Constructing Verifiable Random Functions with Large Input Spaces

Susan Hohenberger[*]
Johns Hopkins University

Brent Waters[†]
University of Texas at Austin

February 18, 2010

Abstract

We present a family of verifiable random functions which are provably secure for exponentially-large input spaces under a non-interactive complexity assumption. Prior constructions required either an interactive complexity assumption or one that could tolerate a factor 2^n security loss for n-bit inputs. Our construction is practical and inspired by the pseudorandom functions of Naor and Reingold and the verifiable random functions of Lysyanskaya. Set in a bilinear group, where the Decisional Diffie-Hellman problem is easy to solve, we require the ℓ-Decisional Diffie-Hellman Exponent assumption in the standard model, without a common reference string. Our core idea is to apply a simulation technique where the large space of VRF inputs is collapsed into a small (polynomial-size) input in the view of the reduction algorithm. This view, however, is information-theoretically hidden from the attacker. Since the input space is exponentially large, we can first apply a collision-resistant hash function to handle arbitrarily-large inputs.

1 Introduction

Verifiable Random Functions (VRFs) were proposed by Micali, Rabin, and Vadhan [24]. VRFs behave similar to Pseudo Random Functions (PRFs) [16] in that an (efficient) attacker should not be able to distinguish the value of $f_K(x)$ from a random value even if it is given oracle access to the function $f_K(\cdot)$ at several other points. However, VRFs have the additional property that the party holding the seed will publish a commitment to the function and is able to *non-interactively* convince a verifier that a given evaluation is correct (i.e., matches the public commitment) without sacrificing the pseudorandom property on other inputs. In addition, the proof must be verifiable without the benefit of a common reference string (CRS). Finally, the verification should remain secure even if the public commitment were setup in a malicious manner.

The VRF definition of security limits the types of tools we can apply to solving the problem and restricts us from using several "traditional" approaches. For example, at first glance it might seem possible to construct VRFs in a straightforward manner by applying PRFs together with a Non-Interactive Zero Knowledge Proof [4, 5] (NIZK) system. A tempting approach is to publish a commitment to a PRF seed and then the seed holder can apply the NIZK machinery to produce non-interactive proofs. A typical proof would at one stage allow a reduction algorithm to simulate proofs (via knowledge of the CRS setup) even when the algorithm has no knowledge of the function's seed. However, since the definition of a Verifiable Random Function disallows the use of a trusted setup, the NIZK paradigm cannot be applied.

Without being able to simulate proofs, any reduction algorithm that proves pseudorandomness faces the following predicament. First, for any x for which it is asked to give out $f_K(x)$ and a proof it must be able to

[*] Supported by NSF CNS-0716142, Department of Homeland Security Grant 2006-CS-001-000001-02 (subaward 641) and a Microsoft Research New Faculty Fellowship.

[†] Supported by NSF CNS-0716199, CNS-0915361, and CNS-0952692, Air Force Office of Scientific Research (AFO SR) under the MURI award for "Collaborative policies and assured information sharing" (Project PRESIDIO), Department of Homeland Security Grant 2006-CS-001-000001-02 (subaward 641), and the Alfred P. Sloan Foundation.

produce the actual (unique) output of $f_K(x)$. Since there is no interaction or trusted setup, the algorithm is not able to "lie" at any stage. Second, the reduction must be able to use an attacker that can distinguish $f_K(x^*)$ from a random value at a certain x^*. In order to make use of this attacker, it follows that the reduction algorithm must *not* know how to evaluate $f_K(\cdot)$ at certain points.

Meeting these two restrictions will require a new approach to constructing pseudorandom functions that moves past traditional constructions. For instance, to prove that the Goldreich, Goldwasser and Micali PRF construction [16] is pseudorandom one must go through several hybrid experiments, where the reduction algorithm will not know how to correctly evaluate the PRF on any input. This approach will not work for proving VRFs, since the reduction algorithm must provide an evaluation and prove (without lying) that it is correct.

Constructing and Proving Security of VRFs. For the reasons above, existing VRF systems employ a different strategy when proving the security of VRFs. Almost all proofs of VRF constructions (that do not rely on interactive assumptions) [24, 14, 1] use a type of "all but one" technique for proving pseudorandoness. In these proofs a reduction algorithm will first guess the attacker's challenge input as some random string w in $\{0,1\}^n$, where n is the bit length of inputs. Next, it will set up the commitment such that it knows the function at all $2^n - 1$ inputs values $x \neq w$. The algorithm then must "hope" that the challenge input lands on w. For instance, the Micali-Rabin-Vadhan (VUF[1]) reduction [24] publishes a commitment $r \pmod N$ such that it knows $2^n - 1$ roots of r for primes p_x where $x \neq w$ and hopes that the attacker provides it the p_w-th root of r.

The main drawback of this style of proof is that the error and time component of the reduction respectively degrade and blowup by a factor 2^n, which is exponential in the input length n. The error reduction decreases by a factor of 2^n from the guessing of the challenge input and the time of the reduction requires $2^n - 1$ steps to "plant" knowledge of $f_K(x)$ for all $x \neq w$. For this reason these VRF systems when applied to large input sizes need to rely on strong assumptions that can absorb the loss of security. Furthermore, in the ℓ-type assumptions used in bilinear map constructions of Dodis-Yampolskiy [14], the number of terms (i.e., ℓ) associated with the assumption increases exponentially in n. [2] In general, we would like to prove security for large input spaces based on a "smaller" and more standard complexity assumption, which contains at most a polynomial number of terms.

Indeed, in their recent paper, Abdalla, Catalano and Fiore [1] stated that an open problem was to construct a VRF "supporting exponentially large (in the security parameter) identity spaces and provably secure under non interactive assumptions".

Our Approach. In this work, we aim to realize VRFs with large input sizes without applying complexity leveraging or interactive assumptions. Our main technique is that we apply a reduction technique where the input space of size 2^n is compressed *in the reduction algorithm's view* to a much smaller space. We can parameterize this compression such that the reduction algorithm knows the PRF value for all but a set S of size $\approx 1/q(\lambda)$ of the input, where $q(\lambda)$ is the (polynomial) number of queries made by an attacker and λ is a security parameter. We then "hope" that the challenge input lands in S and can finish the simulation without aborting a non-negligible fraction of the time.

Our construction makes use of bilinear groups. It has a similar structure to the PRF of Naor-Reingold [27] and the VRF of Lysyanskaya [23]. The setup algorithm will choose a group $\mathbb{G}$ of prime order p along with random group elements $g, h, U_0 = g^{u_0}, \ldots, U_n = g^{u_n}$ for random $u_0, \ldots, u_n \in \mathbb{Z}_p$. The evaluation of the VRF on input $x = x_1 \ldots x_n$ is

$$e(g^{u_0 \prod_{i=1}^{n} u_i^{x_i}}, h).$$

[1] VUF stands for *verifiable unpredictable function.* It relaxes the pseudorandomness requirement of the VRF, so that an (efficient) attacker should not be able to predict the value of $f_K(x)$ even if it is given oracle access to the function $f_K(\cdot)$ and its proof at several other points.

[2] We note that the second construction of Abdalla-Catalano-Fiore [1] has a polynomial number of assumption terms, but exponential degradation in the input size.

Proofs of the VRF are given using a step ladder approach in a manner similar to that appearing in other works [23, 1].

We prove the security of our scheme under the ℓ-Decisional Diffie-Hellman Exponent assumption [7] for $\ell = O(q(\lambda) \cdot n)$. This assumption gives the reduction algorithm g^{a^i} for $i = 1$ to 2ℓ except for a "hole" at $i = \ell$. In our reduction, we associate each U_i value with a value $g^{a^{y_j}}$ for some y_j. (The terms are further randomized so as to information-theoretically hide y_j from the outside. We ignore the randomization terms for this discussion.) For any input x, the reduction can evaluate the function and give a proof if $y_0 + \sum_{i=1}^{n} y_i^{x_i} \neq \ell$. For all other inputs $x \in S$ such that $y_0 + \sum_{i=1}^{n} y_i^{x_i} = \ell$, the reduction algorithm can successfully use an answer to defeat the DDHE assumption.

To achieve a polynomial (in n) reduction we must find a way to put a proper fraction of the inputs in S and to make the distribution of inputs in S close to random across the coins of the reduction. For this final goal, we parameterize and analyze our scheme in a manner similar to the Waters' [29] Identity-Based Encryption system. In this system, Waters showed how to partition a fraction of $\approx 1/q(\lambda)$ of the inputs into what he called a challenge set S. We will apply a similar partitioning approach, except we must adapt it to the multiplicative structure of our VRF.

We finally note that once we achieve a VRF for large enough input size n, we can apply one of two techniques to get a VRF for the input domain of $\{0,1\}^*$. First, we could simply let the setup algorithm choose a collision resistant hash function $H : \{0,1\}^* \rightarrow \{0,1\}^n$. The VRF would first hash the input down to n bits and then apply the core VRF. It is fairly straightforward to show that an attack would imply either finding a collision or attacking the core VRF. Another technique is to apply the tree-based extension given by Micali, Rabin, and Vadhan (MRV) [24] which allows extension to unbounded size inputs. This tree-based technique works if there are no collisions discovered in the core VRF applied at each node (i.e., no two nodes have the same label). In order for this to occur, the core input size must be large, which requires complexity leveraging in the MRV RSA construction, but does not when using our techniques.

1.1 Related Work

The concept of *pseudorandom functions* was proposed by Goldreich, Goldwasser and Micali [16]. They provided a definition and gave a generic method of constructing them from any one-way permutation. An efficient PRF based on the Decisional Diffie-Hellman assumption was proposed by Naor and Reingold [27].

Micali, Rabin and Vadhan [24] proposed the extension to verifiable random functions. They gave an RSA-type construction and proved security under what they called the RSA $s(k)$-Hardness Assumption. Roughly, for input length $a(k)$, the security of the VRF was $s'(k) = s(k)^{1/3}/(\text{poly}(k) \cdot 2^{a(k)})$. They then provided a tree-based method for extending the input size to $\{0,1\}^*$. Their construction elegantly showed how to first give a Verifiable Unpredictable Function (VUF) and then apply the Goldreich-Levin [17] hard core bit technique to get a VRF.

Lysyanskaya [23] provided the first VRF scheme from bilinear maps, which was also constructed as a transformation from a VUF. Our VRF construction follows a similar structure and is inspired by that of Lysyanskaya, although we will give a direct VRF construction without first providing a VUF. Dodis [13] extended the work of Lysyanskaya and showed how to give efficient constructions of a VRF directly (i.e., without going through any generic transformations). His VRF was also *distributed* in the sense that a collection of servers can hold shares of the seed and a certain threshold of these servers must cooperate to compute $f_K(x)$ or distinguish its outputs from random. Unfortunately, both of these works rely on interactive complexity assumptions (for large input spaces.)

Dodis and Yampolskiy [14] gave a very efficient VRF under a non-interactive assumption by applying the deterministic version of Boneh-Boyen [6] signatures. In a bilinear group $\mathbb{G}$ of prime order p, its seed is a single element of $\mathbb{Z}_p$ and its proof is a single element of $\mathbb{G}$. Its main drawback is that its security only holds for small input spaces. For n-bit inputs, the scheme's security relies on the $(\ell = 2^n)$-Decisional Diffie-Hellman Inversion assumption with a 2^n factor blowup in the time component.

Recently, Abdalla, Catalano and Fiore [1] gave two VRF constructions and showed some connections to Identity-Based Encryption [28, 8]. In particular, they showed that any IBE scheme with certain properties (e.g., deterministic key generation) implies VRFs, although some of these properties only appear in random oracle constructions of IBE systems.

Chase and Lysyanskaya [12] introduced a concept that they called a *simultable* VRF. Simultable VRFs allow the use of a common reference string (CRS) in order to simulate a proof of the PRF output. Connections to multi-theorem NIZKs were given. We note that reintroducing a CRS removes some of the fundamental challenges in constructing a VRF that we described above.

Brakerski, Goldwasser, Rothblum and Vaikuntanathan [10] introduced a relaxation of VRFs that they called *weak* VRFs. A weak VRF is similar to a VRF except it only needs to be secure if the attacker is allowed to see queries at inputs chosen *randomly*. While this does not meet the full goals of VRFs, the authors showed that weak VRFs imply NIZKs and provided constructions of weak VRFs from simple assumptions.

Applications of VRFs. VRFs have a variety of interesting applications, partially because they allow a short commitment to an exponential number of pseudorandom bits. Abdalla et al. [1] provide a nice summary of applications where VRFs are used as a building block, including resettable zero-knowledge proofs [25], micropayment schemes [26], updatable zero-knowledge databases [22] and verifiable transaction escrow schemes [21], to name a few. It also appears likely to us that suitable VRFs could be a useful alternative in several applications which, as part of the system, output the value of the PRF together with a proof (interactive or non-interactive) that the evaluation was correct and has some additional properties. Examples of this include compact e-cash [11], keyword search [15], set intersection protocols [18], and adaptive oblivious transfer protocols [20].

2 Definition

Definition 2.1 (Verifiable Random Function) *Let* $F : \{0,1\}^{\mathsf{seed}(\lambda)} \times \{0,1\}^{\mathsf{in}(\lambda)} \to \{0,1\}^{\mathsf{out}(\lambda)}$*, where* $\mathsf{seed}, \mathsf{in}, \mathsf{out}$ *are all polynomials in the security parameter* 1^λ*, be an efficient function. We say that* F *is a* verifiable random function *if there exist algorithms* $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ *such that*

- $\mathsf{Setup}(1^\lambda)$ *outputs a pair of keys* (pk, sk)*;*
- $\mathsf{Prove}_{sk}(x)$ *outputs a pair* $(F_{sk}(x), \pi_{sk}(x))$*, where* $F_{sk}(x)$ *is the function value and* $\pi_{sk}(x)$ *is the proof of correctness; and*
- $\mathsf{Verify}_{pk}(x, y, \pi)$ *verifies that* $y = F_{sk}(x)$ *using the proof* π*.*

Formally, we require the following properties:

1. **Provability:** *For all* $(pk, sk) \in \mathsf{Setup}(1^\lambda)$ *and inputs* $x \in \{0,1\}^{\mathsf{in}(\lambda)}$*, if* $(y, \pi) = \mathsf{Prove}_{sk}(x)$*, then* $\mathsf{Verify}_{pk}(x, y, \pi) = 1$*.*
2. **Uniqueness:** *For all* $(pk, sk) \in \mathsf{Setup}(1^\lambda)$ *and inputs* $x \in \{0,1\}^{\mathsf{in}(\lambda)}$*, there does not exist a tuple* (y_1, y_2, π_1, π_2) *such that:*
   (1) $y_1 \neq y_2$*, (2)* $\mathsf{Verify}_{pk}(x, y_1, \pi_1) = 1$*, and (3)* $\mathsf{Verify}_{pk}(x, y_2, \pi_2) = 1$*.*
3. **Pseudorandomness:** *For all p.p.t. distinguishers* $D = (D_1, D_2)$*, there exists a negligible function* μ *such that:*

$$\Pr[(pk, sk) \leftarrow \mathsf{Setup}(1^\lambda);\ (x, s) \leftarrow D_1^{\mathsf{Prove}(\cdot)}(1^\lambda, pk);\ y_0 = F_{sk}(x);\ y_1 \leftarrow \{0,1\}^{\mathsf{out}(\lambda)};\ b \leftarrow \{0,1\};\ b' \leftarrow D_2^{\mathsf{Prove}(\cdot)}(y_b, s)\ :\ b = b' \wedge x \notin S] \leq \frac{1}{2} + \mu(\lambda),$$

where S *is the set of all inputs that* D *queries to its oracle* Prove*.*

3 Algebraic Settings

Bilinear Groups Let $\mathbb{G}$ and $\mathbb{G}_T$ be algebraic groups. A *bilinear map* is an efficient mapping $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ which is both: (*bilinear*) for all $g \in \mathbb{G}$ and $a, b \leftarrow \mathbb{Z}$, $e(g^a, g^b) = e(g, g)^{ab}$; and (*non-degenerate*) if g generates $\mathbb{G}$, then $e(g, g) \neq 1$.

Assumption 3.1 (ℓ-Decisional Diffie-Hellman Exponent [7, 9]) *Let $\mathbb{G}, \mathbb{G}_T$ be groups of prime order $p \in \Theta(2^\lambda)$. For all p.p.t. adversaries $\mathcal{A}$, there exists a negligible function μ such that*

$$\Pr[g, h \leftarrow \mathbb{G};\ a \leftarrow \mathbb{Z}_p;\ y_0 = e(g, h)^{a^\ell};\ y_1 \leftarrow \mathbb{G}_T;\ b \leftarrow \{0, 1\};$$
$$b' \leftarrow \mathcal{A}(g, h, g^a, \ldots, g^{a^{\ell-1}}, g^{a^{\ell+1}}, \ldots, g^{a^{2\ell}}, y_b)\ :\ b = b'] \leq \frac{1}{2} + \mu(\lambda).$$

4 VRF Construction from the DDHE Assumption

Setup(1^λ) We describe a system for inputs of length n, a polynomial in 1^λ.[3] The setup algorithm first chooses a bilinear group $\mathbb{G}$ of prime order p. It selects random generators $g, h \in \mathbb{G}$. It next selects random values $u_0, u_1, \ldots, u_n \in \mathbb{Z}_p$ and sets $U_0 = g^{u_0}, U_1 = g^{u_1}, \ldots, U_n = g^{u_n}$. It then sets the keys as:

$$pk = (\mathbb{G}, p, g, h, U_0, \ldots, U_n), \quad sk = (\mathbb{G}, p, g, h, u_0, \ldots, u_n).$$

Evaluate(sk, x) For $x \in \{0, 1\}^n$, the function F_{sk} evaluates $x = x_1 x_2 ... x_n$ as:

$$F_{sk}(x) = e(g^{u_0 \prod_{i=1}^n u_i^{x_i}}, h)$$

Prove(sk, x) This algorithm outputs $F_{sk}(x)$ together with a proof π comprised as follows. For $i = 1$ to n, compute $\pi_i = g^{\prod_{j=1}^i u_j^{x_j}}$. Next, compute $\pi_0 = g^{u_0 \prod_{j=1}^n u_j^{x_j}}$. Output the proof

$$\pi = (\pi_0, \pi_1, \ldots, \pi_n).$$

We observe that this formulation of π is redundant. It is not necessary to include π_i when $x_i = 0$, since in this case, we have $\pi_i = \pi_{i-1}$ (for $i > 1$) and $\pi_1 = g$ (for $i = 1$).

Verify(pk, x, y, π) The first step is to verify that all parts of the input are properly encoded group elements; in particular, that the proof $\pi = (\pi_0, \ldots, \pi_n)$ contains legal encodings of elements in $\mathbb{G}$. Next, the proof is verified in a step-by-step manner by checking that

$$e(\pi_1, g) = \begin{cases} e(g, g) & \text{if } x_1 = 0; \\ e(U_1, g) & \text{otherwise.} \end{cases}$$

and then for $i = 2$ to n, it holds that

$$e(\pi_i, g) = \begin{cases} e(\pi_{i-1}, g) & \text{if } x_i = 0; \\ e(\pi_{i-1}, U_i) & \text{otherwise.} \end{cases}$$

and finally that

$$e(\pi_0, h) = y.$$

Output 1 if and only if all checks verify.

[3] Due to the fact that n can be polynomial in the security parameter, we can accept inputs of arbitrary length by first applying a collision resistant hash function.

Efficiency Discussion. The output of the PRF $F_{sk}(\cdot)$ is one element in $\mathbb{G}_T$. As noted above, our representation of π is redundant and can be simplified. For an n-bit input x, the proof π requires at most $\mathbf{ones}(x)+1 \leq n+1$ elements in $\mathbb{G}$, where $\mathbf{ones}(\cdot)$ counts the number of bits set to 1 in the input. (Individual) verification of the VRF output requires $\mathbf{ones}(x)+1 \leq n+1$ pairings, if $e(g,g)$ is provided in the public key.

For applications where several VRF outputs need to be verified at the same time, one can apply standard batching techniques [2] to perform N verifications for n-bit inputs at a cost of $\leq 3n = O(n)$ total pairing operations. The batch verification algorithm takes as input N tuples of the form $(x_i, y_i = f_{sk}(x_i), \pi_i)$ and outputs 1 if and only if all individual proofs verify, with an error rate of 2^{-k} for security parameter k.

The batching algorithm would first verify the respective group memberships of all y_i and all values in $\pi_i = (\pi_{i,0}, \ldots, \pi_{i,n})$. It then chooses random $r_0, \ldots, r_N \in \{0,1\}^k$ and verifies that:

$$e(\prod_{i=1}^{N} \pi_{i,1}^{r_i}, g) = e(g^{\sum_{j=1}^{N}(1-x_j)r_j} \cdot U_1^{\sum_{i=1}^{N} x_i r_i}, g)$$

and then for $t = 2$ to n, it holds that

$$e(\prod_{i=1}^{N} \pi_{i,t}^{r_i}, g) = e(\prod_{i=1}^{N} \pi_{i,t-1}^{x_i r_i}, g) \cdot e(\prod_{i=1}^{N} \pi_{i,t-1}^{x_i r_i}, U_t)$$

To see the above, recall that $e(1,g) = 1$. Finally, we check that

$$e(\prod_{i=1}^{N} \pi_{i,0}^{r_i}, h) = \prod_{i=1}^{N} y_i^{r_i}.$$

Output 1 if and only if all checks verify.

5 Proof of DDHE VRF

Theorem 5.1 *The VRF construction in Section 4 is secure with respect to Definition 2.1 under the ℓ-DDHE assumption.*

Proof. The *provability* property is verifiable in a straightforward manner from the construction. The *uniqueness* property also follows easily from the group structure; that is, for any input, there is only one group element in $\mathbb{G}$ that is the valid output and moreover, that it is not possible (even for an unbounded adversary) to devise a valid proof for another element.

Showing *pseudorandomness* will require more work. To show pseudorandomness, we will employ a proof technique from the Waters IBE system [29] that allows us to partition the inputs into two sets: those the simulator can properly answer and those we hope the adversary chooses as a challenge. The main difficulty in adapting this technique is that Waters was able to manipulate the randomness in the IBE keys during simulation, whereas we are now dealing with a deterministic function evaluation. Nevertheless, by strengthening the complexity assumption and making subtle changes throughout the proof, we are able to complete the argument.

Suppose there is a p.p.t. distinguisher D which makes Q Prove queries in the pseudorandomness game and succeeds with probability $\frac{1}{2} + \epsilon$. Then we show how to use D to create an adversary $\mathcal{B}$ which breaks the ℓ-DDHE assumption with probability $\frac{1}{2} + \frac{3\epsilon}{64Q(n+1)}$, where $\ell = 4Q(n+1)$ and n is the bit length of the VRF input.

On input $(\mathbb{G}, p, g, h, g^a, \ldots, g^{a^{\ell-1}}, g^{a^{\ell+1}}, \ldots, g^{a^{2\ell}}, Y)$, our ℓ-DDHE solver $\mathcal{B}$ proceeds as:

Setup The simulator first sets an integer $m = 4Q$ and chooses an integer, k, uniformly at random between 0 and n. Recall that Q is the number of queries made by the distinguisher and n is the bit length of the VRF input. It then chooses random integers $r_1, \ldots, r_n, r'$ between 0 and $m-1$. Additionally, the simulator chooses random values $s_1, \ldots, s_n, s' \in \mathbb{Z}_p$. These values are all kept internal to the simulator. Intuitively, the r values will be used to embed the challenge, while the s values will be used as blinding factors to present the proper distribution to the distinguisher.

For $x \in \{0,1\}^n$, let $X \subseteq \{1, \ldots, n\}$ be the set of the all i for which $x_i = 1$. To ease our analysis, we define the functions:

$$C(x) = m(1+k) + r' + \sum_{i \in X} r_i \quad , \quad \hat{C}(x,i) = \sum_{j=1}^{i} x_j r_j$$

$$J(x) = s' \prod_{i \in X} s_i \quad , \quad \hat{J}(x,i) = \prod_{j=1}^{i} s_j^{x_j}$$

For inputs $x \in \{0,1\}^n$, we define the binary function

$$K(x) = \begin{cases} 0 & \text{if } r' + \sum_{j=1}^{n} x_j r_j \equiv 0 \mod m; \\ 1 & \text{otherwise.} \end{cases}$$

The simulator sets $U_0 = (g^{a^{m(1+k)+r'}})^{s'}$ and $U_i = (g^{a^{r_i}})^{s_i}$ for $i = 1$ to n. It outputs the public key as $(\mathbb{G}, p, g, h, U_0, \ldots, U_n)$, where implicitly the secret key contains the values $u_0 = a^{m(1+k)+r'} s'$ and $\{u_i = a^{r_i} s_i\}_{i \in [1,n]}$.

Prove The distinguisher, D, will ask for VRF evaluations and proofs. On query input x, the simulator first checks if $C(x) = \ell$ and aborts if this is true. Otherwise, it outputs the value

$$F(x) = e((g^{a^{C(x)}})^{J(x)}, h).$$

It also computes $\pi_0 = (g^{a^{C(x)}})^{J(x)}$ and $\pi_i = (g^{a^{\hat{C}(x,i)}})^{\hat{J}(x,i)}$ for $i = 1$ to n, and then outputs the proof $\pi = (\pi_0, \pi_1, \ldots, \pi_n)$.

Given the above settings, it is easy to verify that for any value of $x \in \{0,1\}^n$:

1. The maximum value of $C(x)$ is $m(1+n) + (1+n)(m-1) < 2m(1+n) = 2\ell$.
2. For any $i \in [1,n]$, the maximum value of $\hat{C}(x,i)$ is $(m-1)n < m(n+1) = \ell$.

Thus, if $C(x) \neq \ell$, then the simulator can always correctly answer all parts of the query.

Response Eventually D will provide a challenge input x^*. If $C(x^*) = \ell$, $\mathcal{B}$ will return the value Y. When D responds with a guess b', $\mathcal{B}$ will also output b' as its ℓ-DDHE guess. If $C(x^*) \neq \ell$, $\mathcal{B}$ outputs a random bit as its ℓ-DDHE guess.

This ends our description of ℓ-DDHE adversary $\mathcal{B}$.

A Series of Games Analysis. We now argue that any successful adversary D against our scheme will have success in the game presented by $\mathcal{B}$. To do this, we first define a sequence of games, where the first game models the real security game and the final game is exactly the view of the adversary when interacting with $\mathcal{B}$. We then show via a series of claims that if D is successful against Game j, then it will also be successful against Game $j+1$.

Game 1: This game is defined to be the same as the VRF security game in Definition 2.1.

Game 2: The same as Game 1, with the exception that we keep a record of each query made by D, which we'll denote as $\overrightarrow{x} = (x^{(1)}, \ldots, x^{(Q)}, x^*)$, where x^* is the challenge input. At the end of the game, we set $m = 4Q$ and choose random integers $\overrightarrow{r} = (r_1, \ldots, r_n, r')$ between 0 and $m-1$ and a random integer k between 0 and n. We define the regular abort indicator function:

$$\tau(\overrightarrow{x}, \overrightarrow{r}, k) = \begin{cases} 1 & \text{if } r' + \sum_{j=1}^{n} x_j^* r_j \neq m(n-k) \ \bigvee_{i=1}^{Q} \ K(x^{(i)}) = 0; \\ 0 & \text{otherwise.} \end{cases}$$

This function $\tau(\overrightarrow{x}, \overrightarrow{r}, k)$ evaluates to 0 if the queries $\overrightarrow{x}$ will not cause a regular abort for the given choice of simulation values $\overrightarrow{r}, k$. Consider the probability over all simulation values for the given set of queries $\overrightarrow{x}$ as $\zeta(\overrightarrow{x}) = \Pr_{\overrightarrow{r},k}[\tau(\overrightarrow{x}, \overrightarrow{r}, k) = 0]$.

As in [29], the simulator estimates $\zeta(\overrightarrow{x})$ as ζ' by evaluating $\tau(\overrightarrow{x}, \overrightarrow{r}, k)$ with fresh random $\overrightarrow{r}, k$ values a total of $O(\epsilon^{-2} \ln(\epsilon^{-1}) \zeta_{\mathsf{min}}^{-1} \ln(\zeta_{\mathsf{min}}^{-1}))$ times. This does not require running the distinguisher again.

D's success in the game is then determined as follows:

1. *Regular Abort.* If $\tau(\overrightarrow{x}, \overrightarrow{r}, k) = 1$, then flip a coin $b \in \{0,1\}$ and say that D wins if $b = 0$ and loses otherwise.
2. *Balancing (Artificial) Abort.*[4] Let $\zeta_{\mathsf{min}} = \frac{1}{8Q(n+1)}$ as derived from Claim 5.2. If $\zeta' \geq \zeta_{\mathsf{min}}$, $\mathcal{B}$ will abort with probability $\frac{\zeta' - \zeta_{\mathsf{min}}}{\zeta'}$ (not abort with probability $\frac{\zeta_{\mathsf{min}}}{\zeta'}$). If it aborts, flip a coin $b \in \{0,1\}$ and say that D wins if $b = 0$ and loses otherwise.
3. Otherwise, D wins if it correctly guessed b' as in the real security game.

Game 3: The same as Game 2, with the exception that $\mathcal{B}$ tests if any abort conditions are satisfied, with each new query, and if so, follows the abort procedure immediately (i.e., flips a coin $b \in \{0,1\}$ and says that D wins if $b = 0$.)

Game 3 is exactly the view of D when interacting with $\mathcal{B}$. We will shortly prove that if D succeeds in Game 1 with probability $\frac{1}{2} + \epsilon$, then it succeeds in Game 3 with probability $\geq \frac{1}{2} + \frac{3\epsilon}{64Q(n+1)}$.

Establishing Three Claims about the Probability of Aborting. Before doing so, we establish one claim which was used above and two claims which will be needed shortly. Our first claim helps us establish a minimum probability that a given set of queries do not cause a *regular* abort. We use this minimum during our balancing abort in Game 2, to "even out" the probability of an abort over all possible queries. In the next two claims, we employ Chernoff Bounds to establish upper and lower bounds for *any* abort (regular or balancing) for any set of queries. The latter two claims will be used in the analysis of D's probability of success in Game 2.

Claim 5.2 *Let* $\zeta_{\mathsf{min}} = \frac{1}{8Q(n+1)}$. *For any query vector* $\overrightarrow{x}$, $\zeta(\overrightarrow{x}) \geq \zeta_{\mathsf{min}}$.

Proof of Claim 5.2 is similar to a related argument in [29] and appears in Appendix A.

Claim 5.3 *For any set of queries* $\overrightarrow{x}$*, the probability that there is an abort (i.e., regular or balancing) is* $\geq 1 - \zeta_{\mathsf{min}} - \frac{3}{8}\zeta_{\mathsf{min}}\epsilon$.

Proof. Let $\zeta_x = \zeta(\overrightarrow{x})$, as defined in Section 5, be the probability that a set of queries $\overrightarrow{x}$ do not cause a regular abort. In Game 2, $T = O(\epsilon^{-2} \ln(\epsilon^{-1}) \zeta_{\mathsf{min}}^{-1} \ln(\zeta_{\mathsf{min}}^{-1}))$ samples are taken to approximate this value as ζ'_x. By Chernoff Bounds, we have that for all $\overrightarrow{x}$,

$$\Pr[T\zeta'_x < T\zeta_x(1 - \frac{\epsilon}{8})] < e^{-[128\epsilon^{-2} \ln((\epsilon/8)^{-1}) \zeta_{\mathsf{min}}^{-1} \ln(\zeta_{\mathsf{min}}^{-1})(\zeta_{\mathsf{min}})(\epsilon/8)^2/2]},$$

[4] In Waters [29], this is called the artificial abort. Recently, Bellare and Ristenpart provided an analysis of the Waters' IBE without the artificial abort [3]. We could use their techniques here for an alternative, tighter analysis, but we would need to expand the input size of our ℓ-DDHE assumption by a factor of $1/\epsilon$, where the distinguisher's advantage is $1/2 + \epsilon$.

which reduces to

$$\Pr[\zeta'_x < \zeta_x(1 - \frac{\epsilon}{8})] < \zeta_{\mathsf{min}}\frac{\epsilon}{8}.$$

Recall that for a measured ζ'_x an artificial abort will not happen with probability $\zeta_{\mathsf{min}}/\zeta'_x$. The probability of aborting is

$$\begin{aligned}
\Pr[\mathsf{abort}] &= 1 - \Pr[\overline{\mathsf{abort}}] = 1 - \Pr[\overline{\mathsf{RA}}]\Pr[\overline{\mathsf{AA}}] = 1 - \zeta_x \Pr[\overline{\mathsf{AA}}] \\
&\geq 1 - \zeta_x(\zeta_{\mathsf{min}}\frac{\epsilon}{8} + \frac{\zeta_{\mathsf{min}}}{\zeta_x(1-\epsilon/8)}) \\
&\geq 1 - (\zeta_{\mathsf{min}}\frac{\epsilon}{8} + \frac{\zeta_{\mathsf{min}}}{1-\epsilon/8}) \\
&\geq 1 - (\frac{\zeta_{\mathsf{min}}\epsilon}{8} + \zeta_{\mathsf{min}}(1 + \frac{2\epsilon}{8})) \\
&\geq 1 - \zeta_{\mathsf{min}} - \zeta_{\mathsf{min}}\frac{3\epsilon}{8}
\end{aligned}$$

□

Claim 5.4 *For any set of queries $\overrightarrow{x}$, the probability that there is no abort (i.e., regular or balancing) is $\geq \zeta_{\mathsf{min}} - \frac{1}{4}\zeta_{\mathsf{min}}\epsilon$.*

Proof. Let $\zeta_x = \zeta(\overrightarrow{x})$, as defined in Section 5, be the probability that a set of queries $\overrightarrow{x}$ do not cause a regular abort. In Game 2, $T = O(\epsilon^{-2}\ln(\epsilon^{-1})\zeta_{\mathsf{min}}^{-1}\ln(\zeta_{\mathsf{min}}^{-1}))$ samples are taken to approximate this value as ζ'_x. By Chernoff Bounds, we have that for all $\overrightarrow{x}$,

$$\Pr[T\zeta'_x > T\zeta_x(1 + \frac{\epsilon}{8})] < e^{-[256\epsilon^{-2}\ln((\epsilon/8)^{-1})\zeta_{\mathsf{min}}^{-1}\ln(\zeta_{\mathsf{min}}^{-1})](\zeta_{\mathsf{min}})(\epsilon/8)^2/4]},$$

which reduces to

$$\Pr[\zeta'_x > \zeta_x(1 + \frac{\epsilon}{8})] < \zeta_{\mathsf{min}}\frac{\epsilon}{8}.$$

The probability of not aborting is equal to the probability of not regular aborting times the probability of not artificial aborting. Recall that for a measured ζ'_x an artificial abort (AA) will not happen with probability $\zeta_{\mathsf{min}}/\zeta'_x$. Therefore, for any x, the $\Pr[\overline{\mathsf{AA}}] \geq (1 - \frac{\zeta_{\mathsf{min}}\epsilon}{8})\frac{\zeta_{\mathsf{min}}}{\zeta_x(1+\epsilon/8)}$. It follows that

$$\Pr[\overline{\mathsf{abort}}] \geq \zeta_x(1 - \frac{\zeta_{\mathsf{min}}\epsilon}{8})\frac{\zeta_{\mathsf{min}}}{\zeta_x(1+\epsilon/8)} \geq \zeta_{\mathsf{min}}(1 - \frac{\epsilon}{8})^2 \geq \zeta_{\mathsf{min}}(1 - \frac{1}{4}\epsilon).$$

□

Analyzing D's Probability of Success in the Games. Define D's probability of success in Game x as $\mathbf{Adv}_D[\text{Game } x]$. We reason about the probability of D's success in the series of games as follows.

Lemma 5.5 *If $\mathbf{Adv}_D[\text{Game 1}] = \frac{1}{2} + \epsilon$, then $\mathbf{Adv}_D[\text{Game 2}] \geq \frac{1}{2} + \frac{3\cdot\epsilon}{64Q(n+1)}$.*

Proof. We begin by observing that $\mathbf{Adv}_D[\text{Game 2}]$ is

$$\begin{aligned}
&= \mathbf{Adv}_D[\text{Game 2}|\mathsf{abort}] \cdot \Pr[\mathsf{abort}] + \mathbf{Adv}_D[\text{Game 2}|\overline{\mathsf{abort}}] \cdot \Pr[\overline{\mathsf{abort}}] && (1)\\
&= \frac{1}{2}\Pr[\mathsf{abort}] + \mathbf{Adv}_D[\text{Game 2}|\overline{\mathsf{abort}}] \cdot \Pr[\overline{\mathsf{abort}}] && (2)\\
&= \frac{1}{2}\Pr[\mathsf{abort}] + \Pr[b = b'|\overline{\mathsf{abort}}] \cdot \Pr[\overline{\mathsf{abort}}] && (3)\\
&= \frac{1}{2}\Pr[\mathsf{abort}] + \Pr[b = b'] \cdot \Pr[\overline{\mathsf{abort}}|b = b'] && (4)\\
&= \frac{1}{2}\Pr[\mathsf{abort}] + (\frac{1}{2} + \epsilon) \cdot \Pr[\overline{\mathsf{abort}}|b = b'] && (5)\\
&\geq \frac{1}{2}(1 - \zeta_{\mathsf{min}} - s_1) + (\frac{1}{2} + \epsilon)(\zeta_{\mathsf{min}} - s_2) && (6)\\
&\geq \frac{1}{2} + \epsilon \cdot \zeta_{\mathsf{min}} - (s_1 + s_2) && (7)\\
&= \frac{1}{2} + \frac{3 \cdot \epsilon \cdot \zeta_{\mathsf{min}}}{8} && (8)\\
&= \frac{1}{2} + \frac{3 \cdot \epsilon}{64Q(n+1)} && (9)
\end{aligned}$$

Equation 2 follows from the fact that, in the case of abort, D's success is determined by a coin flip. It would be very convenient if we could claim that $\mathbf{Adv}_D[\text{Game 2} \mid \overline{\mathsf{abort}}] = \mathbf{Adv}_D[\text{Game 1}]$, but unfortunately, this is false. The event that D wins Game 2 and the event of an abort are not independent; however, we have inserted the balancing abort condition in the attempt to lessen the dependence between these events. Equation 3 simply states that, when there is no abort, D wins if and only if it guesses correctly. Equation 4 follows from Bayes' Theorem. In Equation 5, we observe that $\Pr[b = b']$ is exactly D's success in Game 1.

Now, the purpose of our balancing abort is to even the probability of aborting, for all queries of D, to be roughly ζ_{min}. This will also get rid of the conditional dependence on $b = b'$. There will be a small error, which must be taken into account. Suppose that $\Pr[\mathsf{abort}] \geq 1 - \zeta_{\mathsf{min}} - s_1$ and $\Pr[\overline{\mathsf{abort}}] \geq \zeta_{\mathsf{min}} - s_2$, which must hold for some error values s_1, s_2, then we derive Equation 6. Algebraic manipulation and recalling that $\epsilon \leq \frac{1}{2}$, brings us to Equation 7.

We set $\zeta_{\mathsf{min}} = \frac{1}{8Q(n+1)}$ from Claim 5.2. We know, for all queries, that $\Pr[\mathsf{abort}] \geq 1 - \zeta_{\mathsf{min}} - s_1$ where $s_1 = \frac{3}{8}\zeta_{\mathsf{min}}\epsilon$ from Claim 5.3 and that $\Pr[\overline{\mathsf{abort}}] \geq \zeta_{\mathsf{min}} - s_2$ where $s_2 = \frac{1}{4}\zeta_{\mathsf{min}}\epsilon$ from Claim 5.4. Plugging these values into Equations 7 and 8 establishes the lemma. □

Lemma 5.6 $\mathbf{Adv}_D[\text{Game 3}] = \mathbf{Adv}_D[\text{Game 2}]$.

Proof. We make the explicit observation that these games are equivalent by observing that their only difference is the time at which the regular aborts occur. The artificial abort stage is identical. All public parameters, evaluations and proofs have the same distribution up to the point of a possible abortion. In Game 2, the simulator receives all the queries $\overrightarrow{x}$, then checks if $\tau(\overrightarrow{x}, \overrightarrow{r}, k) = 1$ and aborts, taking a random guess, if so. In Game 3, the simulator checks with each new query x if $K(x) = 0$, which implies that the ending τ evaluation will be 1, and aborts, taking a random guess, if so. Therefore, the output distributions will be the same. □

□

Tightness of the Reduction. Using the (asymptotically) tighter analysis techniques of Hofheinz and Kiltz [19], the $1/n$ factor loss in our reduction, that occurs due to the Balancing Abort in Game 2, could be reduced to $1/\sqrt{n}$. Since the $1/Q$ factor loss is the dominating term in our concrete analysis, this improved analysis may provide only modest gains in practice.

6 Conclusion and Open Directions

Verifiable random functions are an interesting and useful cryptographic primitive, but to date, all known constructions for exponentially-large message spaces required interactive complexity assumptions or their concrete security degraded by an exponential factor. In this work, we presented an efficient construction which can handle arbitrarily-large inputs (by first applying a collision-resistant hash function) based on the ℓ-Decisional Diffie-Hellman Exponent assumption. Our security proof used techniques similar to the Waters IBE [29], where we partitioned the input space into those for which we can provide a proof and those which we cannot. We then showed that with non-negligible probability, the adversary will only query us on inputs for which we can provide proofs, except for the challenge query, for which the proof is unknown. The main technical difference when applying Waters' proof techniques is we must move from an additive to a multiplicative structure, work without randomness in the output to manipulate during the reduction, and operate under a different complexity assumption. Fortunately, we were still able to properly simulate access to an exponentially-large input space using a complexity assumption with only a polynomial-size input.

We believe this work is an important step towards better understanding how to construct verifiable random functions. It leaves open many interesting questions. First, it would be interesting to improve on the efficiency of our construction, especially by realizing a seed, proof size and verification time that are sublinear in the bit-length of the input. Second, one would like to know if it is possible to realize a VRF under a complexity assumption with a fixed input size, such as Decisional Diffie-Hellman. If this is not possible, perhaps one can show that a q-based assumption (with at least a polynomial number of terms) is inherently necessary. Finally, it would be interesting to see if this new construction allows for any additional applications of VRFs or if it can be used to reduce the overall complexity assumptions required by any constructions using VRFs.

References

[1] Michel Abdalla, Dario Catalano, and Dario Fiore. Verifiable random functions from identity-based key encapsulation. In *Advances in Cryptology – EUROCRYPT '09*, volume 5479, pages 554–571, 2009.

[2] Mihir Bellare, Juan A. Garay, and Tal Rabin. Fast batch verification for modular exponentiation and digital signatures. In *Advances in Cryptology – EUROCRYPT '98*, volume 1403 of LNCS, pages 236–250, 1998.

[3] Mihir Bellare and Thomas Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for Waters' IBE scheme. In *EUROCRYPT '09*, volume 5479, pages 407–424, 2009.

[4] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *STOC*, pages 103–112, 1988.

[5] Manuel Blum, Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. Noninteractive zero-knowledge. *SIAM J. Comput.*, 20(6):1084–1118, 1991.

[6] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In *Advances in Cryptology – EUROCRYPT '04*, volume 3027 of *Lecture Notes in Computer Science*, pages 382–400. Springer, 2004.

[7] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In *Advances in Cryptology – EUROCRYPT '05*, volume 3494, pages 440–456, 2005.

[8] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil Pairing. In *CRYPTO '01*, volume 2139 of LNCS, pages 213–229, 2001.

[9] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *Advances in Cryptology – CRYPTO '05*, volume 3621, pages 258–275, 2005.

[10] Zvika Brakerski, Shafi Goldwasser, Guy N. Rothblum, and Vinod Vaikuntanathan. Weak verifiable random functions. In *TCC*, pages 558–576, 2009.

[11] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact E-Cash. In *EUROCRYPT*, volume 3494 of LNCS, pages 302–321, 2005.

[12] Melissa Chase and Anna Lysyanskaya. Simulatable VRFs with applications to multi-theorem NIZK. In *Advances in Cryptology – CRYPTO '07*, volume 4622, pages 303–322, 2007.

[13] Yevgeniy Dodis. Efficient construction of (distributed) verifiable random functions. In *Public Key Cryptography*, volume 2567, pages 1–17, 2003.

[14] Yevgeniy Dodis and Aleksandr Yampolskiy. A Verifiable Random Function with Short Proofs an Keys. In *Public Key Cryptography*, volume 3386, pages 416–431, 2005.

[15] Michael J. Freedman, Yuval Ishai, Benny Pinkas, and Omer Reingold. Keyword search and oblivious pseudorandom functions. In *TCC '05*, volume 3378 of LNCS, pages 303–324, 2005.

[16] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33 (4):792–807, 1986.

[17] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *STOC '89*, pages 25–32, 1989.

[18] Carmit Hazay and Yehuda Lindell. Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries. In *TCC '05*, volume 4948 of LNCS, pages 155–175, 2008.

[19] Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. In *Advances in Cryptology – CRYPTO '08*, volume 5157 of LNCS, pages 21–38, 2008.

[20] Stanislaw Jarecki and Xiaomin Liu. Efficient oblivious pseudorandom function with applications to adaptive OT and secure computation of set intersection. In *TCC '09*, volume 5444 of LNCS, pages 577–594, 2009.

[21] Stanislaw Jarecki and Vitaly Shmatikov. Handcuffing big brother: an abuse-resilient transaction escrow scheme. In *EUROCRYPT '04*, volume 3027 of LNCS, pages 590–608, 2004.

[22] Moses Liskov. Updatable zero-knowledge databases. In *ASIACRYPT '05*, volume 3788 of LNCS, pages 174–198, 2005.

[23] Anna Lysyanskaya. Unique signatures and verifiable random functions from the DH-DDH separation. In *Advances in Cryptology – CRYPTO '02*, volume 2442, pages 597–612, 2002.

[24] Silvio Micali, Michael O. Rabin, and Salil P. Vadhan. Verifiable random functions. In *Symposium on Foundations of Computer Science (FOCS)*, pages 120–130. IEEE Computer Society, 1999.

[25] Silvio Micali and Leonid Reyzin. Soundness in the public-key model. In *CRYPTO '01*, volume 2139 of LNCS, pages 542–565, 2001.

[26] Silvio Micali and Ronald L. Rivest. Micropayments revisited. In *CT-RSA '02*, volume 2271 of LNCS, pages 149–163, 2002.

[27] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *Journal of the ACM*, 51, Number 2:231–262, 2004.

[28] Adi Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO '84*, volume 196 of LNCS, pages 47–53, 1984.

[29] Brent Waters. Efficient identity-based encryption without random oracles. In *Advances in Cryptology – EUROCRYPT '05*, volume 3494 of *Lecture Notes in Computer Science*, pages 320–329. Springer, 2005.

A Proof of Claim 5.2

Proof. In other words, the probability of the simulation *not* triggering a general abort is at least ζ_{min}. This analysis follows that of [29], which we reproduce here for completeness. Without loss of generality, we can assume the adversary always makes the maximum number of queries Q (since the probability of not aborting increases with fewer queries). Fix an arbitrary $\overrightarrow{x} = (x^{(1)}, \ldots, x^{(Q)}, x^*) \in \{0,1\}^n$. Then, with the probability over the choice of $\overrightarrow{r}, k$, we have that $\Pr[\overline{\mathsf{abort}} \text{ on } \overrightarrow{x}]$ is

$$= \Pr[\bigwedge_{i=1}^{Q} K(x^{(i)}) = 1 \ \wedge \ r' + \sum_{j=1}^{n} x_j^* r_j = m(n-k)] \tag{10}$$

$$= (1 - \Pr[\bigvee_{i=1}^{Q} K(x^{(i)}) = 0]) \Pr[r' + \sum_{j=1}^{n} x_j^* r_j = m(n-k) | \bigwedge_{i=1}^{Q} K(x^{(i)}) = 1] \tag{11}$$

$$\geq (1 - \sum_{i=1}^{Q} \Pr[K(x^{(i)}) = 0]) \Pr[r' + \sum_{j=1}^{n} x_j^* r_j = m(n-k) | \bigwedge_{i=1}^{Q} K(x^{(i)}) = 1] \tag{12}$$

$$= (1 - \frac{Q}{m}) \cdot \Pr[r' + \sum_{j=1}^{n} x_j^* r_j = m(n-k) \mid \bigwedge_{i=1}^{Q} K(x^{(i)}) = 1] \tag{13}$$

$$= \frac{1}{n+1} \cdot (1 - \frac{Q}{m}) \cdot \Pr[K(x^*) = 0 \mid \bigwedge_{i=1}^{Q} K(x^{(i)}) = 1] \tag{14}$$

$$= \frac{1}{n+1} \cdot (1 - \frac{Q}{m}) \cdot \frac{\Pr[K(x^*) = 0] \cdot \Pr[\bigwedge_{i=1}^{Q} K(x^{(i)}) = 1] \mid K(x^*) = 0]}{\Pr[\bigwedge_{i=1}^{Q} K(x^{(i)}) = 1]]} \tag{15}$$

$$\geq \frac{1}{(n+1)m} \cdot (1 - \frac{Q}{m}) \cdot \Pr[\bigwedge_{i=1}^{Q} K(x^{(i)}) = 1] \mid K(x^*) = 0] \tag{16}$$

$$= \frac{1}{(n+1)m} \cdot (1 - \frac{Q}{m}) \cdot (1 - \Pr[\bigvee_{i=1}^{Q} K(x^{(i)}) = 0] \mid K(x^*) = 0]) \tag{17}$$

$$\geq \frac{1}{(n+1)m} \cdot (1 - \frac{Q}{m}) \cdot (1 - \sum_{i=1}^{Q} \Pr[K(x^{(i)}) = 0] \mid K(x^*) = 0]) \tag{18}$$

$$= \frac{1}{(n+1)m} \cdot (1 - \frac{Q}{m})^2 \tag{19}$$

$$\geq \frac{1}{(n+1)m} \cdot (1 - \frac{2Q}{m}) \tag{20}$$

$$= \frac{1}{8Q(n+1)} \tag{21}$$

Equations 13 and 16 derive from $\Pr[K(x) = 0] = \frac{1}{m}$ for any query x. Equation 14 gets a factor of $\frac{1}{n+1}$ from the simulator taking a guess of k. Equation 15 follows from Bayes' Theorem. Equation 19 follows from the pairwise independence of the probabilities that $K(x) = 0, K(x') = 0$ for any pair of queries $x \neq x'$, since they will differ in at least one random r_j value. Equation 21 follows from our setting of $m = 4Q$. □