# A Generic Approach to Constructing and Proving Verifiable Random Functions

Rishab Goyal [*] Susan Hohenberger[†] Venkata Koppula[‡] Brent Waters[§]

**Abstract**

Verifiable Random Functions (VRFs) as introduced by Micali, Rabin and Vadhan are a special form of Pseudo Random Functions (PRFs) wherein a secret key holder can also prove validity of the function evaluation relative to a statistically binding commitment.

Prior works have approached the problem of constructing VRFs by proposing a candidate under specific number theoretic setting — mostly in bilinear groups — and then grapple with the challenges of proving security in the VRF environments. These constructions achieved different results and tradeoffs in practical efficiency, tightness of reductions and cryptographic assumptions.

In this work we take a different approach. Instead of tackling the VRF problem as a whole we demonstrate a simple and generic way of building Verifiable Random Functions from more basic and narrow cryptographic primitives. Then we can turn to exploring solutions to these primitives with a more focused mindset. In particular, we show that VRFs can be constructed generically from the ingredients of: (1) a 1-bounded constrained pseudo random function for a functionality that is "admissible hash friendly" , (2) a non-interactive statistically binding commitment scheme (without trusted setup) and (3) a non-interactive witness indistinguishable proofs or NIWIs. The first primitive can be replaced with a more basic puncturable PRF constraint if one is willing to settle for selective security or assume sub-exponential hardness of assumptions.

In addition, we support our generic approach by giving new constructions of constrained PRFs under non bilinear groups and new constructions of perfectly binding commitments from the Learning with Errors (LWE) and Learning Parity with Noise (LPN) assumptions.

## 1 Introduction

Verifiable Random Functions (VRFs) as introduced by Micali, Rabin and Vadhan [MRV99] are a special form of Pseudo Random Functions (PRFs) [GGM84] wherein a secret key holder can also prove validity of the function evaluation relative to a statistically binding commitment. The caveat being that the pseudorandomness of the function on other points should not be sacrificed even after providing polynomially many proofs. The VRF definition forbids *interactivity* or any *setup assumption*, thereby disallowing trivial extensions of PRFs making the problem more challenging and interesting.

Prior works [Lys02, Dod02, DY05, HW10, Jag15, HJ16] have approached the problem of constructing VRFs by proposing a candidate under specific number theoretic setting — mostly in bilinear groups — and then grapple with the challenges of proving security in the VRF environments. These constructions achieved different results and tradeoffs in practical efficiency, tightness of reductions and cryptographic assumptions.

In this work we take a different approach. Instead of tackling the VRF problem as a whole we demonstrate a simple and generic way of building Verifiable Random Functions from more basic and narrow cryptographic primitives. Then we can turn to exploring solutions to these primitives with a more focused mindset.

[*] University of Texas at Austin. Email: goyal@utexas.edu.

[†] Johns Hopkins University. Email: susan@cs.jhu.edu. Supported by the National Science Foundation (NSF) CNS-1228443 and CNS-1414023, the Office of Naval Research under contract N00014-14-1-0333, and a Microsoft Faculty Fellowship

[‡] University of Texas at Austin. Email: k.venkata.vk@gmail.com.

[§] University of Texas at Austin. Email: bwaters@cs.utexas.edu. Supported by NSF CNS-1228599 and CNS-1414082, DARPA SafeWare, Microsoft Faculty Fellowship, and Packard Foundation Fellowship.

In particular, we show that VRFs can be constructed generically from the ingredients of: (1) a 1-bounded constrained pseudo random function [BW13, BGI14, KPTZ13] for a functionality that is "admissible hash friendly" , (2) a non-interactive statistically binding commitment scheme (without trusted setup) and (3) a non-interactive witness indistinguishable proofs or NIWIs[FS90]. The first primitive can be replaced with a more basic puncturable PRF [SW14] constraint if one is willing to settle for selective security or assume sub-exponential hardness of assumptions.

The first benefit of our approach is that by generically breaking down the problem we expose and separate the core features of VRFs. Namely, we can see that in spirit any reduction must both develop a way of constraining itself from knowing the output of the entire PRF space while at the same time be able to develop non-interactive proofs without a common setup. Second, with the VRF problem dissected into constituent parts we can explore and develop number theoretic solutions to each piece. Ideally, this breakdown will help us develop a wider array of solutions and in particular break away from the dependence on bilinear maps. We now look at each primitive in turn.

Beginning with constrained PRFs our goal is to build them for constraints that we call admissible hash[BB04] compatible. In particular, we need a constrained key that can be associated with a string $z \in \{0, 1, \perp\}^n$ where the constrained key can be evaluate any input $x \in \{0, 1\}^n$ where $x \neq z$. For our purposes such a scheme only needs to be secure in the model where an attacker is allowed a single key query. The recent work of Brakerski and Vaikuntanthan [BV15] construct 1-bounded constrained PRF under the learning with errors (LWE) [Reg05] assumption that can handle any constraint in $\mathbf{NC}^1$ which encompasses the admissible hash compatible functionality.

We complement this by providing a new construction of constrained PRFs that is admissible hash friendly in the setting of non-bilinear groups. Our construction is proven secure under the $q$-$\mathsf{DDH}$ problem. Informally, one is given $g, g^a, g^{a^2}, \ldots, g^{a^{q-1}}$ it is hard to distinguish $g^{a^q}$ from a random group element. We note that this problem in composite order groups reduces to the subgroup decision problem [CM14]. In addition, as mentioned above if we assume subexponential hardness of our assumptions or relax to selective security we can instead rely on puncturable PRFs which are realizable from any one way function.

We next turn to constructing non-interactive perfectly binding commitments. The main challenge here is any solution must not utilize a trusted setup since a trusted setup is disallowed in the VRF setting. Naor [Nao91] showed how any certifiably injective one way function gives rise to such a commitment scheme. Injective functions can in turn be based on (certifiable) groups where discrete log is hard.

We develop new constructions for non-interactive perfectly binding commitments from noisy cryptographic assumptions. We show and prove a construction under the Learning with Errors and Learning Parity with Noise (LPN) assumptions. Our LPN solution uses a low-noise variant ($\beta \approxeq \frac{1}{\sqrt{n}}$) of the LPN assumption that has been used in previous public key encryption schemes []. We also develop an approach for proving security under LPN with constant noise. Our solution requires the existence of an explicit error correcting code with certain properties. We leave finding such a code as an interesting open problem.

Finally, we arrive at NIWIs. There are three basic approaches to building NIWIs. First, in the bilinear setting it is known [GOS12] how to construct NIWIs from the decision linear assumption. Second, Barak, Ong and Vadhan [BOV07] showed that two-message public-coin witness indistinguishable proofs (a.k.a. ZAPs [DN00]) imply NIWIs under certain complexity theoretic assumptions that allow for derandomization. Finally, indistinguishability obfuscation [GGH$^+$13] gives rise to NIWI constructions [BP15].

Taking a step back we can see that our approach already leads to constructions of VRFs with new properties. For example, if we build ZAPs from trapdoor permutations and apply the BOV theorem we can achieve multiple constructions of adaptively secure VRFs *without* complexity leveraging that do not use bilinear groups. In addition, given the wide array of choices for building our commitments and constrained PRFs our work reveals developing new techniques for building and proving NIWIs as the primary bottleneck for progress towards VRFs.

## 1.1 Technical Overview

We now give a high level overview of our technical approach. A formal treatment is given in the main body. We break our overview into three pieces. First we describe our generic construction of Verifiable Random Functions. Next, we define admissible hash compatible constrained PRFs and go over our non-bilinear group solution. Finally, we overview our LWE and LPN solutions to non-interactive perfectly binding commitments.

**Constructing VRFs Generically** We first briefly review the definition of a Verifiable Random Function. In the VRF framework, a party runs the $\mathsf{Setup}$ algorithm to generate a pair of secret key SK and public verification key VK. Using the secret key SK, it could efficiently evaluate the function $F_{\mathrm{SK}}(\cdot)$ on any input $x$ as well as a proof $\Pi$ of the statement $y = F_{\mathrm{SK}}(x)$. The verification key could be considered as a statistically binding commitment to the underlying pseudorandom function. A third party verification algorithm $\mathsf{Verify}$ is used to verify a proof $\Pi$ which takes the verification key VK, function evaluation $y$, and message $x$ as additional inputs. First, the soundness condition dictates that for each $(\mathrm{VK}, x)$ pair there should be at most one output $y$ such that $\mathsf{Verify}(\mathrm{VK}, x, y, \pi) = 1$. Importantly, VRFs do not make use of any setup assumption and soundness should hold even in the case of a *maliciously generated* VK. Second, it should also hold that the output of function $F_{\mathrm{SK}}(\cdot)$ is indistinguishable from a random string even after observing polynomially many evaluations and proofs at adversarially chosen points. The latter is formalized as pseudorandomness property of the VRF.

We now give a simple construction from the aforementioned primitives. The VRF setup proceeds as follows. First, a constrained PRF key $K$ is sampled and kept as part of the secret key. Next, a sequence of three independent commitments $c_1, c_2, c_3$ is computed such that each commitment $c_i$ opens to the key $K$.[1] The triple of commitments $(c_1, c_2, c_3)$ is stored as the public verification key and the corresponding randomness used during commitment is included in the secret key. For evaluating the VRF on any input $x$, we first apply an admissible hash function on $x$ and then evaluate the constrained PRF on the output of admissible hash. In short, the VRF output on some input $x$ is $\mathsf{PRF}_K(h(x))$. For proving correctness of evaluation, we use non-interactive witness indistinguishable proofs (NIWIs). In particular, to prove that the output of VRF on some input $x$ is $y$, we create a NIWI proof for the statement that *at least two out of three* commitments $(c_1, c_2, c_3)$ (in the verification key) open to keys $K_1, K_2$ such that $y = \mathsf{PRF}_{K_1}(h(x)) = \mathsf{PRF}_{K_2}(h(x))$. We would like to emphasize that keys $K_1$ and $K_2$ need not be identical as the only constraint that must hold is that the PRF evaluation of input $h(x)$ must be equal to $y$ irrespective of the key (out of $K_1, K_2$) used. The proof verification can be done in a straightforward manner as it simply involves running the NIWI verifier.

Now we briefly sketch the idea behind pseudorandomness proof in the adaptive setting. To prove security we use a "partitioning" argument where roughly $1/Q$ fraction of inputs can be used as challenge and remaining $1 - 1/Q$ fraction will be used for answering evaluation queries, where $Q$ is the number of queries made by an attacker. First step in the reduction is to concretely define the challenge and non-challenge partitions using admissible hash function. Next, we leverage the facts that all the evaluation queries will lie outside the challenge partition[2] and for generating the evaluation proofs we only need openings of two key commitments out of three. At a high level, our goal is to switch all three commitments $c_1, c_2, c_3$ such that they commit to the constrained key $K'$ instead of key $K$, where $K'$ could be used to evaluate the VRF on all points outside the challenge partition. To this end, the reduction proceeds as follows.

First, the challenger makes two crucial modifications — (1) it generates a constrained PRF key $K'$ along with the master key $K$, (2) it computes $c_3$ as a commitment to key $K'$ instead of key $K$. Such a hybrid jump is indistinguishable by the hiding property of the commitment scheme as for generating all the evaluation proofs it does not need the opening for $c_3$. Next, we switch the NIWI witness used to generate the proof. In particular, the challenger now uses openings of $c_2, c_3$ as the NIWI witnesses. This only results in a negligible

[1] Looking ahead, it will be crucial for proving the unique provability property that the commitment scheme used is perfectly binding.

[2] The challenger needs to perform an abort step in case of bad partitioning, however for the above informal exposition we avoid discussing it. More details are provided in Section 4.

dip in the adversary's advantage because for all inputs outside the challenge partition, the PRF evaluation using the master key $K$ and constrained key $K'$ is identical, thus the openings of any two commitments out of $c_1, c_2, c_3$ could be used as the NIWI witness. Applying similar modifications as above in succession, all three commitments $c_1, c_2, c_3$ could be switched to commitments of the constrained key $K'$. If all three commitments open to the constrained key $K'$, then the challenger could directly reduce an attack on the VRF pseudorandomness to an attack on the constrained pseudorandomness of the PRF.

It is also interesting to note that if we use a puncturable PRF instead of an admissible hash compatible constrained PRF, then the same construction could be proven to be selectively secure with only polynomial security loss to the underlying assumptions. The major difference in the proof being the partitioning step, where instead of using the admissible hash function to perform partitioning and aborting in case of bad partitions, the reduction already knows the challenge input at the start, thus it only needs to puncture the PRF key on challenge input in order to use the same sequence of hybrids. This is discussed in detail in Section 4.

**Admissible Hash Compatible Constrained PRFs** A constrained PRF family consists of a setup algorithm that outputs the master PRF key, a constrain algorithm that takes as input the master PRF key and a constraint, and outputs a constrained PRF key. The constrained PRF key can be used to evaluate the PRF at all points satisfied by the constraint. As mentioned in the previous paragraph, for constructing adaptively secure VRFs, we require constrained PRFs for a special class of 'admissible hash compatible' constraints. Each constraint is specified by a string $u \in \{0, 1, \perp\}^n$. Given a constrained key for $u$, it can be used to evaluate the PRF at all points $x$ such that there exists an index $i \leq n$ where $u_i \neq \perp$ and $x_i \neq u_i$. For this work, we require a weaker notion of security which we call 'single-key no-query' security. Here, the adverasary first sends a constrained key query $u$. After receiving the constrained key, it sends a challenge point $x$ such that it does not satisfy the constraint (that is, for all $i \leq n$, either $u_i = \perp$, or $x_i = u_i$). It then receives either the PRF evaluation at $x$ or a uniformly random string, and it must distinguish between the two scenarios.

Our construction, at a high level, is similar to the Naor-Reingold PRF construction. The PRF key consists of $2n$ integers $\{c_{i,b}\}_{i \leq n, b \in \{0,1\}}$ and a group element $g$. To evaluate at a point $x$, we first choose $n$ out of the $2n$ integers, depending on the bits of $x$. Let $t$ denote the product of these $n$ integers. The PRF evaluation is $g^t$. A constrained key for constraint $u \in \{0, 1, \perp\}^n$ consists of $n$ powers of a random integer $a$ in the exponent of $g$: $(g, g^a, \ldots, g^{a^{n-1}})$ and $2n$ integers $\{v_{i,b}\}$. Each $v_{i,b}$ is set to be either $c_{i,b}$ or $c_{i,b}/a$, depending on $u_i$. Using the $v_{i,b}$ and an appropriate $g^{a^k}$ term, one can compute the PRF evaluation at any point $x$ such that it satisfies the constraint (that is, if there exists an $i \leq n$ such that $u_i \neq \perp$ and $x_i \neq u_i$). However, if $x$ does not satisfy the constraint, then one needs to compute $g^{a^n}$ to compute the PRF evaluation at $x$. Using the $n$-$\mathsf{DDH}$ assumption, we can argue that if an adversary can distinguish between the PRF evaluation and a truly random string, then one can use this adversary to distinguish between $g^{a^n}$ and a random group element.

**New Constructions of Non-Interactive Perfectly Binding Commitments** Finally, the third component required for our VRF construction is a non-interactive perfectly binding commitment scheme (without trusted setup). In this work, we give new constructions for this primitive based on the Learning with Errors (LWE) and Learning Parity with Noise (LPN) assumptions. Our LPN construction can be proven secure under the LPN with low noise assumption. Finally, we also give an approach for proving security under LPN with constant noise. This approach relies on the existence of special error correcting codes with 'robust' generator matrix. Currently, we do not have any explicit constructions for this combinatorial object. For simplicity, we only consider single bit commitment schemes.

*LWE Construction:* In this scheme, we will be working in $\mathbb{Z}_q$ for a suitably large prime $q$. The commitment to a bit $b$ consists of a randomly chosen vector $\mathbf{w}$ and $\mathbf{w}^T\mathbf{s} + \mathsf{noise} + b(q/2)$, where $\mathbf{s}$ is a randomly chosen secret vector. However, to ensure perfect binding, we need to have some additional components. The scheme also chooses a random matrix $\mathbf{B}$ from a distribution $\mathcal{D}_1$ and outputs $\mathbf{B}, \mathbf{B}^T\mathbf{s} + \mathsf{noise}$. This distribution has the special property that all matrices from this distribution have 'medium norm' rowspace. This property ensures that there does not exist two distinct vectors $\mathbf{s_1}$ and $\mathbf{s_2}$ such that $\mathbf{B}^T\mathbf{s_1} + \mathsf{noise}_1 = \mathbf{B}^T\mathbf{s_2} + \mathsf{noise}_2$. Finally, to argue computational hiding, we require that a random matrix from this distribution looks uniformly random. If this condition is satisfied, then we can use the LWE assumption to argue that $\mathbf{w}^T\mathbf{s} + \mathsf{noise}$ and $\mathbf{B}^T\mathbf{s} + \mathsf{noise}'$ look uniformly random, thereby hiding the committed bit. Sampling a matrix from the distribution $\mathcal{D}_1$ works as follows: first choose a uniformly random matrix $\mathbf{A}$, then choose a matrix $\mathbf{C}$ with low norm entries, matrix $\mathbf{D}$ with 'medium' entries and output $[\mathbf{A} \mid \mathbf{AC} + \mathbf{D} + \mathsf{noise}]$. For any non zero vector $\mathbf{s}$, if $\mathbf{A}^T\mathbf{s}$ has low norm, then $\mathbf{C}^T\mathbf{As}$ also has low norm, but $\mathbf{D}^T\mathbf{s}$ has medium norm entries, and therefore $[\mathbf{A} \mid \mathbf{AC} + \mathbf{D} + \mathsf{noise}]^T\mathbf{s}$ has medium norm entries.

*Low Noise LPN construction:* This scheme is similar to the LWE construction. Here also, the commitment to a bit $b$ consists of $\mathbf{w}$ and $\mathbf{w}^T\mathbf{s} + b$, where $\mathbf{w}$ and $\mathbf{s}$ are uniformly random vectors in $\mathbb{Z}_2^n$. To ensure that there can be only one vector $\mathbf{s}$, we also choose a matrix $\mathbf{B}$ from a special distribution $\mathcal{D}_2$ and output $\mathbf{B}, \mathbf{B}^T\mathbf{s} + \mathsf{noise}$. In this case, the distribution $\mathcal{D}_2$ is such that all matrices from this distribution have high hamming weight rowspace. To sample from the distribution $\mathcal{D}_2$, one chooses a uniformly random matrix $\mathbf{A}$, a matrix $\mathbf{C}$ with low hamming weight rows and outputs $[\mathbf{A} \mid \mathbf{AC} + \mathbf{G}]$, where $\mathbf{G}$ is the generator matrix of an error correcting code. Here the role of $\mathbf{G}$ is similar to the role of $\mathbf{D}$ in the previous solution: to map any non-zero vector to a high hamming weight vector. An important point here is that we need the rows of $\mathbf{C}$ to have low $(\mathrm{O}(\sqrt{(}n)))$ hamming weight. This is because we want to argue that if $\mathbf{A}^T\mathbf{s}$ has low hamming weight, then so does $\mathbf{C}^T\mathbf{A}^T\mathbf{s}$. Finally, to argue that $\mathcal{D}_2$ looks like the uniform distribution, we need the LPN assumption with low noise [3](since $\mathbf{C}$ has low $(O(\sqrt{n}))$ hamming weight rows).

This construction bears some similarities to the CCA secure encryption scheme of Kiltz et al. [**?**].

*Standard LPN construction:* Finally, we describe an approach for constructing a commitment scheme that can be proven secure under the standard LPN assumption (with constant noise). For this approach, we require a deterministic procedure that can output $l$ matrices $\mathbf{G}_1, \ldots, \mathbf{G}_l$ with the following property: for any matrix $\mathbf{A}$, there exists an index $i$ such that the rowspace of $\mathbf{A} + \mathbf{G}_i$ has high hamming weight. Given such a procedure, our commitment scheme works as follows. The commitment algorithm, on input message $b$, chooses a uniformly random matrix $\mathbf{A}$ and generates $l$ sub-commitments. The $i^{th}$ sub-commitment chooses uniformly random vectors $\mathbf{s_i}, \mathbf{w_i}$ and outputs $(\mathbf{A} + \mathbf{G}_i)^T\mathbf{s_i} + \mathsf{noise}$, $\mathbf{w_i}$ and $\mathbf{w_i}^T\mathbf{s_i} + b$. For perfect binding, we will use the guarantee that there exists an $i$ such that the rowspace of $\mathbf{A} + \mathbf{G}_i$ has high hamming weight. This implies that if $(\mathbf{A} + \mathbf{G}_i)^T\mathbf{s_1} + \mathsf{noise} = (\mathbf{A} + \mathbf{G}_i)^T\mathbf{s_2} + \mathsf{noise}$, then $\mathbf{s_1} = \mathbf{s_2}$. For computational hiding, we need a hybrid argument to switch each sub-commitment to uniformly random.

# 2 Preliminaries

## 2.1 Verifiable Random Functions

Verifiable random functions (VRFs) were introduced by Micali, Rabin and Vadhan [MRV99]. VRFs are keyed functions with input domain $\{\mathcal{X}_\lambda\}_\lambda$, output range $\{\mathcal{Y}_\lambda\}_\lambda$ and consist of three polynomial time algorithms $\mathsf{Setup}$, $\mathsf{Evaluate}$ and $\mathsf{Verify}$ described as follows:

- $\mathsf{Setup}(1^\lambda)$ is a randomized algorithm that on input the security parameter, outputs $(\mathrm{SK}, \mathrm{VK})$. SK is called secret key, and VK verification key.
- $\mathsf{Evaluate}(\mathrm{SK}, x)$ is a (possibly randomized) algorithm, and on input the secret key SK and $x \in \mathcal{X}_\lambda$, it outputs an evaluation $y \in \mathcal{Y}_\lambda$ and a proof $\pi \in \{0,1\}^*$.
- $\mathsf{Verify}(\mathrm{VK}, x, y, \pi)$ is a (possibly randomized) algorithm which uses verification key VK and proof $\pi$ to verify that $y$ is the correct evaluation on input $x$. It outputs 1 (accepts) if verification succeeds, and 0 (rejects) otherwise.

[3] We will be using the (low noise) Knapsack LPN assumption. The Knapsack LPN assumption states that for a uniformly random matrix $\mathbf{A}$ and a matrix $\mathbf{E}$ such that each entry is 1 with probability $p$ and $\mathbf{A}$ has fewer rows than columns, then $(\mathbf{A}, \mathbf{AE})$ look like uniformly random matrices.

**Definition 2.1.** (Adaptively-secure VRF) A pair of polynomial time algorithms $(\mathsf{Setup}, \mathsf{Evaluate}, \mathsf{Verify})$ is an adaptively-secure verifiable random function if it satisfies the following conditions:

- (Correctness) For all $(\mathrm{SK}, \mathrm{VK}) \leftarrow \mathsf{Setup}(1^\lambda)$, and all $x \in \mathcal{X}_\lambda$, if $(y, \pi) \leftarrow \mathsf{Evaluate}(\mathrm{SK}, x)$, then $\Pr[\mathsf{Verify}(\mathrm{VK}, x, y, \pi) = 1] = 1$.
- (Unique Provability) For every $(\mathrm{VK}, x, y_1, \pi_1, y_2, \pi_2)$ such that $y_1 \neq y_2$, the following holds for at least one $i \in \{1, 2\}$:
$$\Pr[\mathsf{Verify}(\mathrm{VK}, x, y_1, \pi_1) = 1] \leq 2^{-\Omega(\lambda)}.$$
- (Pseudorandomness) For any PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ there exists a negligible function $\mathrm{negl}(\cdot)$, such that for all $\lambda \in \mathbb{N}$, $\mathsf{Adv}_{\mathcal{A}}^{\text{adp-VRF}}(\lambda) \leq \mathrm{negl}(\lambda)$, where advantage of $\mathcal{A}$ is defined as
$$\mathsf{Adv}_{\mathcal{A}}^{\text{adp-VRF}}(\lambda) = \left| \Pr\left[ \mathcal{A}_1^{\mathcal{O}_{x^*}}(\mathsf{st}, y_b) \;= b : \begin{array}{c} (\mathrm{SK}, \mathrm{VK}) \leftarrow \mathsf{Setup}(1^\lambda); \quad b \leftarrow \{0,1\} \\ (x^*, \mathsf{st}) = \mathcal{A}_0^{\mathsf{Evaluate}(\mathrm{SK}, \cdot)}(1^\lambda, \mathrm{VK}) \\ (y_1, \pi) \leftarrow \mathsf{Evaluate}(\mathrm{SK}, x^*); \quad y_0 \leftarrow \mathcal{Y}_\lambda \end{array} \right] - \frac{1}{2} \right|,$$
where oracle $\mathcal{O}_{x^*}$ on input $x^*$ outputs $\perp$, otherwise behaves same as $\mathsf{Evaluate}(\mathrm{SK}, \cdot)$.

A weaker notion of security for VRFs is selective pseudorandomness where the adversary must commit to its challenge $x^*$ at the start of the game, that is before the challenger sends VK to $\mathcal{A}$. Then during evaluation phase, $\mathcal{A}$ is allowed to query on polynomially many messages $x \neq x^*$, and $\mathcal{A}$ wins if its guess $b' = b$. The advantage of $\mathcal{A}$ is defined to be $\mathsf{Adv}_{\mathcal{A}}^{\text{sel-VRF}}(\lambda) = |\Pr[\mathcal{A} \text{ wins}] - 1/2|$.

**Definition 2.2.** (Selectively-secure VRF) A pair of polynomial time algorithms $(\mathsf{Setup}, \mathsf{Evaluate}, \mathsf{Verify})$ is called a selectively-secure verifiable random function if it satisfies correctness and unique provability properties (as in Definition 2.1), and for all PPT adversaries $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\text{sel-VRF}}(\lambda)$ is negligible in the security parameter $\lambda$.

## 2.2 Non-Interactive Witness Indistinguishable Proofs

Witness indistinguishable (WI) proofs were introduced by Fiat and Shamir [FS90] as a natural weakening of zero-knowledge (ZK) proofs. At a high level, the witness indistinguishability property says that a proof must not reveal the witness used to prove the underlying statement even if it reveals all possible witnesses corresponding to the statement. Unlike ZK proofs, WI proofs without interaction in the standard model are known to be possible. Barak, Ong and Vadhan [BOV07] provided constructions for one-message (completely non-interactive, with no shared random string or setup assumptions) witness indistinguishable proofs (NIWIs) based on ZAPs (i.e., two-message public-coin witness indistinguishable proofs) and Nisan-Wigderson type pseudorandom generators []. Groth, Ostrovsky and Sahai [GOS12] gave the first NIWI construction from a standard cryptographic assumption, namely the decision linear assumption. Recently, Bitansky and Paneth [BP15] constructed NIWI proofs assuming iO and one-way permutations.

**Definition 2.3.** (NIWI) A pair of PPT algorithms $(\mathcal{P}, \mathcal{V})$ is a NIWI for a language $\mathcal{L} \in \mathbf{NP}$ with witness relation $\mathcal{R}$ if it satisfies the following conditions:

- (Perfect Completeness) For all $(x, w)$ such that $\mathcal{R}(x, w) = 1$,
$$\Pr[\mathcal{V}(x, \pi) = 1 : \; \pi \leftarrow \mathcal{P}(x, w)] = 1.$$
- (Statistical Soundness) For every $x \notin \mathcal{L}$ and $\pi \in \{0,1\}^*$,
$$\Pr[\mathcal{V}(x, \pi) = 1] \leq 2^{-\Omega(|x|)}.$$
- (Witness Indistinguishability) For any sequence $\mathcal{I} = \{(x, w_1, w_2) : \mathcal{R}(x, w_1) = 1 \wedge \mathcal{R}(x, w_2) = 1\}$
$$\{\pi_1 : \pi_1 \leftarrow \mathcal{P}(x, w_1)\}_{(x, w_1, w_2) \in \mathcal{I}} \approx_c \{\pi_2 : \pi_2 \leftarrow \mathcal{P}(x, w_2)\}_{(x, w_1, w_2) \in \mathcal{I}}$$

## 2.3 Perfectly Binding Commitments (with no setup assumptions)

A commitment scheme with message space $\{\mathcal{M}_\lambda\}_\lambda$, randomness space $\{\mathcal{R}_\lambda\}_\lambda$ and commitment space $\{\mathcal{C}_\lambda\}_\lambda$ consists of two polynomial time algorithms — Commit and Verify with the following syntax.

- $\mathsf{Commit}(1^\lambda, m \in \mathcal{M}_\lambda; r \in \mathcal{R}_\lambda)$: The commit algorithm is a randomized algorithm that takes as input the security parameter $\lambda$, message $m$ to be committed and random coins $r$. It outputs a commitment $c$.

- $\mathsf{Verify}(m \in \mathcal{M}_\lambda, c \in \mathcal{C}_\lambda, o \in \mathcal{R}_\lambda)$: The verification algorithm takes as input the message $m$, commitment $c$ and an opening $o$. It outputs either 0 or 1.

For simplicity, we assume that the opening for a commitment is simply the randomness used during the commitment phase. As a result, we do not have a separate 'reveal' algorithm. Below we formally define perfectly binding computationally hiding (PB-CH) commitment schemes with no setup assumptions (i.e., without trusted setup and CRS).

**Definition 2.4.** (PB-CH Commitments) A pair of polynomial time algorithms $(\mathsf{Commit}, \mathsf{Verify})$ is a perfectly binding computationally hiding (PB-CH) commitment scheme if it satisfies the following conditions:

- (Perfect Correctness) For all security parameters $\lambda \in \mathbb{N}$, message $m \in \mathcal{M}_\lambda$ and randomness $r \in \mathcal{R}_\lambda$, if $c = \mathsf{Commit}(1^\lambda, m; r)$, then $\mathsf{Verify}(m, c, r) = 1$.

- (Perfect Binding) For every $(c, m_1, r_1, m_2, r_2)$ such that $m_1 \neq m_2$, the following holds for at least one $i \in \{1, 2\}$:
$$\Pr[\mathsf{Verify}(m_i, c, r_i) = 1] = 0.$$

- (Computationally Hiding) For all security parameters $\lambda \in \mathbb{N}$, messages $m_1, m_2 \in \mathcal{M}_\lambda$,
$$\left\{c_1 : c_1 \leftarrow \mathsf{Commit}(1^\lambda, m_1; r_1); r_1 \leftarrow \mathcal{R}_\lambda\right\} \approx_c \left\{c_2 : c_2 \leftarrow \mathsf{Commit}(1^\lambda, m_2; r_2); r_2 \leftarrow \mathcal{R}_\lambda\right\}$$

Perfectly binding commitments (without trusted setup) can be constructed from *certifiably injective* one-way functions. In this work, we show how to construct them under the Learning Parity with Low Noise assumption [] and Learning with Errors assumption [Reg05]. We would like to point out that the 'no trusted setup' requirement for commitments is essential for our VRF construction. We already know how to construct perfectly binding commitments *with trusted setup* from the LPN assumption [JKPT12], however it is not sufficient for our VRF construction as VRFs disallow trusted setup.

## 2.4 Admissible Hash Functions

A commonly used technique for achieving adaptive security is the *partitioning strategy* where the input space is partitioned into a 'query partition' and a 'challenge partition'. This partitioning is achieved using *admissible hash functions* introduced by Boneh and Boyen [BB04]. Here we state a simplified definition from [HSW14].

**Definition 2.5.** Let $k, \ell$ and $\theta$ be efficiently computable univariate polynomials. Let $h : \{0,1\}^{k(\lambda)} \to \{0,1\}^{\ell(\lambda)}$ be an efficiently computable function and $\mathsf{AdmSample}$ a PPT algorithm that takes as input $1^\lambda$ and an integer $Q$, and outputs $u \in \{0, 1, \perp\}^{\ell(\lambda)}$. For any $u \in \{0, 1, \perp\}^{\ell(\lambda)}$, define $P_u : \{0,1\}^{k(\lambda)} \to \{0, 1\}$ as follows:
$$P_u(x) = \begin{cases} 1 & \text{if for } j \leq \ell(\lambda), \quad u_j = h(x)_j \vee u_j = \perp \\ 0 & \text{otherwise.} \end{cases}$$

We say that $(h, \mathsf{AdmSample})$ is $\theta$-admissible if the following condition holds:

For any efficiently computable polynomial $Q$, for all $x_1, \ldots, x_{Q(\lambda)}, x^* \in \{0,1\}^{k(\lambda)}$, where $x^* \notin \{x_i\}_1^{Q(\lambda)}$,

$$\Pr[(\forall i \leq Q(\lambda), P_u(x_i) = 0) \wedge P_u(x^*) = 1] \geq \frac{1}{\theta(Q(\lambda))}$$

where the probability is taken over $u \leftarrow \mathsf{AdmSample}(1^\lambda, Q(\lambda))$.

**Theorem 2.1** (Admissible Hash Function Family [BB04], simplified proof in [FHPS13])**.** For any efficiently computable polynomial $k$, there exist efficiently computable polynomials $\ell, \theta$ such that there exist $\theta$-admissible function families mapping $k$ bits to $\ell$ bits.

Note that the above theorem is information theoretic, and is not based on any cryptographic assumption.

## 2.5 Constrained Pseudorandom Functions for Admissible Hash Compatible Functions

Constrained pseudorandom functions, introduced by [BW13, BGI14, KPTZ13], are an extension of pseudorandom functions [GGM84] where a party having the master PRF key can compute keys corresponding to any constraint from a constraint class. A constrained key for constraint $C$ can be used to evaluate the PRF on inputs $x$ that satisfy the constraint $C(x) = 0$.[4] However, the constrained key should not reveal PRF evaluations at points not satisfied by the constraint. Constrained PRFs for general circuit constraints can be constructed using multilinear maps [BW13], indistinguishability obfuscation [BZ14] and the learning with errors assumption [BV15]. Note that the construction from LWE only allows a single constrained key query, which is a weaker security definition than the standard fully 'collusion-resistant' notion.

In this work, we will be using a special constraint family which we call 'admissible hash compatible', and the security definition will also be weaker than the standard (fully collusion-resistant) security for constrained PRFs. This enables us to construct this primitive from weaker and standard cryptographic assumptions such as the $q$-DDH assumption [].

**Definition 2.6.** Let $\mathcal{Z}_n = \{0,1,\perp\}^n$. An admissible hash compatible function family $\mathcal{P}_n = \{P_z : \{0,1\}^n \rightarrow \{0,1\} \mid z \in \mathcal{Z}_n\}$ is defined exactly as the predicate $P_u(\cdot)$ in Definition 2.5.

Looking ahead the above admissible hash compatible function family will correspond to the family of constraints for which we assume constrained PRFs. Next, we formally define the syntax, correctness and security properties of constrained PRFs.

**Syntax.** Let $n(\cdot)$ be a polynomial. A constrained pseudorandom function $\mathsf{CPRF}$ with domain $\{\mathcal{X}_\lambda = \{0,1\}^{n(\lambda)}\}_\lambda$, range $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_\lambda$, key space $\mathcal{K} = \{\mathcal{K}_\lambda\}_\lambda$ and constrained key space $\mathcal{K}^c = \{\mathcal{K}^c_\lambda\}_\lambda$ for a family of admissible hash compatible constraints $\{\mathcal{C}_\lambda = \mathcal{P}_{n(\lambda)}\}_\lambda$ consists of three algorithms $\mathsf{Setup}, \mathsf{Constrain}, \mathsf{Evaluate}$ defined as follows. For simplicity of notation, we will refer to $z$ as the constraint instead of $P_z$.

- $\mathsf{Setup}(1^\lambda)$: The setup algorithm takes as input the security parameter $\lambda$ and outputs a PRF key $K \in \mathcal{K}_\lambda$.
- $\mathsf{Constrain}(K, z \in \{0,1,\perp\}^{n(\lambda)})$: The constrain algorithm takes as input a master PRF key $K \in \mathcal{K}_\lambda$, a constraint $z \in \{0,1,\perp\}^{n(\lambda)}$ and outputs a constrained key $K_z \in \mathcal{K}^c_\lambda$.
- $\mathsf{Evaluate}(K \in \mathcal{K}_\lambda \cup \mathcal{K}^c_\lambda, x \in \{0,1\}^{n(\lambda)})$: The evaluation algorithm takes as input a PRF key $K$ (master or constrained), and outputs $y \in \mathcal{Y}$.

[4] We would like to point out that our notation departs from what has been used in the literature. Traditionally, it is considered that the constrained key allows PRF evaluation on points that satisfy $C(x) = 1$. However, we switch the constraint to $C(x) = 0$ for convenience.

**Correctness** For every security parameter $\lambda$, master PRF key $K \leftarrow \mathsf{Setup}(1^\lambda)$, constraint $z \in \{0,1,\perp\}^{n(\lambda)}$, $K_z \leftarrow \mathsf{Constrain}(K, z)$ and $x \in \{0,1\}^{n(\lambda)}$ such that $P_z(x) = 0$, $\mathsf{Evaluate}(K, x) = \mathsf{Evaluate}(K_z, x)$.

We would like to point out that in the above description there is a common evaluation algorithm that accepts both the PRF master key as well as the constrained key. Such an abstraction helps us in simplifying our VRF construction later in Section 4. Note that this is not a restriction on the constrained PRFs as it can achieved without loss of generality from any constrained PRF. Below we define the *single-key no-query* constrained pseudorandomness security notion for constrained PRFs.

**Definition 2.7.** A pair of polynomial time algorithms $(\mathsf{Setup}, \mathsf{Constrain}, \mathsf{Evaluate})$ is a *single-key no-query* secure constrained pseudorandom function for admissible hash compatible constraint family if it satisfies the following conditions:

- (Correctness) For every security parameter $\lambda \in \mathbb{N}$, master PRF key $K \leftarrow \mathsf{Setup}(1^\lambda)$, constraint $z \in \{0,1,\perp\}^{n(\lambda)}$, constrained key $K_z \leftarrow \mathsf{Constrain}(K, z)$ and input $x \in \{0,1\}^{n(\lambda)}$ such that $P_z(x) = 0$, $\mathsf{Evaluate}(K, x) = \mathsf{Evaluate}(K_z, x)$.
- (Single-key No-query Constrained Pseudorandomness) For any PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ there exists a negligible function $\text{negl}(\cdot)$, such that for all $\lambda \in \mathbb{N}$, $\mathsf{Adv}_{\mathcal{A}}^{\text{CPRF}}(\lambda) \leq \text{negl}(\lambda)$, where advantage of $\mathcal{A}$ is defined as

$$\mathsf{Adv}_{\mathcal{A}}^{\text{CPRF}}(\lambda) = \left| \Pr \left[ \mathcal{A}_2(\widetilde{\mathsf{st}}, y_b) \ = b : \begin{array}{c} K \leftarrow \mathsf{Setup}(1^\lambda); \quad (z, \mathsf{st}) = \mathcal{A}_0(1^\lambda) \\ K_z \leftarrow \mathsf{Constrain}(K, z) \\ (x, \widetilde{\mathsf{st}}) = \mathcal{A}_1(\mathsf{st}, K_z); \quad b \leftarrow \{0,1\} \\ y_1 \leftarrow \mathsf{Evaluate}(K, x); \quad y_0 \leftarrow \mathcal{Y}_\lambda \end{array} \right] - \frac{1}{2} \right|.$$

  Also, the challenge point $x$ chosen by $\mathcal{A}$ must satisfy the constraint $P_z(x) = 1$, i.e. it should not be possible to evaluate the PRF on $x$ using constrained key $K_z$.

Note that the above security notion is weaker than the standard fully collusion-resistant security notion, since the adversary gets one constrained key, and then it must distinguish between a random string and the PRF evaluation at a point not satisfying the constraint. This is weaker than the standard security definition in two ways. First, there is only one constrained key query, and second, there are no evaluation queries. However, as we will see in Section 4, this suffices for our construction. Looking ahead, the high level idea is that we will partition the VRF input space using an admissible hash function, and to answer each evaluation query we only need a constrained key since a constrained key lets us evaluate at all points in the query partition.

**Remark 2.1.** Additionally, we want that there exists a polynomial $s(\cdot)$ such that $\forall \lambda \in \mathbb{N}$, $K \in \mathcal{K}_\lambda \cup \mathcal{K}_\lambda^c$, $|K| \leq s(\lambda)$, i.e. size of each PRF key is polynomially bounded.

**Definition 2.8.** A puncturing constraint function family $\mathcal{P}_n = \{P_z : \{0,1\}^n \to \{0,1\} \mid z \in \{0,1\}^n\}$ is defined exactly as the predicate $P_u(\cdot)$ in Definition 2.5.

**Definition 2.9.** A set of polynomial time algorithms $(\mathsf{Setup}, \mathsf{Puncture}, \mathsf{Evaluate})$ is a secure puncturable pseudorandom function if it is a *single-key no-query* secure constrained pseudorandom function (Definition 2.7) for puncturing constraint function family.

## 2.6 Lattice Preliminaries

Given positive integers $n, m, q$ and a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, we let $\Lambda_q^\perp(\mathbf{A})$ denote the lattice $\{\mathbf{x} \in \mathbb{Z}^m \ : \ \mathbf{A} \cdot \mathbf{x} = \mathbf{0} \mod q\}$. For $\mathbf{u} \in \mathbb{Z}_q^n$, we let $\Lambda_q^{\mathbf{u}}(\mathbf{A})$ denote the coset $\{\mathbf{x} \in \mathbb{Z}^m \ : \ \mathbf{A} \cdot \mathbf{x} = \mathbf{u} \mod q\}$.

**Discrete Gaussians.** Let $\sigma$ be any positive real number. The Gaussian distribution $\mathcal{D}_\sigma$ with parameter $\sigma$ is defined by the probability distribution function $\rho_\sigma(\mathbf{x}) = \exp(-\pi \cdot ||\mathbf{x}||^2/\sigma^2)$. For any set $\mathcal{L} \subset \mathcal{R}^m$, define $\rho_\sigma(\mathcal{L}) = \sum_{\mathbf{x}\in\mathcal{L}} \rho_\sigma(\mathbf{x})$. The discrete Gaussian distribution $\mathcal{D}_{\mathcal{L},\sigma}$ over $\mathcal{L}$ with parameter $\sigma$ is defined by the probability distribution function $\rho_{\mathcal{L},\sigma}(\mathbf{x}) = \rho_\sigma(\mathbf{x})/\rho_\sigma(\mathcal{L})$ for all $\mathbf{x} \in \mathcal{L}$.

The following lemma (Lemma 4.4 of [MR07], [GPV08]) shows that if the parameter $\sigma$ of a discrete Gaussian distribution is small, then any vector drawn from this distribution will be short (with high probability).

**Lemma 2.1.** *Let $m, n, q$ be positive integers with $m > n$, $q \geq 2$. Let $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ be a matrix of dimensions $n \times m$, and $\mathcal{L} = \Lambda_q^\perp(\mathbf{A})$. Then*

$$\Pr[||\mathbf{x}|| > \sqrt{m} \cdot \sigma : \mathbf{x} \leftarrow \mathcal{D}_{\mathcal{L},\sigma}] \leq \text{negl}(n).$$

# 3 Cryptographic Assumptions

## 3.1 Learning with Noise (LWE)

The Learning with Errors (LWE) problem was introduced by Regev [Reg05]. The LWE problem has four parameters: the dimension of the lattice $n$, the number of samples $m$, the modulus $q$ and the error distribution $\chi(n)$.

**Assumption 1** (Learning with Errors)**.** *Let $n$, $m$ and $q$ be positive integers and $\chi$ a noise distribution on $\mathbb{Z}$. The Learning with Errors assumption $(n, m, q, \chi)$-$\mathsf{LWE}$, parameterized by $n, m, q, \chi$, states that the following distributions are computationally indistinguishable:*

$$\left\{ (\mathbf{A}, \mathbf{A}^T\mathbf{s} + \mathbf{e}) \;:\; \begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \\ \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \chi^m \end{array} \right\} \approx_c \left\{ (\mathbf{A}, \mathbf{u}) \;:\; \begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \\ \mathbf{u} \leftarrow \mathbb{Z}_q^m \end{array} \right\}$$

Under a quantum reduction, Regev [Reg05] showed that for certain noise distributions, LWE is as hard as worst case lattice problems such as the decisional approximate shortest vector problem ($\mathsf{GapSVP}$) and approximate shortest independent vectors problem ($\mathsf{SIVP}$). The following theorem statement is from Peikert's survey [Pei15].

**Theorem 3.1** ([Reg05])**.** *For any $m \leq \mathsf{poly(n)}$, any $q \leq 2^{\mathsf{poly(n)}}$, and any discretized Gaussian error distribution $\chi$ of parameter $\alpha \cdot q \geq 2 \cdot \sqrt{n}$, solving $(n, m, q, \chi)$-$\mathsf{LWE}$ is as hard as quantumly solving $\mathsf{GapSVP}_\gamma$ and $\mathsf{SIVP}_\gamma$ on arbitrary $n$-dimensional lattices, for some $\gamma = \tilde{O}(n/\alpha)$.*

Later works [Pei09, BLP<sup>+</sup>13] showed classical reductions from LWE to $\mathsf{GapSVP}_\gamma$. Given the current state of art in lattice algorithms, $\mathsf{GapSVP}_\gamma$ and $\mathsf{SIVP}_\gamma$ are believed to be hard for $\gamma = \tilde{O}(2^{n^\epsilon})$, and therefore $(n, m, q, \chi)$-$\mathsf{LWE}$ is believed to be hard for Gaussian error distributions $\chi$ with parameter $2^{-n^\epsilon} \cdot q \cdot \mathsf{poly(n)}$.

**LWE with Short Secrets.** In this work, we will be using a variant of the LWE problem called *LWE with Short Secrets*. In this variant, introduced by Applebaum et al. [ACPS09], the secret vector is also chosen from the noise distribution $\chi$. They showed that this variant is as hard as $\mathsf{LWE}$ for sufficiently large number of samples $m$.

**Assumption 2** (LWE with Short Secrets)**.** *Let $n$, $m$ and $q$ be positive integers and $\chi$ a noise distribution on $\mathbb{Z}$. The LWE with Short Secrets assumption $(n, m, q, \chi)$-$\mathsf{LWE\text{-}ss}$, parameterized by $n, m, q, \chi$, states that the following distributions are computationally indistinguishable* [5]:

$$\left\{ (\mathbf{A}, \mathbf{SA} + \mathbf{E}) \;:\; \begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \\ \mathbf{S} \leftarrow \chi^{n\times n}, \mathbf{E} \leftarrow \chi^{n\times m} \end{array} \right\} \approx_c \left\{ (\mathbf{A}, \mathbf{U}) \;:\; \begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \\ \mathbf{U} \leftarrow \mathbb{Z}_q^{n\times m} \end{array} \right\}.$$

[5] Applebaum et al. showed that $\{(\mathbf{A}, \mathbf{A}^T\mathbf{s}+\mathbf{e}) : \mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{s} \leftarrow \chi^n, \mathbf{e} \leftarrow \chi^m\} \approx_c \{(\mathbf{A}, \mathbf{u}) : \mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{u} \leftarrow \mathbb{Z}_q^m\}$, assuming LWE is hard. However, by a simple hybrid argument, we can replace vectors $\mathbf{s}, \mathbf{e}, \mathbf{u}$ with matrices $\mathbf{S}, \mathbf{E}, \mathbf{U}$ of appropriate dimensions.

### 3.2 Learning Parity with Noise (LPN)

The learning parity with noise is the binary ($\mathbb{Z}_2$) equivalent of the LWE problem. The search version of this problem requires one to solve a set of random linear equations perturbed by noise, and a decision version can be defined as in LWE. This problem is parameterized by the dimension $n$, the number of samples $m$ and the error distribution. Each component of the error vector is chosen independently from the Bernoulli distribution with parameter $p$ for $0 < p < 1/2$. Clearly, if $p = 1/2$, then the LPN distribution is identical to the uniform distribution. If $p = O(1/n)$, then an adversary can distinguish between the LPN distribution and the uniform distribution, given sufficiently many samples. Intuitively, the decision problem gets easier as $p$ decreases.

**Assumption 3** (Learning Parity with Noise)**.** Let $n$, $m$ be positive integers and $p$ be a real number such that $p < 1/2$. The Learning Parity with Noise problem $\mathsf{LPN}_{n,m,p}$, parameterized by the dimension of secret vector $n$, number of samples $m$ and the error probability $p$, states that the following distributions are computationally indistinguishable:

$$\left\{(\mathbf{A}, \mathbf{A}^T\mathbf{s} + \mathbf{e}) \; : \; \begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_2^{n\times m}, \\ \mathbf{s} \leftarrow \mathbb{Z}_2^n, \mathbf{e} \leftarrow \mathsf{Ber}_p^m \end{array}\right\} \approx_c \left\{(\mathbf{A}, \mathbf{u}) \; : \; \begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_2^{n\times m}, \\ \mathbf{u} \leftarrow \mathbb{Z}_2^m \end{array}\right\}$$

In this work, we will be using a variant of $\mathsf{LPN}$ called Knapsack-LPN. For certain range of parameters, this variant can be shown to be equivalent to $\mathsf{LPN}$ [MM11].

**Assumption 4** (Knapsack Learning Parity with Noise)**.** Let $n$, $m$ be positive integers and $p$ be a real number such that $p < 1/2$. The Knapsack Learning Parity with Noise problem $\mathsf{KLPN}_{n,m,p}$, parameterized by integers $n$, $m$ and $p$, states that the following distributions are computationally indistinguishable:

$$\left\{(\mathbf{A}, \mathbf{AE}) \; : \; \begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_2^{n\times m}, \\ \mathbf{E} \leftarrow \mathsf{Ber}_p^{m\times m} \end{array}\right\} \approx_c \left\{(\mathbf{A}, \mathbf{B}) \; : \; \begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_2^{n\times m}, \\ \mathbf{B} \leftarrow \mathbb{Z}_2^{n\times m} \end{array}\right\}$$

Clearly, if $n > m$, then the $\mathsf{KLPN}$ problem is easy. However, if $m > 2n$, then the $\mathsf{KLPN}$ problem is as hard as the $\mathsf{LPN}$ problem. In particular, there exists a reduction from $\mathsf{KLPN}_{n,m,p}$ to $\mathsf{LPN}_{m-n,m,p}$.

### 3.3 $q$-DDH Assumption

Let $\mathcal{G}(1^\lambda)$ be a randomized algorithm that takes as input the security parameter and outputs the description of a group $\mathbb{G}$ of prime order $p = \Theta(2^\lambda)$. The $q$-$\mathsf{DDH}$ assumption states that no PPT adversary, given the sequence $(g, g^a, g^{a^2}, \ldots, g^{a^{n-1}})$ can distinguish between $g^{a^n}$ and a uniformly random group element. In this experiment, $a$ is chosen uniformly at random from $\mathbb{Z}_p^*$. It is called a $q$-type assumption because the experiment is parameterized by the length of the powers-in-exponent sequence given to the adversary. Note that we are working in DDH hard groups. In particular, there is no bilinear pairing operation here.

**Assumption 5.** This assumption is parameterized with an integer $n \in \mathbb{Z}$. Let $(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$, $a \leftarrow \mathbb{Z}_p^*$ and $b \leftarrow \{0, 1\}$. Let $D = (p, \mathbb{G}, g, g^a, \ldots, g^{a^{n-1}})$. Let $T = g^{a^n}$ if $b = 0$, else $T \leftarrow \mathbb{G}$. The advantage of algorithm $\mathcal{A}$ in solving Assumption 5 is defined as

$$\mathsf{Adv}_{\mathcal{A}}^{q\text{-}\mathsf{DDH}} = \left| Pr[b \leftarrow \mathcal{A}(D, T)] - \frac{1}{2} \right|.$$

We say that Assumption 5 holds if for all PPT $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{q\text{-}\mathsf{DDH}}$ is negligible in $\lambda$.

This assumption can also be defined for groups of composite order $N = p, q$ for large primes $p, q$. Chase and Meiklejohn [CM14] showed that in such groups, subgroup decision assumption and DDH implies the $q$-type assumption. However, for simplicity, we will work with prime order groups in this work.

# 4 Constructing Verifiable Random Functions

In this section, we give a generic construction of VRFs from admissible hash functions, perfectly binding commitments, NIWIs and constrained pseudorandom functions for admissible hash compatible constraints. We also prove that it satisfies correctness, unique provability and pseudorandomness properties (as desribed in Definition 2.1). Later in Section 4.4, we give a slighlty modified construction for VRF that is selectively-secure assuming only puncturable pseudorandom functions.

Let $(h, \mathsf{AdmSample})$ be an admissible hash function that hashes $n(\lambda)$ bits to $\ell(\lambda)$ bits, $(\mathcal{P}, \mathcal{V})$ be a NIWI proof system for language $\mathcal{L}$ (where the language will be defined later), $(\mathsf{CS.Commit}, \mathsf{CS.Verify})$ be a perfectly binding commitment scheme with $\{\mathcal{M}_\lambda\}_\lambda, \{\mathcal{R}_\lambda\}_\lambda$ and $\{\mathcal{C}_\lambda\}_\lambda$ as the message, randomness and commitment space, and $\mathsf{CPRF} = (\mathsf{CPRF.Setup}, \mathsf{CPRF.Constrain}, \mathsf{CPRF.Eval})$ be a constrained pseudorandom function with $\{\mathcal{X}_\lambda\}_\lambda, \{\mathcal{Y}_\lambda\}_\lambda, \{\mathcal{K}_\lambda\}_\lambda$ and $\{\mathcal{K}^c_\lambda\}_\lambda$ as its domain, range, key and constrained key spaces. For simplicity assume that $\mathcal{K}_\lambda \cup \mathcal{K}^c_\lambda \subseteq \mathcal{M}_\lambda$, or in other words, all the PRF master keys and constrained keys lie in the message space of the commitment scheme. Also, let $\mathcal{X}_\lambda = \{0,1\}^{\ell(\lambda)}$.

First, we define the language $\mathcal{L}$. It contains instances of the form $(c_1, c_2, c_3, x, y) \in \mathcal{C}^3_\lambda \times \{0,1\}^{n(\lambda)} \times \mathcal{Y}_\lambda$ with the following witness relation:

$$\exists\ i, j \in \{1,2,3\},\ K, K' \in \mathcal{K}_\lambda \cup \mathcal{K}^c_\lambda,\ r, r' \in\ \mathcal{R}_\lambda \text{ such that}$$
$$i \neq j \wedge \mathsf{CS.Verify}(K, c_i, r) = 1 \wedge \mathsf{CS.Verify}(K', c_j, r') = 1 \wedge \mathsf{CPRF.Eval}(K, h(x)) = \mathsf{CPRF.Eval}(K', h(x)) = y.$$

Clearly the above language is in **NP** as it can be verified in polynomial time. Next we describe our construction for VRFs with message space $\{0,1\}^{n(\lambda)}$ and range space $\{\mathcal{Y}_\lambda\}_\lambda$.

## 4.1 Construction

- $\mathsf{Setup}(1^\lambda) \rightarrow (\mathrm{SK}, \mathrm{VK})$. It generates a PRF key for constrained pseudorandom function as $K \leftarrow \mathsf{CPRF.Setup}(1^\lambda)$. It also generates three independent commitments to the key $K$ as $c_i \leftarrow \mathsf{CS.Commit}(1^\lambda, K; r_i)$ for $i \leq 3$ where $r_i$ is sampled as $r_i \leftarrow \mathcal{R}_\lambda$, and sets the secret-verification key pair as $\mathrm{SK} = \left(K, \{(c_i, r_i)\}_{i \leq 3}\right), \mathrm{VK} = (c_1, c_2, c_3)$.

- $\mathsf{Evaluate}(\mathrm{SK}, x) \rightarrow (y, \pi)$. Let $\mathrm{SK} = \left(K, \{(c_i, r_i)\}_{i \leq 3}\right)$. It runs the PRF evaluation algorithm on $x$ as $y = \mathsf{CPRF.Eval}(K, h(x))$. It also computes a NIWI proof $\pi$ for the statement $(c_1, c_2, c_3, x, y) \in \mathcal{L}$ using NIWI prover algorithm $\mathcal{P}$ with $(i = 1, j = 2, K, K, r_1, r_2)$ as the witness, and outputs $y$ and $\pi$ as the evaluation and corresponding proof.

- $\mathsf{Verify}(\mathrm{VK}, x, y, \pi) \rightarrow \{0,1\}$. Let $\mathrm{VK} = (c_1, c_2, c_3)$. It runs NIWI verifier to check proof $\pi$ as $\mathcal{V}((c_1, c_2, c_3, x, y), \pi)$ and accepts the proof (outputs 1) iff $\mathcal{V}$ outputs 1.

## 4.2 Correctness, Unique Provability and Pseudorandomness

**Theorem 4.1.** If $(h, \mathsf{AdmSample})$ is an admissible hash function, $(\mathsf{CS.Commit}, \mathsf{CS.Verify})$ is a secure perfectly binding commitment scheme, $(\mathcal{P}, \mathcal{V})$ is a secure NIWI proof system for language $\mathcal{L}$, and $\mathsf{CPRF}$ is a secure *single-key* constrained pseudorandom function according to Definitions 2.5, 2.4, 2.3, and 2.7 (respectively), then the above construction forms an adaptively-secure VRF satisfying correctness, unique provability and pseudorandomness properties as described in Definition 2.1.

**Correctness.** For every well-formed secret and verification key pair $(\mathrm{SK}, \mathrm{VK}) \leftarrow \mathsf{Setup}(1^\lambda)$, we know that both $c_1$ and $c_2$ are commitments to PRF key $K$ with $r_1$ and $r_2$ as the corresponding openings, where $\mathrm{SK} = \left(K, \{(c_i, r_i)\}_{i \leq 3}\right)$. Therefore, by perfect correctness of the constrained PRF and NIWI proof system, we can conclude that the above construction satisfies the VRF correctness condition.

**Unique Provability.** We will prove this by contradiction. Assume that the above construction does not satisfy unique provability property. This implies that there exists $(\mathrm{VK}, x, y_1, \pi_1, y_2, \pi_2)$ such that $y_1 \neq y_2$ and $\Pr[\mathsf{Verify}(\mathrm{VK}, x, y_i, \pi_i) = 1] > 2^{-\Omega(\lambda)}$ for both $i \in \{1, 2\}$. To prove that this is not possible, we show that at least one of these proof verifications must involve verifying a NIWI proof for an invalid instance. Formal arguments proceed as follows:

- Let $\mathrm{VK} = (c_1, c_2, c_3)$. Since the commitment scheme is *perfectly binding*, we know that for each $i \in \{1, 2, 3\}$ there exists *at most* one key $K_i$ such that there exists an $r_i$ which is a valid opening for $c_i$, i.e. $\mathsf{CS.Verify}(K_i, c_i, r_i) = 1$.
- Suppose $c_i$ is a commitment to key $K_i$ for $i \leq 3$, and $\mathsf{CPRF.Eval}(K_1, x) = \mathsf{CPRF.Eval}(K_2, x) = y_1$. Now since $y_1 \neq y_2$, thus even when $\mathsf{CPRF.Eval}(K_3, x) = y_2$ holds, we know that $(c_1, c_2, c_3, x, y_2) \notin \mathcal{L}$ as no two keys out of $K_1, K_2, K_3$ evaluate to $y_2$ on input $x$. Therefore, at least one proof out of $\pi_1$ and $\pi_2$ is a proof for an incorrect statement.
- However, by statistical soundness of NIWI proof system, we know that for all instances not in $\mathcal{L}$, probability that any proof gets verified is at most $2^{-\Omega(\lambda)}$. Therefore, if the above construction does not satisfy unique provability, then the NIWI proof system is not statistically sound which contradicts our assumption. Hence, unique provability follows from perfect binding property of the commitment scheme and statistical soundness of NIWI proof system.

**Pseudorandomness.** The pseudorandomness proof follows from a sequence of hybrid games. The high level proof idea is as follows. We start by partitioning the input space into query and challenge partition using the admissible hash function. After partitioning we observe that to answer evaluation queries we only need a constrained PRF key which can evaluate on inputs in the query partition, however to give a proof we still need the master PRF key. Next we note that to compute the NIWI proofs we only need openings for any two commitments out of the three. Thus, we could switch one of the strings $c_i$ to commit to the constrained key instead. This follows from the hiding property of the commitment scheme. Now we observe that we only need to compute NIWI proofs for the inputs in the query partition, thus we could use a master key - constrained key pair instead of using the master key - master key pair as the NIWI witness. This follows from witness indistinguishability property of NIWI proof system and the fact that the constrained and master key compute the same output on query partition. Using the same trick two more times, we could move to a hybrid game in which all three strings $c_i$'s are commitments of the constrained key. Finally, in this hybrid we could directly reduce the pseudorandomness security of VRF to constrained pseudorandomness security of the single-key secure constrained PRF. Below we describe the proof in detail.

Let $\mathcal{A}$ be any successful PPT adversary against the above construction in the VRF pseudorandomness game described in Definition 2.1 that makes at most $Q = Q(\lambda)$ evaluation queries, where $Q(\cdot)$ is some polynomial. We argue that such an adversary must break security of at least one of the underlying primitives. To formally prove our claim, we describe the following sequence of games where the first game models the real pseudorandomness security game. In Game $i$, advantage of $\mathcal{A}$ is defined as $\mathsf{Adv}^i_{\mathcal{A}} = |\Pr[\mathcal{A} \text{ wins}] - 1/2|$. We then show via a sequence of claims that if $\mathcal{A}$'s advantage is non-negligible in Game $i$, then it has non-negligible advantage in Game $i + 1$ as well. Finally, in the last game, we use $\mathcal{A}$ to break pseudorandomness security of the constrained PRF, thereby completing our proof. Note that in each successive hybrid we only provide the steps that differ.

**Game 1:** This game is defined as the original pseudorandomness security game described in Definition 2.1.

1. The challenger generates a PRF key for constrained pseudorandom function as $K \leftarrow \mathsf{CPRF.Setup}(1^\lambda)$. It also generates three independent commitments to the key $K$ as $c_i \leftarrow \mathsf{CS.Commit}(1^\lambda, K; r_i)$ for $i \leq 3$ where $r_i$ is sampled as $r_i \leftarrow \mathcal{R}_\lambda$. Finally, it sets the verification key as $\mathrm{VK} = (c_1, c_2, c_3)$, and sends VK to $\mathcal{A}$.

2. $\mathcal{A}$ queries the challenger on polynomially many messages $x_j$ to receive corresponding evaluations and proofs as $(y_j, \pi_j)$, where on each message $x_j$, the challenger computes $y_j = \mathsf{CPRF.Eval}(K, h(x_j))$ and $\pi_j$ as the NIWI proof for the statement $(c_1, c_2, c_3, x_j, y_j) \in \mathcal{L}$ using $(1, 2, K, K, r_1, r_2)$ as the witness.

3. Next, $\mathcal{A}$ sends $x^*$ as its challenge message to the challenger, where $x^*$ should never have been queried by $\mathcal{A}$. The challenger chooses a random bit $b \leftarrow \{0,1\}$. If $b = 1$, the challenger computes $y^*$ honestly as $y^* = \mathsf{CPRF.Eval}(K, h(x^*))$, else it samples $y^* \leftarrow \mathcal{Y}_\lambda$, and sends $y^*$ to $\mathcal{A}$.

4. $\mathcal{A}$ is allowed to query the challenger on polynomially many messages $x_j$ $(\neq x^*)$ as before. The challenger handles these as in pre-challenge query phase.

5. Finally, $\mathcal{A}$ sends its guess $b'$ to the challenger. $\mathcal{A}$ wins if $b' = b$.

**Game 2:** This is same as Game 1, except the challenger additionally samples $u \leftarrow \mathsf{AdmSample}(1^\lambda, Q)$. The challenger uses $u$ to partition the input space into query and challenge partitions. If the challenge message $x^*$ does not lie in challenge partition, or any of the queried messages $x_j$ do not belong to query partition, then the challenger aborts. Also, the challenger performs an additional artifical abort step [Wat05] to even out the probability of abort over all possible sequences of queries. Definition 2.5 states that for all $x_1, \ldots, x_Q, x^*$

$$\Pr[(\forall i \leq Q, P_u(x_i) = 0) \wedge P_u(x^*) = 1] \geq \frac{1}{\theta(Q)} = \tau_{\min}$$

where $\tau_{\min}$ be a non-negligible lower bound on the probability of partitioning by the hash function $h$.

1. The challenger generates a PRF key for constrained pseudorandom function as $K \leftarrow \mathsf{CPRF.Setup}(1^\lambda)$. It samples $u \leftarrow \mathsf{AdmSample}(1^\lambda, Q)$. It also generates three independent commitments to the key $K$ as $c_i \leftarrow \mathsf{CS.Commit}(1^\lambda, K; r_i)$ for $i \leq 3$ where $r_i$ is sampled as $r_i \leftarrow \mathcal{R}_\lambda$. Finally, it sets the verification key as $\mathrm{VK} = (c_1, c_2, c_3)$, and sends VK to $\mathcal{A}$.

2. $\mathcal{A}$ queries the challenger on polynomially many messages $x_j$ to receive corresponding evaluations and proofs as $(y_j, \pi_j)$. For each queried message $x_j$, the challenger first computes $P_u(x_j)$. If $P_u(x_j) = 1$, then it chooses a random bit $\gamma \leftarrow \{0,1\}$ and aborts. If it aborts, then $\mathcal{A}$ wins if $\gamma = 1$. Otherwise, it computes $y_j = \mathsf{CPRF.Eval}(K, h(x_j))$ and $\pi_j$ as the NIWI proof for the statement $(c_1, c_2, c_3, x_j, y_j) \in \mathcal{L}$ using $(1, 2, K, K, r_1, r_2)$ as the witness.

3. Next, $\mathcal{A}$ sends $x^*$ as its challenge message to the challenger, where $x^*$ should never have been queried by $\mathcal{A}$. The challenger computes $P_u(x_j)$. If $P_u(x_j) = 0$, then it chooses a random bit $\gamma \leftarrow \{0,1\}$ and aborts. Otherwise, it chooses a random bit $b \leftarrow \{0,1\}$. If $b = 1$, the challenger computes $y^*$ honestly as $y^* = \mathsf{CPRF.Eval}(K, h(x^*))$, else it samples $y^* \leftarrow \mathcal{Y}_\lambda$, and sends $y^*$ to $\mathcal{A}$.

4. $\mathcal{A}$ is allowed to query the challenger on polynomially many messages $x_j$ $(\neq x^*)$ as before. The challenger handles these as in pre-challenge query phase, i.e. performs the abort step.

5. Finally, $\mathcal{A}$ sends its guess $b'$ to the challenger. After $\mathcal{A}$ terminates and outputs $b'$, the challenger performs an additional abort step to reduce any possible dependence between the event $\mathcal{A}$ winning and abort happening. Let the sequence of queries issued by $\mathcal{A}$ be denoted as $\mathbf{x} = \{x_1, \ldots, x_Q, x^*\}$, and $\tau(\mathbf{x})$ be the probability that the challenger does not abort on this set of issued queries. After $\mathcal{A}$ terminates, $\mathbf{x}$ is well-defined, therefore the challenger estimates the probability of not aborting $\tau(\mathbf{x})$ as $\tau'$ by freshly sampling $w \leftarrow \mathsf{AdmSample}(1^\lambda, Q)$ $T$-times (where $T$ will be defined later in Lemma 4.1), and checking whether for all $j \in [1, Q]$, $P_u(x_j) = 0$ and $P_u(x^*) = 1$. Finally, the challenger artifically aborts with probability $\max(0, 1 - \tau_{\min}/\tau')$. If it aborts then it chooses a random bit $\gamma \leftarrow \{0,1\}$, and $\mathcal{A}$ wins if $\gamma = 1$. Otherwise, $\mathcal{A}$ wins if $b' = b$.

**Game 3:** This is same as Game 2, except the challenger also constrains the PRF key $K$ on constraint $u$. And instead of generating $c_3$ as a commitment to $K$, it is computed as a commitment to the constrained key. Note that the proofs $\pi_j$ for each evaluation query are generated same as before since the witness depends only on $r_1$ and $r_2$.

1. The challenger generates a PRF key for constrained pseudorandom function as $K \leftarrow \mathsf{CPRF.Setup}(1^\lambda)$. It samples $u \leftarrow \mathsf{AdmSample}(1^\lambda, Q)$ and computes the constrained key $K_u \leftarrow \mathsf{CPRF.Constrain}(K, u)$. It generates commitments $c_1$ and $c_2$ as $c_i \leftarrow \mathsf{CS.Commit}(1^\lambda, K; r_i)$, and commitment $c_3$ as $c_3 \leftarrow \mathsf{CS.Commit}(1^\lambda, K_u; r_3)$ where $r_i \leftarrow \mathcal{R}_\lambda$ for $i \leq 3$. Finally, it sets the verification key as $\mathrm{VK} = (c_1, c_2, c_3)$, and sends VK to $\mathcal{A}$.

**Game 4:** This is same as Game 3, except the challenger uses $(2, 3, K, K_u, r_2, r_3)$ as the witness instead of $(1, 2, K, K, r_1, r_2)$ for generating all NIWI proofs.

2. $\mathcal{A}$ queries the challenger on polynomially many messages $x_j$ to receive corresponding evaluations and proofs as $(y_j, \pi_j)$. For each queried message $x_j$, the challenger first computes $P_u(x_j)$. If $P_u(x_j) = 1$, then it chooses a random bit $\gamma \leftarrow \{0, 1\}$ and aborts. If it aborts, then $\mathcal{A}$ wins if $\gamma = 1$. Otherwise, it computes $y_j = \mathsf{CPRF.Eval}(K, h(x_j))$ and $\pi_j$ as the NIWI proof for the statement $(c_1, c_2, c_3, x_j, y_j) \in \mathcal{L}$ using $(2, 3, K, K_u, r_2, r_3)$ as the witness.

4. $\mathcal{A}$ is allowed to query the challenger on polynomially many messages $x_j$ $(\neq x^*)$ as before. The challenger handles these as in pre-challenge query phase, i.e. performs the abort step and uses $(2, 3, K, K_u, r_2, r_3)$ as the witness.

**Game 5:** This is same as Game 4, except the challenger computes $c_1$ as a commitment to $K_u$ instead of key $K$. Note that the proofs $\pi_j$ for each evaluation query are generated same as before since the witness depends only on $r_2$ and $r_3$.

1. The challenger generates a PRF key for constrained pseudorandom function as $K \leftarrow \mathsf{CPRF.Setup}(1^\lambda)$. It samples $u \leftarrow \mathsf{AdmSample}(1^\lambda, Q)$ and computes the constrained key $K_u \leftarrow \mathsf{CPRF.Constrain}(K, u)$. It generates commitment $c_2$ as $c_2 \leftarrow \mathsf{CS.Commit}(1^\lambda, K; r_2)$ and commitments $c_1, c_3$ as $c_i \leftarrow \mathsf{CS.Commit}(1^\lambda, K_u; r_i)$ where $r_i \leftarrow \mathcal{R}_\lambda$ for $i \leq 3$. Finally, it sets the verification key as $\mathrm{VK} = (c_1, c_2, c_3)$, and sends VK to $\mathcal{A}$.

**Game 6:** This is same as Game 5, except the challenger uses $(1, 3, K_u, K_u, r_1, r_3)$ as the witness instead of $(2, 3, K, K_u, r_2, r_3)$ for generating all NIWI proofs.

2. $\mathcal{A}$ queries the challenger on polynomially many messages $x_j$ to receive corresponding evaluations and proofs as $(y_j, \pi_j)$. For each queried message $x_j$, the challenger first computes $P_u(x_j)$. If $P_u(x_j) = 1$, then it chooses a random bit $\gamma \leftarrow \{0, 1\}$ and aborts. If it aborts, then $\mathcal{A}$ wins if $\gamma = 1$. Otherwise, it computes $y_j = \mathsf{CPRF.Eval}(K, h(x_j))$ and $\pi_j$ as the NIWI proof for the statement $(c_1, c_2, c_3, x_j, y_j) \in \mathcal{L}$ using $(1, 3, K_u, K_u, r_1, r_3)$ as the witness.

4. $\mathcal{A}$ is allowed to query the challenger on polynomially many messages $x_j$ $(\neq x^*)$ as before. The challenger handles these as in pre-challenge query phase, i.e. performs the abort step and uses $(1, 3, K_u, K_u, r_1, r_3)$ as the witness.

**Game 7:** This is same as Game 6, except the challenger computes $c_2$ as a commitment to $K_u$ instead of key $K$. Note that the proofs $\pi_j$ for each evaluation query are generated same as before since the witness depends only on $r_1$ and $r_3$.

1. The challenger generates a PRF key for constrained pseudorandom function as $K \leftarrow \mathsf{CPRF.Setup}(1^\lambda)$. It samples $u \leftarrow \mathsf{AdmSample}(1^\lambda, Q)$ and computes the constrained key $K_u \leftarrow \mathsf{CPRF.Constrain}(K, u)$. It generates commitments $c_1, c_2, c_3$ as $c_i \leftarrow \mathsf{CS.Commit}(1^\lambda, K_u; r_i)$ where $r_i \leftarrow \mathcal{R}_\lambda$ for $i \leq 3$. Finally, it sets the verification key as $\mathrm{VK} = (c_1, c_2, c_3)$, and sends VK to $\mathcal{A}$.

## 4.3 Indistinguishability of Hybrid Games

We now establish via a sequence of lemmas that the difference of the adversary's advantage between each adjacent game (described in Section 4.1) is at most negligible in the security parameter. Finally we show that if any adversary wins in the last game, then it wins pseudorandomness game against the constrained PRF challenger as well.

**Lemma 4.1.** If $(h, \mathsf{AdmSample})$ is $\theta$-admissible hash function in the sense of Definition 2.5, then for all PPT $\mathcal{A}$, $\mathsf{Adv}^2_{\mathcal{A}} \geq O(\tau_{\min} \cdot \epsilon)\mathsf{Adv}^1_{\mathcal{A}}$, where $\epsilon$ is the non-negligible lower bound on $\mathsf{Adv}^1_{\mathcal{A}}$.

*Proof.* In Game 2, $T = O(\epsilon^{-2}\tau_{\min}^{-1}\log(1/\epsilon)\log(1/\tau_{\min}))$ samples are taken to approximate the abort probability $\tau(\mathbf{x})$ as $\tau'$. Let $\mathsf{Abort}$ denote the event that the challenger aborts (both artifical and regular). Applying standard concentration bounds on the estimate of $\tau'$ (see [Wat05, HW10]), we could show that the following holds:

$$\Pr[\mathsf{Abort}] \geq 1 - \tau_{\min} - \tau_{\min}\frac{3\epsilon}{8}, \qquad \Pr[\overline{\mathsf{Abort}}] \geq \tau_{\min}(1 - \frac{\epsilon}{4}).$$

Using the above lower bounds, we can conclude that $\mathsf{Adv}^2_{\mathcal{A}} \geq O(\tau_{\min} \cdot \epsilon)\mathsf{Adv}^1_{\mathcal{A}}$. The analysis is same as that provided by Hohenberger and Waters [HW10, Lemmas 5.3 - 5.5]. ■

**Lemma 4.2.** If $(\mathsf{CS.Commit}, \mathsf{CS.Verify})$ is a computationally hiding commitment scheme, then for all PPT $\mathcal{A}$, $|\mathsf{Adv}^2_{\mathcal{A}} - \mathsf{Adv}^3_{\mathcal{A}}| \leq \mathrm{negl}(\lambda)$ for some negligible function $\mathrm{negl}(\cdot)$.

*Proof.* We describe a reduction algorithm $\mathcal{B}$ which plays the hiding security game, and uses $\mathcal{A}$'s advantage in distinguishing between Games 2 and 3. $\mathcal{B}$ runs Step 1 as in Game 3, except it does not compute $c_3$ itself. It sends $K$ and $K_u$ to the commitment scheme challenger and receives $c^*$ as the commitment. It sets $c_3 = c^*$ and opening $r_3 = \epsilon$ (i.e. the empty string). Next, it runs steps 2-5 as in Game 3. If $\mathcal{A}$ wins ($b' = b$), then $\mathcal{B}$ guesses 0 to indicate that $c^*$ was a commitment to $K$, else it guesses 1 to indicate it was a commitment to $K_u$.

We observe that when $c^*$ is generated as a commitment to $K$, then $\mathcal{B}$ simulates exactly the view of Game 2 to $\mathcal{A}$. Otherwise if $c^*$ is chosen as a commitment to $K_u$ the view is of Game 3. In addition, $\mathcal{B}$ does not need to know the real opening of $c^*$ for the NIWI proofs, thus the simulation is perfectly done. Therefore if $\mathsf{Adv}^2_{\mathcal{A}} - \mathsf{Adv}^3_{\mathcal{A}}$ is non-negligible, then $\mathcal{B}$ must also have non-negligible advantage in the hiding game. ■

**Lemma 4.3.** If $(\mathcal{P}, \mathcal{V})$ forms a secure NIWI proof system, then for all PPT $\mathcal{A}$, $|\mathsf{Adv}^3_{\mathcal{A}} - \mathsf{Adv}^4_{\mathcal{A}}| \leq \mathrm{negl}(\lambda)$ for some negligible function $\mathrm{negl}(\cdot)$.

*Proof.* For proving indistinguishability of Games 3 and 4, we define $Q$ intermediate hybrid games between these two, where $Q$ is the number of evaluation queries made by $\mathcal{A}$. Observe that in Game 3, NIWI proofs for all evaluation queries use $(1, 2, K, K, r_1, r_2)$ as the witness, however in Game 4 $(2, 3, K, K_u, r_2, r_3)$ is used as the witness. The proof idea is to use witness indistinguishability of the NIWIs to switch each witness one by one. Concretely, $i^{th}$ intermediate hybrid between Game 3 and 4 proceeds same as Game 3 except it uses $(2, 3, K, K_u, r_2, r_3)$ as the witness for first $i$ queries, i.e. for $j \leq i$, $(2, 3, K, K_u, r_2, r_3)$ is the witness and for $j > i$, $(1, 2, K, K, r_1, r_2)$ is used as the witness. For the analysis, Game 3 and 4 are regarded as the $0^{th}$ and $Q^{th}$ intermediate hybrids (respectively). Below we show that $\mathcal{A}$'s advantage in each consecutive intermediate hybrid is negligibly close.

We describe a reduction algorithm $\mathcal{B}$ which breaks witness indistinguishability of the NIWI proof system, if $\mathcal{A}$ distinguishes between intermediate hybrids $i$ and $i + 1$ with non-negligible probability. $\mathcal{B}$ runs the Step 1 as in Game 3 (i.e., honestly sets up all the key components). Next, it also runs Steps 2-5 same as in Game 3 except the following modifications. For $j \leq i$ i.e. first $i$ evaluation queries, $\mathcal{B}$ computes $\pi_j$ as the NIWI proof for the statement $(c_1, c_2, c_3, x_j, y_j) \in \mathcal{L}$ using $(2, 3, K, K_u, r_2, r_3)$ as the witness. For $(i + 1)^{th}$ evaluation query i.e. $j = i + 1$, $\mathcal{B}$ sends $((c_1, c_2, c_3, x_j, y_j), (1, 2, K, K, r_1, r_2), (2, 3, K, K_u, r_2, r_3))$ as the instance and witnesses to the NIWI challenger and receives $\pi^*$ as the proof. $\mathcal{B}$ sets $\pi_j = \pi^*$ for $j = i + 1$.

For $j > i+1$, it uses $(1, 2, K, K, r_1, r_2)$ as the witness. Finally, if $\mathcal{A}$ wins $(b' = b)$, then $\mathcal{B}$ guesses 0 to indicate that $(1, 2, K, K, r_1, r_2)$ was used as the witness in the NIWI proof, else it guesses 1 to indicate that $(2, 3, K, K_u, r_2, r_3)$ was used.

We observe that when $(1, 2, K, K, r_1, r_2)$ is used as the witness by the NIWI challenger, then $\mathcal{B}$ simulates exactly the view of intermediate hybrid $i$ to $\mathcal{A}$. Otherwise the view is of intermediate hybrid $i+1$. Therefore, $\mathcal{A}$'s advantage in any two consecutive intermediate hybrids is negligibly close as otherwise NIWI proof system does not satisfy witness indistinguishability. Hence, using $Q$ intermediate hybrids we have proved that switching all the witnesses in the NIWI proofs causes at most negligible dip in $\mathcal{A}$'s advantage in Game 3.

Therefore if $\mathsf{Adv}^3_{\mathcal{A}} - \mathsf{Adv}^4_{\mathcal{A}}$ is non-negligible, then the NIWI proof system is not secure in the sense of witness indistinguishability. ■

**Lemma 4.4.** If $(\mathsf{CS.Commit}, \mathsf{CS.Verify})$ is a computationally hiding commitment scheme, then for all PPT $\mathcal{A}$, $|\mathsf{Adv}^4_{\mathcal{A}} - \mathsf{Adv}^5_{\mathcal{A}}| \leq \text{negl}(\lambda)$ for some negligible function $\text{negl}(\cdot)$.

*Proof.* The proof of this lemma is also analogous to that of Lemma 4.2. ■

**Lemma 4.5.** If $(\mathcal{P}, \mathcal{V})$ forms a secure NIWI proof system, then for all PPT $\mathcal{A}$, $|\mathsf{Adv}^5_{\mathcal{A}} - \mathsf{Adv}^6_{\mathcal{A}}| \leq \text{negl}(\lambda)$ for some negligible function $\text{negl}(\cdot)$.

*Proof.* The proof of this lemma is also analogous to that of Lemma 4.3. ■

**Lemma 4.6.** If $(\mathsf{CS.Commit}, \mathsf{CS.Verify})$ is a computationally hiding commitment scheme, then for all PPT $\mathcal{A}$, $|\mathsf{Adv}^6_{\mathcal{A}} - \mathsf{Adv}^7_{\mathcal{A}}| \leq \text{negl}(\lambda)$ for some negligible function $\text{negl}(\cdot)$.

*Proof.* The proof of this lemma is also analogous to that of Lemma 4.2. ■

**Lemma 4.7.** If $\mathsf{CPRF}$ is a secure single-key constrained pseudorandom function (as per Definition 2.7), then for all PPT $\mathcal{A}$, $\mathsf{Adv}^7_{\mathcal{A}} \leq \text{negl}(\lambda)$ for some negligible function $\text{negl}(\cdot)$.

*Proof.* We describe a reduction algorithm $\mathcal{B}$ which plays the single-key constrained pseudorandomness game, and uses $\mathcal{A}$'s advantage in Game 7. $\mathcal{B}$ first samples $u \leftarrow \mathsf{AdmSample}(1^\lambda, Q)$. It sends $u$ to the PRF challenger as the constraint and receives constrained key $K_u$. $\mathcal{B}$ continues to run Step 1 as in Game 7, except now it does not sample the PRF master key $K$ and uses $K_u$ (sent by the challenger) as the constrained key instead. Next, it runs step 2 (i.e., answer evaluation queries) as in Game 7, except it uses $K_u$ for evaluating the PRF. It also runs step 3 as in Game 7, except on challenge input $x^*$, it sends $h(x^*)$ as its challenge to the PRF challenger and forwards the challenger's response $y^*$ to $\mathcal{A}$. Next, it answers evaluation queries as before, and finally performs the artifical abort step. If $\mathcal{B}$ aborts, then it submits a random bit $\gamma \leftarrow \{0, 1\}$ as its guess to the PRF challenger. Otherwise, $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

First, note that the reduction algorithm does not need the PRF master key since whenever $\mathcal{B}$ does not abort during evaluation phase, then it could evaluate the PRF using constrained key $K_u$ because by definition $\mathsf{CPRF.Eval}(K_u, h(x)) = \mathsf{CPRF.Eval}(K, h(x))$ whenever $P_u(x) = 0$. Second, if $P_u(x^*) = 1$, then $h(x^*)$ is a valid challenge point. Finally, we observe that whenever $\mathcal{B}$ doesn't abort, then $\mathcal{B}$ exactly simulates the view of Game 7 to $\mathcal{A}$. Therefore if $\mathsf{Adv}^7_{\mathcal{A}}$ is non-negligible, then $\mathcal{B}$ must also have non-negligible advantage in the single-key constrained pseudorandomness game. ■

## 4.4 Selectively-Secure VRFs

In this section, we give a modified construction which assumes puncturable PRFs instead of constrained PRFs for admissible hash compatible constraints. The trade-off is that we could only prove selective security of this construction. However, if we make sub-exponential security assumptions, then it could be proven to be adaptively-secure as well.

Let $(\mathcal{P}, \mathcal{V})$ be a NIWI proof system for language $\widetilde{\mathcal{L}}$ (where the language will be defined later), $(\mathsf{CS.Commit}, \mathsf{CS.Verify})$ be a perfectly binding commitment scheme with $\{\mathcal{M}_\lambda\}_\lambda, \{\mathcal{R}_\lambda\}_\lambda$ and $\{\mathcal{C}_\lambda\}_\lambda$ as the message, randomness and commitment space, and $\mathsf{PPRF} = (\mathsf{PPRF.Setup}, \mathsf{PPRF.Puncture}, \mathsf{PPRF.Eval})$ be a constrained pseudorandom function with $\{\mathcal{X}_\lambda\}_\lambda, \{\mathcal{Y}_\lambda\}_\lambda, \{\mathcal{K}_\lambda\}_\lambda$ and $\{\mathcal{K}^p_\lambda\}_\lambda$ as its domain, range, key and constrained key spaces. For simplicity assume that $\mathcal{K}_\lambda \cup \mathcal{K}^p_\lambda \subseteq \mathcal{M}_\lambda$, or in other words, all the PRF master keys and constrained keys lie in the message space of the commitment scheme.

First, we define the language $\widetilde{\mathcal{L}}$. It contains instances of the form $(c_1, c_2, c_3, x, y) \in \mathcal{C}^3_\lambda \times \mathcal{X}_\lambda \times \mathcal{Y}_\lambda$ with the following witness relation:

$$\exists\ i, j \in \{1, 2, 3\}\,,\ \ K, K' \in \mathcal{K}_\lambda \cup \mathcal{K}^p_\lambda,\ \ r, r' \in\ \mathcal{R}_\lambda \text{ such that}$$
$$i \neq j \wedge \mathsf{CS.Verify}(K, c_i, r) = 1 \wedge \mathsf{CS.Verify}(K', c_j, r') = 1 \wedge \mathsf{PPRF.Eval}(K, x) = \mathsf{PPRF.Eval}(K', x) = y.$$

Clearly the above language is in **NP** as it can be verified in polynomial time. Next we describe our construction for selectively-secure VRFs with message space $\{\mathcal{X}_\lambda\}_\lambda$ and range space $\{\mathcal{Y}_\lambda\}_\lambda$.

- $\mathsf{Setup}(1^\lambda) \rightarrow (\mathrm{SK}, \mathrm{VK})$. It generates a PRF key for punctured pseudorandom function as $K \leftarrow \mathsf{PPRF.Setup}(1^\lambda)$. It also generates three independent commitments to the key $K$ as $c_i \leftarrow \mathsf{CS.Commit}(1^\lambda, K; r_i)$ for $i \leq 3$ where $r_i$ is sampled as $r_i \leftarrow \mathcal{R}_\lambda$, and sets the secret-verification key pair as $\mathrm{SK} = \Big(K, \{(c_i, r_i)\}_{i \leq 3}\Big)$, $\mathrm{VK} = (c_1, c_2, c_3)$.
- $\mathsf{Evaluate}(\mathrm{SK}, x) \rightarrow (y, \pi)$. Let $\mathrm{SK} = \Big(K, \{(c_i, r_i)\}_{i \leq 3}\Big)$. It runs the PRF evaluation algorithm on $x$ as $y = \mathsf{PPRF.Eval}(K, x)$. It also computes a NIWI proof $\pi$ for the statement $(c_1, c_2, c_3, x, y) \in \widetilde{\mathcal{L}}$ using NIWI prover algorithm $\mathcal{P}$ with $(i = 1, j = 2, K, K, r_1, r_2)$ as the witness, and outputs $y$ and $\pi$ as the evaluation and corresponding proof.
- $\mathsf{Verify}(\mathrm{VK}, x, y, \pi) \rightarrow \{0, 1\}$. Let $\mathrm{VK} = (c_1, c_2, c_3)$. It runs NIWI verifier to check proof $\pi$ as $\mathcal{V}((c_1, c_2, c_3, x, y), \pi)$ and accepts the proof (outputs 1) iff $\mathcal{V}$ outputs 1.

**Theorem 4.2.** If $(\mathsf{CS.Commit}, \mathsf{CS.Verify})$ is a secure perfectly binding commitment scheme, $(\mathcal{P}, \mathcal{V})$ is a secure NIWI proof system for language $\widetilde{\mathcal{L}}$, and $\mathsf{PPRF}$ is a secure puncturable pseudorandom function according to Definitions 2.4, 2.3, and 2.9 (respectively), then the above construction forms a selectively-secure VRF satisfying correctness, unique provability and pseudorandomness properties as described in Definition 2.2.

**Proof Sketch.** Correctness and unique provability of the above scheme could be proven similar to as in Section 4.2. The proof of pseudorandomness is also similar to that provided before with the following differences — (1) since we are only targeting selective security, the reduction algorithm receives the challenge input from the adversary at the start of the game, thus it does not need to perform any partitioning or abort, (2) in the final hybrid game, the reduction algorithm uses the adversary to attack the punctured pseudorandomness property. The main idea in the reduction to punctured pseudorandomness is that since at the start of the game adversary sends the challenge input to the reduction algorithm, the reduction algorithm could get a punctured key from the PRF challenger and use it inside the commitments as well as to answer each evaluation query.

# 5 Perfectly Binding Commitment Schemes

In this section, we give new constructions of perfectly binding non-interactive commitments from the Learning with Errors assumption and the Learning Parity with Noise assumption. These constructions are in the standard model without trusted setup. As mentioned in the introduction, there are already simple solutions [JKPT12] known from LWE/LPN when there is a trusted setup.

We will first present a construction based on the LWE assumption. Next, we will adapt this solution to work with the LPN assumption. However, this adaptation only works with low noise (that is, the Bernoulli parameter is $1/\sqrt{n}$). We also propose a different approach for constructing perfectly binding non-interactive commitments from the standard constant noise LPN problem. This approach reduces to finding error correcting codes with 'robust' generator matrices. Currently, we do not have any explicit [6] constructions for such error correcting codes, and finding such a family of generator matrices is an interesting open problem.

## 5.1 Construction from Learning with Errors

In this commitment scheme, our message space is $\{0,1\}$ for simplicity. To commit to a bit $x$, one first chooses two vectors $\mathbf{s}$, $\mathbf{w}$ and outputs $\mathbf{w}$ and $\mathbf{w}^T\mathbf{s} + x$. Clearly, this is not binding since there could be different $\mathbf{s}$ vectors that open to different messages. Therefore, we need to ensure that the vector $\mathbf{s}$ is fixed. To address this, we choose a matrix $\mathbf{B}$ with certain structure and output $\mathbf{B}$ and $\mathbf{B}^T\mathbf{s} + \mathsf{noise}$. The special structure of the matrix ensures that there cannot be two different vectors $\mathbf{s}_1, \mathbf{s}_2$ and noise vectors $\mathsf{noise}_1, \mathsf{noise}_2$ such that $\mathbf{B}^T\mathbf{s}_1 + \mathsf{noise}_1 = \mathbf{B}^T\mathbf{s}_2 + \mathsf{noise}_2$. Computational hiding of the committed bit follows from the fact that even though $\mathbf{B}$ has special structure, it 'looks' like a random matrix, and therefore we can use the LWE assumption to argue that $\mathbf{B}^T\mathbf{s} + \mathsf{noise}$ looks random, and therefore the message $x$ is hidden.

We will now describe the algorithms formally.

- $\mathsf{Commit}(1^n, x \in \{0,1\})$ : The commitment algorithm first sets the LWE modulus $p = 2^{n^\epsilon}$ for some $\epsilon < 1/2$ and error distribution $\chi = \mathcal{D}_\sigma$ where $\sigma = n^c$ for some constant $c$. Next, it chooses a matrix $\mathbf{A} \leftarrow \mathbb{Z}_p^{n\times n}$, low norm matrices $\mathbf{C} \leftarrow \chi^{n\times n}$, $\mathbf{E} \leftarrow \chi^{n\times n}$ and constructs $\mathbf{D} = p/(4n^{c+1}) \cdot \mathbf{I}$ (here $\mathbf{I}$ is the $n \times n$ identity matrix). Let $\mathbf{B} = [\mathbf{A} \mid \mathbf{AC} + \mathbf{D} + \mathbf{E}]$.

  It then chooses vectors $\mathbf{s} \leftarrow \chi^n$, $\mathbf{w} \leftarrow \mathbb{Z}_p^n$, $\mathbf{e} \leftarrow \chi^{2n}$ and $f \leftarrow \chi$, and computes $\mathbf{y} = \mathbf{B}^T\mathbf{s} + \mathbf{e}$ and $z = \mathbf{w}^T\mathbf{s} + x(p/2) + f$. If either $\|\mathbf{C}\| > n^{c+2}$ or $\|\mathbf{E}\| > n^{c+2}$ or $\|\mathbf{e}\| > 2n^{c+1}$ or $\|\mathbf{s}\| > n^{c+1}$ or $f > p/100$, the commitment algorithm outputs $x$ as the commitment. Else, the commitment consists of $(p, c, \mathbf{B}, \mathbf{w}, \mathbf{y}, z)$.

- $\mathsf{Verify}(\mathsf{com}, x, (\mathbf{C}, \mathbf{E}, \mathbf{e}, \mathbf{s}, f))$ : Let $\mathsf{com} = (p, c, \mathbf{B}, \mathbf{w}, \mathbf{y}, z)$. The verification algorithm first checks if $\|\mathbf{C}\| \le n^{c+2}$, $\|\mathbf{E}\| \le n^{c+2}$, $\|\mathbf{e}\| \le 2n^{c+1}$, $\|\mathbf{s}\| \le n^{c+1}$ and $f \le p/100$. Next, it checks that $\mathbf{B} = [\mathbf{A} \mid \mathbf{AC} + \mathbf{D} + \mathbf{E}]$, $\mathbf{B}^T\mathbf{s} + \mathbf{e} = \mathbf{y}$ and $\mathbf{w}^T\mathbf{s} + x(p/2) + f = z$, where $\mathbf{D} = p/(4n^{c+1}) \cdot \mathbf{I}$. If all these checks pass, it outputs 1.

**Perfect Correctness** Suppose there exist two different openings for the same commitment. Let $\mathbf{s}_1, \mathbf{e}_1, f_1, \mathbf{C}_1, f_1$ and $\mathbf{s}_2, \mathbf{e}_2, f_2, \mathbf{C}_2, f_2$ be the two openings. We will first show that $\mathbf{s}_1 = \mathbf{s}_2$ and $\mathbf{e}_1 = \mathbf{e}_2$. Next, we will argue that if $\mathbf{s}_1 = \mathbf{s}_2$, then the commitment cannot be opened to two different bits.

Suppose $\mathbf{s}_1 \neq \mathbf{s}_2$. Since $\mathbf{B}^T\mathbf{s}_1 + \mathbf{e}_1 = \mathbf{B}^T\mathbf{s}_2 + \mathbf{e}_2$, it follows that $\mathbf{B}^T(\mathbf{s}_1 - \mathbf{s}_2) = \mathbf{e}_2 - \mathbf{e}_1$. Let $\mathbf{e}_1^1$ and $\mathbf{e}_2^1$ denote the first $n$ components of $\mathbf{e}_1$ and $\mathbf{e}_2$ respectively. Then $\mathbf{A}^T(\mathbf{s}_1 - \mathbf{s}_2) = \mathbf{e}_1^1 - \mathbf{e}_2^1$. Note that $\|\mathbf{e}_2 - \mathbf{e}_1\| \le 4n^{c+1}$, and therefore, $\left\|\mathbf{A}^T(\mathbf{s}_1 - \mathbf{s}_2)\right\| \le 4n^{c+1}$.

Since $\mathbf{C}_1$ and $\mathbf{C}_2$ are matrices with low norm entries, it follows that $\|\mathbf{C}_1\| \le n^{c+2}$ and $\|\mathbf{C}_2\| \le n^{c+2}$. This implies $\left\|\mathbf{C}_1^T\mathbf{A}^T(\mathbf{s}_1 - \mathbf{s}_2)\right\| \le 4n^{2c+3}$. Similarly, since $\|\mathbf{E}_1\| \le n^{c+2}$, $\left\|\mathbf{E}_1^T(\mathbf{s}_1 - \mathbf{s}_2)\right\| \le 2n^{2c+3}$. However, since the matrix $\mathbf{D}$ has 'medium-sized' entries, if $\mathbf{s}_1 \neq \mathbf{s}_2$, it follows that $\left\|\mathbf{D}^T(\mathbf{s}_1 - \mathbf{s}_2)\right\|_\infty \ge p/(4n^{c+1})$. Additionally, since $\mathbf{D}$ has medium-sized entries, we could also say that each entry of vector $\mathbf{D}^T(\mathbf{s}_1 - \mathbf{s}_2)$ is at most $p/2$. This is because $\left\|\mathbf{D}^T(\mathbf{s}_1 - \mathbf{s}_2)\right\|_\infty \le \left\|\mathbf{D}^T\right\|_\infty \cdot \|\mathbf{s}_1 - \mathbf{s}_2\|_\infty \le p/(4n^{c+1}) \cdot 2n^{c+1} = p/2$. Therefore, the vector $\mathbf{D}^T(\mathbf{s}_1 - \mathbf{s}_2)$ is sufficiently long, i.e. $\left\|\mathbf{D}^T(\mathbf{s}_1 - \mathbf{s}_2)\right\|_\infty \in \left[p/(4n^{c+1}), p/2\right]$.

Next, let us consider the norm of vector $\mathbf{B}^T(\mathbf{s}_1 - \mathbf{s}_2)$. Recall that $\mathbf{B} = [\mathbf{A} \mid \mathbf{AC} + \mathbf{D} + \mathbf{E}]$. Consider the matrix $\mathbf{X} = [\mathbf{A} \mid \mathbf{AC} + \mathbf{E}]$, i.e. it is same as $\mathbf{B}$ except it does not contain matrix $\mathbf{D}$. Using triangle inequality, we can write that $\left\|\mathbf{X}^T(\mathbf{s}_1 - \mathbf{s}_2)\right\| \le \left\|\mathbf{A}^T(\mathbf{s}_1 - \mathbf{s}_2)\right\| + \left\|\mathbf{C}_1^T\mathbf{A}^T(\mathbf{s}_1 - \mathbf{s}_2)\right\| + \left\|\mathbf{E}_1^T(\mathbf{s}_1 - \mathbf{s}_2)\right\| \le 8n^{2c+3}$. Therefore, we could also say that each entry of vector $\mathbf{X}^T(\mathbf{s}_1 - \mathbf{s}_2)$ is at most $8n^{2c+3}$, i.e. $\left\|\mathbf{X}^T(\mathbf{s}_1 - \mathbf{s}_2)\right\|_\infty \le 8n^{2c+3}$.

[6] We note that most randomly chosen linear codes satisfy this property.

We know that $\mathbf{B}^T(\mathbf{s}_1 - \mathbf{s}_2) = \mathbf{X}^T(\mathbf{s}_1 - \mathbf{s}_2) + [\mathbf{0} \mid \mathbf{D}]^T(\mathbf{s}_1 - \mathbf{s}_2)$. Therefore, given the above bounds, we could conclude that $p/(4n^{c+1}) - 8n^{2c+3} \leq \left\|\mathbf{B}^T(\mathbf{s}_1 - \mathbf{s}_2)\right\|_\infty \leq p/2 + 8n^{2c+3}$. Since, $p = 2^{n^\epsilon}$, we know that for sufficiently large values of $n$, $p/(8n^{c+1}) \leq \left\|\mathbf{B}^T(\mathbf{s}_1 - \mathbf{s}_2)\right\|_\infty < p$. However, this is a contradiction since $\mathbf{B}^T(\mathbf{s}_1 - \mathbf{s}_2) = \mathbf{e}_2 - \mathbf{e}_1$ and $\|\mathbf{e}_2 - \mathbf{e}_1\| \leq 4n^{c+1}$, thus $\left\|\mathbf{B}^T(\mathbf{s}_1 - \mathbf{s}_2)\right\|_\infty < 4n^{c+1}$.

Now, if $\mathbf{s}_1 = \mathbf{s}_2$ and $f_1, f_2$ are both at most $p/100$, then $\mathbf{w}^T\mathbf{s}_1 + f_1$ cannot be equal to $\mathbf{w}^T\mathbf{s}_2 + f_2 + p/2$. This implies that any commitment cannot be opened to two different bits.

### 5.1.1 Computational Hiding

The computational hiding proof follows from a sequence of standard hybrid games. First, we can switch $\mathbf{B}$ to a uniformly random matrix. This follows from LWE with short secrets ($\mathbf{C}$ is the secret here). After that, $\mathbf{y}$ and $z$ can be made random, again using LWE with short secrets. At this point, since $z$ is random, there is no information about the commitment bit $x$. The formal proof is described via a sequence of hybrid experiments.

- **Hybrid 0**: This corresponds to the real experiment.
- **Hybrid 1**: This hybrid experiment is identical to the previous one, except that the challenger never outputs the message in clear.

  The challenger chooses $\mathbf{A} \leftarrow \mathbb{Z}_p^{n\times n}$, low norm matrices $\mathbf{C} \leftarrow \chi^{n\times n}$, $\mathbf{E} \leftarrow \chi^{n\times n}$, constructs $\mathbf{D} = p/(4n^{c+1}) \cdot \mathbf{I}$ and $\mathbf{B} = [\mathbf{A}|\mathbf{AC} + \mathbf{D} + \mathbf{E}]$.

  It then chooses vectors $\mathbf{s} \leftarrow \chi^n$, $\mathbf{w} \leftarrow \mathbb{Z}_p^n$, $\mathbf{e} \leftarrow \chi^n$ and $f \leftarrow \chi$, and computes $\mathbf{y} = \mathbf{B}^T\mathbf{s} + \mathbf{e}$ and $z = \mathbf{w}^T\mathbf{s} + x(p/2) + f$. The commitment consists of $(\mathbf{B}, \mathbf{w}, \mathbf{y}, z)$.
- **Hybrid 2**: In this hybrid, the challenger sets $\mathbf{B}$ to be a uniformly random matrix.

  It chooses $\mathbf{B} \leftarrow \mathbb{Z}_p^{n\times 2n}$, $\mathbf{s} \leftarrow \chi^n$, $\mathbf{w} \leftarrow \mathbb{Z}_p^n$, $\mathbf{e} \leftarrow \chi^n$ and $f \leftarrow \chi$, and computes $\mathbf{y} = \mathbf{B}^T\mathbf{s} + \mathbf{e}$ and $z = \mathbf{w}^T\mathbf{s} + x(p/2) + f$. The commitment consists of $(\mathbf{B}, \mathbf{w}, \mathbf{y}, z)$.
- **Hybrid 3**: In this hybrid, the challenger sets $\mathbf{y}$ and $z$ to be uniformly random.

  It chooses $\mathbf{B} \leftarrow \mathbb{Z}_p^{n\times 2n}$, $\mathbf{w} \leftarrow \mathbb{Z}_p^n$, $\mathbf{y} \leftarrow \mathbb{Z}_p^{2n}$ and $z \leftarrow \mathbb{Z}_p$. The commitment consists of $(\mathbf{B}, \mathbf{w}, \mathbf{y}, z)$.

Let $\mathsf{Adv}_i^{\mathcal{A}}$ denote the advantage of adversary $\mathcal{A}$ in Hybrid $i$. We will now show that for all $i \in \{0, 1, 2\}$, $\mathsf{Adv}_i^{\mathcal{A}} - \mathsf{Adv}_{i+1}^{\mathcal{A}}$ is negligible in the security parameter $n$.

**Claim 5.1.** For any adversary $\mathcal{A}$, $\mathsf{Adv}_0^{\mathcal{A}} - \mathsf{Adv}_1^{\mathcal{A}} \leq \text{negl}(n)$.

*Proof.* The only difference between Hybrid 0 and Hybrid 1 is that the challenger, in hybrid 0, outputs the message in clear if either $\|\mathbf{C}\| > n^{c+2}$ or $\|\mathbf{E}\| > n^{c+2}$ or $\|\mathbf{e}\| > 2n^{c+1}$ or $\|\mathbf{s}\| > n^{c+1}$ or $f > p/100$. From Lemma 2.1, it follows directly that $\Pr[\|\mathbf{C}\| > n^{c+2}$ or $\|\mathbf{E}\| > n^{c+2}$ or $\|\mathbf{e}\| > 2n^{c+1}$ or $\|\mathbf{s}\| > n^{c+1}$ or $f > p/100] \leq \text{negl}(n)$. ■

**Claim 5.2.** Assuming LWE with Short Secrets problem is hard for $p = 2^{n^\epsilon}$ and $\chi = \mathcal{D}_\sigma$ (where $\sigma = n^c$), for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}_1^{\mathcal{A}} - \mathsf{Adv}_2^{\mathcal{A}} \leq \text{negl}(n)$.

*Proof.* Suppose there exists a PPT adversary $\mathcal{A}$ such that $\mathsf{Adv}_1^{\mathcal{A}} - \mathsf{Adv}_2^{\mathcal{A}} = \eta$. Then there exists a PPT algorithm that can break the LWE with short secrets assumption with advantage at least $\eta$. The algorithm $\mathcal{B}$ receives as LWE challenge two matrices $\mathbf{X}, \mathbf{Y}$, where $\mathbf{Y}$ is either a uniformly random matrix, or $\mathbf{Y} = \mathbf{XC}+\mathbf{E}$ and $\mathbf{C}, \mathbf{E}$ are matrices with entries from $\mathcal{D}_\sigma$. $\mathcal{B}$ then chooses a diagonal matrix $\mathcal{D}$ with $\{p/(8n^c)$ as diagonal entries, sets $\mathbf{A} = \mathbf{X}$, $\mathbf{B} = [\mathbf{A}|\mathbf{Y} + \mathbf{D}]$. Next, it chooses vectors $\mathbf{w}, \mathbf{s}, \mathbf{e}, f$ and bit $b$. It sets $\mathbf{y} = \mathbf{B}^T\mathbf{s} + \mathbf{e}$ and $z = \mathbf{w}^T\mathbf{s} + f + b(p/2)$. If the adversary guesses $b$ correctly, then $\mathcal{B}$ guesses that $\mathbf{Y}$ is an LWE sample, else it guesses that $\mathcal{Y}$ is truly random. ■

**Claim 5.3.** Assuming LWE with Short Secrets problem is hard for $p = 2^{n^\epsilon}$ and $\chi = \mathcal{D}_\sigma$ (where $\sigma = n^c$), for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}_2^{\mathcal{A}} - \mathsf{Adv}_3^{\mathcal{A}} \leq \text{negl}(n)$.

*Proof.* Suppose there exists a PPT adversary $\mathcal{A}$ such that $\mathsf{Adv}_2^{\mathcal{A}} - \mathsf{Adv}_3^{\mathcal{A}} = \eta$. Then there exists a PPT algorithm that can break the LWE with short secrets assumption with advantage at least $\eta$. For this reduction, let us assume the reduction algorithm gets as LWE challenge $(\mathbf{X}, \mathbf{w})$ and $(\mathbf{a}, c)$ where $(\mathbf{a}, c)$ are either truly random, or there exist low norm vectors $\mathbf{s}, \mathbf{e}$ and $f$ such that $\mathbf{a} = \mathbf{X}^T\mathbf{s} + \mathbf{e}$ and $c = \mathbf{w}^T\mathbf{s} + f$. The reduction algorithm sets $\mathbf{B} = \mathbf{X}$, $\mathbf{y} = \mathbf{a}$, $z = c + b(p/2)$ and sends $(\mathbf{B}, \mathbf{w}, \mathbf{y}, z)$ to the adversary. If the adversary guesses $b$ correctly, then $\mathcal{B}$ guesses that $\mathbf{a}, c$ are LWE samples, else it guesses that they are truly random. ■

Finally, since $\mathbf{y}, z$ is uniformly random in the final hybrid, any adversary has 0 advantage in the final hybrid game.

## 5.2 Construction from Learning Parity with Low Noise

We will now construct a perfectly binding non-interactive commitment scheme that can be proven secure under the low noise LPN assumption. At a high level, this solution is similar to our LWE solution. The message space is $\{0, 1\}$, and to commit to a bit $x$, we choose a vector $\mathbf{w}$, secret vector $\mathbf{s}$ and output $\mathbf{w}, \mathbf{w}^T\mathbf{s}+x$ as part of the commitment. However, this is not enough, as there could exist $\mathbf{s}_1, \mathbf{s}_2$ such that $\mathbf{w}^T\mathbf{s}_1 + 1 = \mathbf{w}^T\mathbf{s}_2$. To prevent this, the commitment also consists of a matrix $\mathbf{B}$ chosen from a special distribution, and $\mathbf{B}^T\mathbf{s} + \mathsf{noise}'$ fixes the vector $\mathbf{s}$. Drawing parallels with the LWE solution, we use an error correcting code's generator matrix $\mathbf{G}$ instead of the matrix $\mathbf{D}$ used in the LWE solution. Both these matrices have a similar role: to map non-zero vectors to vectors with high hamming weight/high norm.

An important point to note here is that the Bernoulli parameter needs to be $O(1/\sqrt{n})$. This is necessary for proving perfect binding. Recall, in the LWE perfect binding proof, we argue that since $\mathbf{A}^T\mathbf{s}$ has low norm, $\mathbf{C}^T\mathbf{A}^T\mathbf{s}$ also has low norm. For the analogous argument to work here, the error distribution must be $O(1/\sqrt{n})$. In that case, we can argue that if the error distribution has hamming weight fraction at most $1/100\sqrt{n}$ and each row of $\mathbf{C}$ has hamming weight fraction at most $1/100\sqrt{n}$, then $\mathbf{C}^T\mathbf{A}^T\mathbf{s}$ has hamming weight fraction at most $1/10000$. If the noise rate was constant, then we cannot get an upper bound on the hamming weight fraction of $\mathbf{C}^T\mathbf{A}^T\mathbf{s}$.

We will now describe the formal construction. Let $\beta = 1/(100\sqrt{n})$ and $\chi = \mathsf{Ber}_\beta$ the noise distribution. Let $\{\mathbf{G}_n \in \mathbb{Z}_2^{n \times 10n}\}_{n \in \mathbb{N}}$ be a family of generator matrices for error correcting codes where the distance of the code generated by $\mathbf{G}_n$ is at least $4n$.

- $\mathsf{Commit}(1^n, x \in \{0, 1\})$ : Let $m = 10n$. Choose random matrices $\mathbf{A} \leftarrow \mathbb{Z}_2^{n \times m}$, $\mathbf{w} \leftarrow \mathbb{Z}_2^n$ and $\mathbf{C} \leftarrow \chi^{m \times m}$. Let $\mathbf{B} = [\mathbf{A} \mid \mathbf{AC} + \mathbf{G}]$. Choose secret vector $\mathbf{s} \leftarrow \mathbb{Z}_2^n$, error vector $\mathbf{e} \leftarrow \chi^{2m}$ and set $\mathbf{y} = \mathbf{B}^T\mathbf{s} + \mathbf{e}$. If either $\mathsf{Ham\text{-}Wt}(\mathbf{e}) > m/(25\sqrt{n})$ or there exists some row $\mathbf{c}_i$ of matrix $\mathbf{C}$ such that $\mathsf{Ham\text{-}Wt}(\mathbf{c}_i) > m/(50\sqrt{n})$, output the message $x$ in clear as the commitment. Else, let $z = \mathbf{w}^T\mathbf{s} + x$. The commitment string $\mathsf{com}$ is set to be $(\mathbf{B}, \mathbf{w}, \mathbf{y}, z)$.

- $\mathsf{Verify}(\mathsf{com}, x, (\mathbf{s}, \mathbf{e}, \mathbf{C}))$ : Let $\mathsf{com} = (\mathbf{B}, \mathbf{w}, \mathbf{y}, z)$. The verification algorithm first checks that $\mathsf{Ham\text{-}Wt}(\mathbf{e}) \leq m/(25\sqrt{n})$ and all rows $\mathbf{c}_i$ of $\mathbf{C}$ satisfy $\mathsf{Ham\text{-}Wt}(\mathbf{c}_i) \leq m/(50\sqrt{n})$. Next, it checks if $\mathbf{B} = [\mathbf{A} \mid \mathbf{AC} + \mathbf{G}]$, $\mathbf{y} = \mathbf{B}^T\mathbf{s} + \mathbf{e}$ and $z = \mathbf{w}^T\mathbf{s} + x$. If all checks pass, it outputs 1, else it outputs 0.

**Perfect Correctness** First, we will argue perfect correctness. Suppose there exists a commitment $\mathsf{com} = (\mathbf{B}, \mathbf{w}, \mathbf{y}, z)$ that can be opened to two different messages. Then there exist two different reveals $(\mathbf{s}_1, \mathbf{e}_1, \mathbf{C}_1)$ and $(\mathbf{s}_2, \mathbf{e}_2, \mathbf{C}_2)$ such that $\mathbf{B}^T\mathbf{s}_1 + \mathbf{e}_1 = \mathbf{y} = \mathbf{B}^T\mathbf{s}_2 + \mathbf{e}_2$, $\mathbf{w}^T\mathbf{s}_1 + 0 \ = z = \mathbf{w}^T\mathbf{s}_2 + 1$ and $[\mathbf{A}|\mathbf{AC}_1 + \mathbf{G}] = \mathbf{B}$ $= [\mathbf{A}|\mathbf{AC}_2 + \mathbf{G}]$. We will first show that $\mathbf{s}_1 = \mathbf{s}_2$, and then show that this implies perfect binding.

For proving that $\mathbf{s}_1 = \mathbf{s}_2$ and $\mathbf{e}_1 = \mathbf{e}_2$, notice that $\mathbf{B}^T(\mathbf{s}_1 + \mathbf{s}_2) = \mathbf{e}_1 + \mathbf{e}_2$, which implies that $\mathsf{Ham\text{-}Wt}([\mathbf{A}|\mathbf{AC}_1 + \mathbf{G}]^T\ (\mathbf{s}_1{+}\mathbf{s}_2)) \leq 2m/(25\sqrt{n})$ (recall, the hamming weight of $\mathbf{e}_1{+}\mathbf{e}_2$ is at most $2m/(25\sqrt{n})$).

This implies, in particular, $\mathsf{Ham\text{-}Wt}(\mathbf{A}^T(\mathbf{s}_1 + \mathbf{s}_2)) \leq 2m/(25\sqrt{n})$. Since each row of $\mathbf{C}_1$ and $\mathbf{C}_2$ has hamming weight at most $m/(50\sqrt{n})$, $\mathsf{Ham\text{-}Wt}((\mathbf{AC}_1)^T(\mathbf{s}_1{+}\mathbf{s}_2)) \leq m^2/(625n) < n$. As a result, $\mathsf{Ham\text{-}Wt}([\mathbf{A}|\mathbf{AC}_1]^T\ (\mathbf{s}_1+$

$\mathbf{s}_2)) < 2n$. But if $\mathbf{s}_1 \neq \mathbf{s}_2$, then $\textsf{Ham-Wt}(\mathbf{G}^T(\mathbf{s}_1 + \mathbf{s}_2)) \geq 4n$ which implies, using triangle inequality, that $\textsf{Ham-Wt}(\mathbf{B}^T(\mathbf{s}_1 + \mathbf{s}_2)) \geq 2n$. This brings us to a contradiction since $\textsf{Ham-Wt}(\mathbf{e}_1 + \mathbf{e}_2) \leq 2m/(25\sqrt{n}) < n$.

Next, given that $\mathbf{s}_1 = \mathbf{s}_2$, it follows that $\mathbf{w}^T\mathbf{s}_1 + 1 \neq \mathbf{w}^T\mathbf{s}_2$. This concludes our proof.

### 5.2.1 Computational Hiding

For computational hiding, at a high level, we will first switch $\mathbf{B}$ to a uniformly random matrix. This will follow from Knapsack LPN with low $(O(1/\sqrt{n}))$ noise. Next, we will switch $\mathbf{y}$ and $z$ to uniformly random vectors. This will use *LPN* with low $(O(1/\sqrt{n}))$ noise. At this point, the vector $z$ has no information about the committed bit $x$. We will now argue this formally via a sequence of hybrids.

- **Hybrid 0**: This corresponds to the real world.
- **Hybrid 1**: This hybrid is identical to the previous one, except that the challenger does not output the message for 'bad' matrix $\mathbf{C}$ and error vector $\mathbf{e}$.

  It chooses random message $b \leftarrow \{0,1\}$, random matrices $\mathbf{A} \leftarrow \mathbb{Z}_2^{n\times m}$, $\mathbf{w} \leftarrow \mathbb{Z}_2^n$, $\mathbf{C} \leftarrow \chi^{m\times m}$, sets $\mathbf{B} = [\mathbf{A} \mid \mathbf{AC} + \mathbf{G}]$. Next, it chooses secret vector $\mathbf{s} \leftarrow \mathbb{Z}_2^n$, error vector $\mathbf{e} \leftarrow \chi^m$ and sets $\mathbf{y} = \mathbf{B}^T\mathbf{s} + \mathbf{e}$, $z = \mathbf{w}^T\mathbf{s} + b$ and sends $(\mathbf{D}, \mathbf{B}, \mathbf{y}, z)$ to the adversary.
- **Hybrid 2**: In this hybrid, the challenger chooses $\mathbf{B}$ uniformly at random.

  It chooses random message $b \leftarrow \{0,1\}$, random matrices $\mathbf{w} \leftarrow \mathbb{Z}_2^n$, $\mathbf{B} \leftarrow \mathbb{Z}_2^{n\times 2m}$. Next, it chooses secret vector $\mathbf{s} \leftarrow \mathbb{Z}_2^n$, error vector $\mathbf{e} \leftarrow \chi^{2m}$ and sets $\mathbf{y} = \mathbf{B}^T\mathbf{s} + \mathbf{e}$, $z = \mathbf{w}^T\mathbf{s} + b$ and sends $(\mathbf{B}, \mathbf{w}, \mathbf{y}, z)$ to the adversary.
- **Hybrid 3**: In this hybrid, the challenger chooses $\mathbf{y}$ and $z$ uniformly at random.

  It chooses random message $b \leftarrow \{0,1\}$, random matrices $\mathbf{w} \leftarrow \mathbb{Z}_2^n$, $\mathbf{B} \leftarrow \mathbb{Z}_2^{n\times 2m}$. Next, it chooses $\mathbf{y} \leftarrow \mathbb{Z}_2^m$, $z \leftarrow \mathbb{Z}_2^n$ and sends $(\mathbf{B}, \mathbf{w}, \mathbf{y}, z)$ to the adversary.

Let $\textsf{Adv}_i^{\mathcal{A}}$ denote the advantage of adversary $\mathcal{A}$ in Hybrid $i$. We will now show that for all $i \in \{0,1,2\}$, $\textsf{Adv}_i^{\mathcal{A}} - \textsf{Adv}_{i+1}^{\mathcal{A}}$ is negligible in the security parameter $n$.

**Claim 5.4.** For any adversary $\mathcal{A}$, $\textsf{Adv}_0^{\mathcal{A}} - \textsf{Adv}_1^{\mathcal{A}} \leq \text{negl}(n)$.

*Proof.* The only difference between Hybrid 0 and Hybrid 1 is that the challenger, in hybrid 0, outputs the message in clear if either the error vector $\mathbf{e}$ or some row of $\mathbf{C}$ has hamming weight greater than $m/(50\sqrt{n})$. Since $\mathbf{e} \leftarrow \textsf{Ber}_\beta^m$ and $\beta = 1/(100\sqrt{n})$, using Chernoff bounds, $\Pr[\textsf{Ham-Wt}(\mathbf{e}) > m/(50\sqrt{n})] \leq \text{negl}(n)$. We can use a similar argument for the rows of $\mathbf{C}$. ∎

**Claim 5.5.** Assuming the Knapsack Learning Parity with Noise assumption for $\beta = 1/(100\sqrt{n})$, for any PPT adversary $\mathcal{A}$, $\textsf{Adv}_1^{\mathcal{A}} - \textsf{Adv}_2^{\mathcal{A}} \leq \text{negl}(n)$.

*Proof.* Suppose there exists a PPT adversary $\mathcal{A}$ such that $\textsf{Adv}_1^{\mathcal{A}} - \textsf{Adv}_2^{\mathcal{A}} = \epsilon$. We will construct a reduction algorithm $\mathcal{B}$ that breaks the knapsack LPN assumption with advantage $\epsilon$. The reduction algorithm $\mathcal{B}$ receives matrices $\mathbf{X} \in \mathbb{Z}_2^{n\times m}$, $\mathbf{Y} \in \mathbb{Z}_2^{n\times m}$ where $\mathbf{Y}$ is either a uniformly random matrix, or $\mathbf{Y} = \mathbf{XZ}$ for some matrix $\mathbf{Z} \leftarrow \textsf{Ber}_\beta^{m\times m}$. It sets $\mathbf{A} = \mathbf{X}$, $\mathbf{B} = [\mathbf{A} \mid \mathbf{Y} + \mathbf{G}]$, chooses $b \leftarrow \{0,1\}$, $\mathbf{w} \leftarrow \mathbb{Z}_2^n$, $\mathbf{s} \leftarrow \mathbb{Z}_2^n$, $\mathbf{e} \leftarrow \textsf{Ber}_\beta^m$ and sets $\mathbf{y} = \mathbf{B}^T\mathbf{s} + \mathbf{e}$ and $z = \mathbf{w}^T\mathbf{s} + b$. It sends $(\mathbf{B}, \mathbf{w}, \mathbf{y}, z)$ to the adversary and receives bit $b'$ in return. If $b = b'$, the reduction algorithm guesses that $\mathbf{Y} = \mathbf{XZ}$, else it guesses that $\mathbf{Y}$ is uniformly random.

The algorithm $\mathcal{B}$ thus breaks the Knapsack LPN assumption with advantage $\epsilon$. ∎

**Claim 5.6.** Assuming the Extended Learning Parity with Noise assumption for $\beta = 1/(100\sqrt{n})$, for any PPT adversary $\mathcal{A}$, $\textsf{Adv}_2^{\mathcal{A}} - \textsf{Adv}_3^{\mathcal{A}} \leq \text{negl}(n)$.

*Proof.* Suppose there exists a PPT adversary $\mathcal{A}$ such that $\mathsf{Adv}_2^{\mathcal{A}} - \mathsf{Adv}_3^{\mathcal{A}} = \epsilon$. We will construct a reduction algorithm $\mathcal{B}$ that breaks the LPN assumption with advantage $\epsilon$. The reduction algorithm $\mathcal{B}$ receives matrices $\mathbf{X} \in \mathbb{Z}_2^{n\times m}$, matrix $\mathbf{Y} \in \mathbb{Z}_2^{n\times m}$, vector $\mathbf{c} = \mathbf{Y}^T\mathbf{s}$ and $\mathbf{a}$, where $\mathbf{a}$ is either a uniformly random vector, or $\mathbf{a} = \mathbf{X}^T\mathbf{s} + \mathbf{e}$ for some matrix $\mathbf{e} \leftarrow \mathsf{Ber}_\beta^m$. It sets $\mathbf{B} = \mathbf{X}$, $\mathbf{w} = \mathbf{Y}$, chooses $b \leftarrow \{0,1\}$ and sets $z = \mathbf{y} + b$. It sends $(\mathbf{B}, \mathbf{w}, \mathbf{y}, z)$ to the adversary and receives bit $b'$ in return. If $b = b'$, the reduction algorithm guesses that $\mathbf{a} = \mathbf{X}^T\mathbf{s} + \mathbf{e}$, else it guesses that $\mathbf{a}$ is uniformly random.

The algorithm $\mathcal{B}$ thus breaks the Extended LPN assumption with advantage $\epsilon$. ■

Finally, note that any adversary $\mathcal{A}$ has advantage 0 in Hybrid 3. This concludes our proof.

## 5.3 Construction from Learning Parity with Constant Noise

For this construction, we will require a polynomial time algorithm $\mathsf{GenECC}$ that generates 'robust' error correcting code generator matrices. More formally, $\mathsf{GenECC}(1^n)$ takes as input a parameter $n$ and outputs $\ell$ matrices $\mathbf{G}_1, \ldots, \mathbf{G}_\ell$ of dimension $n \times m$ such that the following property holds: for every matrix $\mathbf{A} \in \mathbb{Z}_2^{n\times m}$, there exists an $i \in [\ell]$ such that every non-zero vector in the rowspace of $\mathbf{A} + \mathbf{G}_i$ has hamming weight at least $m/3$.

- $\mathsf{Commit}(1^n, x \in \{0,1\})$ : The commitment algorithm first computes $(\mathbf{G}_1, \ldots, \mathbf{G}_\ell) \leftarrow \mathsf{GenECC}(1^n)$, where $\mathbf{G}_i \in \mathbb{Z}_2^{n\times m}$. Next, it chooses $\mathbf{A} \leftarrow \mathbb{Z}_2^{n\times m}$ and sets $\mathbf{D}_i = [\mathbf{A} + \mathbf{G}_i]$. It chooses secret vectors $\mathbf{s}_i \leftarrow \mathbb{Z}_2^n$ and error vectors $\mathbf{e}_i \leftarrow \chi^m$ for $i \le \ell$. If any of the error vectors have hamming weight greater than $2m\beta$, then the algorithm outputs $x$ in the clear. Else, it chooses $\mathbf{w} \leftarrow \mathbb{Z}_2^n$, sets $\mathbf{y}_i \leftarrow \mathbf{s}_i^T\mathbf{D}_i + \mathbf{e}_i$ for $i \in [\ell]$, $z = \mathbf{s}_i^T\mathbf{w} + x$ and outputs $\mathsf{com} = (\mathbf{A}, \mathbf{B}\{\mathbf{y}_i, \mathbf{z}_i\})$ as the commitment.
- $\mathsf{Verify}(\mathsf{com}, x, (\mathbf{s}, \{\mathbf{e}_i\}))$ : Let $\mathsf{com} = (\mathbf{A}, \mathbf{B}, \{\mathbf{y}_i\}, z)$. The verification algorithm first checks that $\mathbf{y}_i = \mathbf{s}_i^T[\mathbf{A} + \mathbf{G}_i] + \mathbf{e}_i$ for all $i \in [\ell]$ and $\mathbf{z}_i = \mathbf{s}^T\mathbf{w} + x$.

**Perfect Correctness** : Perfect correctness will crucially use the robustness property of $\mathsf{GenECC}$ algorithm. Suppose there exist two vectors $\mathbf{s}_1, \mathbf{s}_2$ and error vectors $\{\mathbf{e}_i^1\}$ and $\{\mathbf{e}_i^2\}$ such that $\mathbf{s}^T\mathbf{D}_i + \mathbf{e}_i^1 = \mathbf{s}^T\mathbf{D}_i + \mathbf{e}_i^2$. Then, for all $i \le \ell$, $(\mathbf{s}_1 + \mathbf{s}_2)^T\mathbf{D}_i$ has hamming weight at most $m\beta$. But by the robustness property, for every $\mathbf{A} \in \mathbb{Z}_2^{n\times m}$, there exists at least one index $i \in [\ell]$ such that the row space of $\mathbf{A} + \mathbf{G}_i$ has hamming weight at least $m/2$. This brings us to a contradiction.

**Computational Hiding Proof Sketch** The proof is fairly simple, and follows from the LPN assumption. First we introduce $\ell$ hybrid experiments, where in the $i^{th}$ experiment, $(\mathbf{y_j}, \mathbf{z_j})$ are random for all $j \le i$. The remaining $(\mathbf{y}_j, \mathbf{z}_j)$ components are same as in the actual construction. The only difference between the $(i-1)^{th}$ and $i^{th}$ hybrid is the distribution of $(\mathbf{y}_i, \mathbf{z}_i)$.

**Hybrid** $\mathsf{Hybrid}_i$ In this experiment, the challenger chooses a matrix $\mathbf{A} \leftarrow \mathbb{Z}_2^{n\times m}$, vector $\mathbf{w} \leftarrow \mathbb{Z}_2^n$ and sets $\mathbf{D}_i = \mathbf{A} + \mathbf{G}_i$. Next, it chooses $\mathbf{s_j} \leftarrow \mathbb{Z}_2^n$ and $\mathbf{e}_j \leftarrow \mathsf{Ber}_\beta^m$ for all $j \le \ell$. For $j \le i$, it chooses $y_j \leftarrow \mathbb{Z}_2^m$ and $z_j \leftarrow \mathbb{Z}_2$. For $j > i$, it chooses the commitment bit $b \leftarrow \{0,1\}$, sets $\mathbf{y}_j = \mathbf{D}_i^T\mathbf{s}_j + \mathbf{e}_j$ and $z = \mathbf{w}^T\mathbf{s} + b$. It sends $(\mathbf{A}, \mathbf{w}, \{\mathbf{y}_i, z_i\}_i)$ to the adversary. The adversary outputs a bit $b'$ and wins if $b = b'$.

Suppose there exists an adversary $\mathcal{A}$ that can distinguish between these two hybrids. Then we can construct a reduction algorithm $\mathcal{B}$ that can break the extended LPN assumption. $\mathcal{B}$ receives $(\mathbf{X}, \mathbf{wy}, z)$ from the LPN challenger, where $\mathbf{y} = \mathbf{X}^T\mathbf{s} + \mathbf{e}$ or is random, and $z = \mathbf{w}^T\mathbf{s}$. It sets $\mathbf{A} = \mathbf{X} - \mathbf{G}_i$. The remaining components can be generated using $\mathbf{A}$ (note that there is a different $\mathbf{s}_i$ for each $i$, so the reduction algorithm does not need the LPN secret $\mathbf{s}$ to generate the remaining components). Depending on whether $\mathbf{y}$ is random or not, $\mathcal{B}$ either simulates Hybrid $i$ or Hybrid $i-1$.

# 6 Constrained PRFs for Admissible Hash Compatible Constraints

In this section, we will construct constrained PRFs for admissible hash compatible constraints under the $q$-DDH assumption. At a high level, our base PRF looks like the Naor-Reingold PRF [NR04]. The PRF key consists of $2n$ integers and a random group generator $g$. The PRF evaluation on an $n$ bit strings is performed as follows: first choose $n$ out of the $2n$ integers depending on the input, compute their product and then output this product in the exponent of $g$.

- $\mathsf{Setup}(1^\lambda)$: The setup algorithm takes as input the security parameter $\lambda$. It first generates a group of prime order as $(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$, where $p$ is a prime, $\mathbb{G}$ is a group of order $p$ and $g$ is a random generator. Next, it chooses $2n$ integers $c_{i,b} \leftarrow \mathbb{Z}_p^*$ for $i \leq n$, $b \in \{0,1\}$. It sets the master PRF key as $K = \big((p, \mathbb{G}, g), \{c_{i,b}\}_{i \leq n, b \in \{0,1\}}\big)$.
- $\mathsf{Constrain}(K, u \in \{0,1,\perp\}^n)$: The constrain algorithm takes as input the master PRF key $K = ((p, \mathbb{G}, g), \{c_{i,b}\}_{i,b})$ and constraint $u \in \{0,1,\perp\}^n$. It first chooses an integer $a \in \mathbb{Z}_p^*$ and computes, for all $i \leq n$, $b \in \{0,1\}$,
$$v_{i,b} = \begin{cases} c_{i,b}/a & \text{if } u_i = b \vee u_i = \perp \\ c_{i,b} & \text{otherwise.} \end{cases}$$
It sets the constrained key as $K_u = \Big((p, \mathbb{G}, g), u, \{g, g^a, g^{a^2}, \ldots, g^{a^{n-1}}\}, \{v_{i,b}\}_{i,b}\Big)$.
- $\mathsf{Evaluate}(K, x \in \{0,1\}^n)$: The evaluation algorithm takes as input a PRF key $K$ (which could be either the master PRF key or constrained PRF key) and an input string $x \in \{0,1\}^n$.

  If $K$ is a master PRF key, then it can be parsed as $K = ((p, \mathbb{G}, g), \{c_{i,b}\}_{i,b})$. The evaluation algorithm computes $t = \prod_{i \leq n} c_{i,x_i}$ and outputs $g^t$.

  If $K$ is a constrained key, then it consists of the group description $(p, \mathbb{G}, g)$, constraint $u \in \{0,1,\perp\}^n$, group elements $(g_0, g_1, \ldots, g_{n-1})$ and $2n$ integers $\{v_{i,b}\}_{i,b}$. The evaluation algorithm first checks if $P_u(x) = 0$. If not, it outputs $\perp$. Else, it computes the product $v = \prod_{i \leq n} v_{i,x_i}$. Next, it counts the number of positions $s$ such that $u_i = x_i \vee u_i = \perp$. It outputs the evaluation as $g_s^v$ (note that since $P_u(x) = 0$, $0 \leq s < n$, and therefore the output is well defined).

**Correctness** We need to show that for any PRF key $K$, any constraint $u \in \{0,1,\perp\}^n$, any key $K_u$ constrained at $u$ and any input $x \in \{0,1\}^n$ such that $P_u(x) = 0$, evaluation at $x$ using the master PRF key $K$ matches the evaluation at $x$ using the constrained key $K_u$.

More formally, let $K \leftarrow \mathsf{Setup}(1^n)$, and let $K = ((p, \mathbb{G}, g), \{c_{i,b}\})$. Let $u \in \{0,1,\perp\}^n$ be any constraint, and let $K_u = ((p, \mathbb{G}, g), u, \{g, g^a, \ldots, g^{a^{n-1}}\}, \{v_{i,b}\})$ be the constrained key. On input $x \in \{0,1\}^n$, the PRF evaluation using the master PRF key computes $t = \prod c_{i,x_i}$ and outputs $h = g^t$.

Let $S = \{i : u_i = x_i \vee u_i = \perp\}$, and let $s = |S|$. Since $P_u(x) = 0$, it follows that $s < n$ (since there is at least one index where $u_i \neq \perp \wedge x_i \neq u_i$). For all $i \in S$, $v_{i,x_i}$ is set to be $c_{i,x_i}/a$, and for all $i \notin S$, $v_{i,x_i} = c_{i,x_i}$. As a result, $v = \prod_i v_{i,x_i} = (\prod_i c_{i,x_i})/a^s$. Therefore, $(g^{a^s})^v = g^t = h$, which is equal to the master key evaluation.

## 6.1 Security

We will now show that the construction described above is secure as per Definition 2.7. Recall, in the single-key no-query security game, the adversary is allowed to query for a single constrained key, after which the adversary must output a challenge point not in the constrained set and then distinguish between the PRF evaluation at the challenge point and a truly random string. We will show that such an adversary can be used to break the $q$-DDH assumption. The reduction algorithm receives as challenge $(g, g^a, g^{a^2}, \ldots, g^{a^{n-1}})$ and $T$, where $T = g^{a^n}$ or a uniformly random group element. The reduction algorithm then receives a constrained key query $u$ from the adversary. The reduction algorithm chooses $2n$ random integers and sends them along with $(g,\, g^a,\, \ldots,\, g^{a^{n-1}})$. Now, the adversary sends a point $x$ such that $P_u(x) = 1$. The reduction algorithm will use $T$ to respond to the adversary. The crucial point here is that the reduction does not need to know $a$ to construct this response.

**Lemma 6.1.** Assuming the $q$-$\mathsf{DDH}$ assumption, for any adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{CPRF}}(n) \leq \mathrm{negl}(n)$.

*Proof.* Suppose there exists an adversary $\mathcal{A}$ such that $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{CPRF}}(n) = \epsilon$. We will use $\mathcal{A}$ to construct a reduction algorithm $\mathcal{B}$ that breaks the $q$-$\mathsf{DDH}$ assumption. The reduction algorithm receives the group description $(p, \mathbb{G}, g)$, $n$ group elements $(g_0,\, g_1,\, \ldots,\, g_{n-1})$ and the challenge term $T$ from the challenger. It then chooses $2n$ random integers $v_{i,b} \leftarrow \mathbb{Z}_p^*$ for all $i \leq n, b \in \{0,1\}$. It receives constrained key query $u$ from $\mathcal{A}$, and sends $((p, \mathbb{G}, g), u, \{g_0, \ldots, g_{n-1}\}, \{v_{i,b}\}_{i,b})$ to $\mathcal{A}$. Next, it receives the challenge input $x \in \{0,1\}^n$ from $\mathcal{A}$ such that $P_u(x) = 1$. The reduction algorithm computes $v = \prod_i v_{i,x_i}$ and sends $T^v$ to the adversary. If $\mathcal{A}$ guesses that the challenge string is random, then $\mathcal{B}$ guesses that $T$ is random, else it guesses that $T = g^{a^n}$, where $g_i = g^{a^i}$.

We now need to argue that $\mathcal{B}$ perfectly simulates the single-key no-query constrained PRF game. First, let us consider the case when $g_i = g^{a^i}$ and $T = g^{a^n}$. The constrained key is distributed as in the actual security game. The reduction algorithm implicitly sets $c_{i,b} = v_{i,b}a$ for all $i, b$ such that $u_i = b \vee u_i = \perp$. On challenge input $x$ such that $P_u(x) = 1$, let $v = \prod_i v_{i,x_i}$. Note that $t = \prod_i c_{i,x_i} = a^n v$. As a result, its outputs $T^v = g^t$ is the correct PRF evaluation at $x$.

Now, suppose $T$ is a uniformly random group element. Then, once again, the constrained key's distribution is identical to the real security game distribution, and the response to PRF challenge is a uniformly random group element. This implies that $\mathcal{B}$ can break the $q$-$\mathsf{DDH}$ assumption with advantage $\epsilon$. ■

# Acknowledgements

We give a large thanks to David Zuckerman for helpful discussions regarding the error correcting code described in Section 5.3.

# References

[ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *CRYPTO*, pages 595–618, 2009.

[BB04] Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In *CRYPTO*, pages 443–459, 2004.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In *PKC*, pages 501–519, 2014.

[BLP+13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 575–584, 2013.

[BOV07] Boaz Barak, Shien Jin Ong, and Salil Vadhan. Derandomization in cryptography. *SIAM J. Comput.*, 2007.

[BP15] Nir Bitansky and Omer Paneth. Zaps and non-interactive witness indistinguishability from indistinguishability obfuscation. In *Theory of Cryptography*, pages 401–427. Springer Berlin Heidelberg, 2015.

[BV15] Zvika Brakerski and Vinod Vaikuntanathan. Constrained key-homomorphic prfs from standard lattice assumptions - or: How to secretly embed a circuit in your PRF. In *Theory of Cryptography*

*- 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, pages 1–30, 2015.

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In *ASIACRYPT*, pages 280–300, 2013.

[BZ14] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In *CRYPTO*, 2014.

[CM14] Melissa Chase and Sarah Meiklejohn. Déjà q: Using dual systems to revisit q-type assumptions. In *EUROCRYPT*, pages 622–639, 2014.

[DN00] Cynthia Dwork and Moni Naor. Zaps and their applications. In *FOCS*, pages 283–293, 2000.

[Dod02] Yevgeniy Dodis. Efficient construction of (distributed) verifiable random functions. In YvoG. Desmedt, editor, *Public Key Cryptography PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 1–17. Springer Berlin Heidelberg, 2002.

[DY05] Yevgeniy Dodis and Aleksandr Yampolskiy. A verifiable random function with short proofs and keys. In *Proceedings of the 8th International Conference on Theory and Practice in Public Key Cryptography*, PKC'05, pages 416–431, Berlin, Heidelberg, 2005. Springer-Verlag.

[FHPS13] Eduarda S. V. Freire, Dennis Hofheinz, Kenneth G. Paterson, and Christoph Striecks. Programmable hash functions in the multilinear setting. In *CRYPTO*, pages 513–530, 2013.

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *STOC*, pages 416–426, 1990.

[GGH$^+$13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, 2013.

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions (extended abstract). In *FOCS*, pages 464–479, 1984.

[GOS12] Jens Groth, Rafail Ostrovsky, and Amit Sahai. New techniques for noninteractive zero-knowledge. *J. ACM*, 59(3):11, 2012.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.

[HJ16] Dennis Hofheinz and Tibor Jager. Verifiable random functions from standard assumptions. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*, pages 336–362, 2016.

[HSW14] Susan Hohenberger, Amit Sahai, and Brent Waters. Replacing a random oracle: Full domain hash from indistinguishability obfuscation. In *EUROCRYPT*, pages 201–220, 2014.

[HW10] Susan Hohenberger and Brent Waters. Constructing verifiable random functions with large input spaces. In *EUROCRYPT*, pages 656–672, 2010.

[Jag15] Tibor Jager. Verifiable random functions from weaker assumptions. In Yevgeniy Dodis and JesperBuus Nielsen, editors, *Theory of Cryptography*, volume 9015 of *Lecture Notes in Computer Science*, pages 121–143. Springer Berlin Heidelberg, 2015.

[JKPT12] Abhishek Jain, Stephan Krenn, Krzysztof Pietrzak, and Aris Tentes. Commitments and efficient zero-knowledge proofs from learning parity with noise. In *Proceedings of the 18th International Conference on The Theory and Application of Cryptology and Information Security*, ASIACRYPT'12, pages 663–680, Berlin, Heidelberg, 2012. Springer-Verlag.

[KPTZ13] Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In *ACM Conference on Computer and Communications Security*, pages 669–684, 2013.

[Lys02] Anna Lysyanskaya. Unique signatures and verifiable random functions from the dh-ddh separation. In *Proceedings of the 22Nd Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '02, pages 597–612, London, UK, UK, 2002. Springer-Verlag.

[MM11] Daniele Micciancio and Petros Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, pages 465–484, 2011.

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

[MRV99] Silvio Micali, Michael Rabin, and Salil Vadhan. Verifiable random functions. In *In Proc. 40th IEEE Symposium on Foundations of Computer Science (FOCS*, pages 120–130. IEEE, 1999.

[Nao91] Moni Naor. Bit commitment using pseudorandomness. *J. Cryptology*, 4(2):151–158, 1991.

[NR04] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004.

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 333–342, 2009.

[Pei15] Chris Peikert. A decade of lattice cryptography. Cryptology ePrint Archive, Report 2015/939, 2015. http://eprint.iacr.org/.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 84–93, 2005.

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *STOC*, pages 475–484, 2014.

[Wat05] Brent Waters. Efficient identity-based encryption without random oracles. In *EUROCRYPT*, pages 114–127, 2005.