

Efficient Unbiasable Verifiable Random Functions via (Non-)Black-Box Reductions

Nicholas Brandt *

[[alloc] init]

Zurich, Switzerland

`crypto@nicholasbrandt.de`

We present conceptually simple and practically efficient constructions of a verifiable random function (VRF) that fulfills strong notions of unbiasability recently introduced by Giunta and Stewart (Eurocrypt 2024). Unbiasability is a crucial property for many applications of VRFs such as leader election in proof-of-stake blockchains. The only previous constructions provide heuristic security in the random oracle model or rely on the decisional Diffie–Hellman assumption with preprocessing which incurs significant overhead. In contrast, our compiler turns any VRF into an unbiasable VRF in the standard model (without any setup) using only symmetric-key primitives; and thus only with negligible computational overhead.

Underlying our approach, we provide a generic black-box reduction for unbiasability of VRFs that incurs subexponential loss. Additionally, under a mild (verifiable) requirement on the underlying VRF we also show a novel non-black-box reduction that incurs only polynomial loss.

This non-black-box technique allows us to replace the folklore “VUF-then-Hash” heuristic with a provably secure “Hash-then-VRF” construction at virtually no cost. In particular, if the underlying VRF hashes its input, our transformation requires only a single evaluation of a hardcore function, e.g. parity of the input (plus a PRF evaluation for full security).

Contents

1	Introduction	2
1.1	Contributions	4
1.2	Related Work	6
2	Technical Overview	7
3	Avoiding Superpolynomial Security Loss	11
4	Practical Construction with Minimal Overhead	12
5	Preliminaries	13
5.1	Cryptographic Primitives	13
5.2	Computational Problems	18
6	Generic Construction of Unbiasable VRFs	19
6.1	Warm-Up: Unbiasability o.I.P.	19
6.2	Full Unbiasability	22
7	Avoiding Superpolynomial Hardness Assumptions via Non-Black-Box Reductions	24
8	Definitional Issues of Unbiasability in [GS24]	28
9	Additional Properties of Our Construction	31
9.1	Practical Considerations	37
10	Separating Full Unbiasability from Unbiasability o.I.P. via Invertibility	40
11	Symmetric VRFs from Injective OWFs and VRFs	41
12	Simplified Corollaries	42
13	Towards a Lattice-Based VRF without Setup	45

* This work was done while the author was at ETH Zurich.

1 Introduction

Verifiable Random Functions [MRV99] (VRF) are the public-key analog of (secret-key) pseudorandom functions [GGM86] (PRF), i.e., VRFs have a secret key sk and a verification key vk . The evaluator of the function (who holds the secret key) can generate a proof that allows (together with the verification key) to check whether the output was indeed computed correctly. Traditionally, VRFs satisfy three properties: correctness, pseudorandomness and unique provability. Correctness and pseudorandomness are inherited from PRFs. They state, informally, that an honestly generated image and proof will verify, and that for any input the output (without knowledge of the secret key or the proof) is indistinguishable from a random string. Unique provability, on the other hand, stands out as it ensures that for each preimage at most a single image can be proven to be correct—even for maliciously generated verification keys. A weaker variant is a verifiable unpredictable function (VUF) that requires unpredictability instead of pseudorandomness.

These properties make VRFs attractive (or even the only viable solution) for applications where there is no trusted setup. Indeed, VRFs have found numerous interesting and important applications, including: signatures with very strong verifiability guarantees [GO93], resettable zero-knowledge proofs [MR01], lottery systems [MR02; LBM20], transaction escrow schemes [JS04], updatable zero-knowledge databases [Lis05], e-cash systems [ASM07; BCK⁺09], and Proof-of-Stake blockchains [CM17; KRD⁺17; DGK⁺18; BCA⁺25].

However, some applications such as leader election (e.g. in Proof-of-Stake blockchains) use VRFs in a way that is not justified by the three properties above. The goal of a leader election protocol is to elect a designated party as the leader such that each party is equally likely to become the leader; yet the leader is fixed before the result of the election is revealed. In a simplified leader election procedure, each (possibly malicious) party P would publish a verification key vk_P . Then each party would evaluate their VRF on a common random input x to obtain their own output y_P and proof π_P . Each party publishes their result y_P and proof π_P , and the party with the lowest output is elected as the leader. Obviously, it is crucial that the VRF guarantees some security against maliciously generated keys (because malicious parties generate them). Now, unique provability guarantees that even for maliciously generated verification keys, the VRF output (for the common preimage x) is unique. However, this guarantee is not strong enough to protect the overall leader election protocol from attacks. Concretely, suppose a malicious party comes up with the following degenerate VRF, if the verification key $vk = 0^\lambda$ is the all zero string, then the VRF output is always 0. For $vk = 0^\lambda$ and any preimage x the image 0 always verifies, while no other image verifies. This VRF fulfills all three properties, but is clearly biased. A malicious party $\hat{P}$ can simply publish their degenerate verification key $vk_{\hat{P}} = 0^\lambda$ and always win the election.

Unbiasability. Thus, a stronger security notion is required for such applications. What we need for applications like leader election is that, intuitively, even if the verification key is maliciously generated, the output (on a random input) should not be too biased.¹ Indeed, several notions of “unbiasability” have been proposed. David, Gazi, Kiayias, and Russell [DGK⁺18] are the first to identify this issue and propose an unbiasedness notion in the Universal Composability framework [Can01]. Their notion is sufficiently strong to securely realize applications like leader election, but it provably cannot be instantiated in the (UC) plain model, i.e., without relying on a trusted setup.² However, given a trusted setup there exists a trivial unbiased solution; simply attach a non-interactive zero-knowledge proof (NIZK) that the output of a pseudorandom permutation was computed correctly.

Later, Esgin, Kuchta, Sakzad, Steinfeld, Zhang, Sun, and Chu [EKS⁺21] introduce a game-based notion of unbiasedness that can be instantiated in the plain model. As Giunta and Stewart [GS24] explain in their technical overview there are some issues with the definition of [EKS⁺21]. Aside from those

¹ In our example towards low values.

² Their VRF functionality suffices to realize a commitment functionality which does not exist in the plain model [CF01].

issues, their definition assumes a trusted setup, and thus their constructions only work in such a trusted setup model. Hence, Giunta and Stewart [GS24] propose two new game-based notions of unbiasedness without any setup. The first notion is called *unbiasedness on independent points (o.i.p.)* and the second notion is called *full unbiasedness* (see Figure 4). At a high level, unbiasedness o.i.p. guarantees that for any adversarially generated verification keys (valid) images for independently uniform preimages are indistinguishable from truly uniform images. A bit more formally, an adversary produces verification keys $\mathbf{vk}_1, \dots, \mathbf{vk}_{a(\lambda)}$ and a “monotone” predicate ρ .³ Then for each $\mathbf{vk}_i$ a list of preimages $(x_{i,1}, \dots, x_{i,b(\lambda)})$ is sampled independently at random. Then the adversary produces images $(y_{1,1}, \dots, y_{a(\lambda),b(\lambda)})$ and proofs $(\pi_{1,1}, \dots, \pi_{a(\lambda),b(\lambda)})$. Now, the predicate is evaluated either on the images $(y_{1,1}, \dots, y_{a(\lambda),b(\lambda)})$, or on independently uniform images. Unbiasedness o.i.p. guarantees that no adversary can bias the outcome of the predicate.

Full unbiasedness differs in that only a single list of preimages $(x_1, \dots, x_{b(\lambda)})$ is sampled independently at random. Now, for each $\mathbf{vk}_i$ the adversary produces images $y_{i,j}$ and proofs $\pi_{i,j}$ such that x_j and $y_{i,j}$ verify against $\mathbf{vk}_i$. This notion is stronger as it also rules out correlations of VRF images under adversarially generated verification keys. In this work we consider both notions; we give constructions for both and in Section 10 we show that unbiasedness o.i.p. is strictly weaker than full unbiasedness—separated by the natural class of invertible VRFs.

Random oracle model. Giunta and Stewart also show that the folklore construction of hashing the output of a verifiable unpredictable function is an unbiased VRF in the random oracle model. While this construction (henceforth VUF-then-hash) is extremely efficient, its unbiasedness proof heavily relies on the fact that the hash function is modeled as truly random. Thus, if instantiated with any concrete hash function the unbiasedness of the construction can only be argued heuristically. In this context we want to highlight recent attacks⁴ on the Fiat–Shamir transform [KRS25] (another random oracle heuristic) showcasing the need for provable security in the real world (i.e. plain model without setup). In Section 9.1 we discuss the folklore VUF-then-Hash construction in more detail and provide an alternative—even more efficient—“Hash-then-VRF” that we prove to be unbiased in the standard model.

Remark 1. From a theoretical point of view VRFs are trivial to instantiate in the random oracle model. VRFs can be easily instantiated by committing to a PRF key inside the VRF verification key, and providing a Fiat–Shamir-based NIZK to the evaluation of the VRF relative to the committed PRF key. However, without a random oracle or trusted setup, this approach fails; this is the reason why VRFs in the standard model are particularly difficult to construct.

Standard model. In this work we focus on the setting without any trusted setup or random oracle. Indeed, Giunta and Stewart also provide a construction of an unbiased VRF in the plain model (without a random oracle or setup). Concretely, they provide a compiler that transforms any VRF into an unbiased VRF based on the hardness of the decisional Diffie–Hellman (DDH) problem with preprocessing. We note that their compiler is not generic, i.e., it is not clear how to replace the DDH assumption with a different (plausibly post-quantum) assumption. The high-level reason is that for their approach Giunta and Stewart need a function with very special properties (they use an ElGamal commitment). As such, their construction is broken by quantum adversaries. Given the importance of unbiased VRFs in various applications, we consider the following question posed by Giunta and Stewart [GS24]:

Can fully unbiased VRFs be based on generic assumptions in the standard model—optimally ones that have plausibly post-quantum instantiations?

³ Monotonicity is a technical property that prevents the adversary from winning trivially by not evaluating their VRF. See Definition 7 and Section 8 for a discussion.

⁴ These attacks do not directly apply to the folklore unbiasedness construction, but they serve as a warning that heuristically secure constructions may in fact be broken in the real world.

1.1 Contributions

We summarize our main contributions:

- We present a generic compiler (Theorem 1) that turns any VRF into a fully unbiased VRF (without a setup or random oracle) using a collision-resistant hash function (CRH) and an injective one-way function (inj. OWF). Our construction is conceptually simpler than the only previous construction of Giunta and Stewart [GS24] (which is broken by quantum adversaries). Because both CRH and inj. OWF are symmetric key primitives and thus considered plausible post-quantum secure, this resolves the open problem of a post-quantum compiler posed in [GS24] affirmatively. We describe the high-level idea and formal construction in Section 2 and give the detailed proof in Section 6.
- Using a novel non-black-box reduction (Theorem 2), we remove the need for subexponential hardness of the underlying assumptions if we assume a natural “latency” property of the underlying VRF that is fulfilled by all known VRF constructions. We describe the high-level idea in Section 3 and provide full details in Section 7.
- Leveraging our non-black-box approach, we present a simple “Hash-then-VRF” construction (Theorem 3), which we compare with the folklore “VUF-then-Hash” construction. We show that—compared to the folklore construction—our construction is provably unbiased and likely even more efficient for concrete parameters. Additionally, we underscore the need for provable unbiasedness by showing how a malicious VUF designer can introduce bias into the VUF-then-Hash construction even if the underlying VUF is perfectly unbiased. Again, we present the high-level idea in Section 4 and provide full details in Section 9.1.

The rest of this paper is organized as follows:

- Section 5 contains technical preliminaries.
- Section 6 contains the formal statements and proofs of our black-box reduction with subexp. loss.
- Section 7 contains the formal statements and proofs of our non-black-box reduction with poly. loss.
- Section 8 discusses several issues with the unbiasedness definition of Giunta and Stewart [GS24] and proposes an equivalent⁵ definition of unbiasedness that avoids these issues. Importantly, while we prove our results formally for the notion of Giunta and Stewart [GS24], we show that they also apply to our new definition.
- Section 9 discusses additional properties of our VRF. The output of our VRF has approximately the same min-entropy as the input even if the VRF key is chosen adversarially. Moreover, our VRF preserves properties like aggregability, key-homomorphism, and NC^1 computability of the underlying VRF.
- Section 10 provides a separation showing that full unbiasedness is strictly stronger than unbiasedness o.i.p.
- Section 11 presents the first symmetric unbiased VRF, where $\text{VRF.Eval}(\text{sk}, x) = \text{VRF.Eval}(x, \text{sk})$ for all sk and x .
- Section 13 discusses how all ingredients of our compiler (from VRF to unbiased VRF) can be instantiated from lattice-based assumptions.

Generic black-box construction. Our first main contribution is a novel simple construction of an unbiased VRF based on generic primitives—answering the above question affirmatively.

Theorem 1 (Unbiased VRF from Subexponential Hardness). Assuming (non-uniform) subexponential security of a VRF, a keyed collision-resistant hash function (CRH), and an injective one-way function (OWF), the construction in Figure 3 is fully unbiased. Without the PRF $h' := h$ the construction is unbiased o.i.p.

⁵ This depends on the exact interpretation of the definition of [GS24]. We show equivalence of our notion with a best-effort interpretation assuming pseudorandomness of the VRF.

Our main construction and its security proofs are substantially simpler than the ones in [GS24]. The security proof of Theorem 1 is fully black-box. Our construction resolves a problem posed by Giunta and Stewart [GS24], namely lifting standard VRFs to unbiased VRFs using only symmetric-key primitives. Giunta and Stewart [GS24] resolve the “pseudorandom vs. unbiasedness” dilemma by relying on a preprocessing assumption. We overcome this “preprocessing” barrier described in [GS24] by using the composition of a OWF and a CRH to be able to (sufficiently) efficiently generate VRF proofs in our reduction.

Non-black-box construction. Our second contribution is a novel non-black-box technique tailored to unbiasedness that allows us to prove security based on polynomial hardness assumptions.

Theorem 2 (Unbiasable VRF from Polynomial Hardness). Assuming (non-uniform) polynomial security of an injective OWF and any latent VRF (cf. Definition 1), the construction in Figure 3 (without the CRH) is a fully unbiased VRF.

The security proof of Theorem 2 relies on a new non-black-box technique that requires a property of the underlying VRF that we call “latency”. Due to space constraints, we explain our non-black-box technique in detail in Section 7.

VRF latency. For both Theorems 2 and 3 we require the underlying VRF to be latent⁶ (cf. Definition 1). This intuitively means that for each verification key there exists a “pseudo” secret key that allows to compute the VRF relative to the given verification key. We stress that all known VRF constructions are latent. In particular, we show that the generic VRF construction of [Bit20; GHK⁺17b] is latent in Section 7.

Practical construction. We present a simple, highly efficient “Hash-then-VRF” construction in Figure 1 as a plug-in replacement for the VUF-then-Hash construction used in practice. In particular, if the underlying VRF hashes its input, our transformation simply xors the output of a hardcore function (e.g. parity of the input) with the output of the underlying VRF. Remarkably, we can prove unbiasedness o.i.p. for this construction in the standard model using our non-black-box technique. Full unbiasedness can be achieved at the cost of multiple hardcore evaluations and a PRF evaluation; these operations are more costly than a single hardcore evaluation, but are still negligible compared to the cost of the underlying VRF.

Theorem 3 (Practical Hash-then-VRF Construction). Assuming a (uniformly) collision-resistant, (non-uniformly) one-way, keyless CRH and a latent VRF, the construction of Figure 1 is pseudorandom against uniform adversaries and unbiased o.i.p. against non-uniform adversaries (both with polynomial security loss). With an additional PRF evaluation as in Figure 3, the construction is fully unbiased against non-uniform adversaries (with polynomial security loss).

Remark 2. For the practical construction in Theorem 3 we require the CRH to be keyless, i.e., the CRH is a hash function that is collision-resistant against uniform adversaries *as well as* one-way against non-uniform adversaries. This means that uniform adversaries cannot find collisions, while non-uniform adversaries cannot invert the hash function. This combination of properties is somewhat non-standard, but generally regarded as plausible for widely deployed hash functions like SHA-256. Note that keyless hash functions cannot be collision-resistant against non-uniform adversaries, since a non-uniform adversary can hardcode a collision.

⁶ The term “latent” alludes to the idea that for each verification key a pseudo secret key lies latent, i.e., it exists but remains hidden; only the (existential) non-black-box reduction can use it.

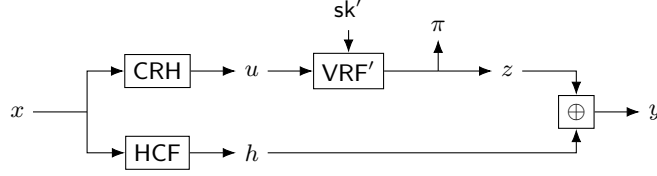


Fig. 1: Our recommended construction for practical applications where (non-uniform) unbiasedness o.i.p. and uniform pseudorandomness are required. Full unbiasedness can be achieved by an additional PRF evaluation as in Figure 3.

1.2 Related Work

Carnemolla, Catalano, Frasca, and Giunta [CCF⁺25]⁷ investigate the unbiasedness of several concrete VRFs. They find that “all known constructions are not natively unbiased.” Our compiler stands in stark contrast to their negative findings. That is, with minimal modifications (hashing the input and xoring a hardcore bit) we can make any VRF unbiased.

Regarding positive results they find that the VRF of Goyal, Hohenberger, Koppula, and Waters [GHK⁺17b] and Bitansky [Bit20] is weakly unbiased (in the sense of [GS24]) which is—however—insufficient for applications like leader election.

They also study the unbiasedness of the VRF of Dodis and Yampolskiy [DY05]. We emphasize that the unbiasedness of the DY-VRF is somewhat complicated. For polynomial-size input domains the DY-VRF is unconditionally unbiased o.i.p. but there is a trivial attack against full unbiasedness (even with a single preimage and two verification keys). For larger input domains, the DY-VRF is still unconditionally unbiased o.i.p. but the pseudorandomness security loss scales exponentially with the domain size. When hashing a large input to a small domain, neither an attack nor a proof of unbiasedness is known. Lastly, we point out that although Carnemolla, Catalano, Frasca, and Giunta [CCF⁺25] provide a proof of full unbiasedness of the DY-VRF in an idealized generic group model, this proof does not translate into the standard model. This is because unique provability requires the concrete group *instantiation* to be certified according to [HJ16], whereas a generic group *cannot* be certified—in other words, a generic group cannot be instantiated.

Dodis and Puniya [DP07] construct a verifiable random permutation⁸ (VRP) from any VRF. Their construction is a simple Feistel construction where the round function is a VRF. However, their VRP is not practically efficient. The reason is that Dodis and Puniya show that superlogarithmically many Feistel rounds are necessary and sufficient for pseudorandomness. Thus, even if the underlying VRF can be computed in NC^1 , the resulting VRP will only be in $\text{NC}^{2+o(1)} \setminus \text{NC}^2$. This is in contrast to our construction (and the one in [GS24]), which remains in NC^1 if the underlying primitives are in NC^1 . Moreover, we notice that although the VRP is a permutation, this only achieves unbiasedness o.i.p.—not full unbiasedness. Suppose that for some degenerate (VRF) key, the VRF round function is the zero function, then the VRP will be the identity. If for each key the preimages are uniform and independent, then the VRP images will be as well, hence the VRP is unbiased o.i.p. On the other hand, if the preimages across VRF instances are correlated (as they are in the full unbiasedness game), then so will the VRP images. For full unbiasedness, however, the images need to be indistinguishable from independent uniform images.

⁷ Presented at FC’26 but not published yet.

⁸ As a permutation, this construction is trivially *unconditionally* unbiased o.i.p.

Reference	Unbias.	Assumptions	Req. security	Setup
[DP07]	o.i.p.	VRF (generic)	poly	–
[GHK ⁺ 17b; Bit20]	$\exists$ weak	COM + CPRF + NIWI (generic)	poly	–
[GHK ⁺ 17b; Bit20]	$\exists$ weak	COM + NIWI (generic)	subexp	–
[EKS ⁺ 21]	?	Module-SIS/-LWE (lattice)	poly	RO
[Ler25b]	fully/?	OMIP _{2dim} (isogeny)	poly	RO/– ^(*)
[Lai24]	?	(square-)GA-DDH (isogeny)	poly	RO
[Mal24]	?	LWE (lattice)	poly	SRS
[GS24]	fully	DDH with preprocessing (group-based)	poly	–
Theorem 1	fully	inj. OWF + CRH + VRF (generic)	subexp	–
Theorem 2	fully	inj. OWF + latent VRF (generic)	poly	–
Theorem 3	o.i.p./fully	keyless CRH + latent VRF (generic)	poly	–
Theorem 17	fully	LWE + NIWI (lattice+generic)	subexp	–

Fig. 2: Some recent VRF constructions. RO stands for random oracle (note that RO-based constructions are trivially unbiased), SRS stands for structured random string (not uniform/transparent). We assume that the NIWI has perfect soundness. Inj. OWF stands for a keyless injective OWF. CRH stands for a collision-resistant hash function. VRF stands for an adaptively pseudorandom, verifiable random function (not necessarily unbiased). CPRF stands for a single-key, key-selectively, query-adaptively pseudorandom, constrained PRF. LWE stands for the learning-with-errors problem with arbitrary modulus. ^(*) Leroux formulates his VRF in the random oracle model, but it can be modified to not require a setup [Ler25a]; although it is based on a novel isogeny assumption. Though, its unbiasedness is only known in the RO model.

Lastly, we mention that all previously post-quantum secure (even biasable ones) VRFs [EKS⁺21; YAZ⁺19; LLH⁺23; Lai24; Gei24; Mal24] (except for [Ler25b]⁹) rely on the random oracle model or a trusted setup. In Figure 2 we summarize some recent VRF constructions and their properties.

2 Technical Overview

As previously mentioned, Giunta and Stewart [GS24] introduce the notion of (full) unbiasedness and present a construction of a fully unbiased VRF based on a VRF and the decisional Diffie–Hellman (DDH) assumption with preprocessing. From their construction we will gain some key insights that guide our own construction. Let us briefly recall their construction. The construction of [GS24] conceptually proceeds in three steps: From an arbitrary VRF they (generically) build a padded VRF

- whose preimages consist of two parts $x = (u, v)$,
- that is unbiased o.i.p., and
- that fulfills a restricted form of pseudorandomness where the first component of the challenge preimage is not queried before.

Then they preprocess the padded VRF’s input such that full adaptive pseudorandomness is achieved. Finally, they add another preprocessing step that lifts unbiasedness o.i.p. to full unbiasedness.

The padded VRF is defined quite simply as $\text{VRF}'.\text{Eval}(\text{sk}, x) := \text{VRF}.\text{Eval}(\text{sk}, u) \oplus v$. Pseudorandomness holds as long as the adversary does not query any preimage $x_i = (u_i, v_i)$ that has the same first component $u_i = u^*$ as the challenge preimage $x^* = (u^*, v^*)$. Also, note that unbiasedness o.i.p. holds because the construction perfectly *extracts randomness* from the second component v of the preimage x in the

⁹ The VRF of Leroux [Ler25b] can be modified to not require a setup [Ler25a] but it is based on a novel isogeny assumption.

unbiasability game. This means that no matter what function table $\text{VRF.Eval}(\text{sk}, \cdot)$ describes, the output on a uniformly random preimage x is uniformly random. Giunta and Stewart also note that any one-to-one VRF (i.e., a bijection) fulfills the restricted pseudorandomness requirement—in particular a two-round Feistel construction.¹⁰

In the second step, Giunta and Stewart compile the padded VRF into a fully unbiased VRF. For this to work they observe that they can prepend the two-round Feistel construction with a tailor-made (keyed) map $\phi_{[\mathbf{a}], \mathbf{b}}$ that fulfills two properties:

- It is a *certified* bijection (to preserve unbiasedability). It must be publicly and efficiently verifiable that the map $\phi_{[\mathbf{a}], \mathbf{b}}$ is a bijection for any key $([\mathbf{a}], \mathbf{b})$.
- It is collision-resistant in the first component (to ensure full pseudorandomness).

Their map is defined as

$$\phi_{[\mathbf{a}], \mathbf{b}}(z, w) := ([\mathbf{a}]^\top(z, w), [\mathbf{b}]^\top(z, w)) \quad (1)$$

where $[e]$ denotes the group element with exponent $e \in \mathbb{Z}_p$ in a cyclic group of order p .

This construction only fulfills unbiasedability o.i.p. Recall that in full unbiasedability the same preimages are evaluated relative to multiple verification keys. So far, the second component v of the preimage guaranteed unbiasedability o.i.p. but for full unbiasedability the same v would be xored onto multiple different images which means the images are not independent anymore. To resolve this issue, in their final step Giunta and Stewart add another preprocessing step where a PRF is evaluated on the verification key using part of the input as the PRF key $\mathbf{k}$. To retain injectivity they include a perfectly binding (ElGamal) commitment to $\mathbf{k}$, resulting in the final construction

$$\text{VRF}''.\text{Eval}(\text{sk}, x) := \text{VRF}'.\text{Eval}(\text{sk}, \cdot) \circ \phi_{[\mathbf{a}], \mathbf{b}} \circ \text{Enc}_{\text{vk}}(x) \quad (2)$$

where the preimage is interpreted as $x = (\mathbf{r}, \mathbf{k}, \text{pp}, \rho)$ and preprocessed by the encoding

$$\text{Enc}_{\text{vk}}(x) := (\mathbf{r}, \text{PRF.Eval}(\mathbf{k}, \text{vk}), \text{pp}, \text{COM.Eval}(\text{pp}, \mathbf{k}; \rho)) \quad (3)$$

For technical reasons (the use of a Chameleon hash function in the unbiasedability proof) the randomness $\mathbf{r}$ is included in the VRF image; it is not clear if this is necessary. Note that for injectivity the commitment must also be injective in the commitment randomness.

Key observations from [GS24]. We make three crucial observations w.r.t. the construction of [GS24] that will guide our construction.

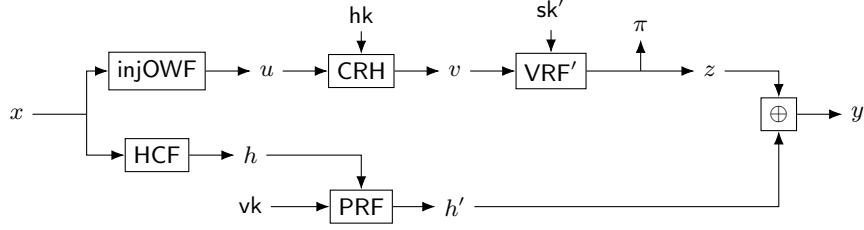
- To base pseudorandomness of the resulting VRF we need to make sure that (for uniformly random preimages) the underlying VRF is never evaluated on the same (internal) input.
- To achieve (even) unbiasedability o.i.p. the VRF evaluation needs to directly¹¹ extract some randomness from the preimage. The intuitive reason is that in the unbiasedability game only the preimages are uniformly random, whereas the verification keys are adversarially controlled. Hence, any value processed through an underlying VRF evaluation is possibly adversarially controlled, e.g. fixed to 0.
- For full unbiasedability the extracted randomness (from the preimage) must depend on the verification key. More concretely, for any verification key the resulting VRF must extract independent randomness from the preimage.

Our construction. Guided by these observations we present our simple constructions of unbiased VRFs based on generic primitives. Our construction requires any VRF, an injective OWF (which implies a PRF), and a collision-resistant hash function (CRH). We depict our construction in Figure 3 and succinctly describe it as

$$y := ((\text{VRF}'.\text{Eval}(\text{sk}, \cdot) \circ \text{CRH.Hash}(\text{hk}, \cdot) \circ \text{injOWF}) \oplus (\text{PRF.Eval}(\cdot, \text{vk}) \circ \text{HCF}))(x) .$$

¹⁰ Two rounds suffice here—in contrast to the result of Dodis and Puniya [DP07]—because only restricted pseudorandomness is required.

¹¹ “Directly” means that the extracted value does not invoke the evaluation of the underlying VRF.



(a) Graphical representation of our construction for full unbiasedness. Our construction has two “extraction branches”: the upper branch extracts randomness from the (inner) VRF output z in the pseudorandomness game, the lower branch extracts randomness (hardcore bit h) from the preimage x in the unbiasedness game. The VRF input is x , the (underlying) secret key is sk' , the verification key is vk (which contains the hash key hk). The VRF output is y and the proof is π . In the lower branch the HCBs serve as the secret key to a PRF, while the verification key is the PRF input.

$\text{VRF.Gen}(1^\lambda)$	$\text{VRF.Eval}(sk, x \in \{0, 1\}^\lambda)$	$\text{VRF.Verify}(vk, x, y, \pi)$
$(vk', sk') \leftarrow \text{VRF'.Gen}(1^\lambda)$	$(sk', hk) := sk$	$(vk', hk) := vk$
$hk \leftarrow \text{CRH.Gen}(1^\lambda)$	$u := \text{injOWF}(x)$	$u := \text{injOWF}(x)$
$vk := (vk', hk)$	$h := \text{HCF}(x) \in \{0, 1\}^{\lambda^{1/2}}$	$h := \text{HCF}(x)$
$sk := (sk', hk)$	$h' := \text{PRF}(h, vk)$	$h' := \text{PRF}(h, vk)$
return (vk, sk)	$v := \text{CRH.Hash}(hk, u)$	$v := \text{CRH.Hash}(hk, u)$
	$(z, \pi) := \text{VRF'.Eval}(sk', v)$	$z := y \oplus h'$
	$y := z \oplus h'$	req $\text{VRF'.Verify}(vk', v, z, \pi) = 1$
	return (y, π)	return 1

(b) Formal construction of our fully unbiased VRF VRF using an injective OWF injOWF with hardcore function HCF and a VRF VRF' . The PRF is applied to the verification key using the hardcore bits as the key. The PRF is invoked on security parameter $\lambda^{1/2}$ and input length $\ell_x(\lambda^{1/2}) = |vk|$.

Fig. 3: Our construction of a fully unbiased VRF VRF .

The main difficulty is to achieve pseudorandomness and unbiasedness simultaneously. On the one hand, in the pseudorandomness game we can rely on VRF outputs to be random. On the other hand, in the unbiasedness game we know that the preimages (and thus the extracted randomness) are random.

The rough idea is to extract randomness directly from the preimage (lower branch) as well as from an (inner) VRF output z evaluated on the preimage (upper branch), and then combine the two. This, intuitively, ensures that in both games (pseudorandomness and unbiasedness) the output is random. Concretely, the extracted randomness h consists of hardcore bits of the preimage relative to some injective OWF. For full unbiasedness, we evaluate a PRF on the extracted hardcore bits using the verification key as input and the hardcore bits as the secret key. The preimage is processed with the inj. OWF , then hashed and evaluated with the underlying VRF. The final output is simply the XOR of the (inner) VRF output z and the PRF output h' .

Security of our construction. At this point, we give an intuitive high-level overview of our security proof; due to space constraints, we defer the detailed proofs to Section 6. To understand the security of our construction let us start from a simplified construction where the inner VRF is evaluated directly on the preimage x and the extracted randomness h is simply the parity of the preimage, and we ignore the PRF, i.e., $h' = h$. Pseudorandomness of this naive construction follows immediately from the pseudorandomness of the inner VRF. One might wonder whether this construction is already unbiased o.i.p. because—even if the inner VRF’s function table is e.g. the zero function—the outer VRF output is the parity of the preimage and thus unbiased. However, for any possible “extraction function” (here parity) the adversary

could potentially produce a verification key whose function table matches the extraction function; then the outer VRF function would be the zero function. The remedy for this is to

- (on the upper branch) apply the inner VRF not on the preimage directly, but first process the preimage with a OWF and then apply the inner VRF,
- (on the lower branch) extract hardcore bits (HCB) corresponding to the OWF from the preimage.

Thus, if the adversary manages to produce a verification key for the inner VRF whose function table is correlated with the HCB, then the adversary must have found an *efficient* way to compute the HCB from the OWF image only. If the OWF is one-way, then this is impossible, hence the construction is unbiased o.i.p.

However, by introducing the OWF in the upper branch we might have compromised pseudorandomness because the OWF might not be collision-resistant. If the (outer) pseudorandomness adversary submits a query x^i and a challenge x^* such that their OWF images $u^i = u^*$ collide, then we cannot reduce from the pseudorandomness of the inner VRF. This is because then the reduction would have to query u^i to the pseudorandomness challenger of the inner VRF, but its challenge would also be $u^* = u^i$. Thus, we require an *injective* OWF to restore pseudorandomness.

Lastly, so far we overlooked a technical subtlety in the unbiasedness game. Recall that, intuitively, in the unbiasedness game the (outer) VRF images are random because they contain an HCB of the (outer) preimage. However, note in the unbiasedness game that the adversary obtains these preimages in line 5 in Figure 4! To apply the indistinguishability of the HCB, the (outer) preimages must only be used to compute the OWF images and the HCBs themselves; so we cannot supply them to the adversary. Our strategy is to (as the reduction) compute the (inner) VRF outputs and proofs ourselves. Note that because of the unique provability of the inner VRF there is anyway only at most a single (inner) VRF output for each (inner) VRF input (and the proofs are only used for verification). Thus, only observing the OWF images u we brute-force¹² the (inner) VRF output and proof. Now, we don't need to supply the (outer) preimages to the adversary. Unfortunately, we have run into yet another problem, namely, our reduction has become inefficient as we now need to brute-force the (inner) VRF output and proof. This prevents us from finally applying the HCB indistinguishability.

The trick we use is to hash the OWF images u before applying the (inner) VRF. Now, the (inner) VRF inputs v are much smaller (say polylogarithmic) and brute-forcing only requires superpolynomial time. If the OWF is sufficiently hard-to-invert (say subexponentially), then we can allow for such a somewhat inefficient reduction. However, we need to remember the collision problem for the pseudorandomness game that we introduced when we applied the OWF for the first time. Thus, we require the hash function to be collision-resistant. In this way, we ensure both unbiasedness and pseudorandomness.

Full unbiasedness. To achieve full unbiasedness we use the same trick as [GS24] and evaluate a PRF on the verification key. Recall that for full unbiasedness a common set of preimages is evaluated with several VRFs, whereas for unbiasedness o.i.p. each VRF instance is evaluated on independent preimages. Notice that without the PRF evaluation, the lower branch (responsible for unbiasedness) in Figure 3 would not depend on the verification key of the VRF at all. Suppose the adversary sets the inner VRF to the zero function (recall that the adversary essentially controls the inner VRF function table). Now, if (in the full unbiasedness game) two VRFs are applied to the same preimage x , then those (outer) VRF images will be the same. Clearly, this is distinguishable from independent uniform images. Hence, we need to introduce a dependency on the verification key of the VRF in the lower extraction branch. Though, we need to be careful, because the adversary can thus also influence the lower branch and thus introduce correlations between the upper and the lower branch, which would bias the (outer) images. By parallel execution of the OWF we obtain polynomially many HCBs from the (outer) preimage. We can use those

¹² Note that we now always output a valid image (if there exists one), whereas the adversary may have chosen not to output a valid image. This is not a problem because of the monotonicity of the adversary's predicate.

HCBs as the secret key of a PRF, and evaluate the PRF on the VRF verification key.¹³ For any two different verification keys, the PRF would output independent uniform values. In this way, we make sure that the adversary cannot induce correlations between the upper and the lower branch.

Comparison to “VUF-then-Hash”. In practice, the invocation of the VUF/VRF is by far the most expensive operation, hence it is favorable to invoke it on small inputs. Concrete implementations of VUF/VRF often dictate specific input lengths, such as 128, 192 or 256 bits. If the VRF’s preimage is slightly larger than 128 bits, e.g. 130 bits, then the VUF-then-Hash construction has to be implemented with an underlying VUF on 192 bits. This induces two negative consequences:

- a large relative size overhead (e.g. $\approx 50\%$ of the proof size),
- computational inefficiency; often the computational costs of VRF evaluations scale superlinearly with the security parameter (e.g. quadratically for pairing based VRFs).

Thus, the VUF-then-Hash construction is particularly inefficient if the input is large and/or mismatched with standard VRF preimage lengths.

Comparison to [GS24]. First, let us compare our results to [GS24] qualitatively. Giunta and Stewart build a fully unbiased VRF based on any VRF, and the decisional Diffie–Hellman (DDH) assumption with preprocessing which does not hold against quantum adversaries. In comparison, for full unbiasedness we need to require a VRF, an injective OWF and a CRH; all of which are believed to resist quantum attacks.

While our constructions are generic, the many special requirements for the construction of [GS24] such as collision-resistance (in the first component of $\phi_{[a],b}$) and full injectivity (of the commitment) strongly limit the number of possible instantiations beyond tailor-made group-based constructions.

Inherent in both works is the dilemma between pseudorandomness and unbiasedness. Namely, in the unbiasedness game the adversary gets to see the VRF preimages, which means that one cannot directly argue that the extracted randomness from the preimage is uniform and independent. To cope with this problem Giunta and Stewart deploy a Chameleon hash function [KR00] to generate suitable preimages for the adversary that are not actually correlated with the VRF images. In doing so they need to allow the DDH adversary to perform (exponential-time) preprocessing. In contrast, we do not need to rely on a Chameleon hash function, but instead on a standard CRH and subexponential hardness of the OWF. We argue that in some sense the preprocessing in the DDH assumption can also be interpreted as a form of (even) exponential hardness. In Section 7 we show how to resolve the fundamental pseudorandomness–unbiasedness dilemma using non-black-box techniques without relying on superpolynomial hardness.

On the constructive side, we view our construction as a more direct distillation of the construction of [GS24] that only uses the three key observations listed above. In particular, we do not need to split the input into two parts, but instead directly extract randomness from the preimage. An essential part that we take from [GS24] is the idea to evaluate a PRF on the verification key to achieve full unbiasedness. However, unlike Giunta and Stewart we do not need to commit to the PRF key because (as part of the preimage) it is already committed to by the inj. OWF’s image.

3 Avoiding Superpolynomial Security Loss

In this section we show how superpolynomial security loss can be avoided if the underlying VRF fulfills a mild (verifiable) requirement that we call “latency”.

Definition 1 (Latent VRF). *A VRF VRF is (perfectly) latent if there exists a deterministic polynomial-time algorithm VRF.LatEval that on input an evaluation key ek and a preimage x , outputs an image y*

¹³ Giunta and Stewart [GS24] discuss this idea in their technical overview. Because they need the PRF to be collision-resistant, they output $\text{PRF.Eval}(k, \text{vk})$ together with a perfectly binding ElGamal commitment to k .

such that the following holds: For each $\mathbf{vk}$ there exists¹⁴ an evaluation key $\mathbf{ek}$ such that for each preimage x the image $y := \text{VRF.LatEval}(\mathbf{ek}, x)$ verifies, or $\text{VRF.LatEval}(\mathbf{ek}, x) = \perp$ if no image verifies relative to x . Formally,

$$\forall \mathbf{vk} \exists \mathbf{ek} \forall x \left(\begin{array}{l} \text{VRF.LatEval}(\mathbf{ek}, x) \neq \perp \iff \exists \pi : \text{VRF.Verify}(\mathbf{vk}, x, \overbrace{\text{VRF.LatEval}(\mathbf{ek}, x)}^{\text{correct image}}, \pi) = 1 \\ \text{VRF.LatEval}(\mathbf{ek}, x) = \perp \iff \underbrace{\forall y, \pi : \text{VRF.Verify}(\mathbf{vk}, x, y, \pi) = 0}_{\text{no valid image exists}} \end{array} \right) \wedge .$$

In other words, a latent VRF ensures that for every verification key (even those for which no secret key exists) there *exists* a latent evaluation key that allows to evaluate the VRF efficiently.

We elaborate in detail on this new notion in Section 7. At this point, we only mention that latency is a very mild requirement, as all known VRFs are latent.

Now, recall the cause for the subexponential security loss in our black-box reduction from the pseudorandomness of the HCF to the unbiasedness of the VRF in Section 2. The reduction needs to brute-force the inner VRF images and proofs (relative to the adversarially supplied $\mathbf{vk}$) for all (random) preimages sampled by the reduction. This is where the latency of the inner VRF comes into play. Suppose for a second that the adversarially supplied $\mathbf{vk}$ is fixed a priori; then we can fix a latent evaluation key $\mathbf{ek}$ for this $\mathbf{vk}$ and hardcode it into the (non-uniform) reduction. Note that the $\mathbf{vk}$ is part of the first message of the unbiasedness adversary. Thus, if we fix a deterministic¹⁵ non-uniform adversary, we in turn also fix the $\mathbf{vk}$ that it outputs. If we allow our reduction to be non-black-box, i.e., it may depend on the adversary, we can fix the adversary and thus hardcode the $\mathbf{vk}$ into our reduction. Now, instead of brute-forcing the inner VRF images and proofs, the reduction can utilize the latent evaluation key $\mathbf{ek}$ to *efficiently* evaluate the inner VRF.

4 Practical Construction with Minimal Overhead

In the previous section we showed how to get rid of the CRH from our generic construction by relying on the latency of the inner VRF. However, even without the CRH, our construction still requires an injective OWF, which is a rather exotic assumption in practice. In this section, we show how to replace the injective OWF with a standard (keyless) hash function, e.g., SHA-256. The benefit is twofold:

- Keyless hash functions are widely deployed and have optimized implementations (even in hardware).
- Many VRFs already preprocess their input with a hash function to support larger inputs.

Replacing the injective OWF by a keyless CRH. Recall the role of the injective OWF in the security proofs of our generic construction in Section 2. The purpose of the injective OWF is twofold: 1) provide hardcore bits h from the input x to the outer VRF to xor onto the inner VRF's output z (in the unbiasedness reduction), and 2) provide (perfect) collision-resistance for the input to the inner VRF v (in the pseudorandomness reduction).

We observe that we don't need perfect collision-resistance for the input to the inner VRF v in order to prove pseudorandomness of the outer VRF. In fact, computational collision-resistance (as provided by a hash function) is sufficient. Here, we remark that a keyless hash function can only be collision-resistant against uniform adversaries, since a non-uniform adversary can hardcode a collision. In practice, uniform collision-resistance is a widely accepted notion, nevertheless, we state it here for clarity. So, if we replace the injective OWF with a keyless hash function, we can still prove pseudorandomness of the outer VRF against uniform adversaries.

¹⁴ Naturally, we require that the bit size of $\mathbf{ek}$ is polynomially related to the bit size of $\mathbf{vk}$.

¹⁵ W.l.o.g. non-uniform algorithms can be derandomized by non-uniformly fixing the randomness that maximizes their success probability.

To prove unbiasedness of our construction, we need to assume that the keyless hash function admits a hardcore bit; as did the injective OWF before.¹⁶ Here, we need to crucially assume that the hardcore function is pseudorandom against *non-uniform* adversaries—or equivalently, that the CRH is one-way against non-uniform adversaries. This is because our reduction from the pseudorandomness of the HCF to the unbiasedness of the VRF in Section 3 is non-black-box and thus non-uniform.

By replacing the injective OWF with a keyless hash function, we can instantiate our generic construction as depicted in Figure 1.

5 Preliminaries

Notation. We use the convention that $s \oplus \perp = \perp$ for any s . For any algorithm $\mathcal{M}$ we say that $\mathcal{M}$ is $\mathbf{t}_{\mathcal{M}}$ -time-bounded if $\mathcal{M}$ runs in time $\mathbf{t}_{\mathcal{M}}(\lambda)$ on any input $1^\lambda || 0 || x$ for any $x \in \{0, 1\}^*$. To differentiate between time bounds for adversaries and time bounds for the evaluation of algorithms, e.g. a OWF, we use $\mathbf{t}_{\mathcal{P}}^{\text{game}}$ to bound the runtime of an adversary in game “game” against primitive $\mathcal{P}$, while $\mathbf{t}_{\mathcal{M}}$ simply denotes the runtime of the algorithm $\mathcal{M}$. Generally, we assume that each Gen algorithm on input security parameter 1^λ requires λ bits of randomness. Generally, we assume that any parameter function and any distribution is efficiently samplable. Moreover, we use a $(\mathbf{t}, \epsilon)$ -notion of security. This typically means that for any $\mathbf{t}$ -time-bounded adversary the probability of breaking the primitive is bounded by ϵ . We use a succinct description to denote the security of a given primitive. As per convention we say a given primitive is

$$\text{polynomially secure iff } \forall \mathbf{t} \in \text{poly}(\lambda) \quad (4)$$

$$\text{superpolynomially secure iff } \forall \mathbf{t} \in 2^{\text{polylog}(\lambda)} \quad (5)$$

$$\text{subexponentially secure iff } \forall \mathbf{t} \in 2^{\lambda^{o(1)}} \quad (6)$$

the primitive is $(\mathbf{t}, 1/\mathbf{t})$ -secure. For any two distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ we write $\mathcal{D}_0 \approx_{\mathbf{t}, \epsilon} \mathcal{D}_1$ to mean that for any $\mathbf{t}$ -time-bounded distinguisher $\mathcal{A}$ its distinguishing advantage is bounded by ϵ , i.e., $|2 \Pr[b \leftarrow \{0, 1\}, x \leftarrow \mathcal{D}_b : \mathcal{A}(x) = b] - 1| \leq \epsilon$. When we say two distributions are (polynomially) computationally indistinguishable, then we mean that they are at least $(\lambda^c, 1/\lambda^c)$ -indistinguishable for all $c \in \mathbb{N}$. For all randomized algorithms we assume that for any security parameter λ the randomness is chosen uniformly at random from (a set efficiently isomorphic to) $\{0, 1\}^\lambda$. This is w.l.o.g. as we can always use a pseudorandom generator [HIL⁺99] (PRG) to stretch the randomness.

When discussing proof systems we refer to a uniformly random string as URS, in contrast we denote a (not uniform) structured random string as SRS.

5.1 Cryptographic Primitives

Definition 2 (Injective One-Way Function). Let injOWF be a deterministic polynomial-time algorithm. Let $\mathcal{D} = (\mathcal{D}_\lambda)_\lambda$ be some efficiently samplable family of domains (we will often omit the domain and simply assume that $\mathcal{D}_\lambda = \{0, 1\}^\lambda$). We define several properties for injOWF .

Injectivity. We say injOWF is injective, if for each $x \neq x' \in \{0, 1\}^*$ it holds that $\text{injOWF}(x) \neq \text{injOWF}(x')$,

One-Wayness. We say injOWF is $(\mathbf{t}, \epsilon)$ -one-way, if for each non-uniform $\mathbf{t}$ -time-bounded adversary $\mathcal{A}$ its inversion probability is bounded by

$$\exists \lambda_0 \in \mathbb{N} : \forall \lambda \geq \lambda_0 : \Pr[x \leftarrow \mathcal{D}_\lambda : \text{injOWF}(\mathcal{A}(1^\lambda, \text{injOWF}(x))) = \text{injOWF}(x)] \leq \epsilon(\lambda) . \quad (7)$$

¹⁶ In practice, for common hash functions like SHA-256, one can conjecture that the parity of the input is hardcore.

Hardcore function. We say a $\text{HCF} : \{0, 1\}^\lambda \rightarrow \{0, 1\}^{\ell_{\text{HCF}}}$ is a $(t_{\text{HCF}}^{\text{ind}}, \epsilon_{\text{HCF}}^{\text{ind}})$ -hardcore function for injOWF if for each non-uniform $t_{\text{HCF}}^{\text{ind}}$ -time-bounded adversary $\mathcal{A}$ its distinguishing advantage is bounded by

$$\exists \lambda_0 \in \mathbb{N} : \forall \lambda \geq \lambda_0 : |\Pr[\mathcal{A}(1^\lambda, \text{injOWF}(x), \text{HCF}(x)) = 1] - \Pr[\mathcal{A}(1^\lambda, \text{injOWF}(x), h) = 1]| \leq \epsilon_{\text{HCF}}^{\text{ind}}(\lambda) \quad (8)$$

where $x \leftarrow \mathcal{D}_\lambda$ and $h \leftarrow \{0, 1\}^{\ell_{\text{HCF}}(\lambda)}$. If $\ell_{\text{HCF}}(\lambda) = 1$ we say HCF is a hardcore bit.

We emphasize that our definition of OWFs requires a keyless OWF, not a family of OWFs. While this distinction is not important for general OWFs, it makes a difference for injective OWFs. Throughout the paper we will use the term OWF to refer to a keyless OWF, and we will use the term OWF family to refer to a family of OWFs.

We note that via parallel composition we can transform a OWF with a single HCB into one with polynomially many HCBs; their security is polynomially related.

Lemma 1 (Polynomially Many Hardcore Bits). Let injOWF be an injective OWF with an HCB $\text{HCF} : \{0, 1\}^\lambda \rightarrow \{0, 1\}$. Then the parallel OWF

$$\text{injOWF}_\otimes : \{0, 1\}^\lambda \rightarrow \{0, 1\}^* : (x_1, \dots, x_{\lambda^{1/2}}) \mapsto (\text{injOWF}(x_1), \dots, \text{injOWF}(x_{\lambda^{1/2}})) \quad (9)$$

is an injective OWF with HCB $\text{HCF}_\otimes : \{0, 1\}^\lambda \rightarrow \{0, 1\}^{\lambda^{1/2}} : (x_1, \dots, x_{\lambda^{1/2}}) \mapsto (\text{HCF}(x_1), \dots, \text{HCF}(x_{\lambda^{1/2}}))$.

For completeness, we state a version of the famous Goldreich–Levin [GL89] theorem and refer to an excellent exposition in the lecture notes of Stephens-Davidowitz [SD23] for details.

Theorem 4 (Goldreich–Levin [GL89] (version extracted from Section 3.4 of [SD23])). Let injOWF be a deterministic polynomial-time algorithm. If there exists a t_{HCF} -time-bounded distinguisher $\mathcal{D}$ for the HCB $\text{HCF}_{\text{GL}} : x^1 || x^2 \mapsto \langle x^1, x^2 \rangle \bmod 2$ with advantage ϵ_{HCF} , then there exists a $(t_{\text{injOWF}}, \epsilon_{\text{injOWF}})$ -inverter $\mathcal{A}$ for injOWF with

$$\epsilon_{\text{injOWF}}(\lambda) \geq \epsilon_{\text{HCF}}(\lambda)^3/4 \quad \text{and} \quad t_{\text{injOWF}}(\lambda) \leq t_{\text{HCF}}(\lambda)/\epsilon_{\text{HCF}}(\lambda)^2. \quad (10)$$

An injective OWF with its Goldreich–Levin hardcore bit implies a non-interactive perfectly binding commitment scheme [Blu81].

Definition 3 (Non-Interactive Commitment). A non-interactive commitment scheme is a probabilistic polynomial-time algorithm COM that on input a value v and randomness r , outputs a commitment c . We define binding and hiding properties for COM .

Perfectly binding. We say COM is perfectly binding if for each $v \neq v'$ and each r, r' it holds that $\text{COM}(v; r) \neq \text{COM}(v'; r')$.

Computational hiding. We say COM is (t, ϵ) -hiding if for each non-uniform (stateful) t -time-bounded adversary $\mathcal{A}$ its hiding advantage is bounded by

$$\exists \lambda_0 \in \mathbb{N} : \forall \lambda \geq \lambda_0 : \left| 2 \Pr \left[\begin{array}{c} r \leftarrow \{0, 1\}^\lambda \\ (v^0, v^1) \leftarrow \mathcal{A}(1^\lambda) : \mathcal{A}(1^\lambda, \text{COM}(v^b, r)) = b \\ b \leftarrow \{0, 1\} \end{array} \right] - 1 \right| \leq \epsilon(\lambda). \quad (11)$$

Our definition of a commitment scheme does not have an opening algorithm, thus any commitment scheme is implicitly perfectly correct in the following sense: For any honestly generated commitment $c = \text{COM}(v; r)$ the opening consists of (v, r) . To open a commitment c the committer sends the opening (v, r) to the receiver who recomputes $\text{COM}(v; r)$ and accepts v as the opened value if it equals c .

Lemma 2 (Injective OWF implies Commitment [Blu81]). Let injOWF be an injective OWF. Let $\text{injOWF}' : \{0, 1\}^\lambda \times \{0, 1\}^{\lambda+1} \rightarrow \{0, 1\}^* : x^1, x^2 \mapsto (\text{injOWF}(x^1), x^2)$ be the modified OWF with the Goldreich–Levin HCB $\text{HCF}'(x^1, x^2) := x_{\lambda+1}^2 + \sum_{i=1}^\lambda x_i^1 x_i^2 \bmod 2$. We define a commitment scheme COM as follows: On input value $v \in \{0, 1\}$ and randomness $(r^1, r^2) \in \{0, 1\}^\lambda \times \{0, 1\}^\lambda$ let $\text{COM}(v; r) :=$

$\text{injOWF}(r^1, r^2 || \beta)$ where $\beta := v + \sum_{i=1}^{\lambda} r_i^1 r_i^2 \bmod 2$. Then COM is perfectly binding. If injOWF is $(\epsilon_{\text{injOWF}}^{\text{inv}}, \epsilon_{\text{injOWF}}^{\text{inv}})$ -one-way, then COM is $(\epsilon_{\text{COM}}^{\text{hide}}, \epsilon_{\text{COM}}^{\text{hide}})$ -hiding with $\epsilon_{\text{COM}}^{\text{hide}}(\lambda) \leq \epsilon_{\text{injOWF}}^{\text{inv}}(\lambda) / \epsilon_{\text{injOWF}}^{\text{inv}}(\lambda)^2$ and $\epsilon_{\text{COM}}^{\text{hide}}(\lambda) \geq 2\epsilon_{\text{injOWF}}^{\text{inv}}(\lambda)^{1/3}$.

Proof. COM is clearly perfectly binding, because injOWF is injective. Concretely, $\text{COM}(v; r) = \text{COM}(v'; r')$ implies $r = r' \wedge \beta = \beta'$ and thus $v = v'$. Hiding follows directly from the Goldreich–Levin theorem with the (pure) inner product $\text{HCF}(x^1, x^2) := \langle x^1, x^2 \rangle \bmod 2$ as hardcore bit. Our hardcore bit has an additive term $x_{\lambda+1}^2$. Let $\mathcal{R}$ be the reduction that plays the adversary for the pure IP-HCB and the challenger for our HCB. When receiving $(\text{injOWF}(x^1), x^2)$ from the challenger, it samples $x_{\lambda+1}^2 \leftarrow \{0, 1\}$ and sends $(\text{injOWF}(x^1), x^2 || x_{\lambda+1}^2) = \text{injOWF}'(x^1, x^2 || x_{\lambda+1}^2)$ to the adversary. After receiving the guess $b' \in \{0, 1\}$ from the adversary, the reduction sends $b'' := b' \oplus x_{\lambda+1}^2$ to the challenger. Note that $\text{HCF}'(x^1, x^2 || x_{\lambda+1}^2) = x_{\lambda+1}^2 \oplus \text{HCF}(x^1, x^2)$. If the adversary wins, i.e., $b' = \text{HCF}'(x^1, x^2 || x_{\lambda+1}^2)$, then $b'' = \text{HCF}(x^1, x^2)$. I.e., the reduction wins whenever the adversary wins. $\square$

Definition 4 (Non-Interactive Witness-Indistinguishable Proof System [FS90; BOV07]). A non-interactive witness-indistinguishable proof system (NIWI) for an NP-relation R without setup is a tuple of polynomial-time algorithms $\Pi_{\text{WI}} = (\text{Prove}, \text{Verify})$ where

- **Prove** on input a statement $\mathbb{x}$ and a witness w , outputs a proof π ,
- **Verify** on input a statement $\mathbb{x}$, a proof π , outputs a bit b indicating the validity of the proof.

We define several properties for NIWIs.

Perfect completeness. For each statement–witness pair $(\mathbb{x}, w) \in R$ it holds that

$$\Pr[\pi \leftarrow \text{Prove}(\mathbb{x}, w) : \text{Verify}(\mathbb{x}, \pi) = 1] = 1. \quad (12)$$

Perfect soundness. For each statement $\mathbb{x} \notin L_R$ and each π it holds that $\text{Verify}(\mathbb{x}, \pi) = 0$.

Adaptive computational witness-indistinguishability. We say a NIWI Π_{WI} is (t, ϵ) -witness-indistinguishable if for each t -time-bounded malicious verifier $\mathcal{A}$ the two following distributions are (t, ϵ) -indistinguishable:

$$\left\{ \begin{array}{l} (\mathbb{x}, w^0, w^1, \text{st}) \leftarrow \mathcal{A}(1^\lambda) \\ (\mathbb{x}, w^0), (\mathbb{x}, w^1) \in R \quad : (\pi, \text{st}) \\ \pi \leftarrow \text{Prove}(\mathbb{x}, w^0) \end{array} \right\} \approx_{t, \epsilon} \left\{ \begin{array}{l} (\mathbb{x}, w^0, w^1, \text{st}) \leftarrow \mathcal{A}(1^\lambda) \\ (\mathbb{x}, w^0), (\mathbb{x}, w^1) \in R \quad : (\pi, \text{st}) \\ \pi \leftarrow \text{Prove}(\mathbb{x}, w^1) \end{array} \right\}. \quad (13)$$

We require that the malicious verifier produce a valid statement–witness pairs.

For simplicity, we will use general NIWIs that can prove any NP statement, although technically we only need a NIWI for a specific relation that has a $\mathsf{C} \subseteq \mathsf{P}$ verifier (typically $\mathsf{C} = \mathsf{NC}^1$) when relying on a commitment and CPRF that are computable in C .

Definition 5 (Pseudorandom Generator). A pseudorandom generator (PRG) is a deterministic polynomial-time algorithm PRG that takes as input a seed $s \in \{0, 1\}^\lambda$ and outputs a string $\text{PRG}(s) \in \{0, 1\}^{\ell(\lambda)}$ with $\ell(\lambda) > \lambda$. We define pseudorandomness.

Pseudorandomness. We say PRG is (t, ϵ) -pseudorandom if for each non-uniform t -time-bounded adversary $\mathcal{A}$ its distinguishing advantage is bounded by

$$\exists \lambda_0 \in \mathbb{N} : \forall \lambda \geq \lambda_0 : \left| \Pr[s \leftarrow \{0, 1\}^\lambda : \mathcal{A}(1^\lambda, \text{PRG}(s)) = 1] - \Pr[y \leftarrow \{0, 1\}^{\ell(\lambda)} : \mathcal{A}(1^\lambda, y) = 1] \right| \leq \epsilon(\lambda). \quad (14)$$

Definition 6 (Collision-Resistant Hash Function). A collision-resistant hash function is a tuple of polynomial-time algorithms $\text{CRH} = (\text{Gen}, \text{Hash})$ where

- **Gen** on input a security parameter 1^λ outputs a hash key hk ,

- **Hash** on input a hash key hk and a preimage $u \in \{0, 1\}^{\ell_u(\lambda)}$ outputs a hash image $v \in \{0, 1\}^{\ell_v(\lambda)}$.

We define collision-resistance.

Collision resistance. We say a CRH CRH is (t, ϵ) -collision-resistant if for each non-uniform t -time-bounded adversary $\mathcal{A}$ its collision probability is bounded by

$$\exists \lambda_0 \in \mathbb{N} : \forall \lambda \geq \lambda_0 : \Pr \left[\begin{array}{c} hk \leftarrow \text{Gen}(1^\lambda) \\ u, u' \leftarrow \mathcal{A}(1^\lambda, hk) \end{array} : \begin{array}{c} \text{Hash}(hk, u) = \text{Hash}(hk, u') \\ \wedge u \neq u' \end{array} \right] \leq \epsilon(\lambda) . \quad (15)$$

Definition 7 (Verifiable Random Function [MRV99]). A verifiable random function (VRF) is a tuple of polynomial-time algorithms $VRF = (\text{Gen}, \text{Eval}, \text{Verify})$ where

- **Gen** on input security parameter 1^λ , outputs a verification key vk and a secret key sk ,
- **Eval** on input secret key sk and preimage $x \in \{0, 1\}^\lambda$, outputs an image $y \in \{0, 1\}^{\ell_v(\lambda)}$ and a proof π ,
- **Verify** on input verification key vk , preimage x , image y and proof π , outputs 1 if the proof is valid and 0 otherwise.

Verify is deterministic, **Gen** is necessarily probabilistic. **Eval** can be probabilistic or deterministic.¹⁷ We define several properties for VRFs.

Unique provability. For any vk , any x , any y_1, y_2 , and any π_1, π_2 it holds that

$$\text{Verify}(vk, x, y_1, \pi_1) = 1 \wedge \text{Verify}(vk, x, y_2, \pi_2) = 1 \implies y_1 = y_2 . \quad (16)$$

Pseudorandomness. We say a VRF VRF is (t, ϵ) -pseudorandom if for each non-uniform t -time-bounded (stateful) adversary $\mathcal{A}$ its pseudorandomness advantage is bounded by

$$\exists \lambda_0 \in \mathbb{N} : \forall \lambda \geq \lambda_0 : \left| 2 \Pr \left[\text{Exp}_{VRF, \mathcal{A}}^{\text{pr}}(\lambda) = 1 \right] - 1 \right| \leq \epsilon(\lambda) \quad (17)$$

with the pseudorandomness game defined in Figure 4.

Unbiasability. [GS24] We say a VRF VRF is (fully) (t, ϵ, a, b) -unbiasable if for each non-uniform t -time-bounded (stateful) adversary $\mathcal{A}$ its unbiasedability advantage is bounded¹⁸ by

$$\exists \lambda_0 \in \mathbb{N} : \forall \lambda \geq \lambda_0 : 2 \Pr \left[\text{Exp}_{VRF, \mathcal{A}}^{\text{fub}}(\lambda) = 1 \right] - 1 \leq \epsilon(\lambda) \quad (18)$$

with the unbiasedability game defined in Figure 4. We say a VRF VRF is (t, ϵ, a, b) -unbiasable on independent points (o.i.p.) if for each non-uniform t -time-bounded (stateful) adversary $\mathcal{A}$ its unbiasedability is bounded by

$$\exists \lambda_0 \in \mathbb{N} : \forall \lambda \geq \lambda_0 : 2 \Pr \left[\text{Exp}_{VRF, \mathcal{A}}^{\text{ipub}}(\lambda) = 1 \right] - 1 \leq \epsilon(\lambda) \quad (19)$$

with the unbiasedability game defined in Figure 4. A predicate $p : (\{0, 1\} \cup \{\perp\})^* \rightarrow \{0, 1\}$ is monotone, iff

$$\forall z \in \{0, 1\}^*, S \subseteq \{1, \dots, |z|\} : \Pr_{r_p}[\rho(z; r_p) = 1] \geq \Pr_{r_p}[\rho(z'; r_p) = 1] \text{ where } z'_i := \begin{cases} z_i & \text{if } i \in S \\ \perp & \text{if } i \notin S \end{cases} . \quad (20)$$

Remark 3 (Domain). In the above definition we fix the VRF domain to be $\{0, 1\}^\lambda$, but more generally the domain can be any finite set. Typically, this set should be efficiently samplable, have an efficient membership test, and (importantly) is independent of the concrete verification key.

If the VRF domain has polynomial size, then (perfect) unbiasedability o.i.p. is still possible (for example by the VRP of [DP07]). However, full unbiasedability with negligible ϵ is impossible by definition. This is because an unbiasedability adversary can approximate the statistical distance of the (tuple of) real VRF evaluations from uniform if the number of (random) preimages b is sufficiently larger than the domain size.

¹⁷ Typically, in NIZK/NIWI-based constructions (in particular [GHK⁺17b; Bit20]) the proofs in **Eval** are randomized, while in most group-based constructions the proofs are unique.

¹⁸ Note the lack of absolute value.

$\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{pr}}(\lambda)$ 1 : $L := \emptyset$ 2 : $(\text{vk}, \text{sk}) \leftarrow \text{VRF.Gen}(1^\lambda)$ 3 : $x^* \leftarrow \mathcal{A}^{\text{O}_{\text{VRF}}}(\text{vk})$ 4 : $b \leftarrow \{0, 1\}$ 5 : $y_0 \leftarrow \{0, 1\}^{\ell_y(\lambda)}$ 6 : $(y_1, \pi_1) \leftarrow \text{VRF.Eval}(\text{sk}, x^*)$ 7 : $b' \leftarrow \mathcal{A}^{\text{O}_{\text{VRF}}}(y_b)$ 8 : req $b = b' \wedge x^* \notin L$ 9 : return 1 $\text{Exp}_{\text{CPRF}, \mathcal{A}}^{\text{ad-pr}}(\lambda)$ 1 : $L := \emptyset$ 2 : $\text{sk} \leftarrow \text{CPRF.Gen}(1^\lambda)$ 3 : $\chi \leftarrow \mathcal{A}^{\text{O}_{\text{CPRF}}}(1^\lambda)$ 4 : $\text{sk}_\chi := \text{CPRF.Const}(\text{sk}, \chi)$ 5 : $x^* \leftarrow \mathcal{A}^{\text{O}_{\text{CPRF}}}(\text{sk}_\chi)$ 6 : $b \leftarrow \{0, 1\}$ 7 : $y_0 \leftarrow \{0, 1\}$ 8 : $y_1 := \text{CPRF.Eval}(\text{sk}, x^*)$ 9 : $b' \leftarrow \mathcal{A}^{\text{O}_{\text{CPRF}}}(y_b)$ 10 : req $b = b' \wedge x^* \notin L \wedge \chi(x^*) = 0$ 11 : return 1 $\text{Exp}_{\text{CPRF}, \mathcal{A}}^{\text{sel-pr}}(\lambda)$ 1 : $L := \emptyset$ 2 : $\text{sk} \leftarrow \text{CPRF.Gen}(1^\lambda)$ 3 : $(x^*, \chi) \leftarrow \mathcal{A}^{\text{O}_{\text{CPRF}}}(1^\lambda)$ 4 : $\text{sk}_\chi := \text{CPRF.Const}(\text{sk}, \chi)$ 5 : $b \leftarrow \{0, 1\}$ 6 : $y_0 \leftarrow \{0, 1\}$ 7 : $y_1 := \text{CPRF.Eval}(\text{sk}, x^*)$ 8 : $b' \leftarrow \mathcal{A}^{\text{O}_{\text{CPRF}}}(y_b, \text{sk}_\chi)$ 9 : req $b = b' \wedge x^* \notin L \wedge \chi(x^*) = 0$ 10 : return 1	$\text{O}_{\text{VRF}}(x)$ 1 : $L := L \cup \{x\}$ 2 : $(y, \pi) \leftarrow \text{VRF.Eval}(\text{sk}, x)$ 3 : return (y, π) $\text{O}_{\text{CPRF}}(x)$ 1 : $L := L \cup \{x\}$ 2 : return $y := \text{CPRF.Eval}(\text{sk}, x)$ $\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{ipub}}(\lambda)$ 1 : $(p, \text{vk}_1, \dots, \text{vk}_{a(\lambda)}) \leftarrow \mathcal{A}(1^\lambda)$ 2 : req p is monotone 3 : $x_{1,1}, \dots, x_{a(\lambda), b(\lambda)} \leftarrow \{0, 1\}^\lambda$ 4 : $y_{1,1}^1, \dots, y_{a(\lambda), b(\lambda)}^1 \leftarrow \{0, 1\}^{\ell_y(\lambda)}$ 5 : $((y_{1,1}^0, \pi_{1,1}^0), \dots, (y_{a(\lambda), b(\lambda)}^0, \pi_{a(\lambda), b(\lambda)}^0)) \leftarrow \mathcal{A}(x_{1,1}, \dots, x_{a(\lambda), b(\lambda)})$ 6 : $b \leftarrow \{0, 1\}$ 7 : for $i \in \{1, \dots, a(\lambda)\}, j \in \{1, \dots, b(\lambda)\}$ 8 : $\beta_{i,j} := \text{VRF.Verify}(\text{vk}_i, x_{i,j}, y_{i,j}^0, \pi_{i,j}^0)$ 9 : if $\beta_{i,j} = 1$ then $z_{i,j} := y_{i,j}^b$ 10 : else $z_{i,j} := \perp$ 11 : $r_p \leftarrow \{0, 1\}^\lambda$ 12 : return $\rho(z_{1,1}, \dots, z_{a(\lambda), b(\lambda)}; r_p) \oplus b$ $\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{fub}}(\lambda)$ 1 : $(p, \text{vk}_1, \dots, \text{vk}_{a(\lambda)}) \leftarrow \mathcal{A}(1^\lambda)$ 2 : req $ \{\text{vk}_1, \dots, \text{vk}_{a(\lambda)}\} = a(\lambda)$ 3 : req p is monotone 4 : $x_1, \dots, x_{b(\lambda)} \leftarrow \{0, 1\}^\lambda$ 5 : $y_{1,1}^1, \dots, y_{a(\lambda), b(\lambda)}^1 \leftarrow \{0, 1\}^{\ell_y(\lambda)}$ 6 : $((y_{1,1}^0, \pi_{1,1}^0), \dots, (y_{a(\lambda), b(\lambda)}^0, \pi_{a(\lambda), b(\lambda)}^0)) \leftarrow \mathcal{A}(x_1, \dots, x_{b(\lambda)})$ 7 : $b \leftarrow \{0, 1\}$ 8 : for $i \in \{1, \dots, a(\lambda)\}, j \in \{1, \dots, b(\lambda)\}$ 9 : $\beta_{i,j} := \text{VRF.Verify}(\text{vk}_i, x_j, y_{i,j}^0, \pi_{i,j}^0)$ 10 : if $\beta_{i,j} = 1$ then $z_{i,j} := y_{i,j}^b$ 11 : else $z_{i,j} := \perp$ 12 : $r_p \leftarrow \{0, 1\}^\lambda$ 13 : return $\rho(z_{1,1}, \dots, z_{a(\lambda), b(\lambda)}; r_p) \oplus b$
---	---

Fig. 4: Pseudorandomness and unbiasedness games. $\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{pr}}$ and $\text{Exp}_{\text{CPRF}, \mathcal{A}}^{\text{pr}}$ denote the pseudorandomness games, $\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{ipub}}$ (resp. $\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{fub}}$) denotes the unbiasedness o.i.p. (resp. full) game for VRF VRF, CPRF CPRF and (stateful) adversary $\mathcal{A}$. The unbiasedness notions are from [GS24]; in the o.i.p. notion each of the VRFs is evaluated on its own (independent) sets of preimages, whereas in the full unbiasedness each VRF is evaluated on a common set of preimages.

Remark 4 (Bounded Proof Length). For any fixed VRF VRF there exists some fixed polynomial $t_{\text{VRF.Eval}}$ that bounds the runtime of its VRF.Eval algorithm. Thus, for any security parameter λ we can—w.l.o.g.—assume that VRF.Verify rejects any proof that is longer than $t_{\text{VRF.Eval}}(\lambda)$ because honestly generated proofs are always of length at most $t_{\text{VRF.Eval}}(\lambda)$. Longer proofs can be considered syntactically invalid.

Definition 8 (Constrained Pseudorandom Function ([BW13; KPT⁺13; BGI14] as per [GHK⁺17b; Bit20])). A constrained pseudorandom function (CPRF) for a constraint class $\mathcal{C}$ is a tuple of polynomial-time algorithms $\text{CPRF} = (\text{Gen}, \text{Eval}, \text{Const})$ where

- **Gen** on input security parameter 1^λ , outputs a secret key sk ,
- **Eval** on input secret key sk and preimage $x \in \{0, 1\}^{\ell_x(\lambda)}$, outputs an image $y \in \{0, 1\}$,
- **Const** on input secret key sk and constraint $\chi \in \mathcal{C}$, outputs a constrained secret key sk_χ .

Eval is deterministic, **Gen** is necessarily probabilistic. We define several properties for CPRFs.

Correctness. We say a CPRF CPRF is (perfectly) correct if for each sk , each $\chi \in \mathcal{C}$, and each $x \in X_\chi := \{x' \in \{0, 1\}^\lambda \mid \chi(x') = 1\}$ it holds that $\text{Eval}(\text{sk}, x) = \text{Eval}(\text{sk}_\chi, x)$.

Pseudorandomness. We say a CPRF CPRF is adaptively single-key (t, ϵ) -pseudorandom if for each non-uniform t -time-bounded (stateful) adversary $\mathcal{A}$ its pseudorandomness advantage is bounded by

$$\exists \lambda_0 \in \mathbb{N} : \forall \lambda \geq \lambda_0 : \left| 2 \Pr \left[\text{Exp}_{\text{CPRF}, \mathcal{A}}^{\text{ad-pr}}(\lambda) = 1 \right] - 1 \right| \leq \epsilon(\lambda) \quad (21)$$

with the pseudorandomness game defined in Figure 4. Analogously, we say a CPRF CPRF is selectively single-key (t, ϵ) -pseudorandom if for each non-uniform t -time-bounded (stateful) adversary $\mathcal{A}$ its pseudorandomness advantage is bounded by

$$\exists \lambda_0 \in \mathbb{N} : \forall \lambda \geq \lambda_0 : \left| 2 \Pr \left[\text{Exp}_{\text{CPRF}, \mathcal{A}}^{\text{sel-pr}}(\lambda) = 1 \right] - 1 \right| \leq \epsilon(\lambda) . \quad (22)$$

Pseudorandomness for constrained PRFs essentially means that PRF images are indistinguishable from random images even given constrained secret keys that don't trivially allow evaluating the function on the challenge preimage.

For completeness we note that PRFs¹⁹ are implied by OWF via [HIL⁺99; GGM86; MP23] with a polynomial security loss. For the constructions of [GHK⁺17b; Bit20] we use CPRFs that support a specific constraint class (namely partition schemes); the details are not relevant for our construction so we refer the reader to [GHK⁺17b; Bit20] for further details.

5.2 Computational Problems

For completeness we define some lattice-based computational problems. In accordance with our notion of OWFs we define the learning-with-errors (LWE) problem as a keyless function; not as a family of functions.

Definition 9 (Learning-with-Errors [Reg05]). Let $q : \mathbb{N} \rightarrow \mathbb{N}$ be a prime modulus, and let $n, m : \mathbb{N} \rightarrow \mathbb{N}$ be dimensions. Let $\chi = (\chi_\lambda)_\lambda$ be a family of error distributions over $\mathbb{Z}_{q(\lambda)}$. Let

$$\text{LWE}_{n, m, q} := \left\{ (\mathbf{A}, \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top) \mid \mathbf{A} \leftarrow \mathbb{Z}_{q(\lambda)}^{m(\lambda) \times n(\lambda)}, \mathbf{s} \leftarrow \mathbb{Z}_{q(\lambda)}^{n(\lambda)}, \mathbf{e} \leftarrow \chi_\lambda^{m(\lambda)} \right\} \quad (23)$$

be the LWE distribution. We say a distribution $\text{LWE}_{n, m, q, \chi}$ is (t, ϵ) -hard if for each non-uniform t -time-bounded adversary $\mathcal{A}$ its distinguishing advantage is bounded by

$$\left| \Pr[\mathcal{A}(1^\lambda, \mathbf{A}, \mathbf{y}^1) = 1] - \Pr[\mathcal{A}(1^\lambda, \mathbf{A}, \mathbf{y}^0) = 1] \right| \leq \epsilon(\lambda) \quad (24)$$

¹⁹ But no constrained PRFs for all constraint classes.

where $(\mathbf{A}, \mathbf{y}^1) \leftarrow \text{LWE}_{n,m,q}$ and $\mathbf{y}^0 \leftarrow \mathbb{Z}_{q(\lambda)}^{m(\lambda)}$.

One-Way function. Let $\text{injOWF}_{n,m,q,\chi} : (\mathbf{A}, \mathbf{s}, \mathbf{e}^\top) \mapsto (\mathbf{A}, \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$ be the LWE one-way function with domain distribution $\mathcal{D}_\lambda := \mathbb{Z}_{q(\lambda)}^{n(\lambda) \times m(\lambda)} \times \mathbb{Z}_{q(\lambda)}^{n(\lambda)} \times \chi_\lambda^{m(\lambda)}$.

Error distribution. As standard notation we write $\chi_{\sigma(\lambda)}$ for the (renormalized) truncated discrete Gaussian distribution with standard deviation $\sigma(\lambda)$ over $\mathbb{Z}$ (truncated at $\lambda \cdot \sigma(\lambda)$). This means that for any error $\mathbf{e} \leftarrow \chi_{\sigma(\lambda)}$ it holds that $\|\mathbf{e}\| \leq \lambda \cdot \sigma(\lambda)$. This distribution is statistically $2^{-\lambda^2}$ -close to the corresponding untruncated Gaussian distribution. For discrete Gaussian error we use the shorthand notation (n, m, q, σ) -LWE for the (n, m, q, χ_σ) -LWE.

As a concrete example, for Theorem 17 we may assume the subexp. hardness of the (n, m, q, σ) -LWE problem with the following parameters:

$$n(\lambda) \geq \lambda \quad q(\lambda) := n(\lambda)^6 \quad m(\lambda) := 3n(\lambda) \log(q(\lambda)) \quad \sigma(\lambda) := n(\lambda)^3. \quad (25)$$

In particular, for these parameters it holds that $\sigma(\lambda) \gtrsim 2\sqrt{n(\lambda)}$. Thus, Regev's reduction [Reg05] implies the hardness of LWE based on the hardness of worst-case lattice problems.

Definition 10 (Learning-with-Rounding [BPR12]). Let $q, p : \mathbb{N} \rightarrow \mathbb{N}$ be prime moduli, and let $n, m : \mathbb{N} \rightarrow \mathbb{N}$ be dimensions. Let

$$\text{LWR}_{n,m,q,p} := \left\{ \left(\mathbf{A}, [\mathbf{s}^\top \mathbf{A}]_{p(\lambda)} \mid \mathbf{A} \leftarrow \mathbb{Z}_{q(\lambda)}^{m(\lambda) \times n(\lambda)}, \mathbf{s} \leftarrow \mathbb{Z}_{q(\lambda)}^{n(\lambda)} \right) \right\} \quad (26)$$

be the LWR distribution. We say a distribution $\text{LWR}_{n,m,q,p}$ is (t, ϵ) -hard if for each non-uniform t -time-bounded adversary $\mathcal{A}$ its distinguishing advantage is bounded by

$$\left| \Pr[\mathcal{A}(1^\lambda, \mathbf{A}, \mathbf{y}^1) = 1] - \Pr[\mathcal{A}(1^\lambda, \mathbf{A}, \mathbf{y}^0) = 1] \right| \leq \epsilon(\lambda) \quad (27)$$

where $(\mathbf{A}, \mathbf{y}^1) \leftarrow \text{LWR}_{n,m,q,p}$ and $\mathbf{y}^0 \leftarrow \mathbb{Z}_{p(\lambda)}^{m(\lambda)}$.

6 Generic Construction of Unbiasable VRFs

We now state our formal results and give proofs for our statements.

6.1 Warm-Up: Unbiasability o.I.P.

As correctness and unique provability follow immediately, we prove pseudorandomness and unbiasability o.i.p. for our construction.

Construction 1 (Unbiasable o.I.P. VRF). Let injOWF be an injective OWF. Let CRH be a CRH. Let VRF' be a VRF. We define a VRF VRF in Figure 3 with $\text{PRF}(h, \text{vk}) := h$. VRF evaluation (ignoring the VRF proof) is described succinctly by

$$y := ((\text{VRF}'.\text{Eval}(\text{sk}, \cdot) \circ \text{CRH}.\text{Hash}(\text{hk}, \cdot) \circ \text{injOWF}) \oplus \text{HCF})(x). \quad (28)$$

Theorem 5 (Pseudorandomness). Let injOWF be an injective OWF with output length ℓ_u and with a single hardcore bit $\text{HCF} : \{0, 1\}^\lambda \rightarrow \{0, 1\}$. Let VRF' be a $(t_{\text{VRF}'}^{\text{pr}}, \epsilon_{\text{VRF}'}^{\text{pr}})$ -pseudorandom VRF. Let CRH be a $(t_{\text{CRH}}^{\text{cr}}, \epsilon_{\text{CRH}}^{\text{cr}})$ -CRH with input length ℓ_u and output length ℓ_v . Then VRF from Construction 1 is $(t_{\text{VRF}}^{\text{pr}}, \epsilon_{\text{VRF}}^{\text{pr}})$ -pseudorandom for

$$\epsilon_{\text{VRF}}^{\text{pr}}(\lambda) \geq t_{\text{VRF}}^{\text{pr}}(\lambda)^2 \epsilon_{\text{CRH}}^{\text{cr}}(\lambda) + \epsilon_{\text{VRF}'}^{\text{pr}}(\ell_v(\lambda)) \quad (29)$$

$$t_{\text{VRF}}^{\text{pr}}(\lambda) \leq \frac{\min(t_{\text{CRH}}^{\text{cr}}(\lambda) - \lambda^2, t_{\text{VRF}'}^{\text{pr}}(\ell_v(\lambda))) - t_{\text{CRH}.\text{Gen}}(\lambda)}{t_{\text{CRH}.\text{Hash}}(\lambda) + t_{\text{injOWF}}(\lambda)}. \quad (30)$$

Proof. We show pseudorandomness through a single hybrid game. Let $\mathcal{A}$ be a non-uniform $\mathbf{t}_{\text{VRF}}^{\text{pr}}$ -time-bounded adversary for the pseudorandomness game in Figure 4. Denote by $\mathbf{q}(\lambda)$ the maximal number of queries that $\mathcal{A}$ makes.

Game 0 (Original game): This is the original game where the adversary queries preimages x^i and challenge x^* to receive images y^i , proofs π^i , and the challenge image y^* for a random $b \leftarrow \{0, 1\}$.

Game 1 (Abort on collisions): Now, we abort the game if $v^* \in \{v_1, \dots, v_{\mathbf{q}(\lambda)}\}$. To justify this step we need the injectivity of injOWF and that the hash key hk is generated honestly. Suppose that $\mathcal{A}$ submits $x^i \neq x^*$ (hence $u^i \neq u^*$ by injectivity of injOWF) such that $v^i = v^*$. Then we can submit $u^i \neq u^*$ to the CR challenger for CRH. In this reduction, we can honestly generate proofs for $\mathcal{A}$'s queries as before. Hence,

$$\Pr[\mathbf{G}_1(\lambda) = 1] \geq \Pr[\mathbf{G}_0(\lambda) = 1] - \mathbf{q}(\lambda)^2 \epsilon_{\text{CRH}}^{\text{cr}}(\lambda) . \quad (31)$$

Since $\mathcal{A}$ runs in time $\mathbf{t}_{\text{CRH}}(\lambda) \leq \mathbf{t}_{\text{VRF}}^{\text{pr}}(\lambda)$, the reduction runs in time $\mathbf{t}_{\text{VRF}}^{\text{pr}}(\lambda) + \lambda^2$ (plus other housekeeping) to select the collision.

Now, we bound the adversary's advantage in this game. We reduce from the pseudorandomness of the underlying VRF VRF' , and the fact that all VRF inputs v^i, v^* are different. Let $\mathcal{R}$ be the following reduction: it plays the adversary for the pseudorandomness game of VRF' . Once it obtains the vk' from the PR challenger, it samples $\text{hk} \leftarrow \text{CRH.Gen}(1^\lambda)$ and submits $\text{vk} := (\text{vk}', \text{hk})$ to the PR adversary $\mathcal{A}$. On query x^i from $\mathcal{A}$, the reduction computes $u^i := \text{injOWF}(x^i)$ and $v^i := \text{CRH.Hash}(\text{hk}, u^i)$. It queries v^i to the PR challenger to obtain the inner image z^i and proof π^i . It then returns the outer image $y^i := z^i \oplus h^i$ (where $h^i := \text{HCF}(x^i)$) and proof $\pi := \pi^i$ to $\mathcal{A}$. On challenge x^* from $\mathcal{A}$, the reduction proceeds analogously, but instead obtains a random value z^* from the PR challenger. Thus,

$$\Pr[\mathbf{G}_1(\lambda) = 1] \leq 1/2 + \epsilon_{\text{VRF}'}^{\text{pr}}(\lambda) . \quad (32)$$

Overall, we find

$$\epsilon_{\text{VRF}}^{\text{pr}}(\lambda) := \Pr[\mathbf{G}_0(\lambda) = 1] - 1/2 \quad (33)$$

$$\leq \mathbf{q}(\lambda)^2 \epsilon_{\text{CRH}}^{\text{cr}}(\lambda) + \epsilon_{\text{VRF}'}^{\text{pr}}(\ell_v(\lambda)) \quad (34)$$

$$\leq \mathbf{t}_{\text{VRF}}^{\text{pr}}(\lambda)^2 \epsilon_{\text{CRH}}^{\text{cr}}(\lambda) + \epsilon_{\text{VRF}'}^{\text{pr}}(\ell_v(\lambda)) . \quad (35)$$

The reduction against the pseudorandomness game is $\mathbf{t}_{\text{VRF}'}^{\text{pr}}$ -time-bounded for

$$\mathbf{t}_{\text{VRF}'}^{\text{pr}}(\ell_v(\lambda)) \leq \mathbf{t}_{\text{VRF}}^{\text{pr}}(\lambda) \cdot (\mathbf{t}_{\text{CRH.Hash}}(\lambda) + \mathbf{t}_{\text{injOWF}}(\lambda)) + \mathbf{t}_{\text{CRH.Gen}}(\lambda) . \quad (36)$$

Recall that $\mathbf{t}_{\text{CRH.Hash}}$ resp. $\mathbf{t}_{\text{injOWF}}$ describe the time needed to evaluate the CRH resp. OWF, i.e., they are some fixed polynomials. $\square$

Theorem 6 (Unbiasability o.I.P.). Let injOWF be an injective $(\mathbf{t}_{\text{injOWF}}, \epsilon_{\text{injOWF}})$ -OWF with output length ℓ_u and with a single hardcore bit $\text{HCF} : \{0, 1\}^\lambda \rightarrow \{0, 1\}$. Let VRF' be a perfectly uniquely provable VRF. Let CRH be a CRH with input length ℓ_u and output length ℓ_v . Then VRF is $(\mathbf{t}_{\text{VRF}}^{\text{ipub}}, \epsilon_{\text{ipub}})$ -unbiasable o.i.p. for

$$\epsilon_{\text{VRF}}^{\text{ipub}}(\lambda) \geq 2\mathbf{a}(\lambda)\mathbf{b}(\lambda)\epsilon_{\text{injOWF}}^{\text{inv}}(\lambda)^{1/3} \quad (37)$$

$$\mathbf{t}_{\text{VRF}}^{\text{ipub}}(\lambda) \leq \mathbf{t}_{\text{injOWF}}^{\text{inv}}(\lambda)/3\epsilon_{\text{injOWF}}^{\text{inv}}(\lambda)^{2/3} - \mathbf{t}_{\text{CRH.Setup}}(\lambda) - \mathbf{a}(\lambda)\mathbf{b}(\lambda)\lambda \cdot \left(\frac{\mathbf{t}_{\text{injOWF}}(\lambda) + \mathbf{t}_{\text{CRH.Hash}}(\lambda)}{+2\mathbf{t}_{\text{VRF}', \text{Eval}}(\ell_v(\lambda))\mathbf{t}_{\text{VRF}', \text{Verify}}(\ell_v(\lambda))} \right) . \quad (38)$$

Proof. First, recall the definition of unbiasedability: the adversary should not be able to produce a (mono-tone) predicate that distinguishes between honestly generated VRF images and random strings. So, to show unbiasedability, we need to modify the real unbiasedability game in a way that is undetectable to the adversary; so that eventually the predicate is fed with random strings instead of honest VRF images

without changing the output distribution of the predicate noticeably.

Note that the only source of randomness in the unbiasedability game is the set of the preimages $x_{i,j}$. In fact the entropy of the preimages directly translates (through the lower branch of our construction) to the VRF output. If we could argue that the hardcore bit $h_{i,j}$ is pseudorandom, then we could simply switch it from the real hardcore function evaluation $h_{i,j} := \text{HCF}(x_{i,j})$ to an independent uniformly random bit $h_{i,j} \leftarrow \{0,1\}$ and be done with the proof. The only obstacle to invoking the pseudorandomness of the hardcore bit is that in the original unbiasedability game the preimages are given to the adversary in line 5 of the $\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{ipub}}$ game. So, instead of directly invoking the HCF pseudorandomness, we first modify the unbiasedability game such that the challenger brute-forces the VRF images and proofs $y_{i,j}, \pi_{i,j}$ for the preimages $x_{i,j}$ itself. Note that this requires time that is exponential in the input size $\ell_v(\lambda) := |v|$ of the inner VRF. Because v is the output of the CRH, its length is comparatively small, e.g. $\log(\lambda)^2$. Hence, brute-forcing does not take an exponential time in the input length (i.e., 2^λ) but only superpolynomial time $2^{\log(\lambda)^2}$. Now, we are in a game where the preimages $x_{i,j}$ are exclusively used to compute the OWF images $u_{i,j}$ and the hardcore bits $h_{i,j}$. Because we assume that the OWF is secure against subexponential adversaries (in λ not $\log(\lambda)^2$) and our reduction only requires superpolynomial time to brute-force the VRF images, this allows us to switch the hardcore bits $h_{i,j}$ to uniformly random bits $h_{i,j} \leftarrow \{0,1\}$.

Let us proceed with the formal proof. We show unbiasedability by a sequence of hybrid games. Let $\mathcal{A}$ be a non-uniform $\text{t}_{\text{VRF}}^{\text{ipub}}$ -time-bounded adversary for the unbiasedability game in Figure 4.

Game 0 (Original game): This is the original game where the adversary produces the images $y_{i,j}^0$ and proofs $\pi_{i,j}^0$ (given the preimages $x_{i,j}$).

Game 1 (Self-Compute $y_{i,j}^0$ and $\pi_{i,j}^0$): After sampling $x_{i,j} \leftarrow \{0,1\}^\lambda$, the challenger computes the intermediate values $h_{i,j}, v_{i,j}$ as defined in VRF.Eval . Let $\ell_v(\lambda) := |v_{i,j}|$. Then it brute-forces in time $2^{\text{t}_{\text{VRF}'}. \text{Eval}(\ell_v(\lambda))} \text{t}_{\text{VRF}'}. \text{Verify}(\ell_v(\lambda))$ an image $z_{i,j} \in \{0,1\}$ and proof $\pi_{i,j} \in \{0,1\}^{\leq \text{t}_{\text{VRF}'}. \text{Eval}(\ell_v(\lambda))}$ such that $\text{VRF}'.\text{Verify}(vk'_i, v_{i,j}, z_{i,j}, \pi_{i,j}) = 1$. It sets $z_{i,j} := \perp$ and $\pi_{i,j} := \perp$ if no such $z_{i,j}, \pi_{i,j}$ exist. The challenger sets $\hat{y}_{i,j}^0 := z_{i,j} \oplus h_{i,j}$, samples $\hat{y}_{i,j}^1 \leftarrow \{0,1\}^{\ell_v(\lambda)}$ and computes $\hat{z}_{i,j}$ analogously.

We argue that $\beta_{i,j} = 1 \implies y_{i,j}^0 = \hat{y}_{i,j}^0$, i.e., if the adversary's image and proof verify, then the challenger's image equals the adversary's image. This follows directly from the perfect unique provability of VRF' . Consequently, $(y_{i,j}^0)_{i,j} \leq (\hat{y}_{i,j}^0)_{i,j}$ and thus $\rho(y_{1,1}^0, \dots, y_{\mathfrak{a}(\lambda), \mathfrak{b}(\lambda)}^0) \leq \rho(\hat{y}_{1,1}^0, \dots, \hat{y}_{\mathfrak{a}(\lambda), \mathfrak{b}(\lambda)}^0)$.

We find that

$$\Pr[\mathbf{G}_1(\lambda) = 1] = \Pr[\mathbf{G}_1(\lambda) = 1 \wedge b = 0] + \Pr[\mathbf{G}_1(\lambda) = 1 \wedge b = 1] \quad (39)$$

$$= \Pr[\mathbf{G}_1(\lambda) = 1 \wedge b = 0] + \Pr[\mathbf{G}_0(\lambda) = 1 \wedge b = 1] \quad (40)$$

$$\geq \Pr[\mathbf{G}_0(\lambda) = 1 \wedge b = 0] + \Pr[\mathbf{G}_0(\lambda) = 1 \wedge b = 1] \quad (41)$$

$$= \Pr[\mathbf{G}_0(\lambda) = 1] . \quad (42)$$

Now, notice that in this game (because the challenger computes $y_{i,j}^0$ and $\pi_{i,j}^0$ itself) the adversary in line 5 of the $\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{ipub}}$ no longer gets to see the preimages $x_{i,j}$.

We stress that the underlying VRF's proofs $\pi_{i,j}$ can have length at most $\text{t}_{\text{VRF}'}. \text{Eval}(\ell_v(\lambda))$ (recall Remark 4). Thus, the brute-forcing only has to cover proofs of length at most $\text{t}_{\text{VRF}'}. \text{Eval}(\ell_v(\lambda))$ because longer proofs are considered syntactically invalid. For example for $\ell_v(\lambda) \in \text{polylog}(\lambda)$ the term $\text{t}_{\text{VRF}'}. \text{Eval}(\ell_v(\lambda))$ is superpolynomial in λ , not exponential in λ . In this example, our challenger (i.e., HCF reduction/adversary) has runtime $2^{\text{polylog}(\lambda)}$ whereas we assume that the OWF is subexponentially secure, e.g. against $2^{\lambda^{1/2}}$ -time adversaries.

Game 2. k (Switch HCB to random): Let $i \cdot \mathfrak{b}(\lambda) + j := k$ and let $\iota \cdot \mathfrak{b}(\lambda) + j := \kappa$. For each $\kappa \leq k$ the challenger now samples the hardcore bit $h_{\iota,j} \leftarrow \{0,1\}$ at random, for each $\kappa \geq k$ it computes $h_{\iota,j} := \langle x_{\iota,j}^1, x_{\iota,j}^2 \rangle \bmod 2$ as before. For each $k \in \{1, \dots, \mathfrak{a}(\lambda) \cdot \mathfrak{b}(\lambda)\}$ note that in Game 2.($k-1$) we only used the preimage $x_{i,j}$ to directly compute the OWF image $u_{i,j} = \text{injOWF}(x_{i,j}^1) \| x_{i,j}^2$ and the HCB

$h_{i,j} = \langle x_{i,j}^1, x_{i,j}^2 \rangle \pmod{2}$ during VRF.Eval . Thus,

$$|\Pr[G_{2,k}(\lambda) = 1] - \Pr[G_{2,(k-1)}(\lambda) = 1]| \leq \epsilon_{\text{HCF}}^{\text{ind}}(\lambda) . \quad (43)$$

Moreover, note that Game 1 is equal to Game 2.0.

Game 3 (Random $y_{i,j}^0$): This game is Game 2. k for $k := \alpha(\lambda)\mathfrak{b}(\lambda)$ in that the challenger samples $y_{i,j}^0 \leftarrow \{0, 1\}$ for all i, j . Thus, the adversary's view is independent of the underlying bit b , i.e.,

$$\Pr[G_3 = 1] = 1/2 . \quad (44)$$

Overall, we see that

$$\epsilon_{\text{VRF}}^{\text{ipub}}(\lambda) \leq \Pr[\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{ipub}}(\lambda) = 1] - 1/2 \quad (45)$$

$$\leq \alpha(\lambda)\mathfrak{b}(\lambda)\epsilon_{\text{HCF}}^{\text{ind}}(\lambda) \quad (46)$$

$$\leq 2\alpha(\lambda)\mathfrak{b}(\lambda)\epsilon_{\text{injOWF}}^{\text{inv}}(\lambda)^{1/3} \quad (47)$$

where the last inequality follows from Theorem 4. The adversary against the HCB indistinguishability game is $\mathfrak{t}_{\text{HCF}}^{\text{ind}}$ -time-bounded for

$$\mathfrak{t}_{\text{HCF}}^{\text{ind}}(\lambda) \leq \alpha(\lambda)\mathfrak{b}(\lambda)\lambda \cdot \left(\frac{\mathfrak{t}_{\text{injOWF}}(\lambda) + \mathfrak{t}_{\text{CRH.Hash}}(\lambda)}{+2^{\mathfrak{t}_{\text{VRF'}.Eval}(\ell_v(\lambda))}\mathfrak{t}_{\text{VRF'}.Verify}(\ell_v(\lambda))} \right) + \mathfrak{t}_{\text{CRH.Gen}}(\lambda) + \mathfrak{t}_{\text{VRF}}^{\text{ipub}}(\lambda) . \quad (48)$$

By Theorem 4 we have that

$$\mathfrak{t}_{\text{VRF}}^{\text{ipub}}(\lambda) \geq \frac{\mathfrak{t}_{\text{injOWF}}^{\text{inv}}(\lambda)}{3\epsilon_{\text{injOWF}}^{\text{inv}}(\lambda)^{2/3}} - \mathfrak{t}_{\text{CRH.Gen}}(\lambda) - \alpha(\lambda)\mathfrak{b}(\lambda)\lambda \cdot \left(\frac{\mathfrak{t}_{\text{injOWF}}(\lambda) + \mathfrak{t}_{\text{CRH.Hash}}(\lambda)}{+2^{\mathfrak{t}_{\text{VRF'}.Eval}(\ell_v(\lambda))}\mathfrak{t}_{\text{VRF'}.Verify}(\ell_v(\lambda))} \right) . \quad (49)$$

□

We simplify the expressions of Theorems 5 and 6 in a corollary in Section 12.

6.2 Full Unbiasability

Again, correctness and unique provability follow immediately, thus we prove pseudorandomness and full unbiasability for construction.

Construction 2 (Fully Unbiasable VRF). Let injOWF be an injective OWF with poly. many HCBs HCF. Let CRH be a CRH. Let PRF be a PRF. Let VRF' be a VRF. We define a VRF VRF in Figure 3. VRF evaluation (ignoring the VRF proof) is described succinctly by

$$y := ((\text{VRF'}.Eval(\text{sk}, \cdot) \circ \text{CRH.Hash}(\text{hk}, \cdot) \circ \text{injOWF}) \oplus (\text{PRF.Eval}(\cdot, \text{vk}) \circ \text{HCF}))(x) . \quad (50)$$

Theorem 7 (Pseudorandomness). Let injOWF be an injective OWF. Let VRF' be a $(\mathfrak{t}_{\text{VRF'}}^{\text{pr}}, \epsilon_{\text{VRF'}}^{\text{pr}})$ -pseudorandom VRF. Let PRF be a PRF. Let CRH be a $(\mathfrak{t}_{\text{CRH}}^{\text{cr}}, \epsilon_{\text{CRH}}^{\text{cr}})$ -CRH with output length ℓ_v . Then VRF from Construction 2 is $(\mathfrak{t}_{\text{VRF}}, \epsilon_{\text{VRF}})$ -pseudorandom for

$$\epsilon_{\text{VRF}}^{\text{pr}}(\lambda) \geq \mathfrak{t}_{\text{VRF}}^{\text{pr}}(\lambda)^2 \epsilon_{\text{CRH}}^{\text{cr}}(\lambda) + \epsilon_{\text{VRF'}}^{\text{pr}}(\lambda) \quad (51)$$

$$\mathfrak{t}_{\text{VRF}}^{\text{pr}}(\lambda) \leq \frac{\min(\mathfrak{t}_{\text{CRH}}^{\text{cr}}(\lambda) - \lambda^2, \mathfrak{t}_{\text{VRF'}}^{\text{pr}}(\ell_v(\lambda))) - \mathfrak{t}_{\text{CRH.Gen}}(\lambda)}{\mathfrak{t}_{\text{CRH.Hash}}(\lambda) + \mathfrak{t}_{\text{injOWF}}(\lambda) + \mathfrak{t}_{\text{HCF}}(\lambda)} . \quad (52)$$

The proof is essentially analogous to the proof of Theorem 5.

Theorem 8 (Full Unbiasability). Let injOWF be an injective OWF with polynomially many $(\mathfrak{t}_{\text{HCF}}, \epsilon_{\text{HCF}})$ -HCBs. Let VRF' be a perfectly uniquely provable VRF. Let PRF be a selectively $(\mathfrak{t}_{\text{PRF}}^{\text{pr}}, \epsilon_{\text{PRF}}^{\text{pr}})$ -pseudorandom PRF. Let CRH be a CRH with output length ℓ_v . Then VRF is fully $(\mathfrak{t}_{\text{VRF}}^{\text{fub}}, \epsilon_{\text{VRF}}^{\text{fub}})$ -unbiasable for

$$\epsilon_{\text{VRF}}^{\text{fub}}(\lambda) \geq \alpha(\lambda)\mathfrak{b}(\lambda)(\epsilon_{\text{HCF}}^{\text{ind}}(\lambda) + \epsilon_{\text{PRF}}^{\text{pr}}(\lambda^{1/2})) \quad (53)$$

$$t_{\text{VRF}}^{\text{fub}}(\lambda) \leq \min(t_{\text{HCF}}^{\text{inv}}(\lambda), t_{\text{PRF}}^{\text{pr}}(\lambda^{1/2})) - t_{\text{CRH.Gen}}(\lambda) - a(\lambda)b(\lambda)\lambda \cdot \left(t_{\text{injOWF}}(\lambda) + t_{\text{CRH.Hash}}(\lambda) + t_{\text{HCF}}(\lambda) + 2^{t_{\text{VRF'}.Eval}(\ell_v(\lambda))} t_{\text{VRF'}.Verify}(\ell_v(\lambda)) \right). \quad (54)$$

Proof. The basic idea of the proof is analogous to the proof of Theorem 6. The difference is that once we switched the hardcore bits $h_{i,j} := \text{HCF}(x_{i,j})$ to uniformly random bits $h_{i,j} \leftarrow \{0,1\}^{\lambda^{1/2}}$ we also invoke the pseudorandomness of the PRF to switch the values $h'_{i,j} := \text{PRF.Eval}(h_j, \text{vk}_i)$ to independent uniformly random bits $h'_{i,j} \leftarrow \{0,1\}$. Note here that all $h'_{i,j} \leftarrow \{0,1\}$ are indeed independent because all verification keys are different by the definition of the full unbiasedness game.

Let us proceed with the formal proof. We show unbiasedness by a sequence of hybrid games. Let $\mathcal{A}$ be a non-uniform $t_{\text{VRF}}^{\text{fub}}$ -time-bounded adversary for the unbiasedness game in Figure 4.

Game 0 (Original game): This is the original game where the adversary produces the images $y_{i,j}^0$ and proofs $\pi_{i,j}^0$ (given the preimages $x_{i,j}$).

Game 1 (Self-Compute $y_{i,j}^0$ and $\pi_{i,j}^0$): After sampling $x_{i,j} \leftarrow \{0,1\}^\lambda$, the challenger computes the intermediate values $h'_{i,j}, v_j$ as defined in VRF.Eval . Let $\ell_v(\lambda) := |v_j|$. Let $\ell_h(\lambda) := |h_{i,j}|$. Then it brute-forces in time $2^{t_{\text{VRF'}.Eval}(\ell_v(\lambda))} t_{\text{VRF'}.Verify}(\ell_v(\lambda))$ an image $z_{i,j} \in \{0,1\}$ and proof $\pi_{i,j} \in \{0,1\}^{\leq t_{\text{VRF'}.Eval}(\ell_v(\lambda))}$ such that $\text{VRF'}.Verify(\text{vk}_i, v_j, z_{i,j}, \pi_{i,j}) = 1$. It sets $z_{i,j} := \perp$ and $\pi_{i,j} := \perp$ if no such $z_{i,j}, \pi_{i,j}$ exist. The challenger sets $\hat{y}_{i,j}^1 := z_{i,j} \oplus h'_{i,j}$, samples $\hat{y}_{i,j}^0 \leftarrow \{0,1\}^{\ell_v(\lambda)}$ and computes $\hat{z}_{i,j}$ analogously.

We argue that $\beta_{i,j} = 1 \implies y_{i,j}^0 = \hat{y}_{i,j}^0$ (cf. line 9 in Figure 4), i.e., if the adversary's image and proof verify, then the challenger's image equals the adversary's image. This follows directly from the perfect unique provability of VRF' . Consequently, $(y_{i,j}^0)_{i,j} \leq (\hat{y}_{i,j}^0)_{i,j}$ and thus $\rho(y_{1,1}^0, \dots, y_{a(\lambda),b(\lambda)}^0) \leq \rho(\hat{y}_{1,1}^0, \dots, \hat{y}_{a(\lambda),b(\lambda)}^0)$.

We find that

$$\Pr[G_1(\lambda) = 1] = \Pr[G_1(\lambda) = 1 \wedge b = 0] + \Pr[G_1(\lambda) = 1 \wedge b = 1] \quad (55)$$

$$= \Pr[G_1(\lambda) = 1 \wedge b = 0] + \Pr[G_0(\lambda) = 1 \wedge b = 1] \quad (56)$$

$$\geq \Pr[G_0(\lambda) = 1 \wedge b = 0] + \Pr[G_0(\lambda) = 1 \wedge b = 1] \quad (57)$$

$$= \Pr[G_0(\lambda) = 1]. \quad (58)$$

Game 2.j (Switch HCB to random): For each $j \leq j$ the challenger now samples random bits $h'_{1,j}, \dots, h'_{a(\lambda),j} \leftarrow \{0,1\}$. Moreover, it samples the HCBs $h_j \leftarrow \{0,1\}^{\ell_h(\lambda)}$ at random, instead of $h_j := \text{HCF}(x_j)$ as before. Note that in Game 2.(j-1) we only used the preimage x_j to directly compute the OWF image $u_j := \text{injOWF}(x_j)$ and the HCBs $h_j := \text{HCF}(x_j)$ during VRF.Eval . Thus,

$$|\Pr[G_{2,j}(\lambda) = 1] - \Pr[G_{2,(j-1)}(\lambda) = 1]| \leq \epsilon_{\text{HCF}}^{\text{ind}}(\lambda). \quad (59)$$

Moreover, note that Game 1 is equal to Game 2.0.

Game 2.j.i (Sample $h'_{i,j}$): For each $j \leq j$ the challenger now samples $h'_{1,j}, \dots, h'_{a(\lambda),j} \leftarrow \{0,1\}$. Moreover, it samples the HCBs $h_j \leftarrow \{0,1\}^{\ell_h(\lambda)}$ at random, instead of $h_j := \text{HCF}(x_j)$ as before. For each $i \leq i$ the challenger now samples $h'_{i,j} \leftarrow \{0,1\}$, instead of $h'_{i,j} := \text{PRF.Eval}(h_j, \text{vk}_i)$ as before.

Note that the (since Game 2.j.(i-1) uniform) HCBs h_j are only used as the secret key of the PRF for evaluating the PRF on all $\text{vk}_1, \dots, \text{vk}_{a(\lambda)}$. Thus, we can reduce this change from the pseudorandomness of the PRF. Let $\mathcal{R}$ be the following reduction. It receives verification keys $\text{vk}_1, \dots, \text{vk}_{a(\lambda)}$ from the unbiasedness adversary. The reduction

- samples preimages $x_1, \dots, x_{b(\lambda)} \leftarrow \{0,1\}^\lambda$,
- samples HCBs $h_j \leftarrow \{0,1\}^{\ell_h(\lambda)}$,
- sets HCB $h_j := \text{HCF}(x_j)$ for all $j \geq j$,
- samples bits $h'_{1,j}, \dots, h'_{a(\lambda),j} \leftarrow \{0,1\}$ for all $j \leq j$, and
- samples bits $h'_{i,j} \leftarrow \{0,1\}$ for all $i \leq i$,
- queries vk_i to the pseudorandomness challenger to obtain $h'_{i,j}$ for all $i \geq i$,

- sets bits $h'_{\iota,j} := \text{PRF.Eval}(h_j, \text{vk}_\iota)$ for all $\iota \in \{1, \dots, i\}$ and $j \geq j$.

Then it computes the final values $z_{1,1}, \dots, z_{\mathbf{a}(\lambda), \mathbf{b}(\lambda)}$ of the upper branch and returns $y_{\cdot,\cdot} := z_{\cdot,\cdot} \oplus h'_{\cdot,\cdot}$ to the unbiasedness adversary. Finally, the reduction forwards the response of the adversary. Thus,

$$|\Pr[\mathbf{G}_{2,j,i}(\lambda) = 1] - \Pr[\mathbf{G}_{2,j,i-1}(\lambda) = 1]| \leq \epsilon_{\text{PRF}}^{\text{pr}}(\lambda^{1/2}). \quad (60)$$

Lastly, note that Game 2.j equals Game 2.j.0, and that Game 2.j.a(λ) equals Game 2.(j+1).

Game 3 (Random $y_{i,j}^0$): This game is Game 2.j.i for $i := \mathbf{a}(\lambda)$ and $j := \mathbf{b}(\lambda)$ in that the challenger samples $y_{i,j}^0 \leftarrow \{0, 1\}$ for all i, j . Now, the adversary's view is independent of the underlying bit b , i.e.,

$$\Pr[\mathbf{G}_3 = 1] = 1/2. \quad (61)$$

Overall, we see that

$$\epsilon_{\text{VRF}}^{\text{fub}}(\lambda) \leq \Pr[\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{fub}}(\lambda) = 1] - 1/2 \quad (62)$$

$$\leq \mathbf{a}(\lambda)\mathbf{b}(\lambda)(\epsilon_{\text{HCF}}^{\text{ind}}(\lambda) + \epsilon_{\text{PRF}}^{\text{pr}}(\lambda^{1/2})). \quad (63)$$

The adversary in the HCB indistinguishability (resp. pseudorandomness) game is $\mathbf{t}_{\text{HCF}}^{\text{inv}}$ -time-bounded (resp. $\mathbf{t}_{\text{PRF}}^{\text{pr}}$ -time-bounded) for

$$\mathbf{t}_{\text{HCF}}^{\text{inv}}(\lambda) = \mathbf{t}_{\text{PRF}}^{\text{pr}}(\lambda^{1/2}) \leq \mathbf{a}(\lambda)\mathbf{b}(\lambda)\lambda \cdot \left(\mathbf{t}_{\text{injOWF}}(\lambda) + \mathbf{t}_{\text{CRH.Hash}}(\lambda) + \mathbf{t}_{\text{HCF}}(\lambda) \right) + 2^{\mathbf{t}_{\text{VRF}. \text{Eval}}(\ell_v(\lambda))} \mathbf{t}_{\text{VRF}. \text{Verify}}(\ell_v(\lambda)) + \mathbf{t}_{\text{CRH.Gen}}(\lambda) + \mathbf{t}_{\text{VRF}}^{\text{fub}}(\lambda). \quad (64)$$

□

Remark 5 (Notions of Reducibility). Both Constructions 1 and 2 are fully black-box in the sense of [RTV04]. More precisely, all reductions used in the proofs of Theorems 5 to 8 are BBB reductions in the sense of [BBF13].

Remark 6 (On Uniform Security). While we state our results in the non-uniform model, where adversaries have access to a length-dependent advice string, the results of this section hold analogously in the uniform model.

7 Avoiding Superpolynomial Hardness Assumptions via Non-Black-Box Reductions

The goal of this section is to avoid the undesirable assumption of superpolynomial or subexponential security of the underlying primitives that is required in Corollaries 3 and 4. This is relevant in practice, as it directly reflects in the chosen key and proof sizes in deployed systems. Towards this goal, we first introduce the notion of a *latent* VRF where for each (potentially malformed) verification key vk there exists a corresponding evaluation key²⁰ ek that allows to efficiently evaluate the VRF relative to vk . Then, we show how to instantiate a latent VRF. Lastly, we demonstrate how we can use the latency of the underlying VRF to rely only on the polynomial hardness of the OWF in the proof of unbiasedness, e.g. in the proof of Theorem 8.

Definition 1 (Latent VRF). A VRF VRF is (perfectly) latent if there exists a deterministic polynomial-time algorithm VRF.LatEval that on input an evaluation key ek and a preimage x , outputs an image y such that the following holds: For each vk there exists²¹ an evaluation key ek such that for each preimage

²⁰ The evaluation key must obviously not be efficiently computable, otherwise pseudorandomness would be trivially broken. Moreover, the evaluation key need not be in the image of VRF.Gen .

²¹ Naturally, we require that the bit size of ek is polynomially related to the bit size of vk .

x the image $y := \text{VRF.LatEval}(\text{ek}, x)$ verifies, or $\text{VRF.LatEval}(\text{ek}, x) = \perp$ if no image verifies relative to x . Formally,

$$\forall \text{vk} \exists \text{ek} \forall x \left(\begin{array}{l} \text{VRF.LatEval}(\text{ek}, x) \neq \perp \iff \exists \pi : \text{VRF.Verify}(\text{vk}, x, \overbrace{\text{VRF.LatEval}(\text{ek}, x)}^{\text{correct image}}, \pi) = 1 \\ \text{VRF.LatEval}(\text{ek}, x) = \perp \iff \underbrace{\forall y, \pi : \text{VRF.Verify}(\text{vk}, x, y, \pi) = 0}_{\text{no valid image exists}} \end{array} \right) \wedge .$$

In other words, a latent VRF ensures that for every verification key (even those for which no secret key exists) there *exists* a latent evaluation key that allows to evaluate the VRF efficiently.

Constructing latent VRFs. In all known group-based VRF constructions, the verification key consists of group elements. Each verification key has a corresponding secret key (and thus latent key) that consists of the discrete logarithms of these group elements.

We now show that the NIWI-based VRF of Goyal, Hohenberger, Koppula, and Waters [GHK⁺17b] and Bitansky [Bit17; Bit20] is latent if the underlying NIWI is perfectly sound. By relying on subexponential security of the underlying primitives (only assuming a puncturable PRF which is implied by standard OWFs) or assuming a constrained PRF (cf. [GHK⁺17b; Bit20]), we can achieve adaptive pseudorandomness. Since our goal in this section is to avoid superpolynomial hardness, we opt to present the variant with the CPRF. To avoid additional overhead, we omit the explicit definition of a partitioning scheme (or admissible hash function) and instead refer the interested reader to [GHK⁺17b; Bit20] for the full construction.

Construction 3 (NIWI-Based VRF [GHK⁺17b; Bit20]). Let COM be a non-interactive commitment scheme. Let Π_{WI} be a NIWI. Let Hash be a partitioning scheme. Let CPRF be a constrained PRF. We present the VRF VRF in [GHK⁺17b; Bit20] as follows:

VRF.Gen(1^λ)	VRF.Eval(sk, x)
$k \leftarrow \text{CPRF.Gen}(1^\lambda)$	$(k, r_2, r_3) := \text{sk}$
$r_1, r_2, r_3 \leftarrow \{0, 1\}^\lambda$	$y := \text{CPRF.Eval}(k, \text{Hash}(x))$
$c_1 := \text{COM.Commit}(k; r_1)$	$\mathbb{w} := (2, 3, k, k, r_2, r_3)$
$c_2 := \text{COM.Commit}(k; r_2)$	$\pi \leftarrow \Pi_{\text{WI}}.\text{Prove}(\mathbb{w}_{\text{vk}, x, y}, \mathbb{w})$
$c_3 := \text{COM.Commit}(k; r_3)$	return (y, π)
$\text{vk} := (c_1, c_2, c_3)$	
$\text{sk} := (k, r_2, r_3)$	$\text{VRF.Verify}(\text{vk}, x, y, \pi)$
return (vk, sk)	return $\Pi_{\text{WI}}.\text{Verify}(\mathbb{w}_{\text{vk}, x, y}, \pi)$

with statement

$$\mathbb{w}_{\text{vk}, x, y} := \left\{ \begin{array}{l} \exists i \neq j \in \{1, 2, 3\}, \tilde{k}_i, \tilde{k}_j, \tilde{r}_i, \tilde{r}_j : \\ y = \text{PRF.Eval}(\tilde{k}_i; \text{Hash}(x)) = \text{PRF.Eval}(\tilde{k}_j; \text{Hash}(x)) \\ \wedge c_i = \text{COM.Commit}(\tilde{k}_i; \tilde{r}_i) \wedge c_j = \text{COM.Commit}(\tilde{k}_j; \tilde{r}_j) \end{array} \right\}. \quad (65)$$

For the sake of completeness, we restate the result from [GHK⁺17b; Bit20] about their VRF.

Theorem 9 (Selective Version of Theorem 2 in [GHK⁺17b] (Theorem 4.1. in the eprint version [GHK⁺17a])). Let COM be a perfectly binding, computationally hiding non-interactive commitment (Definition 3). Let Π_{WI} be a perfectly sound, computationally witness-indistinguishable NIWI (Definition 4). Let Hash be a partitioning scheme. Let CPRF be a selectively pseudorandom CPRF (Def-

inition 8) that supports the (puncturing) constraint class $\mathcal{C} := \{\{0, 1\}^\lambda \setminus \{x\} \mid x \in \{0, 1\}^\lambda\}$.²² Then the VRF from Construction 3 is selectively pseudorandom and uniquely provable.

As usual (by guessing the challenge preimage) we can also achieve adaptive pseudorandomness from Theorem 9 when assuming subexponential security of the underlying primitives. Alternatively, we can assume a CPRF that supports the constraint class of a partitioning scheme to get adaptive pseudorandomness from polynomially secure primitives. Such CPRFs are known from lattice-based assumptions [BV15], indistinguishability obfuscation [BZ14], or DDH [Bit20].

Theorem 10 (Theorem 2 in [GHK⁺17b] (Theorem 4.1. in the eprint version [GHK⁺17a])). Let COM be a perfectly binding, computationally hiding non-interactive commitment (Definition 3). Let Π_{WI} be a perfectly sound, computationally witness-indistinguishable NIWI (Definition 4). Let Hash be a partitioning scheme. Let CPRF be an adaptively pseudorandom CPRF (Definition 8) that supports the constraint class of partitioning scheme Hash. Then the VRF from Construction 3 is adaptively pseudorandom and uniquely provable.

We show that their VRF is also latent.

Lemma 3 (Latency of the [GHK⁺17b; Bit20] VRF). Let COM be a perfectly binding non-interactive commitment. Let Π_{WI} be a perfectly sound NIWI. Let Hash be a partitioning scheme. Let CPRF be a PRF. Then the VRF from Construction 3 is latent.

Proof Sketch. Let $\text{vk} = (c_1, c_2, c_3)$ be any verification key. Because COM is perfectly binding, there exist three unique²³ k_1, k_2, k_3 and some r_1, r_2, r_3 such that $c_1 = \text{COM.Commit}(k_1; r_1)$, $c_2 = \text{COM.Commit}(k_2; r_2)$, and $c_3 = \text{COM.Commit}(k_3; r_3)$. For each verification key vk we define the evaluation key as $\text{ek} = (k_1, k_2, k_3)$ and the latent evaluation $\text{VRF.LatEval}(\text{ek}, x) := \text{maj}_i(\text{PRF.Eval}(k_i, \text{Hash}(x)))$.²⁴ If an image y verifies relative to a preimage x , then by the perfect soundness of the NIWI it follows that the image must be the majority of the three PRF evaluations. $\square$

Using latent VRFs. Now, we prove the analog of Theorem 8 with an underlying latent VRF (the analog of Theorem 6 holds similarly for unbiasedness o.i.p.).

Theorem 11 (Full Unbiasability with Latent VRF). Let injOWF be an injective OWF with polynomially many $(t_{\text{HCF}}, \epsilon_{\text{HCF}})$ -HCBs. Let VRF' be a (perfectly) uniquely provable, (perfectly) latent VRF. Let PRF be a selectively $(t_{\text{PRF}}^{\text{pr}}, \epsilon_{\text{PRF}}^{\text{pr}})$ -pseudorandom PRF. Let CRH be the trivial (identity) CRH. Then VRF from Construction 2 is fully $(t_{\text{VRF}}^{\text{fub}}, \epsilon_{\text{VRF}}^{\text{fub}})$ -unbiasable for

$$\epsilon_{\text{VRF}}^{\text{fub}}(\lambda) \geq \alpha(\lambda)\beta(\lambda)(\epsilon_{\text{HCF}}^{\text{ind}}(\lambda) + \epsilon_{\text{PRF}}^{\text{pr}}(\lambda)) \quad (66)$$

$$t_{\text{VRF}}^{\text{fub}}(\lambda) \leq \min(t_{\text{HCF}}^{\text{inv}}(\lambda), t_{\text{PRF}}^{\text{pr}}(\lambda)) - t_{\text{CRH.Gen}}(\lambda) - \alpha(\lambda)\beta(\lambda)\lambda \cdot \left(t_{\text{injOWF}}(\lambda) + t_{\text{CRH.Hash}}(\lambda) + t_{\text{HCF}}(\lambda) + t_{\text{VRF}'.\text{LatEval}}(\lambda) \right). \quad (67)$$

Proof Sketch. Let $\mathcal{A}$ be the non-uniform adversary against the (full) unbiasedness game with the maximal winning probability. Such an adversary $\mathcal{A}$ exists (and we can assume it is deterministic) because we consider non-uniform adversaries.²⁵ Recall that in the unbiasedness game (cf. Figure 4) the adversary first produces a state st , a predicate ρ and verification keys $\text{vk}_1, \dots, \text{vk}_{\alpha(\lambda)}$. Importantly, these verification keys do not depend on any external randomness, e.g. from the challenger. Hence, for each security parameter λ the verification keys vk_i are fixed. By Construction 2 this also fixes the verification keys vk'_i of the underlying *latent* VRF VRF' . By the latency of VRF' this fixes (not necessarily

²² Such a CPRF follows from any PRF or OWF via [HIL⁺99; GGM86].

²³ Define $k_i := \perp$ if no opening exists.

²⁴ The image is $\perp$ if there is no majority.

²⁵ This follows via an averaging argument by fixing the $\mathcal{A}$'s randomness non-uniformly.

unique) evaluation keys ek'_i for each verification key vk'_i . We now use these fixed evaluation keys ek'_i and hardcode them into the modified challenger of the hybrid arguments. We modify Game 1 as follows: Instead of brute-forcing the image $z_{i,j}$ and proof $\pi_{i,j}$, we efficiently compute the image using the latent evaluation key ek'_i as $z_{i,j} := \text{VRF}'.\text{LatEval}(\text{ek}'_i, x_{i,j})$. Note that functionally this game is equivalent to Game 1 in the proof of Theorem 8. This is because if the preimage $x_{i,j}$ has a valid image $z_{i,j} \neq \perp$, then $\text{VRF}'.\text{LatEval}$ will produce it, otherwise it will produce $\perp$ as before. The remaining hybrids are unchanged. Thus, the only change is the efficiency in computing the images $z_{i,j}$. Consequently, we replace the term $2^{\text{t}_{\text{VRF}'.\text{Eval}}(\ell_v(\lambda))} \text{t}_{\text{VRF}'.\text{Verify}}(\ell_v(\lambda))$ (the at least superpolynomial time for brute-forcing the image and proof) by the term $\text{t}_{\text{VRF}'.\text{LatEval}}(\lambda)$ (the polynomial time for computing the image using the latent evaluation key). $\square$

Remark 7 (No Need for CRH). We emphasize that if the underlying VRF VRF' is latent, then we don't need to invoke the CRH at all! In Theorem 11 we simply use the identity function as the CRH. Recall that in Theorems 6 and 8 the purpose of the CRH was to reduce the time necessary to brute-force the image and proof in the formal proofs. This compression is not necessary anymore because we can hardcode the evaluation keys corresponding to the adversary's verification keys to efficiently compute the VRF images. This non-black-box trick is what allows us to avoid relying on superpolynomial security assumptions.

Corollary 1 (Unbias. VRF via [GHK⁺17b; Bit20]). A VRF with polynomial (adaptive) pseudorandomness and full unbiasedness can be constructed from the following assumptions:

- an injective OWF with poly. security, a perfectly sound NIWI with poly. security, and a CPRF with constraints for a partitioning scheme (as in Theorem 10) with poly. security, or
- an injective OWF with subexp. security, and a perfectly sound NIWI with subexp. security.

Proof Sketch. A visual outline of the proof (of the second statement) is given in Figure 9. An injective OWF implies a perfectly binding non-interactive commitment (by Lemma 2). A perf. binding commitment, a perf. sound NIWI and a constrained PRF (all with poly. security) imply a latent VRF with poly. adaptive pseudorandomness (by Lemma 3 and Theorem 10). Alternatively, an injective OWF and a perf. sound NIWI (both with subexp. security) imply a latent VRF with subexp. selective pseudorandomness (by Theorem 9) which—in turn—implies a latent VRF with poly. adaptive pseudorandomness (by guessing the challenge preimage). Finally, an injective OWF with polynomially many HCBs and poly. security, and a latent VRF with poly. adaptive security imply a VRF with poly. adaptive pseudorandomness and full unbiasedness (by Theorem 11). $\square$

Remark 8 (Notions of Reducibility). While the reductions in the proofs of Theorems 6 and 8 are of type BNB in the sense of [BBF13] (or BBB when additionally assuming pseudorandomness of the HCF) the reduction used in the proof of Theorem 11 is a BNN reduction (or semi-BB in the sense of [RTV04]). This means that

- Constructions 1 and 2 use the underlying (latent) VRF in a black-box way,
- the unbiasedness reduction in the proof of Theorem 11 depends (in a non-black-box way) on the code²⁶ of the unbiasedness adversary,
- the unbiasedness reduction in the proof of Theorem 11 uses the underlying (latent) VRF in a non-black-box way.

Remark 9 (On Uniform Security). Because we need to hardcode the evaluation keys into the unbiasedness reduction in the proof of Theorem 11, we need to fix the adversary's randomness non-uniformly. Hence, in contrast to the results of Section 6 the results of this section hold only in the non-uniform model.

²⁶ The reduction has the evaluation keys hardcoded that correspond to the adversary's verification keys.

We leave it as an open problem to show unbiasedability under polynomial security assumptions in the uniform model.

8 Definitional Issues of Unbiasedability in [GS24]

In this section we discuss several definitional issues concerning the unbiasedability definition of Giunta and Stewart [GS24] (as defined in Figure 4).

- **Falsifiability:** The unbiasedability definition of [GS24] *may* not be efficiently falsifiable in the sense of [Nao03; GW11]. On a high-level, this means that given a transcript of the unbiasedability game, it is not possible to verify in polynomial time whether the game was won (by the transcript). The reason is that we don’t know how to check whether the predicate ρ (given by the adversary in the first flow) is monotone (in the sense of [GS24]). In practice, this should not be a problem because typically the predicate of interest is fixed by the application, e.g. in leader election the predicate is a threshold function of the minimum of all valid images.
As a workaround, unbiasedability could be parameterized by such a fixed monotone predicate, making the notion efficiently falsifiable, or one only quantifies over all “admissible” adversaries that produce monotone predicates.
- **Unconditional adversary:** We cannot formulate an unconditional but inefficient adversary for the full unbiasedability notion of [GS24]. Note that many cryptographic games like PRG/PRF pseudorandomness, or onewayness of OWFs have an unconditional but inefficient adversary that breaks the game with maximal advantage. For example, in the case of PRG pseudorandomness, an inefficient adversary can simply evaluate the PRG on all possible seeds and check whether any PRG output matches its challenge. In general, for unbiasedability o.i.p. such an adversary can indeed *not* exist²⁷ because the real distribution may be *equal* to the random distribution. In contrast, in the full unbiasedability game the real and the random distribution of images have statistical distance at least $1 - 2^{-(\lambda - a(\lambda)^{L_y(\lambda)})^{b(\lambda)}}$. Thus, intuitively an unbounded adversary *should* be able to distinguish the two distributions. However, in the notion of Giunta and Stewart those two distributions are distinguished by the *efficient* (PPT computable) predicate ρ even if the adversary itself is unbounded. It is not clear if an unbounded adversary can produce an efficient predicate that distinguishes the two (statistically distant) distributions.
- **Predicate runtime:** Giunta and Stewart state that the predicate should be “PPT computable”. They don’t specify whether the polynomial runtime bound of the predicate can depend on the adversary, or whether it is (universally) fixed in advance. In the former case, the challenger cannot have a fixed polynomial runtime because it needs to evaluate the predicate. In particular, the predicate’s runtime could be bounded by $\lambda^{|A|}$. In the latter case, the effective distinguisher (the predicate) only has some *fixed* polynomial runtime (as opposed to arbitrary polynomial runtime) which significantly weakens the security statement as such.

Remark 10 (Fixed Unbiasedability Notion). For our following results we want to fix a rigorous notion of unbiasedability as described by Giunta and Stewart.

We only quantify over “admissible” adversaries that produce monotone predicates whose runtime is bounded by the runtime of the adversary. This means that the unbiasedability game is falsifiable but not efficiently falsifiable because the runtime needed to evaluate the predicate is not an a priori fixed polynomial but can depend on the adversary.²⁸

We propose an alternative definition of unbiasedability that subsumes the notion of [GS24] but avoids the above issues.

²⁷ In particular for VRPs.

²⁸ Efficient falsifiability means that the challenger has a fixed polynomial runtime.

Definition 11 ((Two-Stage) Full Unbiasability). We say a uniquely provable VRF VRF is two-stage (fully) $(t, \epsilon, \mathbf{a}, \mathbf{b})$ -unbiasable if for each non-uniform, t -time-bounded, (two-stage) monotone adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ its unbiasedability advantage is bounded by

$$2 \Pr \left[\text{Exp}_{\text{VRF}, \mathcal{A}}^{2\text{sfub}}(\lambda) = 1 \right] - 1 \leq \epsilon(\lambda) \quad (68)$$

with the unbiasedability game defined in Figure 5. We say the adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is (partially) monotone, iff²⁹

$$\forall \mathbf{vk}', y', r_{\mathcal{A}}, z \in \{0, 1\}^*, S \subseteq \{1, \dots, |z|\} : \Pr_{r_{\mathcal{A}}} [\mathcal{A}_2(\mathbf{vk}', y', z; r_{\mathcal{A}}) = 1] \geq \Pr_{r_{\mathcal{A}}} [\mathcal{A}_2(\mathbf{vk}', y', z'; r_{\mathcal{A}}) = 1] \quad (69)$$

where $z'_i := \begin{cases} z_i & \text{if } i \in S \\ \perp & \text{if } i \notin S \end{cases}$.

To differentiate between unbiasedability notions we call the notion of [GS24] (full) GS-unbiasability (resp. o.i.p.), whereas we refer to the notion of Definition 11 as (full) 2s-unbiasability (resp. o.i.p.). The main difference between our 2s-unbiasability notion and GS-unbiasability is that our notion requires indistinguishability of the images $z_{i,j}$ under adversarial verification keys *even given honestly generated images $y'_{i,j}$ for the same preimages x_j* .

On an intuitive level, this gives the adversary more power, and thus the 2s-unbiasability notion is at least as strong as GS-unbiasability. We argue that our notion better captures application scenarios where preimages are not only evaluated on adversarially chosen verification keys but also on honestly generated verification keys. For example, in the leader election application, the adversary may only own a certain fraction of the stake, i.e., the adversary controls only a fraction of the verification keys. However, the common preimage x is evaluated by all users, including the honest users whose images are captured as y' .

Perhaps unsurprisingly, we show that 2s-unbiasability and GS-unbiasability (as fixed in Remark 10) are actually equivalent in the following sense.

Theorem 12 (Equivalence of GS- and 2s-Unbiasability). Let VRF be a VRF. There exists a fixed polynomial poly (depending on VRF) s.t. the following holds:

- If VRF is $(t_{2\text{sfub}}, \epsilon_{2\text{sfub}}, \mathbf{a}, \mathbf{b})$ -2s-unbiasable, then it is $(t_{\text{fub}}, \epsilon_{\text{fub}}, \mathbf{a}, \mathbf{b})$ -GS-unbiasable where

$$t_{2\text{sfub}}(\lambda) \geq 3t_{\text{fub}}(\lambda) + \mathbf{a}(\lambda)\mathbf{b}(\lambda) \text{poly}(\lambda) \quad \text{and} \quad \epsilon_{\text{fub}}(\lambda) := \epsilon_{2\text{sfub}}(\lambda) . \quad (70)$$

- If VRF is $(t_{\text{fub}}, \epsilon_{\text{fub}}, \mathbf{a}, \mathbf{b})$ -GS-unbiasable and $(t_{\text{pr}}, \epsilon_{\text{pr}})$ -pseudorandom, then it is $(t_{2\text{sfub}}, \epsilon_{2\text{sfub}}, \mathbf{a}, \mathbf{b})$ -2s-unbiasable where

$$t_{\text{fub}}(\lambda) \geq t_{2\text{sfub}}(\lambda) + \mathbf{a}(\lambda)\mathbf{b}(\lambda) \text{poly}(\lambda) \quad \text{and} \quad \epsilon_{2\text{sfub}}(\lambda) \geq \epsilon_{\text{fub}}(\lambda) - \epsilon_{\text{pr}}(\lambda) \quad \text{and} \quad t_{\text{pr}}(\lambda) := t_{2\text{sfub}}(\lambda) . \quad (71)$$

Proof. For simplicity and w.l.o.g. we assume the predicate ρ is deterministic.

2s-unbiasability $\implies$ GS-unbiasability. Let $\mathcal{A}^{\text{fub}}$ be a GS-unbiasability adversary. We define a 2s-unbiasability adversary $\mathcal{A}^{2\text{sfub}}$ as follows: The first stage $\mathcal{A}_1^{2\text{sfub}}(1^\lambda; r_{\mathcal{A}})$ computes $(p, \mathbf{vk}) := \mathcal{A}^{\text{fub}}(1^\lambda; r_{\mathcal{A}})$ and outputs it to the 2s-unbiasability challenger. The 2s-unbiasability challenger responds with preimages $x_1, \dots, x_{\mathbf{b}(\lambda)}$. The 2s-unbiasability adversary $\mathcal{A}_1^{2\text{sfub}}(x_1, \dots, x_{\mathbf{b}(\lambda)}; r_{\mathcal{A}})$ then computes the images $(y_{1,1}, \pi_{1,1}), \dots, (y_{\mathbf{a}(\lambda), \mathbf{b}(\lambda)}, \pi_{\mathbf{a}(\lambda), \mathbf{b}(\lambda)}) := \mathcal{A}^{\text{fub}}(x_1, \dots, x_{\mathbf{b}(\lambda)}; r_{\mathcal{A}})$ and outputs them to the 2s-unbiasability challenger. In the second stage, the 2s-unbiasability adversary $\mathcal{A}_2^{2\text{sfub}}(\mathbf{vk}'_1, \dots, \mathbf{vk}'_{\mathbf{a}(\lambda)}, y'_{1,1}, \dots, y'_{\mathbf{a}(\lambda), \mathbf{b}(\lambda)}, z_{1,1}, \dots, z_{\mathbf{a}(\lambda), \mathbf{b}(\lambda)}; r_{\mathcal{A}})$ computes $(p, \mathbf{vk}) := \mathcal{A}^{\text{fub}}(1^\lambda; r_{\mathcal{A}})$ and outputs $\rho(z_{1,1}, \dots, z_{\mathbf{a}(\lambda), \mathbf{b}(\lambda)})$ to the GS-unbiasability challenger (ignoring the $\mathbf{vk}'_i$ and y'_i).

²⁹ Note that here we use short notation by omitting indices, e.g. $\mathbf{vk}'$ to mean $\mathbf{a}$ many verification keys.

$\text{Exp}_{\text{VRF}, \mathcal{A}=(\mathcal{A}_1, \mathcal{A}_2)}^{2\text{sfub}}(\lambda)$	
1 :	// sample randomness
2 :	$r_{\mathcal{A}} \leftarrow \{0, 1\}^{t_{\mathcal{A}}(\lambda)}$
3 :	$r_{\text{VRF.Gen}}^1, \dots, r_{\text{VRF.Gen}}^{a(\lambda)} \leftarrow \{0, 1\}^{t_{\text{VRF.Gen}}(\lambda)}$
4 :	// get adversarial verification keys
5 :	$(\text{vk}_1, \dots, \text{vk}_{a(\lambda)}) := \mathcal{A}_1(1^\lambda; r_{\mathcal{A}})$
6 :	// sample random preimages
7 :	$x_1, \dots, x_{b(\lambda)} \leftarrow \{0, 1\}^\lambda$
8 :	// sample random images
9 :	$y_{1,1}^1, \dots, y_{a(\lambda), b(\lambda)}^1 \leftarrow \{0, 1\}^{\ell_y(\lambda)}$
10 :	// get images for adversarial verification keys
11 :	$((y_{1,1}^0, \pi_{1,1}^0), \dots, (y_{a(\lambda), b(\lambda)}^0, \pi_{a(\lambda), b(\lambda)}^0)) := \mathcal{A}_1(x_1, \dots, x_{b(\lambda)}; r_{\mathcal{A}})$
12 :	$b \leftarrow \{0, 1\}$
13 :	// generate honest verification keys
14 :	for $i \in \{1, \dots, a(\lambda)\}$
15 :	$(\text{vk}'_i, \text{sk}'_i) := \text{VRF.Gen}(1^\lambda; r_{\text{VRF.Gen}}^i)$
16 :	for $i \in \{1, \dots, a(\lambda)\}, j \in \{1, \dots, b(\lambda)\}$
17 :	$\beta_{i,j} := \text{VRF.Verify}(\text{vk}'_i, x_j, y_{i,j}^0, \pi_{i,j}^0)$
18 :	// set valid images
19 :	if $\beta_{i,j} = 1$ then
20 :	$z_{i,j} := y_{i,j}^b$
21 :	// overwrite invalid images
22 :	else
23 :	$z_{i,j} := \perp$
24 :	// evaluate images for honest verification keys
25 :	$(y'_{i,j}, \pi'_{i,j}) := \text{VRF.Eval}(\text{sk}'_i, x_j; 0^\lambda)$
26 :	return $\mathcal{A}_2(\text{vk}'_1, \dots, \text{vk}'_{a(\lambda)}, y'_{1,1}, \dots, y'_{a(\lambda), b(\lambda)}, z_{1,1}, \dots, z_{a(\lambda), b(\lambda)}; r_{\mathcal{A}}) \oplus b$

Fig. 5: Alternative (full) two-stage unbiasedness game. $\text{Exp}_{\text{VRF}, \mathcal{A}}^{2\text{sfub}}$ denotes the full unbiasedness game for a verifiable random function VRF and adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. Note that both stages are run with the same randomness $r_{\mathcal{A}}$. In contrast to the notion of Giunta and Stewart [GS24] the distinguisher $\mathcal{A}_2$ additionally gets honestly generated verification keys vk'_i and images $y'_{i,j}$ for the same preimages x_j .

The success probability of the 2s-unbiasability adversary is the same as for the GS-unbiasability adversary. The runtime of the 2s-unbiasability adversary is at most thrice the runtime of the GS-unbiasability adversary plus the time to generate the (honest) verification keys, the (honest) images, verify images, and evaluate the predicate (at most the runtime of the GS-unbias adversary). Letting poly be a polynomial that exceeds $\mathbf{t}_{\text{VRF.Gen}}$, $\mathbf{t}_{\text{VRF.Eval}}$ and $\mathbf{t}_{\text{VRF.Verify}}$ we find that $\mathbf{t}_{\text{fub}}(\lambda) \leq 3\mathbf{t}_{2\text{sfub}}(\lambda) + \mathbf{a}(\lambda)\mathbf{b}(\lambda)\text{poly}(\lambda)$.

GS-unbiasability $\implies$ 2s-unbiasability. Let $\mathcal{A}^{2\text{sfub}}$ be a 2s-unbiasability adversary. We define a GS-unbiasability adversary $\mathcal{A}^{\text{fub}}$ as follows: $\mathcal{A}^{\text{fub}}(1^\lambda; r_{\mathcal{A}})$

- parses its randomness as $(r'_{\mathcal{A}}, r_{\text{VRF.Gen}}^1, \dots, r_{\text{VRF.Gen}}^{\mathbf{a}(\lambda)}, y'_{1,1}, \dots, y'_{\mathbf{a}(\lambda), \mathbf{b}(\lambda)}) := r_{\mathcal{A}}$,
- computes (adversarial) verification keys $(\mathbf{vk}_1, \dots, \mathbf{vk}_{\mathbf{a}(\lambda)}) := \mathcal{A}_1^{2\text{sfub}}(1^\lambda; r'_{\mathcal{A}})$,
- computes (honest) verification keys $(\mathbf{vk}'_i, \mathbf{sk}'_i) := \text{VRF.Gen}(1^\lambda; r_{\text{VRF.Gen}}^i)$,
- computes the predicate $\rho(z_{1,1}, \dots, z_{\mathbf{a}(\lambda), \mathbf{b}(\lambda)}) := \mathcal{A}_2^{2\text{sfub}}(\mathbf{vk}'_1, \dots, \mathbf{vk}'_{\mathbf{a}(\lambda)}, y'_{1,1}, \dots, y'_{\mathbf{a}(\lambda), \mathbf{b}(\lambda)}, z_{1,1}, \dots, z_{\mathbf{a}(\lambda), \mathbf{b}(\lambda)}; r'_{\mathcal{A}})$.

In the second flow the GS-unbiasability adversary $\mathcal{A}^{\text{fub}}(x_1, \dots, x_{\mathbf{b}(\lambda)}; r_{\mathcal{A}})$ computes the images and proofs $((y_{1,1}, \pi_{1,1}), \dots, (y_{\mathbf{a}(\lambda), \mathbf{b}(\lambda)}, \pi_{\mathbf{a}(\lambda), \mathbf{b}(\lambda)})) := \mathcal{A}_1^{2\text{sfub}}(x_1, \dots, x_{\mathbf{b}(\lambda)}; r'_{\mathcal{A}})$ and outputs them.

The runtime of the GS-unbiasability adversary is at most the runtime of the 2s-unbiasability adversary plus the time to generate the (honest) verification keys, the (honest) images, verify images, and compute (the description of) the predicate. Letting poly be a polynomial that exceeds $\mathbf{t}_{\text{VRF.Gen}}$ and $\mathbf{t}_{\text{VRF.Verify}}$ we find that $\mathbf{t}_{2\text{sfub}}(\lambda) \leq \mathbf{t}_{\text{fub}}(\lambda) + \mathbf{a}(\lambda)\mathbf{b}(\lambda)\text{poly}(\lambda)$.

Note that the only difference between the view of $\mathcal{A}_2^{\text{fub}}$ as described above and the view of $\mathcal{A}_2^{2\text{sfub}}$ in the original 2s-unbiasability game is that in the above game the (honest) images $y'_{i,j}$ are random whereas in the original 2s-unbiasability game they are real evaluations $(y'_{i,j}, \pi'_{i,j}) := \text{VRF.Eval}(\mathbf{sk}'_i, x_j; 0^\lambda)$. Because $\mathcal{A}_2^{2\text{sfub}}$ is $\mathbf{t}_{2\text{sfub}}$ -time-bounded and VRF is $(\mathbf{t}_{\text{pr}}, \epsilon_{\text{pr}})$ -pseudorandom, the difference in the output distribution of $\mathcal{A}_2^{2\text{sfub}}$ between the game above and the original 2s-unbiasability game is at most ϵ_{pr} . This follows from a straightforward reduction from the pseudorandomness of VRF. Hence, $\epsilon_{\text{fub}}(\lambda) \geq \epsilon_{2\text{sfub}}(\lambda) - \epsilon_{\text{pr}}(\lambda)$. $\square$

Remark 11. This means that GS-unbiasability implies 2s-unbiasability³⁰; in particular all results from Sections 6, 7 and 13 carry over to our 2s-unbiasability notion.

If conceptual simplicity is desirable, then the honestly generated verification keys $\mathbf{vk}'$ and images y' can be omitted from the 2s-unbiasability game. In this case, the 2s-unbiasability game is equivalent to the GS-unbiasability game, even without assuming pseudorandomness.

9 Additional Properties of Our Construction

In this section we discuss several additional properties that our constructions offer.

Key-Homomorphism. The notion of key-homomorphic VRFs (KH-VRF) was introduced by Malavolta [Mal24]. In KH-VRFs the (secret) key space $\mathcal{K}$, codomain $\mathcal{Y}$, and proof space $\mathcal{P}$ are endowed with a group structure $((\mathcal{K}, \otimes), (\mathcal{Y}, \otimes), (\mathcal{P}, \otimes))$ such that for all $\mathbf{sk}_1, \mathbf{sk}_2 \in \mathcal{K}$ and $x \in \mathcal{X}$ it holds that

$$\text{VRF.Eval}(\mathbf{sk}_1 \otimes \mathbf{sk}_2, x) = \text{VRF.Eval}(\mathbf{sk}_1, x) \otimes \text{VRF.Eval}(\mathbf{sk}_2, x) \in \mathcal{Y} \times \mathcal{P}. \quad (72)$$

As described by Malavolta [Mal24] this property immediately implies a t -out-of- n threshold VRF (as in [BLM⁺13]). The basic idea is to t -out-of- n share the VRF secret key $\mathbf{sk}$ among all n users, then each user evaluates the VRF preimage with their own local secret key share $\mathbf{sk}_i$. With t such partial evaluations (images and proofs) the overall VRF image y and proof π can be publicly homomorphically recombined via Lagrange interpolation. We remark that in this context of recombining the partial VRF evaluations the preimage x can—in principle—be publicly known.

³⁰ For pseudorandom VRFs.

If the underlying VRF is key-homomorphic, then our Constructions 1 and 2 fulfill a slightly weaker form of key-homomorphism than the one described above. For this exposition we choose Construction 1 (with the identity hash function), but the analogous statement holds for Construction 2 (with the identity hash function). Let $\mathcal{K}'$, $\mathcal{Y}'$ and $\mathcal{P}'$ be the secret key space, codomain and proof space of the underlying VRF VRF' with group structures $(\mathcal{K}, \otimes')$, $(\mathcal{Y}, \otimes')$ and $(\mathcal{P}, \otimes')$. Let $\mathcal{X}$, $\mathcal{K} = \mathcal{K}'$, $\mathcal{Y}$, and $\mathcal{P} = \mathcal{P}'$ be the domain, secret key space, codomain and proof space of the resulting VRF with group structures $(\mathcal{K}, \otimes')$ and $(\mathcal{P}, \otimes')$. Note that our resulting VRF does *not* have a group structure on the codomain $\mathcal{Y}$ but instead there exists an (efficiently computable) map

$$\phi : \mathcal{X} \times \mathcal{Y} \times \mathcal{Y} \rightarrow \mathcal{Y} : (x, y_1, y_2) \mapsto y_3 := ((y_1 \oplus \text{HCF}(x)) \otimes' (y_2 \oplus \text{HCF}(x))) \oplus \text{HCF}(x) \quad (73)$$

that additionally to the two VRF images takes a VRF preimage as input.³¹ For our resulting VRF VRF , for all $\text{sk}_1, \text{sk}_2 \in \mathcal{K}$ and $x \in \mathcal{X}$ it holds that

$$\begin{aligned} y_3 = \phi(x, y_1, y_2) \quad \text{where} \quad & \begin{aligned} (y_1, \pi_1) &\leftarrow \text{VRF.Eval}(\text{sk}_1, x) \\ (y_2, \pi_2) &\leftarrow \text{VRF.Eval}(\text{sk}_2, x) \\ (y_3, \pi_3) &\leftarrow \text{VRF.Eval}(\text{sk}_1 \otimes' \text{sk}_2, x) \end{aligned} \end{aligned} \quad (74)$$

The only difference to the key-homomorphism notion of [Mal24] is that the computation $y_3 = \phi(x, y_1, y_2)$ requires the knowledge of the preimage x whereas in Malavolta's notion the image can be computed as $y_3 = y_1 \otimes' y_2$ without the preimage. We note that for the main application of threshold VRFs this is not a restriction because the preimage at which the threshold VRF is evaluated can be publicly known at reconstruction time.

Aggregability. Again, Malavolta [Mal24] describes the notion of public³² aggregability for VRFs. In this notion multiple verification keys $\text{vk}_1, \dots, \text{vk}_n$, images $y_1, \dots, y_n$ and proofs $\pi_1, \dots, \pi_n$ for a single preimage x can be aggregated into a single verification key $\tilde{\text{vk}}$, image $\tilde{y}$ and proof $\tilde{\pi}$. Malavolta requires four properties: aggregate correctness, aggregate pseudorandomness, aggregate binding and aggregate uniqueness. As mentioned in Section 1.2 the construction and hence the definitions of [Mal24] are in the SRS model. For the reader's convenience we reproduce (and simplify³³) the definitions of [Mal24] in the standard model.

Definition 12 (Aggregability). A VRF VRF is aggregable if there exist two deterministic algorithms Agg and AggVerify :

- Agg on input preimage x and verification key-image-proof triples $(\text{vk}_i, y_i, \pi_i)_i$ outputs a verification key $\tilde{\text{vk}}$, an image $\tilde{y}$ and a proof $\tilde{\pi}$,
- AggVerify on input a list of verification keys $(\text{vk}_i)_i$, preimage x , image $\tilde{y}$ and proof $\tilde{\pi}$ outputs 1 if the image $\tilde{y}$ is valid for the preimage x under the aggregation of the verification keys $(\text{vk}_i)_i$.

We define several properties for aggregable VRFs:

Aggregate Correctness. For any $n \in \mathbb{N}$, any security parameter λ and any preimage $x \in \{0, 1\}^\lambda$ it holds that

$$\Pr \left[\begin{aligned} & (\text{vk}_1, \text{sk}_1), \dots, (\text{vk}_n, \text{sk}_n) \leftarrow \text{VRF.Gen}(1^\lambda) \\ & (y_i, \pi_i) \leftarrow \text{VRF.Eval}(\text{sk}_i, x) \quad : \text{AggVerify}((\text{vk}_i)_i, x, \tilde{y}, \tilde{\pi}) = 1 \\ & (\tilde{\text{vk}}, \tilde{y}, \tilde{\pi}) := \text{Agg}(x, (\text{vk}_i, y_i, \pi_i)_i) \end{aligned} \right] = 1. \quad (75)$$

³¹ Here, $\oplus$ means the bitwise XOR operation and HCF is a hardcore function of the injective one-way function injOWF used in the construction.

³² For context, Cohen, Goldwasser, and Vaikuntanathan [CGV15] introduce aggregate PRFs, while and Liang, Banegas, and Mitrokovska [LBM20] introduce privately aggregate VRFs.

³³ We also fix some typos in the original definitions.

Aggregate Pseudorandomness. We say an aggregable VRF VRF is (t, ϵ) -aggregate pseudorandom if for each non-uniform (stateful) t -time-bounded adversary $\mathcal{A}$ it holds that

$$2 \Pr \left[\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{agg-pr}}(\lambda) = 1 \right] - 1 \leq \epsilon(\lambda) . \quad (76)$$

where the experiment is defined in Figure 6. Aggregate pseudorandomness means that the aggregate VRF image is pseudorandom even if only a single aggregated image is pseudorandom.

Aggregate Binding. We say an aggregable VRF VRF is (t, ϵ) -aggregate binding if for each uniform t -time-bounded adversary $\mathcal{A}$ it holds that

$$\Pr \left[\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{agg-bd}}(\lambda) = 1 \right] \leq \epsilon(\lambda) . \quad (77)$$

Aggregate binding means that no adversary can find a set of verification keys, a preimage x and a different set of VRF images y'_i (relative to the preimage x and verification keys) that when aggregated is also accepted relative to the honestly generated aggregate proof. We stress that—due to a simple compression argument—this notion cannot be fulfilled without a CRS against non-uniform adversaries.

Aggregate Uniqueness.³⁴ For any $(\text{vk}_i, y_i, \pi_i)_{i \in [n]}$, any x and any y^1, π^1, y^2, π^2 it holds that

$$\text{VRF.Verify}(\tilde{\text{vk}}, x, y^1, \pi^1) = 1 \wedge \text{VRF.Verify}(\tilde{\text{vk}}, x, y^2, \pi^2) = 1 \implies y^1 = y^2 \quad (78)$$

where $(\tilde{\text{vk}}, \tilde{y}, \tilde{\pi}) := \text{Agg}(x, (\text{vk}_i, y_i, \pi_i)_i)$. Aggregate uniqueness is a weaker form of unique provability that only requires uniqueness for aggregate verification keys.

Remark 12 (Subtleties in Aggregate Pseudorandomness). Malavolta technically requires every verification key to have a corresponding secret key (a stronger requirement than our latency in Definition 1) to compute the VRF image–proof pairs $(y_i, \pi_i)_i$ for the aggregation algorithm. This requirement is not necessary; instead we directly (but still inefficiently) compute these pairs. Note that the inefficient computation of those pairs (or the secret keys in Malavolta’s definition) makes the aggregate pseudorandomness game not efficiently falsifiable in the sense of [Nao03; GW11].

A subtle difference between our notion and Malavolta’s notion is in the distribution of the proofs π^i , in Malavolta’s notion the proofs follow the honest distribution of VRF proofs, while in our notion there might not be an honest distribution, because a (malformed) verification key might not have a corresponding secret key. We stress that—assuming perfect aggregate correctness and perfect unique provability of the VRF—this actually makes no difference. The reason is that under perfect aggregate correctness and uniqueness the aggregate image $\tilde{y}_1$ is uniquely determined by the aggregate verification key $\tilde{\text{vk}}$ and the preimage x^* . While the proof $\tilde{\pi}_1$ could be distributed arbitrarily, note that the proof is actually *not* given to the adversary.

Theorem 13 (Aggregability Preservation). If the underlying VRF in Constructions 1 and 2 (with the identity hash function as in Theorem 11) is aggregable, then the resulting VRF is also aggregable.

Proof. We present the proof for Construction 1, the result holds analogously for Construction 2. Importantly, we need that both constructions use the trivial (identity) hash function; otherwise the underlying VRF would not necessarily be evaluated on the same (inner) preimage v even though the outer preimage x is the same because the hash key would be different. Let VRF' be the underlying aggregable VRF. We define the aggregate algorithm VRF.Agg and VRF.AggVerify as follows:

³⁴ Because the definition in [Mal24] is underspecified, this is our best-effort interpretation of aggregate uniqueness.

$\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{agg-pr}}(\lambda)$
<pre> 1 : $L := \emptyset$ 2 : $(\text{vk}_0, \text{sk}_0) \leftarrow \text{VRF.Gen}(1^\lambda)$ 3 : $((\text{vk}_i)_i, x^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\text{VRF}}}(\text{vk}_0)$ 4 : $b \leftarrow \{0, 1\}$ 5 : $\tilde{y}_0 \leftarrow \{0, 1\}^{\ell_y(\lambda)}$ 6 : $(y^0, \pi^0) \leftarrow \text{VRF.Eval}(\text{sk}_0, x^*)$ 7 : bruteforce (y^i, π^i) s.t. $\text{VRF.Verify}(\text{vk}_i, x^*, y^i, \pi^i) = 1$ $\parallel (\perp, \perp)$ if none exists 8 : $(\tilde{\text{vk}}, \tilde{y}_1, \tilde{\pi}_1) := \text{VRF.Agg}(x^*, (\text{vk}_i, y^i, \pi^i)_{i=0, \dots})$ 9 : $b' \leftarrow \mathcal{A}^{\mathcal{O}_{\text{VRF}}}(\tilde{\text{vk}}, \tilde{y}_b)$ 10 : req $b = b' \wedge x^* \notin L$ 11 : return 1 </pre>
$\mathcal{O}_{\text{VRF}}(x)$
<pre> 1 : $L := L \cup \{x\}$ 2 : $(y, \pi) \leftarrow \text{VRF.Eval}(\text{sk}, x)$ 3 : return (y, π) </pre>
$\text{Exp}_{\text{VRF}, \mathcal{A}}^{\text{agg-bd}}(\lambda)$
<pre> 1 : $(x, \text{vk}_1, \dots, \text{vk}_n, y_1, \dots, y_n, \pi_1, \dots, \pi_n, y'_1, \dots, y'_n) \leftarrow \mathcal{A}(1^\lambda)$ 2 : req $(y_1, \dots, y_n) \neq (y'_1, \dots, y'_n)$ 3 : for $i \in [n]$ 4 : req $\text{VRF.Verify}(\text{vk}_i, x, y_i, \pi_i) = 1$ 5 : $(\tilde{\text{vk}}, \tilde{y}, \tilde{\pi}) := \text{VRF.Agg}(x, (\text{vk}_i, y_i, \pi_i)_i)$ 6 : $(\tilde{\text{vk}}', \tilde{y}', \tilde{\pi}') := \text{VRF.Agg}(x, (\text{vk}_i, y'_i, \perp)_i)$ 7 : req $\text{VRF.AggVerify}((\text{vk}_i)_i, x, \tilde{y}', \tilde{\pi}) = 1$ 8 : return 1 </pre>

Fig. 6: Aggregate pseudorandomness and binding games. Note that the aggregate pseudorandomness game is not efficiently falsifiable in the sense of [Nao03; GW11].

$\text{VRF.Agg}(x, (\text{vk}_i, y_i, \pi_i)_{i \in [n]})$	$\text{VRF.AggVerify}((\text{vk}_i)_{i \in [n]}, x, \tilde{y}, \tilde{\pi})$
1 : for $i \in [n]$	1 : $\tilde{z} := \tilde{y} \oplus \text{HCF}(x)$
2 : $z_i := y_i \oplus \text{HCF}(x)$	2 : $v := \text{injOWF}(x)$
3 : $v := \text{injOWF}(x)$	3 : req $\text{VRF'.AggVerify}((\text{vk}_i)_i, v, \tilde{z}, \tilde{\pi}) = 1$
4 : $(\tilde{\text{vk}}, \tilde{z}, \tilde{\pi}) := \text{VRF'.Agg}(\text{vk}_i, v, z_i, \pi_i)$	4 : return 1
5 : $\tilde{y} := \tilde{z} \oplus \text{HCF}(x)$	
6 : return $(\tilde{\text{vk}}, \tilde{y}, \tilde{\pi})$	

Correctness. Aggregate correctness of the resulting VRF follows trivially from the aggregate correctness of the underlying VRF.

Pseudorandomness. Aggregate pseudorandomness follows from the aggregate pseudorandomness of the underlying VRF via the following reduction $\mathcal{R}$: The reduction receives a verification key vk'_0 from the challenger of the underlying VRF's aggregate pseudorandomness game. The reduction forwards $\text{vk}_0 := \text{vk}'_0$ to the adversary of the aggregate pseudorandomness game of the resulting VRF. Whenever the adversary asks for an evaluation on some preimage x_i the reduction computes $v_i := \text{injOWF}(x_i)$ and queries it to its own challenger who responds with an image-proof pair (z_i, π_i) . The reduction then forwards $(y_i := z_i \oplus \text{HCF}(x_i), \pi_i)$ to the adversary. Once the adversary submits verification keys $(\text{vk}_i)_i$ and a challenge preimage x^* , the reduction forwards $(\text{vk}_i := \text{vk}'_i)_i$ and $v^* := \text{injOWF}(x^*)$ to its own challenger. The challenger responds with an aggregate verification key $\tilde{\text{vk}}'$, an aggregate image $\tilde{z}_b$. The reduction forwards $(\tilde{\text{vk}} := \tilde{\text{vk}}', \tilde{y}_b := \tilde{z}_b \oplus \text{HCF}(x^*))$ to the adversary who responds with a guess b' . The reduction simply forwards b' to its own challenger. Note that if the challenger chooses $b = 0$, then the (internal) challenge image $\tilde{z}_b$ is uniformly random and thus the external challenge image $\tilde{y}_b$ is also uniformly random. On the other hand, if $b = 1$, then $\tilde{y}_b := \tilde{z}_b \oplus \text{HCF}(x^*)$ is the correctly evaluated aggregate image.

Binding. Aggregate binding follows via a straightforward reduction (similar to pseudorandomness). Though, we reiterate that aggregate binding only makes sense against uniform adversaries or in the CRS model. Note that the unbiasedness proof of Theorem 11 requires security of the underlying primitives and achieves security against non-uniform adversaries.

Uniqueness. Perfect aggregate uniqueness follows directly from the perfect unique provability of the underlying VRF. $\square$

Efficient Computability. As described in Section 1.2 the VRP construction of Dodis and Puniya [DP07] does achieve (unconditionally) unbiasedness o.i.p. but there is an efficient attack against its full unbiasedness (as we show in Section 10). Moreover, their VRP construction requires superlogarithmically many Feistel rounds, so even if each round is computable in NC^1 (constant fan-in circuits of logarithmic depth), then the overall construction cannot be computable in NC^2 (constant fan-in circuits of depth $\log(\lambda)^2$).

In contrast, our construction does not require multiple rounds of evaluations; in fact it only requires

- a single evaluation of the underlying VRF,
- a single evaluation of the injective OWF,
- a single evaluation of the CRH (if the underlying VRF is not latent),
- a single evaluation of a PRF (for Construction 2).

The computation of the HCF and the final XOR are essentially for free because they can be computed very efficiently in NC^1 .³⁵ Now, if all three components are computable in NC^1 , then the overall construction is computable in NC^1 as well. We remark that computability in NC^1 is optimal because PRFs in AC^0 don't exist. [LMN89]

Entropic Unbiasability. While unbiasability—as defined by Giunta and Stewart [GS24] (Definition 7)—is a strong notion, its unbiasability guarantee is predicated on the assumption that the input is *totally uniformly* distributed. Depending on the application this might not be the case, in particular if the VRF input is (at least partially) adversarially controlled. A more robust assumption is that the input distribution has some min-entropy. Given that we cannot hope for an indistinguishability-type notion if the VRF verification key as well as the input distribution is (at least partially) adversarially controlled, the best we can do is argue about the min-entropy of the VRF output.

Definition 13 (Entropy Loss). *A VRF VRF has entropy loss at most δ if for each verification key vk the VRF output's min-entropy is at most δ smaller than the min-entropy of the VRF input. Formally, for each security parameter $\lambda \in \mathbb{N}$, each verification key vk and each distribution $\mathcal{D}_\lambda$ with min-entropy $H_\infty[\mathcal{D}_\lambda] \geq h_\infty(\lambda)$ it holds that*

$$H_\infty[x \leftarrow \mathcal{D}_\lambda : y] \geq h_\infty(\lambda) - \delta(\lambda) \quad (79)$$

where y is the unique image such that $\exists \pi : \text{VRF.Verify}(\text{vk}, x, y, \pi) = 1$.³⁶

Application. This unbiasability notion proves useful in scenarios where a VRF is evaluated on a common (public) user-generated input. For example consider a typical leader election protocol where each user i publishes a share $x_i \in \{0, 1\}^\lambda$ and then each user's VRF is evaluated on the input $x := \bigoplus_i x_i$. An adversary that observes all honest users' shares and then publishes its share can trivially control the value x . In contrast, if instead the VRF is evaluated³⁷ on $\tilde{x} := x_1 || x_2 || \dots$, then (even for a single honest user) the value x has at least min-entropy λ .

Instantiation. By instantiating Construction 2 with the VRP of Dodis and Puniya [DP07] we obtain a VRF that is δ -entropically unbiasable. If the PRF in Construction 2 is polynomially hard, then we achieve $\delta(\lambda) = \lambda^\nu$ for any $\nu \in \mathbb{R}_{\geq 0}$, if the PRF is subexponentially hard, we achieve $\delta(\lambda) = \log(\lambda)^\nu$ for any $\nu \in \mathbb{R}_{\geq 1}$.

Theorem 14 (Entropically Unbiasable VRF). Let PRF be a PRF with seed length δ . Let VRP' be a VRP. Then VRF from Construction 2 (instantiated with VRP VRP' and PRF PRF) is δ -entropically unbiasable.

Proof Sketch. Let $\mathcal{D}_\lambda$ be a distribution over $\{0, 1\}^\lambda$ with min-entropy $H_\infty[\mathcal{D}_\lambda] \geq h_\infty(\lambda)$. For each verification key vk the variable z in the upper branch has full entropy $H_\infty[\mathcal{D}_\lambda]$ because injOWF is injective and VRP' is injective for each vk . The variable h' in the lower branch has min-entropy at most $\delta(\lambda)$ because the seed length of the PRF is at most $\delta(\lambda)$. Thus, the lower branch can annihilate at most $\delta(\lambda)$ bits of entropy from the upper branch. $\square$

Remark 13. If we instantiate the injective OWF with a (keyless) one-way permutation, then we even obtain a VRF whose *relative* min-entropy is almost preserved. Namely, both the VRF input and VRF output have length λ .

³⁵ This is optimal because the parity function is not computable in $\text{AC}^0 \subsetneq \text{NC}^0$.

³⁶ $y := \perp$ if no such image exists.

³⁷ Obviously, this requires to evaluate the VRF on much larger inputs.

Full unbiasedness in the URS model. Recall that we cannot hope for the VRF output to have more entropy than the VRF input. Moreover, without any setup we cannot hope for the VRF output distribution to be close to uniform because such a VRF would constitute a seedless extractor which does not exist for arbitrary sources [CG85]. However, if we assume a uniform random string (URS) $\text{urs} \leftarrow \{0, 1\}^\lambda$ is available, we can—perhaps surprisingly—achieve full unbiasedness, even if the preimages only have a small amount of min-entropy. Namely, we can apply a seeded extractor (e.g. via the leftover hash lemma [ILL89]) to the VRF output of Construction 2 (again instantiated with a VRP), e.g. $y' := \text{Ext}(\text{urs}, y)$ where $(y, \pi) \leftarrow \text{VRF.Eval}(\text{sk}, x)$. For example, if the preimage x only has entropy $\lambda^{2\epsilon}$ for some $\epsilon \in \mathbb{R}_{\geq 0}$, then the output distribution y' is computationally close to the uniform distribution on $\{0, 1\}^{\lambda^\epsilon}$ for any polynomially bounded adversary.

Note that simply extracting (close-to-uniform) randomness directly from the VRF preimage does not work, because it might induce collisions on the inner VRF inputs that break pseudorandomness.

In general, the VRF input only needs to have slightly more min-entropy than the logarithm of the adversary’s runtime for this approach to work. Lastly, we note that the assumption of a URS is much more realistic than the assumption of a structured random string (SRS) which requires a trusted setup or multi-party protocol to generate.

9.1 Practical Considerations

In this section we discuss how unbiasedness is typically achieved in practical applications. A simple way to achieve unbiasedness is the folklore VUF-then-Hash construction. This construction first computes a VUF on the preimage and then hashes the resulting VUF image together with the VUF’s verification key and the preimage. Giunta and Stewart [GS24] show that this VUF-then-Hash construction (where the verification key and preimage are included in the hash input) is indeed unbiased if the hash function is modeled as a truly random function. However, this construction has two downsides.

- First, its unbiasedness security is unconditional in the random oracle model whereas we prove in Section 8 that unconditional (full) unbiasedness is impossible. This suggests that a proof in the random oracle model might not indicate unbiasedness when any concrete (efficient) hash function is used.
- Second, we show how this construction can, in the worst case, even make the resulting VRF biasable, even if the underlying VUF was not.

We demonstrate that compared to the very efficient VUF-then-Hash construction the reverse “Hash-then-VRF” construction can be even more efficient, and it provides *provable* unbiasedness in the standard model.

On the Folklore VUF-then-Hash Construction. To underscore the relevance of our provably unbiased Constructions 1 and 2 we discuss several downsides of the folklore VUF-then-Hash construction. Giunta and Stewart [GS24] show that this VUF-then-Hash construction (where the verification key and preimage are included in the hash input) is indeed unbiased if the hash function is modeled as a truly random function. In this idealized random oracle model, the full GS-unbiasedness (Definition 7) holds unconditionally. This indicates a mismatch between the random oracle model and the standard model (where the hash function is fixed) because—as we discuss in Section 8—full unbiasedness³⁸ cannot hold unconditionally in the standard model (due to the existence of an unbounded adversary). Hence, in the standard model³⁹ the full unbiasedness of the construction *must* be based on some computational assumption, e.g. the collision-resistance of the hash function. While we are not able to formally prove

³⁸ Actually, full 2s-unbiasedness as defined in Definition 11.

³⁹ In the real world the hash function would typically be instantiated by a fixed (even keyless) hash function like SHA-256.

$\text{VUF}'.\text{Gen}(1^\lambda)$	$\text{VUF}'.\text{Eval}(\text{sk}', x)$	$\text{VUF}'.\text{Verify}(\text{vk}', x, y i, \pi)$
$r'' \leftarrow 1^\lambda$	$r'' := \text{sk}'$	$\text{vk}'' := \text{vk}'$
$(\text{vk}'', \text{sk}'') := \text{VUF}''.\text{Gen}(1^\lambda; r'')$	$(\text{vk}'', \text{sk}'') := \text{VUF}''.\text{Gen}(1^\lambda; r'')$	req $\text{VUF}''.\text{Verify}(\text{vk}'', x, y, \pi) = 1$
$\text{vk}' := \text{vk}''$	$(y, \pi) := \text{VUF}''.\text{Eval}(\text{sk}'', x)$	for $\iota \in [0, i-1]$
$\text{sk}' := r''$	for $i \in [\lambda]$	$z^\iota := \text{Hash.Eval}(\text{hk}, \text{vk}'' x y \iota)$
return (vk', sk')	$z^i := \text{Hash.Eval}(\text{hk}, \text{vk}'' x y i)$	req $z_0^\iota = 0$
	if $z_0^i = 0$	$z^i := \text{Hash}(\text{hk}, \text{vk}'' x y i)$
	return $(y i, \pi)$	req $z_0^i = 1$
	return $(\perp, \perp)$	return 1

Fig. 7: Rejection-sampling construction of a VUF that is unbiased but becomes biasable in the VUF-then-Hash construction. Let VUF' be an underlying VUF. We assume that the hash key hk is either empty or might even be generated by a trusted third party.

the *biasability* of the VUF-then-Hash construction, we give some intuition why it seems hard to prove biasability. More importantly, we argue that a malicious VUF designer can strongly bias the VUF-then-Hash construction.

Why proving biasability of VUF-then-Hash is hard. It is possible that the VUF-then-Hash construction emulates Construction 2 as in Theorem 11. Concretely, it could be that

- the VUF VUF in the VUF-then-Hash construction preprocesses its input with an injective (keyless) one-way function injOWF s.t. $\text{VUF.Eval}(\text{sk}, x) = \text{VUF}'.\text{Eval}(\text{sk}, \text{injOWF}(x))$,
- the underlying VUF VUF' is latent (Definition 1),
- the hash function evaluates $\text{Hash}(\text{vk}, x, y) = \text{PRF.Eval}(\text{HCF}(x), \text{vk}) \oplus y$ where HCF is a hardcore function of injOWF .

Since Theorem 11 shows that this construction is fully unbiased, it is not possible—without further restrictions—to prove the biasability of the VUF-then-Hash construction in the standard model.

Remark 14. It is extremely unlikely that real hash functions (like SHA-256) behave like the hash function Hash above. Thus, the above should be considered as a formal argument for the *hardness of proving biasability* and not as direct evidence for or against the biasability of the VUF-then-Hash construction.

Possibility of (covert) bias in the VUF-then-Hash construction. Given the above barrier for proving biasability, we instead focus on a different scenario where a given hash function Hash (e.g. SHA-256) is fixed. Given any fixed hash function we can design a “rejection-sampling” VUF in Figure 7 that—when plugged into the VUF-then-Hash construction—makes the overall construction biased *even o.i.p.* We emphasize that this construction is contrived, and likely not a practical concern, but it highlights the fundamental issue of relying on heuristic security arguments for unbiasedability in the standard model.

The idea is to evaluate an underlying VUF VUF' on the actual preimage x to obtain a tentative image y . That image y is then appended with the first counter i s.t. the actual hash image $\text{Hash}(\text{vk}, x, y||i)$ has the least significant bit set to 0. Notice that when the hash function is sufficiently balanced (e.g. a random function), then the counter will be at most constant (in expectation). Since the hash construction iterates over the counter $i \in [\lambda]$, the advantage of the o.i.p. unbiasedability adversary is overwhelming by a Chernoff bound.

The VUF VUF inherits its unpredictability from the underlying VUF VUF' . While the unbiasedness of the VUF VUF' is not clear, recall that the claim of the VUF-then-Hash construction is to transform *any* even biasable VUF into an unbiased one. No matter whether VUF' is unbiased, the resulting VUF-then-Hash VUF VUF is biased because all valid images have the least significant bit set to 0. Thus, in the worst-case the VUF-then-Hash construction actually induces bias into the overall construction even if the underlying VUF VUF' is unbiased.

Figure 7 exemplifies the possibility of injecting (possibly undetected) bias into the VUF-then-Hash construction by choosing the underlying VUF VUF' in a malicious way. Put plainly, it is possible to backdoor a VUF such that the resulting VUF-then-Hash construction is biased.

Note that for Figure 7 it is quite obvious that such a VUF is biased towards 0 in the least significant bit and should thus not be deployed (despite it being provably unpredictable). However, a malicious entity, e.g. a government agency, could produce a more sophisticated VUF construction (and influence its standardization) that has some covert bias that is only known to the entity. Yet, the resulting VUF can be proven unpredictable.

We stress that this strategy of inducing bias (via rejection sampling) applies universally to *any sufficiently balanced*, a priori fixed hash function (and potentially even multiple ones simultaneously) and moreover it can bias *any* predicate of the VRF image (not only the least significant bit).

In our opinion this underscores the importance of Constructions 1 and 2 in applications that require *provable* unbiasedness in the standard model—especially given their practical competitiveness with the VUF-then-Hash construction due to their small computational overhead.

Lastly, we emphasize that our constructions Constructions 1 and 2 preserve several useful properties (see Section 9) that are likely not retained by the VUF-then-Hash construction.

On Our Hash-then-VRF Construction. Next, we describe how our Hash-then-VRF construction alleviates such issues. In this subsection we assume that the reader has our basic construction, i.e., Construction 1, in mind. The reader is welcome to skip the rest of this section and return to it after reading Sections 6 and 7 for in-depth explanation of the underlying construction. In our formal presentation Construction 1 requires a (single) injective OWF, a keyed CRH and an underlying VRF.⁴⁰ Recall that the purpose of the injective OWF is to extract entropy via the hardcore function HCF from the preimage while avoiding collisions (on the u value, cf. Figure 3) in the pseudorandomness game against non-uniform adversaries. The injectivity of the OWF ensures collision-resistance on u unconditionally; thus even against non-uniform adversaries. If we instead only care about pseudorandomness against *uniform* adversaries,⁴¹ we can omit the injective OWF and use a *keyless* CRH (e.g. a common hash function like SHA-256). The hardcore function HCF can then be a HCF⁴² of the CRH and not the injective OWF. We give a graphical representation of this instantiation in Figure 1. This construction (with a latent VRF, cf. Definition 1) remains unbiased via the non-black-box analysis of Theorem 11.⁴³ Concretely, it is unbiased against non-uniform adversaries assuming non-uniform one-wayness of the CRH. Moreover, it is pseudorandom against uniform adversaries.⁴⁴ For practical parameters it is even more efficient than the VUF-then-Hash construction, because

⁴⁰ Our construction can also be based on a VUF, if the resulting function should only be unpredictable instead of pseudorandom.

⁴¹ This is often an acceptable security notion for practical applications.

⁴² E.g. the canonical Goldreich–Levin hardcore bit.

⁴³ Its unbiasedness cannot be shown via the black-box analysis of Theorem 8 because the reduction loses a factor of $2^{\text{poly}(|u|)}$, i.e., the resulting inverter for CRH runs in time $2^{\text{poly}(|u|)}$. However, the CRH cannot be one-way against such inverters because a preimage can be guessed in time $2^{\text{poly}(|u|)}$ with high probability.

⁴⁴ A non-uniform adversary can always hardcode a collision of the keyless CRH, so that we cannot invoke the pseudorandomness of the underlying VRF.

- the expensive (underlying) VRF is evaluated on a hash image (potentially much smaller than the outer VRF’s preimage) where the hash image length often exactly matches the preimage length of VRF implementations,
- the CRH is evaluated only on the (outer) VRF’s preimage (not including the VRF verification key and a VRF image as in the VUF-then-Hash construction),
- the HCF can be a simple operation (like a parity in the Goldreich–Levin HCB) which has negligible cost compared to the VRF evaluation.

Because this instantiation of our Hash-then-VRF construction improves the concrete efficiency as well as the unbiasedness guarantee we recommend it for practical applications whenever the underlying VRF is latent (cf. Definition 1). We stress that all known VRF constructions are latent.

Corollary 2 (Hash-then-VRF Security). Assuming a (uniformly) collision-resistant, (non-uniformly) one-way, keyless CRH and a latent VRF, the construction of Figure 1 is pseudorandom against uniform adversaries and unbiased o.i.p. against non-uniform adversaries (both with polynomial security loss).

Remark 15. Even the construction with full unbiasedness where an additional PRF evaluation is required in the lower branch (analog to Construction 2), our Hash-then-VRF construction (with an additional PRF evaluation of the verification key) likely remains more efficient than the VUF-then-Hash construction if the outer and inner VRF’s preimage lengths differ. This is because the evaluation of a PRF (e.g. AES or HMAC) is typically cheaper than a VRF evaluation by orders of magnitude.

10 Separating Full Unbiasability from Unbiasability o.I.P. via Invertibility

We show that full unbiasedness is strictly stronger than unbiasedness o.i.p. (as defined in Definition 7) in the strong sense that a VRF can be *even unconditionally perfectly* unbiased o.i.p., yet there exists an efficient attack against its full unbiasedness. Concretely, in Section 1.2 we discussed that the VRP of Dodis and Puniya [DP07] achieves unbiasedness o.i.p. but fails to achieve full unbiasedness. The unbiasedness o.i.p. of *any* VRP holds—in fact—unconditionally and perfectly because for any verification key the VRP is guaranteed to be a permutation; so each (independent) preimage in the unbiased o.i.p. game will be mapped to a unique (and also independent) image. We argued that unbiasedness *may* not hold for the VRP of [DP07] if the underlying VRF is degenerate in the following way: it could be that the underlying VRF has two different verification keys that both describe the same function (e.g. the zero function). Then for those verification keys the VRP will be the identity function, i.e., the images for the two verification keys will always be the same and not independent.

Now, we show a more surprising result. We show that full unbiasedness for invertible VRFs (including [DP07]) *cannot* hold—irrespective of whether the underlying VRF is degenerate or not. Recall that the VRP of [DP07] is based on a Feistel construction which makes it invertible using the secret key.

Definition 14 (Invertible VRF). We say a VRF VRF is invertible if there exists an algorithm VRF.Inv such that for each key pair $(\text{vk}, \text{sk}) \leftarrow \text{VRF.Gen}(1^\lambda)$ and each preimage $x \in \{0, 1\}^\lambda$ it holds that $\text{VRF.Inv}(\text{sk}, y) = x$ where $(y, \pi) := \text{VRF.Eval}(\text{sk}, x)$.

Remark 16. Note that our notion of (perfect) invertibility requires the VRF to be injective.

Theorem 15 (Invertibility vs. Full Unbiasability). Let VRF be an invertible VRF with default domain $\{0, 1\}^\lambda$. Then there exists an adversary $\mathcal{A}$ parameterized by some $c \in \mathbb{N}$ that breaks the full $(\mathbf{t}, \epsilon, \mathbf{a}, \mathbf{b})$ -unbiasedness of VRF for $\mathbf{a}(\lambda) = 2$, $\mathbf{b}(\lambda) = 1$, $\mathbf{t}(\lambda) \leq \lambda^c$ and $\epsilon(\lambda) := 1 - 2^{-\lambda}$.

Proof. We describe an adversary $\mathcal{A}$ that breaks the full unbiasedness of VRF . The adversary $\mathcal{A}$ samples two key pairs $(\text{vk}_0, \text{sk}_0), (\text{vk}_1, \text{sk}_1) \leftarrow \text{VRF.Gen}(1^\lambda)$. The adversary’s predicate ρ on input y_0, y_1 (both

evaluated on the same preimage $x \leftarrow \{0, 1\}^\lambda$ first computes $x' := \text{VRF.Inv}(\text{sk}_0, y_0)$ and $(y'_1, \pi'_1) \leftarrow \text{VRF.Eval}(\text{sk}_1, x')$. The predicate then outputs 1 (real) iff $y_1 = y'_1$.

Assume that the full unbiasedness challenger evaluates the predicate on the real images y_0, y_1 . Then by invertibility the predicate will recover the challenger's preimage $x' = x$. Then by correctness the image y'_1 will match the predicate's second argument y_1 . In this case the predicate will always decide correctly.

In contrast, assume that the challenger evaluates the predicate on uniformly random images y_0, y_1 . Let $x' \in \{0, 1\}^\lambda \cup \{\perp\}$ be the value that the predicate internally inverted from the first image y_0 . Because the second image y_1 is sampled uniformly and independently, the probability that $y_1 = y'_1$ is $2^{-\ell_y(\lambda)} \leq 2^{-\lambda}$. The last inequality follows from the injectivity of the VRF VRF . Hence, in this case the predicate will decide incorrectly with probability at most $2^{-\lambda}$.

Letting $c \in \mathbb{N}$ be a constant s.t. $\lambda^{c/2}$ exceeds the runtime of VRF.Gen and VRF.Eval and VRF.Inv we find that the runtime of the adversary is at most λ^c . $\square$

11 Symmetric VRFs from Injective OWFs and VRFs

Dual PRFs are (non-verifiable) pseudorandom functions that remain pseudorandom even when the key and the input are swapped. Concrete constructions are extensively used in real-world applications (see for example [BBG⁺23]). Bellare and Lysyanskaya [BL15; BL24] construct dual PRFs generically from (keyless) OWPs, or—against uniform adversaries—from (keyless) collision-resistant hash functions and one-way functions.

A stronger version of dual PRFs are *symmetric PRFs*, i.e., $\text{PRF.Eval}(\text{sk}, x) = \text{PRF.Eval}(x, \text{sk})$ for all sk, x .

Construction 4 (Symmetric Unbiasable VRF Construction). Let $\text{injOWF} : \{0, 1\}^\lambda \rightarrow \{0, 1\}^{\ell_{\text{injOWF}}(\lambda)}$ be an injective OWF with poly. many HCBs HCF . Let VRF' be a latent VRF. In Figure 8 we define our symmetric unbiased VRF VRF by casting the symmetric PRF construction of [BL15; BL24] as a suitable modification of Construction 2.

We observe that

- even injective OWFs (as opposed to OWPs) plus CRH suffice for symmetric and thus dual PRFs,
- the construction in Figure 8 yields symmetric VRFs from subexp. secure injective OWFs, CRH and VRFs,
- instantiating Figure 8 with the trivial CRH yields symmetric VRFs from injective OWFs and any latent VRF.

Because Construction 4 is a more complex modification of the Construction 2, we refrain from a detailed quantitative analysis as in Section 6 in favor of an easier presentation.

Theorem 16 (Informal). Let injOWF be an injective OWF. Let CRH be a collision-resistant hash function. Let VRF' be a pseudorandom VRF. Then VRF from Figure 8 is symmetric, pseudorandom and unbiased. The same holds if CRH is the identity function and VRF' is latent.

Proof Sketch. Symmetry follows by inspection. Pseudorandomness follows (analogously to Construction 2) via two hybrid steps:

- In the first step we replace the hardcore bits h_{key} with independent uniformly random bits. This is justified by the pseudorandomness of the hardcore function HCF because the uniform randomness r is only used to compute the hardcore bits h_{key} and the OWF image u_{key} .
- In the second step we replace the (internal) VRF challenge image z_{x^*} with a uniformly random string. This is justified by the pseudorandomness of the VRF VRF' (because the used VRF secret key sk_{key} and hash key h_{key} are now honestly generated) and the injectivity of the OWF injOWF plus the collision resistance of CRH , so that the internal VRF is not queried on the same input as the challenge u_{x^*} .

Unbiasability follows almost symmetrically:

- First, we brute-force the (internal) VRF evaluations z_x (or use the latent VRF evaluation key) to not query the unbiasedness adversary on the preimages x .
- Second, we replace the hardcore bits $h_x^{\text{VRF}'} \parallel h_x^{\text{CRH}}$ for each preimage x with independent uniformly random bits. This is justified by the pseudorandomness of the hardcore function HCF because the uniform preimages x are only used to compute the hardcore bits $h_x^{\text{VRF}'} \parallel h_x^{\text{CRH}}$ and the OWF image u_x . Recall that the adversary does not get to see the preimages x anymore, as we brute-force the internal VRF evaluation (or use the latent VRF evaluation key).
- Lastly, we replace the (internal) VRF challenge image z_{key} with a uniformly random string. This is justified by the pseudorandomness of the VRF VRF' (because the used VRF secret key sk_x and hash key hk_x are now honestly generated) and the injectivity of the OWF injOWF plus the collision resistance of CRH, so that the internal VRF is not queried on the same input as the challenge. Recall that all verification keys are different, thus the corresponding OWF images u_{key} are as well, hence the internal VRF is not queried on the same input as the challenge.

Finally, unique provability follows from the unique provability of the underlying VRF VRF' . $\square$

12 Simplified Corollaries

We simplify the general statement of Theorems 5 and 6 for polynomial (resp. superpolynomial) security.

Corollary 3. Let injOWF be an injective, subexponentially secure OWF. Let VRF' be a perfectly uniquely provable, superpolynomially secure VRF. Let CRH be a superpolynomially secure CRH with output length $\ell_v(\lambda) = \lambda^{1/\log_2 \log_2(\lambda)}$. Then VRF from Construction 1 is subexponentially unbiased o.i.p. for arbitrary $\mathbf{a}, \mathbf{b} \in \text{poly}(\lambda)$, and superpolynomially pseudorandom.

Proof. Fix $\mathbf{a}, \mathbf{b} \in \text{poly}(\lambda)$. Fix a VRF VRF' , a CRH CRH and a OWF injOWF . Let $c_{\max}$ be such that $\lambda^{c_{\max}}$ grows faster than any polynomial runtime associated with the algorithms of injOWF , CRH, and VRF' , and $\mathbf{a}, \mathbf{b}$.

Pseudorandomness. Let $\epsilon_{\text{VRF}}^{\text{pr}}(\lambda) := 1/\mathbf{t}_{\text{VRF}}^{\text{pr}}(\lambda)$ for simplicity. For any $\mathbf{t}_{\text{VRF}}^{\text{pr}}(\lambda) = 2^{\log_2(\lambda)^{c_{\text{VRF}}^{\text{pr}}}}$ (with $c_{\text{VRF}}^{\text{pr}} \geq 1$) we can find $\mathbf{t}_{\text{CRH}}^{\text{cr}}(\lambda) = 1/\epsilon_{\text{CRH}}^{\text{cr}}(\lambda) = 2^{\log_2(\lambda)^{c_{\text{CRH}}^{\text{cr}}}}$ and $\mathbf{t}_{\text{VRF}'}^{\text{pr}}(\lambda) = 1/\epsilon_{\text{VRF}'}^{\text{pr}}(\lambda) = 2^{\log_2(\lambda)^{c_{\text{VRF}'}^{\text{pr}}}}$ such that Equations (29) and (30) hold. In particular this holds for $c_{\text{VRF}'}^{\text{pr}} := c_{\text{CRH}}^{\text{pr}} := 2c_{\text{VRF}}^{\text{pr}}$. For Equation (29) observe that asymptotically

$$4^{1/c_{\text{VRF}}^{\text{pr}}} \log_2(\lambda) \leq (\log_2(\lambda)/\log_2 \log_2(\lambda))^2 \quad (80)$$

$$\iff 4 \log_2(\lambda)^{c_{\text{VRF}}^{\text{pr}}} \leq (\log_2(\lambda)/\log_2 \log_2(\lambda))^{2c_{\text{VRF}}^{\text{pr}}} \quad (81)$$

$$\implies 3 \log_2(\lambda)^{c_{\text{VRF}}^{\text{pr}}} \leq (\log_2(\lambda)/\log_2 \log_2(\lambda))^{2c_{\text{VRF}}^{\text{pr}}} - 1 \quad (82)$$

$$\implies 2^{-3 \log_2(\lambda)^{c_{\text{VRF}}^{\text{pr}}}} \geq 2^{-\log_2(\lambda)^{2c_{\text{VRF}}^{\text{pr}}}} + 2^{-(\log_2(\lambda)/\log_2 \log_2(\lambda))^{2c_{\text{VRF}}^{\text{pr}}}} \quad (83)$$

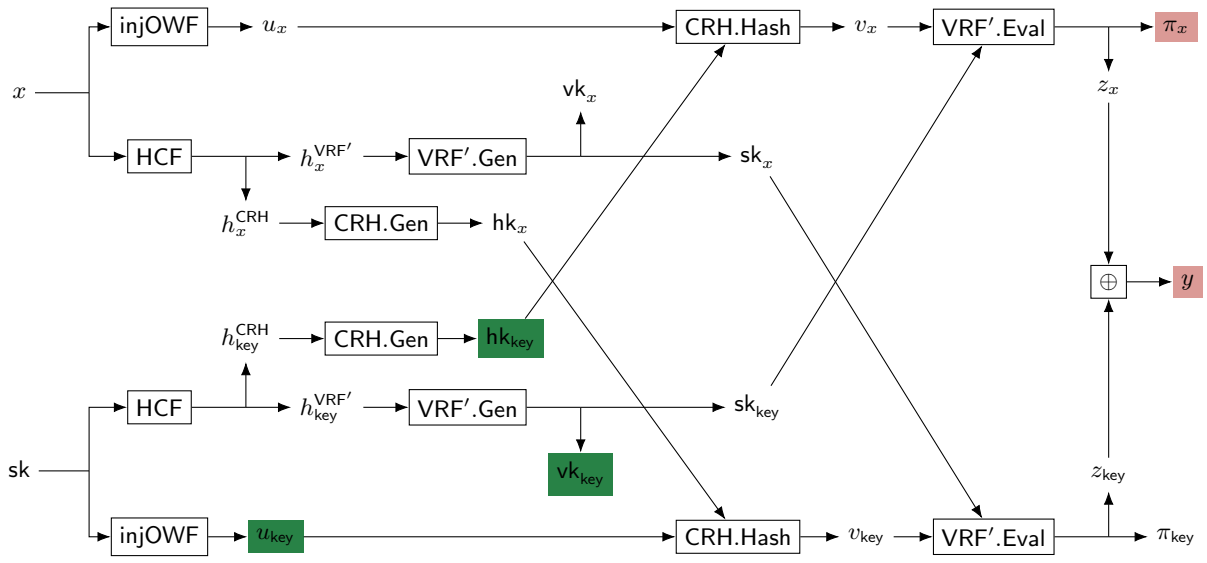
$$\iff \epsilon_{\text{VRF}}^{\text{pr}}(\lambda)^3 \geq \epsilon_{\text{CRH}}^{\text{cr}}(\lambda) + \epsilon_{\text{VRF}'}^{\text{pr}}(\ell_v(\lambda)) \quad (84)$$

$$\implies \epsilon_{\text{VRF}}^{\text{pr}}(\lambda) \geq \mathbf{t}_{\text{VRF}}^{\text{pr}}(\lambda)^2 \epsilon_{\text{CRH}}^{\text{cr}}(\lambda) + \epsilon_{\text{VRF}'}^{\text{pr}}(\ell_v(\lambda)) . \quad (29)$$

For Equation (30) observe that asymptotically

$$2^{1/c_{\text{VRF}}^{\text{pr}}} \log_2(\lambda) \leq (\log_2(\lambda)/\log_2 \log_2(\lambda))^2 \quad (85)$$

$$\implies \log_2(\lambda)^{c_{\text{VRF}}^{\text{pr}}} \leq (\log_2(\lambda)/\log_2 \log_2(\lambda))^{2c_{\text{VRF}}^{\text{pr}}} - 2 \log_2(\lambda^{c_{\max}}) \quad (86)$$



(a) Detailed graphical overview of our symmetric unbiaseable VRF VRF using an injective OWF injOWF with hardcore function HCF and a latent VRF VRF'. Structurally, the construction resembles the symmetric PRF construction of [BL15; BL24]. The verification key contains vk_{key} , u_{key} and $h_{\text{key}}^{\text{CRH}}$, and the VRF output and proof are y and π_x .

VRF.Gen(1^λ)	VRF.Eval($\text{sk}, x \in \{0, 1\}^\lambda$)
$r \leftarrow \{0, 1\}^\lambda$ $u_{\text{key}} := \text{injOWF}(r)$ $h_{\text{key}}^{\text{VRF}'} h_{\text{key}}^{\text{CRH}} := \text{HCF}(r) \in \{0, 1\}^{\ell_{\text{HCF}}(\lambda)}$ $(\text{vk}_{\text{key}}, \text{sk}_{\text{key}}) := \text{VRF}'.\text{Gen}(1^{\ell_{\text{HCF}}(\lambda)/2}; h_{\text{key}}^{\text{VRF}'})$ $h_{\text{key}} := \text{CRH.Setup}(1^{\ell_{\text{HCF}}(\lambda)/2}; h_{\text{key}}^{\text{CRH}})$ $\text{vk} := (\text{vk}_{\text{key}}, u_{\text{key}}, h_{\text{key}})$ $\text{sk} := r$ return (vk, sk)	$r := \text{sk}$ $u_{\text{key}} := \text{injOWF}(r)$ $h_{\text{key}}^{\text{VRF}'} h_{\text{key}}^{\text{CRH}} := \text{HCF}(r) \in \{0, 1\}^{\ell_{\text{HCF}}(\lambda)}$ $(\text{vk}_{\text{key}}, \text{sk}_{\text{key}}) := \text{VRF}'.\text{Gen}(1^{\ell_{\text{HCF}}(\lambda)/2}; h_{\text{key}}^{\text{VRF}'})$ $h_{\text{key}} := \text{CRH.Setup}(1^{\ell_{\text{HCF}}(\lambda)/2}; h_{\text{key}}^{\text{CRH}})$ $u_x := \text{injOWF}(x)$ $h_x^{\text{VRF}'} h_x^{\text{CRH}} := \text{HCF}(x) \in \{0, 1\}^{\ell_{\text{HCF}}(\lambda)}$ $(\text{vk}_x, \text{sk}_x) := \text{VRF}'.\text{Gen}(1^{\ell_{\text{HCF}}(\lambda)/2}; h_x^{\text{VRF}'})$ $h_{\text{key}} := \text{CRH.Gen}(1^{\ell_{\text{HCF}}(\lambda)/2}; h_x^{\text{CRH}})$ $v_{\text{key}} := \text{CRH.Hash}(h_{\text{key}}, u_{\text{key}})$ $v_x := \text{CRH.Hash}(h_{\text{key}}, u_x)$ $(z_{\text{key}}, \pi_{\text{key}}) := \text{VRF}'.\text{Eval}(\text{sk}_x, v_{\text{key}})$ $(z_x, \pi_x) := \text{VRF}'.\text{Eval}(\text{sk}_{\text{key}}, v_x)$ $y := z_x \oplus z_{\text{key}}$ $\pi := \pi_x$ return (y, π)
VRF.Verify (vk, x, y, π) $(\text{vk}_{\text{key}}, u_{\text{key}}, h_{\text{key}}) := \text{vk}$ $u_x := \text{injOWF}(x)$ $h_x := \text{HCF}(x) \in \{0, 1\}^{\ell_{\text{HCF}}(\lambda)}$ $(\text{vk}_x, \text{sk}_x) := \text{VRF}'.\text{Gen}(1^{\ell_{\text{HCF}}(\lambda)/2}; h_x)$ $(z_{\text{key}}, \pi_{\text{key}}) := \text{VRF}'.\text{Eval}(\text{sk}_x, u_{\text{key}})$ $z_x := y \oplus z_{\text{key}}$ $\pi_x := \pi$ req $\text{VRF}'.\text{Verify}(\text{vk}_{\text{key}}, u_x, z_x, \pi_x) = 1$ return 1	

(b) Formal description of our symmetric unbiaseable VRF VRF.

Fig. 8: Construction of our symmetric unbiaseable VRF VRF using an injective OWF injOWF with hardcore function HCF, CRH CRH and a VRF VRF'.

$$\iff 2^{\log_2(\lambda)^{c_{\text{VRF}}^{\text{pr}}}} \leq \frac{2^{(\log_2(\lambda)/\log_2 \log_2(\lambda))^{2c_{\text{VRF}}^{\text{pr}}}}}{4\lambda^{c_{\text{max}}}} \quad (87)$$

$$\implies 2^{\log_2(\lambda)^{c_{\text{VRF}}^{\text{pr}}}} \leq \frac{2^{(\log_2(\lambda)/\log_2 \log_2(\lambda))^{2c_{\text{VRF}}^{\text{pr}}}} - 2\lambda^{c_{\text{max}}}}{3\lambda^{c_{\text{max}}}} \quad (88)$$

$$\iff 2^{\log_2(\lambda)^{c_{\text{VRF}}^{\text{pr}}}} \leq \frac{\min\left(2^{\log_2(\lambda)^{2c_{\text{VRF}}^{\text{pr}}}}, 2^{(\log_2(\lambda)/\log_2 \log_2(\lambda))^{2c_{\text{VRF}}^{\text{pr}}}}\right) - 2\lambda^{c_{\text{max}}}}{3\lambda^{c_{\text{max}}}} \quad (89)$$

$$\iff t_{\text{VRF}}^{\text{pr}}(\lambda) \leq \frac{\min(t_{\text{CRH}}^{\text{cr}}(\lambda), t_{\text{VRF}'}^{\text{pr}}(\ell_v(\lambda))) - 2\lambda^{c_{\text{max}}}}{3\lambda^{c_{\text{max}}}} \quad (90)$$

$$\implies t_{\text{VRF}}^{\text{pr}}(\lambda) \leq \frac{\min(t_{\text{CRH}}^{\text{cr}}(\lambda) - \lambda^2, t_{\text{VRF}'}^{\text{pr}}(\ell_v(\lambda))) - t_{\text{CRH.Gen}}(\lambda)}{t_{\text{CRH.Hash}}(\lambda) + t_{\text{injOWF}}(\lambda) + t_{\text{HCF}}(\lambda)}. \quad (30)$$

Unbiasability. Let $\epsilon_{\text{VRF}}^{\text{ipub}}(\lambda) := 1/t_{\text{VRF}}^{\text{ipub}}(\lambda)$ for simplicity. For any $t_{\text{VRF}}^{\text{ipub}}(\lambda) = 2^{\lambda^{\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)}}$ (with $\mathfrak{o}_{\text{VRF}}^{\text{ipub}} \in o(1) \cap \omega(\log_2 \log_2(\lambda)/\log_2(\lambda))$) we can find $t_{\text{injOWF}}^{\text{inv}}(\lambda) = 1/\epsilon_{\text{injOWF}}^{\text{inv}}(\lambda) = 2^{\lambda^{\mathfrak{o}_{\text{injOWF}}^{\text{inv}}(\lambda)}}$ (with $\mathfrak{o}_{\text{injOWF}}^{\text{inv}} \in o(1)$) such that Equations (37) and (38) hold. In particular this holds for

$$\mathfrak{o}_{\text{injOWF}}^{\text{inv}}(\lambda) := \max(2\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda), \lambda^{1/\log_2 \log_2 \log_2(\lambda)}). \quad (91)$$

For Equation (37) observe that asymptotically

$$\log_2(3) - 1 \leq \mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda) \log_2(\lambda) \quad (92)$$

$$\iff 3\lambda^{\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)} \leq 2\lambda^{2\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)} \quad (93)$$

$$\implies \lambda^{\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)} \leq \lambda^{2\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)/3 - 1 - 2c_{\text{max}} \log_2(\lambda)} \quad (94)$$

$$\iff 2^{-\lambda^{\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)}} \geq 2\lambda^{2c_{\text{max}}} 2^{-\lambda^{\mathfrak{o}_{\text{injOWF}}^{\text{inv}}(\lambda)}/3} \quad (95)$$

$$\iff \epsilon_{\text{VRF}}^{\text{ipub}}(\lambda) \geq 2\lambda^{2c_{\text{max}}} \epsilon_{\text{injOWF}}^{\text{inv}}(\lambda)^{1/3} \quad (96)$$

$$\implies \epsilon_{\text{VRF}}^{\text{ipub}}(\lambda) \geq 2\mathfrak{a}(\lambda)\mathfrak{b}(\lambda)\epsilon_{\text{injOWF}}^{\text{inv}}(\lambda)^{1/3} \quad (37)$$

where we use that $\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda) \in \omega(\log_2 \log_2(\lambda)/\log_2(\lambda))$ for Equation (94). For Equation (38) we distinguish two cases:

- If $2\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda) \leq \lambda^{1/\log_2 \log_2 \log_2(\lambda)}$, then

$$\lambda^{1/\log_2 \log_2 \log_2(\lambda)} \cdot 5/6 \leq \lambda^{1/\log_2 \log_2 \log_2(\lambda)} - 2 \quad (97)$$

$$\iff 2^{\lambda^{1/\log_2 \log_2 \log_2(\lambda)} \cdot 5/6} \leq 2^{\lambda^{1/\log_2 \log_2 \log_2(\lambda)}}/4 \quad (98)$$

$$\implies 2^{\lambda^{\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)} \cdot 5/3} \leq 2^{\lambda^{1/\log_2 \log_2 \log_2(\lambda)}}/4 \quad (99)$$

$$\implies 2^{\lambda^{\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)} \cdot 5/3} \leq 2^{\lambda^{1/\log_2 \log_2 \log_2(\lambda)}}/3 - \lambda^{4c_{\text{max}}} 2^{(\lambda^{c_{\text{max}}/\log_2 \log_2(\lambda)})} \quad (100)$$

$$\iff 2^{\lambda^{\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)} \cdot 5/3} \leq 2^{\lambda^{\mathfrak{o}_{\text{injOWF}}^{\text{inv}}(\lambda)}}/3 - \lambda^{4c_{\text{max}}} 2^{(\lambda^{c_{\text{max}}/\log_2 \log_2(\lambda)})} \quad (101)$$

- If $2\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda) \geq \lambda^{1/\log_2 \log_2 \log_2(\lambda)}$, then

$$\log_2(5/4)/\log_2(\lambda) \leq \mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda) \quad (102)$$

$$\iff \lambda^{\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)} \cdot 5/4 \leq \lambda^{2\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)} \quad (103)$$

$$\implies \lambda^{\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)} \cdot 5/3 \leq \lambda^{2\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)} - 2 \quad (104)$$

$$\iff 2^{\lambda^{\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)} \cdot 5/3} \leq 2^{\lambda^{2\mathfrak{o}_{\text{VRF}}^{\text{ipub}}(\lambda)}}/4 \quad (105)$$

$$\implies 2^{\lambda^{\text{ipub}}_{\text{VRF}}(\lambda) \cdot 5/3} \leq 2^{\lambda^{2\text{e}^{\text{ipub}}_{\text{VRF}}(\lambda)}/3} - \lambda^{4c_{\text{max}}} 2^{(\lambda^{c_{\text{max}}/\log_2 \log_2(\lambda)})} \quad (106)$$

$$\iff 2^{\lambda^{\text{ipub}}_{\text{VRF}}(\lambda) \cdot 5/3} \leq 2^{\lambda^{\text{oinv}_{\text{injOWF}}(\lambda)}/3} - \lambda^{4c_{\text{max}}} 2^{(\lambda^{c_{\text{max}}/\log_2 \log_2(\lambda)})} \quad (107)$$

Then we conclude that

$$2^{\lambda^{\text{ipub}}_{\text{VRF}}(\lambda) \cdot 5/3} \leq 2^{\lambda^{\text{oinv}_{\text{injOWF}}(\lambda)}/3} - \lambda^{4c_{\text{max}}} 2^{(\lambda^{c_{\text{max}}/\log_2 \log_2(\lambda)})} \quad (108)$$

$$\iff \mathfrak{t}_{\text{VRF}}^{\text{ipub}}(\lambda)^{5/3} \leq \mathfrak{t}_{\text{injOWF}}^{\text{inv}}(\lambda)/3 - \lambda^{4c_{\text{max}}} 2^{\ell_v(\lambda)^{c_{\text{max}}}} \quad (109)$$

$$\implies \mathfrak{t}_{\text{VRF}}^{\text{ipub}}(\lambda)^{5/3} \leq \mathfrak{t}_{\text{injOWF}}^{\text{inv}}(\lambda)/3 - \lambda^{c_{\text{max}}} - \lambda^{2c_{\text{max}}+1} \cdot \left(2\lambda^{c_{\text{max}}} + 2^{\ell_v(\lambda)^{c_{\text{max}}}} \ell_v(\lambda)^{c_{\text{max}}} \right) \quad (110)$$

$$\implies \mathfrak{t}_{\text{VRF}}^{\text{ipub}}(\lambda) \leq \mathfrak{t}_{\text{injOWF}}^{\text{inv}}(\lambda)/3 \epsilon_{\text{injOWF}}^{\text{inv}}(\lambda)^{2/3} - \mathfrak{t}_{\text{CRH.Gen}}(\lambda) - \mathfrak{a}(\lambda)\mathfrak{b}(\lambda)\lambda \cdot \left(\frac{\mathfrak{t}_{\text{injOWF}}(\lambda) + \mathfrak{t}_{\text{CRH.Hash}}(\lambda)}{+ 2^{\mathfrak{t}_{\text{VRF'}.Eval}(\ell_v(\lambda))} \mathfrak{t}_{\text{VRF'}.Verify}(\ell_v(\lambda))} \right). \quad (38)$$

□

We analogously simplify the general statement of Theorem 8.

Corollary 4. Let PRF be a superpolynomially secure PRF. Let all other primitives be as in Corollary 3. Then VRF from Construction 2 is subexponentially fully unbiased, and superpolynomially pseudorandom.

The proof follows analogously to the proof of Corollary 3.

13 Towards a Lattice-Based VRF without Setup

VRFs are in particular used in leader election protocols of proof-of-stake blockchains. Given the vast amount of assets managed by these protocols, a rigorous proof of security is paramount. While many other applications have plausibly post-quantum instantiations, VRFs (without setup or random oracle) are an outlier.

The goal of this section is to investigate a path towards an unbiased VRF *without setup* from concrete post-quantum assumptions. In Figure 9 we outline a possible approach for an unbiased VRF. We need two ingredients: (subexponentially hard) LWE and a perfectly sound NIWI. The latter can be realized with a statistically sound NIZK in the uniform random string (URS) model under a worst-case derandomization assumption. We stress that, to obtain unique provability (without setup), it is necessary that the NIWI has *perfect* soundness. Otherwise, there might exist a verification key (i.e., a NIWI CRS) that allows to prove different images (i.e., false statements).

Theorem 17 (Unbiasable VRF from LWE and NIWI). Assuming the subexponential hardness of LWE (with arbitrary modulus), and a subexponentially witness-indistinguishable, perfectly sound NIWI (without setup), there exists a fully unbiased, polynomially adaptively pseudorandom, perfectly uniquely provable VRF.

The NIWI is implied by a subexp. zero-knowledge, statistically sound URS-NIZK, and $\mathbf{E} \not\subseteq \bigcap_{c \geq 0} \mathbf{NSIZE}[2^{c\lambda}]$, i.e., a (uniformly) linear-exponential-time computable function requires a $2^{\Omega(\lambda)}$ -size nondeterministic circuit.

This theorem follows from applying our compiler to the generic VRF construction of Bitansky [Bit20] and Goyal, Hohenberger, Koppula, and Waters [GHK⁺17b] where the injective OWF is instantiated from LWE (cf. Lemma 4).

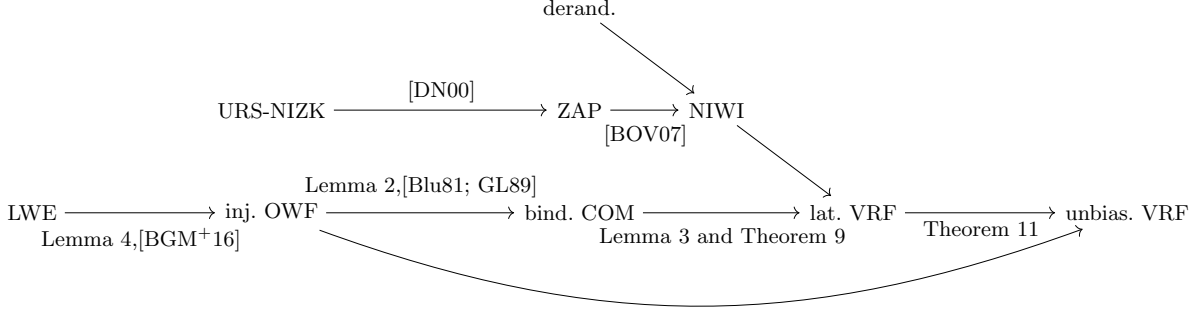


Fig. 9: Overview of our construction from LWE and derandomization. We require the following: the URS-NIZK is statistical sound, the commitment scheme is perfectly binding, the ZAP is public-coin.

Remark 17 (Derandomization Assumptions). The complexity-theoretic assumption $E \not\subseteq \bigcap_{c \geq 0} \text{NSIZE}[2^{c\lambda}]$ is a non-uniform assumption in the sense that a uniform class $E := \text{DTIME}[2^{\mathcal{O}(\lambda)}]$ needs to have a language outside the *non-uniform* class $\text{NSIZE}[2^{c\lambda}]$. If one prefers a purely uniform assumption, then one may alternatively assume that $\exists c \geq 0 : E \not\subseteq \text{AMTIME}[2^{c\lambda}]$ [GSTS03] where $\text{AMTIME}[t]$ is the class of Arthur–Merlin games where Arthur runs in $\text{BPTIME}[t]$.

The reason why these types of assumptions are called “derandomization” assumptions is that $E \not\subseteq \text{SIZE}[2^{c\lambda}]$ implies derandomization with polynomial overhead, i.e., $P = \text{BPP}$ [IW97].

Alternatively, the derandomization assumption can be lifted if the VRF is allowed to be non-uniform, i.e., the VRF algorithms get as additional input a length-dependent advice that is independent of the other inputs.

An injective OWF from LWE. First, we prove that LWE implies a keyless injective OWF. Our starting point is a PRG *with setup* from [GKV⁺20] that is based on the hardness of learning-with-rounding (LWR). Optimally, we would like to assume only polynomial hardness of LWE with arbitrary modulus. We note that the hardness of LWR with polynomial modulus cannot be based on the hardness of LWE with polynomial modulus [NR23] if the dimension m is not a priori fixed. Fortunately, for the PRG construction of [GKV⁺20] the dimension m can be fixed a priori; hence we can base the hardness of LWR with polynomial modulus on the hardness of LWE with polynomial modulus [BGM⁺16].

Lemma 4 (Injective OWF from LWE (based on [GKV⁺20])). There exists a polynomially hard injective OWF injOWF based on the polynomial hardness of (n, m, q, n^2) -LWE with modulus $q(\lambda) \geq n(\lambda)^6$.

Proof Sketch. First, we show injectivity. Let $\text{PRG} = (\text{PRG.Setup}, \text{PRG.Eval})$ be the PRG with setup from [GKV⁺20] (see Section 3 in [GKV⁺20]) with polynomial parameters

$$n(\lambda) \geq \lambda \quad q(\lambda) := n(\lambda)^6 \quad m(\lambda) := 3n(\lambda) \log(q(\lambda)) \quad p(\lambda) := 4m(\lambda) \quad \rho(\lambda) := q(\lambda)/2. \quad (111)$$

Let $D := \rho(\lambda) \cdot (I_n \mid 0^{n(\lambda) \times (m(\lambda) - n(\lambda))})$ be a fixed matrix. Then

$$\text{injOWF}(\text{pp}, s) := (\text{pp}, \text{PRG.Eval}(\text{pp}, s)) = \left(\text{pp}, [s^\top \text{pp}]_{p(\lambda)} \right) \quad (112)$$

is an injective OWF with domain

$$\mathcal{D}_\lambda := \left\{ (\text{pp} = (B \mid BR + D), s) \mid \begin{array}{l} B \in \mathbb{Z}_{q(\lambda)}^{n(\lambda) \times m(\lambda)}, \text{rank}(B) = n(\lambda) \\ R \in \{-1, 1\}^{m(\lambda) \times m(\lambda)}, s \in \{0, 1\}^{n(\lambda)} \end{array} \right\}$$

and codomain $\mathcal{C} := \mathbb{Z}_{q(\lambda)}^{n(\lambda) \times 2m(\lambda)} \times \mathbb{Z}_{p(\lambda)}^{2m(\lambda)}$.

We leverage the injectivity of PRG (with setup) which states formally:

$$\forall \mathbf{pp} \in \text{PRG.Setup}(1^\lambda) : \text{PRG.Eval}(\mathbf{pp}, s) = \text{PRG.Eval}(\mathbf{pp}, s') \implies s = s'. \quad (113)$$

Thus, let

$$\left(\mathbf{pp}, \lfloor s^\top \mathbf{pp} \rfloor_{p(\lambda)} \right) = \text{injOWF}(\mathbf{pp}, s) = \text{injOWF}(\mathbf{pp}', s') = \left(\mathbf{pp}', \lfloor s'^\top \mathbf{pp}' \rfloor_{p(\lambda)} \right). \quad (114)$$

It follows obviously that $\mathbf{pp} = \mathbf{pp}'$ and thus $\text{PRG.Eval}(\mathbf{pp}, s) = \lfloor s^\top \mathbf{pp} \rfloor_{p(\lambda)} = \lfloor s'^\top \mathbf{pp}' \rfloor_{p(\lambda)} = \text{PRG.Eval}(\mathbf{pp}, s')$. By the injectivity of PRG it follows that $s = s'$. Thus, injOWF is injective.

Now, we argue one-wayness by a straightforward reduction to the pseudorandomness of PRG. The reduction $\mathcal{R}$ gets parameters $\mathbf{pp}$ and an image $y' \in \mathcal{C}$ from the pseudorandomness challenger. It forwards the image $y := (\mathbf{pp}, y')$ to the OWF challenger. The reduction obtains a preimage x . If $\text{PRG.Eval}(\mathbf{pp}, x) = y'$, then $\mathcal{R}$ outputs $b' := 0$ (pseudorandom), otherwise it outputs a random bit $b' \leftarrow \{0, 1\}$. We find that

$$\epsilon_{\text{PRG}}^{\text{pr}}(\lambda) \geq \epsilon_{\text{injOWF}}^{\text{inv}}(\lambda) - \Pr[y' \in \mathcal{C} \mid b = 1] = \epsilon_{\text{injOWF}}^{\text{inv}}(\lambda) - (2/q(\lambda))^{m(\lambda)}. \quad (115)$$

Finally, $\epsilon_{\text{PRG}}^{\text{pr}}(\lambda)$ is polynomially bounded by Theorem 3.1. in [GKV⁺20] if $(\mathbf{n}, \mathbf{m}, \mathbf{q}, \mathbf{p})$ -LWR is polynomially hard. Let $\mathbf{b}(\lambda) \leq q(\lambda)/2m(\lambda)p(\lambda)$ be an LWE error bound, e.g. $\mathbf{b}(\lambda) := q(\lambda)^{1/2}$ for our parameters. Then we can apply Theorem 3 in [BGM⁺16]. Thus, polynomial hardness of $(\mathbf{n}, \mathbf{m}, \mathbf{q}, \mathbf{p})$ -LWR is implied by polynomial hardness of $(\mathbf{n}, \mathbf{m}, \mathbf{q}, \mathbf{n}^2)$ -LWE with standard deviation $n(\lambda)^2 \leq \mathbf{b}(\lambda)/\lambda$. $\square$

Remark 18. The domain $\mathcal{D}_\lambda$ of the injective OWF injOWF in Lemma 4 is not completely uniform but it is exponentially close to uniform on $\mathbb{Z}_{q(\lambda)}^{n(\lambda) \times m(\lambda)} \times \mathbb{Z}_{q(\lambda)}^{n(\lambda) \times m(\lambda)} \times \{0, 1\}^{n(\lambda)}$.

On NIWIs from LWE. In this section we discuss the gaps that prevent us from constructing a NIWI (and thus an unbiased VRF) from plain LWE. Unfortunately, we cannot—as Chatterjee, Garg, Hajiabadi, Khurana, Liang, Malavolta, Pandey, and Shiehian [CGH⁺21] state—directly instantiate NIWIs (for NP, without setup) from LWE (see Section 1.3 in [CGH⁺21]). In the following we briefly survey the current state-of-the-art for non-interactive proofs (see Figure 10).

- Subexponentially hard LWE implies SRS-NIWIs (for all NP) with *computational* soundness [BFJ⁺20; GJJ⁺20; LVW19].
- Chatterjee, Garg, Hajiabadi, Khurana, Liang, Malavolta, Pandey, and Shiehian [CGH⁺21] construct a (computationally sound) URS-ZAP for $\text{NP} \cap \text{coNP}$ from polynomially hard LWE.
- Peikert and Shiehian [PS19] construct a (statistically sound) SRS-NIZK (resp. a computationally sound URS-NIZK) for NP from polynomially hard LWE.
- Canetti and Lichtenberg [CL18] construct a (statistically sound) NIZK by instantiating the FLS-paradigm with doubly-enhanced certifiably injective trapdoor functions.
- Barak, Ong, and Vadhan [BOV07] derandomize a statistically sound URS-ZAP [DN00] to obtain a NIWI (without setup) with perfect soundness.
- Recently, Branco, Choudhuri, Döttling, Jain, Malavolta, and Srinivasan [BCD⁺25] obtained a URS-NIZK from the decisional Diffie–Hellman assumption and a variant of the learning-with-parity (LPN) assumption. While this comes tantalizingly close to a lattice-based VRF, the DDH assumption does not hold against quantum adversaries [Sho94].

In conclusion, an LWE-based statistically sound URS-ZAP or URS-NIZK would yield the necessary NIWI for our construction under derandomization. Other approaches for constructing NIWIs we are aware of are either pairing-based [GOS12] or based on indistinguishability obfuscation [BP15]. We note that the NIWI from [BP15] is only statistically sound but our application requires perfect soundness.

For completeness, we also mention that a VRF with uniform domain would imply a URS-NIZK and under derandomization a NIWI via the construction of Brakerski, Goldwasser, Rothblum, and Vaikuntanathan [BGR⁺09] and Barak, Ong, and Vadhan [BOV03]. Thus, basing VRFs on post-quantum as-

Setup / Soundness	Computational	Statistical
URS	NIWI [LVW19; BFJ ⁺ 20; GJJ ⁺ 20] from subexp. LWE NIZK [PS19] from poly. LWE	NIZK [BCD ⁺ 25] from DDH and LPN NIZK [CL18] from DECITDF
SRS	ZAP for $\text{NP} \cap \text{coNP}$ [CGH ⁺ 21] from poly. LWE	NIZK [PS19] from poly. LWE

Fig. 10: Overview of non-interactive proofs. URS stands for a uniformly random string, SRS for a structured random string. A statistically sound URS-NIZK would suffice for unbiased VRFs (under derandomization). DDH stands for the decisional Diffie–Hellman assumption and LPN for the learning with parity assumption. DECITDF stands for double-enhanced certifiably injective trapdoor function.

sumptions would also yield URS-NIZKs and NIWI based on the same assumptions (up to derandomization).

On CPRFs from LWE. Bitansky [Bit20] and Goyal, Hohenberger, Koppula, and Waters [GHK⁺17b] require a constrained PRF (CPRF) to achieve adaptive pseudorandomness from polynomially hard assumptions. The CPRF needs to support a sufficiently expressive constraint function class (in particular partitioning schemes). We refer the interested reader to [Bit20; GHK⁺17b] for details. The CPRF construction of Brakerski and Vaikuntanathan [BV15] supports such constraints. However, they require two different lattice assumptions; namely, LWE and a one-dimensional SIS (1D-SIS) assumption. Thus, such a CPRF is based solely on lattice assumptions but not exclusively on plain LWE.

Other Generic Approaches. We mention that there is another generic approach towards constructing VRFs (besides [GHK⁺17b; Bit20]), namely, Abdalla, Catalano, and Fiore [ACF09] build VRFs from a specific class of identity-based key-encapsulation mechanisms (IB-KEM) that they call VRF-suitable IB-KEMs. VRF-suitable IB-KEMs fulfill two additional properties (verbatim from [ACF09] up to typography):

- Unique decryption: Let id_0 be any valid identity and C a ciphertext encrypted under id_0 . We require that no valid identity id can satisfy (unless with negligible probability) $\text{Dec}(C, \text{sk}'_{\text{id}}, \text{id}, \text{aux}'_{\text{id}}) = \text{Dec}(C, \text{sk}''_{\text{id}}, \text{aux}''_{\text{id}})$ where $(\text{sk}'_{\text{id}}, \text{aux}'_{\text{id}}) \leftarrow \text{KeyDer}(\text{msk}, \text{id})$, $(\text{sk}''_{\text{id}}, \text{aux}''_{\text{id}}) \leftarrow \text{KeyDer}(\text{msk}, \text{id})$.
- Pseudorandom decapsulation: Let C be an encapsulation produced using identity id_0 , we require the session key to look random even if the decapsulation algorithm is executed using the secret key corresponding to any other $\bar{\text{id}}$.

Roughly, the VRF verification key $\text{vk} = (\text{mpk}, C_0)$ comprises the IB-KEM’s master public key mpk and a ciphertext C_0 generated relative to mpk and an arbitrary identity id_0 . To evaluate the VRF on a preimage, the preimage is interpreted as an identity and the corresponding VRF image is the user secret key corresponding to that identity.

We note that while the application of their transformation to the IB-KEM of Sakai and Kasahara [SK03] works as stated, their condition of unique decryption is not sufficient to obtain a VRF from *each* (VRF-suitable) IB-KEM. The reason is that for VRFs unique provability must hold for *every* verification key $\text{vk} = (\text{mpk}, C_0)$. However, if the ciphertext C_0 is not a valid ciphertext under mpk, id_0 , then the VRF-suitability as stated in [ACF09] does not yield uniqueness because its condition is violated. In the case of group-based IB-KEMs (in particular for [SK03])—typically—any group element constitutes a valid ciphertext under any mpk, id_0 ; thus the condition of VRF-suitability is trivially satisfied. However, for the transformation to work for *each* (suitable) IB-KEM we need to require a stronger form of unique decryption.

- Strong unique decryption: Let id_0 be any valid identity and C *any* element of the ciphertext space. We require that no valid identity id can satisfy (unless with negligible probability) $\text{Dec}(C, \text{sk}'_{\text{id}}, \text{id}, \text{aux}'_{\text{id}}) = \text{Dec}(C, \text{sk}''_{\text{id}}, \text{aux}''_{\text{id}})$ where $(\text{sk}'_{\text{id}}, \text{aux}'_{\text{id}}) \leftarrow \text{KeyDer}(\text{msk}, \text{id})$, $(\text{sk}''_{\text{id}}, \text{aux}''_{\text{id}}) \leftarrow \text{KeyDer}(\text{msk}, \text{id})$.

In contrast to the weaker unique decryption condition, this condition does not follow automatically even if the IB-KEM construction has (honestly generated) unique user secret keys, because for an identity there could be two different user secret keys *for a malformed master public key*.

Remark 19. We are currently not aware of any lattice-based IB-KEM that satisfies this stronger notion of unique decryption (or even unique user secret keys); hence our focus on the approaches of [GHK⁺17b; Bit20] towards a lattice-based VRF.

References

- [ACF09] M. Abdalla, D. Catalano, and D. Fiore. Verifiable Random Functions from Identity-Based Key Encapsulation. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 554–571. Springer, Berlin, Heidelberg, April 2009.
- [ASM07] M. H. Au, W. Susilo, and Y. Mu. Practical Compact E-Cash. In J. Pieprzyk, H. Ghodosi, and E. Dawson, editors, *ACISP 07*, volume 4586 of *LNCS*, pages 431–445. Springer, Berlin, Heidelberg, July 2007.
- [BBF13] P. Baecher, C. Brzuska, and M. Fischlin. Notions of Black-Box Reductions, Revisited. In K. Sako and P. Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 296–315. Springer, Berlin, Heidelberg, December 2013.
- [BBG⁺23] M. Backendal, M. Bellare, F. Günther, and M. Scarlata. When Messages Are Keys: Is HMAC a Dual-PRF? In H. Handschuh and A. Lysyanskaya, editors, *CRYPTO 2023, Part III*, volume 14083 of *LNCS*, pages 661–693. Springer, Cham, August 2023.
- [BCA⁺25] J. Burdges, E. C. Crites, H. K. Alper, A. Stewart, and S. Vasilyev. Sassafras: Efficient Batch Single Leader Election. In M. Fischlin and V. Moonsamy, editors, *ACNS 2025, Part I*, volume 15825 of *LNCS*, pages 244–273. Springer, Cham, June 2025.
- [BCD⁺25] P. Branco, A. R. Choudhuri, N. Döttling, A. Jain, G. Malavolta, and A. Srinivasan. Black-Box Non-interactive Zero Knowledge from Vector Trapdoor Hash. In S. Fehr and P.-A. Fouque, editors, *EUROCRYPT 2025, Part IV*, volume 15604 of *LNCS*, pages 64–92. Springer, Cham, May 2025.
- [BCK⁺09] M. Belenkiy, M. Chase, M. Kohlweiss, and A. Lysyanskaya. Compact E-Cash and Simulatable VRFs Revisited. In H. Shacham and B. Waters, editors, *PAIRING 2009*, volume 5671 of *LNCS*, pages 114–131. Springer, Berlin, Heidelberg, August 2009.
- [BFJ⁺20] S. Badrinarayanan, R. Fernando, A. Jain, D. Khurana, and A. Sahai. Statistical ZAP Arguments. In A. Canteaut and Y. Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 642–667. Springer, Cham, May 2020.
- [BGI14] E. Boyle, S. Goldwasser, and I. Ivan. Functional Signatures and Pseudorandom Functions. In H. Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 501–519. Springer, Berlin, Heidelberg, March 2014.
- [BGM⁺16] A. Bogdanov, S. Guo, D. Masny, S. Richelson, and A. Rosen. On the Hardness of Learning with Rounding over Small Modulus. In E. Kushilevitz and T. Malkin, editors, *TCC 2016-A, Part I*, volume 9562 of *LNCS*, pages 209–224. Springer, Berlin, Heidelberg, January 2016.
- [BGR⁺09] Z. Brakerski, S. Goldwasser, G. N. Rothblum, and V. Vaikuntanathan. Weak Verifiable Random Functions. In O. Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 558–576. Springer, Berlin, Heidelberg, March 2009.
- [Bit17] N. Bitansky. Verifiable Random Functions from Non-interactive Witness-Indistinguishable Proofs. In Y. Kalai and L. Reyzin, editors, *TCC 2017, Part II*, volume 10678 of *LNCS*, pages 567–594. Springer, Cham, November 2017.
- [Bit20] N. Bitansky. Verifiable Random Functions from Non-interactive Witness-Indistinguishable Proofs. *Journal of Cryptology*, 33(2):459–493, April 2020.
- [BL15] M. Bellare and A. Lysyanskaya. Symmetric and Dual PRFs from Standard Assumptions: A Generic Validation of an HMAC Assumption. Cryptology ePrint Archive, Report 2015/1198, 2015.
- [BL24] M. Bellare and A. Lysyanskaya. Symmetric and Dual PRFs from Standard Assumptions: A Generic Validation of a Prevailing Assumption. *Journal of Cryptology*, 37(4):33, October 2024.

- [BLM⁺13] D. Boneh, K. Lewi, H. W. Montgomery, and A. Raghunathan. Key Homomorphic PRFs and Their Applications. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 410–428. Springer, Berlin, Heidelberg, August 2013.
- [Blu81] M. Blum. Coin Flipping by Telephone. In A. Gersho, editor, *CRYPTO’81*, volume ECE Report 82-04, pages 11–15. U.C. Santa Barbara, Dept. of Elec. and Computer Eng., 1981.
- [BOV03] B. Barak, S. J. Ong, and S. P. Vadhan. Derandomization in Cryptography. In D. Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 299–315. Springer, Berlin, Heidelberg, August 2003.
- [BOV07] B. Barak, S. J. Ong, and S. Vadhan. Derandomization in Cryptography. *SIAM Journal on Computing*, 37(2):380–400, 2007. eprint: <https://doi.org/10.1137/050641958>.
- [BP15] N. Bitansky and O. Paneth. ZAPs and Non-Interactive Witness Indistinguishability from Indistinguishability Obfuscation. In Y. Dodis and J. B. Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 401–427. Springer, Berlin, Heidelberg, March 2015.
- [BPR12] A. Banerjee, C. Peikert, and A. Rosen. Pseudorandom Functions and Lattices. In D. Pointcheval and T. Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 719–737. Springer, Berlin, Heidelberg, April 2012.
- [BV15] Z. Brakerski and V. Vaikuntanathan. Constrained Key-Homomorphic PRFs from Standard Lattice Assumptions - Or: How to Secretly Embed a Circuit in Your PRF. In Y. Dodis and J. B. Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 1–30. Springer, Berlin, Heidelberg, March 2015.
- [BW13] D. Boneh and B. Waters. Constrained Pseudorandom Functions and Their Applications. In K. Sako and P. Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 280–300. Springer, Berlin, Heidelberg, December 2013.
- [BZ14] D. Boneh and M. Zhandry. Multiparty Key Exchange, Efficient Traitor Tracing, and More from Indistinguishability Obfuscation. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 480–499. Springer, Berlin, Heidelberg, August 2014.
- [Can01] R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001.
- [CCF⁺25] D. Carnemolla, D. Catalano, V. Frasca, and E. Giunta. On the (Un)biasability of Existing Verifiable Random Functions. Cryptology ePrint Archive, Report 2025/2176, 2025.
- [CF01] R. Canetti and M. Fischlin. Universally Composable Commitments. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 19–40. Springer, Berlin, Heidelberg, August 2001.
- [CG85] B. Chor and O. Goldreich. Unbiased Bits from Sources of Weak Randomness and Probabilistic Communication Complexity (Extended Abstract). In *26th FOCS*, pages 429–442. IEEE Computer Society Press, October 1985.
- [CGH⁺21] R. Chatterjee, S. Garg, M. Hajiabadi, D. Khurana, X. Liang, G. Malavolta, O. Pandey, and S. Shiehian. Compact Ring Signatures from Learning with Errors. In T. Malkin and C. Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 282–312, Virtual Event. Springer, Cham, August 2021.
- [CGV15] A. Cohen, S. Goldwasser, and V. Vaikuntanathan. Aggregate Pseudorandom Functions and Connections to Learning. In Y. Dodis and J. B. Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 61–89. Springer, Berlin, Heidelberg, March 2015.
- [CL18] R. Canetti and A. Lichtenberg. Certifying Trapdoor Permutations, Revisited. In A. Beimel and S. Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 476–506. Springer, Cham, November 2018.
- [CM17] J. Chen and S. Micali. Algorand, 2017. arXiv: 1607.01341 [cs.CR].
- [DGK⁺18] B. David, P. Gazi, A. Kiayias, and A. Russell. Ouroboros Praos: An Adaptively-Secure, Semi-synchronous Proof-of-Stake Blockchain. In J. B. Nielsen and V. Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 66–98. Springer, Cham, 2018.
- [DN00] C. Dwork and M. Naor. Zaps and Their Applications. In *41st FOCS*, pages 283–293. IEEE Computer Society Press, November 2000.
- [DP07] Y. Dodis and P. Puniya. Feistel Networks Made Public, and Applications. In M. Naor, editor, *EUROCRYPT 2007*, volume 4515 of *LNCS*, pages 534–554. Springer, Berlin, Heidelberg, May 2007.

- [DY05] Y. Dodis and A. Yampolskiy. A Verifiable Random Function with Short Proofs and Keys. In S. Vaudenay, editor, *PKC 2005*, volume 3386 of *LNCS*, pages 416–431. Springer, Berlin, Heidelberg, January 2005.
- [EKS⁺21] M. F. Esgin, V. Kuchta, A. Sakzad, R. Steinfeld, Z. Zhang, S. Sun, and S. Chu. Practical Post-quantum Few-Time Verifiable Random Function with Applications to Algorand. In N. Borisov and C. Díaz, editors, *FC 2021, Part II*, volume 12675 of *LNCS*, pages 560–578. Springer, Berlin, Heidelberg, March 2021.
- [FS90] U. Feige and A. Shamir. Zero Knowledge Proofs of Knowledge in Two Rounds. In G. Brassard, editor, *CRYPTO’89*, volume 435 of *LNCS*, pages 526–544. Springer, New York, August 1990.
- [Gei24] M. Geihs. Great-LaKeys: An Improved Threshold-PRF and a Novel Exponent-VRF from LWR. Cryptology ePrint Archive, Report 2024/996, 2024.
- [GGM86] O. Goldreich, S. Goldwasser, and S. Micali. How to Construct Random Functions. *Journal of the ACM*, 33(4):792–807, October 1986.
- [GHK⁺17a] R. Goyal, S. Hohenberger, V. Koppula, and B. Waters. A Generic Approach to Constructing and Proving Verifiable Random Functions. Cryptology ePrint Archive, Report 2017/021, 2017.
- [GHK⁺17b] R. Goyal, S. Hohenberger, V. Koppula, and B. Waters. A Generic Approach to Constructing and Proving Verifiable Random Functions. In Y. Kalai and L. Reyzin, editors, *TCC 2017, Part II*, volume 10678 of *LNCS*, pages 537–566. Springer, Cham, November 2017.
- [GJJ⁺20] V. Goyal, A. Jain, Z. Jin, and G. Malavolta. Statistical Zaps and New Oblivious Transfer Protocols. In A. Canteaut and Y. Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 668–699. Springer, Cham, May 2020.
- [GKV⁺20] R. Goyal, V. Koppula, S. Vusirikala, and B. Waters. On Perfect Correctness in (Lockable) Obfuscation. In R. Pass and K. Pietrzak, editors, *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 229–259. Springer, Cham, November 2020.
- [GL89] O. Goldreich and L. A. Levin. A Hard-Core Predicate for all One-Way Functions. In *21st ACM STOC*, pages 25–32. ACM Press, May 1989.
- [GO93] S. Goldwasser and R. Ostrovsky. Invariant Signatures and Non-Interactive Zero-Knowledge Proofs are Equivalent (Extended Abstract). In E. F. Brickell, editor, *CRYPTO’92*, volume 740 of *LNCS*, pages 228–245. Springer, Berlin, Heidelberg, August 1993.
- [GOS12] J. Groth, R. Ostrovsky, and A. Sahai. New Techniques for Noninteractive Zero-Knowledge. *J. ACM*, 59(3), June 2012.
- [GS24] E. Giunta and A. Stewart. Unbiasable Verifiable Random Functions. In M. Joye and G. Leander, editors, *EUROCRYPT 2024, Part IV*, volume 14654 of *LNCS*, pages 142–167. Springer, Cham, May 2024.
- [GSTS03] D. Gutfreund, R. Shaltiel, and A. Ta-Shma. Uniform hardness vs. randomness tradeoffs for Arthur-Merlin games. In *18th IEEE Annual Conference on Computational Complexity, 2003. Proceedings*. Pages 33–47, 2003.
- [GW11] C. Gentry and D. Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In L. Fortnow and S. P. Vadhan, editors, *43rd ACM STOC*, pages 99–108. ACM Press, June 2011.
- [HIL⁺99] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A Pseudorandom Generator from any One-way Function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.
- [HJ16] D. Hofheinz and T. Jager. Verifiable Random Functions from Standard Assumptions. In E. Kushilevitz and T. Malkin, editors, *TCC 2016-A, Part I*, volume 9562 of *LNCS*, pages 336–362. Springer, Berlin, Heidelberg, January 2016.
- [ILL89] R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random Generation from one-way functions (Extended Abstracts). In *21st ACM STOC*, pages 12–24. ACM Press, May 1989.
- [IW97] R. Impagliazzo and A. Wigderson. P = BPP if E Requires Exponential Circuits: Derandomizing the XOR Lemma. In *29th ACM STOC*, pages 220–229. ACM Press, May 1997.
- [JS04] S. Jarecki and V. Shmatikov. Handcuffing Big Brother: an Abuse-Resilient Transaction Escrow Scheme. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 590–608. Springer, Berlin, Heidelberg, May 2004.

- [KPT⁺13] A. Kiayias, S. Papadopoulos, N. Triandopoulos, and T. Zacharias. Delegatable pseudorandom functions and applications. In A.-R. Sadeghi, V. D. Gligor, and M. Yung, editors, *ACM CCS 2013*, pages 669–684. ACM Press, November 2013.
- [KR00] H. Krawczyk and T. Rabin. Chameleon Hashing and Signatures. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2000.
- [KRD⁺17] A. Kiayias, A. Russell, B. David, and R. Oliynykov. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. In J. Katz and H. Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 357–388. Springer, Cham, August 2017.
- [KRS25] D. Khovratovich, R. D. Rothblum, and L. Soukhanov. How to Prove False Statements: Practical Attacks on Fiat-Shamir. In Y. T. Kalai and S. F. Kamara, editors, *CRYPTO 2025, Part VI*, volume 16005 of *LNCS*, pages 3–26. Springer, Cham, August 2025.
- [Lai24] Y.-F. Lai. Cappybara and Tsubaki: Verifiable Random Functions from Group Actions and Isogenies. *CiC*, 1(3):1, 2024.
- [LBM20] B. Liang, G. Banegas, and A. Mitroksa. Statically Aggregate Verifiable Random Functions and Application to E-Lottery. *Cryptography*, 4(4), 2020.
- [Ler25a] A. Leroux. Personal Communication. In 2025.
- [Ler25b] A. Leroux. Verifiable Random Function from the Deuring Correspondence and Higher Dimensional Isogenies. In S. Fehr and P.-A. Fouque, editors, *EUROCRYPT 2025, Part VII*, volume 15607 of *LNCS*, pages 167–194. Springer, Cham, May 2025.
- [Lis05] M. Liskov. Updatable Zero-Knowledge Databases. In B. K. Roy, editor, *ASIACRYPT 2005*, volume 3788 of *LNCS*, pages 174–198. Springer, Berlin, Heidelberg, December 2005.
- [LLH⁺23] Y. Li, S. Liu, S. Han, D. Gu, and J. Weng. Simulatable verifiable random function from the LWE assumption. *Theoretical Computer Science*, 957:113826, 2023.
- [LMN89] N. Linial, Y. Mansour, and N. Nisan. Constant Depth Circuits, Fourier Transform, and Learnability. In *30th FOCS*, pages 574–579. IEEE Computer Society Press, 1989.
- [LVW19] A. Lombardi, V. Vaikuntanathan, and D. Wichs. 2-Message Publicly Verifiable WI from (Subexponential) LWE. Cryptology ePrint Archive, Report 2019/808, 2019.
- [Mal24] G. Malavolta. Key-Homomorphic and Aggregate Verifiable Random Functions. In E. Boyle and M. Mahmoody, editors, *TCC 2024, Part IV*, volume 15367 of *LNCS*, pages 98–129. Springer, Cham, December 2024.
- [MP23] N. Mazon and R. Pass. Counting Unpredictable Bits: A Simple PRG from One-Way Functions. In G. N. Rothblum and H. Wee, editors, *TCC 2023, Part I*, volume 14369 of *LNCS*, pages 191–218. Springer, Cham, 2023.
- [MR01] S. Micali and L. Reyzin. Soundness in the Public-Key Model. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 542–565. Springer, Berlin, Heidelberg, August 2001.
- [MR02] S. Micali and R. L. Rivest. Micropayments Revisited. In B. Preneel, editor, *CT-RSA 2002*, volume 2271 of *LNCS*, pages 149–163. Springer, Berlin, Heidelberg, February 2002.
- [MRV99] S. Micali, M. O. Rabin, and S. P. Vadhan. Verifiable Random Functions. In *40th FOCS*, pages 120–130. IEEE Computer Society Press, October 1999.
- [Nao03] M. Naor. On Cryptographic Assumptions and Challenges (Invited Talk). In D. Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 96–109. Springer, Berlin, Heidelberg, August 2003.
- [NR23] P. Newton and S. Richelson. A Lower Bound for Proving Hardness of Learning with Rounding with Polynomial Modulus. In H. Handschuh and A. Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 805–835. Springer, Cham, August 2023.
- [PS19] C. Peikert and S. Shiehian. Noninteractive Zero Knowledge for NP from (Plain) Learning with Errors. In A. Boldyreva and D. Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 89–114. Springer, Cham, August 2019.
- [Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In H. N. Gabow and R. Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.
- [RTV04] O. Reingold, L. Trevisan, and S. P. Vadhan. Notions of Reducibility between Cryptographic Primitives. In M. Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 1–20. Springer, Berlin, Heidelberg, February 2004.

- [SD23] N. Stephens-Davidowitz. The Goldreich-Levin Theorem. 2023. URL: https://www.noahsd.com/crypto_lecture_notes/CS4830_Lecture_6___The_Goldreich_Levin_Theorem__hardcore_bits_from_any_OWF.pdf.
- [Sho94] P. W. Shor. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In *35th FOCS*, pages 124–134. IEEE Computer Society Press, November 1994.
- [SK03] R. Sakai and M. Kasahara. ID based Cryptosystems with Pairing on Elliptic Curve. Cryptology ePrint Archive, Report 2003/054, 2003.
- [YAZ⁺19] R. Yang, M. H. Au, Z. Zhang, Q. Xu, Z. Yu, and W. Whyte. Efficient Lattice-Based Zero-Knowledge Arguments with Standard Soundness: Construction and Applications. In A. Boldyreva and D. Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 147–175. Springer, Cham, August 2019.