Verifiable Random Functions from Weaker Assumptions*

Tibor Jager

Horst Görtz Institute for IT Security
Ruhr-University Bochum
tibor.jager@rub.de

Abstract. The construction of a verifiable random function (VRF) with large input space and full adaptive security from a static, non-interactive complexity assumption, like decisional Diffie-Hellman, has proven to be a challenging task. To date it is not even clear that such a VRF exists. Most known constructions either allow only a small input space of polynomially-bounded size, or do not achieve full adaptive security under a static, non-interactive complexity assumption.

The only known constructions without these restrictions are based on non-static, so-called "q-type" assumptions, which are parametrized by an integer q. Since q-type assumptions get stronger with larger q, it is desirable to have q as small as possible. In current constructions, q is either a polynomial (e.g., Hohenberger and Waters, Eurocrypt 2010) or at least linear (e.g., Boneh *et al.*, CCS 2010) in the security parameter.

We show that it is possible to construct relatively simple and efficient verifiable random functions with full adaptive security and large input space from non-interactive q-type assumptions, where q is only *logarithmic* in the security parameter. Interestingly, our VRF is essentially identical to the verifiable *unpredictable* function (VUF) by Lysyanskaya (Crypto 2002), but very different from Lysyanskaya's VRF from the same paper. Thus, our result can also be viewed as a new, direct VRF-security proof for Lysyanskaya's VUF. As a technical tool, we introduce and construct *balanced* admissible hash functions.

1 Introduction

Verifiable random functions. Verifiable random functions (VRFs) can be seen as the public-key equivalent of pseudorandom functions. Each function V_{sk} is associated with a secret key sk and a corresponding public verification key vk. Given sk, an element X from the domain of V_{sk}, and $Y = V_{sk}(X)$, it is possible to create a *non-interactive, publicly verifiable* proof π that Y was computed correctly. For security, *unique provability* is required. This means that for each X only one unique value Y such that the statement "$Y = V_{sk}(X)$" can be proven may exist. Note that unique provability is a very strong requirement: not even the party that creates sk (possibly maliciously) may be able to create fake proofs. These additional features should not affect the pseudorandomness of the function on other inputs. Verifiable random functions are strongly related to verifiable *unpredictable* functions (VUFs), where the weaker notion of *unpredictability* instead of pseudorandomness is required.

Their strong properties make VRFs useful for applications like resettable zero-knowledge proofs [30], lottery systems [31], transaction escrow schemes [26], updatable zero-knowledge databases [27], or e-cash [3, 4]. VRFs can also be seen as verifiably unique digital signatures (called *invariant signatures* in [23]), their uniqueness makes them *strongly unforgeable* [10, 35].

The difficulty of constructing VRFs. In particular the *unique provability* requirement makes it very difficult to construct verifiable random functions. For instance, the natural attempt of combining a pseudorandom function with a non-interactive zero-knowledge proof system fails, since zero-knowledge proofs are inherently simulatable, which contradicts uniqueness. More generally, any reduction which attempts to prove pseudorandomness of a candidate construction faces the following problem.

- On the one hand, the reduction *must* be able to compute the unique function value $Y := V_{sk}(X)$ for preimages X selected by the attacker, along with a proof of correctness π. Due to the unique provability, there exists only one unique value Y such that the statement "$Y = V_{sk}(X)$" can be proven, thus the reduction is not able to "lie" by outputting false values $\tilde{Y}$.
  Note that this stands in contrast to typical reductions for pseudorandom functions, like the Naor-Reingold construction [33] for instance, where due to the absence of proofs the reduction is be able to output incorrect values.
- On the other hand, the reduction *must not* be able to compute $Y^* = V_{sk}(X^*)$ for a particular X^*, as it must be able to use an attacker that distinguishes Y^* from random to break a complexity assumption.

Most previous works [29, 28, 16, 17, 1] constructed VRFs with only small input spaces of polynomially-bounded size.[1] The only two exceptions are due to Hohenberger and Waters [25] and Boneh *et al.* [9], who constructed verifiable random functions with full adaptive security that allow an input space of exponential size.

VRFs with large input spaces from non-interactive assumptions. Hohenberger and Waters [25] provided the first fully-secure VRF with exponential-size input space whose security is based on a non-interactive complexity assumption. The security proof relies on a q-type assumption, where an algorithm receives as input a list of group elements

$$(g, h, g^x, \dots, g^{x^{q-1}}, g^{x^{q+1}}, \dots, g^{x^{2q}}, T) \in \mathbb{G}^{2q+1} \times \mathbb{G}_T$$

where $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear map. The assumption is that no efficient algorithm is able to distinguish $T = e(g, h)^{x^q}$ from a random group element with probability significantly better than $1/2$. The proof given in [25] requires that $q = \Theta(Q \cdot k)$, where

[1] Or, alternatively but usually equivalently, based on interactive complexity assumptions or with only weaker *selective* security.

k is the security parameter and Q is the number of function evaluations $V_{sk}(X)$ queried by the attacker in the security experiment. Note that in particular Q can be very large, as it is only bounded by a polynomial in the security parameter.

The construction of Boneh *et al.* [9] is based on the assumption where the algorithm receives as input a list of group elements

$$(g, h, g^x, \ldots, g^{x^q}, T) \in \mathbb{G}^{q+2} \times \mathbb{G}_T$$

and the algorithm has to distinguish $T = e(g, h)^{1/x}$ from random. The proof in [9] requires $q = \Theta(k)$. *Is it possible to construct VRFs with large input and full adaptive security from weaker q-type assumptions?*

Our contribution. We construct verifiable random functions with exponential-size input space, full adaptive security, and based on a q-type assumption with *very small* q. More precisely, $q = O(\log k)$ depends only *logarithmically* on the security parameter. The VRF construction essentially corresponds to the verifiable *unpredictable* function of Lysyanskaya [28], which inspired many very similar VRF constructions with either weaker security or based on stronger assumptions [25, 1, 16].

As a technical tool, we introduce the notion of *balanced* admissible hash functions (balanced AHFs), which are standard admissible hash functions [8] with an extra property (cf. the explanations below and in Section 4), and may be useful for applications beyond VRFs. We show how to construct balanced AHFs from codes with suitable minimal distance.

VRF construction. Let $\mathbb{G}, \mathbb{G}_T$ be groups with bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, and let $C : \{0,1\}^k \to \{0,1\}^n$ be a hash function. We construct a VRF with domain $\{0,1\}^k$ and range $\mathbb{G}_T$. The verification key of our VRF consists of C along with $2n+2$ random elements of $\mathbb{G}$

$$vk = \left(g, h, (g_{i,j})_{(i,j) \in [n] \times \{0,1\}}\right)$$

The secret key consists of the discrete logarithms $\alpha_{i,j}$ such that $g^{\alpha_{i,j}} = g_{i,j}$ for $(i,j) \in [n] \times \{0,1\}$.

The function is evaluated on input $X \in \{0,1\}^k$ by first computing

$$(C_1, \ldots, C_n) := C(X) \qquad \text{and} \qquad \alpha_X := \prod_{i=1}^{n} \alpha_{i,C_i}$$

and finally

$$V_{sk}(X) := e(g, h)^{\alpha_X}$$

A proof that $V_{sk}(X) = e(g, h)^{\alpha_X}$ consists of group elements $(\pi_1, \ldots, \pi_n)$ where

$$\pi_i := \pi_{i-1}^{\alpha_{i,C_i}}$$

for $i \in [n]$ and with $\pi_0 := g$. Correctness of proofs is verified with the bilinear map.

Similarity to Lysyanskaya's VUF. We note that our VRF construction is nearly identical to a VUF (resp. unique signature) construction of Lysyanskaya [28], but very different from the VRF construction of [28]. To explain this in more detail, recall that Lysyanskaya [28] followed a much more complex approach:

1. Construct a VUF based on a "computational" complexity assumption (in contrast to a "decisional" complexity assumption)
2. Turn this VUF into a VRF with single-bit output, by using a Goldreich-Levin hard-core predicate [22]. This step is not as simple as it may appear, because Micali *et al.* [29] show in their initial VRF paper that this only yields a VRF with *polynomially-bounded* input space (due to the fact that the randomness of the Goldreich-Levin hard-core predicate must be public to allow verifiability, which in turn leads to the problems discussed in [34]).
3. Turn this single-bit-VRF into a VRF with many-bit output (still with poly-bounded input space), by applying a generic construction from [29]. Note that this generic construction requires many evaluations of the underlying single-bit VRF.
4. In order to extend the VRF to a larger input space, apply another generic tree-based construction of [29]. Note that again this requires many evaluations of the underlying VRF.

In contrast, our direct VRF security proof of (essentially) the VUF-construction of Lysyanskaya yields directly a – in comparison much more simple and efficient – VRF with exponential-sized input space, adaptive security, and many-bit output. We rely on the new notion of *balanced* admissible hash functions in our security analysis.

Our security analysis and the need for balanced AHFs. We prove security under the $q\mathsf{DDH}$-assumption, which states that given

$$(g, h, g^x, \ldots, g^{x^q}, T)$$

it is hard to distinguish $T = e(g,h)^{x^{q+1}}$ from random.

A $q\mathsf{DDH}$-challenge is embedded into the view of the attacker by setting

$$g_{i,j} := g^{x+\alpha_{i,j}}$$

where $\alpha_{i,j} \xleftarrow{\$} \mathbb{Z}_{|\mathbb{G}|}$ is a random blinding term, but *only for* $O(\log k)$ *carefully selected* indices (i,j). This careful embedding essentially *partitions* the domain $\{0,1\}^k$ of the VRF into two sets $\mathcal{X}_0, \mathcal{X}_1$, such that

– For all values $X \in \mathcal{X}_1$ we have

$$V_{sk}(X) = e(g^{\prod_{i=0}^{q} \gamma_i x^i}, h) \quad \text{and} \quad \pi_j = g^{\prod_{i=0}^{q} \gamma_{j,i} x^i} \quad \forall 1 \leq j \leq n \qquad (1)$$

where the γ_i and $\gamma_{j,i}$ are integers in $\mathbb{Z}_{|\mathbb{G}|}$ which are known to the reduction. Note that the polynomials in the exponent of Equations (1) have degree at most q, thus $V_{sk}(X)$ and $\pi_1, \ldots, \pi_n$ can be computed, given the values $(g, g^x, \ldots, g^{x^q})$ from the $q\mathsf{DDH}$ challenge and the integers $\gamma_i, \gamma_{j,i}$.

– For all values $X^* \in \mathcal{X}_0$ the reduction is able to compute integers γ_i such that

$$Y^* = e(g^{\prod_{i=0}^{q} \gamma_i x^i}, h) \cdot T^{\gamma_{q+1}}$$

such that if $T = e(g,h)^{x^{q+1}}$ then it holds that $Y^* = V_{sk}(X^*)$. Note that if T is random, then so is Y^*.

Let $\{X^{(1)}, \ldots, X^{(Q)}\}$ denote the set of inputs on which the VRF-attacker queries the evaluation of the VRF with corresponding proof, and let X^* denote the element such that the attacker attempts to distinguish $V_{sk}(X^*)$ from random. The reduction will succeed, if it holds that $\{X^{(1)}, \ldots, X^{(Q)}\} \subseteq \mathcal{X}_1$ and $X^* \in \mathcal{X}_0$.

Instantiating C with an admissible hash function ensures that with non-negligible probability it simultaneously holds that $\{X^{(1)}, \ldots, X^{(Q)}\} \subseteq \mathcal{X}_1$ and $X^* \in \mathcal{X}_0$. However, unfortunately this is not yet sufficient to make the analysis of the success probability of our reduction go through, due to the incompatibility of partitioning proofs with "decisional" complexity assumptions, like $q\mathsf{DDH}$. Intuitively, the problem stems from the fact that two different sequences of queries made by the attacker may cause the simulator to abort with different probabilities. This issue was explained in great detail in [37, 5, 14].

Therefore we introduce the stronger notion of *balanced* AHFs. Essentially, a balanced AHF ensures that the upper bound γ_{max} and the lower bound γ_{min} on the probability in

$$\gamma_{\mathsf{max}} \geq \Pr[\{X^{(1)}, \ldots, X^{(Q)}\} \subseteq \mathcal{X}_1 \ \ \wedge \ \ X^* \in \mathcal{X}_0] \geq \gamma_{\mathsf{min}}$$

are reasonably close. This is a typical requirement for partitioning proofs based on decisional complexity assumptions, it occurs both in reductions with and without the "artificial abort" [37, 5]. This suggests that the notion of balanced AHFs may find applications beyond the construction of VRFs.

We stress that we achieve a reduction from a q-type assumption with $q = O(\log k)$ only if we instantiate the VRF construction with a *specific* AHF, essentially the code-based AHF of [19, 28]. The reason is that this is the only construction we are aware of which allows us to embed the given $q\mathsf{DDH}$-challenge into at most $O(\log k)$ carefully selected public-key elements $g_{i,j}$ in the way described above. We still have to prove that their AHF is also a *balanced* AHF.

More related work. VRFs were introduced by Micali, Rabin, and Vadhan [29], along with verifiable *unpredictable* functions (VUFs), a generic conversion from VUFs to VRFs based on Goldreich-Levin hard-core predicates [22], and a VUF-construction (with small input space) based on the RSA assumption. Specific, number-theoretic constructions of VRFs can be found in [29, 28, 16, 17, 1, 25, 9]. Note that most of these constructions either do not achieve full adaptive security for large input spaces, or are based on much stronger, *interactive* complexity assumptions. In particular, the VRF construction of Dodis [16] with outer error-correcting code is based on a q-type assumption (there called the $\mathsf{sf\text{-}DDH}$ *assumption of order* q) with $q = O(\log k)$, but this assumption is *interactive*. We wish to avoid interactive assumptions to prevent circular arguments, as explained by Naor [32].

Abdalla *et al.* gave generic constructions of VRFs from so-called *VRF-suitable identity-based KEMs* [1, 2]. While the conference version of this paper [1] considered only selective security, the full version [2] contains proofs that the construction from [1] achieves full security, under either under the complexity assumption from [25] with polynomially-bounded q, or, alternatively, under a q-type assumption with $q = O(k)$ when combined with an admissible hash function.

Brakerski *et al.* [11] introduced the relaxed notion of *weak* VRFs, along with simple and efficient constructions, and proofs that neither VRFs, nor weak VRFs can be constructed (in a black-box way) from one-way permutations. Fiore and Schröder [18] proved that verifiable random functions are not even implied (in a black-box sense) by trapdoor permutations. Several works introduced related primitives, like simulatable VRFs [12] and constrained VRFs [21].

At Eurocrypt 2006, Cheon [15] described an algorithm, which computes the discrete logarithm x on input $(g, g^x, \dots, g^{x^q})$. This algorithm is faster by a factor of $\sqrt{q}$ than generic algorithms for the standard discrete logarithm problem where only (g, g^x) is given. This shows that q-type assumptions are particularly problematic when q is large. The security loss must be compensated with larger group parameters, at the cost of efficiency. We stress that Cheon's algorithm is only much faster than generic algorithms for the standard discrete logarithm problem if q is very large (say, $q = 2^{40}$). However, Cheon's algorithm gives no apparent reason to criticise q-type assumptions for small q, like $q \leq 40$.

On avoiding q-Type assumptions altogether. Chase and Meiklejohn [13] present a conversion that allows to replace q-type assumption in certain applications with a *static* (that is, not q-type) subgroup hiding assumption, by leveraging the *dual-systems* techniques of Waters [36]. It is natural to ask whether these techniques can be used to construct verifiable random functions from static assumptions. Unfortunately, the conversion of [13] requires to add *randomization*. Thus, when applying it to known VRF constructions like [17], then this contradicts the unique provability requirement. Accordingly, Chase and Meiklejohn were able to prove that the VRF of Dodis and Yampolski [17] forms a secure pseudorandom function under a static assumption, but not that it is a secure verifiable random function.

We leave the construction of a verifiable random function with large input space and full adaptive security from a static assumption, like Decisional Diffie-Hellman, as an open problem.

2 Preliminaries

For a vector $K \in \{0,1\}^n$ we write K_i to denote the i-th component of K. If A is a finite set, then $a \stackrel{\$}{\leftarrow} A$ denotes the action of sampling a uniformly random from A. If A is a probabilistic algorithm, then we write $a \stackrel{\$}{\leftarrow} A$ to denote the action of computing a by running A with uniformly random coins. We define $[n] := \{1, \dots, n\} \subset \mathbb{N}$ as the set of all positive integers up to n.

2.1 Verifiable Unpredictable/Random Functions

Let $(\mathsf{Gen}, \mathsf{Eval}, \mathsf{Vfy})$ be the following algorithms.

- Algorithm $(vk, sk) \xleftarrow{\$} \mathsf{Gen}(1^k)$ takes as input a security parameter k and outputs a key pair (vk, sk). We say that sk is the *secret key* and vk is the *verification key*.
- Algorithm $(Y, \pi) \xleftarrow{\$} \mathsf{Eval}(sk, X)$ takes as input secret key sk and $X \in \{0,1\}^k$, and outputs a function value $Y \in \mathcal{Y}$, where $\mathcal{Y}$ is a finite set, and a proof π. We write $V_{sk}(X)$ to denote the function value Y computed by Eval on input (sk, X).
- Algorithm $\mathsf{Vfy}(vk, X, Y, \pi) \in \{0,1\}$ takes as input verification key vk, $X \in \{0,1\}^k$, $Y \in \mathcal{Y}$, and proof π, and outputs a bit.

$\mathbf{Initialize}:$	$\mathbf{Evaluate}(X):$	$\mathbf{Challenge}(X^*):$
$b \xleftarrow{\$} \{0,1\}$	$(Y, \pi) \xleftarrow{\$} \mathsf{Eval}(sk, X)$	$(Y_0, \pi) \xleftarrow{\$} \mathsf{Eval}(sk, X^*)$
$(vk, sk) \xleftarrow{\$} \mathsf{Gen}(1^k)$	**Return** (Y, π)	$Y_1 \xleftarrow{\$} \mathcal{Y}$
Return vk		**Return** Y_b

$\mathbf{Finalize}^{\mathsf{VUF}}(X^*, Y^*):$	$\mathbf{Finalize}^{\mathsf{VRF}}(b'):$
$(Y, \pi) \xleftarrow{\$} \mathsf{Eval}(sk, X^*)$	**If** $b' = b$ **then**
If $Y^* = Y$ **then**	**Return** 1
Return 1	**Else Return** 0
Else Return 0	

Fig. 1. Procedures defining the security experiments for VUFs and VRFs.

Definition 1. *We say that* $(\mathsf{Gen}, \mathsf{Eval}, \mathsf{Vfy})$ *is a* verifiable random function *(VRF) if all the following properties hold.*

Correctness. *Algorithms* Gen, Eval, Vfy *are polynomial-time algorithms, and for all* $(vk, sk) \xleftarrow{\$} \mathsf{Gen}(1^k)$ *and all* $X \in \{0,1\}^k$ *holds: if* $(Y, \pi) \xleftarrow{\$} \mathsf{Eval}(sk, X)$*, then* $\mathsf{Vfy}(vk, X, Y, \pi) = 1$.

Unique provability. *For all* $(vk, sk) \xleftarrow{\$} \mathsf{Gen}(1^k)$ *and all* $X \in \{0,1\}^k$*, there does not* exist *any tuple* (Y_0, π_0, Y_1, π_1) *such that* $Y_0 \neq Y_1$ *and* $\mathsf{Vfy}(vk, X, Y_0, \pi_0) = \mathsf{Vfy}(vk, X, Y_1, \pi_1) = 1$.

Pseudorandomness. *Consider an attacker* $\mathcal{A}$ *with access (via oracle queries) to the procedures defined in Figure 1. Let* $G^{\mathcal{A}}_{\mathsf{VRF}}$ *denote the game where* $\mathcal{A}$ *first queries* $\mathbf{Initialize}$*, then* $\mathbf{Challenge}$*, then* $\mathbf{Finalize}^{\mathsf{VRF}}$*, where the output of* $\mathbf{Finalize}^{\mathsf{VRF}}$ *is the output of the game. Moreover,* $\mathcal{A}$ *may arbitrarily issue* $\mathbf{Evaluate}$*-queries, but only after querying* $\mathbf{Initialize}$ *and before querying* $\mathbf{Finalize}^{\mathsf{VRF}}$*. We say that* $\mathcal{A}$ *is* legitimate*, if* $\mathcal{A}$ *never queries* $\mathbf{Evaluate}(X)$ *and* $\mathbf{Challenge}(X^*)$ *with* $X = X^*$ *throughout the game.*

We define the advantage of $\mathcal{A}$ in breaking the pseudorandomness as

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{VRF}}(k) := 2 \cdot \Pr[G_{\mathsf{VUF}}^{\mathcal{A}} = 1] - 1$$

Definition 2. *We say that* $(\mathsf{Gen}, \mathsf{Eval}, \mathsf{Vfy})$ *is a* verifiable unpredictable function *(VUF) if the* correctness *and* unique provability *properties from Definition 1 hold, and we have:*

Unpredictability. *Consider an attacker $\mathcal{A}$ with access (via oracle queries) to the procedures defined in Figure 1. Let $G_{\mathsf{VUF}}^{\mathcal{A}}$ denote the game where $\mathcal{A}$ first queries* $\mathbf{Initialize}$*, then an arbitrary number of* $\mathbf{Evaluate}$*-queries, then* $\mathbf{Finalize}^{\mathsf{VUF}}$*, and the output of* $\mathbf{Finalize}^{\mathsf{VUF}}$ *is the output of the game. We say that $\mathcal{A}$ is* legitimate*, if $\mathcal{A}$ never queries $\mathbf{Evaluate}(X)$ and $\mathbf{Challenge}(X^*)$ with $X = X^*$ throughout the game.*
We define the advantage of $\mathcal{A}$ in breaking the unpredictability as

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{VUF}}(k) := \Pr[G_{\mathsf{VUF}}^{\mathcal{A}} = 1]$$

2.2 q-Diffie-Hellman Assumptions

In the sequel let $\mathbb{G}, \mathbb{G}_T$ begroups of prime order, with bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$.

$\mathbf{Initialize}^{q\mathsf{CDH}}$:	$\mathbf{Finalize}^{q\mathsf{CDH}}(T)$:
$g, h \xleftarrow{\$} \mathbb{G}; x \xleftarrow{\$} \mathbb{Z}_{\lvert\mathbb{G}\rvert}$ **Return** $(g, g^x, \ldots, g^{x^q}, h)$	**If** $T = e(g^{x^{q+1}}, h)$ **then Return** 1 **Else Return** 0
$\mathbf{Initialize}^{q\mathsf{DDH}}$:	$\mathbf{Finalize}^{q\mathsf{DDH}}(b')$:
$g, h \xleftarrow{\$} \mathbb{G}; x \xleftarrow{\$} \mathbb{Z}_{\lvert\mathbb{G}\rvert}; b \xleftarrow{\$} \{0,1\}$ $T_0 := e(g,h)^{x^{q+1}}, T_1 \xleftarrow{\$} \mathbb{G}_T$ **Return** $(g, g^x, \ldots, g^{x^q}, h, T_b)$	**If** $b' = b$ **then Return** 1 **Else Return** 0

Fig. 2. Procedures defining the q-Diffie Hellman assumptions.

Definition 3. *Let $G_{\mathcal{B}}^{q\mathsf{DDH}}$ be the game with $\mathcal{B}$ and the procedures defined in Figure 2, where $\mathcal{B}$ calls* $\mathbf{Initialize}^{q\mathsf{DDH}}$*, then* $\mathbf{Finalize}^{q\mathsf{DDH}}$*, and the output of* $\mathbf{Finalize}^{q\mathsf{DDH}}$ *is the output of the game. We denote with*

$$\mathsf{Adv}_{\mathcal{B}}^{q\mathsf{DDH}}(k) := 2 \cdot \Pr\left[G_{\mathcal{B}}^{q\mathsf{DDH}} = 1\right] - 1$$

the advantage *of $\mathcal{A}$ in breaking the qDDH-assumption in $(\mathbb{G}, \mathbb{G}_T)$.*

Definition 4. *Let $G_{\mathcal{B}}^{q\mathsf{CDH}}$ be the game with $\mathcal{B}$ and the procedures defined in Figure 2, where $\mathcal{B}$ calls* $\mathbf{Initialize}^{q\mathsf{CDH}}$*, then* $\mathbf{Finalize}^{q\mathsf{CDH}}$*, and the output of* $\mathbf{Finalize}^{q\mathsf{CDH}}$ *is the output of the game. We denote with*

$$\mathsf{Adv}_{\mathcal{B}}^{q\mathsf{CDH}}(k) := \Pr\left[G_{\mathcal{B}}^{q\mathsf{CDH}} = 1\right]$$

the advantage *of $\mathcal{A}$ in breaking the qCDH-assumption in $(\mathbb{G}, \mathbb{G}_T)$.*

3 Main Construction

Let $\mathbb{G}, \mathbb{G}_T$ be groups of prime order with bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, such that each group element has a unique representation, and that group membership can be tested efficiently.

Let $\mathcal{VF} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Vfy})$ be the following construction.

Generation. Algorithm $\mathsf{Gen}(1^k)$ chooses an admissible hash function $C : \{0,1\}^k \to \{0,1\}^n$ and two random generators $g, h \xleftarrow{\$} \mathbb{G}$. Then it computes $g_{i,j} := g^{\alpha_{i,j}}$, where $\alpha_{i,j} \xleftarrow{\$} \mathbb{Z}_{|\mathbb{G}|}$ and for $(i,j) \in [n] \times \{0,1\}$. The keys are defined as

$$vk := \left(C, g, h, (g_{i,j})_{(i,j) \in [n] \times \{0,1\}}\right) \quad \text{and} \quad sk := (\alpha_{i,j})_{(i,j) \in [n] \times \{0,1\}}$$

Evaluation. On input $X \in \{0,1\}^k$, algorithm $\mathsf{Eval}(sk, X)$ first computes $C(X)$. For $i \in [n]$ let $C(X)_i$ denote the i-th bit of $C(X) \in \{0,1\}^n$. Then the algorithm determines the function value by computing $a_X := \prod_{i=1}^{n} \alpha_{i,C(X)_i}$ and setting

$$Y := e(g,h)^{a_X}.$$

The corresponding proof $\pi = (\pi_1, \ldots, \pi_n)$ is computed recursively by first defining $\pi_0 := g$ and then setting

$$\pi_i := \pi_{i-1}^{\alpha_{i,C(X)_i}} \quad \text{for all} \quad i \in [n]$$

The algorithm outputs (Y, π).

Verification. Algorithm $\mathsf{Vfy}(vk, X, Y, \pi)$ checks the consistency of π using the bilinear map. It first tests if X and π contain only valid group elements. Then it computes $C(X) = (C(X)_1, \ldots, C(X)_n) \in \{0,1\}^n$, defines $\pi_0 := g$, and outputs 1 if and only if all the following equations are satisfied.

$$\begin{aligned} e(\pi_i, g) &= e(\pi_{i-1}, g_{i,C(X)_i}) \quad \text{for all} \quad i \in [n] \\ Y &= e(\pi_n, h) \end{aligned}$$

It is straightforward to verify that the above construction is *correct* in the sense of Definitions 1 and 2. Furthermore, the *unique provability* follows from the group structure and the fact that even an unbounded attacker is not able to devise a proof π for a different group element. It remains to prove *pseudorandomness*.

4 Balanced Admissible Hash Functions

Standard admissible hash functions (AHFs) were introduced by Boneh and Boyen [8], a simplified definition was given by Freire et al. [19]. For our application, we will need AHFs with stronger properties, therefore we have to extend the notion of AHFs to *balanced* AHFs. The essential difference between balanced AHFs and the standard definition (e.g. [20, Definition 3]) is that previous works required only a reasonable *lower* bound on the probability in Equation (3) below. In contrast, the security analysis of our VRF construction will essentially require reasonable *upper and lower bounds*, and that these bounds are sufficiently close.

Definition 5. *Let $k \in \mathbb{N}$ and $n = n(k)$ be a polynomial, and let $C : \{0,1\}^k \to \{0,1\}^{n(k)}$ be an efficiently computable function. Let $F_K : \{0,1\}^k \to \{0,1\}$ be defined as*

$$F_K(X) := \begin{cases} 0, \text{ if } \forall i : C(X)_i = K_i \quad \vee \quad K_i = \perp \\ 1, \text{ else.} \end{cases} \tag{2}$$

We say that C is a balanced admissible hash function *(balanced AHF), if there exists an efficient algorithm $\mathsf{AdmSmp}(1^k, Q, \delta)$, which takes as input (Q, δ) where $Q = Q(k) \in \mathbb{N}$ is polynomially bounded and $\delta = \delta(k) \in (0,1]$ is non-negligible, and outputs $K \in \{0,1,\perp\}^n$ such that for all $X^{(1)}, \ldots, X^{(Q)}, X^* \in \{0,1\}^k$ with $X^* \notin \{X^{(1)}, \ldots, X^{(Q)}\}$ holds that*

$$\gamma_{\mathsf{max}}(k) \geq \Pr[F_K(X^{(1)}) = \cdots = F_K(X^{(Q)}) = 1 \ \wedge \ F_K(X^*) = 0] \geq \gamma_{\mathsf{min}}(k) \tag{3}$$

where $\gamma_{\mathsf{max}}(k)$ and $\gamma_{\mathsf{min}}(k)$ satisfy that the function $\tau(k)$ defined as

$$\tau(k) := 2 \cdot \gamma_{\mathsf{min}}(k) \cdot \delta(k) - \gamma_{\mathsf{max}}(k) + \gamma_{\mathsf{min}}(k) \tag{4}$$

is non-negligible. The probability is taken over the choice of K.

Remark 1. The definition of τ essentially condenses two requirements, namely (1) that γ_{min} is non-negligible, and (2) that the difference $\gamma_{\mathsf{max}} - \gamma_{\mathsf{min}}$ is "reasonably" small, where "reasonably" depends on γ_{min} and δ. The definition of function τ may appear very specific, however, such a term appears typically in security analyses that follow the approach of Bellare and Ristenpart [5]. Therefore we think this is exactly what is needed for typical applications of balanced AHFs. See Lemma 1, for instance.

Instantiating balanced admissible hash functions. Efficient *standard* admissible hash functions are known to exist [28, 8, 19]. For instance, there is a simple construction from codes with suitable minimal distance [28, 19]. In this section we will show that such codes also yield a *balanced* AHF. In contrast to [28, 19], we have to show both upper and lower bounds, and choose certain parameters more carefully to ensure that (4) is a non-negligible function.

Theorem 1. *Let $(C_k)_{k \in \mathbb{N}}$ with $C_k : \{0,1\}^k \to \{0,1\}^n$ be a family of codes with minimal distance nc for a constant c. Then $(C_k)_{k \in \mathbb{N}}$ is a family of balanced admissible hash functions. Moreover, $\mathsf{AdmSmp}(1^k, Q, \delta)$ outputs $K \in \{0,1,\perp\}^n$ with* exactly *$d = \left\lfloor \frac{\ln(2Q+Q/\delta)}{-\ln((1-c))} \right\rfloor$ components not equal to $\perp$.*

Proof. Consider the algorithm AdmSmp which sets

$$d := \left\lfloor \frac{\ln(2Q + Q/\delta)}{-\ln((1-c))} \right\rfloor$$

and chooses K uniformly random from $(\{0,1\} \cup \{\perp\})^n$ with exactly d components not equal to $\perp$.[2]

Fix $X^{(1)}, \ldots, X^{(Q)}, X^* \in \{0,1\}^k$ with $X^* \notin \{X^{(1)}, \ldots, X^{(Q)}\}$ for the analysis of this algorithm.

[2] Note that this algorithm is identical to the algorithm from [20, Theorem 2], except that we have chosen d slightly differently.

Upper bound. Note that we have $\Pr[F_K(X^*) = 0] = 2^{-d}$, and thus

$$\begin{aligned}\gamma_{\mathsf{max}} := 2^{-d} &= \Pr[F_K(X^*) = 0] \\ &\geq \Pr[F_K(X^*) = 0] \cdot \Pr[F_K(X^{(1)}) = \cdots = F_K(X^{(Q)}) = 1 \mid F_K(X^*)] \\ &= \Pr[F_K(X^*) = 0 \wedge F_K(X^{(1)}) = \cdots = F_K(X^{(Q)}) = 1].\end{aligned}$$

Lower bound. We first observe that for any two strings $X, X^* \in \{0,1\}^k$ with $X \neq X^*$ holds that

$$\Pr[F_K(X) = 0 \mid F_K(X^*) = 0] \leq (1-c)^d.$$

To see this, consider an experiment where two code words $C(X)$ and $C(X^*)$ are given, with $X, X^* \in \{0,1\}^k$ and $X \neq X^*$, and we sample d pairwise distinct positions $i_1, \ldots, i_d \xleftarrow{\$} [n]$. Since $C(X)$ and $C(X^*)$ differ in at least nc positions, the probability that $C(X)_{i_1} = C(X^*)_{i_1}$ is at most $(n - nc)/n = 1 - c$. The probability that $C(X)_{i_j} = C(X^*)_{i_j}$ for all $j \in [d]$ is thus at most $(1-c)^d$.

A union bound yields that

$$\Pr[F_K(X^{(1)}) = 0 \vee \cdots \vee F_K(X^{(Q)}) = 0 \mid F_K(X^*) = 0] \leq Q(1-c)^d$$

which implies

$$\Pr[F_K(X^{(1)}) = 1 \wedge \cdots \wedge F_K(X^{(Q)}) = 1 \mid F_K(X^*) = 0] \geq 1 - Q(1-c)^d$$

This yields the lower bound

$$\begin{aligned}\gamma_{\mathsf{min}} :=&(1 - Q(1-c)^d) \cdot 2^{-d} \\ \leq& \Pr[F_K(X^{(1)}) = 1 \wedge \cdots \wedge F_K(X^{(Q)}) = 1 \mid F_K(X^*) = 0] \cdot \Pr[F_K(X^*) = 0] \\ =& \Pr[F_K(X^{(1)}) = \cdots = F_K(X^{(Q)}) = 1 \wedge F_K(X^*) = 0]\end{aligned}$$

Balancedness of bounds. Finally, it remains to show that for polynomial Q and non-negligible δ the function τ from (4) is non-negligible. We first compute (omitting the parameter k from functions to simplify notation):

$$\begin{aligned}\tau :=&2 \cdot \delta \cdot \gamma_{\mathsf{min}} - \gamma_{\mathsf{max}} + \gamma_{\mathsf{min}} \\ =&2 \cdot \delta \cdot (1 - Q(1-c)^d) \cdot 2^{-d} - 2^{-d} + (1 - Q(1-c)^d) \cdot 2^{-d} \\ =&2^{-d} \cdot \left(2\delta - (2\delta + 1) \cdot Q(1-c)^d\right)\end{aligned}$$

Now we will show that if d is chosen as above, then both 2^{-d} and $2\delta - (2\delta+1) \cdot Q(1-c)^d$ are non-negligible. Thus, their product is non-negligible as well.

We have

$$2^{-d} = 2^{-\left\lfloor \frac{\ln(2Q + Q/\delta)}{-\ln((1-c))} \right\rfloor} \geq 2^{\frac{\ln(2Q+Q/\delta)}{\ln((1-c))}}$$

and

$$\begin{aligned}2\delta - (2\delta + 1) \cdot Q(1-c)^d &= 2\delta - (2\delta + 1) \cdot Q(1-c)^{\left\lfloor \frac{\ln(2Q+Q/\delta)}{-\ln((1-c))} \right\rfloor} \\ &\geq 2\delta - (2\delta + 1) \cdot Q(2Q + Q/\delta)^{-1} \\ &= 2\delta - (2\delta Q + Q)(2Q + Q/\delta)^{-1} \\ &= 2\delta - \delta(2\delta Q + Q)(2\delta Q + Q)^{-1} = \delta\end{aligned}$$

which both are non-negligible since c is a constant, $Q \in \mathbb{N}$, and $\delta \in (0,1]$ is non-negligible.

5 $\mathcal{VF}$ is a Verifiable Random Function

Theorem 2. *If $\mathcal{VF}$ is instantiated with the balanced admissible hash function from Theorem 1, then for any legitimate attacker $\mathcal{A}$ that breaks the* pseudorandomness *of $\mathcal{VF}$ in time $t_{\mathcal{A}}$ with advantage $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{VRF}}$ by making at most Q* Eval*-queries, there exists an algorithm $\mathcal{B}$ that breaks the q-DDH assumption with $q = \left\lfloor \frac{\ln(2Q+Q/\delta)}{-\ln((1-c))} \right\rfloor - 1$ in time $t_{\mathcal{B}} \approx t_{\mathcal{A}}$ and with advantage*

$$\mathsf{Adv}_{\mathcal{B}}^{q\mathsf{DDH}}(k) \geq \tau(k)$$

where $2 \cdot \delta$ is a non-negligible lower bound on $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{VRF}}(k)$, and $\tau(k)$ is a non-negligible function.

$\mathbf{Initialize}:$
$\mathtt{bad} := 0$
$K \xleftarrow{\$} \mathsf{AdmSmp}(1^k, Q, \delta)$
For $(i,j) \in [n] \times \{0,1\}$ **do**
  $\alpha_{i,j} \xleftarrow{\$} \mathbb{Z}_{|\mathbb{G}|}$
  If $K_i = j$ **then** $g_{i,j} := g^{x+\alpha_{i,j}}$
  Else $g_{i,j} := g^{\alpha_{i,j}}$
$vk := \big(C, g, h, (g_{i,j})_{(i,j)}\big)$
Return vk

$\mathbf{Evaluate}(X):$
$(Y, \pi) := \bot$
If $F_K(X) \neq 1$ **then**
  $\mathtt{bad} := 1;$
Else
  $Y := e(g^{P_{K,n,X}(x)}, h)$
  For $j \in [n]$ **do**
    $\pi_j := g^{P_{K,j,X}(x)}$
  $\pi := (\pi_1, \ldots, \pi_n)$
Return (Y, π)

$\mathbf{Challenge}(X^*):$
$Y^* := \bot$
If $F_K(X) = 1$ **then**
  $\mathtt{bad} := 1$
Else
  Compute $\gamma_0, \ldots, \gamma_{q+1}$ s.t.
    $P_{K,n,X^*}(x) = \sum_{i=0}^{q+1} \gamma_i x^i$
  $Y^* := T^{\gamma_{q+1}} \cdot \prod_{i=1}^{q} e((g^{x^i})^{\gamma_i}, h)$
Return Y^*

$\mathbf{Finalize}^{\mathsf{VRF}}(b'):$
If $\mathtt{bad} = 1$ **then** $c' \xleftarrow{\$} \{0,1\}$
Else $c' := b'$
Return c'

Fig. 3. Procedures for the simulation of the VRF pseudorandomness experiment by $\mathcal{B}$.

Proof. Algorithm $\mathcal{B}$ receives as input $(g, g^x, \ldots, g^{x^q}, h, T)$ and runs algorithm $\mathcal{A}$ as a subroutine. Whenever $\mathcal{A}$ queries $\mathbf{Initialize}$, $\mathbf{Evaluate}$, $\mathbf{Challenge}$, or $\mathbf{Finalize}$, $\mathcal{B}$ executes the corresponding procedure from Figure 3. Let us give some remarks on these procedures.

Initialization. The values (g, h, g^x) in $\mathbf{Initialize}$ are from the qDDH-challenge. Recall that $2 \cdot \delta$ is a non-negligible lower bound on $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{VRF}}(k)$, and Q is the upper bound on the number of $\mathbf{Evaluate}$-queries.

Note that $\mathcal{B}$ computes the $g_{i,j}$-values exactly as in the original Gen-algorithm, by choosing $\alpha_{i,j} \xleftarrow{\$} \mathbb{Z}_{|\mathbb{G}|}$ and setting $g_{i,j} := g^{\alpha_{i,j}}$, but with the exception that

$$g_{i,K_i} := g^{x+\alpha_{i,K_i}}.$$

for all $(i, j) \in [n] \times \{0, 1\}$ with $K_i = j$. Due to our choice of an admissible hash function according to Theorem 1, there are *exactly* $q + 1$ components K_i of K which are not equal to $\perp$.

Finally, note that all g_{i,K_i}-values are distributed correctly, and that this set-up defines the secret key *implicitly* as $sk := (\log_g g_{i,j})_{(i,j)\in[n]\times\{0,1\}}$. Thus, the function $V_{sk}(X)$ is well-defined for all X (but $\mathcal{B}$ will not be able to evaluate V_{sk} on all inputs X, as explained below).

Helping definitions. In order to explain how $\mathcal{B}$ responds to $\mathbf{Evaluate}$ and $\mathbf{Challenge}$ queries made by $\mathcal{A}$, let us define two sets $I_{K,w,X}$ and $J_{K,w,X}$, which depend on an AHF key K, a VRF input $X \in \{0, 1\}^k$, and integer $w \in \mathbb{N}$ with $1 \leq w \leq n$, as

$$I_{K,w,X} := \{i \in [w] : K_i = C(X)_i\} \qquad \text{and} \qquad J_{K,w,X} := [w] \setminus I_{K,w,X}$$

Note that $I_{K,w,X}$ denotes the set of all indices $i \in [w] \subseteq [n]$ such that $K_i = C(X)_i$, and $J_{K,w,X}$ denotes the set of all indices in $[w]$ which are not contained in $I_{K,w,X}$. Based on these sets, we define polynomials $P_{K,w,X}(x)$

$$P_{K,w,X}(x) = \prod_{i \in I_{K,w,X}} (x + \alpha_{i,K_i}) \cdot \prod_{i \in J_{K,w,X}} \alpha_{i,K_i} \in \mathbb{Z}_{|\mathbb{G}|}[x]$$

Now we can make the following observations:

1. For all X with $F_K(X) = 1$, the set $I_{K,w,X}$ contains at most q elements, and thus the polynomial $P_{K,w,X}(x)$ has degree at most q.
   This implies that if $F_K(X) = 1$, then $\mathcal{B}$ can efficiently compute $g^{P_{K,w,X}(x)}$ for all $w \in [n]$. To this end, $\mathcal{B}$ first computes the coefficients $\gamma_0, \ldots, \gamma_q$ of the polynomial $P_{K,w,X}(x) = \sum_{i=0}^{q} \gamma_i x^i$ with degree at most q, and then
   $$g^{P_{K,w,X}(x)} := g^{\sum_{i=0}^{q} \gamma_i x^i} = \prod_{i=0}^{q} (g^{x^i})^{\gamma_i}$$
   using the terms $(g, g^x, \ldots, g^{x^q})$ from the q-DDH challenge.
2. If $F_K(X) = 0$, then $P_{K,n,X}(x)$ has degree $q + 1$. We do not know how $\mathcal{B}$ can efficiently compute $g^{P_{K,n,X}(x)}$ in this case.

Responding to $\mathbf{Evaluate}$*-queries.* If $F_K(X) = 1$, then procedure $\mathbf{Evaluate}$ computes the group elements $g^{P_{K,w,X}(x)}$ as explained above. Note that in this case the response to the $\mathbf{Evaluate}(X)$-query of $\mathcal{A}$ is correct. However, if $F_K(X) = 0$, then the response of $\mathcal{B}$ is incorrect.

Responding to the **Challenge**-*query.* If $F_K(X^*) = 0$, then procedure **Challenge** computes

$$Y^* := T^{\gamma_{q+1}} \cdot \prod_{i=1}^{q} e((g^{x^i})^{\gamma_i}, h) = T^{\gamma_{q+1}} \cdot e(g^{\sum_{i=1}^{q} \gamma_i x^i}, h)$$

where $\gamma_0, \ldots, \gamma_{q+1}$ are the coefficients of the degree-$(q+1)$-polynomial $P_{K,n,X^*}(x) = \sum_{i=0}^{q+1} \gamma_i x^i$. Note that if $T = e(g,h)^{x^{q+1}}$, then it holds that $Y^* = V_{sk}(X^*)$. Moreover, if T is uniformly random, then so is Y^*.

Analysis of $\mathcal{B}$'s running time. The running time $t_{\mathcal{B}}$ of $\mathcal{B}$ consists essentially of the running time $t_{\mathcal{A}}$ of $\mathcal{A}$ plus a minor number of additional operations, thus we have $t_{\mathcal{B}} \approx t_{\mathcal{A}}$.

Analysis of $\mathcal{B}$'s success probability. The simulation of the challenger by $\mathcal{B}$ is perfect, unless $\texttt{bad} := 1$ is set. This happens only if $\mathcal{A}$ queries $\mathbf{Evaluate}(X)$ with $F_K(X) \neq 1$, or $\mathbf{Challenge}(X^*)$ with $F_K(X^*) = 1$. Since the AHF key K is information-theoretically hidden in vk, the terms γ_{max} and γ_{min} from Equation (3) are upper and lower bounds on the probability that $\texttt{bad} := 1$ is never set throughout the experiment.

Lemma 1.

$$\mathsf{Adv}_{\mathcal{B}}^{q\mathsf{CDH}}(k) \geq 2 \cdot \gamma_{\mathsf{min}} \cdot \delta - \gamma_{\mathsf{max}} + \gamma_{\mathsf{min}}$$

The proof of Lemma 1 follows the approach of Bellare and Ristenpart [5] very closely, therefore it is deferred to Appendix A. This approach allows us to provide an analysis without the "artificial abort" of Waters [37]. The latter has also been used to analyze the VRF of Hohenberger and Waters [24], but leads to a less tight reduction.

Remark 2. Note that the lower bound on $\mathsf{Adv}_{\mathcal{B}}^{q\mathsf{CDH}}(k)$ in Lemma 1 is only useful, if δ and γ_{min} are non-negligible and γ_{max} and γ_{min} are sufficiently close. This is where we need the balancedness of admissible hash function C.

Observe that since we instantiate C with a balanced AHF and δ is a non-negligible lower bound on $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{VRF}}(k)/2$, the function

$$\tau(k) := 2 \cdot \gamma_{\mathsf{min}} \cdot \delta - \gamma_{\mathsf{max}} + \gamma_{\mathsf{min}}$$

is non-negligible. This concludes the proof of Theorem 2.

6 $\mathcal{VF}$ is a Verifiable Unpredictable Function

In this section we prove that construction $\mathcal{VF}$ also is a secure VUF. Note that this construction is essentially identical to the VUF of Lysyanskaya [28], only the proof is based on a different complexity assumption.

The main purpose of this section is to show that for the VUF-security proof of $\mathcal{VF}$ an even weaker (but still $O(\log k)$) q-type assumption is sufficient. We can base security on a $q\mathsf{CDH}$ assumption that is weaker in two ways. First, it is the *computational* version of the $q\mathsf{DDH}$ assumption. Second, we need only $q = \lfloor (\ln 2Q)/c \rfloor - 1$. Thus, in contrast to the VRF-security proof, q is independent of the advantage of the attacker.

6.1 Admissible Hash Functions

In order to prove that $\mathcal{VF}$ is a VUF, it will suffice to instantiate $\mathcal{VF}$ with a standard (that is, not necessarily balanced) admissible hash function C. We recall the standard definition of admissible hash functions (AHFs) from Freire et al. [19].

Definition 6 ([19]). *Let $k \in \mathbb{N}$ and $n = n(k)$ be a polynomial, and let $C : \{0,1\}^k \to \{0,1\}^{n(k)}$ be an efficiently computable function. Let $F_K : \{0,1\}^k \to \{0,1\}$ be defined as in Equation (2). We say that C is an* admissible hash function *(AHF), if there exists an efficient algorithm $\mathsf{AdmSmp}(1^k, Q)$, which takes as input polynomial $Q = Q(k) \in \mathbb{N}$, and computes $K \in (\{0,1\} \cup \{\perp\})^n$ such that for all $X^{(1)}, \ldots, X^{(Q)}, X^* \in \{0,1\}^k$ with $X^* \notin \{X^{(1)}, \ldots, X^{(Q)}\}$ holds that*

$$\Pr[F_K(X^{(1)}) = \cdots = F_K(X^{(Q)}) = 1 \;\wedge\; F_K(X^*) = 0] \geq \gamma_{\mathsf{min}}(k) \tag{5}$$

such that $\gamma_{\mathsf{min}}(k)$ non-negligible. The probability is taken over the choice of K.

Instantiating Admissible Hash Functions. A simple and efficient construction of AHFs can be found in [19] (based on [28]), we capture their existence in the following lemma.

Lemma 2 ([28, 19]). *Let S be a set and $(C_k)_{k\in\mathbb{N}}$ with $C_k : \{0,1\}^k \to S^n$ be a family of codes, with minimal distance nc for a constant c and such that $|S|$ is bounded by a polynomial in k. Then $(C_k)_{k\in\mathbb{N}}$ is an admissible hash function, where $\mathsf{AdmSmp}(Q)$ outputs $K \in S \cup \{\perp\}^n$ with* exactly *$d := \lfloor (\ln 2Q)/c \rfloor$ components not equal to $\perp$ and $\gamma_{\mathsf{min}} \geq (1 - Q(1-c)^d) \cdot 2^{-d}$.*

Remark 3. Note that even though the last two statements of the above theorem were not made explicit in previous works, they are implicitly contained in the proof of [20, Theorem 2].

6.2 Security Analysis

Theorem 3. *If $\mathcal{VF}$ is instantiated with the admissible hash function from Lemma 2, then for any legitimate attacker $\mathcal{A}$ that breaks the* unpredictability *of $\mathcal{VF}$ in time $t_{\mathcal{A}}$ with advantage $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{VUF}}$ by making at most Q Eval-queries, there exists an algorithm $\mathcal{B}$ that breaks the $q\mathsf{CDH}$ assumption with $q = \lfloor (\ln 2Q)/c \rfloor - 1$ in time $t_{\mathcal{B}} \approx t_{\mathcal{A}}$ and with advantage*

$$\mathsf{Adv}_{\mathcal{B}}^{q\mathsf{CDH}}(k) \geq \mathsf{Adv}_{\mathcal{A}}^{\mathsf{VUF}}(k) \cdot (1 - Q(1-c)^d) \cdot 2^{-d}$$

where $d := \lfloor (\ln 2Q)/c \rfloor = q + 1$.

The proof of this theorem is nearly identical to the proof of Theorem 2, but the analysis of the success probability of $\mathcal{B}$ is much simpler, because we consider *unpredictability* instead of *pseudorandomness*. Therefore we only sketch the proof.

Proof. Algorithm $\mathcal{B}$ receives as input $(g, g^x, \ldots, g^{x^q}, h, T)$ and runs algorithm $\mathcal{A}$ as a subroutine. Whenever $\mathcal{A}$ issues a query ($\mathbf{Initialize}$, $\mathbf{Evaluate}$, $\mathbf{Finalize}$), then $\mathcal{B}$ executes the corresponding procedure from Figure 4.

$\mathbf{Initialize}(X):$
$\mathtt{bad} := 0$
$K \xleftarrow{\$} \mathsf{AdmSmp}(1^k, Q, \delta)$
For $(i,j) \in [n] \times \{0,1\}$ **do**
  $\alpha_{i,j} \xleftarrow{\$} \mathbb{Z}_{|\mathbb{G}|}$
  If $K_i = j$ **then** $h_{i,j} := g^{x+\alpha_{i,j}}$
  Else $h_{i,j} := g^{\alpha_{i,j}}$
$vk := \big(C, g, h, (h_{i,j})_{(i,j)}\big)$
Return vk

$\mathbf{Evaluate}(X):$
$(Y,\pi) := \perp$
If $F_K(X) \neq 1$ **then**
  $\mathtt{bad} := 1;$
Else
  $Y := e(g^{P_{K,n,X}(x)}, h)$
  For $j \in [n]$ **do**
    $\pi_j := g^{P_{K,j,X}(x)}$
  $\pi := (\pi_1, \ldots, \pi_n)$
Return (Y,π)

$\mathbf{Finalize}^{\mathsf{VUF}}(X^*, Y^*):$
If $F_K(X^*) = 0$ **then**
  $\mathtt{bad} := 1$
If $\mathtt{bad} = 1$ **then Return** $\perp$
Compute $\gamma_0, \ldots, \gamma_{q+1}$
  s.t. $P_{K,n,X^*}(x) = \sum_{i=0}^{q+1} \gamma_i x^i$
$T := \Big(Y^* / e(g^{\sum_{i=1}^{q} \gamma_i x^i}, h)\Big)^{1/\gamma_{q+1}}$
Return T

Fig. 4. Procedures for the simulation of the VUF unpredictability experiment by $\mathcal{B}$.

The running time $t_{\mathcal{B}}$ of $\mathcal{B}$ consists essentially of the running time $t_{\mathcal{A}}$ of $\mathcal{A}$ plus a minor number of additional operations, thus we have $t_{\mathcal{B}} \approx t_{\mathcal{A}}$. Note that $\mathcal{B}$ simulates the original VUF security experiment perfectly, if $\mathtt{bad} = 0$ throughout the game. Note also that

$$Y^* = e(g,h)^{\sum_{i=0}^{q+1} \gamma_i x^i} \implies T = e(g,h)^{x^{q+1}}$$

The choice of K is information-theoretically hidden in vk. Thus,

$$\begin{aligned} \mathsf{Adv}_{\mathcal{B}}^{q\mathsf{CDH}}(k) &\geq \mathsf{Adv}_{\mathcal{A}}^{\mathsf{VUF}}(k) \cdot \Pr[\mathtt{bad} = 0] \\ &\geq \mathsf{Adv}_{\mathcal{A}}^{\mathsf{VUF}}(k) \cdot \gamma_{\mathsf{min}}(k) = \mathsf{Adv}_{\mathcal{A}}^{\mathsf{VUF}}(k) \cdot (1 - Q(1-c)^d) \cdot 2^{-d} \end{aligned}$$

Acknowledgements. We thank the anonymous reviewers of TCC 2015 for their helpful comments.

References

1. Michel Abdalla, Dario Catalano, and Dario Fiore. Verifiable random functions from identity-based key encapsulation. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 554–571, Cologne, Germany, April 26–30, 2009. Springer, Berlin, Germany.
2. Michel Abdalla, Dario Catalano, and Dario Fiore. Verifiable random functions: Relations to identity-based key encapsulation and new constructions. *Journal of Cryptology*, 27(3):544–593, July 2014.

3. Man Ho Au, Willy Susilo, and Yi Mu. Practical compact e-cash. In Josef Pieprzyk, Hossein Ghodosi, and Ed Dawson, editors, *ACISP 07*, volume 4586 of *LNCS*, pages 431–445, Townsville, Australia, July 2–4, 2007. Springer, Berlin, Germany.
4. Mira Belenkiy, Melissa Chase, Markulf Kohlweiss, and Anna Lysyanskaya. Compact e-cash and simulatable VRFs revisited. In Hovav Shacham and Brent Waters, editors, *PAIRING 2009*, volume 5671 of *LNCS*, pages 114–131, Palo Alto, CA, USA, August 12–14, 2009. Springer, Berlin, Germany.
5. Mihir Bellare and Thomas Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for Waters' IBE scheme. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 407–424, Cologne, Germany, April 26–30, 2009. Springer, Berlin, Germany.
6. Mihir Bellare and Thomas Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for Waters' IBE scheme. Cryptology ePrint Archive, Report 2009/084, 2009. `http://eprint.iacr.org/`.
7. Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Berlin, Germany.
8. Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 443–459, Santa Barbara, CA, USA, August 15–19, 2004. Springer, Berlin, Germany.
9. Dan Boneh, Hart William Montgomery, and Ananth Raghunathan. Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 10*, pages 131–140, Chicago, Illinois, USA, October 4–8, 2010. ACM Press.
10. Dan Boneh, Emily Shen, and Brent Waters. Strongly unforgeable signatures based on computational Diffie-Hellman. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *PKC 2006*, volume 3958 of *LNCS*, pages 229–240, New York, NY, USA, April 24–26, 2006. Springer, Berlin, Germany.
11. Zvika Brakerski, Shafi Goldwasser, Guy N. Rothblum, and Vinod Vaikuntanathan. Weak verifiable random functions. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 558–576. Springer, Berlin, Germany, March 15–17, 2009.
12. Melissa Chase and Anna Lysyanskaya. Simulatable VRFs with applications to multi-theorem NIZK. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 303–322, Santa Barbara, CA, USA, August 19–23, 2007. Springer, Berlin, Germany.
13. Melissa Chase and Sarah Meiklejohn. Déjà Q: Using dual systems to revisit q-type assumptions. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 622–639, Copenhagen, Denmark, May 11–15, 2014. Springer, Berlin, Germany.
14. Sanjit Chatterjee and Palash Sarkar. HIBE with short public parameters without random oracle. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 145–160, Shanghai, China, December 3–7, 2006. Springer, Berlin, Germany.
15. Jung Hee Cheon. Security analysis of the strong Diffie-Hellman problem. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 1–11, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Berlin, Germany.
16. Yevgeniy Dodis. Efficient construction of (distributed) verifiable random functions. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 1–17, Miami, USA, January 6–8, 2003. Springer, Berlin, Germany.
17. Yevgeniy Dodis and Aleksandr Yampolskiy. A verifiable random function with short proofs and keys. In Serge Vaudenay, editor, *PKC 2005*, volume 3386 of *LNCS*, pages 416–431, Les Diablerets, Switzerland, January 23–26, 2005. Springer, Berlin, Germany.

18. Dario Fiore and Dominique Schröder. Uniqueness is a different story: Impossibility of verifiable random functions from trapdoor permutations. In Ronald Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 636–653, Taormina, Sicily, Italy, March 19–21, 2012. Springer, Berlin, Germany.
19. Eduarda S. V. Freire, Dennis Hofheinz, Kenneth G. Paterson, and Christoph Striecks. Programmable hash functions in the multilinear setting. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 513–530, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Berlin, Germany.
20. Eduarda S.V. Freire, Dennis Hofheinz, Kenneth G. Paterson, and Christoph Striecks. Programmable hash functions in the multilinear setting. Cryptology ePrint Archive, Report 2013/354, 2013. `http://eprint.iacr.org/`.
21. Georg Fuchsbauer. Constrained verifiable random functions. In Michel Abdalla and Roberto De Prisco, editors, *Security and Cryptography for Networks - 9th International Conference, SCN 2014, Amalfi, Italy, September 3-5, 2014. Proceedings*, volume 8642 of *Lecture Notes in Computer Science*, pages 95–114. Springer, 2014.
22. Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *21st ACM STOC*, pages 25–32, Seattle, Washington, USA, May 15–17, 1989. ACM Press.
23. Shafi Goldwasser and Rafail Ostrovsky. Invariant signatures and non-interactive zero-knowledge proofs are equivalent (extended abstract). In Ernest F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 228–245, Santa Barbara, CA, USA, August 16–20, 1992. Springer, Berlin, Germany.
24. Susan Hohenberger and Brent Waters. Realizing hash-and-sign signatures under standard assumptions. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 333–350, Cologne, Germany, April 26–30, 2009. Springer, Berlin, Germany.
25. Susan Hohenberger and Brent Waters. Constructing verifiable random functions with large input spaces. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 656–672, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany.
26. Stanislaw Jarecki and Vitaly Shmatikov. Handcuffing big brother: an abuse-resilient transaction escrow scheme. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 590–608, Interlaken, Switzerland, May 2–6, 2004. Springer, Berlin, Germany.
27. Moses Liskov. Updatable zero-knowledge databases. In Bimal K. Roy, editor, *ASIACRYPT 2005*, volume 3788 of *LNCS*, pages 174–198, Chennai, India, December 4–8, 2005. Springer, Berlin, Germany.
28. Anna Lysyanskaya. Unique signatures and verifiable random functions from the DH-DDH separation. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 597–612, Santa Barbara, CA, USA, August 18–22, 2002. Springer, Berlin, Germany.
29. Silvio Micali, Michael O. Rabin, and Salil P. Vadhan. Verifiable random functions. In *40th FOCS*, pages 120–130, New York, New York, USA, October 17–19, 1999. IEEE Computer Society Press.
30. Silvio Micali and Leonid Reyzin. Soundness in the public-key model. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 542–565, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Berlin, Germany.
31. Silvio Micali and Ronald L. Rivest. Micropayments revisited. In Bart Preneel, editor, *CT-RSA 2002*, volume 2271 of *LNCS*, pages 149–163, San Jose, CA, USA, February 18–22, 2002. Springer, Berlin, Germany.
32. Moni Naor. On cryptographic assumptions and challenges (invited talk). In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 96–109, Santa Barbara, CA, USA, August 17–21, 2003. Springer, Berlin, Germany.

33. Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th FOCS*, pages 458–467, Miami Beach, Florida, October 19–22, 1997. IEEE Computer Society Press.
34. Moni Naor and Omer Reingold. From unpredictability to indistinguishability: A simple construction of pseudo-random functions from MACs (extended abstract). In Hugo Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 267–282, Santa Barbara, CA, USA, August 23–27, 1998. Springer, Berlin, Germany.
35. Ron Steinfeld, Josef Pieprzyk, and Huaxiong Wang. How to strengthen any weakly unforgeable signature into a strongly unforgeable signature. In Masayuki Abe, editor, *CT-RSA 2007*, volume 4377 of *LNCS*, pages 357–371, San Francisco, CA, USA, February 5–9, 2007. Springer, Berlin, Germany.
36. Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 619–636, Santa Barbara, CA, USA, August 16–20, 2009. Springer, Berlin, Germany.
37. Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 114–127, Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany.

A Proof of Lemma 1

Let $G^{q\mathsf{DDH}}_{\mathcal{B}(\mathcal{A})}$ denote the $q\mathsf{DDH}$ security experiment with $\mathcal{B}$ running $\mathcal{A}$ as a subroutine as described above. Let $\mathtt{good}$ denote the event that variable $\mathtt{bad}$ is never set to 1. Then, since $\mathcal{B}$ outputs a random bit if $\mathtt{bad} := 1$ is set, it holds that

$$\begin{aligned}\Pr[G^{q\mathsf{DDH}}_{\mathcal{B}(\mathcal{A})} = 1] &= \Pr[G^{q\mathsf{DDH}}_{\mathcal{B}(\mathcal{A})} = 1 \wedge \mathtt{good}] + \Pr[\neg\mathtt{good}] \cdot \Pr[G^{q\mathsf{DDH}}_{\mathcal{B}(\mathcal{A})} = 1 \mid \neg\mathtt{good}] \\ &= \Pr[G^{q\mathsf{DDH}}_{\mathcal{B}(\mathcal{A})} = 1 \wedge \mathtt{good}] + \Pr[\neg\mathtt{good}] \cdot 1/2\end{aligned}$$

and therefore

$$\begin{aligned}\mathsf{Adv}^{q\mathsf{DDH}}_{\mathcal{B}}(k) &= 2 \cdot \Pr[G^{q\mathsf{DDH}}_{\mathcal{B}(\mathcal{A})} = 1] - 1 \\ &= 2 \cdot \Pr[G^{q\mathsf{DDH}}_{\mathcal{B}(\mathcal{A})} = 1 \wedge \mathtt{good}] - \Pr[\mathtt{good}] \qquad (6)\end{aligned}$$

Thus, it remains to derive suitable bounds on $\Pr[G^{q\mathsf{DDH}}_{\mathcal{B}(\mathcal{A})} = 1 \wedge \mathtt{good}]$ and $\Pr[\mathtt{good}]$. We will need the following lemma from [5, 7].

Lemma 3 ([5, 7]). *Let G_i and G_j be two games which proceed identical until* $\mathtt{bad} = 1$. *Then*

- $\Pr[G_i$ *sets* $\mathtt{bad} = 1] = \Pr[G_j$ *sets* $\mathtt{bad} = 1]$
- $\Pr[G_i = b \wedge G_i$ *does not set* $\mathtt{bad} = 1] = \Pr[G_j = b \wedge G_j$ *does not set* $\mathtt{bad} = 1]$ *for any b.*

A simpler-to-analyze game. Following Bellare and Ristenpart [5], we now gradually make changes to game $G^{q\mathsf{DDH}}_{\mathcal{B}(\mathcal{A})}$, until we reach game G_3, which will be easier to analyze. In the sequel let $\mathtt{good}_i$ denote the event that $\mathtt{bad}$ is never set to $\mathtt{bad} = 1$ in Game i.

Procedures for Game G_1:

$\underline{\mathbf{Evaluate}_1(X):}$
$(Y,\pi) := \perp$
If $F_K(X) \neq 1$ **then**
  $\mathtt{bad} := 1$
Else
  $(Y,\pi) \xleftarrow{\$} \mathsf{Eval}(sk, X)$
Return (Y,π)

$\underline{\mathbf{Challenge}_1(X^*):}$
$Y^* := \perp$
If $F_K(X) = 1$ **then**
  $\mathtt{bad} := 1$
Else
  If $b = 1$ **then**
    $(Y^*,\pi) \xleftarrow{\$} \mathsf{Eval}(sk, X)$
  Else $Y^* \xleftarrow{\$} \mathbb{G}_T$
Return Y^*

$\underline{\mathbf{Finalize}_1(b'):}$
If $\mathtt{bad} = 1$ **then** $c' \xleftarrow{\$} \{0,1\}$
Else $c' := b'$
If $c' = b$ **then Return** 1
Else Return 0

$\underline{\mathbf{Initialize}_1(X):}$
$\mathtt{bad} := 0$
$(vk, sk) \xleftarrow{\$} \mathsf{Gen}_C(1^k)$
$b \xleftarrow{\$} \{0,1\}$
$K \xleftarrow{\$} \mathsf{AdmSmp}(1^k, Q, \delta)$
Return vk

Procedures for Game G_2 (new instructions are highlighted in boxes):

$\underline{\mathbf{Evaluate}_2(X):}$
$(Y,\pi) := \perp$
If $F_K(X) \neq 1$ **then**
  $\mathtt{bad} := 1$
  $\boxed{(Y,\pi) \xleftarrow{\$} \mathsf{Eval}(sk, X)}$
Else
  $(Y,\pi) \xleftarrow{\$} \mathsf{Eval}(sk, X)$
Return (Y,π)

$\underline{\mathbf{Challenge}_2(X^*):}$
$Y^* := \perp$
If $F_K(X) = 1$ **then**
  $\mathtt{bad} := 1$
  $\boxed{\textbf{If } b = 1 \textbf{ then}}$
    $\boxed{(Y^*,\pi) \xleftarrow{\$} \mathsf{Eval}(sk, X)}$
  $\boxed{\textbf{Else } Y^* \xleftarrow{\$} \mathbb{G}_T}$
Else
  If $b = 1$ **then**
    $(Y^*,\pi) \xleftarrow{\$} \mathsf{Eval}(sk, X)$
  Else $Y^* \xleftarrow{\$} \mathbb{G}_T$
Return Y^*

$\underline{\mathbf{Finalize}_2(b'):}$
If $\mathtt{bad} = 1$ **then** $\boxed{c' := b'}$
Else $c' := b'$
If $c' = b$ **then Return** 1
Else Return 0

Procedures for Game G_3 (new instructions are highlighted in boxes):

$\underline{\mathbf{Evaluate}_3(X):}$
$\boxed{\mathbf{X} := \mathbf{X} \cup \{X\}}$
$(Y,\pi) \xleftarrow{\$} \mathsf{Eval}(sk, X)$
Return (Y,π)

$\underline{\mathbf{Challenge}_3(X^*):}$
If $b = 1$ **then**
  $(Y^*,\pi) \xleftarrow{\$} \mathsf{Eval}(sk, X)$
Else $Y^* \xleftarrow{\$} \mathbb{G}_T$
Return Y^*

$\underline{\mathbf{Initialize}_3(X):}$
$\mathtt{bad} := 0$
$(vk, sk) \xleftarrow{\$} \mathsf{Gen}_C(1^k)$
$b \xleftarrow{\$} \{0,1\}$
$\boxed{\mathbf{X} := \emptyset}$
Return vk

$\underline{\mathbf{Finalize}_3(b'):}$
$\boxed{K \xleftarrow{\$} \mathsf{AdmSmp}(1^k, Q, \delta)}$
$\boxed{\textbf{For } X \in \mathbf{X} \textbf{ do}}$
  $\boxed{\textbf{If } F_K(X) \neq 1 \textbf{ then } \mathtt{bad} := 1}$
$\boxed{\textbf{If } F_K(X^*) = 1 \textbf{ then } \mathtt{bad} := 1}$
$c' := b'$
If $c' = b$ **then Return** 1
Else Return 0

Fig. 5. Procedures defining the sequence of games in the proof of Lemma 1.

Game 0. We define $G_0 := G^{q\mathsf{DDH}}_{\mathcal{B}(\mathcal{A})}$, which implies

$$\Pr[G^{q\mathsf{DDH}}_{\mathcal{B}(\mathcal{A})} = 1 \wedge \mathtt{good}] = \Pr[G_0 = 1 \wedge \mathtt{good}_0] \quad \text{and} \quad \Pr[\mathtt{good}] = \Pr[\mathtt{good}_0]$$

Game 1. In this game the procedures $\mathbf{Initialize}_1$, $\mathbf{Evaluate}_1$, $\mathbf{Challenge}_1$, and $\mathbf{Finalize}_1$ described in Figure 5 are used. Note that $\mathbf{Initialize}_1$ generates a normal VRF key pair (vk, sk), and $\mathbf{Evaluate}_1$ and $\mathbf{Challenge}_1$ use the secret key sk to evaluate the VRF and to create the challenge.

However, note that sk is only used in $\mathbf{Evaluate}_1(X)$-queries with $F_K(X) = 1$, and $\mathbf{Challenge}_1(X^*)$-queries with $F_K(X^*) = 0$. This mimics the simulation of $\mathcal{B}$ perfecty, in particular all outputs computed by these procedures are distributed *exactly* like in Game 0. This implies that

$$\Pr[G_1 = 1 \wedge \mathtt{good}_1] = \Pr[G_0 = 1 \wedge \mathtt{good}_0] \quad \text{and} \quad \Pr[\mathtt{good}_1] = \Pr[\mathtt{good}_0]$$

Game 2. In this game we set $\mathbf{Initialize}_2 := \mathbf{Initialize}_1$, and define $\mathbf{Finalize}_2$, $\mathbf{Evaluate}_2$, and $\mathbf{Challenge}_2$ as depicted in Figure 5. Note that Games G_2 and G_1 proceed identical until $\mathtt{bad}$ is set, thus by Lemma 3 we have

$$\Pr[G_2 = 1 \wedge \mathtt{good}_2] = \Pr[G_1 = 1 \wedge \mathtt{good}_1] \quad \text{and} \quad \Pr[\mathtt{good}_2] = \Pr[\mathtt{good}_1]$$

Game 3. Note that the outputs of procedures $\mathbf{Evaluate}_2$ and $\mathbf{Challenge}_2$ are independent of K, only $\mathbf{Finalize}_2$ depends on K. Therefore we can simplify our description of the game, by choosing K only at the end of the game, and checking only then if $\mathtt{bad}$ needs to be set to $\mathtt{bad} := 1$.

Formally, in Game G_3 the procedures $\mathbf{Initialize}_3$, $\mathbf{Evaluate}_3$, $\mathbf{Challenge}_3$, and $\mathbf{Finalize}_3$ described in Figure 5 are used. All changes are purely conceptual, thus we have

$$\Pr[G_3 = 1 \wedge \mathtt{good}_3] = \Pr[G_2 = 1 \wedge \mathtt{good}_2] \quad \text{and} \quad \Pr[\mathtt{good}_3] = \Pr[\mathtt{good}_2]$$

Note also that now K is chosen only *after* $\mathcal{A}$ asks $\mathbf{Finalize}_3$.

Analysis of Game G_3. It remains to derive bounds on $\Pr[G_3 = 1 \wedge \mathtt{good}_3]$ and $\Pr[\mathtt{good}_3]$. Let $\mathcal{X}$ denote the set

$$\mathcal{X} := \{(X^{(1)}, \dots, X^{(Q)}, X^*) : X^* \neq X^{(i)}, 1 \leq i \leq Q\}$$

of all sequences of queries a legitimate attacker $\mathcal{A}$ may ask, and let $\mathbf{X}^* \in \mathcal{X}$. Let $\gamma(\mathbf{X}^*)$ denote the probability of $\mathtt{good}_3$ (over the choice of K), if the particular sequence $\mathbf{X}^*$ of queries is asked. Note that $\gamma(\mathbf{X}^*)$ equals the probability in Equation (3), so that γ_{min} is a lower bound on the smallest value of $\gamma(\mathbf{X}^*)$ over all $\mathbf{X}^* \in \mathcal{X}$, and γ_{max} is an upper bound on the largest value of $\gamma(\mathbf{X}^*)$ over all $\mathbf{X}^* \in \mathcal{X}$. Let $\mathsf{Q}(\mathbf{X}^*)$ denote the event that the execution of Game G_3 results in the particular sequence $\mathbf{X}^*$. Then we can state the following lemma (which corresponds to [6, Lemma 3.4]).

Lemma 4. *For any $\mathbf{X}^*$ as defined above holds that*

$$\Pr[G_3 = 1 \wedge \mathsf{good}_3 \wedge \mathsf{Q}(\mathbf{X}^*)] = \gamma(\mathbf{X}^*) \cdot \Pr[G_3 = 1 \wedge \mathsf{Q}(\mathbf{X}^*)]$$
$$\Pr[\mathsf{good}_3 \wedge \mathsf{Q}(\mathbf{X}^*)] = \gamma(\mathbf{X}^*) \cdot \Pr[\mathsf{Q}(\mathbf{X}^*)]$$

The proof of Lemma 4 is nearly identical to the proof of [6, Lemma 3.4], and therefore deferred to Appendix B.

Now we can compute

$$\begin{aligned}
\mathsf{Adv}^{q\mathsf{DDH}}_{\mathcal{B}}(k) &= 2 \cdot \Pr[G^{q\mathsf{DDH}}_{\mathcal{B}(\mathcal{A})} = 1 \wedge \mathsf{good}] - \Pr[\mathsf{good}] && (7)\\
&= 2 \cdot \Pr[G_3 = 1 \wedge \mathsf{good}_3] - \Pr[\mathsf{good}_3] && (8)\\
&= 2 \cdot \sum_{\mathbf{X}^* \in \mathcal{X}} \Pr[G_3 = 1 \wedge \mathsf{good}_3 \wedge \mathsf{Q}(\mathbf{X}^*)] - \sum_{\mathbf{X}^* \in \mathcal{X}} \Pr[\mathsf{good}_3 \wedge \mathsf{Q}(\mathbf{X}^*)] && (9)\\
&= 2 \cdot \sum_{\mathbf{X}^* \in \mathcal{X}} \gamma(\mathbf{X}^*) \cdot \Pr[G_3 = 1 \wedge \mathsf{Q}(\mathbf{X}^*)] - \sum_{\mathbf{X}^* \in \mathcal{X}} \gamma(\mathbf{X}^*) \cdot \Pr[\mathsf{Q}(\mathbf{X}^*)] && (10)\\
&\geq 2 \cdot \gamma_{\mathsf{min}} \cdot \sum_{\mathbf{X}^* \in \mathcal{X}} \Pr[G_3 = 1 \wedge \mathsf{Q}(\mathbf{X}^*)] - \gamma_{\mathsf{max}} \cdot \sum_{\mathbf{X}^* \in \mathcal{X}} \Pr[\mathsf{Q}(\mathbf{X}^*)]\\
&= 2 \cdot \gamma_{\mathsf{min}} \cdot \Pr[G_3 = 1] - \gamma_{\mathsf{max}} && (11)\\
&= 2 \cdot \gamma_{\mathsf{min}} \cdot (\mathsf{Adv}^{\mathcal{VF}}_{\mathcal{A}}(k) + 1)/2 - \gamma_{\mathsf{max}}\\
&= \gamma_{\mathsf{min}} \cdot \mathsf{Adv}^{\mathcal{VF}}_{\mathcal{A}}(k) - \gamma_{\mathsf{max}} + \gamma_{\mathsf{min}}\\
&\geq 2 \cdot \gamma_{\mathsf{min}} \cdot \delta - \gamma_{\mathsf{max}} + \gamma_{\mathsf{min}} && (12)
\end{aligned}$$

Here, (7) is due to Equation (6), (8) follows from the sequence of games described above, (9) and (11) follow from the fact that we sum over mutually exclusive events $\mathsf{Q}(\mathbf{X}^*)$ with $\sum_{\mathbf{X}^* \in \mathcal{X}} \Pr[\mathsf{Q}(\mathbf{X}^*)] = 1$, (10) is by Lemma 4, and (12) by the definition of $\delta \leq \mathsf{Adv}^{\mathcal{VF}}_{\mathcal{A}}(k)/2$.

B Proof of Lemma 4

The execution of AdmSmp in Game 3 uses random coins which are independent of the rest of the game. Therefore, the set of random coins underlying Game 3 can be seen as a cross product $\Omega = \Omega' \times R_K$, where each member is a pair $(\omega', r_K) \in \Omega$ such that r_K denotes the random coins used by algorithm AdmSmp, and ω' denotes all other coins of the experiment and the attacker.

Note that that any particular choice $\mathbf{X}^*$ of a sequence of queries made by $\mathcal{A}$ depends only on ω', because in Game 3 algorithm AdmSmp is executed in the $\mathbf{Finalize}_3$-procedure, when the sequence of queries $\mathbf{X}^*$ issued by the attacker is already fixed. Thus, for all $\mathbf{X}^* \in \mathcal{X}$ let $\Omega'(\mathbf{X}^*)$ denote the set of all $\omega' \in \Omega'$ that produce the particular sequence of queries $\mathbf{X}^*$. Similarly, note that the probability that Game 3 outputs 1 depends only on Ω'.

Let $\Omega'_1 \subseteq \Omega'$ denote the set of all $\omega' \in \Omega'$ such that the experiment outputs 1. Let $R_{\mathsf{good}}(\mathbf{X}^*) \subseteq R_K$ denote the set of all coins leading to an AHF key K such that for $\mathbf{X}^* = (X^{(1)}, \dots, X^{(Q)}, X^*)$ holds that

$$F_K(X^{(1)}) = \cdots = F_K(X^{(Q)}) = 1 \quad \wedge \quad F_K(X^*) = 0$$

Then the set of coins such that $G_3 = 1$ is $\Omega'_1 \times R_K$, and the set of coins leading to $\mathtt{good}_3 \wedge \mathsf{Q}(\mathbf{X}^*)$ is $\Omega'(\mathbf{X}^*) \times R_{\mathsf{good}}(\mathbf{X}^*)$. Now we can compute

$$\begin{aligned}
\Pr[G_3 = 1 \wedge \mathtt{good}_3 \wedge \mathsf{Q}(\mathbf{X}^*)] &= \frac{|(\Omega'_1 \times R_K) \cap (\Omega'(\mathbf{X}^*) \times R_{\mathsf{good}}(\mathbf{X}^*))|}{|\Omega' \times R_K|} \\
&= \frac{|(\Omega'_1 \cap \Omega'(\mathbf{X}^*)) \times R_{\mathsf{good}}(\mathbf{X}^*)|}{|\Omega' \times R_K|} \\
&= \frac{|\Omega'_1 \cap \Omega'(\mathbf{X}^*)| \cdot |R_{\mathsf{good}}(\mathbf{X}^*)|}{|\Omega'| \cdot |R_K|} \\
&= \frac{|\Omega'_1 \cap \Omega'(\mathbf{X}^*)| \cdot |R_K|}{|\Omega'| \cdot |R_K|} \cdot \frac{|R_{\mathsf{good}}(\mathbf{X}^*)|}{|R_K|} \\
&= \frac{|(\Omega'_1 \cap \Omega'(\mathbf{X}^*)) \times R_K|}{|\Omega' \times R_K|} \cdot \frac{|R_{\mathsf{good}}(\mathbf{X}^*)|}{|R_K|} \\
&= \Pr[G_3 = 1 \wedge \mathsf{Q}(\mathbf{X}^*)] \cdot \gamma(\mathbf{X}^*)
\end{aligned}$$

and

$$\begin{aligned}
\Pr[\mathtt{good}_3 \wedge \mathsf{Q}(\mathbf{X}^*)] &= \frac{|\Omega'(\mathbf{X}^*) \times R_{\mathsf{good}}(\mathbf{X}^*)|}{|\Omega' \times R_K|} \\
&= \frac{|\Omega'(\mathbf{X}^*)| \cdot |R_{\mathsf{good}}(\mathbf{X}^*)|}{|\Omega'| \cdot |R_K|} \\
&= \frac{|\Omega'(\mathbf{X}^*)| \cdot |R_K|}{|\Omega'| \cdot |R_K|} \cdot \frac{|R_{\mathsf{good}}(\mathbf{X}^*)|}{|R_K|} \\
&= \frac{|\Omega'(\mathbf{X}^*) \times R_K|}{|\Omega' \times R_K|} \cdot \frac{|R_{\mathsf{good}}(\mathbf{X}^*)|}{|R_K|} \\
&= \Pr[\mathsf{Q}(\mathbf{X}^*)] \cdot \gamma(\mathbf{X}^*)
\end{aligned}$$