2012 IEEE 53rd Annual Symposium on Foundations of Computer Science

How to Construct Quantum Random Functions

Mark Zhandry
Stanford University, USA
mzhandry@stanford.edu

***Abstract*—In the presence of a quantum adversary, there are two possible definitions of security for a pseudorandom function. The first, which we call standard-security, allows the adversary to be quantum, but requires queries to the function to be classical. The second, quantum-security, allows the adversary to query the function on a quantum superposition of inputs, thereby giving the adversary a superposition of the values of the function at many inputs at once. Existing techniques for proving the security of pseudorandom functions fail when the adversary can make quantum queries. We give the first quantum-security proofs for pseudorandom functions by showing that some classical constructions of pseudorandom functions are quantum-secure. Namely, we show that the standard constructions of pseudorandom functions from pseudorandom generators or pseudorandom synthesizers are secure, even when the adversary can make quantum queries. We also show that a direct construction from lattices is quantum-secure. To prove security, we develop new tools to prove the indistinguishability of distributions under quantum queries.**

In light of these positive results, one might hope that all standard-secure pseudorandom functions are quantum-secure. To the contrary, we show a separation: under the assumption that standard-secure pseudorandom functions exist, there are pseudorandom functions secure against quantum adversaries making classical queries, but insecure once the adversary can make quantum queries.

***Keywords*-Quantum; Pseudorandom Function**

I. Introduction

In their seminal paper, Goldreich, Goldwasser, and Micali [7] answer the question of how to construct a function that looks random to classical adversaries. Specifically, they define a pseudorandom function (PRF) as a function PRF with the following property: no efficient classical algorithm, when given oracle access to PRF, can distinguish PRF from a truly random function. They then construct such a pseudorandom function from pseudorandom generators. Since then, pseudorandom functions have also been built from pseudorandom synthesizers [11], as well as directly from hard problems [1], [4], [6], [9], [12], [14]. Pseudorandom functions have become an important tool in cryptography: for example, they are used in the construction of identification protocols, block ciphers, and message authentication codes.

To define pseudorandom functions in the presence of a quantum adversary, two approaches are possible. The first is what we call standard-security: the quantum adversary can only make *classical* queries to the function, but all the computation between the queries may be quantum. The second, which we call quantum-security, allows the adversary to make *quantum* queries to the function. That is, the adversary can send a quantum superposition of inputs to the function, and receives a superposition of the corresponding outputs in return. We call pseudorandom functions that are secure against quantum queries Quantum Pseudorandom Functions, or QPRFs. Constructing secure QPRFs will be the focus of this paper.

Quantum-secure pseudorandom functions (QPRFs) have several applications. Whenever a pseudorandom function is used in the presence of a quantum adversary, security against quantum queries captures a wider class of attacks. Thus, the conservative approach to cryptosystem design would dictate using a quantum-secure pseudorandom function. Further, in any instance where a pseudorandom function might be evaluated on a superposition, quantum-security is required. For example, classically, pseudorandom functions work as message authentication codes (MACs). In the quantum world, however, it may be possible for a quantum adversary to query the MAC on a superposition of messages, thus necessitating the use of a quantum-secure pseudorandom function.

Lastly, quantum-secure pseudorandom functions can be used to simulate quantum-accessible random oracles [2]. Unlike the classical setting, where a random oracle can be simulated on the fly, simulating a quantum-accessible random oracle requires defining the entire function up front before any queries are made. Zhandry [16] observes that if the number of queries is *a-priori* bounded by q, $2q$-wise independent functions are sufficient. However, whenever the number of quantum queries is not known in advance, quantum-secure pseudorandom functions seem necessary for simulating quantum-accessible random oracles.

A. Proving Quantum Security

Goldreich, Goldwasser, and Micali show how to build a pseudorandom function PRF from any length-doubling

0272-5428/12 $26.00 © 2012 IEEE
DOI 10.1109/FOCS.2012.37

pseudorandom generator G. This construction is known as the GGM construction. Pseudorandom generators can, in turn, be built from any one-way function, as shown by Håstad et al. [8]. The security proof of Håstad et al. is entirely black-box, meaning that it carries over to the quantum setting immediately if the underlying one-way function is secure against quantum adversaries. However, we will now see that the classical proof of security for the GGM construction does not hold in the quantum world.

At a high level, implicit in the GGM construction is a binary tree of depth n, where each leaf corresponds to an input/output pair of PRF. To evaluate PRF, we start at the root, and follow the path from root to the leaf corresponding to the input. The security proof consists of two hybrid arguments: the first across levels of the tree, and the second across the nodes in a particular level. The first step has only polynomially many hybrids since the tree's depth is a polynomial. For the second step, a classical adversary can only query PRF on polynomially many points, so the paths used to evaluate PRF only visit polynomially many nodes in each level. Therefore, we only need polynomially many hybrids for the second hybrid argument. This allows any adversary A that breaks the security of PRF with probability ϵ to be converted into an adversary B that breaks the security of G with probability only polynomially smaller that ϵ.

In the quantum setting, A may query PRF on a superposition of all inputs, so the response to even a single query could require visiting all nodes in the tree. Each level of the tree may have exponentially many nodes, so the second hybrid argument above would need exponentially many hybrids in the quantum setting. This means B breaks the security of G with only exponentially small probability. All existing security proofs for pseudorandom functions from standard assumptions suffer from similar weaknesses.

B. Our Results

We answer the question of how to construct a function that looks random to *quantum* adversaries. The answer is simple: many of the constructions of standard-secure pseudorandom functions are in fact quantum-secure. However, as explained above, new techniques are required to actually prove security.

We start by showing that, given the existence of a standard-secure pseudorandom function, there are standard-secure pseudorandom functions that are not quantum-secure. Thus a standard-secure PRF may not be secure as a QPRF.

Next, for several classical constructions of pseudorandom functions, we now can show how to modify the classical security proof to prove quantum security. Our general technique is as follows: first define a seemingly stronger security notion for the underlying cryptographic primitive. Next, use ideas from the classical proof to show that any adversary A that breaks the quantum-security of the pseudorandom function can be turned into an adversary B that breaks this stronger notion of security for the primitive. Lastly, use new techniques to show the equivalence of this stronger notion of security and the standard notion of security in the quantum setting.

We use this approach to prove the security of the following pseudorandom functions:

- The construction from length-doubling pseudorandom generators (PRGs) due to Goldreich, Goldwasser, and Micali [7]. Since pseudorandom generators can be built from one-way functions in the quantum setting, this shows that one-way functions secure against quantum adversaries imply quantum-secure pseudorandom functions.
- The construction from pseudorandom synthesizers due to Naor and Reingold [11].
- The direct construction based on the Learning With Errors problem due to Banerjee, Peikert, and Rosen [1].

C. Are Quantum Oracles Better Than Samples?

In the GGM proof, the first hybrid argument over the levels of the tree essentially transforms an adversary for PRF into an algorithm solving the following problem: distinguish a random oracle from an oracle whose outputs are random values from the underlying pseudorandom generator. The next hybrid argument shows how to use such an algorithm do distinguish a single random value from a single output of the pseudorandom generator, thus breaking the security of the pseudorandom generator.

The first hybrid argument carries over into the quantum setting, but it is this second hybrid that causes difficulties for the reasons outlined above. To complete the security proof, we need to show that having quantum access to an oracle whose outputs are drawn from a distribution is no better than having access to a single sample from the same distribution. We show exactly that:

Theorem I.1. *Let D_1 and D_2 be efficiently sampleable distributions on a set $\mathcal{Y}$, and let $\mathcal{X}$ be some other set. Let O_1 and O_2 be the distributions of functions from $\mathcal{X}$ to $\mathcal{Y}$ where for each $x \in \mathcal{X}$, $O_i(x)$ is chosen independently according to D_i. Then if A is an efficient quantum algorithm that uses quantum queries to distinguish oracles drawn from O_1 from oracles drawn from O_2, we can construct an efficient quantum algorithm B that distinguishes samples from D_1 and D_2.*

In the classical case, any algorithm A making q queries to an oracle O only sees q outputs. Thus, given q samples from D_1 or D_2, we can lazily simulate the oracles O_1 or

O_2, getting an algorithm that distinguishes q samples of D_1 from q samples of D_2. A simple hybrid argument shows how to get an algorithm that distinguishes one sample.

In the quantum setting, any quantum algorithm A making even a single quantum query to O_i gets to "see" all the outputs at once, meaning we need exponentially many samples to simulate O_i exactly. However, while we cannot lazily simulate the oracles O_i given q samples from D_i, we can approximately simulate O_i given polynomially many samples from D_i. Basically, for each input, set the output to be one of the samples, chosen at random from the collection of samples. While not quite the oracle O_i, we show that this is sufficiently indistinguishable from O_i. Thus, we can use A to distinguish a polynomial number of samples of D_1 from the same number of samples of D_2. Like in the classical case, a simple hybrid argument shows how to distinguish just one sample.

II. Preliminaries and Notation

We say that $\epsilon = \epsilon(n)$ is negligible if, for all polynomials $p(n)$, $\epsilon(n) < 1/p(n)$ for large enough n.

For an integer k, we will use non-standard notation and write $[k] = \{0, ..., k-1\}$ to be the set of non-negative integers less than k. We write the set of all n bit strings as $[2]^n$. Let $x = x_1...x_n$ be a string of length n. We write $x_{[a,b]}$ to denote the substring $x_a x_{a+1}...x_b$.

A. Functions and Probabilities

Given two sets $\mathcal{X}$ and $\mathcal{Y}$, define $\mathcal{Y}^{\mathcal{X}}$ as the set of functions $f : \mathcal{X} \to \mathcal{Y}$. If a function f has maps $\mathcal{X}$ to $\mathcal{Y} \times \mathcal{Z}$, we can think of f as two functions: one that maps $\mathcal{X}$ to $\mathcal{Y}$ and one that maps $\mathcal{X}$ to $\mathcal{Z}$. In other words, we can equate the sets of functions $(\mathcal{Y} \times \mathcal{Z})^{\mathcal{X}}$ and $\mathcal{Y}^{\mathcal{X}} \times \mathcal{Z}^{\mathcal{X}}$.

Given $f \in \mathcal{Y}^{\mathcal{X}}$ and $g \in \mathcal{Z}^{\mathcal{Y}}$, let $g \circ f$ be the composition of f and g. That is, $g \circ f(x) = g(f(x))$. If $\mathcal{F} \subseteq \mathcal{Y}^{\mathcal{X}}$, let $g \circ \mathcal{F}$ be the set of functions $g \circ f$ for $f \in \mathcal{F}$. Similarly, if $\mathcal{G} \subseteq \mathcal{Z}^{\mathcal{Y}}$, $\mathcal{G} \circ f$ is the set of functions $f \circ g$ where $g \in \mathcal{G}$. Define $\mathcal{G} \circ \mathcal{F}$ accordingly.

Given a distribution D and some event `event`, we write $\Pr_{x \leftarrow D}[\texttt{event}]$ to represent the probability that `event` happens when x is drawn from D. For a given set $\mathcal{X}$, we will sometimes abuse notation and write $\mathcal{X}$ to denote the uniform distribution on $\mathcal{X}$.

Given a distribution D on $\mathcal{Y}^{\mathcal{X}}$ and a function $g \in \mathcal{Z}^{\mathcal{Y}}$, define the distribution $g \circ D$ over $\mathcal{Z}^{\mathcal{X}}$ where we first draw f from D, and output the composition $g \circ f$. Given $f \in \mathcal{Y}^{\mathcal{X}}$ and a distribution E over $\mathcal{Z}^{\mathcal{X}}$, define $E \circ f$ and $E \circ D$ accordingly.

Given a distribution D on a set $\mathcal{Y}$, and another set $\mathcal{X}$, define $D^{\mathcal{X}}$ as the distribution on $\mathcal{Y}^{\mathcal{X}}$ where the output for each input is chosen independently according to D.

The distance between two distributions D_1 and D_2 over a set $\mathcal{X}$ is

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

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

B. Quantum Computation

Here we state some basic facts about quantum computation needed for the paper, and refer the reader to Nielsen and Chuang [15] for a more in depth discussion.

Fact 1. *Any classical efficiently computable function f can be implemented efficiently by a quantum computer. Moreover, f can be implemented as an oracle which can be queried on quantum superpositions.*

The following is a result from Zhandry [16]:

Fact 2. *For any sets $\mathcal{X}$ and $\mathcal{Y}$, we can efficiently "construct" a random oracle from $\mathcal{X}$ to $\mathcal{Y}$ capable of handling q quantum queries, where q is a polynomial. More specifically, the behavior of any quantum algorithm making at most q queries to a $2q$-wise independent function is identical to its behavior when the queries are made to a random function.*

Given an efficiently sampleable distribution D over a set $\mathcal{Y}$, we can also "construct" a random function drawn from $D^{\mathcal{X}}$ as follows: Let $\mathcal{Z}$ be the set of randomness used to sample from D, and let $f(r)$ be the element $y \in \mathcal{Y}$ obtained using randomness $r \in \mathcal{Z}$. Then $D^{\mathcal{X}} = f \circ \mathcal{Z}^{\mathcal{X}}$, so we first construct a random function $O' \in \mathcal{Z}^{\mathcal{X}}$, and let $O(x) = f(O'(x))$.

We will denote a quantum algorithm A given classical oracle access to an oracle O as A^O. If A has quantum access, we will denote this as $A^{|O\rangle}$.

C. Cryptographic Primitives

In this paper, we always assume the adversary is a quantum computer. However, for any particular primitive, there may be multiple definitions of security, based on how the adversary is allowed to interact with the primitive. Here we define pseudorandom functions and two security notions, as well as two definitions of indistinguishability for distributions. The definitions of pseudorandom generators and synthesizers appear in the relevant sections.

Definition II.1 (PRF)**.** *A pseudorandom function is a function $\mathsf{PRF} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, where $\mathcal{K}$ is the key-space, and $\mathcal{X}$ and $\mathcal{Y}$ are the domain and range. $\mathcal{K}$, $\mathcal{X}$, and $\mathcal{Y}$ are implicitly functions of the security parameter n. We write $y = \mathsf{PRF}_k(x)$.*

Definition II.2 (Standard-Security)**.** *A pseudorandom function* PRF *is standard-secure if no efficient quantum adversary A making* classical *queries can distinguish between a truly random function and the function* PRF_k *for a random k. That is, for every such A, there exists a negligible function $\epsilon = \epsilon(n)$ such that*

$$\left| \Pr_{k \leftarrow \mathcal{K}}[A^{\mathsf{PRF}_k}() = 1] - \Pr_{O \leftarrow \mathcal{Y}^{\mathcal{X}}}[A^{O}() = 1] \right| < \epsilon \ .$$

Definition II.3 (Quantum-Security)**.** *A pseudorandom function* PRF *is quantum-secure if no efficient quantum adversary A making* quantum *queries can distinguish between a truly random function and the function* PRF_k *for a random k.*

We call such quantum-secure pseudorandom functions Quantum Random Functions, or QPRFs.

We now provide some definitions of indistinguishablility for distributions. The standard notion of indistinguishability is that no efficient quantum algorithm can distinguish a sample of one distribution from a sample of the other:

Definition II.4 (Indistinguishability)**.** *Two distributions D_1 and D_2 over a set $\mathcal{Y}$ are computationally (resp. statistically) indistinguishable if no efficient (resp. computationally unbounded) quantum algorithm A can distinguish a sample of D_1 from a sample of D_2. That is, for all such A, there is a negligible function ϵ such that*

$$\left| \Pr_{y \leftarrow D_1}[A(y) = 1] - \Pr_{y \leftarrow D_2}[A(y) = 1] \right| < \epsilon \ .$$

For our work, we will also need a new, seemingly stronger notion of security, which we call oracle-indistinguishability. The idea is that no efficient algorithm can distinguish between oracles whose outputs are distributed according to either D_1 or D_2:

Definition II.5 (Oracle-Indistinguishability)**.** *Two distributions D_1 and D_2 over a set $\mathcal{Y}$ are computationally (resp. statistically) oracle-indistinguishable if, for all sets $\mathcal{X}$, no efficient (resp. computationally unbounded) quantum algorithm B can distinguish $D_1^{\mathcal{X}}$ from $D_2^{\mathcal{X}}$ using a polynomial number of quantum queries. That is, for all such B and $\mathcal{X}$, there is a negligible function ϵ such that*

$$\left| \Pr_{O \leftarrow D_1^{\mathcal{X}}}[B^{|O\rangle}() = 1] - \Pr_{O \leftarrow D_2^{\mathcal{X}}}[B^{|O\rangle}() = 1] \right| < \epsilon \ .$$

For this paper, we will primarily be discussing computationally bounded adversaries, so we will normally take indistinguishability and oracle-indistinguishability to mean the computational versions.

These these definitions of indistinguishability in hand, we can now formulate Theorem I.1 as follows:

Theorem I.1. Let D_1 and D_2 be efficiently sampleable distributions over a set $\mathcal{Y}$. Then D_1 and D_2 are indistinguishable if and only if they are also oracle-indistinguishable.

III. Separation Result

In this section, we show our separation result:

Theorem III.1. *If secure PRFs exist, then there are standard-secure PRFs that are not QPRFs.*

Proof: Let PRF be a standard-secure pseudorandom function with key-space $\mathcal{K}$, domain $\mathcal{X}$, and co-domain $\mathcal{Y}$. We will construct a new pseudorandom function that is periodic with some large, secret period. Classical adversaries will not be able to detect the period, and thus cannot distinguish this new function from random. However, an adversary making quantum queries can detect the period, and thus distinguish our new function from random.

Interpret $\mathcal{X}$ as $[N]$, where N is the number of elements in $\mathcal{X}$. Assume without loss of generality that $\mathcal{Y}$ contains at least N^2 elements (if not, we can construct a new pseudorandom function with smaller domain but larger range in a standard way). We now construct a new pseudorandom function $\mathsf{PRF}'_{(k,a)}(x) = \mathsf{PRF}_k(x \mod a)$ where:

- The key space of PRF' is $\mathcal{K}' = \mathcal{K} \times \mathcal{A}$ where $\mathcal{A} = \mathbb{Z} \cap (N/2, N]$. That is, a key for PRF' is a pair (k, a) where k is a key for PRF, and a is an integer in the range $(N/2, N]$.
- The domain is $\mathcal{X}' = [N']$ where N' is the smallest power of 2 greater than $4N^2$.

The following two claims are proved in the full version [17]:

Claim 1. *If* PRF *is standard-secure, then so is* PRF'

Sketch of Proof. Since PRF is a standard-secure pseudorandom function, we can replace it with a truly random function in the definition of PRF', and no efficient adversary making classical queries will notice. But we are then left with a function that has a large random period where every value in the period is chosen randomly. This function will look truly random unless the adversary happens to query two points that differ by a multiple of the period. But by the birthday bound, this will only happen with negligible probability. ∎

Claim 2. *If* PRF *is quantum-secure, then* PRF' *is not.*

Sketch of Proof. If we allow quantum queries to PRF', we can use the period finding algorithm of Boneh and Lipton [3] to find a. With a, it is easy to distinguish PRF' from a random oracle. Unfortunately, the period finding

algorithm requires PRF' to have some nice properties, but these properties are satisfied if PRF is quantum-secure. ■

Thus one of PRF and PRF' is standard-secure but not quantum-secure, as desired. ■

We have shown that for pseudorandom functions, security against classical queries does not imply security against quantum queries. In the next sections, we will show, however, that several of the standard constructions in the literature are nevertheless quantum-secure.

IV. Pseudorandom Functions from Pseudorandom Generators

We give the construction of pseudorandom functions from pseudorandom generators due to Goldreich, Goldwasser, and Micali [7], the so-called GGM construction. We also prove its security in a new way that makes sense in the quantum setting. First, we define pseudorandom generators:

Definition IV.1 (PRG)**.** *A pseudorandom generator (PRG) is a function $G : \mathcal{X} \to \mathcal{Y}$. $\mathcal{X}$ and $\mathcal{Y}$ are implicitly indexed by the security parameter n.*

Definition IV.2 (Standard-Security)**.** *A pseudorandom function G is standard-secure if the distributions $G \circ \mathcal{X}$ and $\mathcal{Y}$ are computationally indistinguishable.*

Construction 1 (GGM-PRF)**.** *Let $G : \mathcal{K} \to \mathcal{K}^2$ be a length-doubling pseudorandom generator. Write $G(x) = (G_0(x), G_1(x))$ where G_0, G_1 are functions from $\mathcal{K}$ to $\mathcal{K}$. Then we define the GGM pseudorandom function $\mathsf{PRF} : \mathcal{K} \times [2]^n \to \mathcal{K}$ where*

$$\mathsf{PRF}_k(x) = G_{x_1}(...G_{x_{n-1}}(G_{x_n}(k))...) \ .$$

That is, the function PRF takes a key k in $\mathcal{K}$ and an n-bit input string. It first applies G to k. It keeps the left or right half of the output depending on whether the last bit of the input is 0 or 1. What remains is an element in $\mathcal{K}$, so the function applies G again, keeps the left or right half depending on the second-to-last bit, and so on.

As described in the introduction, the standard proof of security fails to prove quantum-security. Using Theorem I.1, we show how to work around this problem. We defer the proof of Theorem I.1 to Section VII, and instead assume it is true. We first define a stronger notion of security for pseudorandom generators, which we call oracle-security:

Definition IV.3 (Oracle-Security)**.** *A pseudorandom generator $G : \mathcal{X} \to \mathcal{Y}$ is oracle-secure if the distributions $G \circ \mathcal{X}$ and $\mathcal{Y}$ are oracle-indistinguishable.*

$G \circ \mathcal{X}$ is efficiently sampleable since we can sample a random value in $\mathcal{X}$ and apply G to it. Then, $G \circ \mathcal{X}$ and $\mathcal{Y}$ are both efficiently sampleable, so Theorem I.1 gives:

Corollary IV.4. *If G is a secure PRG, then it is also oracle-secure.*

We now can prove the security of Construction 1.

Theorem IV.5. *If G is a standard-secure PRG, then PRF from Construction 1 is a QPRF.*

Proof: We adapt the security proof of Goldreich et al. to convert any adversary for PRF into an adversary for the oracle-security of G. Then Corollary IV.4 shows that this adversary is impossible under the assumption that G is standard-secure.

Suppose a quantum adversary A distinguishes PRF from a random oracle with probability ϵ. Define hybrids H_i as follows: Pick a random function $P \leftarrow \mathcal{K}^{[2]^{n-i}}$ (that is, random function from $(n-i)$-bit strings into $\mathcal{K}$) and give A the oracle

$$O_i(x) = G_{x_1}(...G_{x_i}(P(x_{[i+1,n]}))...) \ .$$

H_0 is the case where A's oracle is random. When $i = n$, $P \leftarrow \mathcal{K}^{[2]^{n-i}}$ is a random function from the set containing only the empty string to $\mathcal{K}$, and hence is associated with the image of the empty string, a random element in $\mathcal{K}$. Thus H_n is the case where A's oracle is PRF. Let ϵ_i be the probability A distinguishes H_i from H_{i+1}. That is,

$$\epsilon_i = \Pr[A^{|O_i\rangle}() = 1] - Pr[A^{|O_{i+1}\rangle}() = 1] \ .$$

A simple hybrid argument shows that $|\sum_i \epsilon_i| = \epsilon$

We now construct a quantum algorithm B breaking the oracle-security of G. B is given quantum access to an oracle $P : [2]^{n-1} \to \mathcal{K}^2$, and distinguishes $P \leftarrow \left(\mathcal{K}^2\right)^{[2]^{n-1}}$ from $P \leftarrow G \circ \mathcal{K}^{[2]^{n-1}}$. That is, B is given either a random function from $(n-1)$-bit strings into $\mathcal{K}^2$, or G applied to a random function from $(n-1)$-bit strings into $\mathcal{K}$, and distinguishes the two cases as follows:

- Pick a random i in $\{0, ..., n-1\}$
- Let $P^{(i)} : [2]^{n-i-1} \to \mathcal{K}^2$ be the oracle $P^{(i)}(x) = P(0^i x)$
- Write $P^{(i)}$ as $(P_0^{(i)}, P_1^{(i)})$ where $P_b^{(i)} : [2]^{n-i-1} \to \mathcal{K}$ are the left and right halfs of the output of $P^{(i)}$.
- Construct the oracle $O : [2]^n \to \mathcal{K}$ where

$$O(x) = G_{x_1}(...G_{x_i}(P^{(i)}_{x_{i+1}}(x_{[i+2,n]}))...) \ .$$

- Simulate A with oracle O, and output whatever A outputs.

Notice that each quantum query to O results in one quantum query to P, so B makes the same number of queries that A does.

Fix i, and let B_i be the algorithm B using this i. In the case where P is truly random, so is $P^{(i)}$, as are $P_0^{(i)}$ and $P_1^{(i)}$. Thus $O = O_i$, the oracle in hybrid H_i. When

P is drawn from $G \circ \mathcal{K}^{[2]^{n-1}}$, then $P^{(i)}$ is distributed according to $G \circ \mathcal{K}^{[2]^{n-i-1}}$, and so $P_b \leftarrow G_b \circ \mathcal{K}^{[2]^{n-i-1}}$. Thus $O = O_{i+1}$, the oracle in hybrid H_{i+1}. For fixed i, we then have that the quantity

$$\Pr_{P \leftarrow (\mathcal{K}^2)^{[2]^{n-1}}}[B_i^{|P\rangle}() = 1] - \Pr_{P \leftarrow G \circ \mathcal{K}^{[2]^{n-1}}}[B_i^{|P\rangle}() = 1]$$

is equal to ϵ_i. Averaging over all i and taking the absolute value, we have that the distinguishing probability of B,

$$\left| \Pr_{P \leftarrow (\mathcal{K}^2)^{[2]^{n-1}}}[B^{|P\rangle}() = 1] - \Pr_{P \leftarrow G \circ \mathcal{K}^{[2]^{n-1}}}[B^{|P\rangle}() = 1] \right| ,$$

is equal to

$$\left| \frac{1}{n} \sum_i \epsilon_i \right| = \epsilon/n \ .$$

Thus B breaks the oracle security of G with probability only polynomially smaller than the probability A distinguishes PRF from a random oracle.

■

V. Pseudorandom Functions from Synthesizers

In this section, we show that the construction of pseudorandom functions from pseudorandom synthesizers due to Naor and Reingold [11] is quantum-secure.

Definition V.1 (Synthesizer)**.** *A pseudorandom synthesizer is a function $S : \mathcal{X}^2 \to \mathcal{Y}$. $\mathcal{X}$ and $\mathcal{Y}$ are implicitly indexed by the security parameter n.*

Definition V.2 (Standard-Security)**.** *A pseudoreandom synthesizer $S : \mathcal{X}^2 \to \mathcal{Y}$ is standard-secure if, for any set $\mathcal{Z}$, no efficient quantum algorithm A making classical queries can distinguish a random function from $O(z_1, z_2) = S(O_1(z_1), O_2(z_2))$ where $O_b \leftarrow \mathcal{X}^{\mathcal{Z}}$. That is, for any such A and $\mathcal{Z}$, there exists a negligible function ϵ such that*

$$\left| \Pr_{\substack{O_1 \leftarrow \mathcal{X}^{\mathcal{Z}} \\ O_2 \leftarrow \mathcal{X}^{\mathcal{Z}}}}[A^{S(O_1,O_2)}() = 1] - \Pr_{O \leftarrow \mathcal{Y}^{\mathcal{Z} \times \mathcal{Z}}}[A^{O}() = 1] \right| < \epsilon \ ,$$

where $S(O_1, O_2)$ means the oracle that maps (z_1, z_2) into $S(O_1(z_1), O_2(z_2))$.

Construction 2 (NR-PRF)**.** *Given a pseduorandom synthesizer $S : \mathcal{X}^2 \to \mathcal{X}$, let ℓ be an integer and $n = 2^\ell$. We let $\mathsf{PRF}_k(x) = \mathsf{PRF}_k^{(\ell)}(x)$ where $\mathsf{PRF}^{(i)} : \left(\mathcal{X}^{2 \times 2^i}\right) \times [2]^{2^i} \to \mathcal{X}$ is defined as*

$$\begin{aligned} \mathsf{PRF}^{(0)}_{a_{1,0},a_{1,1}}(x) &= a_{1,x} \\ \mathsf{PRF}^{(i)}_{A_1^{(i-1)},A_2^{(i-1)}}(x) &= S(\mathsf{PRF}^{(i-1)}_{A_1^{(i-1)}}(x_{[1,2^{i-1}]}), \\ &\qquad \mathsf{PRF}^{(i-1)}_{A_2^{(i-1)}}(x_{[2^{i-1}+1,2^i]})) \ , \end{aligned}$$

where

$$\begin{aligned} A_1^{(i-1)} &= (a_{1,0}, a_{1,1}, a_{2,0}, a_{2,1}, ..., a_{2^{i-1},0}, a_{2^{i-1},1}) \\ A_2^{(i-1)} &= (a_{2^{i-1}+1,0}, a_{2^{i-1}+1,1}, a_{2,0}, a_{2,1}, ..., a_{2^i,0}, a_{2^i,1}) \end{aligned}$$

That is, PRF takes a key k consisting of $2 \times 2^\ell$ elements of $\mathcal{X}$, and takes bit strings x of length 2^ℓ as input. It uses x to select 2^ℓ of the elements in the key, and pairs them off. It then applies S to each of the pairs, obtaining $2^{\ell-1}$ elements of $\mathcal{X}$. Next, PRF pairs these elements and applies S to these pairs again, and continues in this way until there is one element left, which becomes the output.

The following theorem is proved in the full version [17]:

Theorem V.3. *If S is a standard-secure synthesizer, then PRF from Construction 2 is a QPRF.*

Sketch of Proof. The proof is very similar to that of the security of the GGM construction: we define a new notion of security for synthesizers, called quantum-security, and use the techniques of Naor and Reingold to prove that quantum-security implies that Construction 2 is quantum secure. Unlike the GGM case, the equivalence of quantum- and standard-security for synthesizers is not an immediate consequence of Theorem I.1. Nevertheless, we prove the equivalence, completing the proof of security for Construction 2. ■

VI. Direct Construction of Pseudorandom Functions

In this section, present the construction of pseudorandom functions from Banerjee, Peikert, and Rosen [1]. We show that this construction is quantum-secure.

Let p, q be integers with $q > p$. Let $\lfloor x \rceil_p$ be the map from $\mathbb{Z}_q$ into $\mathbb{Z}_p$ defined by first rounding x to the nearest multiple of q/p, and then interpreting the result as an element of $\mathbb{Z}_p$. More precisely, $\lfloor x \rceil_p = \lfloor (p/q)x \rceil \mod p$ where the multiplication and division in $(p/q)x$ are computed in $\mathbb{R}$.

Construction 3. *Let p, q, m, ℓ be integers with $q > p$. Let $\mathcal{K} = \mathbb{Z}_q^{n \times m} \times (\mathbb{Z}^{n \times n})^\ell$. We define $\mathsf{PRF} : \mathcal{K} \times [2]^\ell \to \mathbb{Z}_p^{m \times n}$ as follows: For a key $k = (\mathbf{A}, \{\mathbf{S}_i\})$, let*

$$\mathsf{PRF}_k(x) = \left\lfloor \mathbf{A}^t \prod_{i=1}^{\ell} \mathbf{S}_i^{x_i} \right\rceil_p \ .$$

The function PRF uses for a key an $n \times m$ matrix $\mathbf{A}$ and ℓ different $n \times n$ matrices $\mathbf{S}_i$, where elements are integers mod q. It uses its ℓ-bit input to select a subset of the $\mathbf{S}_i$, which it multiplies together. The product is then multiplied by the transpose of $\mathbf{A}$, and the whose result is rounded mod p.

Next is an informal statement of the security of PRF, whose proof appears in the full version [17]:

Theorem VI.1. *Let* PRF *be as in Construction 3. For an appropriate choice of integers p, q, m, ℓ and distribution χ on $\mathbb{Z}$, if we draw $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$ and $S_i \leftarrow \chi^{n \times n}$ and the Learning With Errors (*LWE*) problem is hard for modulus q and distribution χ, then* PRF *is a QPRF.*

Sketch of Proof. We follow the ideas from the previous sections and define a new notion of hardness for LWE, which we call oracle-hard, and show its equivalence to standard hardness. We then show that oracle-hardness implies Construction 3 is quantum-secure. This part is similar to the proof of Banerjee et al., with some changes to get it to work in the quantum setting. ∎

VII. Distinguishing Oracle Distributions

In this section, we describe some tools for arguing that a quantum algorithm cannot distinguish between two oracle distributions, culminating in a proof for Theorem I.1. Let $\mathcal{X}$ and $\mathcal{Y}$ be sets. We start by recalling two theorems of Zhandry [16]:

Theorem VII.1. *Let A be a quantum algorithm making q quantum queries to an oracle $H : \mathcal{Y}^{\mathcal{X}}$. If we draw H from some distribution D, then for every z, the quantity $\Pr_{H \leftarrow D}[A^{|H\rangle}() = z]$ is a linear combination of the quantities $\Pr_{H \leftarrow D}[H(x_i) = r_i \forall i \in \{1, ..., 2q\}]$ for all possible settings of the x_i and r_i.*

Theorem VII.2. *Fix q, and let D_λ be a family of distributions on $\mathcal{Y}^{\mathcal{X}}$ indexed by $\lambda \in [0, 1]$. Suppose there is an integer d such that for every $2q$ pairs $(x_i, r_i) \in \mathcal{X} \times \mathcal{Y}$, the function $p(\lambda) = \Pr_{H \leftarrow D_\lambda}[H(x_i) = r_i \forall i \in \{1, ..., 2q\}]$ is a polynomial of degree at most d in λ. Then for any quantum algorithm A making q quantum queries, the output distributions under D_λ and D_0 are $2\lambda d^2$-close.*

We now show a similar result:

Theorem VII.3. *Fix q, and let E_r be a family of distributions on $\mathcal{Y}^{\mathcal{X}}$ indexed by $r \in \mathbb{Z}^+ \bigcup \{\infty\}$. Suppose there is an integer d such that for every $2q$ pairs $(x_i, r_i) \in \mathcal{X} \times \mathcal{Y}$, the function $p(\lambda) = \Pr_{H \leftarrow E_{1/\lambda}}[H(x_i) = r_i \forall i \in \{1, ..., 2q\}]$ is a polynomial of degree at most d in λ. Then for any quantum algorithm A making q quantum queries, the output distributions under E_r and E_∞ are $\pi^2 d^3/3r$-close.*

Sketch of Proof. Let $D_\lambda = E_{1/\lambda}$. We see that the conditions of Theorems VII.3 and VII.2 are identical, with the following exception: Theorem VII.2 requires D_λ to be a distribution for all $\lambda \in [0, 1]$, while Theorem VII.3 only requires D_λ to be a distribution when $1/\lambda$ is an integer (and when $\lambda = 0$). The proof is thus similar in flavor to that of Theorem VII.2, with the following exception: the proof of Theorem VII.2 uses well-known bounds on polynomials f where $f(x) \in [0, 1]$ for all $x \in [0, 1]$. However, we need similar bounds on polynomials f where $f(x)$ is only required to be in $[0, 1]$ for x where $1/x$ is an integer. Such polynomials are far less understood, and we need to prove suitable bounds under these relaxed assumptions. The proof is in the full version [17]. ∎

In the next section, we apply Theorem VII.3 to a new class of distributions.

A. Small-Range Distributions

We now apply Theorem VII.3 to a new distribution on oracles, which we call small-range distributions. Given a distribution D on $\mathcal{Y}$, define $\mathsf{SR}_r^D(\mathcal{X})$ as the following distribution on functions from $\mathcal{X}$ to $\mathcal{Y}$:

- For each $i \in [r]$, chose a random value $y_i \in \mathcal{Y}$ according to the distribution D.
- For each $x \in \mathcal{X}$, pick a random $i \in [r]$ and set $O(x) = y_i$.

We will often omit the domain $\mathcal{X}$ when is is clear from context.

The following is proved in the full version [17]:

Lemma VII.4. *Fix k. For any $\mathcal{X}$, the probabilities in each of the marginal distributions of $\mathsf{SR}_r^D(\mathcal{X})$ over k inputs are polynomials in $1/r$ of degree k.*

An alternate view of this function is to choose $g \leftarrow D^{[r]}$ and $f \leftarrow [r]^{\mathcal{X}}$, and output the composition $g \circ f$. That is, $\mathsf{SR}_r^D(\mathcal{X}) = D^{[r]} \circ [r]^{\mathcal{X}}$. In other words, we choose a random function f from $\mathcal{X}$ to $[r]$, and compose it with another random function g from $[r]$ to $\mathcal{Y}$, where outputs are distributed according to D. We call this distribution a small-range distribution because the set of images of any function drawn from the distribution is bounded to at most r points, which for $r << \mathcal{Y}$ will be a small subset of the co-domain. Notice that, as r goes to infinity, f will be injective with probability 1, and hence for each x, $g(f(x))$ will be distributed independently according to D. That is, $\mathsf{SR}_\infty^D(\mathcal{X}) = D^{\mathcal{X}}$. We can then use Theorem VII.3 to bound the ability of any quantum algorithm to distinguish $\mathsf{SR}_r^D(\mathcal{X})$ from $\mathsf{SR}_\infty^D(\mathcal{X}) = D^{\mathcal{X}}$:

Corollary VII.5. *The output distributions of a quantum algorithm making q quantum queries to an oracle either drawn from $\mathsf{SR}_r^D(\mathcal{X})$ or $D^{\mathcal{X}}$ are $\ell(q)/r$-close, where $\ell(q) = \pi^2(2q)^3/3 < 27q^3$.*

We observe that this bound is tight: in the full version we show that the quantum collision finding algorithm of Brassard, Høyer, and Tapp [5] can be used to distinguish $\mathsf{SR}_r^D(\mathcal{X})$ from $D^{\mathcal{X}}$ with optimal probability. This shows that Theorem VII.3 is tight.

B. The Equivalence of Indistinsguishability and Oracle-Indistinguishability

We now use the above techniques to explore the relationship between indistinguishability and oracle-

indistinguishability and to prove Theorem I.1. Clearly, oracle-indistinguishability implies standard indistinguishability: if A distinguishes D_1 from D_2, then the algorithm $B^{|O\rangle}()$ that picks any $x \in \mathcal{X}$ and returns $A(O(x))$ breaks the oracle-indistinguishability.

In the other direction, in the classical world, if B makes q queries to O, we can simulate O using q samples, and do a hybrid across the samples. This results in an algorithm that breaks the standard indistinuishability. However, in the quantum world, each query might be over a superposition of exponentially many inputs. Therefore there will be exponentially many hybrids, causing the proof to fail.

In the statistical setting, this question has been answered by Boneh et al. [2]. They show that if a (potentially unbounded) quantum adversary making q queries distinguishes $D_1^{\mathcal{X}}$ from $D_2^{\mathcal{X}}$ with probability ϵ, then D_1 and D_2 must $\Omega(\epsilon^2/q^4)$-far. We now have to tools to extend this result to the computational setting (and improve the result for the statistical setting in the process) by proving Theorem I.1.

Proof of Theorem I.1. Let B be an (efficient) quantum adversary that distinguishes $D_1^{\mathcal{X}}$ from $D_2^{\mathcal{X}}$ with non-negligible probability ϵ, for distributions D_1 and D_2 over $\mathcal{Y}$. That is, there is some set $\mathcal{X}$ such that

$$\left| \Pr_{O \leftarrow D_1^{\mathcal{X}}}[B^{|O\rangle}() = 1] - \Pr_{O \leftarrow D_2^{\mathcal{X}}}[B^{|O\rangle}() = 1] \right| = \epsilon \ .$$

Our goal is to construct an (efficient) quantum algorithm A that distinguishes a sample of D_1 from a sample of D_2. To this end, choose r so that $\ell(q)/r = \epsilon/4$, where $\ell(q)$ is the polynomial from Corollary VII.5. That is, $r = 4\ell(q)/\epsilon$. No quantum algorithm can distinguish $\mathsf{SR}_r^{D_i}(\mathcal{X})$ from $D_i^{\mathcal{X}}$ with probability greater than $\ell(q)/r = \epsilon/4$. Thus, it must be that the quantity

$$\left| \Pr_{O \leftarrow \mathsf{SR}_r^{D_1}(\mathcal{X})}[B^{|O\rangle}() = 1] - \Pr_{O \leftarrow \mathsf{SR}_r^{D_2}(\mathcal{X})}[B^{|O\rangle}() = 1] \right|$$

is at least $\epsilon/2$. We now define $r+1$ hybrids H_i as follows: For $j = 0, ..., i-1$, draw y_j from D_1. For $j = i, ..., r-1$, draw y_j from D_2. Then give B the oracle O where for each x, $O(x)$ is a randomly selected y_i. H_r is the case where $O \leftarrow \mathsf{SR}_r^{D_1}$, and H_0 is the case where $O \leftarrow \mathsf{SR}_r^{D_2}$. Hence H_0 and H_r are distinguished with probability at least $\epsilon/2$. Let

$$\epsilon_i = \Pr_{O \leftarrow H_{i+1}}[B^{|O\rangle}() = 1] - \Pr_{O \leftarrow H_i}[B^{|O\rangle}() = 1]$$

be the probability that B distinguishes H_{i+1} from H_i. Then $|\sum_{i=1}^{r} \epsilon_i| \geq \epsilon/2$.

We construct an algorithm A that distinguishes between D_1 and D_2 with probability $\epsilon/2r$. A, on inputs y, does the following:

- Choose a random $i \in [r]$.
- Construct a random oracle $O_0 \leftarrow [r]^{\mathcal{X}}$.
- Construct random oracles $O_1 \leftarrow D_1^{\{0,...,i-1\}}$ and $O_2 \leftarrow D_2^{\{i+1,...,r-1\}}$.
- construct the oracle O where $O(x)$ is defined as follows:
  - Compute $j = O_0(x)$.
  - If $j = i$, output y.
  - Otherwise, if $j < i$, output $O_1(j)$ and if $j > i$, output $O_2(j)$.
- Simulate B with the oracle O, and output the output of B.

Let A_i be the algorithm A using i. If $y \leftarrow D_1$, B sees hybrid H_{i+1}. If $y \leftarrow D_2$, B sees H_i. Therefore, we have that

$$\Pr_{y \leftarrow D_1}[A_i(y) = 1] - \Pr_{y \leftarrow D_2}[A_i(y) = 1] = \epsilon_i \ .$$

Averaging over all i, we get that A's distinguishing probability, $|\Pr_{y \leftarrow D_1}[A(y) = 1] - \Pr_{y \leftarrow D_2}[A(y) = 1]|$, is equal to

$$\left| \frac{1}{r} \sum_{i=1}^{r} \epsilon_i \right| \geq \frac{\epsilon}{2r} = \frac{\epsilon^2}{8\ell(q)} \ .$$

Thus, A is an (efficient) algorithm that distinguishes D_1 from D_2 with non-negligible probability. Hence, it breaks the indistinguishability of D_1 and D_2. ■

Notice that by removing the requirement that B be an efficient algorithm, we get a proof for the statistical setting as well, so that if any computationally unbounded quantum algorithm making q quantum queries distinguishes $D_1^{\mathcal{X}}$ from $D_2^{\mathcal{X}}$ with probability ϵ, then D_1 and D_2 must be $\Omega(\epsilon^2/\ell(q)) = \Omega(\epsilon^2/q^3)$-far, improving the result of Boneh et al. by a factor of q.

Now that Theorem I.1 is proved, we have completed the proof of security for the GGM construction (Construction 1) in the quantum setting. With some modifications to the proof, we can also prove prove the quantum security for Constructions 2 and 3, as shown in the full version [17].

VIII. Conclusion

We have shown that not all pseudorandom functions secure against classical queries are also secure against quantum queries. Nevertheless, we demonstrate the security of several constructions of pseudorandom functions against quantum queries. Specifically, we show that the construction from pseudorandom generators [7], the construction from pseudorandom synthesizers [11], and the direct construction based on the Learning With Errors problem [1] are all secure against quantum algorithms making quantum queries. We accomplish these results by providing more tools for bounding the ability of a

quantum algorithm to distinguish between two oracle distributions. We leave as an open problem proving the quantum security of some classical uses of pseudorandom functions. We have two specific instances in mind:

- Pseudorandom permutations (Block Ciphers) secure against quantum queries. We know how to build pseudorandom permutations from pseudorandom functions in the classical setting ([10], [13]). Classically, the first step to prove security is to replace the pseudorandom functions with truly random functions, which no efficient algorithm can detect. The second step is to prove that no algorithm can distinguish this case from a truly random permutation. For this construction to be secure against quantum queries, a quantum-secure pseudorandom function is clearly needed. However, it is not clear how to transform the second step of the proof to handle quantum queries.
- Message Authentication Codes (MACs) secure against quantum queries. MACs can be built from pseudorandom functions and proven existentially unforgeable against a classical adaptive chosen message attack. If we allow the adversary to ask for an authentication on a superposition of messages, a new notion of security is required. One possible definition of security is that, after q queries, no adversary can produce $q+1$ classical valid message/tag pairs. Given a pseudorandom function secure against quantum queries, proving this form of security reduces to proving the impossibility of the following: After q quantum queries to a random oracle O, output $q+1$ input/output pairs of O with non-negligible probability.

Acknowledgments

We would like to thank Dan Boneh for his guidance and many insightful discussions. This work was supported by NSF and DARPA. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Department of Defense or the U.S. Government.

References

[1] A. Banerjee, C. Peikert, and A. Rosen. Pseudorandom Functions and Lattices. *Advances in Cryptology — EUROCRYPT 2012*, pages 1–26, 2011.

[2] D. Boneh, Ö. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, and M. Zhandry. Random Oracles in a Quantum World. In *Advances in Cryptology — ASIACRYPT 2011*, 2011.

[3] D. Boneh and R. J. Lipton. Quantum Cryptanalysis of Hidden Linear Functions. *Advances in Cryptology — CRYPTO 1995*, 1995.

[4] D. Boneh, H. Montgomery, and A. Raghunathan. Algebraic Pseudorandom Functions with Improved Efficiency from the Augmented Cascade. *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS)*, pages 1–23, 2010.

[5] G. Brassard, P. Høyer, and A. Tapp. Quantum Algorithm for the Collision Problem. *ACM SIGACT News (Cryptology Column)*, 28:14–19, 1997.

[6] Y. Dodis and A. Yampolskiy. A Verifiable Random Function with Short Proofs and Keys. *Public Key Cryptography*, 2005.

[7] O. Goldreich, S. Goldwasser, and S. Micali. How to Construct Random Functions. *Journal of the ACM (JACM)*, 33(4):792–807, 1986.

[8] J. Håstad, R. Impagliazzo, L. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[9] A. B. Lewko and B. Water. Efficient Pseudorandom Functions From the Decisional Linear Assumption and Weaker Variants. *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS)*, pages 1–17, 2009.

[10] M. Luby and C. Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM J. Comput.*, 17(2):373–386, Apr. 1988.

[11] M. Naor and O. Reingold. Synthesizers and Their Application to the Parallel Construction of Pseudo-Random Functions. *Proceedings of the 36th IEEE Symposium on Foundations of Computer Science (FOCS)*, 1995.

[12] M. Naor and O. Reingold. Number-Theoretic Constructions of Efficient Pseudo-Random Functions. *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science (FOCS)*, 51(2):231–262, 1997.

[13] M. Naor and O. Reingold. On the Construction of Pseudorandom Permutations : Luby-Rackoff Revisited. *Journal of Cryptology*, (356):29–66, 1999.

[14] M. Naor, O. Reingold, and A. Rosen. Pseudo-Random Functions and Factoring. *Proceedings of the 32nd Annual ACM Symposium on the Theory of Computing (STOC)*, pages 1–23, 2000.

[15] M. A. Nielsen and I. Chuang. Quantum Computation and Quantum Information. *American Journal of Physics*, 70(5):558, 2000.

[16] M. Zhandry. Secure Identity-Based Encryption in the Quantum Random Oracle Model. In *Advances in Cryptology — CRYPTO 2012*, 2012. Full version available at the Cryptology ePrint Archives: http://eprint.iacr.org/2012/076/.

[17] M. Zhandry. How to Construct Quantum Random Functions, April 2012. Full version available at the Cryptology ePrint Archives: http://eprint.iacr.org/2012/182/.