ACM STOC*, pages 11–20. ACM Press, May 2000.

NRSW20. Jonas Nick, Tim Ruffing, Yannick Seurin, and Pieter Wuille. MuSig-DN: Schnorr multi-signatures with verifiably deterministic nonces. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 1717–1731. ACM Press, November 2020.

Pei09. Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 333–342. ACM Press, May / June 2009.

PPY22. Giuseppe Persiano, Duong Hieu Phan, and Moti Yung. Anamorphic encryption: Private communication against a dictator. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 34–63. Springer, Cham, May / June 2022.

Reg05. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

Rog64. C. A. Rogers. *Packing and Covering*, volume 54 of *Cambridge Tracts in Mathematics and Mathematical Physics*. Cambridge University Press, 1964.

Set20. Srinath Setty. Spartan: Efficient and general-purpose zkSNARKs without trusted setup. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part III*, volume 12172 of *LNCS*, pages 704–737. Springer, Cham, August 2020.

STW24. Srinath T. V. Setty, Justin Thaler, and Riad S. Wahby. Unlocking the lookup singularity with Lasso. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part VI*, volume 14656 of *LNCS*, pages 180–209. Springer, Cham, May 2024.

Tha13. Justin Thaler. Time-optimal interactive proofs for circuit evaluation. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 71–89. Springer, Berlin, Heidelberg, August 2013.

YY98. Adam Young and Moti Yung. Finding length-3 positive cunningham chains and their cryptographic significance. In *International Algorithmic Number Theory Symposium*, pages 289–298. Springer, 1998.

ZBK+22. Arantxa Zapico, Vitalik Buterin, Dmitry Khovratovich, Mary Maller, Anca Nitulescu, and Mark Simkin. Caulk: Lookup arguments in sublinear time. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 3121–3134. ACM Press, November 2022.

## A Auxiliary Lemmas

In the following we report simple auxiliary lemmas that we leveraged in our results.

**Lemma 11.** *Let $U$ be a discrete set. Let $S_1, \ldots, S_k$ form a partition of $U$ such that $|S_1| = |S_2| = \cdots = |S_k|$. Consider the following distributions:*

- *$A$: sample a uniformly random $X \leftarrow\!\!\$\ U$. Let $Y$ be the (unique) subset of the above-mentioned partition that contains $X$. Return $(X, Y)$.*
- *$B$: sample a uniformly random $W \leftarrow\!\!\$\ \{S_1, \ldots, S_k\}$. Let $Z$ be uniformly random element of $W$. Return $(Z, W)$.*

*Then $A$ and $B$ are perfectly indistinguishable.*

*Proof.* At first, consider only the distributions $X$ and $Z$. For each $u \in U$, $\Pr[X = u] = 1/|U|$. Instead

$$\Pr[Z = u] = \frac{1}{k|w|} = (k \cdot |U|/k)^{-1} = 1/|U|.$$

Therefore, $X$ and $Z$ are identically distributed. Furthermore, let $f$ be the function mapping an element $u \in U$ to its subset. Notice that $f$ is a deterministic function such that $Y = f(X)$ and $W = f(Z)$. Therefore, indistinguishability between and $X$ and $Z$ implies the same for $A$ and $B$. □

**Lemma 12.** *Let $D$ be uniformly random $\{0,1\}$-matrix of size $L \times l$. Let $R_2$ and $R_p$ denote two uniformly random vectors over $\mathbb{Z}_2^L$ and $\mathbb{Z}_p^L$ respectively, while $y_2$ and $y_p$ are uniformly random over $\mathbb{Z}_2^l$ and $\mathbb{Z}_p^l$ respectively. Assume that*

$$\Delta[(D, y_2, Dy_2), (D, \alpha_2 \leftarrow\!\!\$\ \mathcal{I}_{R_2}(D), R_2)] \leq \epsilon_2,$$
$$\Delta[(D, y_p, Dy_p), (D, \alpha_p \leftarrow\!\!\$\ \mathcal{I}_{R_p}(D), R_p)] \leq \epsilon_p.$$

*If $g(\cdot, \cdot)$ denotes the CRT recombination function, then*

$$\Delta[(D, g(y_2, y_p), g(Dy_2, Dy_p)), (D, g(\alpha_2, \alpha_p), g(R_2, R_p)] \leq \min(\epsilon_2, \epsilon_p)$$

*Proof.* Consider the distributions $(D, g(y_2, y_p) \mod 2, g(Dy_2, Dy_p) \mod 2)$ and $(D, g(\alpha_2, \alpha_p) \mod 2, g(R_2, R_p) \mod 2$. Since reducing modulo 2 is a deterministic function, the statistical distance between these distributions cannot exceed that of the original (i.e., modulo 2p) distributions. By assumption, the latter is exactly $\epsilon_2$. Similarly, by reasoning modulo $p$, we can upper bound the target statistical distance by $\epsilon_p$. The claim immediately follows. □

**Lemma 13.** *Let $D$ be a $\{0,1\}$-matrix of size $L \times l$ sampled from some distribution $\chi$. Let $R_2$ and $R_p$ denote two uniformly random vectors over $\mathbb{Z}_2^L$ and $\mathbb{Z}_p^L$ respectively, while $y_2$ and $y_p$ are uniformly random over $\mathbb{Z}_2^l$ and $\mathbb{Z}_p^l$ respectively. Assume that*

$$(D, y_2, Dy_2) \equiv_C (D, \alpha_2 \leftarrow\!\!\$\ \mathcal{I}_{R_2}(D), R_2),$$

$$(D, y_p, Dy_p) \equiv_C (D, \alpha_p \leftarrow\!\!\$\ \mathcal{I}_{R_p}(D), R_p).$$

*If $g(\cdot, \cdot)$ denotes the CRT recombination function, then*

$$(D, g(y_2, y_p), g(Dy_2, Dy_p)) \equiv_C (D, g(\alpha_2, \alpha_p), g(R_2, R_p).$$

*Proof.* Suppose, by contradiction, that an efficient PPT distinguisher $\mathcal{A}$ has a non-negligible advantage $\epsilon$ when distinguishing $(D, g(y_2, y_p), g(Dy_2, Dy_p))$ from $(D, g(\alpha_2, \alpha_p), g(R_2, R_p)$. Consider the following sequence of game hops:

1. $\mathsf{Game}_0$: $\mathcal{A}$ always receives a sample from $(D, g(y_2, y_p), g(Dy_2, Dy_p))$. We generate the sample by sampling $D \leftarrow\!\!\$\ \chi$, computing $(y_2, Dy_2)$ and $(y_p, Dy_p)$, and then combining them via CRT.

2. $\mathsf{Game}_1$: we change the way in which we generate the sample. Specifically, instead of sampling $(y_p, Dy_p)$ we replace it with $(\alpha_p, R_p)$, with both terms defined as above. We claim this is computationally indistinguishable from the previous one by reducing to the computational indistinguishability between $(D, y_p, Dy_p)$ and $(D, \alpha_p \leftarrow\!\!\$\, \mathcal{I}_{R_p}(D), R_p)$. That is

$$|\Pr[\mathsf{Game}_0 \to 1] - \Pr[\mathsf{Game}_1 \to 1]| \leq \epsilon_p.$$

3. $\mathsf{Game}_2$: we further change the sampling strategy. Akin to the previous hop, instead of sampling $(y_2, Dy_2)$ we replace it with $(\alpha_2, R_2)$. The resulting game is computationally indistinguishable from the previous one.

$$|\Pr[\mathsf{Game}_1 \to 1] - \Pr[\mathsf{Game}_2 \to 1]| \leq \epsilon_2.$$

Note that in the last game we are essentially sampling according to $(D, g(\alpha_2, \alpha_p), g(R_2, R_p)$. Hence any PPT $\mathcal{A}$ can have an advantage at most $\epsilon_2 + \epsilon_p$. □

# B Technical Arguments Pertaining to Section 5

## B.1 Indistinguishability Argument

Fix $D \in \mathbb{Z}_p^{L \times \ell}$ for a generic prime $p$, with $L$ and $\ell$ as defined before. We define for every $R \in \mathbb{Z}_p^L$ the set

$$\mathcal{I}_R(D) := \{\boldsymbol{y} \in \mathbb{Z}_p^\ell | D\boldsymbol{y} = R\}$$

which we can informally view as the "bucket" containing all the $\boldsymbol{y}$ vectors such that $Dy = R$. We note that for distinct $R \neq S$, $\mathcal{I}_R(D) \cap \mathcal{I}_S(D) = \emptyset$. Furthermore, assuming $D$ has rank $r$, each "bucket" has size $p^{\ell - r}$. Finally, note that each $\boldsymbol{y} \in \mathbb{Z}_p^\ell$ belongs to one bucket, namely $\boldsymbol{y} \in \mathcal{I}_{Dy}(D)$. Therefore, we can partition the space $\mathbb{Z}_p^\ell$ into $p^r$ "buckets" $\mathcal{I}_R(D)$ of $p^{\ell - r}$ elements each. Furthermore, each bucket corresponds to exactly one output value $R \in \mathbb{Z}_p^L$.

*Uniformly Random $D$.* With the established preliminaries in mind, we outline a simple hybrid argument:

- $H_0$: you sample a uniformly random binary matrix $D$, $y \leftarrow\!\!\$\, \mathbb{Z}_q^\ell$, and output $(D, y, R) = (D, y, Dy)$.
- $H_1$: you sample a uniformly random binary matrix $D$ as above. If $D$ has rank $L$, sample $\boldsymbol{y} \leftarrow\!\!\$\, \mathbb{Z}_q^\ell$ and output $(D, \boldsymbol{y}, R) = (D, \boldsymbol{y}, D\boldsymbol{y})$; otherwise, you abort. Since the probability of having rank $L$ is overwhelming, the hybrids are statistically indistinguishable. More precisely, they are within statistical distance

$$\Delta(H_0, H_1) \leq \left(\frac{3}{4}\right)^{\lambda^2} \cdot (1 + \lambda \log(3/2)),$$

  by the reasoning of Theorem 3.
- $H_2$: you sample a uniformly random binary $D$ as in $H_1$ (and still abort if the rank is not $L$). Next, you sample $R \in \mathbb{Z}_p^L$ and pick a uniformly random $y \leftarrow\!\!\$\, \mathcal{I}_R(D)$. Finally, you output $(D, \boldsymbol{y}, R)$.
  To show the indistinguishability of $H_2$ and $H_1$ we argue that the hybrids are indistinguishable for any fixed $D$ such that $r_{q'}(D) = L$. We call the two hybrids obtained by fixing $D$ as such (instead of sampling it at random) $H_1'$ and $H_2'$ respectively. Let us apply the aforementioned partition argument to our setting. We now have $p^L$ disjoint buckets of size $p^{\ell - L}$ each. We analyze the two modified hybrids separately:
  - In $H_2'$, you pick a random bucket $\mathcal{I}_R(D) \subseteq \mathbb{Z}_p^\ell$ and a random point $y$ inside of it.
  - In $H_1'$, you pick a random point $R \in \mathbb{Z}_p^\ell$, retrieve $\mathcal{I}_R(D)$, and then pick a random point $\boldsymbol{y} \leftarrow\!\!\$\, \mathcal{I}_R(D)$ in the bucket.

  The two distributions are clearly the same. We attach a formal proof in Section A (Lemma 11) for the sake of rigor. Since this is true for all fixed $D$, the same holds for any sampling of $D$, as long as the sampling is identical in both hybrids. Thus, in particular, the claim must hold when $D$ is randomly sampled as per $H_1$ and $H_2$'s specifications.

In conclusion, we get that $\Delta(H_0, H_2) \leq \left(\frac{3}{4}\right)^{\lambda^2} \cdot (1 + \lambda \log(3/2))$.

Next, we analyze the same hybrids, but modulo 2. In this case, our analysis is simplified by well-established statistical tools such as Lemma 1 (i.e., the leftover hash lemma). For each binary $L \times \ell$ matrix $D$ we set $f_D : \{0,1\}^\ell \to \{0,1\}^L$. It is a well-known fact that the family $F = \{f_D | D \in \mathbb{Z}_2^{L \times \ell}\}$ is universal. Therefore, if $y \leftarrow\!\$\ \mathbb{Z}_2^\ell$ the leftover hash lemma states that $(D, Dy)$ is *statistically* indistinguishable from $(D, R)$, with $R$ uniformly random and independent over $\mathbb{Z}_2^L$. More precisely, the two distributions are within distance $2^{(L-\ell)/2}$, which is $\mathsf{negl}(\lambda)$ in our setting. Next, consider the hybrids $(D, Dy, \mathcal{I}_{Dy}(D))$ and $(D, R, \mathcal{I}_R(D))$. Notice that $\mathcal{I}_a(b)$ is a deterministic function of $a$ and $b$, so it cannot make indistinguishable distributions easier to distinguish. Finally, let $\alpha \leftarrow\!\$\ \mathcal{I}_{Dy}(D)$, $\beta \leftarrow\!\$\ \mathcal{I}_R(D)$, and consider the hybrids $(D, D\boldsymbol{y}, \alpha)$ and $(D, R, \beta)$. Because $\alpha$ and $\beta$ are obtained from $\mathcal{I}_{Dy}(D)$ and $\mathcal{I}_R(D)$ respectively, and through the same randomized function, the hybrids are indistinguishable.

We are almost done, albeit for two details calling for clarification. While we provided two independent indistinguishability arguments modulo 2 and $p$, we should somehow combine them into a single indistinguishability argument modulo $2p$. To this end, we can once again leverage the Chinese Remainder Theorem, which essentially implies that $\mathbb{Z}_{2p}$ is isomorphic to $\mathbb{Z}_2 \times \mathbb{Z}_p$ for prime $p$. From a high-level viewpoint, if pair $(a, b)$ is pseudorandom over $\mathbb{Z}_2 \times \mathbb{Z}_p$ then the corresponding $c \in \mathbb{Z}_{2p}$ such that $c \equiv a \mod 2$ and $c \equiv b \mod p$ is also pseudorandom. We provide a formal proof in Section A. Furthermore, we observe that the original linear system pertains to a matrix of the type $(\mathbf{1}|D)$, where $\mathbf{1}$ denotes a column of length $L$ consisting of 1s. This gap is easily filled by recalling how we got rid of the $\mathbf{1}$ column in Theorem 3, i.e., by picking a random $y_0 \in \mathbb{Z}_{q-1}$ and going from the initial linear system $(1|D) \cdot \begin{pmatrix} y_0 \\ \boldsymbol{y} \end{pmatrix} = R$ to $Dy = R - \mathbf{y_0}$ ($\mathbf{y_0}$ denotes a column of $y_0$'s). Thinking in reverse, we can view the initial system as the result of applying a randomized function to the final one. Concretely, let us start from the indistinguishability of $(D, Dy, \alpha)$ and $(D, R, \beta)$ modulo $q - 1 = 2p$, which we just proved. For $y_0 \leftarrow\!\$\ \mathbb{Z}_{q-1}$ it immediately follows that

$$\left((\mathbf{1}|D), Dy + \mathbf{y_0}, \begin{pmatrix} y_0 \\ \alpha \end{pmatrix}\right) \equiv_S \left((\mathbf{1}|D), R + \mathbf{y_0}, \begin{pmatrix} y_0 \\ \beta \end{pmatrix}\right),$$

and the two hybrids represent exactly the distributions pertaining to a honest PRF evaluation and an equivocated key, respectively.

*Pseudorandom $D$.* We again run two independent arguments modulo 2 and $q'$, starting with the latter. We already showed that for a fixed $D$ of rank $L$, the distributions $(D, \boldsymbol{y}, D\boldsymbol{y})$ and $(D, \boldsymbol{y}, R)$ with $y \leftarrow\!\$\ \mathcal{I}_R(D)$ are statistically close. With this in mind, consider the following hybrids:

- $H_0$: you sample a uniformly random key $\boldsymbol{k} \leftarrow\!\$\ \mathcal{K}$ and output

  $$(\boldsymbol{k}, \{x_i\}_{i \in [L]}, \{\mathsf{H}(\boldsymbol{k}, x_i)\}_{i \in [L]}, \boldsymbol{y}, D\boldsymbol{y})$$

  with

  $$D = \begin{pmatrix} \mathsf{H}(\boldsymbol{k}, x_1) \\ \vdots \\ \mathsf{H}(\boldsymbol{k}, x_L) \end{pmatrix},$$

  and $(x_1, \ldots, x_L)$ adversarially generated.
- $H_1$: we generate sample $(\boldsymbol{k}, \{x_i\}_{i \in [L]}, \{\mathsf{H}(k, x_i)\}_{i \in [L]}, \boldsymbol{y}, D\boldsymbol{y})$, but abort if $D$ has not rank greater or equal than $L$. As argued before, for any PPT adversary generating the $L$ PRF inputs $x_i$ the abort probability should be negligible, otherwise there exists an efficient algorithm to distinguish $\mathsf{H}$ from a uniformly random function. This in turn upper bounds the probability of distinguishing the two games.
- $H_2$: we generate $k$ and $D$ as before, but instead of sampling a uniformly random $\boldsymbol{y}$ we instead sample $R \leftarrow\!\$\ \mathbb{Z}_p^L$ and $\boldsymbol{y} \leftarrow\!\$\ \mathcal{I}_R(D)$. Then output $(\boldsymbol{k}, \{x_i\}_{i \in [L]}, \{\mathsf{H}(k, x_i)\}_{i \in [L]}, \boldsymbol{y}, R)$. By recycling the previous reasoning, the $D$ matrix obtained in $H_1$ and the one implicit in $H_2$ are both full-rank, and sampled according to the same strategy. Therefore, the two hybrids are statistically indistinguishable.

We now move on to the indistinguishability argument modulo 2. While in the previous section we relied on the leftover hash lemma for a simpler argument, we slightly deviate from that approach. Instead, we get to the same conclusion by recycling the modulo argument modulo $q'$ and adapting it to $\mathbb{F}_2$. That is, we again use a sequence of three hybrids akin to the previous one, with the major difference being that we view $D$,$y$, and $R$ as elements of $\mathbb{F}_2$.

Finally, we ought to combine the two arguments into a single one that shows indistinguishability modulo $2q'$. The details of this step resemble those of the analogous step in the previous section. We defer a formal proof to Section A.

*$t$-Wise Independent Hash.* If we tune $t$ to be larger or equal than $L$, and fix $\mathsf{H}$ to be a $t$-wise independent hash instead, the previous argument immediately carries over.

# C Technical Arguments Pertaining to Section 6

## C.1 Indistinguishability Argument

We start with a simple three-step hybrid argument, concerning the indistinguishability of the outer key.

- We start from hybrid $H_0 : (D, \boldsymbol{k}, D\boldsymbol{k})$, with $D \leftarrow\!\!\$\, \mathbb{Z}_q^{L\times\lambda}$ and $\boldsymbol{k} \leftarrow\!\!\$\, \mathbb{Z}_q^{\lambda}$.
- Let $H_1 : (D, \boldsymbol{k}, D\boldsymbol{k})$ with $D$ sampled at random from the set invertible matrices $L \times \lambda$ matrices over $\mathbb{Z}_q$. $H_1$ and $H_1$ are statistically indistinguishable, as defined above.
- Let $H_2 : (D, D^{-1}R, R)$ with $R \leftarrow\!\!\$\, \mathbb{Z}_q^{L}$. Notice that this hybrid is identical to $H_1$ up to renaming. Hence, $H_0$ and $H_2$ are statistically indistinguishable.
- Finally, the indistinguishability of $H_0$ and $H_2$ implies that of $(D, \boldsymbol{k}, \lfloor D\boldsymbol{k} \rfloor_p)$ and $(D, D^{-1}R, \lfloor R \rfloor_p)$. Indeed, because $\lfloor \cdot \rfloor_p$ is a deterministic function, a distinguisher for these two distributions can be easily converted into one discriminating between $H_0$ and $H_2$.

Finally, the inner key $\boldsymbol{k}_{in}$ can be dealt with as in the DDH-based construction: you generate it on the fly at equivocation time, use it to evaluate $\mathsf{H}(\boldsymbol{k}_{in}, \cdot)$ and return it along with $\boldsymbol{k}_{out}$