Secure Identity-Based Encryption in the Quantum Random Oracle Model

Mark Zhandry
Stanford University, USA
mzhandry@stanford.edu

Abstract

We give the first proof of security for an identity-based encryption scheme in the *quantum* random oracle model. This is the first proof of security for any scheme in this model that requires no additional assumptions. Our techniques are quite general and we use them to obtain security proofs for two random oracle hierarchical identity-based encryption schemes and a random oracle signature scheme, all of which have previously resisted quantum security proofs, even using additional assumptions. We also explain how to remove the extra assumptions from prior quantum random oracle model proofs. We accomplish these results by developing new tools for arguing that quantum algorithms cannot distinguish between two oracle distributions. Using a particular class of oracle distributions, so called *semi-constant* distributions, we argue that the aforementioned cryptosystems are secure against quantum adversaries.

Keywords: Quantum, Random Oracle, IBE, Signatures

1 Introduction

While quantum computation is not yet viable, Shor [Sho97] showed that when fully realized, quantum computers will break most of the cryptosystems used today, namely those based on factoring and discrete log. This has sparked the field of post-quantum cryptography, namely the search for systems secure against quantum adversaries. To be secure in this setting, a system must have an underlying difficult problem for quantum computers, as well as a security reduction showing how to solve this problem using an adversary that breaks the system. Problems based on lattices have so far resisted quantum attacks, and there is a considerable amount of literature on lattice-based cryptosystems. However, random oracle security proofs for these constructions typically only consider classical adversaries, thus failing to show security in the quantum world.

The random oracle model [BR93] is one area where many classical proofs lack quantum equivalents. This model is of interest because random oracle schemes tend to be more efficient than their standard model counterparts. Consequently, the most efficient lattice-based schemes are often constructed in the random oracle model. For example, Gentry, Peikert, and Vaikuntanathan [GPV08] show how to construct signatures and identity-based encryption (IBE). Cash et al. [CHKP10] and Agrawal, Boneh, and Boyen [ABB10] give hierarchical IBE schemes both in the standard model and the random oracle model, with the random oracle constructions being more efficient than the corresponding standard model schemes. Gordon, Katz, and Vaikuntanathan [GKV10] give a group signature scheme, while Boneh and Freeman [BF11] demonstrate a homomorphic signature scheme, all in the random oracle model.

All of these papers prove security against adversaries with classical access to the random oracle. However, when the scheme is instantiated, the random oracle is replaced with a hash function H, which a quantum adversary may evaluate on a quantum superposition of inputs. To model this ability, it is necessary to allow a quantum adversary to make quantum queries to the random oracle. We call this model the quantum random oracle model.

Proving security in the quantum random oracle model presents many challenges. Among them is the difficulty of efficiently simulating the random oracle. In the classical case, the random oracle is simulated on the fly, only generating randomness as needed. In the quantum setting, the adversary can query the oracle on an exponential superposition of inputs. Therefore, to make even the first query look random to the adversary, it would seem that we would need exponential randomness.

In addition to this difficulty, there are classical random oracle techniques that do not make sense if the adversary has quantum access to the random oracle. One such technique is that of Bellare and Rogaway [BR93] for proving the security of the Full Domain Hash signature scheme. In this technique, the reduction algorithm is given a challenge c to solve, and randomly guesses which oracle query the adversary will use to break the scheme. The algorithm embeds c into the response for this query, and if both the guess is correct and the adversary breaks the scheme, then the algorithm will be able to solve c.

In the quantum setting, this argument no longer applies because each oracle query might be over a superposition of exponentially many inputs. We could choose a random query and plug c into all outputs for that query, but this will not look like a random oracle to the adversary. Alternatively, we could choose one oracle input and plug c into the corresponding output for all queries. However, our chance of guessing correctly will then be exponentially small.

1.1 Our Contributions

We resolve some of the issues outlined above by giving a quantum analog of the technique of Bellare and Rogaway [BR93] and demonstrating how to simulate a random oracle without any additional computational assumptions. Specifically, we:

- Describe a new way to argue that quantum algorithms cannot distinguish between two distributions of oracles.
- Apply this approach to a new type of distribution of oracles, which we call *semi-constant* distributions, showing that they cannot be distinguished from random oracles.
- Use our results on semi-constant distributions to prove that the random oracle IBE scheme of Gentry et al. [GPV08] is secure against quantum adversaries. The basic idea is to plug the challenge c into a small fraction of inputs, making the oracle seen by the adversary a semi-constant distribution. The adversary thus behaves as though the oracle is random. If the adversary happens to use any of the inputs in this fraction, we are able to solve c.
- Show that this technique is general by applying it to the random oracle hierarchical IBE schemes of Cash et al. [CHKP10] and Agrawal et al. [ABB10].
- Prove that the generic Full Domain Hash signature scheme [BR93] is secure against quantum adversaries, though this remains a theoretical result until a trapdoor permutation is found that is secure against quantum adversaries.

- Use our techniques and k-wise independent functions to solve the problem of simulating quantum random oracles, thus making the above results unconditional.

1.2 Related Work

Quantum random oracles have been used in several prior works. For example, Bennett et al. [BBBV97] prove several quantum complexity results, including a proof that quantum computers cannot solve all of NP, relative to a random oracle. In a cryptographic setting, quantum random oracles have been used by Aaronson [Aar09] to construct quantum money and by Brassard and Salvail [BS08] and Brassard et al. [BHK$^+$11] to construct quantum analogs of Merkle's Puzzles.

Boneh et al. [BDF$^+$11] give a random oracle scheme that is secure against quantum adversaries with classical access to the oracle, but is insecure once the adversary has quantum access to the oracle. Despite this result, they show that there are circumstances in which a classical random oracle security reduction can also be used in the quantum setting. However, their proof simulates the random oracle using a pseudorandom function (PRF) that is secure against quantum adversaries, hence making their results conditional on the existence of such a PRF. Additionally, their techniques do not apply to the schemes analyzed in this paper.

There has also been progress toward converting classical security proofs into quantum proofs outside of the random oracle model. For instance, Unruh[Unr10] shows that classical statistical security in Canetti's universal composabitily (UC) model implies quantum statistical security. Hallgren, Smith, and Song [HSS11] give a two-party protocol that is quantum computationally secure in the UC.

2 Preliminaries

A function $\epsilon(n)$ is negligible if it is non-negative and smaller than any inverse polynomial. That is, for any polynomial $p(n)$, $\epsilon(n) < 1/p(n)$ for all sufficiently large n.

A probabilistic polynomial time (PPT) algorithm is a classical randomized algorithm that runs in time polynomial in the size of its input. We also call such algorithms efficient.

2.1 Weight Assignments

A weight assignment on a set $\mathcal{X}$ is a function $D : \mathcal{X} \to \mathbb{R}$ such that $\sum_{x\in\mathcal{X}} D(x) = 1$. We sometimes write $\Pr_D[event]$ to represent the sum of the weights of all outcomes consistent with that event. A distribution on X is a weight-assignment D such that $D(x) \geq 0$ for all $x \in \mathcal{X}$. If D is a distribution, way that x occurs with probability $D(x)$. Let $U_\mathcal{X}$ denote the uniform distribution over $\mathcal{X}$. That is, $U_\mathcal{X}(x) = 1/|\mathcal{X}|$. When the set $\mathcal{X}$ is clear, we may omit the subscript.

We define the distance between two weight assignments D_1 and D_2 over a set $\mathcal{X}$ as

$$|D_1 - D_2| = \sum_{x\in\mathcal{X}} |D_1(x) - D_2(x)|$$

If $|D_1 - D_2| \leq \epsilon$, we say that D_1 and D_2 are ϵ-close.

Given a set of weight assignments D_y over $\mathcal{X}$, indexed by $y \in \mathcal{Y}$, and a weight assignment D

over $\mathcal{Y}$, we can define a weight assignment D' over $\mathcal{X}$ where

$$D'(x) = \sum_{y \in \mathcal{Y}} D(y) D_y(x)$$

We write $D' = \sum_{y \in \mathcal{Y}} D(y) D_y$. Say each of the D_ys are actually distributions. Given two weight assignments D_1 and D_2 over $\mathcal{Y}$, define $D'_1 = \sum_{y \in \mathcal{Y}} D_1(y) D_y$ and $D'_2 = \sum_{y \in \mathcal{Y}} D_2(y) D_y$. It is not difficult to show that the distance between D'_1 and D'_2 is at most the distance between D_1 and D_2.

Consider the set of functions $H : \mathcal{X} \to \mathcal{Y}$ for sets $\mathcal{X}$ and $\mathcal{Y}$, denoted by $\mathcal{H}_{\mathcal{X},\mathcal{Y}}$. Consider a weight assignment D on $\mathcal{H}_{\mathcal{X},\mathcal{Y}}$. Let $\mathcal{W} \subseteq \mathcal{X}$. We define the marginal weight assignment $D_{\mathcal{W}}$ of D on $\mathcal{H}_{\mathcal{W},\mathcal{Y}}$ where the weight of a function $H_{\mathcal{W}} : \mathcal{W} \to \mathcal{Y}$ is equal to the sum of the weights of all $H \in \mathcal{H}_{\mathcal{X},\mathcal{Y}}$ that agree with $H_{\mathcal{W}}$ on $\mathcal{W}$. In other words,

$$D_{\mathcal{W}}(H_{\mathcal{W}}) = \Pr_D[H(w) = H_{\mathcal{W}}(w) \forall w \in \mathcal{W}]$$

We call two weight assignments D_1 and D_2 on $\mathcal{H}_{\mathcal{X},\mathcal{Y}}$ k-wise equivalent if for all $\mathcal{W} \subseteq \mathcal{X}$ of size k, the marginal weight assignments $D_{1,\mathcal{W}}$ and $D_{2,\mathcal{W}}$ (of D_1 and D_2) over $\mathcal{H}_{\mathcal{W},\mathcal{Y}}$ are identical.

2.2 Quantum Computation

A quantum algorithm is an algorithm executed on a quantum computer that produces a classical output. Most of the background in quantum computation needed to understand this paper is for the proof of Theorem 3.1. Since this proof appears in Appendix B, we present the necessary background there. In the meantime, we recall a couple basic facts about quantum computation, and refer the reader to [NC00] for a more thorough discussion.

Fact 1. Any classical computation can be implemented on a quantum computer.

Fact 2. Any function that has an efficient classical algorithm computing it can be implemented efficiently as a quantum-accessible oracle.

Fact 3. Given a quantum algorithm A with oracle access to an oracle O, each oracle O defines a probability distribution of the outputs of A. Hence, any weight assignment of oracles leads to a weight assignment of outputs of A, and if two weight assignments D_1 and D_2 are a distance ϵ apart, the weight assignment of the outputs of A under these distributions are a distance at most ϵ apart.

2.3 Cryptographic Primitives

Here we give a brief sketch of a few cryptographic primitives, and refer to Appendix A for more details. All primitives depend on a security parameter n.

An encryption scheme E is a triple of PPT algorithms $(\mathsf{E.Gen}, \mathsf{E.Enc}, \mathsf{E.Dec})$, where $\mathsf{E.Gen}(1^n)$ generates secret/public keys $(\mathsf{sk}, \mathsf{pk})$, $\mathsf{E.Enc}_{\mathsf{pk}}$ encrypts a message, and $\mathsf{E.Dec}_{\mathsf{sk}}$ decrypts a ciphertext. We use the indistinguishability under chosen plaintext attack (IND-CPA) notion of security [GM84].

An identity-based encryption (IBE) scheme $\mathsf{IBE} = (\mathsf{IBE.Gen}, \mathsf{IBE.Extract}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec})$ is a 4-tuple of PPT algorithms where $\mathsf{IBE.Gen}(1^n)$ generates master secret/public keys $(\mathsf{msk}, \mathsf{mpk})$, $\mathsf{IBE.Extract}_{\mathsf{msk}}$ generates secret keys for given identities, $\mathsf{IBE.Enc}_{\mathsf{mpk}}$ encrypts a message to an identity, and $\mathsf{IBE.Dec}$ decrypts a ciphertext sent to an identity by using the corresponding secret key. We use the indistinguishability under chosen plaintext attack (IND-ID-CPA) notion of security [BF01].

A signature scheme $\mathsf{S} = (\mathsf{S.Gen}, \mathsf{S.Sign}, \mathsf{S.Ver})$ is a triple of PPT algorithms where $\mathsf{S.Gen}(1^n)$ generates secret/public keys $(\mathsf{sk}, \mathsf{pk})$, $\mathsf{S.Sign}_{\mathsf{sk}}$ signs a message, and $\mathsf{S.Ver}_{\mathsf{pk}}$ verifies a signature. We use the existential unforgeablility under chosen message attack (UF-CMA) notion of security [GMR88].

A pre-image sampleable function (PSF) is a quadruple of algorithms $\mathsf{F} = (\mathsf{F.Gen}, \mathsf{F.Sample}, f, f^{-1})$ where $\mathsf{F.Gen}$ generates private/public keys $(\mathsf{sk}, \mathsf{pk})$, f_{pk} is a function, $\mathsf{F.Sample}$ samples x from a distribution D such that $f_{\mathsf{pk}}(x)$ is uniform, and $f_{\mathsf{sk}}^{-1}(y)$ samples from D conditioned on $f_{\mathsf{pk}}(x) = y$. For security, we use the notion of one-wayness.

A trapdoor permutation (TDP) is a special case of a PSF where f is bijective and $\mathsf{F.Sample}$ simply returns a random element in the domain of f_{pk}. Since $\mathsf{F.Sample}$ is already defined, we omit it from the specification.

2.4 Random Oracle Model

In the Random Oracle Model, we assume the existence of a random function H, and give all parties oracle access to this function. If A makes queries to an oracle H, we denote this as A^H. If a system $\mathcal{S}$ uses a random oracle in its specification, we denote this as $\mathcal{S}^H$. The algorithms comprising any cryptographic protocol can use H, as can the adversary. Thus we modify the security games for all cryptographic systems to allow the adversary to make random oracle queries.

When a random oracle scheme is implemented, some suitable hash function H is included in the specification. Any algorithm (adversary included) now replaces oracle queries with evaluations of this hash function. In the quantum setting, because a quantum algorithm can evaluate H on an arbitrary superposition, we must allow the quantum adversary to make quantum queries to the random oracle. We call this the quantum random oracle model.

3 Distinguishing Oracles With Quantum Queries

In this section, we give some tools for arguing that quantum algorithms cannot distinguish between two distributions of oracles.

Meyer and Pommersheim [MP11] show an impossibility result for classifying oracles drawn from a distribution using quantum queries. Using similar techniques, we argue the impossibility of distinguishing between two oracle distributions with quantum queries:

Theorem 3.1. *If D_1 and D_2 are $2q$-wise equivalent weight assignments on oracles, then the output weight assignment of any algorithm A making q quantum queries is the same under both distributions.*

This is proved in Appendix B. We now provide a new pseudometric for distributions on oracles:

Definition 3.2. *Let D_1 and D_3 be two weight assignments on oracles. Then $|D_1 - D_3|_{(k)}$ is the minimum of $|D_2 - D_1|$ over all weight assignments D_2 that are k-wise equivalent D_1.*

We argue that the minimum exists: Setting $D_2 = D_1$ shows $|D_1 - D_3|_{(k)} \leq |D_1 - D_3|$. Thus to find D_2, we are minimizing $|D_2 - D_3|$ (a continuous function in D_2) over the set of assignments D_2 that are k-wise similar to D_1 (a closed set) and at most $|D_1 - D_3|$ away from D_2 (a compact set). Thus we are minimizing over a compact set, so the minimum is actually obtained.

We also state without proof that $|D_1 - D_3|_{(k)}$ is a pseudometric. Namely, $|D - D|_{(k)} = 0$ for all D, $|D_1 - D_2|_{(k)} = |D_2 - D_1|_{(k)}$, and the triangle inequality is satisfied.

We now show that this metric is useful for quantum algorithms:

Corollary 3.3. *Let D_1 and D_3 be two oracle distributions. Then the output weight assignment of any q-query quantum algorithm A under D_1 and D_3 are $|D_1 - D_3|_{(2q)}$-close.*

Proof. By definition, there is an assignment D_2 which is $2q$-wise equivalent to D_1 such that $|D_2 - D_3| = |D_1 - D_3|_{(2q)}$. By Theorem 3.1, the behaviors of the A under D_1 and D_2 are identical. The behavior of A under D_3 is $|D_2 - D_3| = |D_1 - D_3|_{(2q)}$-close to that under D_2, and hence D_1. □

4 Semi-Constant Distributions

In this section, we define a class of distributions on oracles in $\mathcal{H}_{\mathcal{X},\mathcal{Y}}$ called semi-constant distributions. Our motivation for these distributions is to support quantum Full Domain Hash-type arguments, where a random value is inserted into a small but significant fraction of oracle inputs. The following definition captures the essence of the this idea:

Definition 4.1. *Define SC_λ^n, the order n semi-constant distribution, as the distribution over $\mathcal{H}_{\mathcal{X},\mathcal{Y}}$ resulting from the following process:*

- *First, for each $i \in \{1, ..., n\}$, pick a random element y_i from $\mathcal{Y}$.*
- *For each $x \in \mathcal{X}$, do one of the following:*
  - *With probability λ, pick a random $i \in \{1, ..., n\}$ and set $H(x) = y_i$. Call this input a distinguish input to H.*
  - *Otherwise, set $H(x)$ to be a random element in $\mathcal{Y}$.*

We now follow the ideas of Section 3 and show that $|\mathsf{SC}_\lambda^n - U|_{(2q)}$ is small, where U is the uniform distribution. Then, by Corollary 3.3, a quantum adversary cannot distinguish SC_λ^n from U. We start with a general theorem:

Theorem 4.2. *Fix k. Suppose we have a family of distributions D_λ over $\mathcal{H}_{\mathcal{X},\mathcal{Y}}$ parametrized by $\lambda \in [0, 1]$. Let $\mathcal{P}$ be the collection of probabilities for the marginal distributions over all sets of k inputs. Suppose there are integers d and Δ such that for each $p \in \mathcal{P}$:*

- *p is a polynomial in λ of degree at most d.*
- *The λ^j coefficient of p is 0 for each $j \in \{1, ..., \Delta\}$.*

Then $|D_\lambda - D_0|_{(k)} < 4\zeta(2\Delta + 2)\lambda^{\Delta+1}(d - \Delta)^{2\Delta+2}$ where ζ denotes the Riemann Zeta function.

Before proving this theorem, we show how to apply it to SC_λ^n:

Lemma 4.3. *Fix k. The probabilities in each of the marginal distributions of SC_λ^n over k inputs are polynomials in lambda of degree k such that the λ^1 coefficient is 0.*

This, proved in Appendix D, shows that for any k, we can set $d = k$ and $\Delta = 1$. Let $k = 2q_H$, and recall that SC_0^n is the uniform distribution. Given that $\zeta(4) = \pi^4/90$, we get:

Corollary 4.4. *The distribution of outputs of a quantum algorithm making q_H queries to an oracle drawn from SC_λ^n is at most a distance $\ell(q_H)\lambda^2$ away from the case when the oracle is drawn from the uniform distribution, where $\ell(q_H) = 4\pi^4(2q_H - 1)^4/90 \approx 69q_H^4$*

We note that using standard quantum query results, it is possible to prove this corollary for a distance of $ak^b\lambda^1$. However, as we will explain in Section 5, λ^1 is not sufficient for our purposes.

We do not know if this bound is tight. In Appendx D, we adapt the collision search algorithm of Brassard et al. [BHT97] to SC^1_λ, and demonstrate that $\ell(q_H) = \Omega(q_H^3)$.

It now remains to prove Theorem 4.2:

Proof of Theorem 4.2.

Suppose we have a family of distributions D_λ for $\lambda \in [0,1]$ over $\mathcal{H}_{\mathcal{X},\mathcal{Y}}$ meeting the criteria of Theorem 4.2. That is, let $\mathcal{P}$ be the collection of probabilities for the marginal distributions over all sets of k inputs. There are integers d and Δ such that for each $p \in \mathcal{P}$, p is a polynomial in λ of degree at most d and the λ^j coefficient is 0 for $j \in \{1, ..., \Delta\}$.

Our goal is to find a D'_λ such that $|D'_\lambda - D_0| < \epsilon = 4\zeta(2\Delta+2)\lambda^{\Delta+1}(d-\Delta)^{2\Delta+2}$ and that is k-wise equivalent to D_λ. This will show that $|D_\lambda - D_0|_{(k)} < \epsilon$.

Let $p(\lambda) \in \mathcal{P}$. Let $s(\lambda) = \frac{p(\lambda)-p(0)}{\lambda^{\Delta+1}}$. $s(\lambda)$ is then a $d-\Delta-1$ degree polynomial. Fix $d-\Delta$ different values $\lambda_i \in (0,1]$ for $i \in \{1, ..., d-\Delta\}$, and let $s_i = s(\lambda_i)$. Then $s(\lambda)$ is the unique $d-\Delta-1$ degree polynomial satisfying $s(\lambda_i) = s_i$. Therefore, we can interpolate the pairs (λ_i, s_i) using Lagrange polynomials:

$$s(\lambda) = \sum_{i=1}^{d-\Delta} s_i \ell_i(\lambda) = \sum_{i=1}^{d-\Delta} s(\lambda_i)\ell_i(\lambda)$$

Where $\ell_i(\lambda)$ is the Lagrange polynomial

$$\ell_i(\lambda) = \prod_{j=1, j\neq i}^{d-\Delta} \left(\frac{\lambda - \lambda_j}{\lambda_i - \lambda_j}\right)$$

Then we get

$$p(\lambda) \quad = \quad p(0) + \lambda^{\Delta+1} \sum_{i=1}^{d-\Delta} \frac{p(\lambda_i) - p(0)}{\lambda_i^{\Delta+1}} \ell_i(\lambda) = \left(1 - \sum_{i=1}^{d-\Delta} a_i(\lambda)\right) p(0) + \sum_{i=1}^{d-\Delta} a_i(\lambda) p(\lambda_i)$$

Where

$$a_i(\lambda) = \left(\frac{\lambda}{\lambda_i}\right)^{\Delta+1} \ell_i(\lambda) = \left(\frac{\lambda}{\lambda_i}\right)^{\Delta+1} \prod_{j=1, j\neq i}^{d-\Delta} \left(\frac{\lambda - \lambda_j}{\lambda_i - \lambda_j}\right)$$

Notice that $a_i(\lambda)$ does not depend in any way on p, so we can choose the same λ_i and $a_i(\lambda)$ for each probability in each marginal distribution of k variables. Thus, we can define

$$D'_\lambda = \left(1 - \sum_{i=1}^{d-\Delta} a_i(\lambda)\right) D_0 + \sum_{i=1}^{d-\Delta} a_i(\lambda) D_{\lambda_i}$$

The sum of the weights for D_0 and D_{λ_i} add up to 1, so this is a valid weight assignment. Moreover, by construction, all of the polynomials for the marginals of k inputs are identical to those of D_λ. Thus D'_λ is k-wise similar to D_λ. Therefore,

$$|D_\lambda - D_0|_{(k)} \leq |D'_\lambda - D_0| \leq \sum_{i=1}^{d-\Delta} |a_i(\lambda)| + \left|\sum_{i=1}^{d-\Delta} a_i(\lambda)\right| \leq 2 \sum_{i=1}^{d-\Delta} |a_i(\lambda)|$$

Thus to prove the theorem, we find λ_i to make this quantity small:

Claim 1. *If we set* $\lambda_i = \left(\frac{i}{d-\Delta}\right)^2$ *for* $i \in \{1, ..., d-\Delta\}$*, then as long as* $\lambda \leq \lambda_1 = 1/(d-\Delta)^2$*, we have that* $\sum_i |a_i(\lambda)| < 2\zeta(2\Delta+2)(d-\Delta)^{2\Delta+2}\lambda^{\Delta+1}$

This is proved in Appendix C and proves the theorem for the case $\lambda \leq \left(\frac{1}{d-\Delta}\right)^2$. If $\lambda > \left(\frac{1}{d-\Delta}\right)^2$, then we are looking for a D'_λ such that $|D'_\lambda - D_0|$ is less than

$$4\zeta(2\Delta+2)\lambda^{\Delta+1}(d-\Delta)^{2\Delta+2} > 4\zeta(2\Delta+2) > 4$$

Since two distributions can have a distance at most 2, D_λ itself satisfies the conditions for D'_λ in this case. This complete the proof.

□

5 Quantum Security Arguments

In this section, we explore the random oracle proof technique of Bellare and Rogaway [BR93]. In this technique, we plug a challenge c into a randomly chosen hash query, and hope that the adversary uses c to break the system. If so, we can use the adversary to solve the underlying problem.

In the quantum setting, the straight-forward application of this technique breaks down. The adversary can now query the hash function on a superposition of exponentially many inputs. If we plug c to one of those inputs, the probability that the adversary uses c is exponentially small.

We instead plug c into some small but significant fraction of the inputs, so that the oracle is distributed according to SC^1_λ. As we have shown, a quantum algorithm cannot detect that this oracle is not random, but now the adversary uses c with significant probability.

5.1 Identity-Based Encryption from Lattices

Here we prove the security of the IBE scheme from Gentry et al. [GPV08]. Their scheme is constructed from an encryption scheme $\mathsf{E} = (\mathsf{E.Gen}, \mathsf{E.Enc}, \mathsf{E.Dec})$, for which there exists a trapdoor allowing the computation of secret keys from public keys.

More specifically, let $\mathsf{F} = (\mathsf{F.Gen}, \mathsf{F.Sample}, f, f^{-1})$ be a pre-image sampleable function (PSF). Suppose $\mathsf{E.Gen}(1^n)$ works as follows: generate $(\mathsf{msk}, \mathsf{mpk}) \leftarrow \mathsf{F.Gen}(1^n)$. Then, sample $\mathsf{sk} \leftarrow \mathsf{F.Sample}(1^n)$, and compute $\mathsf{pk} = f_{\mathsf{mpk}}(\mathsf{sk})$. Output $(\mathsf{sk}, (\mathsf{pk}, \mathsf{mpk}))$.

Gentry et al. give such an encryption scheme based on the hardness of lattice problems. Their security reduction treats the adversary as a black box, so their proof holds in the quantum setting. They then prove the security of the IBE scheme $\mathsf{IBE} = (\mathsf{IBE.Gen} = \mathsf{F.Gen}, \mathsf{IBE.Extract}^H, \mathsf{IBE.Enc}^H, \mathsf{IBE.Dec})$ where H maps identities to public keys of E and:

- $\mathsf{IBE.Extract}_{\mathsf{msk}}(\mathsf{id})^H = f^{-1}_{\mathsf{msk}}(H(\mathsf{id}))$
- $\mathsf{IBE.Enc}_{\mathsf{mpk}}(\mathsf{id}, m)^H = \mathsf{E.Enc}_{H(\mathsf{id})}(m)$
- $\mathsf{IBE.Dec}_{\mathsf{sk}_{\mathsf{id}}}(c) = \mathsf{E.Dec}_{\mathsf{sk}_{\mathsf{id}}}(c)$

Theorem 5.1. *Let* E *and* F *be as above, and suppose that* E *is quantum IND-CPA-secure. If we model* H *as a random oracle, then* IBE *is quantum IND-ID-CPA-secure.*

Proof. Let A be a quantum adversary making q_H hash queries, q_E extract queries queries, that breaks IBE with advantage ϵ. Assume without loss of generality that A never produces an identity it already received a secret key for.

Let Game_0 be the standard attack game for IBE: the challenger generates $(\mathsf{msk}, \mathsf{mpk})$ from $\mathsf{IBE.Gen}$, and sends mpk to the adversary. The adversary can make (classical) extraction queries on identities id_i, to which the challenger responds with $\mathsf{IBE.Extract}^H_{\mathsf{msk}}(\mathsf{id}_i)$, and (quantum) hash queries to the random oracle H. A then produces an identity id, along with messages m_0 and m_1. The challenger chooses a random bit b, and responds with $\mathsf{IBE.Enc}_{\mathsf{mpk}}(\mathsf{id}, m_b)$. A is allowed to make more extraction and hash queries, and produces a bit b'. A wins if $b' = b$ and for all i, $\mathsf{id}_i \neq \mathsf{id}$. By definition, this happens with probability $\frac{1}{2} + \epsilon$.

Let $\lambda \in (0, 1)$ to be chosen later. Let $\mathcal{M}$ be the identity space. Let Game_1 be Game_0, except that we choose a subset $\mathcal{X}$ of $\mathcal{M}$ as follows: for each $\mathsf{id} \in \mathcal{M}$, we put id in $\mathcal{X}$ with probability λ. If A asks for a secret key for an identity in $\mathcal{X}$, abort, and flip a coin to decide if A wins. If the identity generated by A is *not* in $\mathcal{X}$, also abort and flip a coin to decide if A wins. Otherwise, follow the criteria in Game_0 to decide if A wins. The identity A generated must be distinct from the identities it received secret keys for, so the probabilities that each of these identities are in $\mathcal{X}$ is independent. Therefore, the no-abort conditions are met with probability at least $\lambda(1 - \lambda q_E)$. Thus, A wins in Game_1 with probability at least $\frac{1}{2} + \lambda(1 - \lambda q_E)\epsilon$.

Let Game_2 be Game_1, except that we choose a random pk in the public key space of E, and set $H(x) = pk$ for all x in $\mathcal{X}$. H is now distributed according to SC^1_λ, so by Corollary 4.4, the probability A wins in Game_2 is at least $\frac{1}{2} + \lambda(1 - \lambda q_E)\epsilon - \ell(q_H)\lambda^2$.

Now we describe an adversary B that breaks E. Assume B has quantum access to two random oracles O_1 and O_2. O_1 maps identities to randomness used by Sample. O_2 maps identities to bits, outputting 1 with probability is λ. In Section 6, we will show how to efficiently simulate these oracles. On input $(\mathsf{mpk}, \mathsf{pk})$, B works as follows:

- Send mpk to A and simulate A, playing the role of challenger to A.
- Construct a (quantum) oracle H such that

$$H(\mathsf{id}) = \begin{cases} \mathsf{pk} & \text{if } O_2(\mathsf{id}) = 1 \\ f_{\mathsf{mpk}}(\mathsf{Sample}(; O_1(\mathsf{id}))) & \text{otherwise} \end{cases}$$

  where $\mathsf{Sample}(; r)$ means run Sample with randomness r.
- When A asks for the secret key for id_i, compute $O_2(\mathsf{id}_i)$. If the result is 1, output a random bit and abort. Otherwise, respond with $\mathsf{sk}_i = \mathsf{Sample}(; O_1(\mathsf{id}_i))$.
- When A produces the challenge query $(\mathsf{id}^*, m_0, m_1)$, check if $O_2(\mathsf{id}^*) = 1$. If so, send (m_0, m_1) to B's challenger. Otherwise, output a random bit and abort.
- When the challenger responds with a ciphertext c, send c to A.
- When A outputs a bit b', output the same bit.

Let $\mathcal{X}$ be the set of identities id such that $O_2(\mathsf{id}) = 1$. We can then see that the abort conditions are equivalent to Game_2. For each extract query id_i that succeeds, we have that $O_2(\mathsf{id}_i) = 0$, so

$$f_{\mathsf{mpk}}(\mathsf{sk}_i) = f_{\mathsf{mpk}}(\mathsf{Sample}(; O_1(\mathsf{id}_i))) = H(\mathsf{id}_i)$$

Thus sk_i is a correct secret key for id_i, and it is distributed correctly (since it is a random pre-image of $H(\mathsf{id}_i)$). If B does not abort on the challenge query, it means $O_2(\mathsf{id}^*) = 1$, so $H(\mathsf{id}^*) = \mathsf{pk}$. Also, we can see that H is distributed according to SC^1_λ. Therefore, if B aborts, we win with probability $\frac{1}{2}$, and if we do not abort, we win if and only if A wins. Thus, the view of A as a subroutine of B is the same as in Game_2, and B wins with the same probability that A does. Therefore, B wins with probability at least $\frac{1}{2} + \lambda(1 - \lambda q_E)\epsilon - \ell(q_H)\lambda^2$.

The advantage of B is at least

$$\lambda(1 - \lambda q_E)\epsilon - \ell(q_H)\lambda^2 = \lambda\epsilon - (\ell(q_H) + \epsilon q_E)\lambda^2$$

We can now choose λ. The maximum occurs when $\lambda = \epsilon/2(\ell(q_H) + q_E\epsilon)$, which gives

$$Adv_B \geq \frac{\epsilon^2}{4(\ell(q_H) + \epsilon q_E)} \geq \frac{\epsilon^2}{4(\ell(q_H) + q_E)}$$

□

The proof of Theorem 5.1 demonstrates why a bound of $aq_H^b\lambda$ for the probability of distinguishing SC^n_λ from a truly random oracle is insufficient. In this case, in Game_2, the success probability will be at least $\frac{1}{2} + \lambda(1 - \lambda q_E)\epsilon - aq_H^b\lambda < \frac{1}{2} + \lambda(\epsilon - aq_H^b)$. For $\epsilon < aq_H^b$ (which is guaranteed for large enough q_H since $\epsilon < 1$), this bound is below $\frac{1}{2}$ and thus useless for trying to prove security.

5.2 Hierarchical IBE from Lattices

In this section, we show the general idea behind adapting the techniques above to proving the security of the hierarchical identity-based encryption (HIBE) scheme of Agrawal et al. [ABB10] and Cash et al. [CHKP10].

In a HIBE scheme, identities are structured as a tree, with the identity of any node containing the identity of its parent as a proper prefix. Any node on the tree can produce private keys for any nodes in the subtree rooted at that node. We allow an adversary to adaptively take control of an arbitrary number of nodes in the tree (and thus the subtrees rooted there). An HIBE scheme is secure if the adversary cannot decrypt messages encrypted to an identity id^* of the adversaries choice but not under its control.

In [ABB10], the random oracle scheme has an oracle H that maps identities to some random quantities. The reduction has the following high-level structure:

- Guess which level w of the tree contains the identity id^*
- For each level i, generate some random quantities R_i.
- For each level i, simulate a separate random oracle for identities at that level. For i, guess which query number q_i will contain the hash of the level-i parent of id^*.
- Answer the jth random oracle query at level i on $\mathsf{id}_{i,j}$ as follows: if $j = q_i$, output R_i. Otherwise, output a random value.
- Answer secret key queries on id in some special way, but fail if for all prefixes id_i of id, $\mathsf{id}_i = \mathsf{id}_{i,q_i}$. That is, fail if $H(\mathsf{id}_i) = R_i$.
- When the adversary generates the identity id^*, we succeed if both the adversary succeeds and if id^* is at level w and all prefixes id^*_i of id^* satisfy $\mathsf{id}^*_i = \mathsf{id}_{i,q_i}$.

We now show how to prove security by repeatedly applying the arguments of Theorem 5.1. Basically, we iterate over levels i, and insert R_i into a λ_i fraction of identities at that level. In iteration i, we say the adversary wins if it won in the previous iteration, the level-i prefix of the chosen identity id^* is in the λ_i fraction of distinguished identities (that is, $H(\mathsf{id}_i^*) = R_i$), and no signature query is. If the iteration i advantage is ϵ_i, then using the same techniques as in Theorem 5.1, we can set λ_i so that

$$\epsilon_i \geq \frac{\epsilon_{i-1}^2}{4\ell(q_H) + q_E}$$

We will say that in iteration 0, the adversary wins if it normally would win and we guessed which level id^* belonged to correctly. That is, $\epsilon_0 = \epsilon/d$, where ϵ is the adversaries advantage in the standard game. This gives us a total advantage after iteration d of at least

$$\frac{(\epsilon/d)^{2^d}}{4(\ell(q_H) + q_E)^{(2^d-1)}} = 4(\ell(q_H) + q_E)\left(\frac{\epsilon}{4d(\ell(q_H) + q_E)}\right)^{2^d}$$

Notice that the dependence on d in doubly-exponential, whereas in the original scheme it was singly exponential. Thus, for the same security parameters, this proof only works for much smaller depth than the classical proof.

These techniques apply as well to the random oracle HIBE of Cash et al. [CHKP10], though their reduction is a bit more complicated, as there is a second random oracle G which needs to be handled in a similar way.

5.3 Signatures from Trapdoor Permutations

Here we discuss the security of the Full Domain Hash (FDH) signature scheme:

Definition 5.2 (FDH Signatures)**.** *Let* $\mathsf{F} = (\mathsf{F.Gen}, f, f^{-1})$ *be a trapdoor permutation, and a hash function* H *that maps messages to images of* f*, let* $\mathsf{S}^H = (\mathsf{S.Gen} = \mathsf{F.Gen}, \mathsf{S.Sign}^H, \mathsf{S.Ver}^H)$ *where:*

- $\mathsf{S.Sign}^H_{\mathsf{sk}}(m) = f^{-1}_{\mathsf{sk}}(H(m))$
- $\mathsf{S.Ver}^H_{\mathsf{pk}}(m, \sigma) = \begin{cases} \mathsf{accept} & \text{\textit{if} } f_{\mathsf{pk}}(\sigma) = H(m) \\ \mathsf{reject} & \text{\textit{ortherwise}} \end{cases}$

We now state the main theorem of this section:

Theorem 5.3. *Let* F *be a quantum one-way trapdoor permutation. If we model* H *as a random oracle, then* S *is quantum UF-CMA-secure*

The proof of this theorem is very similar to that of Theorem 5.1, and appears in Appendix E.

6 Simulating Random Oracles

In this last section, we explain how to efficiently simulate random oracles. All quantum random oracle proofs to date require the reduction algorithm to have quantum access to a random oracle, but a truly random oracle requires exponentially many bits of randomness to construct. We show that this is not a problem:

Theorem 6.1. *Any quantum algorithm A making quantum queries to random oracles O_i can be efficiently simulated by a quantum algorithm B, which has the same output distribution, but makes no queries.*

Proof. We construct an algorithm B which simulates A, and answers queries to oracle O_i with evaluations of efficient functions f_i. Boneh et al. [BDF$^+$11] use pseudorandom functions (PRF) for the f_i. At first glance, this seems like the only option, as we need a function f_i that cannot be distinguished from random.

Notice, however, that f_i need not be secure against all adversaries, just the adversary we are simulating. We know that our adversary makes q_i queries to oracle O_i, so it suffices to have f_i be PRFs secure for up to q_i quantum queries. In the classical setting, q_i-wise independent functions (functions that are q_i-wise equivalent to a random function) serve as *perfectly* secure PRFs for up to q_i classical queries. We could hope that something similar holds in the quantum world: indeed, according to Theorem 3.1, if f_i is $2q_i$-wise equivalent to a random function, then the behavior of our adversary is the same when the oracle is random and when it is f_i. Thus if the f_i are $2q_i$-wise independent, algorithm A, as a subroutine of B, behaves identically to the case where A is given truly random oracles. Hence, the output distribution of B is identical to that of A.

Efficient constructions of k-wise independent functions have been known for some time [Jof74, KM94], and they have been used extensively in the derandomization literature [Lub85, ABI86, KM93]. One common approach to construct a k-wise independent function f from $\mathcal{X}$ to $\mathcal{Y}$ is to assume that $N = |\mathcal{Y}|$ is a prime power and interpret $\mathcal{Y}$ as the field $\mathbb{F}_N$. Then define a matrix C with the following properties:

- The entries are elements in $\mathbb{F}_N$.
- There are $|\mathcal{X}|$ rows and some small number r of columns.
- Each subset of k rows is linearly independent.

One such example is the Vandermonde matrix, which is used by Alon et al.[ABI86]. To define the function f, we then pick a random vector v in $\mathbb{F}_N^r$. $f(x)$ is then the xth element of of the vector Cv. Since any k rows of C are linearly independent, any k elements of Cv are independent, and hence f is k-wise independent. The key to making this efficient is that to compute $f(x)$, we only need the xth row of C, which we can compute on the fly. □

Hence, we can simulate O_1 from Theorem 5.1. To simulate O_2, which outputs a bit such that $O_2(x) = 1$ with probability λ, approximate λ by some rational number a/b where b is a prime power, and construct a k-wise independent function f' with range $\mathcal{Y} = \{1, ..., b\}$. Then set

$$f(x) = \begin{cases} 1 & \text{if } f'(x) \leq a \\ 0 & \text{otherwise} \end{cases}$$

7 Conclusion

We have shown how to adapt certain classical random oracle arguments to the quantum random oracle model. Specifically, we gave quantum security proofs for the IBE scheme of Gentry et al. [GPV08] and the Full Domain Hash signature scheme. We achieved this by defining a distribution of oracles, called semi-constant distributions, and showing that such oracles cannot be distinguished from a random oracle by a quantum adversary. We also show how these techniques can be applied to the random oracle HIBE schemes of Cash et al. [CHKP10] and Agrawal et al. [ABB10]. Lastly, we have shown how to remove the need for quantum-secure pseudorandom functions from prior work.

Although we have made progress toward converting classical random oracle proofs into quantum proofs, there is still much work to be done. For example, the construction of signatures from identification protocols by Fiat and Shamir [FS87], though similar to the proofs in this paper, still needs a quantum proof. The difficulty in the Fiat-Shamir reduction is that the plugging step initiates the underlying identification protocol, and there is no obvious quantum analog for this strategy. Also, different types of security arguments, such as Fujisaki and Okamoto's generic conversion of weakly secure encryption schemes into a CCA-secure encryption scheme [FO99], still lack a quantum proof of security. Lastly, while quantum-secure PRF's are no longer necessary for proving security in the quantum random oracle model, their construction from other primitives still remains an interesting open problem.

References

[Aar09] Scott Aaronson. Quantum copy-protection and quantum money. *Computational Complexity, 2009. CCC'09. 24th*, 2009.

[ABB10] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. *Advances in Cryptology - CRYPTO 2010*, pages 98–115, 2010.

[ABI86] Noga Alon, Lasz Babai, and A. Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *Journal of algorithms*, 7(4):567–583, 1986.

[BBBV97] C.H. Bennett, Ethan Bernstein, Gilles Brassard, and U. Vazirani. Strengths and weaknesses of quantum computing. *Arxiv preprint quant-ph/9701001*, 1997.

[BDF$^+$11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In *Advances in Cryptology - AsiaCrypt 2011*, 2011.

[BF01] Dan Boneh and Matthew Franklin. Identity-based encryption from the Weil pairing. *Advances in Cryptology - CRYPTO 2001*, 32(3):586–615, 2001.

[BF11] Dan Boneh and David Freeman. Homomorphic signatures for polynomial functions. *Advances in Cryptology - EUROCRYPT 2011*, 6632:149–168, 2011.

[BHK$^+$11] Gilles Brassard, Peter Høyer, Kassem Kalach, Marc Kaplan, Sophie Laplante, and Louis Salvail. Merkle Puzzles in a Quantum World. *Advances in Cryptology - CRYPTO 2011*, pages 391–410, 2011.

[BHT97] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum algorithm for the collision problem. *ACM SIGACT News (Cryptology Column)*, 28:14–19, 1997.

[BR93] Mihir Bellare and Philip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM conference on Computer and communications security*, number November 1993, pages 62–73. ACM, 1993.

[BS08] Gilles Brassard and Louis Salvail. Quantum Merkle Puzzles. *Second International Conference on Quantum, Nano and Micro Technologies (ICQNM 2008)*, pages 76–79, February 2008.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. *Advances in Cryptology - EUROCRYPT 2010*, pages 523–552, 2010.

[FO99] E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Advances in Cryptology - CRYPTO 1999*, pages 79–79. Springer, 1999.

[FS87] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology - Crypto 1986*, pages 186–194. Springer, 1987.

[GKV10] S. Gordon, Jonathan Katz, and Vinod Vaikuntanathan. A Group Signature Scheme from Lattice Assumptions. *Advances in Cryptology - ASIACRYPT 2010*, pages 395–412, 2010.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of computer and system sciences*, pages 270–299, 1984.

[GMR88] S. Goldwasser, S. Micali, and R.L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. *Proceedings of the fourtieth annual ACM symposium on Theory of computing - STOC '08*, page 197, 2008.

[HSS11] Sean Hallgren, Adam Smith, and Fang Song. Classical cryptographic protocols in a quantum world. *Advances in CryptologyâĂŞCRYPTO 2011*, pages 411–428, 2011.

[Jof74] A. Joffe. On a set of almost deterministic k-independent random variables. *the Annals of Probability*, 2(1):161–162, 1974.

[KM93] D. Koller and N. Megiddo. Constructing small sample spaces satisfying given constraints. In *Proceedings of the ACM Symposium on Theory of Computing*, pages 268–277. ACM, 1993.

[KM94] Howard Karloff and Yishay Mansour. On Construction of k -wise Independent Random Variables. *Proceedings of the ACM Symposium on Theory of Computing*, 17:564–573, 1994.

[Lub85] Michael Luby. A simple parallel algorithm for the maximal independent set problem. In *Proceedings of the ACM symposium on Theory of computing*, pages 1–10. ACM, 1985.

[MP11] David Meyer and James Pommersheim. On the uselessness of quantum queries. *Theoretical Computer Science*, (March):1–12, 2011.

[NC00] Michael A. Nielsen and Isaac Chuang. Quantum Computation and Quantum Information. *American Journal of Physics*, 70(5):558, 2000.

[Sho97] P.W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[Unr10] Dominique Unruh. Universally composable quantum multi-party computation. *Advances in CryptologyâĂŞEUROCRYPT 2010*, pages 486–505, 2010.

A Cryptographic Primitives

Definition A.1 (Encryption Scheme)**.** *A public key encryption scheme* E *is a triple of PPT algorithms* $(\mathsf{E.Gen}, \mathsf{E.Enc}, \mathsf{E.Dec})$ *where*

- $\mathsf{E.Gen}(1^n)$ *generates a secret/public key pair* $(\mathsf{sk}, \mathsf{pk})$*.*
- $\mathsf{E.Enc}_{\mathsf{pk}}(m)$ *computes a ciphertext* c*.*
- $\mathsf{E.Dec}_{\mathsf{sk}}(c)$ *computes the plaintext* m*, such that* $\mathsf{E.Dec}_{\mathsf{sk}}(\mathsf{E.Enc}_{\mathsf{pk}}(m)) = m$*.*

We use the standard chosen plaintest attack (CPA) game [GM84] to model security:

- The challenger generates $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{E.Gen}(1^n)$, and sends pk to the adversary A.
- A generates the challenge plaintext (m_0, m_1).
- The challenger chooses a bit b, and responds with the challenge ciphertext $c = \mathsf{E.Enc}_{\mathsf{pk}}(m_b)$.
- A makes a guess b' for b.

The advantage of A is ϵ, where $\frac{1}{2} + \epsilon$ is the probability that $b = b'$. We say that E is (quantum) IND-CPA-secure if all efficient (quantum) algorithms have only negligible advantage (in n).

An identity-based encryption (IBE) scheme IBE is an encryption scheme where identities serve as public keys, and a trusted authority hands out the corresponding secret keys. In particular:

Definition A.2 (IBE Scheme)**.** *An identity-based encryption (IBE) scheme* IBE *is a quadruple of PPT algorithms* $(\mathsf{IBE.Gen}, \mathsf{IBE.Extract}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec})$ *where*

- $\mathsf{IBE.Gen}(1^n)$ *generates a master secret/public key pair* $(\mathsf{msk}, \mathsf{mpk})$*.*
- *The trusted authority uses* $\mathsf{IBE.Extract}_{\mathsf{msk}}(\mathsf{id})$ *to compute a secret key* $\mathsf{sk}_{\mathsf{id}}$ *corresponding to the identity* id*.*
- $\mathsf{IBE.Enc}_{\mathsf{mpk}}(\mathsf{id}, m)$ *encrypts* m *to identity* id*.*
- $\mathsf{IBE.Dec}_{\mathsf{sk}_{\mathsf{id}}}(c)$ *decrypts* c*, such that*

$$\mathsf{IBE.Dec}_{\mathsf{IBE.Extract}_{\mathsf{msk}}(\mathsf{id})}(\mathsf{IBE.Enc}_{\mathsf{mpk}}(\mathsf{id}, m)) = m$$

We use the standard CPA attack game for IBE schemes, where the adversary can choose the identity id for which it will attempt to decrypt, and can get the secret keys corresponding to any other identities [BF01]. Specifically,

- The challenger generates $(\mathsf{msk}, \mathsf{mpk}) \leftarrow \mathsf{IBE.Gen}(1^n)$, and sends mpk to the adversary A.
- The adversary can make extraction queries on identities id_i, to which the challenger responds with $\mathsf{IBE.Extract}_{\mathsf{msk}}(\mathsf{id}_i)$.
- The adversary generates an identity id, and two messages (m_0, m_1).
- The challenger chooses a random bit b, and responds with $\mathsf{IBE.Enc}_{\mathsf{id}}(m_b)$.
- A makes a guess b' for b.

The advantage of A is ϵ, where $\frac{1}{2} + \epsilon$ is the probability that $\mathsf{id} \neq \mathsf{id}_i$ for any i and $b = b'$. We say that IBE is (quantum) IND-ID-CPA-secure if all efficient (quantum) algorithms have only negligible advantage (in n).

Definition A.3 (Signature Scheme)**.** *A signature scheme* S *is a triple of PPT algorithms* $(\mathsf{S.Gen}, \mathsf{S.Sign}, \mathsf{S.Ver})$ *where*

- $\mathsf{S.Gen}(1^n)$ *generates a secret/public key pair* $(\mathsf{sk}, \mathsf{pk})$*.*
- $\mathsf{S.Sign}_{\mathsf{sk}}(m)$ *computes a signature* σ *on the message* m*.*
- $\mathsf{S.Ver}_{\mathsf{pk}}(m, \sigma)$ *outputs* accept *or* reject*, such that* $\mathsf{S.Ver}_{\mathsf{pk}}(m, \mathsf{S.Sign}_{\mathsf{sk}}(m)) = \mathsf{accept}$*.*

We will use the standard chosen message attack (CMA) game to define security [GMR88]:

- The challenger generates $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{S.Gen}(1^n)$, and sends pk to the adversary A.
- A can make signature queries on messages m_i, to which the challenger responds with $\mathsf{S.Sign}_{\mathsf{sk}}(m_i)$.
- A produces a forgery candidate (m, σ).

The advantage of A is the probability that $m \neq m_i$ for any i and $\mathsf{Ver}_{\mathsf{pk}}(m, \sigma) = \mathsf{accept}$. We say that S is (quantum) UF-CMA-secure if all efficient (quantum) adversaries have negligible advantage (in n).

Definition A.4 (PSF)**.** *A pre-image sampleable function (PSF) is a quadruple of algorithms* $\mathsf{F} = (\mathsf{F.Gen}, \mathsf{F.Sample}, f, f^{-1})$ *where:*

- $\mathsf{F.Gen}(1^n)$ *generates a secret/public key pair* $(\mathsf{sk}, \mathsf{pk})$*.*
- f_{pk} *is a function.*
- $\mathsf{F.Sample}$ *samples* x *from a distribution* D *such that* $f_{\mathsf{pk}}(x)$ *is uniform.*
- $f_{\mathsf{sk}}^{-1}(y)$ *samples from* D *conditioned on* $f_{\mathsf{pk}}(x) = y$*.*

We model security with the following game:

- The challenger generates $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{F.Gen}(1^n)$, and sends pk to the adversary A. It also sends $y = f_{\mathsf{pk}}(\mathsf{F.Sample}())$ to A.
- A makes a guess x.

The advantage of A is the probability that $f_{\mathsf{pk}}(x) = y$. We say that F is (quantum) one-way if all efficient (quantum) algorithms have negligible advantage.

A trapdoor permutation (TDP) is a special case of a PSF where f is bijective and $\mathsf{F.Sample}$ simply returns a random element in the domain of f_{pk}. Since $\mathsf{F.Sample}$ is already defined, we omit it from the specification.

B Quantum Computation

We give some background on quantum computation. For a more thorough discussion, please see [NC00]. We also prove Theorem 3.1, which states that the output distribution of a quantum algorithm making q_H queries to a quantum oracle drawn from some distribution D depends only on the marginal distributions of D on all subsets of $2q$ inputs.

Let $\mathcal{H}$ be a Hilbert space with an inner product $\langle\cdot|\cdot\rangle$, and let $B = \{|x\rangle\}$ be an orthonormal basis for $\mathcal{H}$, index by a parameter $x \in \mathcal{X}$. The state of a quantum system on $\mathcal{H}$ is specified by a vector $|\phi\rangle \in \mathcal{H}$ of norm 1.

Quantum Measurement Given a state $|\phi\rangle$, we can measure $|\phi\rangle$ in the basis B, obtaining the value $x \in \mathcal{X}$ with probability $|\langle x|\phi\rangle|^2$. Thus, to each $|\phi\rangle$, we associate a distribution D_ϕ where $D_\phi(x) = |\langle x|\phi\rangle|^2$. The normalization constant and the fact that B is an orthonormal basis ensure that this is in fact a valid distribution. After measurement, the system is in state $|x\rangle$.

If $B = \{|x, y\rangle\}$ for $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, then we can also perform a partial measurement over $\mathcal{X}$. The distribution over $\mathcal{X}$ is the marginal distribution of D_ϕ when restricted to $\mathcal{X}$. That is, we obtain the value x with probability $\sum_{y\in\mathcal{Y}} |\langle x, y|\phi\rangle|^2$. The resulting state is the result of projecting $|\phi\rangle$ to the space spanned by $\{|x, y\rangle\}$ for $y \in \mathcal{Y}$, and renormalizing so that the norm is 1.

Quantum Algorithms A quantum algorithm A over a Hilbert space $\mathcal{H}$ with an orthonormal basis B is specified by unitary transformation U. The input to A is an element $x_0 \in X$. The system is initialized to the basis state $|x_0\rangle$, and U is applied to the system, obtaining the final state $|\phi\rangle = U|x_0\rangle$. Then the state is sampled according to the distribution D_ϕ.

Let $\mathcal{X}$ and $\mathcal{Y}$ be sets such that $\mathcal{Y}$ is a commutative group with addition operation $\oplus$. For notational convenience, we assume that every element has order at most 2 ($y \oplus y$ is the identity for all $y \in \mathcal{Y}$). Given a function $H : \mathcal{X} \to \mathcal{Y}$ and third set $\mathcal{Z}$, define the orthonormal basis B as the set $\{|x, y, z\rangle\}$ for $x \in \mathcal{X}$, $y \in \mathcal{Y}$, and $z \in \mathcal{Z}$. Define the unitary transformation $H_{\texttt{trans}}$ over the Hilbert space spanned by B as the transformation that takes $|x, y, z\rangle$ into $|x, y \oplus H(x), z\rangle$. $H_{\texttt{trans}}$ is unitary, it's own inverse, and Hermitian. As an abuse of notation, we will frequently use $H_{\texttt{trans}}$ and H interchangeably.

A quantum algorithm A making q quantum queries to H is then specified by a sequence of unitary transformations $U_0, ..., U_q$. The evaluation of A then consists of alternately applying U_i and H to the initial state $|x\rangle$. We call

$$U_{i-1}H...U_1HU_0|x\rangle$$

the state of A before the ith query, and

$$HU_{i-1}H...U_1HU_0|x\rangle$$

the state after the ith query. The final state of the algorithm is

$$U_qH...U_1HU_0|x\rangle$$

We say that a quantum algorithm is efficient if q is a polynomial, and all the U_i are composed of a polynomial number of universal basis gates (the Hadamard, CNOT, and phase shift gates are commonly used).

Classical queries to an oracle are made by partial sampling over $\mathcal{X}$ before performing the query.

Density Matrix Suppose we have quantum states $|\phi_w\rangle$ for $w \in \mathcal{W}$, and we have a weight assignment D on $\mathcal{W}$. Then we define the density matrix

$$\rho = \sum_{w\in\mathcal{W}} D(x)|\phi_w\rangle\langle\phi_w|$$

The weight of an outcome y from measurement is given by $\langle y|\rho|y\rangle$. If D is a distribution, then this weight assignment is also a distribution, and the weight corresponds to the probability of obtaining y if we first pick w with probability $D(w)$, and then sample $|\phi_w\rangle$. In this case, the density matrix ρ summarizes the statistical behavior of the quantum system.

Suppose we have two weight assignments D_1 and D_2 on $\mathcal{W}$. Let D_1' and D_2' be the corresponding weight assignments of outcomes of measurement. Since the weight assignment induced by each $|\phi_w\rangle$ is in fact a distribution, $|D_1' - D_2'| \leq |D_1 - D_2|$.

For this paper, we will focus on the case where $\mathcal{W}$ is the set of oracles H from $X \to \mathcal{Y}$, and $|\phi_H\rangle$ is the final state of some quantum algorithm A that makes q queries to the oracle H.

B.1 Proof of Theorem 3.1

Proof of Theorem 3.1.

Let A be some quantum oracle algorithm. Suppose right before the first query, the state of the system is $|\phi_0\rangle = \sum_{xyz} \alpha_{xyz}|xyz\rangle$ (which is independent of the oracle). Assume for notational convenience that all the transition matrices U_i are identical and equal to U (the proof in the general case is essentially identical). Let $|\phi_{q,H}\rangle$ be the state of A after q queries to oracle H. That is,

$$|\phi_{q,H}\rangle = UHU....UH|\phi_0\rangle$$

Let ρ_q be the density matrix for A after q queries when the oracle H is drawn from a distribution D:

$$\rho_q = \sum_H \Pr_D[H]|\phi_{q,H}\rangle\langle\phi_{q,H}| = \sum_H \Pr_D[H]UHU...UH|\phi_0\rangle\langle\phi_0|HU^*...U^*HU^*$$

Observe that

$$\begin{array}{rcl} (UH)_{xyzx'y'z'} & = & \langle xyz|UH|x'y'z'\rangle \\ & = & \langle xyz|U|x'y'\oplus H(x')z'\rangle = U_{xyz,x'y'\oplus H(x')z'} \end{array}$$

We can now evaluate ρ_q component-wise:

$$\begin{aligned}(\rho_q)_{xyz,x'y'z'} &= \sum_H \Pr_D[H](UHU...UH|\phi_0\rangle\langle\phi_0|HU^*...U^*HU^*)_{xyz,x'y'z'} \\ &= \sum_H \Pr_D[H]((UH)(UH)...(UH)|\phi_0\rangle\langle\phi_0|(UH)^*...(UH)^*(UH)^*)_{xyz,x'y'z'} \\ &= \sum_H \Pr_D[H] \sum_{x_q y_q z_q} \sum_{x'_q y'_q z'_q} \\ &\qquad U_{xyzx_q y_q \oplus H(x_q) z_q}((UH)...(UH)|\phi_0\rangle\langle\phi_0|(UH)^*...(UH)^*)U^*_{x'y'z'x'_q y'_q \oplus H(x'_q) z'_q} \\ &\vdots \\ &= \sum_H \Pr_D[H] \sum_{x_q y_q z_q} \sum_{x'_q y'_q z'_q} \cdots \sum_{x_1 y_1 z_1} \sum_{x'_1 y'_1 z'_1} \\ &\qquad U_{xyzx_q y_q \oplus H(x_q) z_q} U_{x_q y_q z_q x_{q-1} y_{q-1} \oplus H(x_{q-1}) z_{q-1}} ... U_{x_2 y_2 z_2, x_1 y_1 \oplus H(x_1) z_1} \alpha_{x_1 y_1 z_1} \\ &\qquad \alpha^*_{x'_1 y'_1 z'_1} U^*_{x'_2 y'_2 z'_2, x'_1 y'_1 \oplus H(x'_1) z'_1} ... U^*_{x'_q y'_q z'_q x'_{q-1} y'_{q-1} \oplus H(x'_{q-1}) z'_{q-1}} U^*_{x'y'z'x'_q y'_q \oplus H(x'_q) z'_q}\end{aligned}$$

Now we can rearrange the order of summation as

$$\sum_{x_q y_q z_q} \sum_{x'_q y'_q z'_q} \cdots \sum_{x_1 y_1 z_1} \sum_{x'_1 y'_1 z'_1} \sum_H \Pr_D[H]$$

Next, notice that the summand only depends on $H(x_i), H(x'_i)$ for $i \in \{1, ..., q\}$. This means, in the H sum, we can sum out all the inputs x for which $x \neq x_i, x'_i$. Letting $r_i = H(x_i), r'_i = H(x'_i)$, we have that the summation over H simplifies to

$$\sum_{r_1 ... r_q, r'_1 ... r'_q} \Pr_D[H(x_i) = r_i, H(x'_i) = r'_i \; \forall i \in \{1, ..., q\}]$$

Putting it back together,

$$\begin{aligned}(\rho_q)_{xyz,x'y'z'} &= \sum_{x_q y_q z_q} \sum_{x'_q y'_q z'_q} \cdots \sum_{x_1 y_1 z_1} \sum_{x'_1 y'_1 z'_1} \sum_{r_1 ... r_q, r'_1 ... r'_q} \\ &\qquad \Pr_D[H(x_i) = r_i, H(x'_i) = r'_i \; \forall i \in \{1, ..., q\}] \\ &\qquad U_{xyzx_q y_q \oplus r_q z_q} U_{x_q y_q z_q x_{q-1} y_{q-1} \oplus r_{q-1} z_{q-1}} ... U_{x_2 y_2 z_2, x_1 y_1 \oplus r_1 z_1} \alpha_{x_1 y_1 z_1} \\ &\qquad \alpha^*_{x'_1 y'_1 z'_1} U^*_{x'_2 y'_2 z'_2, x'_1 y'_1 \oplus r'_1 z'_1} ... U^*_{x'_q y'_q z'_q x'_{q-1} y'_{q-1} \oplus r'_{q-1} z'_{q-1}} U^*_{x'y'z'x'_q y'_q \oplus r'_q z'_q}\end{aligned}$$

Thus, the density matrix ρ_q, which contains all the statistical information about the algorithm A, only depends on the marginal distributions on the subsets of $2q$ inputs. □

C Proof of Theorem 4.2

We complete the proof of Theorem 4.2 by giving a proof of Claim 1.

Proof of Claim 1. Recall that Claim 1 states that if we set $\lambda_i = \left(\frac{i}{d-\Delta}\right)^2$ for $i \in \{1, ..., d-\Delta\}$, as long as $\lambda \leq \lambda_1 = 1/(d-\Delta)^2$, $\sum_i |a_i(\lambda)| \leq 2\zeta(2\Delta+2)(d-\Delta)^{2\Delta+2}\lambda^{\Delta+1}$

Consider the quantity

$$\left|\frac{a_i(\lambda)}{\lambda^{\Delta+1}}\right| = \frac{1}{\lambda_i^{\Delta+1}} \prod_{j=1,j\neq i}^{d-\Delta} \frac{|\lambda - \lambda_j|}{|\lambda_i - \lambda_j|}$$

As long as $\lambda \leq \lambda_i$ for all i (which is equivalent to $\lambda \leq \lambda_1$), then $|\lambda - \lambda_i| \leq \lambda_i$, so

$$\left|\frac{a_i(\lambda)}{\lambda^{\Delta+1}}\right| \leq \frac{1}{\lambda_i^{\Delta+1}} \frac{\prod_{j=1,j\neq i}^{d-\Delta} \lambda_j}{\prod_{j=1,j\neq i}^{d-\Delta} |\lambda_i - \lambda_j|} \leq \left(\frac{d-\Delta}{i}\right)^{2(\Delta+1)} \frac{\prod_{j=1,j\neq i}^{d-\Delta} j^2}{\prod_{j=1,j\neq i}^{d-\Delta} \left|i^2 - j^2\right|}$$

Notice that $\prod_{j=1,j\neq i}^{d-\Delta} j^2 = \left(\frac{(d-\Delta)!}{i}\right)^2$ and that

$$\begin{aligned}
\prod_{j=1,j\neq i}^{d-\Delta} \left|i^2 - j^2\right| &= \prod_{j=1,j\neq i}^{d-\Delta} |i-j|(i+j) \\
&= \left(\prod_{j=1}^{i-1}(i-j)\right)\left(\prod_{j=i+1}^{d-\Delta}(j-i)\right)\left(\prod_{j=1}^{d-\Delta}(i+j)\right)/2i \\
&= (i-1)!(d-\Delta-i)!\frac{d-\Delta+i)!}{2i(i!)} \\
&= \frac{(d-\Delta+i)!(d-\Delta-i)!}{2i^2}
\end{aligned}$$

Combining these together,

$$|a_i(\lambda)| \leq \lambda^{\Delta+1}\left(\frac{(d-\Delta)}{i}\right)^{2(\Delta+1)} \frac{2((d-\Delta)!)^2}{(d-\Delta+i)!(d-\Delta-i)!}$$

Write this quantity as

$$\left(2\lambda^{\Delta+1}(d-\Delta)^{2(\Delta+1)}((d-\Delta)!)^2\right)\left(\frac{1}{(d-\Delta)+i)!(d-\Delta-i)!}\right)\left(\frac{1}{i^{2\Delta+2}}\right)$$

The first term is constant in i. The second term is strictly decreasing in i, since to go from i to $i+1$, we multiply by $\frac{d-\Delta-i}{d-\Delta+i+1} < 1$.

For $i = 1$, the product of the first two terms is

$$2\lambda^{\Delta+1}(d-\Delta)^{2(\Delta+1)}\frac{d-\Delta}{d-\Delta+1} < 2\lambda^{\Delta+1}(d-\Delta)^{2(\Delta+1)}$$

Thus, for $i \geq 1$, the product of the first two terms is less than this quantity, hence

$$|a_i(\lambda)| < 2\lambda^{\Delta+1}(d-\Delta)^{2\Delta+2}\frac{1}{i^{2\Delta+2}}$$

Summing over all i gives us:

$$\sum_{i=1}^{d-\Delta} |a_i(\lambda)| < 2(d-\Delta)^{2\Delta+2}\lambda^{\Delta+1}\sum_{i=1}^{d-\Delta}\frac{1}{i^{2\Delta+2}}$$

This last summation is the truncation of the p-series with $p = 2\Delta + 2$, and is strictly less than the limit of the series - the Riemann Zeta function ζ evaluated at p. Thus,

$$\sum_{i=1}^{d-\Delta} |a_i(\lambda)| < 2\zeta(2\Delta + 2)\lambda^{\Delta+1}(d - \Delta)^{2\Delta+2}$$

□

D More on Semi-Constant Distributions

In this section, we prove Lemma 4.3, which states that, for any n, the marginal distributions for any k inputs of oracles drawn from SC_λ^n specified by polynomials of degree k for which the λ^1 coefficient is 0. We also describe a quantum algorithm for finding collisions in SC_λ^1 using $O(\lambda^{-2/3})$ oracle queries.

D.1 Proof of Lemma 4.3

Proof of Lemma 4.3. Recall that SC_λ^n is defined as follows:

- For each $i \in \{1, ..., n\}$, pick y_i at random from $\mathcal{Y}$.
- For each $x \in \mathcal{X}$, with probability $1 - \lambda$, set $H(x)$ to be a random element of $\mathcal{Y}$. With probability λ, set $H(x) = y_i$ for a random i.

Suppose $\mathcal{Y}$ contains N elements. Let $\alpha((x_j), (r_j)) = \Pr[H(x_i) = r_j \forall i \in \{1, ..., k\}]$, the probability that x_j maps to r_j for k values of x_j and r_j.Suppose there are ℓ distinct r_j, denoted t_m, and let k_m be the number of r_j equal to t_m (note that $\sum_{m=1}^{\ell} k_m = k$). Let $\mathcal{F}$ be the set of functions $f : \{1, ..., n\} \to \{1, ..., \ell, \perp\}$. Each f is associated to the event that $y_i = t_{f(i)}$ if $f(i) \neq \perp$, and $y_i \neq t_m$ for any m if $f(i) = \perp$.

Given an f, let $\mathtt{num}_{f,m}$ be the number of i such that $f(i) = m$. The probability that $O(x_j) = t_m$ is $1/N$ if we are choosing the output at random, and otherwise, it is the fraction of $i \in \{1, ..., n\}$ such that $f(i) = m$. In other words, this probability is

$$\frac{1 - \lambda}{N} + \lambda \frac{\mathtt{num}_{f,m}}{n} = \frac{1}{N} + \lambda\left(\frac{\mathtt{num}_{f,m}}{n} - \frac{1}{N}\right)$$

Since there are k_m copies of each t_m among the r_j,

$$\Pr[H(x_i) = r_j \forall i \in \{1, ..., k\} | f] = \prod_{i=m}^{\ell} \left(\frac{1}{N} + \lambda\left(\frac{\mathtt{num}_{f,m}}{n} - \frac{1}{N}\right)\right)^{k_m}$$

Summing over all f gives

$$\alpha((x_i), (r_j)) = \sum_f \Pr[f] \prod_{i=m}^{\ell} \left(\frac{1}{N} + \lambda\left(\frac{\mathtt{num}_{f,m}}{n} - \frac{1}{N}\right)\right)^{k_m}$$

This is a polynomial in λ. It is a sum of products of $\sum_{m=1}^{\ell} k_m = k$ monomials in λ, so its total degree is at most k. Now, we shall approximate this to first order in λ:

$$\begin{aligned}
\alpha((x_i),(r_j)) &= \sum_f \Pr[f] \prod_{m=1}^{\ell} \left(\frac{1}{N} + \lambda \Big(\frac{\mathtt{num}_{f,m}}{n} - \frac{1}{N} \Big) \right)^{k_m} \\
&= \sum_f \Pr[f] \prod_{m=1}^{\ell} \left(\frac{1}{N^{k_m}} + k_m \frac{1}{N^{k_m - 1}} \lambda \Big(\frac{\mathtt{num}_{f,m}}{n} - \frac{1}{N} \Big) \right) + O(\lambda^2) \\
&= \frac{1}{N^{\sum_m k_m}} \sum_f \Pr[f] \prod_{m=1}^{\ell} \left(1 + k_m N \lambda \Big(\frac{\mathtt{num}_{f,m}}{n} - \frac{1}{N} \Big) \right) + O(\lambda^2) \\
&= \frac{1}{N^k} \sum_f \Pr[f] \left(1 + N\lambda \sum_{m=1}^{\ell} k_m \Big(\frac{\mathtt{num}_{f,m}}{n} - \frac{1}{N} \Big) \right) + O(\lambda^2) \\
&= \frac{1}{N^k} + \frac{\lambda}{N^{k-1}} \left(\frac{1}{n} \sum_{m=1}^{\ell} k_m \sum_f \Pr[f] \mathtt{num}_{f,m} - \frac{k}{N} \right) + O(\lambda^2)
\end{aligned}$$

Notice that $\sum_f \Pr[f]\mathtt{num}_{f,m}$ is the expected number of i such that $f(i) = m$. This is equivalent to the expected number of i such that $y_i = t_m$. Since each $y_i = t_m$ with probability $1/N$, and there are n such y_i, this expected value is n/N. Thus,

$$\alpha((x_i),(r_j)) = \frac{1}{N^k} + \frac{\lambda}{N^{k-1}} \left(\frac{1}{n} \sum_{m=1}^{\ell} k_m \frac{n}{N} - \frac{k}{N} \right) + O(\lambda^2) = \frac{1}{N^k} + O(\lambda^2)$$

Hence, the λ^1 coefficient is 0. This completes the lemma. □

D.2 Finding Collisions in Semi-Constant Distributions

Here we explore the problem of finding a collision in an oracle drawn from a semi-constant distribution SC_λ^1 over $\mathcal{H}_{\mathcal{X},\mathcal{Y}}$. Our motivation for studying the collision problem is as follows: we state without proof that a classical algorithm making q_H queries can only distinguish SC_λ^1 from random with probability $O(\lambda^2 q_H^2)$. Further, querying random points, and outputting 1 if a collision is found achieves this bound. Thus, a collision search is the best way to distinguish SC_λ^1 from random in the classical setting, and the same may also be true in the quantum setting

Let $N = |\mathcal{Y}|$ be the number of elements in $\mathcal{Y}$, and assume that $\lambda >> \frac{1}{N}$ so that finding a collision requires (with high probability) finding a collision in the distinguished inputs.

Let c be the minimum constant such that Corollary 4.4 is true for $\ell(q) = O(q^c)$. Specifically, c is the constant such that no quantum algorithm can distinguish SC_λ^n from random with probability $\omega(q_H^c \lambda^2)$, but some quantum algorithm can with probability $O(q_H^c \lambda^2)$. We know that $c \leq 4$. We will now show that $c \geq 3$, using the following algorithm. The algorithm is basically the algorithm of Brassard et al. [BHT97], but modified for our purposes. It operates as follows:

- Select a subset $\mathcal{W} \subseteq \mathcal{X}$ of size $\lambda^{-1/2}$. For each $x \in \mathcal{W}$, store the pair $(x, H(x))$ in a table, sorted by the second coordinate. Check if there is a collision in $\mathcal{W}$. If so, output this collision.

- Construct the oracle $O(x)$ which is 1 if and only if $x \notin \mathcal{W}$ and $H(x) = H(x_0)$ for some $x_0 \in \mathcal{W}$. Since the entries to $\mathcal{W}$ are sorted by the second coordinate, this test can be performed efficiently.
- Run Grover's algorithm on this oracle to find an $x \notin \mathcal{W}$ such that $H(x) = H(x_0)$ for some $x_0 \in \mathcal{W}$.
- Output (x, x_0).

The probability that we found a distinguished input in the first step is

$$1 - (1-\lambda)^{|\mathcal{W}|} = 1 - (1-\lambda)^{\lambda^{-1/2}} = \lambda^{1/2} + O(\lambda)$$

Grover's algorithm finds an x such that $O(x) = 1$ in using $O(f^{-1/2})$ queries, where f is the fraction of inputs to O that map to 1. Thus, f is the fraction of x such that $x \notin \mathcal{W}$ (which is most of them) and $H(x) = H(x_0)$ for some $x_0 \in \mathcal{W}$. If we found an distinguished input, this fraction is (in expectation) at least λ.

If we found a distinguished input in the first step, Grover's algorithm will thus find an x such that $O(x) = 1$ with high probability using $O(\lambda^{-1/2})$ queries. The total number of oracle queries is then at most $O(\lambda^{-1/2})$ to find a collision with expected probability $O(\lambda^{1/2})$.

Using this algorithm, we can distinguish SC^1_λ from uniform by simply testing if the output is indeed a collision. Thus we use $O(\lambda^{-1/2})$ queries to distinguish with probability $O(\lambda^{-1/2})$. This probability is at most $O(\lambda^2 q^c) = O(\lambda^{2-c/2})$. Hence, $c \geq 3$.

E Proof of Theorem 5.3

We prove Theorem 5.3, which states that the FDH signature scheme is secure in the quantum random oracle model if the underlying trapdoor permutation is secure against quantum adversaries.

Proof of Theorem 5.3. Suppose towards contradiction that there is a quantum adversary A making q_H hash queries, q_S signature queries, that breaks S with probability ϵ. Assume without loss of generality that A never tries to forge a signature on a message it already received a secret key for.

Let Game_0 be the standard attack game for S: the challenger generates $(\mathsf{sk}, \mathsf{pk})$ from $\mathsf{S.Gen}$, and sends pk to the adversary. The adversary can make (classical) signature queries on messages m_i, to which the challenger responds with $\mathsf{S.Sign}^H_{\mathsf{sk}}(m_i)$, and (quantum) hash queries to the random oracle H. A wins if it can produce a pair (m, σ) such that $m \neq m_i$ for any i, and $\mathsf{S.Ver}^H_{\mathsf{pk}}(m, \sigma) = \mathsf{accept}$. The success probability in Game_0 is ϵ.

Let $\lambda \in (0, 1)$ to be chosen later. Let $\mathcal{M}$ be the message space. Let Game_1 be Game_0, except that we choose a subset $\mathcal{X}$ of $\mathcal{M}$ as follows: for each $m \in \mathcal{M}$, we put m in $\mathcal{X}$ with probability λ. If A asks for a signature on a message in $\mathcal{X}$, abort and A loses. If the message A attempts to forge is *not* in $\mathcal{X}$, also abort and A loses. Otherwise, follow the criteria in Game_0 to decide if A wins. The message A tries to forge must be distinct from the messages it received signatures for, so the probabilities that each of these messages are in $\mathcal{X}$ is independent. Therefore, the no-abort conditions are met with probability at least $\lambda(1 - \lambda q_E)$. Thus, A wins in Game_1 with probability at least $\lambda(1 - \lambda q_E)\epsilon$.

Let Game_2 be Game_1, except that we choose a random y in the public key space of S, and set $H(x) = y$ for all x in $\mathcal{X}$. H is now distributed according to SC^1_λ, so by Corollary 4.4, the probability A wins is at least Game_2 is at least $\lambda(1 - \lambda q_E)\epsilon - \ell(q_H)\lambda^2$.

Now we are ready to define an algorithm B what inverts f. Give B access to two random oracles O_1 and O_2. O_1 maps messages to inputs to f. O_2 maps messages to bits, where the probability of outputting 1 is λ. On input (pk, y), B works as follows:

- Send pk to A, and simulate A, playing the role of challenger to A.
- Construct the (quantum) oracle H such that

$$H(m) = \begin{cases} y & \text{if } O_2(m) = 1 \\ f_{\mathsf{pk}}(O_1(m)) & \text{otherwise} \end{cases}$$

- When A makes a signature query on a message m_i, compute $O_2(m_i)$. If the result is 1, abort. Otherwise, return $\sigma_i = O_1(m_i)$.
- When A returns a forgery candidate (m, σ), test if $O_2(m) = 1$ and $f_{\mathsf{pk}}(\sigma) = y$. If so, output σ. Otherwise, abort.

Using an analysis similar to that of Theorem 5.1, we get that the advantage of B is at least

$$\frac{\epsilon^2}{4(\ell(q_H) + q_S)}$$

When we set $\lambda = \epsilon/2(\ell(q_H) + q_S\epsilon)$. □