CRYPTOGRAPHY IN THE AGE OF QUANTUM COMPUTERS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Mark Zhandry
May 2015

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Dan Boneh) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Gregory Valiant)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Ryan Williams)

Approved for the Stanford University Committee on Graduate Studies

Abstract

It is well established that full-fledged quantum computers, when realized, will completely break many of today's cryptosystems. This looming threat has led to the proposal of so-called "post-quantum" systems, namely those that appear resistant to quantum attacks. We argue, however, that the attacks considered in prior works model only the near future, where the attacker may be equipped with a quantum computer, but the end-users implementing the protocols are still running classical devices.

Eventually, quantum computers will reach maturity and everyone — even the end-users — will be running quantum computers. In this event, attackers can interact with the end-users over quantum channels, opening up a new set of attacks that have not been considered before. This thesis puts forth new security models and new security analyses showing how to ensure security against such quantum channel attacks. In particular, we re-build many core cryptographic functionalities, including pseudorandom functions, encryption, digital signatures, and more, resulting in the first protocols that are safe to use in a ubiquitous quantum computing world. Along the way, we resolve several open problems in quantum query complexity, such as the Collision Problem for random functions, the Set Equality Problem, and the Oracle Interrogation Problem.

Acknowledgments

I would like to begin by thanking my advisor, Dan Boneh. I am so grateful for all of his help, support, insights, and guidance throughout my Ph.D. journey. From helpfully tearing apart my first paper so that I could get into my first conference to walking me through the faculty application process, Dan has been an unparalleled mentor, colleague, and friend. I truly respect and appreciate his knowledge, judgment, and generosity. As I begin the next phase of my academic career, one of my guiding principles will be "what would Dan do?"

I am thankful for the Theory and Applied Crypto groups at Stanford for all the great lunches and reading groups we have had over the years. Thank you for sharing interesting research and for patiently letting me practice my talks on you. I have learned a lot from you all and value your friendships.

I would like to thank the IBM Crypto group, Microsoft Research New England, and MIT's Computer Science and Artificial Intelligence Lab (CSAIL) for hosting me during various times during the final year of my Ph.D. program. I would especially like to acknowledge Tal Rabin, Yael Kalai, and Vinod Vaikuntanathan for being my hosts and de-facto advisors while I was away from Stanford. You made an academic home away from home for me.

I would also like to thank my collaborators, both on works that are in this thesis and outside. The list is too long to enumerate here, but I would particularly like to recognize Amit Sahai, Sanjam Garg, Brent Waters, Craig Gentry, Shai Halevi, Scott Aaronson, Daniel Wichs, Andrew Childs, Adam O'Neill, Nir Bitansky, Omer Paneth, and many more. It has been an honor and a pleasure to work with and learn from you. You have pushed me to become a better researcher and collaborator, and I continue to look up to you as role models.

Lastly, I would like to thank my family for their unconditional love and support through the years. My parents encouraged my passion for learning from an early age, teaching my five year-old self algebra and my nine year-old self how to blow up transistors. I am also lucky to have a wife who has been by my side since the first days of college and adulthood. She has kept me sane, motivated, happy, healthy, organized, and grammatically correct. I would not be where I am or who I am today without her.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Quantum physics has the potential to fundamentally change the face of cryptography. On the negative side, quantum computers can break many existing cryptosystems [Sho94], in particular most public key systems used today. On the positive side, new quantum protocols will, for example, allow parties to securely exchange keys without relying on any computational assumptions [BB84]. However, even in a world where quantum computers are a part of daily life, classical cryptosystems will still be required. For example, classical storage will likely be far more economical than quantum storage, and thus classical encryption protocols will be needed for cost-effective encrypted hard-drives. Therefore, it is important to understand the feasibility of classical cryptosystems in a quantum world.

Any post-quantum cryptosystem — that is, a classical system that is immune to quantum attacks — must have an underlying computational problem that is difficult for quantum computers. Hard problems on lattices, such as finding the shortest vector in a high-dimensional lattice, have been used to build a variety of cryptosystems. Moreover, the these hard problems have so far resisted quantum attacks, and some evidence suggests that they may be hard for quantum computers [Reg02]. Therefore, a promising choice for a post-quantum system is one based on lattices.

While many systems based on lattices have been proposed, their security is typically only analyzed against classical adversaries. This means the attacks considered and the proof techniques used are classical. Unfortunately, security against such classical attacks does not immediately imply security against quantum attacks, even when the underlying computational problem is hard for quantum computers. This leads to very important questions including:

How should we model quantum attacks on classical cryptosystems?

How do we build systems secure against such attacks?

How do we analyze such systems and argue their security?

1.1 Our Contributions

In this thesis, we address the questions above by giving new quantum security models for classical protocols, and new constructions and analysis techniques for these models.

1.1.1 New quantum security models

First, we argue that the classical analyses mentioned above can typically be translated into *some* security against quantum adversaries. However, we argue that the resulting security analysis only reflects the near future, where end-users are still running classical devices and communicating over classical channels, but the users want to protect their communication from new quantum threats.

In more detail, the *classical* security analysis of a lattice-based cryptosystem consists of three parts: an underlying hard problem on lattices (such as finding the shortest vector), a well-defined classical security model, and a reduction showing how to convert any classical adversary breaking security in the model into a classical algorithm breaking the supposedly hard lattice problem. Thus, if the underlying problem is hard for classical computers, no such classical adversary can exist in the model. For example, for encryption, a common security model is chosen plaintext security, where the adversary is trying to decrypt a ciphertext, and in order to help him achieve this task, the adversary has access to an encryption oracle that encrypts arbitrary plaintexts of the adversary's choice. To prove classical security in this model, one would show how to convert any efficient classical adversary that can decrypt given access to an encryption oracle into an efficient classical algorithm for the shortest vector problem (SVP).

Typically, such proofs work in a "black box" manner, only looking at the inputs and outputs of the adversary (where oracle queries are considered part of the input/output behavior), and not how the adversary is actually implemented. Thus, we can usually translate such proofs immediately into quantum security proofs. The result is a conversion from any *quantum* adversary that can decrypt given (still classical) access to an encryption oracle into an efficient quantum algorithm for SVP. Thus, if SVP is hard for quantum computers, no such quantum adversary can exist. Note that since oracle queries are part of the input/output behavior of the adversary, this translation can only rule out adversaries with *classical* access to the encryption oracle, as the proofs are only guaranteed to work for classical input/output behavior.

Generally, translating classical security proofs to the quantum setting captures the attacks where the adversary has a quantum computer, but is limited to interacting with the protocol using classical communication. Intuitively, the adversary is only able to take advantage of quantum physics to enhance his *computational* power. This captures the near-future scenario where end-users are still implementing their systems on classical devices, which inherently cannot communicate over quantum channels. Thus, a quantum adversary trying to break into the device is limited to classical interaction. This is the setting generally considered in "post-quantum" cryptography.

Quantum channel attacks. A natural question is, then, what happens if the adversary *can* attack the system over a quantum channel. Such attacks are likely in a world where even end-users are using quantum devices, and hence can potentially interact with the adversary using quantum communication. Even if the end-users are not running full-fledged quantum computers, their devices may exhibit some quantum effects. For example, modern processors have reached the point where quantum tunneling is an important consideration.

What are the right ways to model these enhanced quantum interaction — or quantum channel — attacks? The natural approach is to take cues from the classical security definitions, and replace the adversary's classical interaction with quantum interaction. For encryption as mentioned above, this approach would yield a model where the adversary is given *quantum* access to an encryption oracle to help it decrypt the ciphertext. This means the adversary can query the oracle on quantum *superpositions* of messages: quantum states that are simultaneously every message in the message space. In response, the adversary receives a corresponding superposition of encryptions of all messages. Seeing superpositions of exponentially-many ciphertexts may give the adversary extra information to help it break the protocol.

Our new models. We give new quantum security models for the following systems:

- **Quantum random oracle model.** In some cases, even in the post-quantum setting, quantum interaction is possible. One example is the *random oracle model*. Here, the protocol and the attacker have black box access to a random function oracle, and security is proved in this setting. The random oracle model is a heuristic model for hash functions; ultimately when the protocol is implemented the random oracle is replaced by a concrete hash function. In this case, the adversary evaluates the hash function for himself and can, for example, evaluate the hash function on quantum superpositions of inputs. Therefore, in our heuristic oracle model, quantum interaction with the oracle must be allowed, even if the rest of the adversary's interaction with the rest of the protocol is classical. We call the resulting model the *quantum random oracle model*.

- **Pseudorandom functions.** Pseudorandom functions (PRFs) are one of the fundamental building blocks in symmetric key cryptography. Roughly, a PRF is a keyed function that, when given (classical) oracle access to the function (but not the key), the function is indistinguishable from a truly random function. We define *quantum* pseudorandom functions (QPRFs) as PRFs that are secure even when given *quantum* oracle access to the function. Aaronson [Aar09] first raised the question of constructing QPRFs.

- **Message authentication codes and signatures.** Message authentication codes (MACs) are used to ensure data integrity. Messages can be signed using a secret key, and signatures — called tags — are verified using the same key. Classical security stipulates that, even after

seeing tags for many messages, it is computationally infeasible to forge a tag on a new message. The straightforward generalization of this attack allows the adversary to obtain signatures on superpositions of messages. Unfortunately, as the superposition of messages may include *every* message in the message space, it is not clear what a "new" message should be. In particular, the adversary can obtain signatures on a uniform superposition of all messages, measure or observe the superposition, and then obtain a tag on a random message. He can even do this for each one of his signing queries, resulting in q tags on random messages, where q is the number of queries made. We therefore propose a new notion of a forgery, where we require the adversary to produce tags on $q + 1$ messages. Such a forgery notion is similar to what is used for blind signatures [PS96].

A signature scheme is a publicly verifiable version of a MAC. We define security for signatures analogously.

- **Encryption.** As mentioned above, one way to define security for encryption is to allow the adversary an encryption oracle to help it decrypt a plaintext. Actually, the classical definition requires that the adversary cannot distinguish an encryption of a message m_0 from an encryption of message m_1, where the "challenge" messages m_0, m_1 are chosen by the adversary. This reflects the fact that we consider even leaking a single bit of information a break of the system, and that the adversary may have some influence over the messages being sent.

  One way to translate this notion into the quantum setting is to allow quantum access to the encryption oracle, *and* to allow the adversary's challenge messages to actually be quantum superpositions of messages. This means the adversary is trying to distinguish one superposition of ciphertexts from another. Unfortunately, the natural way of formalizing this definition yields notions that can always be broken. Thus such notions are not useful.

  Instead, we define security using the arguably more natural approach, where the challenge messages remain classical, but the adversary still has quantum access to the encryption oracle. We believe this is a reasonable relaxation, as it seems to capture the setting where the end-user is using the protocol to encrypt *classical* messages, even though he is running a quantum computer. If the user wishes to encrypt a quantum state, an actual quantum protocol should probably be used (such as in [BCG$^+$02]).

  Our definition also allows us to give a natural notion of chosen ciphertext security, where the adversary is additionally given quantum access to a decryption oracle.

1.1.2 New Quantum Proof Techniques

With our new security models in hand, we now ask the question: are any existing post-quantum schemes secure in our models? To answer this question, we would need to open up the classical proofs, and see if the proofs work even when the adversary is allowed quantum interaction.

Case study: pseudorandom functions. A pseudorandom function (PRF) is a keyed function $\mathsf{PRF} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ with the property that, for a random key $k \xleftarrow{R} \mathcal{K}$, the function $F(x) = \mathsf{PRF}(k, x)$ is computationally indistinguishable from a truly random function $F : \mathcal{X} \to \mathcal{Y}$ when given only oracle access to F. Since we require the adversary to run in polynomial time, it clearly can only make a polynomial number of queries. In other words, the adversary can only "see" a polynomial number of outputs of the function. When we move to the quantum setting, we allow the adversary to make quantum queries to F. Now, even if the adversary makes only a single query, it could potentially be a query on a superposition of all exponentially many inputs. Thus, while the number of queries is still a polynomial, the number of outputs "seen" by the adversary is potentially exponential.

It is possible to conjecture, say, that any given PRF candidate is quantum secure — for example, the Advanced Encryption Standard (AES) could be a quantum pseudorandom function. However, for known constructions of PRFs from simpler primitives, the security analysis inherently requires that the adversary only "sees" a polynomial number of outputs, and therefore these analyses are insufficient for quantum security. This limitation, in the case of the GGM PRF [GGM86], was observed by Aaronson [Aar09].

In more detail, Goldreich, Goldwasser, and Micali [GGM86] are the first to define (classical) pseudorandom functions, and show how to construct PRFs from any pseudorandom generator (PRG). One way to view their construction is as a technique to expand the domain size of a PRF. Suppose we have $\mathsf{PRF}_1 : \mathcal{K} \times \mathcal{X} \to \mathcal{K}$ and $\mathsf{PRF}_2 : \mathcal{K} \times \{0,1\} \to \mathcal{K}$. Then we can construct a new PRF $\mathsf{PRF} : \mathcal{K} \times (\mathcal{X} \times \{0,1\}) \to \mathcal{K}$ where $\mathsf{PRF}(k, (x, b)) = \mathsf{PRF}_2(\mathsf{PRF}_1(x), b)$. That is, we use PRF_1 to derive a key for PRF_2, which we then evaluate on the bit b. PRFs where the domain size is a constant, such as PRF_2, are better known as pseudorandom generators. By iterating this construction many times starting with $\mathcal{X} = \{0,1\}$, the GGM construction is obtained.

In the classical setting, the GGM proof proceeds by defining a few hybrid experiments.

- **Hybrid 0.** This is the case where the adversary is given oracle access to the oracle $(x, b) \longmapsto \mathsf{PRF}(k, (x, b)) = \mathsf{PRF}_2(\mathsf{PRF}_1(k, x), b)$ for a random key k.
- **Hybrid 1.** In this case, the adversary is given a "hybrid" oracle. A truly random function G is chosen, and the adversary is given the oracle $(x, b) \longmapsto \mathsf{PRF}_2(G(x), b)$.
- **Hybrid 2.** Now the adversary is given a truly random oracle $(x, b) \longmapsto F(x, b)$ for a truly random function F.

PRF is secure if **Hybrid 0** and **Hybrid 2** are indistinguishable. Notice that the difference between **Hybrid 0** and **Hybrid 1** is that the function $x \longmapsto \mathsf{PRF}_1(k, x)$ for a random k is replaced with a truly random oracle G. Thus, the security of PRF_1 shows that these hybrids are indistinguishable. It then suffices to prove that **Hybrid 1** and **Hybrid 2** are indistinguishable, which implies that **Hybrid 0** and **Hybrid 2** are indistinguishable.

To that end, recall that the (classical) adversary only makes a polynomial number of classical queries. Let q be the number of queries. Suppose for the moment that the set $S = \{x_1, \ldots, x_q\}$ of x values the adversary queries on is known in advance. Now, suppose that we have at our disposal q oracles $H_1, \ldots, H_q$ where either $H_i(b) = \mathsf{PRF}_2(k_i, b)$ for all i and for independently random k_i, or each H_i is an independently chosen truly random function. Then we can simulate the adversary: when the adversary queries on (x_i, b), we simply query $H_i(b)$ and send the response back to the adversary. In the case where H_i are PRF_2, the adversary sees **Hybrid 1**. In the case where the H_i are random functions, the adversary sees **Hybrid 2**. It is straightforward to generalize this to the setting where the x_i are not known in advance. Thus, to prove the indistinguishability of **Hybrids 1** and **2**, it suffices to show that the two cases for the sequence of H_i oracles are indistinguishable. If $q = 1$, then this follows from the security of PRF_2. For $q > 1$, a simple hybrid argument shows that the security also follows from PRF_2, though there is a security loss of a factor of q in the reduction. Polynomial losses in security are generally considered acceptable, so this completes the proof.

When we move to the quantum query setting, the indistinguishability of **Hybrid 0** and **Hybrid 1** goes through unaffected. However, the second part of the proof, showing that **Hybrid 1** and **Hybrid 2** are indistinguishable, breaks down. The issue is that the queries could be on superpositions of all messages, so the set S of queried x values is potentially the entire domain $\mathcal{X}$, which has exponential size. This means the straightforward proof has a security loss of a factor of $|\mathcal{X}|$ (as opposed to q), which is exponential. Such security losses are generally considered unacceptable, so a new proof is required.

Our new technique. Writing $\mathsf{PRF}_2(k) = (\mathsf{PRF}_2(k, 0), \mathsf{PRF}_2(k, 1))$, we see that we want to show the following: if $\mathsf{PRF}_2(k)$ for a random k is indistinguishable from a truly random string y, then the oracles $\mathsf{PRF}_2(G(\cdot))$ and $F(\cdot)$ where G and F are random functions are computationally indistinguishable, even given a polynomial number of quantum queries to the oracles. To argue this, we need a reduction that takes any algorithm $\mathcal{A}$ that distinguished $\mathsf{PRF}_2(G(\cdot))$ from F, and uses it to construct an algorithm $\mathcal{B}$ that can distinguish a single sample $\mathsf{PRF}_2(k)$ from a random string y. Using a simple hybrid argument, it even suffices to construct, for some polynomial function r, a distinguisher $\mathcal{B}$ that distinguishes r samples from the distribution $\mathsf{PRF}_2(\cdot)$ from r truly random samples.

How would such a distinguisher work? Somehow, it seems that $\mathcal{B}$ needs to simulate the oracle seen by $\mathcal{A}$ in such a way that if the r samples are from PRF_2, then the oracle is $\mathsf{PRF}_2(G(\cdot))$, and if the r samples are truly random, then the oracle is $F(\cdot)$. Perhaps the most natural approach is to choose, for every input x to $\mathcal{A}$'s oracle, one of the r samples, and set the output of the oracle on x to be that sample. Therefore, if the samples come from PRF_2, then the oracle is $x \longmapsto \mathsf{PRF}_2(k_{i_x})$ for random $i_x \in \{1, \ldots, r\}$, whereas if the samples are truly random, then the oracle is $x \longmapsto y_{i_x}$ for random i_x.

Unfortunately, such oracles are not the correct oracles we wanted to simulate. For example, consider the oracle given by y_{i_x}, which has only r possible outputs for a polynomial r, but has $|\mathcal{X}|$ inputs, which is exponential. We call such oracles *small-range* functions. Such small range functions have very many collisions, exponentially more than a truly random function. Therefore, if a collision in a small-range function can be found, the oracle can be distinguished from a random oracle. One may hope that by setting $r \gg q$, however, the adversary cannot find a collision in the oracle, and thus the oracle will hopefully be indistinguishable from a random function. Indeed, in the classical case, this works for $r > \Omega(q^2)$.

In the quantum query case, there are two problems. For starters, prior quantum collision lower bounds are for specific oracle classes, and do not apply to our small-range functions. Second, it may be that, while finding collisions in a small-range function is hard, they can still be distinguished from a random function by other means.

To remedy these limitations, we introduce new techniques for arguing the indistinguishability of oracles by quantum queries. Applying our techniques to small-range distributions, we obtain that for $r > \Omega(q^3)$, the adversary cannot distinguish the small range function $x \longmapsto y_{i_x}$ cannot be distinguished from a random function (which in particular implies that collisions cannot be found). We similarly obtain that $x \longmapsto \mathsf{PRF}_2(k_{i_x})$ is indistinguishable from $\mathsf{PRF}_2(G(\cdot))$ for $r > \Omega(q^3)$. Thus, our simulator $\mathcal{B}$ approximately simulates the oracle distributions $\mathcal{A}$ expects, and from this we show that $\mathcal{B}$ successfully distinguishes r samples from PRF_2 from r random strings. A simple hybrid argument shows how to obtain a distinguisher for a single sample.

Techniques for other primitives. Our new proof technique using small-range distributions is also useful for proving the security of message authentication codes, signatures, and encryption. However, in many cases, this technique alone is not enough, and additional techniques need to be developed. In Chapters 3 and 4, we summarize several of our new quantum proof techniques. In addition to being useful for fully quantum security proofs, these techniques are also useful for security proofs in the Quantum Random Oracle Model.

1.1.3 New Constructions

For pseudorandom functions, as we explained above, we show that many existing constructions are actually secure against quantum channel attacks. However, in many cases, we do not have security proofs for existing constructions. It may be that many of the existing constructions *are* secure, but need more advanced technique to prove their security. However, others may actually insecure against quantum channel attacks. For example, in Section 6.2.2, we show that the Carter-Wegman MAC is insecure against quantum query attacks.

In either case, we need new constructions that can be analyzed using our techniques. We give a variety of new constructions:

- We show that a minor modification to the Carter-Wegman MAC *is* secure against quantum queries.
- We give several generic conversions from post-quantum digital signatures into signatures secure against quantum queries, based on either concrete primitives or using the random oracle heuristic.
- We give new constructions of encryption in both the secret key and public key setting.

All of our new constructions are based on existing protocols, and have efficiency comparable to the prior schemes. Therefore, even in the case where we cannot prove the quantum channel security of existing schemes, we obtain quantum channel security with minimal overhead compared to post-quantum security.

1.1.4 Other Results.

Along the way to developing new security analyses for the quantum world, we resolve several open problem in quantum query complexity. These include:

- A tight bound of $\theta(N^{1/3})$ quantum query complexity of finding collisions for random functions. Prior results [AS04, Yue14] only analyzed the case where the output of the function is larger than its input, whereas our result works for arbitrary input/output sizes. Moreover, ours is the first analyses that is tight for the case of random functions.
- A tight lower bound of $\Omega(N^{1/3})$ for the element distinctness problem. The best prior lower bound was $\Omega(N^{1/5})$.
- An exact characterization of the oracle interrogation problem for random functions, where the goal is to produce k input/output pairs of the function using fewer than k queries. The only previously analyzed case was for single-bit functions due to van Dam [van98], who gave an attack requiring roughly $k/2$ queries. We give exact upper bounds and lower bounds for the general case. For constant-size outputs, we show that ck queries suffice to recover k outputs, for some constant $c < 1$ that depends on the output size. However, for very large outputs, k queries are required.

1.2 Related Work

Quantum random oracles. Quantum accessible random oracles have been used in several prior works. Brassard and Salvail [BS08] and Brassard et al. [BHK$^+$11] analyze variants of Merkle puzzles in the quantum random oracle model (QROM). Bennet et al. [BBBV97] show that relative to a random oracle, a quantum computer cannot solve all of NP.

Subsequent to our work, several other works have investigated classical cryptosystems in the quantum random oracle model. For example, Dagdelen et al. [DFG13] investigate the Fiat-Shamir transformation, arguing that a natural class of proof strategies cannot prove its security in the quantum random oracle model, though this class does not capture all of the proof strategies covered in this thesis. Ambainis, Rosmanis, and Unruh [ARU14] give more evidence of the difficulty of proving Fiat-Shamir by showing that, relative to an oracle, the protocol is insecure. Their proof relies on some of the techniques developed here. Unruh [Unr14] gives an alternative to the Fiat-Shamir transformation that *is* secure in the quantum random oracle model.

Quantum superposition attacks. Concurrently and independently of this work, Damgård et al. [DFNS14] consider quantum superposition attacks on classical cryptographic protocols. Specifically, they examine secret sharing and multiparty computation in a model where an adversary may corrupt a superposition of subsets of players, and build zero knowledge protocols that are secure, even when a dishonest verifier can issue challenges on superpositions.

General conditions for quantum security. Some progress toward identifying sufficient conditions under which classical protocols are also quantum immune has been made by Unruh [Unr10] and Hallgren et al. [HSS11]. Unruh shows that any scheme that is statistically secure in Cannetti's universal composability (UC) framework [Can01] against classical adversaries is also statistically secure against quantum adversaries. Hallgren et al. show that for many schemes, this is also true in the computational setting. These results, however, do not apply to the primitives studied here and do not consider quantum superposition attacks.

1.3 Publications Contained in This Thesis

The results in this thesis are based on results in the following published papers:

- *Random Oracles in a Quantum World*, with Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, and Christian Schaffner (in ASIACRYPT 2011).
- *Secure Identity-Based Encryption in the Quantum Random Oracle Model* (in CRYPTO 2012).
- *How to Construct Quantum Random Functions* (in FOCS 2012).
- *Quantum-Secure Message Authentication Codes*, with Dan Boneh (in EUROCRYPT 2013).
- *Secure Signatures and Chosen Ciphertext Security in a Quantum Computing World*, with Dan Boneh (in CRYPTO 2013).
- *A Note on the Quantum Collision and Set Equality Problems* (in Quantum Information and Computation).

Chapter 2

Preliminaries

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 write $[k] = \{1, ..., k\}$. We will sometimes associate the set $[2] = \{1, 2\}$ with the set of bits $\{0, 1\}$; using this notation, the set of n bit strings can be written as $[2]^n$. Let $x = x_1...x_n$ be a string of length n. We write $x_{[a,b]}$ to denote the substring $x_a x_{a+1}...x_b$.

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 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 $\mathbb{Z}$. In other words, exponents distribute: $(\mathcal{Y} \times \mathcal{Z})^{\mathcal{X}} = \mathcal{Y}^{\mathcal{X}} \times \mathcal{Z}^{\mathcal{X}}$.

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 functions in $\mathcal{Y}^{\mathcal{X}}$ 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.

2.1 Background on Quantum Computation

The quantum system A is a complex Hilbert space $\mathcal{H}$ with inner product $\langle\cdot|\cdot\rangle$. The state of a quantum system is given by a vector $\|\psi\rangle$ of unit norm ($\langle\psi|\psi\rangle = 1$). Given quantum systems $\mathcal{H}_1$ and $\mathcal{H}_2$, the joint quantum system is given by the tensor product $\mathcal{H}_1 \otimes \mathcal{H}_2$. Given $|\psi_1\rangle \in \mathcal{H}_1$ and $|\psi_2\rangle \in \mathcal{H}_2$, the product state is given by $|\psi_1\rangle|\psi_2\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2$. Given a quantum state $|\psi\rangle$ and an orthonormal basis $B = \{|b_0\rangle, ..., |b_{d-1}\rangle\}$ for $\mathcal{H}$, a measurement of $|\psi\rangle$ in the basis B results in a value b_i with probability $|\langle b_i|\psi\rangle|^2$, and the state $|\psi\rangle$ is "collapsed" to the state $|b_i\rangle$. We let $b \overset{B}{\leftarrow} |\psi\rangle$ denote the distribution on b_i obtained by sampling $|\psi\rangle$ in basis B. when the basis B is obvious from context, we omit B and write $b \leftarrow |\psi\rangle$

A unitary transformation over a d-dimensional Hilbert space $\mathcal{H}$ is a $d \times d$ matrix $\mathbf{U}$ such that $\mathbf{U}\mathbf{U}^\dagger = \mathbf{I}_d$, where $\mathbf{U}^\dagger$ represents the conjugate transpose. A quantum algorithm operates on a product space $\mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{work}$ and consists of n unitary transformations $\mathbf{U}_1, ..., \mathbf{U}_n$ in this space. $\mathcal{H}_{in}$ represents the input to the algorithm, $\mathcal{H}_{out}$ the output, and $\mathcal{H}_{work}$ the work space. A classical input x to the quantum algorithm is converted to the quantum state $|x, 0, 0\rangle$. Then, the unitary transformations are applied one-by-one, resulting in the final state

$$|\psi_x\rangle = \mathbf{U}_n ... \mathbf{U}_1 |x, 0, 0\rangle \ .$$

The final state is measured, obtaining (a, b, c) with probability $|\langle a, b, c|\psi_x\rangle|^2$. The output of the algorithm is b.

Efficient Quantum Algorithms. An *efficient* quantum algorithm is a quantum algorithm such that:

- the unitary matrices $\mathbf{U}_i$ above come from a finite "basis" set.
- Let the *size* of a Hilbert space $\mathcal{H}$ be the logarithm of the dimension: $\log_2 \dim \mathcal{H}$. Then the size of $\mathcal{H}_{out}$ and $\mathcal{H}_{work}$ is a polynomial in the size of $\mathcal{H}_{in}$. Typically, $\mathcal{H}_{in}, \mathcal{H}_{out}, \mathcal{H}_{work}$ will be be the tensor product of many qubits. In this case, the size of each space is just the number of qubits. Then this requirement becomes that the total number of qubits is a polynomial in the input size of the algorithm.
- The number of matrices, n, is a polynomial in the size of $\mathcal{H}_{in}$.

Quantum-accessible Oracles. We will implement an oracle $O : \mathcal{X} \to \mathcal{Y}$ by a unitary transformation $\mathbf{O}$ where

$$\mathbf{O}|x, y, z\rangle = |x, y + O(x), z\rangle$$

where $+ : \mathcal{X} \times \mathcal{X} \to \mathcal{X}$ is some group operation on $\mathcal{X}$. Suppose we have a quantum algorithm that makes quantum queries to oracles $O_1, ..., O_q$. Let $|\psi_0\rangle$ be the state of the algorithm before any

queries, and let $\mathbf{U}_1, ..., \mathbf{U}_q$ be the unitary transformations applied between queries. The final state of the algorithm will be

$$\mathbf{U}_q\mathbf{O}_q...\mathbf{U}_1\mathbf{O}_1|\psi_0\rangle$$

We can also have an algorithm make classical queries to O_i. In this case, the input to the oracle is measured before applying the transformation $\mathbf{O}_i$.

Fix an oracle $O : \mathcal{X} \to \mathcal{Y}$. Let $O^{(q)} : \mathcal{X}^q \to \mathcal{Y}^q$ be the oracle that maps $\mathbf{x}$ into $O(\mathbf{x}) = (O(x_1), O(x_2), ..., O(x_q))$. Observe that any quantum query to $O^{(q)}$ can be implemented using q quantum queries to O, where the unitary transformations between queries just permute the registers. We say that an algorithm that makes a single query to $O^{(q)}$ makes q non-adaptive queries to O. An efficient quantum oracle algorithm is one where the number of oracle queries is polynomial, and the computation between each query is efficient as defined above.

When $\mathcal{A}$ is allowed to make queries to an oracle O, we will often write $\mathcal{A}^{|O\rangle}(x)$ to denote the execution of $\mathcal{A}$ on input x with *quantum* oracle access to O. We will often write $\mathcal{A}^{O}(x)$ to denote the case where A only has classical access to O. If $\mathcal{A}$ is allowed quantum queries to oracles $O_1, O_2, \ldots,$ we write $\mathcal{A}^{|O_1\rangle,|O_2\rangle,\cdots}(x)$. Similar notation is used for multiple classical oracles, as well as a mix of classical and quantum oracles.

The Density Matrix. Suppose the state of a quantum system depends on some hidden random variable $z \in \mathcal{Z}$, which is distributed according to a distribution D. That is, if the hidden variable is z, the state of the system is $|\psi_z\rangle$. We can then define the density matrix of the quantum system as

$$\boldsymbol{\rho} = \sum_{z\in\mathcal{Z}} \Pr_D[z]|\psi_z\rangle\langle\psi_z|$$

Applying a unitary matrix $\mathbf{U}$ to the quantum state corresponds to the transformation

$$\boldsymbol{\rho} \to \mathbf{U}\boldsymbol{\rho}\mathbf{U}^\dagger$$

A partial measurement on some registers has the effect of zeroing out the terms in $\boldsymbol{\rho}$ where those registers are not equal. For example, if we have two registers x and y, and we measure the x register, then the new density matrix is

$$\boldsymbol{\rho}'_{x,y,x',y'} = \begin{cases} \boldsymbol{\rho}_{x,y,x',y'} & \text{if } x = x' \\ 0 & \text{otherwise} \end{cases}$$

The probability a quantum algorithm outputs x is simply the value in the xth diagonal entry of $\boldsymbol{\rho}$. Thus all statistical information about a quantum algorithm is contained in its density matrix.

2.2 Post-quantum Cryptographic Primitives

All of the cryptographic primitives we discuss are efficiently computable on a *classical* computer. However, we consider security only against *quantum* adversaries. All primitives we discuss implicitly involve a security parameter that affects running times, key spaces, message spaces, and parameter sizes. When we say a function is negligible, we mean negligible in the implicit security parameter. We also take efficient to mean polynomial-time in the security parameter (whether discussing efficient classical algorithms or efficient quantum computation).

2.2.1 Notions without interaction

First we describe security for those primitives whose definition involves a non-interactive adversary. This means the adversary $\mathcal{A}$ receives some input and tries to solve some task without any interaction. This is in contrast to the primitives discussed in Section 2.2.2, where security is defined using an interactive game where the adversary interacts with the primitive in question in order to solve its task. Since the security models here are non-interactive, there is no difference from the post-quantum setting where the protocols are implemented on a classical computer (but security is desired against quantum adversaries) and the fully quantum setting where the protocols are implemented on a quantum computer. Moreover, for primitives in this section, classical security analyses can typically be translated into post quantum analyses.

Definition 2.1. A secure *one-way function* is a classically efficient function $f : \mathcal{X} \to \mathcal{Y}$ that cannot be inverted efficiently on a quantum computer. That is, for any efficient quantum adversary $\mathcal{A}$,

$$\Pr[f(\ \mathcal{A}(f(x))\ ) = f(x) : x \xleftarrow{R} \mathcal{X}] < \mathsf{negl}$$

Definition 2.2. A secure *pseudorandom generator* is classically efficient function $g : \mathcal{X} \to \mathcal{Y}$ (with $|\mathcal{Y}| > |\mathcal{X}|$) that has outputs that are indistinguishable from random strings by quantum computers. That is, for any efficient quantum adversary $\mathcal{A}$,

$$|\Pr[\mathcal{A}(g(x)) = 1 : x \xleftarrow{R} \mathcal{X}] - \Pr[\mathcal{A}(y) : y \xleftarrow{R} \mathcal{Y}]| < \mathsf{negl}$$

The classical proof that one-way functions imply pseudorandom generators [HILL99] carries over to the quantum setting. Thus, one-way functions secure against quantum adversaries imply pseudorandom generators secure against quantum adversaries. The other direction is trivial.

Definition 2.3. A *collision-resistant hash function* is a collection $\mathcal{H}$ of classically efficient functions $h : \mathcal{X} \to \mathcal{Y}$ (with $|\mathcal{Y}| \ll |\mathcal{X}|$) for which finding collisions is intractable by efficient quantum computers. That is, for any efficient quantum adversary $\mathcal{A}$,

$$\Pr[h(m_0) = h(m_1) : h \xleftarrow{R} \mathcal{H}, (m_0, m_1) \xleftarrow{R} \mathcal{A}(h)] < \mathsf{negl}$$

Injective trapdoor functions. An *injective trapdoor function* is an injective one-way function for which there is a secret trapdoor that allows for efficient inverting.

Definition 2.4. An injective trapdoor function is a tuple $(\mathsf{Gen}, F, F^{-1})$ of classically efficient algorithms with associated spaces $\mathcal{X}, \mathcal{Y}$ such that

- $\mathsf{Gen}()$ is a randomized algorithm on no inputs that outputs a secret inversion key ik and public evaluation key ek
- $F(\mathsf{fk}, x)$ is a deterministic function that takes as input the public key function ek and an input $x \in \mathcal{X}$, and outputs some $y \in \mathcal{Y}$.
- $F^{-1}(\mathsf{ik}, y)$ is a deterministic function that takes as input the secret inversion key ik and an image point $y \in \mathcal{Y}$, and outputs some point $x \in \mathcal{X}$.
- F^{-1} is the inverse of F. That is,

  $$\Pr[F^{-1}(\mathsf{ik}, F(\mathsf{ek}, x)) = x : (\mathsf{ik}, \mathsf{ek}) \xleftarrow{R} \mathsf{Gen}(), x \xleftarrow{R} \mathcal{X}] = 1$$

  In particular, this means $F(\mathsf{ek}, \cdot)$ must be injective.
- F is one-way without the secret key. That is, for any efficient quantum algorithm $\mathcal{A}$,

  $$\Pr[\mathcal{A}(\mathsf{ek}, y) = x : (\mathsf{ik}, \mathsf{ek}) \xleftarrow{R} \mathsf{Gen}(), x \xleftarrow{R} \mathcal{X}, y \leftarrow F(\mathsf{ek}, x)] < \mathsf{negl} \tag{2.2.1}$$

A trapdoor permutation is an injective trapdoor function where $\mathcal{Y} = \mathcal{X}$ (and in particular, this means $F(\mathsf{ek}, \cdot)$ is a permutation).

Post-quantum injective trapdoor functions where built by Peikert and Waters [PW08] based on the hardness of the Learning With Errors problem, a standard lattice assumption. We note that the Peikert-Waters scheme departs slightly from the notion above: in their notion, there is also a sampling algorithm that samples from some distribution D on $\mathcal{X}$. Security is defined with respect to inputs x drawn from D, instead of uniform x. That is, $x \xleftarrow{R} \mathcal{X}$ in Equation 2.2.1 is replaced with $x \xleftarrow{R} D$. It is generally straightforward to modify protocols using "ideal" injective trapdoor functions into protocols using this relaxed notion.

Until recently, the only known trapdoor *permutations* where based on factoring, and are therefore insecure in the quantum setting. A very recent work of Bitansky, Paneth, and Wichs [BPW15] shows how to build trapdoor permutations from program obfuscation. Some existing obfuscation constructions [GGH+13] are not known to be susceptible to quantum attacks. However, the constructions are extremely impractical, and the security is not very well understood. Therefore, no *practical* post-quantum trapdoor permutations currently exist.

Pre-image Sampleable Functions. Whereas injective trapdoor functions can be thought of as a relaxation of trapdoor permutations to arbitrary injective functions, pre-image sampleable functions (PSFs) can be thought of as a relaxation of trapdoor permutations to arbitrary *surjective* functions. However, we will make some additional restrictions on PSFs which will make them very useful in several scenarios.

Definition 2.5. A pre-image sampleable function (PSF) is a tuple of classically efficient algorithms $(\mathsf{Gen}, F, F^{-1})$ with associated spaces $\mathcal{X}, \mathcal{Y}$ such that

- $\mathsf{Gen}()$ is a randomized algorithm on no inputs that outputs a secret inversion key ik and public function key fk
- $F(\mathsf{fk}, x)$ is a deterministic function that takes as input the public key fk and an input $x \in \mathcal{X}$, and outputs some $y \in \mathcal{Y}$.
- $F^{-1}(\mathsf{ik}, y)$ is a randomized function that takes as input the secret key ik and an image point $y \in \mathcal{Y}$, and samples some point $x' \in \mathcal{X}$.
- F^{-1} inverts F. This means that the distribution $x \xleftarrow{R} F^{-1}(\mathsf{ik}, y)$ is the same as the conditional distribution $x \xleftarrow{R} \mathcal{X}$ conditioned on $F(\mathsf{fk}, x) = y$. In particular, this means

$$\Pr[F(\mathsf{fk}, F^{-1}(\mathsf{ik}, y)) = y : (\mathsf{ik}, \mathsf{fk}) \xleftarrow{R} \mathsf{Gen}(), y \xleftarrow{R} \mathcal{X}] = 1$$

- F is one-way without the secret key. That is, for any efficient quantum algorithm $\mathcal{A}$,

$$\Pr[F(\mathsf{fk}, \mathcal{A}(\mathsf{fk}, y)) = y : (\mathsf{ik}, \mathsf{fk}) \xleftarrow{R} \mathsf{Gen}(), y \xleftarrow{R} \mathcal{Y}] < \mathsf{negl}$$

Up to this point, PSFs are a generalization of trapdoor permutations. Now we add two additional requirements that make PSFs incomparable to trapdoor permutations:

- F has high pre-image min-entropy. That is, for every $y \in \mathcal{Y}$ and for every $(\mathsf{ik}, \mathsf{fk}) \xleftarrow{R} \mathsf{Gen}()$, the min-entropy of the distribution $F^{-1}(\mathsf{ik}, y)$ is at least least 1. We can also require even higher min-entropy, such as $-\log(\mathsf{negl})$.
- F is collision resistant. That is, for every efficient quantum adversary $\mathcal{A}$,

$$\Pr[F(\mathsf{fk}, x_0) = F(\mathsf{fk}, x_1) : (\mathsf{ik}, \mathsf{fk}) \xleftarrow{R} \mathsf{Gen}, (x_0, x_1) \xleftarrow{R} \mathcal{A}(\mathsf{fk})] < \mathsf{negl}$$

Gentry, Peikert, and Vaikuntanathan [GPV08] give a post-qantum pre-image sampleable function whose security is based on the hardness of the Short Integer Solution problem, a standard lattice assumption. Similar to the Peikert-Waters injective trapdoor function, the Gentry-Peikert-Vaikuntanathan PSFs depart somewhat from the ideal notion described above. First, security again

follows for x chosen from a non-uniform D. Moreover, F^{-1} only produces pre-images from a distribution statistically close to the desired distribution. However, this relaxation suffices for our purposes, though it makes the analysis more complicated. For ease of exposition, we describe all our protocols based on PSFs using the ideal notion described above.

Chameleon hash functions. A *chameleon hash function* is a special type of randomized hash function. Like plain hash functions, it is expected to be collision resistant and one-way. However, a chameleon hash function also comes hash a secret trap-door that allows inverting in a very strong sense. Namely, given the trap-door, any hash h, and any message m, it is possible to produce random coins that cause m to hash to h.

Definition 2.6. A chameleon hash function is a tuple of classically efficient algorithms $(\mathsf{Gen}, \mathsf{H}, \mathsf{Inv})$ with associated spaces $\mathcal{M}, \mathcal{Y}, \mathcal{R}$ such that:

- $\mathsf{Gen}()$ is a randomized algorithm on no inputs that outputs a secret inversion key sk and public function key fk
- $\mathsf{H}(\mathsf{fk}, m; r)$ is deterministic (when considering r as input), takes as input a public key fk, a message $m \in \mathcal{M}$, and randomness $r \in \mathcal{R}$, and outputs values in $\mathcal{Y}$.
- For any $(\mathsf{ik}, \mathsf{fk}) \leftarrow \mathsf{Gen}()$, the distribution of $\mathsf{H}(\mathsf{fk}, m; r)$ for random $m \xleftarrow{R} \mathcal{M}$ and random $r \xleftarrow{R} \mathcal{R}$ is uniform
- $\mathsf{Inv}(\mathsf{ik}, h, m)$ is a (potentially randomized) algorithm that takes as input a *secret* key, an value $h \in \mathcal{Y}$, and message $m \in \mathcal{M}$, and produces an r such that $\mathsf{H}(\mathsf{ek}, m, r) = h$. Moreover, r is distributed identically uniform conditioned on $\mathsf{H}(\mathsf{fk}, m, r) = h$.
- $H(\mathsf{fk}, \cdot, \cdot)$ is collision resistant. That is, for any efficient algorithm $\mathcal{A}$,

$$\Pr[\mathsf{H}(\mathsf{fk}, m_0; r_0) = \mathsf{H}(\mathsf{fk}, m_1; r_1) : (\mathsf{ik}, \mathsf{fk}) \xleftarrow{R} \mathsf{Gen}(), (m_0, m_1) \xleftarrow{R} \mathcal{A}(\mathsf{ek})] < \mathsf{negl}$$

Cash et al. [CHKP10] build chameleon hash functions based on the Short Integer Solution problem. Again, their construction departs from the ideal notion described above. Namely, the randomness in their construction comes from a non-uniform distribution, and Inv only produces a distribution that is negligibly-close to correct. Nonetheless, their construction suffices for our purposes.

2.2.2 Notions with Interaction

We now give the security definitions for primitives where the security game is potentially interactive. In these notions, quantum interaction gives a potentially stronger notion than classical interaction, even when only considering quantum queries. Here we present the notions allowing only classical

interaction, which are suitable for the near future where the systems are implemented on a classical device, but security is desired against quantum adversaries. We note that for these "post-quantum" definitions, classical security analyses generally hold. In Chapter 6, we give our new security models that allow for quantum interaction, and are therefore sufficient for schemes implemented on quantum devices.

Pseudorandom functions. A *pseudorandom function* is an important building block in cryptography, from which most of symmetric-key cryptography can be derived. It is a keyed function that "looks like" a random function when only given black-box oracle access, but not the key.

Definition 2.7. A pseudorandom function (PRF) is a classically efficient function $\mathsf{PRF} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ that is indistinguishable from a random function. That is, for any efficient quantum adversary $\mathcal{A}$ with *classical* access to an oracle,

$$\Pr[\mathcal{A}^{\mathsf{PRF}(k,\cdot)}() = 1] - \Pr[\mathcal{A}^{O(\cdot)}() = 1] < \mathsf{negl}$$

Goldreich, Goldwasser, and Micali [GGM86] show how to build (classical) PRFs from any pseudorandom generator. Their analysis carries over into the post-quantum setting, showing in particular that PRFs can be build from any one-way function.

Message authentication codes. A *message authentication code* is a procedure used to ensure data integrity. Very roughly, it is a keyed (possibly randomized) procedure that is hard to compute without the key, even given many values of the funciton.

Definition 2.8. A message authentication code MACs is a pair of classically efficient functions $(\mathsf{Sig}, \mathsf{Ver})$ where

- $\mathsf{Sig}(k, m)$ is a keyed (possibly randomized) procedure that outputs a tag σ for a message m.
- $\mathsf{Ver}(k, m, \sigma)$ is a keyed verification procedure, that takes as input a message and candidate tag, and either accepts or rejects.
- Verification always accepts valid tags. That is, for every message m, $\Pr[\mathsf{Ver}(k, m, \sigma) = \mathsf{acc} : k \xleftarrow{R} \mathcal{K}; \sigma \xleftarrow{R} \mathsf{Sig}(k, m)] = 1$

Moreover, a MAC satisfies one of several security requirements:

- $(\mathsf{Sig}, \mathsf{Ver})$ is strongly existentially unforgeable under a chosen message attack (strongly EUF-CMA secure) if all efficient quantum adversaries $\mathcal{A}$ have negligible advantage in the following experiment. A random key $k \in \mathcal{K}$ is chosen. $\mathcal{A}$ makes a polynomial number of adaptive classical signing queries on messages $m_i, i = 1, \ldots, q$, to which it receives tags $\sigma_i = \mathsf{Sig}(k, m_i)$. Then, the adversary produces a pair (m^*, σ^*). The advatnage of $\mathcal{A}$ is defined as the probability that

σ^* is a valid tag for m^* and (m^*, σ^*) is a new message/tag pair, that is, $\mathsf{Ver}(k, m^*, \sigma^*) = \mathsf{acc}$ and $(m^*, \sigma^*) \notin \{(m_i, \sigma_i)\}_{i \in [q]}$.

- $(\mathsf{Sig}, \mathsf{Ver})$ is q-time strongly EUF-CMA secure if it is strongly EUF-CMA secure for adversaries limited to making q signing queries.
- We can also make weak variants of the above definitions by requiring the adversary to produce (m^*, σ^*) where m^* is a new message (as apposed to only requiring that (m^*, σ^*) is a new message/tag pair). That is, $m^* \notin \{m_i\}_{i \in [q]}$.

Any classical PRF (with sufficiently many output bits) gives a secure classical MAC, and this fact carries over to the post-quantum setting. Post-quantum PRFs can also be used to instantiate the Carter-Wegman MAC; the resulting scheme will be secure in the sense above.

Digital signatures. A *digital signature* is a public-key version of a message authentication code.

Definition 2.9. A signature scheme is a tuple of classically efficient functions $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ where

- $\mathsf{Gen}()$ is a randomized procedure of no inputs that produces a secret signing key sk and a public verification key vk.
- $\mathsf{Sig}(\mathsf{sk}, m)$ is a secret key (possibly randomized) procedure that outputs a signature σ for a message m.
- $\mathsf{Ver}(\mathsf{vk}, m, \sigma)$ is a public key verification procedure, that takes as input a message and candidate tag, and either accepts or rejects.
- Verification always accepts valid tags. That is, for every message m, $\Pr[\mathsf{Ver}(\mathsf{vk}, m, \sigma) = \mathsf{acc} : (\mathsf{sk}, \mathsf{vk}) \xleftarrow{R} \mathsf{Gen}(); \sigma \xleftarrow{R} \mathsf{Sig}(\mathsf{sk}, m)] = 1$

Moreover, a signature scheme satisfies one of several security requirements, very similar to the security requirements made of MACs:

- $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ is strongly existentially unforgeable under a chosen message attack (strongly EUF-CMA secure) if all efficient quantum adversaries $\mathcal{A}$ have negligible advantage in the following experiment. A signing and verification key pair $(\mathsf{sk}, \mathsf{vk}) \xleftarrow{R} \mathsf{Gen}()$ is generated. $\mathcal{A}$ receives vk, and then makes a polynomial number of adaptive classical signing queries on messages $m_i, i = 1, \ldots, q$, to which it receives signatures $\sigma_i = \mathsf{Sig}(\mathsf{sk}, m_i)$. Then, the adversary produces a pair (m^*, σ^*). The advantage of $\mathcal{A}$ is defined as the probability that σ^* is a valid tag for m^* and (m^*, σ^*) is a new message/tag pair, that is, $\mathsf{Ver}(\mathsf{vk}, m^*, \sigma^*) = \mathsf{acc}$ and $(m^*, \sigma^*) \notin \{(m_i, \sigma_i)\}_{i \in [q]}$.

- $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ is strongly existentially unforgeable under a *random* message attack (strongly EUF-RMA secure) if the messages m_i are chosen at uniformly and independently random instead of being chosen arbitrarily by the adversary.
- $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ is *universally* unforgeable under a random message attack (UUF-RMA secure) if the message m^* is chosen uniformly at random, as are the m_i.
- We can make q-time variants of any of the above definitions by restricting to adversaries where q is bounded.
- We can also make weak variants of the above definitions by requiring the adversary to produce (m^*, σ^*) where m^* is a new message (as apposed to only requiring that (m^*, σ^*) is a new message/tag pair). That is, $m^* \notin \{m_i\}_{i \in [q]}$. Notice that, for the UUF-RMA definition, there will be no collisions between m^* and the m_i, so the distinction between strong and weak security is irrelevant. Therefore, we omit the modifier and just refer to UUF-RMA security.

Rompel [Rom90] shows that signatures can be built from any one-way function, and this result carries over into the post-quantum setting. Various other signature schemes based on hard lattice problems also remain secure against quantum adversaries, assuming the lattice problems remain hard for quantum computers.

Private key encryption. A symmetric key encryption scheme (also called a private key encryption scheme) is a procedure to ensure message privacy.

Definition 2.10. A symmetric key encryption scheme is a pair of a classically efficient procedures $(\mathsf{Enc}, \mathsf{Dec})$ where:

- $\mathsf{Enc}(k, m)$ is a keyed randomized procedure that, given a message m, produces a ciphertext c.
- $\mathsf{Dec}(k, c)$ is a keyed deterministic procedure that, given a ciphertext c produces a message m.
- Decryption is the inverse of encryption. That is, for all messages m, $\Pr[\mathsf{Dec}(k, c) = m : k \xleftarrow{R} \mathcal{K}, c \xleftarrow{R} \mathsf{Enc}(k, m)] = 1$.

Additionally, we make one of several security requirements for $(\mathsf{Enc}, \mathsf{Dec})$:

- $(\mathsf{Enc}, \mathsf{Dec})$ is secure against a chosen plaintext attack (CPA-secure) if all efficient quantum adversaries $\mathcal{A}$ have negligible advantage in the following experiment. A random key $k \xleftarrow{R} \mathcal{K}$ and a random bit b are chosen, and $\mathcal{A}$ is given oracle access to the (randomized) function $\mathsf{Enc}_{k,b}(m_0, m_1) \longrightarrow \mathsf{Enc}(k, m_b)$. Fresh random coins are chosen for each query to the function. Then, $\mathcal{A}$ produces a guess b' for b. The advantage of $\mathcal{A}$ is the probability of guessing correctly, minus the probability a random guess is correct. That is, the advantage of $\mathcal{A}$ is the quantity $\left| \Pr[b \xleftarrow{R} \mathcal{A}^{\mathsf{Enc}_{k,b}}() : k \xleftarrow{R} \mathcal{K}, b \xleftarrow{R} \{0,1\}] - \frac{1}{2} \right|$

- $(\mathsf{Enc}, \mathsf{Dec})$ is one-time secure if it is CPA-secure against adversaries limited to making a single query.
- $(\mathsf{Enc}, \mathsf{Dec})$ is secure against a chosen ciphertext attack (CCA-secure) if $\mathcal{A}$ is addionally allowed a decryption oracle. That is, all efficient quantum adversaries $\mathcal{A}$ have negligible advantage in the following experiment. A random key $k \xleftarrow{R} \mathcal{K}$ and a random bit b are chosen. Then $\mathcal{A}$ is given access to a pair of stateful oracles. The first oracle is the randomized function $\mathsf{Enc}_{k,b}(m_0, m_1) \longmapsto \mathsf{Enc}(k, m_b)$, which now also adds the resulting ciphertext c to an initially empty table T. The second oracle is the deterministic function

$$\mathsf{Dec}_k(c) \longmapsto \begin{cases} \perp & \text{if } c \in T \\ \mathsf{Dec}(k, c) & \text{if } c \notin T \end{cases}$$

  where $\perp$ is a special symbol indicating failure. Then, $\mathcal{A}$ produces a guess b' for b. The advantage of $\mathcal{A}$ is the probability of guessing correctly, minus the probability a random guess is correct. That is, the advantage of $\mathcal{A}$ is the quantity $\left|\Pr[b \xleftarrow{R} \mathcal{A}^{\mathsf{Enc}_{k,b}, \mathsf{Dec}_k}() : k \xleftarrow{R} \mathcal{K}, b \xleftarrow{R} \{0,1\}] - \frac{1}{2}\right|$

  Notice that the requirement that $\mathsf{Dec}_k(c)$ outputs $\perp$ for $c \in T$ is required to avoid trivial attacks. Recall that $c \in T$ is equal to $\mathsf{Enc}(k, m_b)$ for known m_0, m_1. Therefore, if $m_0 \neq m_1$, seeing the decryption of $c \in T$ would allow for determining b.

Private key CCA-secure encryption can be build from any PRF, which in turn can be build from any one-way function.

Public key encryption. An asymmetric key encryption scheme (also called a public key encryption scheme) is public key analog of private key encryption.

Definition 2.11. An asymmetric key encryption scheme is a tuple of a classically efficient procedures $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ where:

- $\mathsf{Gen}()$ is a randomized provedure of no inputs that outputs a secret message decrypion key dk and a public encryption key ek.
- $\mathsf{Enc}(\mathsf{ek}, m)$ is a public key randomized procedure that, given a message m, produces a ciphertext c.
- $\mathsf{Dec}(\mathsf{dk}, c)$ is a secret key deterministic procedure that, given a ciphertext c produces a message m.
- Decryption is the inverse of encryption. That is, for all messages m, $\Pr[\mathsf{Dec}(\mathsf{dk}, c) = m : (\mathsf{dk}, \mathsf{ek}) \xleftarrow{R} \mathsf{Gen}(), c \xleftarrow{R} \mathsf{Enc}(\mathsf{ek}, m)] = 1$.

Additionally, we make one of several security requirements for $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$:

- $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is secure against a chosen plaintext attack (CPA-secure) if all efficient quantum adversaries $\mathcal{A}$ have negligible advantage in the following experiment. A random secret/public key pair $(\mathsf{dk}, \mathsf{ek}) \leftarrow \mathsf{Gen}()$ and a random bit b are chosen, and $\mathcal{A}$ is given the encryption key ek. Moreover, $\mathcal{A}$ is given oracle access to the (randomized) function $\mathsf{Enc}_{\mathsf{ek},b}(m_0, m_1) \longrightarrow \mathsf{Enc}(\mathsf{ek}, m_b)$. Fresh random coins are chosen for each query to the function. Then, $\mathcal{A}$ produces a guess b' for b. The advantage of $\mathcal{A}$ is the probability of guessing correctly, minus the probability a random guess is correct. That is, the advantage of $\mathcal{A}$ is the quantity

$$\left| \Pr[b \xleftarrow{R} \mathcal{A}^{\mathsf{Enc}_{\mathsf{ek},b}}(\mathsf{ek}) : (\mathsf{dk}, \mathsf{ek}) \xleftarrow{R} \mathsf{Gen}(), b \xleftarrow{R} \{0,1\}] - \frac{1}{2} \right|$$

  Notice that in the asymmetric setting, $\mathcal{A}$ can encrypt messages for itself. This allows one to show, by a simple hybrid argument, that CPA-security is equivalent to 1-time security, where $\mathcal{A}$ is only allowed a single query to $\mathsf{Enc}_{\mathsf{ek},b}$.

- $(\mathsf{Enc}, \mathsf{Dec})$ is secure against a chosen ciphertext attack (CCA-secure) if $\mathcal{A}$ is additionally allowed a decryption oracle. That is, all efficient quantum adversaries $\mathcal{A}$ have negligible advantage in the following experiment. A random secret/public key pair $(\mathsf{dk}, \mathsf{ek}) \leftarrow \mathsf{Gen}()$ and a random bit b are chosen, and $\mathcal{A}$ is given the encryption key ek. Then $\mathcal{A}$ is given access to a pair of stateful oracles. The first oracle is the randomized function $\mathsf{Enc}_{\mathsf{ek},b}(m_0, m_1) \longmapsto \mathsf{Enc}(\mathsf{ek}, m_b)$, which now also adds the resulting ciphertext c to an initially empty table T. The second oracle is the deterministic function

$$\mathsf{Dec}_{\mathsf{dk}}(c) \longmapsto \begin{cases} \perp & \text{if } c \in T \\ \mathsf{Dec}(\mathsf{dk}, c) & \text{if } c \notin T \end{cases}$$

  where $\perp$ is a special symbol indicating failure. Then, $\mathcal{A}$ produces a guess b' for b. The advantage of $\mathcal{A}$ is the probability of guessing correctly, minus the probability a random guess is correct. That is, the advantage of $\mathcal{A}$ is the quantity

$$\left| \Pr[b \xleftarrow{R} \mathcal{A}^{\mathsf{Enc}_{\mathsf{ek},b}, \mathsf{Dec}_{\mathsf{dk}}}(\mathsf{ek}) : (\mathsf{dk}, \mathsf{ek}) \xleftarrow{R} \mathsf{Gen}(), b \xleftarrow{R} \{0,1\}] - \frac{1}{2} \right|$$

  Again, a simple hybrid argument shows that public key CCA-security is equivalent to the case where $\mathcal{A}$ is only allowed a single query to $\mathsf{Enc}_{\mathsf{ek},b}$ (though still allowed arbitrarily many queries to $\mathsf{Dec}_{\mathsf{dk}}$).

There are many possibilities for post-quantum CPA secure public key encryption, starting with the work of Ajtai and Dwork [AD97]. Post-quantum CCA security can be obtained, for example, from any post-quantum identity-based encryption (IBE) scheme (see below for IBE definitions and instantiations) using the IBE-to-CCA conversion of Boneh et al. [BCHK07].

Identity-based encryption. Identity-based encryption is a variant of public key encryption where user's public keys are simply some identifying information, such as an email address. Identity-based encryption offers an alternative to the standard public key infrastructure. A user then goes to a central authority, who generates the corresponding secret key for the user's identity.

Definition 2.12. An identity-based encryption scheme (IBE) is a tuple of a classically efficient procedures $(\mathsf{Gen}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ where:

- $\mathsf{Gen}()$ is a randomized procedure of no inputs that outputs a master secret key msk and a master encryption key mek.
- $\mathsf{KeyGen}(\mathsf{msk}, \mathsf{id})$ is a (potentially randomized) procedure that requires the master secret, and generates the user decryption key $\mathsf{dk}_{\mathsf{id}}$ for user id.
- $\mathsf{Enc}(\mathsf{mek}, \mathsf{id}, m)$ is a public key randomized procedure that, given an identity id and message m, produces a ciphertext c encrypted to the identity id.
- $\mathsf{Dec}(\mathsf{dk}_{\mathsf{id}}, c)$ is a procedure requiring the user secret key for identity id that, given a ciphertext c produces a message m.
- Decryption is the inverse of encryption for the same identity. That is, for all messages m and identities $\int \cdots \int$,

$$\Pr[\mathsf{Dec}(\mathsf{dk}_{\mathsf{id}}, c) = m : (\mathsf{msk}, \mathsf{mek}) \xleftarrow{R} \mathsf{Gen}(), c \xleftarrow{R} \mathsf{Enc}(\mathsf{mek}, \mathsf{id}, m), \mathsf{dk}_{\mathsf{id}} \xleftarrow{R} \mathsf{KeyGen}(\mathsf{msk}, \mathsf{id})] = 1$$

- All efficient quantum adversaries $\mathcal{A}$ have negligible advantage in the following experiment. A master secret key and encryption key par $(\mathsf{msk}, \mathsf{mek}) \xleftarrow{R} \mathsf{Gen}()$ are generated, a random bit $b \xleftarrow{R} \{0, 1\}$ is chosen, and mek is given to $\mathcal{A}$. An empty table T is initialized. $\mathcal{A}$ is then given access to two stateful oracles. The first oracle is the randomized function

$$\mathsf{Enc}_{\mathsf{mek},b}(\mathsf{id}, m_0, m_1) \longmapsto \begin{cases} \perp & \text{if } \mathsf{id} \in T \\ \mathsf{Enc}(\mathsf{mek}, m_b) & \text{if } \mathsf{id} \notin T \end{cases}$$

  which also updates the table T by adding id. Call queries to this oracle *challenge* queries. The second oracle is the potentially randomized function

$$\mathsf{KeyGen}_{\mathsf{msk}}(\mathsf{id}) \longmapsto \begin{cases} \perp & \text{if } \mathsf{id} \in T \\ \mathsf{KeyGen}(\mathsf{msk}, \mathsf{id}) & \text{if } \mathsf{id} \notin T \end{cases}$$

  which also updates the table T by adding id. Call queries to this oracle *keygen* queries. Then, $\mathcal{A}$ produces a guess b' for b. The advantage of $\mathcal{A}$ is the probability of guessing correctly, minus

the probability a random guess is correct. That is, the advantage of $\mathcal{A}$ is the quantity

$$\left| \Pr[b \xleftarrow{R} \mathcal{A}^{\mathsf{Enc}_{\mathsf{mek},b}, \mathsf{KeyGen}_{\mathsf{msk}}}(\mathsf{mek}) : (\mathsf{msk}, \mathsf{mek}) \xleftarrow{R} \mathsf{Gen}(), b \xleftarrow{R} \{0,1\}] - \frac{1}{2} \right|$$

Notice that the requirement that $\mathcal{A}$ cannot access both $\mathsf{Enc}_{\mathsf{mek},b}$ and $\mathsf{KeyGen}_{\mathsf{msk}}$ on the same identity id is required to block trivial attacks. Namely, if $\mathcal{A}$ could query $\mathsf{KeyGen}_{\mathsf{msk}}$ on id, obtaining $\mathsf{dk}_{\mathsf{id}}$, then it could use $\mathsf{dk}_{\mathsf{id}}$ to decrypt a ciphertext $c \xleftarrow{R} \mathsf{Enc}_{\mathsf{mek},b}(\mathsf{id}, m_0, m_1)$, obtaining the message m_b. From m_b, $\mathcal{A}$ can determine b.

Also notice that, by a simple hybrid argument, we can assume that $\mathcal{A}$ only makes a single query to the $\mathsf{Enc}_{\mathsf{mek},b}$ oracle.

While the original construction of identity-based encryption based on bilinear maps [BF01] is insecure against quantum adversaries, there are many candidate constructions based on lattices, such as [ABB10a].

Chapter 3

The Rank Method

In this chapter we introduce the rank method which is a general approach to proving lower bounds on quantum algorithms. The setup is as follows: we give a quantum algorithm A access to some quantity $H \in \mathcal{H}$. By access, we mean that the final state of the algorithm is some fixed function of H. In this paper, $\mathcal{H}$ will be a set of functions, and A will be given oracle access to $H \in \mathcal{H}$ by allowing A to make q quantum oracle queries to H, for some q. For now, we will treat $\mathcal{H}$ abstractly, and return to the specific case where $\mathcal{H}$ is a set of functions later.

The idea behind the rank method is that, if we treat the final states of the algorithm on different H as vectors, the space spanned by these vectors will be some subspace of the overall Hilbert space. If the dimension of this subspace is small enough, the subspace (and hence all of the vectors in it) must be reasonably far from most of the vectors in the measurement basis. This allows us to bound the ability of such an algorithm to achieve some goal.

For $H \in \mathcal{H}$, let $|\psi_H\rangle$ be the final state of the quantum algorithm $\mathcal{A}$, before measurement, when given access to H. Suppose the different $|\psi_H\rangle$ vectors all lie in a space of dimension d. Let $\mathbf{\Psi}_{\mathcal{A},\mathcal{H}}$ be the the $|\mathcal{H}| \times d$ matrix whose rows are the various vectors $|\psi_H\rangle$.

Definition 3.1. For a quantum algorithm A given access to some value $H \in \mathcal{H}$, we define the *rank*, denoted $\mathrm{rank}(\mathcal{A}, \mathcal{H})$, as the rank of the matrix $\mathbf{\Psi}_{\mathcal{A},\mathcal{H}}$.

The rank of an algorithm A seemingly contains very little information: it gives the dimension of the subspace spanned by the $|\psi_H\rangle$ vectors, but gives no indication of the orientation of this subspace or the positions of the $|\psi_H\rangle$ vectors in the subspace. Nonetheless, we demonstrate how the success probability of an algorithm can be bounded from above knowing only the rank of $\mathbf{\Psi}_{\mathcal{A},\mathcal{H}}$. The following is prove in Section 7.1:

Lemma 3.2. *Let $\mathcal{A}$ be a quantum algorithm that has access to some value $H \in \mathcal{H}$ drawn from some distribution D and produces some output $w \in \mathcal{W}$. Let $R : \mathcal{H} \times \mathcal{W} \to \{\mathtt{True}, \mathtt{False}\}$ be a binary*

relation. Then the probability that $\mathcal{A}$ outputs some w such that $R(H, w) =$ `True` *is at most*

$$\left(\max_{w \in \mathcal{W}} \Pr_{H \leftarrow D}[R(H, w)]\right) \times \text{rank}(\mathcal{A}, \mathcal{H}) \ .$$

In other words, the probability that A succeeds in producing $w \in \mathcal{W}$ for which $R(H, w)$ is true is at most $\text{rank}(\mathcal{A}, \mathcal{H})$ *times the best probability of success of any algorithm that ignores H and just outputs some fixed w.*

We now move to the specific case of oracle access to bound the rank. $\mathcal{H}$ is now some set of functions from $\mathcal{X}$ to $\mathcal{Y}$, and our algorithm A makes q quantum oracle queries to a function $H \in \mathcal{H}$. Concretely, A is specified by $q + 1$ unitary matrices $\mathbf{U}_i$, and the final state of A on input H is the state

$$\mathbf{U}_q \mathbf{H} \mathbf{U}_{q-1} \cdots \mathbf{U}_1 \mathbf{H} \mathbf{U}_0 |0\rangle$$

where $\mathbf{H}$ is the unitary transformation mapping $|x, y, z\rangle$ into $|x, y + H(x), z\rangle$, representing an oracle query to the function H. To use the rank method (Lemma 3.2) for our purposes, we need to bound the rank of such an algorithm. First, we define the following quantity:

$$C_{k,q,n} \equiv \sum_{r=0}^{q} \binom{k}{r} (n-1)^r \ .$$

We now show an upper bound of the rank of an algorithm making a bounded number of queries to an oracle. The following is proved in Section 7.2:

Theorem 3.3. *Let $\mathcal{X}$ and $\mathcal{Y}$ be sets of size m and n and let H_0 be some function from $\mathcal{X}$ to $\mathcal{Y}$. Let $\mathcal{S}$ be a subset of $\mathcal{X}$ of size k and let $\mathcal{H}$ be some set of functions from $\mathcal{X}$ to $\mathcal{Y}$ that are equal to H_0 except possibly on points in $\mathcal{S}$. If A is a quantum algorithm making q queries to an oracle drawn from $\mathcal{H}$, then*

$$\text{rank}(\mathcal{A}, \mathcal{H}) \leq C_{k,q,n} \ .$$

3.0.3 A Simple Example

Suppose our task is to, given one quantum query to an oracle $H : \mathcal{X} \to \mathcal{Y}$, produce two distinct pairs (x_0, y_0) and (x_1, y_1) such that $H(x_0) = y_0$ and $H(x_1) = y_1$. Suppose further that H is drawn from a pairwise independent set $\mathcal{H}$. We will now see that the rank method leads to a bound on the success probability of any quantum algorithm A.

Corollary 3.4. *No quantum algorithm A, making a single query to a function $H : \mathcal{X} \to \mathcal{Y}$ drawn from a pairwise independent set $\mathcal{H}$, can produce two distinct input/output pairs of H, except with probability at most $|\mathcal{X}|/|\mathcal{Y}|$.*

Proof. Let $m = |\mathcal{X}|$ and $n = |\mathcal{Y}|$. Since no outputs of H are fixed, we will set $\mathcal{S} = \mathcal{X}$ in Theorem 3.3, showing that the rank of the algorithm A is bounded by $C_{m,1,n} = 1+m(n-1) < mn$. If an algorithm makes no queries to H, the best it can do at outputting two distinct input/output pairs is to just pick two arbitrary distinct pairs, and output those. The probability that this zero-query algorithm succeeds is at most $1/n^2$. Then Lemma 3.2 tells us that A succeeds with probability at most $\text{rank}(A, \mathcal{H})$ times this amount, which equates to $\frac{m}{n}$. □

For $m > n$, this bound is trivial. However, for m smaller than n, this gives a non-trivial bound, and for m exponentially smaller than n, this bound is negligible.

3.1 Main Application: The Oracle Interrogation Problem

In this section, we consider the following problem: given q quantum queries to a random oracle $H : \mathcal{X} \to \mathcal{Y}$, produce $k > q$ distinct pairs (x_i, y_i) such that $y_i = H(x_i)$. Let $n = |\mathcal{Y}|$ be the size of the range. We will be interested in two cases: (1) where the size of the range n is very large, say much larger than the number of queries, and (2) when the size of the range is very small, say a constant.

In the classical setting, when $k \leq q$, this problem is easy, since we can just pick an arbitrary set of k different x_i values, and query the oracle on each value. For $k > q$, no adversary of even unbounded complexity can solve this problem, except with probability $1/n^{k-q}$, since for any set of k inputs, at least $k - q$ of the corresponding outputs are completely unknown to the adversary. Therefore, for large n, we have have a sharp threshold: for $k \leq q$, this problem can be solved efficiently with probability 1, and for even $k = q + 1$, this problem cannot be solved, even inefficiently, except with negligible probability.

In the quantum setting, the $k \leq q$ case is the same as before, since we can still query the oracle classically. However, for $k > q$, the quantum setting is more challenging. The adversary can potentially query the random oracle on a superposition of all inputs, so he "sees" the output of the oracle on all points. Proving that it is still impossible to produce k input/output pairs is thus more complicated, and existing methods fail to prove that this problem is difficult. Therefore, it is not immediately clear that we have the same sharp threshold as before.

In Section 3.1.1 we use the rank method to bound the probability that any (even computationally unbounded) quantum adversary succeeds. Then in Section 3.1.2 we show that our bound is tight by giving an efficient algorithm for this problem that achieves the lower bound. In particular, we prove the following theorem:

Theorem 3.5. *Let A be a quantum algorithm making q queries to a random oracle $H : \mathcal{X} \to \mathcal{Y}$ whose range has size n, and produces $k > q$ pairs $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$. The probability that the x_i values are distinct and $y_i = H(x_i)$ for all $i \in [k]$ is at most $\frac{1}{n^k} C_{k,q,n}$. Moreover, there is a quantum algorithm that exactly acheives this success probability.*

We consider two cases:

- Exponentially-large range $\mathcal{Y}$ and polynomial k, q. In this case, even when $k = q+1$ the success probability is negligible. That is, to produce even one additional input/output pair is hard. Thus, we get the same sharp threshold as in the classical case
- Constant size range $\mathcal{Y}$ and polynomial k, q. Then even if q is a constant fraction of k we can still produce k input/output pairs with overwhelming probability using only q quantum queries. This is in contrast to the classical case, where the success probability for $q = ck$, $c < 1$, is negligible in k.

3.1.1 A Tight Upper Bound

Theorem 3.6. *Let A be a quantum algorithm making q queries to a random oracle $H : \mathcal{X} \to \mathcal{Y}$ whose range has size n, and produces $k > q$ pairs $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$. The probability that the x_i values are distinct and $y_i = H(x_i)$ for all $i \in [k]$ is at most $\frac{1}{n^k} C_{k,q,n}$.*

Proof. Before giving the complete proof, we sketch the special case where k is equal to the size of the domain. In this case, any quantum algorithm that outputs k distinct input/output pairs must output *all* input/output pairs. Similar to the proof of Corollary 3.4, we will set $\mathcal{S} = \mathcal{X}$, and use Theorem 3.3 to bound the rank of A at $C_{k,q,n}$. Now, any algorithm making *zero* queries succeeds with probability at most $1/n^k$. Lemma 3.2 then bounds the success probability of any q query algorithm as

$$\frac{1}{n^k} C_{k,q,n} \ .$$

Now for the general proof: first, we will assume that the probability A outputs any particular sequence of x_i values is independent of the oracle H. We will show how to remove this assumption later. We can thus write

$$|\psi_H^q\rangle = \sum_{\mathbf{x}} \alpha_{\mathbf{x}} |\mathbf{x}\rangle |\phi_{H,\mathbf{x}}\rangle$$

where $\alpha_{\mathcal{X}}$ are complex numbers whose square magnitudes sum to one, and $|\mathbf{x}\rangle|\phi_{H,\mathbf{x}}\rangle$ is the normalized projection of $|\psi_H^q\rangle$ onto the space spanned by $|\mathbf{x}, w\rangle$ for all w. The probability that A succeeds is equal to

$$\sum_H \Pr[H] \sum_{\mathbf{x}} |\langle \mathbf{x}, H(\mathbf{x}) | \psi_H^q \rangle|^2 = \sum_H \Pr[H] \sum_{\mathbf{x}} |\alpha_{\mathbf{x}}|^2 |\langle H(\mathbf{x}) | \phi_{H,\mathbf{x}} \rangle|^2 \ .$$

First, we reorder the sums so the outer sum is the sum over $\mathbf{x}$. Now, we write $H = (H_0, H_1)$ where H_0 is the oracle restricted to the components of $\mathbf{x}$, and H_1 is the oracle restricted to all other inputs. Thus, our probability is:

$$\frac{1}{n^m} \sum_{\mathbf{x}} |\alpha_{\mathbf{x}}|^2 \sum_{H_0, H_1} \left| \langle H_0(\mathbf{x}) | \phi_{(H_0,H_1),\mathbf{x}} \rangle \right|^2 \ .$$

Using the same trick as we did before, we can replace $|\langle H(\mathbf{x})|\phi_{H,\mathbf{x}}\rangle|$ with the quantity

$$\left|\text{proj}_{\text{span}|\phi_{(H_0,H_1),\mathbf{x}}\rangle}|H_0(\mathcal{X})\rangle\right| \ ,$$

which is bounded by $\left|\text{proj}_{\text{span}\{|\phi_{(H_0',H_1),\mathbf{x}}\rangle\}}|H_0(\mathbf{x})\rangle\right|$ as we vary H_0' over oracles whose domain is the components of $\mathbf{x}$. The probability of success is then bounded by

$$\frac{1}{n^m}\sum_{\mathbf{x}}|\alpha_{\mathbf{x}}|^2\sum_{H_0,H_1}\left|\text{proj}_{\text{span}\{|\phi_{(H_0',H_1),\mathbf{x}}\rangle\}}|H_0(\mathbf{x})\rangle\right|^2 \ .$$

We now perform the sum over H_0. Like in the proof of Corollary 3.4, the sum evaluates to $\dim\text{span}\{|\phi_{(H_0',H_1),\mathbf{x}}\rangle\}$. Since the $|\phi_{(H_0',H_1),\mathbf{x}}\rangle$ vectors are projections of $|\psi_H^q\rangle$, this dimension is bounded by $\dim\text{span}\{|\psi^q_{(H_0',H_1)}\rangle\}$. Let $\mathcal{H}$ be the set of oracles (H_0',H_1) as we vary H_0', and consider A acting on oracles in $\mathcal{H}$. Fix some oracle H_0^* from among the H_0' oracles, and let $\mathcal{S}$ be the set of components of $\mathbf{x}$. Then (H_0',H_1) differs from (H_0^*,H_1) only on the elements of $\mathcal{S}$. Since $|\mathcal{S}|\leq k$, Lemma 3.2 tells us that $\text{rank}(A,\mathcal{H})\leq C_{k,q,n}$. But

$$\text{rank}(A,\mathcal{H}) = \dim\text{span}\{|\psi^q_{(H_0',H_1)}\rangle\}$$

Therefore, we can bound the success probability by

$$\frac{1}{n^m}\sum_{\mathbf{x}}|\alpha_{\mathbf{x}}|^2\sum_{H_1}C_{k,q,n} \ .$$

Summing over all n^{m-k} different H_1 values and all $\mathbf{x}$ values gives a bound of

$$\frac{1}{n^k}C_{k,q,n}$$

as desired.

So far, we have assume that A produces $\mathbf{x}$ with probability independent of H. Now, suppose our algorithm A does not produce $\mathbf{x}$ with probability independent of the oracle. We construct a new algorithm B with access to H that does the following: pick a random oracle O with the same domain and range as H, and give A the oracle $H+O$ that maps x into $H(x)+O(x)$. When A produces k input/output pairs (x_i,y_i), output the pairs $(x_i,y_i-O(x_i))$. (x_i,y_i) are input/output pairs of $H+O$ if and only if $(x_i,y_i-O(x_i))$ are input/output pairs of H. Further, A still sees a random oracle, so it succeeds with the same probability as before. Moreover, the oracle A sees is now independent of H, so B outputs $\mathbf{x}$ with probability independent of H. Thus, applying the above analysis to B shows that B, and hence A, produce k input/output pairs with probability at

most

$$\frac{1}{n^k} C_{k,q,n}$$

□

For this paper, we are interested in the case where $n = |\mathcal{Y}|$ is exponentially large, and we are only allowed a polynomial number of queries. Suppose $k = q + 1$, the easiest non-trivial case for the adversary. Then, the probability of success is

$$\frac{1}{n^{q+1}} \sum_{r=0}^{q} \binom{q+1}{r} (n-1)^r = 1 - \left(1 - \frac{1}{n}\right)^{q+1} \leq \frac{q+1}{n} \ . \tag{3.1.1}$$

Therefore, to produce even one extra input/output pair is impossible, except with exponentially small probability, just like in the classical case. This proves the first part of Theorem 3.5.

3.1.2 The Optimal Attack

In this section, we present a quantum algorithm for the problem of computing $H(x_i)$ for k different x_i values, given only $q < k$ queries:

Theorem 3.7. *Let $\mathcal{X}$ and $\mathcal{Y}$ be sets, and fix integers $q < k$, and k distinct values $x_1, ..., x_k \in \mathcal{X}$. There exists a quantum algorithm A that makes q non-adaptive quantum queries to any function $H : \mathcal{X} \to \mathcal{Y}$, and produces $H(x_1), ..., H(x_k)$ with probability $C_{k,q,n}/n^k$, where $n = |\mathcal{Y}|$.*

The algorithm is similar to the algorithm of [van98], though generalized to handle arbitrary range sizes. This algorithm has the same success probability as in Theorem 3.6, showing that both our attack and lower bound of Theorem 3.6 are optimal. This proves the second part of Theorem 3.5.

Proof. Assume that $\mathcal{Y} = \{0, ..., n-1\}$. For a vector $\mathbf{y} \in \mathcal{Y}^k$, let $\Delta(\mathbf{y})$ be the number of coordinates of $\mathbf{y}$ that do not equal 0. Also, assume that $x_i = i$.

Initially, prepare the state that is a uniform superposition of all vectors $\mathbf{y} \in \mathcal{Y}^k$ such that $\Delta(\mathbf{y}) \leq q$:

$$|\psi_1\rangle = \frac{1}{\sqrt{V}} \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} |\mathbf{y}\rangle$$

Notice that the number of vectors of length k with at most q non-zero coordinates is exactly

$$\sum_{r=0}^{q} \binom{k}{r} (n-1)^r = C_{k,q,n} \ .$$

We can prepare the state efficiently as follows: Let $\mathsf{Setup}_{k,q,n} : [C_{k,q,n}] \to [n]^k$ be the following function: on input $\ell \in [C_{k,q,n}]$,

- Check if $\ell \leq C_{k-1,q,n}$. If so, compute the vector $\mathbf{y}' = \mathsf{Setup}_{k-1,q,n}(n)$, and output the vector $\mathbf{y} = (0, \mathbf{y}')$.
- Otherwise, let $\ell' = \ell - C_{k-1,q,n}$. It is easy to verify that $\ell' \in [(n-1)C_{k-1,q-1,n}]$.
- Let $\ell'' \in C_{k-1,q-1,n}$ and $y_0 \in [n] \setminus \{0\}$ be the unique such integers such that $\ell' = (n-1)\ell'' + y_0 - n$.
- Let $\mathbf{y}' = \mathsf{Setup}_{k-1,q-1,n}(\ell'')$, and output the vector $\mathbf{y} = (y_0, \mathbf{y}')$.

The algorithm relies on the observation that a vector $\mathbf{y}$ of length k with at most q non-zero coordinates falls into one of either two categories:

- The first coordinate is 0, and the remaining $k-1$ coordinates form a vector with at most q non-zero coordinates
- The first coordinate is non-zero, and the remaining $k-1$ coordinates form a vector with at most $q-1$ non-zero coordinates.

There are $C_{k-1,q,n}$ vectors of the first type, and $C_{k-1,q-1,n}$ vectors of the second type for each possible setting of the first coordinate to something other than 0. Therefore, we divide $[A_{k,q,n}]$ into two parts: the first $C_{k-1,q,n}$ integers map to the first type, and the remaining $(n-1)C_{k-1,q-1,n}$ integers map to vectors of the second type.

We note that Setup is efficiently computable, invertible, and its inverse is also efficiently computable. Therefore, we can prepare $|\psi_1\rangle$ by first preparing the state

$$\frac{1}{\sqrt{C_{k,q,n}}} \sum_{\ell \in [C_{k,q,n}]} |\ell\rangle$$

and reversibly converting this state into $|\phi_1\rangle$ using $\mathsf{Setup}_{k,q,n}$.

Next, let $F : \mathcal{Y}^k \to [k]^q$ be the function that outputs the indexes i such that $\mathbf{y}_i \neq 0$, in order of increasing i. If there are fewer than q such indexes, the function fills in the remaining spaces the first indexes such that $\mathbf{y}_i = 0$ If there are more than q indexes, the function truncates to the first q. F is realizable by a simple classical algorithm, so it can be implemented as a quantum algorithm. Apply this algorithm to $|\psi_1\rangle$, obtaining the state

$$|\psi_2\rangle = \frac{1}{\sqrt{C_{k,q,n}}} \sum_{\mathbf{y}:\Delta(\mathbf{y}) \leq q} |\mathbf{y}, F(\mathbf{y})\rangle$$

Next, let $G : \mathcal{Y}^k \to \mathcal{Y}_q$ be the function that takes in vector $\mathbf{y}$, computes $\mathbf{x} = F(\mathbf{y})$, and outputs the vector $(y_{x_1}, y_{x_2}, ..., y_{x_q})$. In other words, it outputs the vector of the non-zero components of $\mathbf{y}$, padding with zeros if needed. This function is also efficiently computable by a classical algorithm,

so we can apply if to each part of the superposition:

$$|\psi_3\rangle = \frac{1}{\sqrt{C_{k,q,n}}} \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} |\mathbf{y}, F(\mathbf{y}), G(\mathbf{y})\rangle$$

Now we apply the Fourier transform to the $G(\mathbf{y})$ part, obtaining

$$|\psi_4\rangle = \frac{1}{\sqrt{C_{k,q,n}}} \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} |\mathbf{y}, F(\mathbf{y})\rangle \sum_{\mathbf{z}} e^{-i\frac{2\pi}{n}\langle \mathbf{z}, G(\mathbf{y})\rangle} |\mathbf{z}\rangle$$

Now we can apply H to the $F(\mathbf{y})$ part using q non-adaptive queries, adding the answer to the $\mathbf{z}$ part. The result is the state

$$|\psi_5\rangle = \frac{1}{\sqrt{C_{k,q,n}}} \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} |\mathbf{y}, F(\mathbf{y})\rangle \sum_{\mathbf{z}} e^{-i\frac{2\pi}{n}\langle \mathbf{z}, G(\mathbf{y})\rangle} |\mathbf{z} + H(F(\mathbf{y}))\rangle$$

We can rewrite this last state as follows:

$$|\psi_5\rangle = \frac{1}{\sqrt{C_{k,q,n}}} \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} e^{i\frac{2\pi}{n}\langle H(F(\mathbf{y})), G(\mathbf{y})\rangle} |\mathbf{y}, F(\mathbf{y})\rangle \sum_{\mathbf{z}} e^{-i\frac{2\pi}{n}\langle \mathbf{z}, G(\mathbf{y})\rangle} |\mathbf{z}\rangle$$

Now, notice that $H(F(\mathbf{y}))$ is the vector of H applied to the indexes where $\mathbf{y}$ is non-zero, and that $G(\mathbf{y})$ is the vector of values of $\mathbf{y}$ that those points. Thus the inner product is

$$\langle H(F(\mathbf{y})), G(\mathbf{y})\rangle = \sum_{i:y_i\neq 0} H(i) \times y_i = \sum_{i=0}^{k} H(i) y_i = \langle H([k]), \mathbf{y}\rangle \ .$$

The next step is to uncompute the $\mathbf{z}$ and $F(\mathbf{y})$ registers, obtaining

$$|\psi_6\rangle = \frac{1}{\sqrt{C_{k,q,n}}} \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} e^{i\frac{2\pi}{n}\langle H([k]), \mathbf{y}\rangle} |\mathbf{y}\rangle$$

Lastly, we perform a Fourier transform the remaining space, obtaining

$$|\psi_7\rangle = \frac{1}{\sqrt{C_{k,q,n} n^k}} \sum_{\mathbf{z}} \left(\sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} e^{i\frac{2\pi}{n}\langle H([k]) - \mathbf{z}, \mathbf{y}\rangle} \right) |\mathbf{z}\rangle$$

Now measure. The probability we obtain $H([k])$ is

$$\frac{1}{C_{k,q,n} n^k} \left| \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} 1 \right|^2 = \frac{C_{k,q,n}}{n^k}$$

as desired. □

As we have already seen, for exponentially-large $\mathcal{Y}$, this attack has negligible advantage for any $k > q$. However, if $n = |\mathcal{Y}|$ is constant, we can do better. The error probability is

$$\sum_{r=q+1}^{k} \binom{k}{r} \left(1 - \frac{1}{n}\right)^r \left(\frac{1}{n}\right)^{k-r} = \sum_{s=0}^{k-q-1} \binom{k}{s} \left(\frac{1}{n}\right)^s \left(1 - \frac{1}{n}\right)^{k-s} .$$

This is the probability that k consecutive coin flips, where each coin is heads with probability $1/n$, yields fewer than $k - q$ heads. Using the Chernoff bound, if $q > k(1 - 1/n)$, this probability is at most

$$e^{-\frac{n}{2k}(q-k(1-1/n))^2} .$$

For a constant n, let ϵ be any constant with $0 < \epsilon < 1/n$. If we use $q = (1 - \epsilon)k$ queries, the error probability is less than

$$e^{-\frac{n}{2k}(k(1/n-\epsilon))^2} = e^{-\frac{nk}{2}(1/n-\epsilon)^2},$$

which is exponentially small in k. Thus, for constant n, and any constant ϵ with $0 < \epsilon < 1/n$, using $q = (1 - \epsilon)k$ quantum queries, we can determine k input/output pairs with overwhelming probability. This is in contrast to the classical case, where with any constant fraction of k queries, we can only produce k input/output pairs with negligible probability. As an example, if H outputs two bits, it is possible to produce k input/output pairs of of H using only $q = 0.8k$ quantum queries with overwhelming probability. However, with $0.8k$ classical queries, we can output k input/output pairs with probability at most $4^{-0.2k} < 0.76^k$.

3.1.3 Generalizing to Other Distributions

Here we generalize the bound of Section 3.1.1 to non-uniform distributions on oracles. Here, we consider a case where the oracle H is non-uniform, but each output has some significant min-entropy, and show that the bound of Theorem 3.6 still roughly applies.

Theorem 3.8. *Fix sets $\mathcal{X}$ and $\mathcal{Y}$, and distributions D_x on $\mathcal{Y}$ for each $x \in \mathcal{X}$. Let H be a function from $\mathcal{X}$ to $\mathcal{Y}$ where, for each x, $H(x)$ is drawn independently according to D_x. Then any quantum algorithm making q quantum queries to H can only produce $q + 1$ input/output pairs of H with probability at most $(q+1)/\lfloor 2^{H_\infty} \rfloor$, where H_∞ be the minimum over all $x \in \mathcal{X}$ of the min-entropy of the distribution D_x.*

We prove Theorem 3.8 by converting an algorithm violating the theorem to an algorithm violating Theorem 3.6.

First, we need the following technical lemma:

Lemma 3.9. *Fix and integer r. Let D be a distribution of a set $\mathcal{X}$ such that $\Pr[x \leftarrow D] < 1/r$ for all x. Then we can construct a distribution D' on injective functions from $[r]$ into $\mathcal{X}$ with the property that $\Pr[x : f \xleftarrow{R} D', i \xleftarrow{R} [s], x \leftarrow f(i)] = \Pr[x : x \xleftarrow{R} D]$ for all y. In other words, we can generate x according D by drawing a random value i in $[r]$, a random injective function f from D', and evaluating $f(i)$.*

Proof. Pick an arbitrary ordering of elements in $\mathcal{X}$. Then there is a one-to-one correspondence between subsets of $\mathcal{X}$ of size r and strictly monotonically increasing functions from $[r]$ to $\mathcal{X}$. Therefore, it suffices to show how do sample subsets $\mathcal{T} \subseteq \mathcal{X}$ of size r such that sampling $\mathcal{T}$ and then picking a random element of $\mathcal{T}$ simulates the distribution D. We give the algorithm *SampleSubset*, which takes as input a set $\mathcal{X}$, a distribution D on $\mathcal{X}$, and an integer r such that where $\Pr[x \leftarrow D] \leq 1/r$, and samples from a distribution of subsets of size r with the desired properties:

Algorithm 1 *SampleSubset*$(\mathcal{X}, D, r)$

If $r = 1$, draw $x \leftarrow D$, output $\{x\}$, and exit.
Otherwise, let p_L be the smallest non-zero probability in D.
Let p_H be the largest probability in D.
Let $p^* \leftarrow \min(p_L, \frac{1}{r} - p_H)$.
Let $\mathcal{T}$ be the set of the r elements in $\mathcal{X}$ with the smallest non-zero probabilities.
With probability rp^*, output $\mathcal{T}$ and exit.
Otherwise, let D' be the distribution where

$$\Pr[x \leftarrow D'] = \begin{cases} \frac{\Pr[x \leftarrow D] - p^*}{1 - rp^*} & \text{if } x \in \mathcal{T} \\ \frac{\Pr[x \leftarrow D]}{1 - rp^*} & \text{otherwise} \end{cases}$$

Let $\mathcal{F}$ be the set of x such that $\Pr[x \leftarrow D'] = \frac{1}{r}$.
If $|\mathcal{F}| = r$, then output $\mathcal{F}$ and exit.
Otherwise, let D'' be the distribution where

$$\Pr[x \leftarrow D''] = \begin{cases} 0 & \text{if } x \in \mathcal{F} \\ \frac{\Pr[x \leftarrow D']}{1 - |\mathcal{F}|/r} & \text{otherwise} \end{cases}$$

Sample T_0 from *SampleSubset*$(\mathcal{X}, D'', r - |\mathcal{F}|)$
Output $T_0 \cup \mathcal{F}$.

We now prove that *SampleSubsets* works as promised. We need to show that D' and D'' are distributions. Since p^* is at most the smallest probability in D, all the probabilities in D' are non-negative. Moreover, by adding up all the probabilities in D', we see that they sum to 1, so D' is in fact a distribution. This means all the probabilities in D'' are non-negative as well. Using the fact that all elements in $\mathcal{F}$ have probability $1/r$ under D', we see that the probabilities in D'' also sum to 1, so D'' is also a distribution. The fact that D' is a distribution also shows that $|\mathcal{F}| \leq r$, since otherwise the probabilities would sum to greater than 1.

Next, we explain why the recursive call to *SampleSubset* is valid. That is, that $\Pr[x \leftarrow D''] \leq \frac{1}{r'}$

where $r' = r - |\mathcal{F}|$. For D', the maximum probability is at most

$$\frac{p_H}{1 - rp^*} \leq \frac{p_H}{1 - (\frac{1}{r} - p_H)} = \frac{r}{r + (r-1)/p_H} \leq \frac{r}{r + (r-1)r} = \frac{1}{r}$$

For D'', the maximum probability is at most (and in fact less than) $\frac{1/r}{1-|\mathcal{F}|/r} = \frac{1}{r-|\mathcal{F}|} = \frac{1}{r'}$, as desired. Also, *SampleSubset* is never called with $r' = 0$, since in this case we would have already outputted $\mathcal{F}$.

Now, we need to show that this sampling algorithm actually terminates. We look at two cases:

- $p^* = p_L$. Let x_L be an element in $\mathcal{T}$ with $\Pr[x_L \leftarrow D] = p_L$. Observe that under D' and hence D'', x_L has probability 0.
- $p^* = \frac{1}{r} - p_H$. Let x_H be an element with $\Pr[x_H \leftarrow D] = p_H$. Under D', $\Pr[x_H \leftarrow D'] = \frac{1}{r}$, so x_H is included in $\mathcal{F}$. Therefore, under D'', x_H has probability 0.

This means that in each recursive call to *SampleSubset*, the number of x with positive probability decreases by at least 1. Since $\mathcal{X}$ is finite, eventually, the number of x with positive probability will equal r (it cannot be less since all probabilities in D are at most than $1/r$, meaning there are at least r such elements).

It remains to be proven that our sampling algorithm gives the desired distribution. In the case $r = 1$, then we just output sets $\{x\}$ where $x \leftarrow D$, which is correct. Otherwise, with probability rp^*, we output $\mathcal{T}$. In this case, drawing a random value from $\mathcal{T}$ gives us each element with probability p^*. Since p^* is at most p_L, we have not over-sampled any element. If we do not output $\mathcal{T}$, we then need to sample subsets to match the distribution D'. If any x has $\Pr[x \leftarrow D'] = \frac{1}{r}$, then x must be in every subset, so we set it aside in the set $\mathcal{F}$. We then need to draw $r' = r - |\mathcal{F}|$ additional elements not in $\mathcal{F}$ to match the correct distribution. It is straightforward to show that this is achieved by calling $SampleSubset(\mathcal{X}, D'', r - |\mathcal{F}|)$.

□

We are now ready to prove Theorem 3.8:

Proof. Recall that we have an algorithm A making q queries to a random oracle H where outputs are drawn from distributions D_x and produces $q+1$ input/output pairs with probability ϵ. Additionally, for all $x \in \mathcal{X}$, the min-entropy of D_x is at least H_∞.

We now generate H in a different way: for each x, we know that D_x has min-entropy at least H_∞. This means that the most probable element in D_x has probability at most $1/2^{H_\infty} \leq 1/\lfloor 2^{H_\infty} \rfloor$. Let $r = \lfloor 2^{H_\infty} \rfloor$. Lemma 3.9 shows that there is a distribution D'_x on injective functions from $[r]$ to $\mathcal{Y}$ such that sampling from D_x is equivalent to sampling a random $i \xleftarrow{R} [r]$, sampling a random $f \xleftarrow{R} D'_x$, and outputting $f(i)$. Therefore, if we let F be a random oracle from $\mathcal{X}$ to $[r]$, and G an

oracle mapping each $x \in \mathcal{X}$ to a function sampled from D'_x, the oracle that on input x computes $f = G(x)$ and outputs $y = f(x)$ is distributed identically to H.

We can now construct an algorithm B that violates Theorem 3.6. B can make q quantum queries to a random function F from $\mathcal{X}$ into $[r]$. B first builds the function G, and then simulates A, answering A's queries to H using F and G as above. Answering A's queries is potentially problematic since extra information is computed — the outputs of F and G. In order for B to properly answer A's queries without becoming entangled, B must uncompute these extra values. Since B knows G, it can uncompute G easily. Uncomputing F would normally require making a second query to F, but this is unacceptable since then B would make $2q$ queries instead of q. However, the function f outputted by G is injective, meaning we can invert it, which allows us to uncompute the output of F by applying f^{-1} to the output of H. Therefore, each query A makes requires only a single query to F.

With probability ϵ, A produces $q+1$ distinct input/output pairs (x_i, y_i) for H. B then computes the functions $f_i = G(x_i)$, and outputs the pairs $(x_i, f_i^{-1}(y_i))$. These pairs will all be distinct and valid input/output pairs of F. Since F is a random oracle, Theorem 3.6 shows that $\epsilon < (q+1)/r$. Since $r = \lfloor 2^{H_\infty} \rfloor$, this completes the proof.

□

Chapter 4

Other New Techniques

We present several technical results that will be useful for arguing quantum security. We begin by stating all of the lemmas, and in the following sections we will give several applications of these theorems.

We first describe an important lemma that will facilitate many of our proofs. The following theorem relates the behavior of a quantum oracle adversary $\mathcal{A}$ to the distribution on the oracle restricted to small subsets of inputs.

Lemma 4.1. *Let $\mathcal{A}$ be a quantum algorithm making q quantum queries to an oracle $H : \mathcal{X} \to \mathcal{Y}$, and c classical queries. If we draw H from some distribution D, then for every z, the quantity $\Pr_{H \leftarrow D}[\mathcal{A}^{|H\rangle}() = z]$ is a linear combination of the quantities $\Pr_{H \leftarrow D}[H(x_i) = r_i \forall i \in [2q + c]]$ for all possible settings of the x_i and r_i.*

The proof of Theorem 4.1 appears in Section 7.3. This theorem shows, for example, that a $2q$-wise independent function is indistinguishable from a truly random function when given only q quantum oracle queries to the function, a fact that we will use in Section 4.1 to show how to efficiently simulate random oracles.

We now use the theorem to derive two additional theorems that provide means to argue the indistinguishability of distributions of oracles by quantum queries.

Lemma 4.2. *Fix q, and let D_λ be a family of distributions on $\mathcal{Y}^{\mathcal{X}}$ indexed by $\lambda \in [0, 1]$. Suppose there are integers d and Δ such that for every $2q$ pairs $(x_i, r_i) \in \mathcal{X} \times \mathcal{Y}$, the function $p(\lambda) = \Pr_{H \leftarrow D_\lambda}[H(x_i) = r_i \forall i \in [2q]]$ satisfies:*

- *p is a polynomial in λ of degree at most d.*
- *$p^{(i)}(0)$, the ith derivative of p at 0, is 0 for each $i \in [\Delta - 1]$.*

Then any quantum algorithm $\mathcal{A}$ making q quantum queries can only distinguish D_λ from D_0 with probability at most $\frac{4^\Delta}{(2\Delta)!}\lambda^\Delta d^{2\Delta}$.

In particular, if each $p(\lambda)$ is a degree-d polynomial such that $p'(0) = 0$, then $\mathcal{A}$ can only distinguish D_λ from D_0 with probability at most $\frac{2}{3}\lambda^2 d^4$.

Lemma 4.3. *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 are integers d and Δ such that for every $2q$ pairs $(x_i, r_i) \in \mathcal{X} \times \mathcal{Y}$, the function $p(\lambda) = \Pr_{H \leftarrow E_{1/\lambda}}[H(x_i) = r_i \forall i \in [2q]]$ satisfies:*

- *p is a polynomial in λ of degree at most d.*
- *$p^{(i)}(0)$, the ith derivative of p at 0, is 0 for each $i \in [\Delta - 1]$.*

Then any quantum algorithm $\mathcal{A}$ making q quantum queries can only distinguish E_r from E_∞ with probability at most $2^{2-\Delta}\zeta(2\Delta)d^{3\Delta}(1/r)^\Delta$.

In particular, if each $p(\lambda)$ is a degree-d polynomial, then $\mathcal{A}$ can only distinguish E_r from E_∞ with probability at most $\pi^2 d^3/3r$.

These two theorems show that in some cases, we can understand the distinguishing advantage of a quantum oracle algorithm simply by analyzing the oracle distributions themselves.

We note that, by setting $D_{1/r} = E_r$, Lemmas 4.2 and 4.3 appear almost the same. The main difference is that Lemma 4.2 requires D_λ to be a valid distribution for all $\lambda \in [0, 1]$, which in particular implies that $0 \leq p(\lambda) \leq 1$ for all $\lambda \in [0, 1]$. In contrast, Lemma 4.3, when setting $D_{1/r} = E_r$, only requires D_λ to be a distribution on the reciprocals of integers. Thus, in particular, this means $p(\lambda)$ may not be in $[0, 1]$ for, say, $\lambda = 2/3$. Thus, the conditions of Lemma 4.3 are weaker, and as a result a weaker indistinguishability guarantee is obtained. The proofs appear in Sections 7.4 and 7.5.

In Section 4.2, we use Lemma 4.2 to prove that a class of distributions, called *semi-constant* distributions, is indistinguishable from uniform. In Sections 4.3, 4.4, and 4.5, we use Lemma 4.3 to solve a variety of problems.

The next lemma allows us to extract useful information from intermediate stages of a quantum computation without completely destroying the computation. In the classical setting, it is always possible to read the state of the algorithm without disturbing it. However, the quantum uncertainty principle dictates that this is not possible for quantum algorithms: observing the intermediate quantum state of a quantum computation will disturb the state and therefore disturb the outcome of the computation. The following shows that if we only observe a very small portion of the state, the outcome of the computation can still be useful.

Lemma 4.4. *Let $\mathcal{A}$ be a quantum algorithm, and let $\Pr[x]$ be the probability that $\mathcal{A}$ outputs x. Let $\mathcal{A}'$ be another quantum algorithm obtained from $\mathcal{A}$ by pausing $\mathcal{A}$ at an arbitrary stage of execution,*

performing a partial measurement that obtains one of k outcomes, and then resuming $\mathcal{A}$. Let $\Pr'[x]$ *be the probability $\mathcal{A}'$ outputs x. Then* $\Pr'[x] \geq \Pr[x]/k$.

Lemma 4.4 means, for example, that if you measure just one qubit at an intermediate point in a computation, the probability of a particular output drops by at most a factor of two. Lemma 4.4 will be important for several quantum security arguments in Section 6.3

4.1 Application 1: Simulating Random Oracles

In this section, we explain how to efficiently simulate quantum-accessible random oracles. In the classical setting, truly random oracles can be simulated efficiently on fly, only generating the outputs of the oracle as needed. However, with a quantum accessible oracle, it is possible to query the oracle on a superposition of all exponentially-many inputs. The simulator then needs to answer with exponentially-many outputs. Simulating the random oracle perfectly would thus require an exponential amount of randomness, rendering the simulation inefficient. We show that this is not a problem in the case where the number of queries the adversary makes is a priori bounded:

Theorem 4.5. *Any quantum algorithm $\mathcal{A}$ making a polynomially-bounded number quantum queries to random oracles O_i can be efficiently simulated by a quantum algorithm $\mathcal{B}$, which has the same output distribution, but makes no queries.*

Proof. We construct an algorithm $\mathcal{B}$ which simulates $\mathcal{A}$, and answers queries to oracle O_i with evaluations of efficient functions f_i. One possible choice is to use pseudorandom functions (PRF) for the f_i. Such PRFs would need to be secure, even against quantum queries. PRFs of any flavor necessarily involve a computational assumption, which we would like to avoid. 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 4.1, 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 $\mathcal{A}$, as a subroutine of $\mathcal{B}$, behaves identically to the case where $\mathcal{A}$ is given truly random oracles. Hence, the output distribution of $\mathcal{B}$ is identical to that of $\mathcal{A}$.

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 the vector $C \cdot v$. Since any k rows of C are linearly independent, any k elements of $C \cdot v$ 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 random oracles provided provided that the number of elements N in the codomain is a prime power. If N is not a prime power, write $N = N_1 \cdot N_2 \cdots N_s$ where the N_i are prime powers. Then we can write $\mathcal{Y} = \mathcal{Y}_1 \times \mathcal{Y}_2 \times \cdots \times \mathcal{Y}_s$ where $|\mathcal{Y}_i| = N_i$, and construct k-wise independent functions $f_i : \mathcal{X} \to \mathcal{Y}_i$. Then the function $f(x) = (f_1(x), f_2(x), \ldots, f_s(x))$ is a k-wise independent function from $\mathcal{X}$ to $\mathcal{Y}$.

We can also simulate biased random oracles. For example, suppose $O(x)$ is 1 with probability λ, and 0 otherwise. We approximate λ by some rational number a/b where b is a prime power, and construct a k-wise independent function f' with range $\mathcal{Y} = [b]$. Then set

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

This simulates an oracle O_2 where every output bit is chosen to be 1 with independent probability a/b. Since a/b is only an approximation to λ, this does not perfectly simulate the desired O_2. However, by choosing a, b so that $|a/b - \lambda| \leq 2^{-t}/|\mathcal{X}|$, the approximated and desired distributions on oracles will be (2^{-t})-close, so the distributions are quantum indistinguishable. If $|\mathcal{X}| = 2^n$ for a polynomial n, as is the case for our applications, then we can choose a, b to be approximately $t + n$-bits. Therefore, we can simulate O_2 with exponentially-small error in polynomial time.

We will show in Section 4.6 that we can even chose a, b so that $|a/b - \lambda| \leq 2^{-t}$, and the oracles will still be distinguishable with only exponentially-small probability. Therefore, we can actually choose a, b to be approximately t bits, yielding a more efficient simulation.

4.2 Application 2: Semi-Constant Distributions

We now introduce a class of distributions on oracles, which we call *semi-constant* distributions, and we use Lemma 4.2 to argue that the distributions are indistinguishable from random functions by algorithms making quantum oracle queries to the functions. We also present quantum attacks that are within a polynomial factor of optimal.

Definition 4.6. Define $\mathsf{SC}_\lambda(\mathcal{X}, \mathcal{Y})$, the semi-constant distribution, as the distribution over $\mathcal{Y}^\mathcal{X}$ resulting from the following process:

- First, pick a random element y from $\mathcal{Y}$.
- For each $x \in \mathcal{X}$, do one of the following:
  - With probability λ, set $H(x) = y$. We call x a distinguished input to H.
  - Otherwise, set $H(x)$ to be a random element in $\mathcal{Y}$.

When $\mathcal{X}, \mathcal{Y}$ are clear from constant, we usually omit them and just write SC_λ. First, notice that SC_0 is the uniform distribution, since in this case, the probability that an input is distinguished is 0. We bound the ability of a quantum algorithm to distinguish between SC and $\mathsf{SC}_0 = U$ using Lemma 4.2.

Lemma 4.7. *Fix k. For each k pairs (x_i, r_i), the probability $\Pr_{H \leftarrow \mathsf{SC}}[H(x_i) = r_i \forall i \in [k]]$ is a degree-k polynomial in λ whose first derivative is 0 at $\lambda = 0$.*

This, proved below, shows that for any q, we can set $d = k = 2q$ and $\Delta = 2$ in the assumptions of Lemma 4.2. We immediately get:

Corollary 4.8. *The distribution of outputs of a quantum algorithm making q queries to an oracle drawn from SC is at most a distance $\frac{32}{3}q^4\lambda^2$ away from the case when the oracle is drawn from the uniform distribution.*

We do not know if this bound is tight. In Section 4.2.1, we adapt the collision search algorithm of Brassard et al. [BHT97] to SC, obtaining a distinguishing probability of $\Omega(q^3\lambda^2)$.

We now prove Lemma 4.7:

Proof. Recall that SC is defined as follows:

- Pick y 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 $H(x) = y$.

Suppose $\mathcal{Y}$ contains N elements. Let $\alpha(\{x_i\}_{i\in[k]}, \{r_i\}_{i\in[k]}) = \Pr[H(x_i) = r_i \ \forall i \in [k]]$ the probability that x_i maps to r_i for k values of x_i and r_i. Our goal is to show that $\alpha(\{x_i\}_{i\in[k]}, \{r_i\}_{i\in[k]})$, as a function of λ, is a polynomial of degree at most k whose first derivative at $\lambda = 0$ is 0.

We will assume that all of the x_i are distinct. Otherwise, let $i_1 \neq i_2$ be two indices such that $x_{i_1} = x_{i_1}$. If $r_{i_1} = r_{i_2}$, then we can drop x_{i_2} and r_{i_2} without affecting the value of α. In other words,

$$\alpha(\{x_i\}_{i\in[k]}, \{r_i\}_{i\in[k]}) = \alpha(\{x_i\}_{i\in[k]\setminus i_2}, \{r_i\}_{i\in[k]\setminus i_2}) \ .$$

Then we can inductively apply the lemma for tuples of $k-1$ elements, obtaining that

$$\alpha(\{x_i\}_{i\in[k]}, \{r_i\}_{i\in[k]})$$

is a degree $k-1$ polynomial whose first derivative is 0 at $\lambda = 0$.

If $r_{i_1} \neq r_{i_2}$, then $\alpha(\{x_i\}_{i\in[k]}, \{r_i\}_{i\in[k]})$ is identically zero since H cannot take on two values at the same input. The zero function is a degree 0 polynomial whose first derivative is 0 at $\lambda = 0$.

Now suppose there are ℓ distinct r_i, denoted by t_m for $m \in [\ell]$, and let k_m be the number of r_i equal to t_m (note that $\sum_{m=1}^{\ell} k_m = k$). Let $\mathcal{F}$ be the set $\{1, ..., \ell, \perp\}$. For each $f \in \mathcal{F}$, if $f = \perp$, f is associated with the event that $t_m \neq y$ for all m, and otherwise, f is associated with the event that $t_f = y$.

Let $\mathtt{eq}_{\mathtt{f},\mathtt{m}}$ be 1 if $f = m$, 0 otherwise. The probability that $O(x_i) = t_m$ is $1/N$ if we are choosing the output at random, and otherwise, it is 1 if $t_m = y$ and 0 otherwise. In other words, this probability is

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

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

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

Summing over all f gives

$$\alpha(\{x_i\}_{i\in[k]}, \{r_i\}_{i\in[k]}) = \sum_f \Pr[f] \prod_{m=1}^{\ell} \left(\frac{1}{N} + \lambda\left(\mathtt{eq}_{f,m} - \frac{1}{N}\right)\right)^{k_m} .$$

This is a polynomial in λ. It is a sum of products of $\sum k_m = k$ monomials in λ, so its total degree is at most k. Now, we shall approximate this to first order in λ, and show that the first-order coefficient is 0:

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

Notice that $\sum_f \Pr[f]\mathsf{eq}_{f,m}$ is the probability that $t_m = y$, which is equal to $1/N$. Thus, all the terms in the first-order coefficient cancel out, so the λ^1 coefficient is 0. This completes the lemma. □

4.2.1 An Attack on Semi-Constant Distributions

Here we explore the problem of finding a collision in an oracle drawn from a small-range distribution SC over $\mathcal{Y}^{\mathcal{X}}$. Our motivation for studying the collision problem is as follows: a classical algorithm making q queries can only distinguish SC from random with probability $O(\lambda^2 q^2)$, since it can only distinguish the two cases if it happens to query two distinguished points. 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 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, with high probability, all collisions consist of distinguished inputs.

Let c be the minimum constant such that SC is indistinguishable from uniform except with probability $O(q^c\lambda^2)$. Corollary 4.15 shows us 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:

- Let $p = (q-1)/2$, and assume $p\lambda \ll 1$ (since otherwise, a collision could be found easily with a *classical* search). Select a subset $\mathcal{W} \subseteq \mathcal{X}$ of size p. 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 for p queries, looking for an $x \notin \mathcal{W}$ such that $H(x) = H(x_0)$ for some $x_0 \in \mathcal{W}$.
- Output (x, x_0).

The first step uses p queries. If the algorithm is given an oracle from SC, the probability that we find a distinguished input in the first step is

$$1 - (1-\lambda)^{|\mathcal{W}|} = 1 - (1-\lambda)^p = \Omega(p\lambda)$$

Grover's algorithm [Gro96] finds an x such that $O(x) = 1$ with probability $\Omega(p^2 f)$ using p 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 a distinguished input, this fraction is (in expectation) at least λ.

If we found a distinguished input in the first step (which happens with probability $\Omega(p\lambda)$), Grover's algorithm will find an x such that $O(x) = 1$ with probability $\Omega(p^2\lambda)$ using p queries. Thus, our algorithm uses $2p$ queries, and finds a collision with probability $\Omega(p^3\lambda^2)$.

If this algorithm is given a random oracle instead, the probability that we find a collision is negligible. By adding an extra check that the output of the algorithm is indeed a collision (requiring 1 extra query since $H(x_0)$ has already been recorded), we get an algorithm using $2p+1 = q$ quantum queries that distinguishes SC from random with probability $\Omega(p^3\lambda^2) = \Omega(q^3\lambda^2)$, showing that $c \geq 3$.

4.3 Application 3: Quantum Collision Resistance of Random Functions

In this section, we tightly characterize the feasibility of finding collisions in a quantum random oracle. Let M be the number of inputs, and N the number of possible outputs.

Previous Work. Brassard, Høyer, and Tapp [BHT97] give a quantum algorithm (henceforth called the BHT algorithm) requiring $O(M^{1/3})$ quantum queries to any two-to-one function f to produce a collision with overwhelming probability. Ambainis [Amb03] gives an $O(M^{2/3})$ algorithm (which we will call Ambainis's algorithm) for finding a collision in an arbitrary function f, guaranteed that it contains at least one collision. This latter problem is related to the so-called element distinctness problem, where one is asked to distinguish between an injective function and a function with a single collision.

On the lower bound side, most of the prior work has also focused on two-to-one functions. In the case where the co-domain is at least as large as the domain, Aaronson and Shi [AS04], Ambainis [Amb05], and Kutin [Kut05] prove an $\Omega(M^{1/3})$ lower bound for two-to-one functions. The results also generalize to r-to-one functions. These results also imply an $\Omega(M^{2/3})$ lower bound for the element distinctness problem.

While the above results provide matching upper and lower bounds for the problems they analyze, they have a couple crucial limitations:

- Random functions are, with overwhelming probability, not r-to-one functions. Many images will have only one pre-image, and others will have many pre-images. Thus, the above results are not directly applicable to random functions, and hence do not apply to the cryptographic applications discussed above.

- The above lower bounds are proved by showing that $\Omega(M^{1/3})$ queries are necessary to distinguish a two-to-one function from an injective function. Since a collision is a proof that a function is not injective, this implies an $\Omega(M^{1/3})$ lower bound for finding a collision. In essence, these works show that the *collision detection* problem — that is, deciding if there exists a collision — is hard when the co-domain is at least as large as the domain. As the collision detection problem is easier than the *collision finding* problem — actually computing a collision — these lower bounds are *stronger*. However, the collision detection problem is trivial in the case where the co-domain is smaller than the domain, as collisions are guaranteed to exist. Therefore, the works mentioned above cannot be extended to prove anything about the collision finding problem in this setting. For random functions, it is perfectly reasonable to discuss the collision finding problem when the co-domain is much, much smaller than the domain, and moreover, for cryptographic applications, the co-domain is almost always smaller than the domain.

- Finally, the lower bounds only directly make implications about quantum algorithms with high success probability (which we will call high-advantage algorithms). However, for cryptographic applications, we desire security against adversaries that have even a very small success probability (so-called low-advantage algorithms): an attack that works one out of a million times is still considered a break. Therefore, a complete lower bound would bound the number of queries needed to achieve any desired success probability, or equivalently bound the success probability of any algorithm making a specified number of queries. While it may be possible to open up and re-work the proofs to make statements about low-advantage adversaries, existing works do not do this and it is unclear what the result would be.

Yuen [Yue14] overcomes some of the above limitations by proving, in the case that the domain and co-domain are the same size, the BHT algorithm does indeed produce a collision for random functions after $O(N^{1/3})$ queries. For a lower bound, Yuen proves that $\Omega(N^{1/5}/\log N)$ queries are

necessary to produce a collision, using a combination of the r-to-one lower bounds from [AS04] and the quantum adversary method. However, the proof works by proving the indistinguishability of a random function from a random injective function, and thus still requires the co-domain to be at least as large as the domain. Moreover, it does not yield any concrete statements about adversaries with low success probability.

Here, we completely resolve the quantum collision problem for random functions, showing that $\Theta(N^{1/3})$ quantum queries are necessary and sufficient for for finding collisions. Our result also holds with arbitrary domain and co-domain sizes, whereas prior works all required the co-domain to be at least as large as the domain. As such, our result is the first to hold in the case where the co-domain is much smaller than the domain and collisions are guaranteed to exist.

4.3.1 The Lower Bound

Theorem 4.9. *There is a universal constant C such that the following holds. Let $f : [M] \to [N]$ be a random function. Then any algorithm making q quantum queries to f outputs a collision for f with probability at most $C(q+2)^3/N$. Moreover, if $M \leq N$, then then any algorithm making q quantum queries cannot even distinguish f from a random injective function, except with probability at most Cq^3/N.*

Thus, to obtain a collision with bounded error requires $q = \Omega(N^{1/3})$ quantum queries. We prove this theorem using Lemma 4.3.

The case $\mathbf{M \leq N}$. As a first step, we prove the theorem for $M \leq N$. We will prove the distinguishing part of the theorem: any quantum algorithm making q quantum queries cannot distinguish a random function from a random injective function, except with probability Cq^3/N. This implies that the probability of actually finding a collision in a random f is at most $C(q+2)^3/N$: we use two extra queries to check if the collision is correct. The output of this check will distinguish the two cases.

For the proof, we define distributions $\mathcal{D}_r$ on functions from $[M]$ to $[N]$, where sampling is obtained by the following process:

1. Pick a random function $g : [M] \to [r]$.
2. Let $S = \{g(x) : x \in [M]\}$. That is, S is the range of g.
3. Pick a random *injective* function $h : S \to [N]$. (Such functions exist since $|S| \leq M \leq N$)
4. Output the function $f = h \circ g$.

Another way of viewing the distribution $\mathcal{D}_r$ is in terms of collision profiles, used in Yuen's original proof [Yue14]. A collision profile counts, for each i, the number of image points of multiplicity i.

Then the distribution $\mathcal{D}_r$ can be thought of as choosing a collision profile corresponding to a random function from $[M]$ to $[r]$, and then choosing a random function from $[M]$ to $[N]$ with the specified collision profile.

We note three special cases of the distribution $\mathcal{D}_r$:

- $\mathcal{D}_1$. This distribution just outputs a random constant function.
- $\mathcal{D}_N$. This distribution outputs a truly random function from $[M]$ to $[N]$. Indeed, the function g will be a random function, and the function h can be expanded to a random permutation on $[N]$. The composition is therefore a random function.
- $\mathcal{D}_\infty$. This distribution outputs a random injective function from $[M]$ to $[N]$. The function g is, with probability 1, an injective function. Since h is injective, the composition $h \circ g$ is also injective. Since h is a random injective function, the composition is random.

Therefore, our goal is to show that q quantum queries can only distinguish $\mathcal{D}_N$ and $\mathcal{D}_\infty$ with probability Cq^3/N. We need the following lemma:

Lemma 4.10. *Fix k pairs $(x_i, y_i) \in [M] \times [N]$, and let $p(r) = \Pr_{f \leftarrow \mathcal{D}_r}[f(x_i) = y_i \forall i \in \{1, \ldots, k\}]$. Then p is a polynomial in $1/r$ of degree at most $k - 1$.*

Applying Lemma 4.3, we see that $\mathcal{D}_N$ is indistinguishable from $\mathcal{D}_\infty$ with probability $\pi^2(2q)^3/3N$. Setting $C = 8\pi^2/3 < 27$ prove this part of Theorem 4.9.

It remains to prove Lemma 4.10. Due to the symmetry of the distributions $\mathcal{D}_r$, we have that $\Pr_{f \leftarrow \mathcal{D}_r}[f(x) = y] = 1/N$ for any $(x, r) \in [M] \times [N]$. Thus, the claim is true for $k = 1$. Now, assume the claim is true for each $k' < k$.

We can assume without loss of generality that all of the x_i are distinct. If not, and $x_i = x_j$, there are two cases:

- $y_i = y_j$. In this case, we can delete the pair (x_j, y_j) since it is redundant. We then invoke Lemma 4.10 using the remaining $k - 1$ tuples.
- $y_i \neq y_j$. Then $p(r) = 0$ since $f(x_i)$ cannot simultaneously equal y_i and y_j.

Let $\ell = |\{y_i : i \leq k - 1\}|$ be the number of distinct y_i values, not including y_k. Invoking Lemma 4.10 inductively on the first $k - 1$ values, we see that

$$p'(r) = \Pr_{f \leftarrow \mathcal{D}_r}[f(x_i) = y_i \forall i \in \{1, \ldots, k - 1\}]$$

is a polynomial of degree at most $k - 2$ in $1/r$. We now wish to study the conditional probability

$$\Pr_{f \leftarrow \mathcal{D}_r}[f(x_k) = y_k : f(x_i) = y_i \forall i \in \{1, \ldots, k - 1\}]$$

Suppose $y_k = y_i$ for some $i < k$. Since h is injective, this happens exactly when $g(x_k) = g(x_i)$, which happens with probability $1/r$. Now suppose $y_k \neq y_i$ for any $i < k$. This means $g(x_k) \neq g(x_i)$ for all $i < k$, which happens with probability $1 - \ell/r$. In this case, $h(g(x_k)) = y_k$ with probability $1/(N - \ell)$. Then the probability $f(x_k) = y_k$, conditioned on the other values, is $\frac{(1-\ell/r)}{N-\ell}$.

In either case, the value

$$\Pr_{f \leftarrow \mathcal{D}_r}[f(x_k) = y_k : f(x_i) = r_i \forall i \in \{1, \ldots, k-1\}]$$

is a polynomial of degree 1 in $1/r$. Combining with $p'(r)$, we see that $p(r)$ is a polynomial of degree at most $k - 1$ in $1/r$. This completes the proof of Lemma 4.10, and therefore proves Theorem 4.9 for the case $M \leq N$.

The case M > N. If $M > N$, find some K such that $KN \geq M$. Suppose there was an algorithm $\mathcal{A}$ that makes q queries to a random function $f : [M] \to [N]$, and outputs a collision x_1, x_2 with probability ϵ. We now construct an algorithm $\mathcal{B}$ that makes q queries to a random function $q : [M] \to [KN]$ and outputs a collision with probability ϵ/K. $\mathcal{B}$ works as follows:

- Equate the sets $[KN]$ and $[K] \times [N]$ using any arbitrary mapping.
- Let $f : M \to [N]$ be the function obtained from g by dropping the $[K]$ part of the output.
- $\mathcal{B}$ simulates $\mathcal{A}$, and when $\mathcal{A}$ makes a query to f, $\mathcal{B}$ forwards the query to g, and drops the $[K]$ part of the response before returning the response to $\mathcal{A}$. Since quantum operations are reversible, we cannot just disregard the extra part. One approach is to uncompute the bits using a second query to g. However, this doubles the number of queries used. Another better option is to initialize the register where the $[K]$ part will be written to as $\frac{1}{\sqrt{K}} \sum_{i \in [K]} |i\rangle$. Recall that quantum queries are typically implemented by adding the response of the query into supplied register (mod K). After the query is performed, regardless of the output in the $[K]$ part, the state of the register will still be $\frac{1}{\sqrt{K}} \sum_{i \in [K]} |i\rangle$, unentangled with the other registers. Thus we can ignore the register and re-use it for subsequent queries.
- When $\mathcal{A}$ outputs a candidate collision (x_1, x_2), $\mathcal{B}$ outputs (x_1, x_2) as a collision for g.

We note that the $[K]$ part of output of g is completely independent of $\mathcal{A}$'s view. Also, if $f(x_1) = f(x_2)$, then $g(x_1) = g(x_2)$ exactly when the $[K]$ part of the outputs of g are the same in both cases. This happens with probability $1/K$, independent of $\mathcal{A}$'s view. Thus, if $\mathcal{A}$ outputs a collision for f with probability ϵ, $\mathcal{B}$ will output a collision for g with probability ϵ/K.

Now $\mathcal{B}$ solves the case where the co-domain (which has size KN) is larger than the domain (which has size M). Therefore, $\epsilon/K \leq C(q+2)^3/(KN)$, and so we have that $\epsilon \leq C(q+2)^3/N$.

Remark. Notice that, even if we are only interested in high-advantage adversaries making $N^{1/3}$ queries in the $M > N$ case, when invoking Theorem 4.9 for $M \leq N$, we will actually have a low-advantage adversary making potentially far fewer than $N^{1/3}$ queries. Therefore, the fact that Theorem 4.9 handles low-advantage adversaries is crucial for this step, and we cannot rely on results for only high-advantage adversaries, such as most results from the literature. For example, a result showing $q = \Omega(N^{1/3})$ for high advantage adversaries in the $M \geq N$ case, when opened up, could imply a bound of $O(q/N^{1/3})$ for the success probability of a q query algorithm. Applying the reduction above, we would obtain a bound of $O(qM^{2/3}/N)$ for the success probability of a q query algorithm in the $M > N$ setting. Such a result is weaker than what we obtain, and moreover becomes meaningless when $M = \Omega(N^{3/2})$.

4.3.2 An Optimal Attack

Recall the following theorem of Ambainis [Amb03]:

Theorem 4.11 ([Amb03] Theorem 3)**.** *Let $f : [M'] \to [N]$ be a function that has at least one collision. Then there is a quantum algorithm making $O((M')^{2/3})$ quantum queries to f that finds the collision with bounded error.*

We use Ambainis's algorithm as a black box to prove the following:

Theorem 4.12. *Let $f : [M] \to [N]$ be a random function, and suppose $M = \Omega(N^{1/2})$. Then there is a quantum algorithm making $O(N^{1/3})$ quantum queries to f which produces a collision with constant probability.*

The proof is simple, and basically follows the reduction used by Aaronson and Shi [AS04] to prove their lower bound on the element distinctness problem. Let $M' = N^{1/2}/2$. Assume for now that $M' \leq M$. Choose a random subset $S \subset [M]$ of size M'. With probability approaching $1/e$ as N goes to ∞, S will contain at least one collision for f. Let f' be the restriction of f to S, and run Aimbanis's algorithm on f'. Then with probability essentially $1/e$, Ambainis's algorithm will return a collision. The query complexity of Ambainis's algorithm is $O((M')^{2/3}) = O(N^{1/3})$, as desired. In the case where $M \gg M'$, this can be repeated multiple times to obtain arbitrarily high success probabilities.

If M' is not significantly smaller than M, then there is a constant probability p that no collisions will be found. However, we can tweak the above algorithm to give us a collision with probability arbitrarily close to $1 - p$.

4.4 Application 4: Lower Bound for Set Equality

In this section, we give an optimal lower bound for the *Set Equality* problem. In the set equality problem, two injective functions f and g are given with domain $[M]$ and codomain $[N]$ with

$N \geq 2M$, with the promise that either the ranges of f and g overlap completely, or they are disjoint. The goal is to distinguish the two cases. If the ranges are identical, then the BHT collision finding algorithm [BHT97] can be adapted to find a *claw* of f and g: two points x_1, x_2 such that $f(x_1) = g(x_2)$. Thus, using $O(M^{1/3})$ quantum queries, it is possible to distinguish the two cases. Midrijanis [Mid04] shows a $\Omega(M^{1/5}/\log M)$ lower bound for this problem, which is the previous best lower bound.

We show that $\Omega(M^{1/3})$ queries are required to solve the Set Equality Problem, thus giving tight upper and lower bounds. Improving the lower bound to $\Omega(N^{1/3})$ has important implications in relating classical and quantum query complexity. In particular, Aaronson and Ambainis [AA11] show that, for every permutation-symmetric function f, the classical randomized query complexity $D(f)$ is at most the ninth power of the quantum query complexity: $D(f) = \tilde{O}(Q(f)^9)$ (where $\tilde{O}$ hides logarithmic factors). This result uses Midrijanis's lower bound for the set equality problem as a black box. Improving the lower bound to $\Omega(N^{1/3})$ gives an improved $D(f) = \tilde{O}(Q(f)^7)$ for all permutation-symmetric functions.

Theorem 4.13. *Let f and g be injective functions from $[M]$ to $[N]$ ($N \geq 2M$), conditioned on either:*

- *The ranges of f and g are identical*
- *The ranges of f and g are distinct.*

Then any algorithm making q quantum queries can only distinguish the two cases with probability at most $O(q^3/M)$. In particular, any quantum algorithm achieving constant success probability must make $q = \Omega(M^{1/3})$ quantum queries.

Note that Theorem 4.13 is an *worst-case* theorem. Below, we will prove an average-case version, where f and g are *random* injective functions, subject to their ranges being identical or distinct. However, in the case of Set Equality, there is a simple worst-case to average-case reduction showing that these two problems equivalent. Namely, given an average-case distinguisher $\mathcal{A}$, we can construct a worst-case distinguisher $\mathcal{B}$, which constructs random permutations σ_1 and σ_2, and simulates $\mathcal{A}$'s oracles as $\sigma_1 \circ f \circ \sigma_2$ and $\sigma_1 \circ g \circ \sigma_2$. Then the oracles seem by $\mathcal{A}$ are indeed random injective functions, subject to their ranges either being identical or distinct.

We are now ready to prove Theorem 4.13. We consider three distributions on f, g:

(1) $f = h \circ f', g = h \circ g'$ where f', g' are random permutations on $[M]$ and h is a random injective function from $[M]$ to $[N]$. Since f' and g' have the same range, f and g are random injective functions, subject to their ranges being identical.

(2) $f = h \circ f', g = h \circ g'$ where f', g' are random *functions* from $[M]$ to $[M]$ and h is still a random injective function from $[M]$ to $[N]$.

(3) f', g' are random functions from $[M]$ to $[r]$ where $r \geq M$, S is the union of the ranges of f', g', and h is a random injective function from S to $[N]$ (since $|S| \leq 2M \leq N$, such functions exist). As r goes to infinity, f' and g' become random injective with distinct ranges, with probability approaching 1. Then f and g are random injective functions with distinct ranges.

We now argue that case (1) is indistinguishable from case (2), which is indistinguishable from case (3) as r goes to infinity, which completes the proof.

The only difference between (1) and (2) is that f' and g' are switched from random injective functions with co-domain M to random functions. Theorem 4.9 shows that the probability of distinguishing these two cases is $O(q^3/M)$.

Notice that (2) is actually identical to (3) in the case $r = M$. Indeed, the set S in case (3) for $r = M$ can be extended to $[M]$ without changing the distribution of outputs. Therefore, we actually need to show that case (3) with $r = M$ is indistinguishable from case (3) with $r = \infty$.

Now, think of f' and g' as a single function with domain $[M] \times \{0, 1\} \equiv [2M]$ and range $[r]$. That is,

$$(f', g')(x) = \begin{cases} f'(x) & \text{if } x \leq M \\ g'(x - M) & \text{if } x > M \end{cases}$$

Similarly, think of f and g as a single function from $[2M]$ to $[N]$. Then in case (3), the combined function (f', g') is just a random function. Then case (3) actually corresponds to the distributions $\mathcal{D}_r$ from Section 4.3 on functions from $[2M]$ to $[N]$. The analysis from that section shows that $\mathcal{D}_M$ and $\mathcal{D}_\infty$ are indistinguishable, except with probability at most $O(q^3/M)$.

Piecing together, Cases (1) and (3) are indistinguishable, except with probability $O(q^3/M)$, which completes the proof of Theorem 4.13.

4.5 Application 5: Small-Range Distributions

We introduce a new tool, called *small-range distributions*, which allow for simulating a large oracle using very few samples. We then develop new quantum oracle techniques to prove that the simulated oracle is indistinguishable from the large oracle, showing that the simulation is effective.

Let $\mathcal{X}$ and $\mathcal{Y}$ be sets. Given a distribution D on $\mathcal{Y}$, define the small range distribution $\mathsf{SR}_r^D(\mathcal{X})$ 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. 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 $r \ll \mathcal{Y}$ 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 wish to show that $\mathsf{SR}_r^D(\mathcal{X})$ for "small" r is indistinguishable from the corresponding "large-range" function $D^{\mathcal{X}}$ where every output is sampled independently according to D. This will allow us to simulate the large-range $D^{\mathcal{X}}$ using relatively few samples from D. For this, we need the following:

Lemma 4.14. *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.*

We can then use Lemma 4.3 to bound the ability of any quantum algorithm to distinguish $\mathsf{SR}_r^D(\mathcal{X})$ from $\mathsf{SR}_r^D(\mathcal{X}) = D^{\mathcal{X}}$:

Corollary 4.15. *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 Section 4.5.2 we show that the quantum collision finding algorithm of Brassard, Høyer, and Tapp [BHT97] can be used to distinguish $\mathsf{SR}_r^D(\mathcal{X})$ from $D^{\mathcal{X}}$ with optimal probability. This shows that Lemma 4.3 is tight. In Section 4.5.1, we strengthen Corollary 4.15, showing that a small range distribution is indistinguishable from a random function, even when given additional information about the function.

We now prove Lemma 4.14:

Proof of Lemma 4.14. Our goal is to show that, for each of the marginal distributions over k inputs to SR_r^D, each probability is a polynomial in $1/r$ of degree at most k.

Fix some x_i and y_i for $i \in [k]$. We consider the probability that $O(x_i) = y_i$ for all $i \in [k]$. We can assume without loss of generality that the x_i are distinct. Otherwise, there are i, j such that $x_i = x_j$. If $y_i \neq y_j$, then the probability is 0 (O is not a function in this case). If $y_i = y_j$, the $O(x_j) = y_j$ condition is redundant and can be removed, reducing this to the $k-1$ case. By induction on k, the resulting probability is a polynomial of degree at most $k - 1 < k$.

Recall that $\mathsf{SR}_r^D = D^{[r]} \circ [r]^{\mathcal{X}}$ and D is a distribution on $\mathcal{Y}$. Let $O_1 \leftarrow [r]^{\mathcal{X}}$ and $O_2 \leftarrow D^{[r]}$. Let O_1' be the restriction of O_1 to $\{x_0, ..., x_{k-1}\}$. Each O_1' then occurs with probability $1/r^k$. Now,

$$\begin{aligned}
\Pr_{O \leftarrow \mathsf{SR}_r^D}[O(x_i) = y_i \forall i \in [k]] &= \Pr_{O_1 \leftarrow [r]^{\mathcal{X}}, O_2 \leftarrow D^{[r]}}[O_2(O_1(x_i)) = y_i \forall i \in [k]] \\
&= \Pr_{O_1' \leftarrow [r]^{\{x_0,\ldots,x_{k-1}\}}, O_2 \leftarrow D^{[r]}}[O_2(O_1'(x_i)) = y_i \forall i \in [k]] \\
&= \frac{1}{r^k} \sum_{O_1'} \Pr_{O_2 \leftarrow D^{[r]}}[O_2(O_1'(x_i)) = y_i \forall i \in [k]]
\end{aligned}$$

We now associate with each O'_1 a partition P of $[k]$ into r disjoint subsets P_j for $j \in [r]$. The elements of P_j are the indices i such that $O'_1(x_i) = j$. Thus:

$$\Pr_{O \leftarrow \mathsf{SR}^D_r}[O(x_i) = y_i \forall i \in [k]] = \frac{1}{r^k} \sum_{P=(P_j)} \Pr[O_2(j) = y_i \forall j \in [r], \forall i \in P_j]$$

Since $O_2 \leftarrow D^{[r]}$, the distribution of outputs of O_2 for each j are independent. Thus the probabilities $\Pr[O_2(j) = y_i \forall i \in P_j]$ are also independently distributed. Thus,

$$\Pr_{O \leftarrow \mathsf{SR}^D_r}[O(x_i) = y_i \forall i \in [k]] = \frac{1}{r^k} \sum_{P=(P_j)} \prod_{j \in [r]} \Pr[O_2(j) = y_i \forall i \in P_j]$$

Since there are only k elements, at most k of the P_js are non-empty. Thus, we can associate to each partition P another partition Q of $[k]$ into $k_Q \leq k$ non-empty subsets, and a strictly increasing function from f_Q from $[k_Q] \to [r]$. The association is as follows: $Q_{j'} = P_{f_Q(j')}$ and $P_j = \emptyset$ if j has no pre-image under f_Q. This allows us to write:

$$\Pr_{O \leftarrow \mathsf{SR}^D_r}[O(x_i) = y_i \forall i \in [k]] = \frac{1}{r^k} \sum_{Q=(Q_{j'})} \sum_{f_Q} \prod_{j' \in [k_Q]} \Pr[O_2(f_Q(j')) = y_i \forall i \in Q_{j'}]$$

We now notice that, for fixed j', if the y_i are all equal for $i \in Q_{j'}$, then since $O_2 \leftarrow D^{[r]}$, $\Pr[O_2(f_Q(j')) = y_i \forall i \in Q_{j'}] = D(y_i)$ where i is any index in $Q_{j'}$. Otherwise, $\Pr[O_2(j) = y_i \forall i \in Q_{j'}] = 0$ since O_2 needs to be a function. Thus we can write $\Pr[O_2(j) = y_i \forall i \in Q_{j'}] = D(y_i)\sigma(Q_{j'})$ where $\sigma(S)$ is 1 if y_i are all equal for $i \in S$, and 0 otherwise. Thus,

$$\Pr_{O \leftarrow \mathsf{SR}^D_r}[O(x_i) = y_i \forall i \in [k]] = \frac{1}{r^k} \sum_{Q=(Q_{j'})} \sum_{f_Q} \prod_{j' \in [k_Q]} D(y_i)\sigma(Q_{j'})$$

The summand does not depend on f_Q, so let c_Q be the number of f_Q. Then we can write

$$\Pr_{O \leftarrow \mathsf{SR}^D_r}[O(x_i) = y_i \forall i \in [k]] = \frac{1}{r^k} \sum_{Q=(Q_{j'})} c_Q \prod_{j' \in [k_Q]} \Pr[O_2(j') = y_i \forall i \in Q_{j'}]$$

The Q we are summing over are independent of r, as is the product in the above expression. c_Q is equal to the number of ways of picking k_Q distinct elements of $[r]$, which is $\binom{r}{k_Q}$, and is thus polynomial of degree k_Q in r (and hence a polynomial of degree at most k). Therefore, performing the sum, $\Pr_{O \leftarrow \mathsf{SR}^D_r}[O(x_i) = y_i \forall i \in [k]]$ is a polynomial of degree at most k in r, divided by r^k. The result is a polynomial of degree at most k in $1/r$. □

4.5.1 A Strengthening of Corollary 4.15

We now strengthen Corollary 4.15. Recall that we can think of the small range distributions as the composition of a random function $f \in [r]^{\mathcal{X}}$ and a random function $g \in \mathcal{Y}^{[r]}$. Corollary 4.15 shows that the small range distribution $g \circ f$ is indistinguishable from a random function $h \in \mathcal{Y}^{\mathcal{X}}$ using quantum queries. Here, we show that this holds even if the adversary is given (quantum) oracle access to the function f.

Theorem 4.16. *Consider two distributions D_1 and D_2 on oracles from $\mathcal{X}$ into $[r] \times \mathcal{Y}$:*

- *D_1: generate a random oracle $f : \mathcal{X} \to [r]$ and a random oracle $h : \mathcal{X} \to \mathcal{Y}$, and output the oracle that maps x to $(f(x), h(x))$.*
- *D_2: generate a random oracle $f : \mathcal{X} \to [r]$ and a random oracle $g : [r] \to \mathcal{Y}$, and output the oracle that maps x to $(f(x), g(f(x)))$.*

Then the probability that any q-quantum query algorithm distinguishes D_1 from D_2 is at most $O(q^3/r)$.

Proof. Consider the following distributions on oracles $(f(m), h(m))$:

1. f and h are uniformly random oracles. This is the distribution D_1.
2. Let $O : \mathcal{X} \to [r]$, $f' : [r] \to [r]$ and $h' : [r] \to \mathcal{Y}$ be random oracles. Let $f(x) = f'(O(x))$ and $h(x) = h'(O(x))$. Think of (f, h) as a single random oracle from $\mathcal{X}$ into $[r] \times \mathcal{Y}$, (f', h') as a single random oracle from $[r]$ into $[r] \times \mathcal{Y}$, and $(f, h) = (f', h') \circ O$. We therefore have that (f, h) itself is a small-range distribution. Thus, Corollary 4.15 shows that this is indistinguishable from (1), except with probability $O(q^3/r)$.
3. Let O, h', f, h be as in 1, except let f' be a random permutation on $[r]$. Theorem 4.9 shows that this distribution is indistinguishable from (2) except with probability $O(q^3/r)$.
4. Let f', h', h be as in (3), except now draw f as a random function from $\mathcal{X}$ to $[r]$, and set $O(m) = {f'}^{-1}(f(x))$. This is the same distribution on oracles (f, h) as in (3). The oracle seen by the adversary is $(f(m), h(m) = h'({f'}^{-1}(f(m))))$ where f, h are random and f' is a random permutation. Now define $P(x) = h'({f'}^{-1}(x))$ so that $h(x) = P(f(m))$. Since h' is random and f' is a permutation, this distribution on P is random. Therefore, this is distribution D_2

It follows that D_1 and D_2 are indistinguishable, except with probability $O(q^3/r)$ as desired. □

4.5.2 An Optimal Attack on Small-Range Distributions

In this section, we give a quantum distinguisher that distinguishes $\mathsf{SR}_R^{\mathcal{Y}}$ from a random function with probability (asymptotically) matching the bound of Corollary 4.15. Our algorithm is basically the

collision finding algorithm of Brassard, Høer, and Tapp [BHT97], with a check at the end to verify that a collision is found. The algorithm has oracle access to a function O from $\mathcal{X}$ to $\mathcal{Y}$, which is either $\mathsf{SR}_r^{\mathcal{Y}}$ or a random function. It is given as input the integer r, the number of queries q, and operates as follows:

- Let $p = (q-1)/2$. Pick a set S of p points in $\mathcal{X}$ at random, and check that there is no collision on S by making p classical queries to O. Sort the elements of S, and store the pairs $(s, O(s))$ as a table for efficient lookup.
- Construct the oracle $O'(x) = \begin{cases} 1 & \text{if } x \notin S \text{ and } O(x) = O(s) \text{ for some } s \in S \\ 0 & \text{otherwise} \end{cases}$
- Run Grover's algorithm [Gro96] on O' for p iterations to look for a point x such that $O'(x) = 1$.
- Check that there is an $s \in S$ such that $O(x) = O(s)$ by making one more classical query to O.

Before analyzing this construction, we explain what Grover's algorithm does. It takes as input an oracle O' mapping some space $\mathcal{X}$ into $\{0,1\}$, and tries to find an x such that $O'(x) = 1$. Specifically, if N points map to 1, then after q queries to O', Grover's algorithm will output an x such that $O'(x) = 1$ with probability $\Theta(q^2 N/|\mathcal{X}|)$

We now analyze this construction. The first step takes p queries to O. If we find a collision, we are done. Otherwise, we have p points that map to p different values. Call this set of values T. The oracle O' outlined in the second step makes exactly one query to O for each query to O'. The number of points in x such that $O'(x) = 1$ is the number of points x in $\mathcal{X} \setminus S$ (which is $|X| - p$) such that $O(x) \in T$. In the random oracle case, the probability that $O(x)$ is one of p random values is $p/|\mathcal{Y}|$, so the expected number of such x is $(|\mathcal{X}| - p)p/|\mathcal{Y}|$. Thus, after p iterations, Grover's algorithm will output such an x with probability $\Theta(p^3(|\mathcal{X}| - p)/|\mathcal{X}||\mathcal{Y}|))$. In the $\mathsf{SR}_r^{\mathcal{Y}}$ case, since there are only r possible outputs, the probability that x maps to T is p/r, so the expected number of such x is $p(|\mathcal{X}| - p)/r$. Thus, Grover's algorithm will output such an x with probability $\Theta(p^3(|\mathcal{X}| - p)/r|\mathcal{X}|)$.

The difference in these probabilities is $\Theta(p^3(1/r - 1/|\mathcal{Y}|)(|\mathcal{X}| - p)/|\mathcal{X}|$. If we let $|\mathcal{Y}|$ be at least $2r$ and $|\mathcal{X}|$ at least $2p + 1 = q$, we see that we distinguish $\mathsf{SR}_r^{\mathcal{Y}}$ from random with probability $\Omega(p^3/r) = \Omega(q^3/r)$, thus matching the bound of Corollary 4.15. This shows that the corollary is optimal, and hence Lemma 4.3 is optimal for the case $\Delta = 0$.

4.6 Application 6: An Oracle Indistinguishability Theorem

In this section, we prove a general theorem about the indistinguishability of oracles by quantum queries. The setup is as follows: there is a set $\mathcal{Y}$ and two distributions D_0, D_1 on $\mathcal{Y}$. We consider two cases, where either (1) D_0 and D_1 are statistically indistinguishable, or (2) where D_0 and D_1 are computationally indistinguishable. We now consider the oracles $D_0^{\mathcal{X}}$ and $D_1^{\mathcal{X}}$ for arbitrarily large

sets $\mathcal{X}$, and ask whether they are statistically (resp. computationally) indistinguishable. If they are, we call the distributions D_0 and D_1 statistically (resp. computationally) *oracle* indistinguishable.

We show that oracle indistinguishability is actually equivalent to regular indistinguishability:

Theorem 4.17. *Let D_0 and D_1 be efficiently sampleable distributions on a set $\mathcal{Y}$, and let $\mathcal{X}$ be some other (potentially exponential-sized) set. Suppose $\mathcal{A}$ is a computationally unbounded (resp. efficient) quantum algorithm that uses q quantum queries to distinguish oracles $D_0^{\mathcal{X}}$ from $D_1^{\mathcal{X}}$ with advantage ϵ. Then there is a computationally unbounded (resp. efficient) quantum algorithm $\mathcal{B}$ that distinguishes D_0 from D_1 with advantage $\epsilon^2/216q^3$.*

In particular, if D_0 and D_1 are statistically (resp. computationally) indistinguishable, then they are also statistically (resp. computationally) oracle indistinguishable.

If allowing only classical queries, then Theorem 4.17 is straightforward to prove. Since the adversary only makes q classical queries to the oracle H drawn from $D_b^{\mathcal{X}}$, its view can be simulated efficiently using only q samples from D_b. Essentially, every classical query is answered using a new sample. We thus obtain an algorithm distinguishing q samples from D_0 from D_1 with probability ϵ. A simple hybrid argument then shows how to distinguish a single sample of D_0 from D_1 with probability ϵ/q. The reduction is very efficient, and therefore works in both the computational and statistical settings.

In the quantum query case, however, the above argument fails because the adversary can query on a superposition of all exponentially-many inputs. Thus perfect simulation of $D_b^{\mathcal{X}}$ would require exponentially-many samples of D_b. The result would be a distinguisher that has advantage $\epsilon/|\mathcal{X}|$, which is negligible even for large ϵ. Moreover, it is not clear how to carry out the reduction efficiently.

In the statistical setting, it is straightforward to prove a weaker version of Theorem 4.17 using the hybrid method of [BBBV97]. However, the argument inherently uses the statistical closeness of D_0 and D_1, and it is not clear how ot extend the argument to the computational setting, which requires a reduction. We now show how to give such a reduction using small range distributions.

Proof of Theorem 4.17. Let B be an (efficient) quantum adversary $\mathcal{A}$ that distinguishes $D_0^{\mathcal{X}}$ from $D_1^{\mathcal{X}}$ with probability ϵ, for distributions D_0 and D_1 over $\mathcal{Y}$. That is, there is some set $\mathcal{X}$ such that

$$\left| \Pr_{O \leftarrow D_0^{\mathcal{X}}}[\mathcal{B}^{|O\rangle}() = 1] - \Pr_{O \leftarrow D_1^{\mathcal{X}}}[\mathcal{B}^{|O\rangle}() = 1] \right| = \epsilon \ .$$

Our goal is to construct an (efficient) quantum algorithm $\mathcal{B}$ that distinguishes a single sample of D_0 from a sample of D_1. To this end, choose r so that $\ell(q)/r = \epsilon/4$, where $\ell(q)$ is the polynomial from Corollary 4.15 (namely $\ell(q) = 27q^3$). That is, $r = 4\ell(q)/\epsilon$. No quantum algorithm can distinguish $\mathsf{SR}_r^{D_b}(\mathcal{X})$ from $D_b^{\mathcal{X}}$ with probability greater than $\ell(q)/r = \epsilon/4$. Thus, it must be that

$$\left| \Pr_{O \leftarrow \mathsf{SR}_r^{D_0}(\mathcal{X})}[\mathcal{A}^{|O\rangle}() = 1] - \Pr_{O \leftarrow \mathsf{SR}_r^{D_1}(\mathcal{X})}[\mathcal{A}^{|O\rangle}() = 1] \right| \geq \epsilon/2$$

We now define $r+1$ hybrids H_i as follows: For $j = 0, ..., i-1$, draw y_j from D_0. For $j = i, ..., r-1$, draw y_j from D_1. Then give B the oracle O where for each x, $O(x)$ is a randomly selected y_i. H_r is the case where $O \leftarrow \mathsf{SR}_r^{D_0}$, and H_0 is the case where $O \leftarrow \mathsf{SR}_r^{D_1}$. Hence H_0 and H_r are distinguished with probability at least $\epsilon/2$. Let

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

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

We construct an algorithm $\mathcal{B}$ that distinguishes between D_1 and D_2 with probability $\epsilon/2r$. $\mathcal{B}$, 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 $O_1 \leftarrow D_0^{\{0,...,i-1\}}$ and $O_2 \leftarrow D_1^{\{i+1,...,r-1\}}$.
- 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 $\mathcal{A}$ with the oracle O, and output the output of $\mathcal{A}$.

Let $\mathcal{B}_i$ be the algorithm $\mathcal{B}$ using i. If $y \leftarrow D_0$, $\mathcal{A}$ sees hybrid H_{i+1}. If $y \leftarrow D_1$, $\mathcal{A}$ sees H_i. Therefore, we have that

$$\Pr_{y \leftarrow D_0}[\mathcal{B}_i(y) = 1] - \Pr_{y \leftarrow D_1}[\mathcal{B}_i(y) = 1] = \epsilon_i \ .$$

Averaging over all i, we get that $\mathcal{B}$'s distinguishing probability is

$$\left|\Pr_{y \leftarrow D_0}[\mathcal{B}(y) = 1] - \Pr_{y \leftarrow D_1}[\mathcal{B}(y) = 1]\right| = \left|\frac{1}{r}\sum_{i=1}^{r} \epsilon_i\right| \geq \frac{\epsilon}{2r} = \frac{\epsilon^2}{8\ell(q)} \ .$$

Thus, $\mathcal{B}$ is an (efficient) algorithm that distinguishes D_0 from D_1 with non-negligible probability. Hence, it breaks the indistinguishability of D_0 and D_1. We note that the above conversion from $\mathcal{A}$ to $\mathcal{B}$ is efficient, meaning the proof works equally well in the computational and statistical settings. □

4.6.1 A Strengthening of Theorem 4.17 in the Statistical Setting

Here, we generalize Theorem 4.17 to the case where the oracles the adversary is distinguishing have different distributions for every output. That is, the adversary is given an oracle O from either of

two distributions on oracles. The oracle distributions are such that each output is independently (though not necessarily identically) distributed, where D_x is the distribution of outputs for input x for the first oracle distribution, and D'_x is the corresponding output distribution for the second oracle distribution. We further require that D_x and D'_x are statistically close, and want to argue that the resulting oracle distributions are statistically indistinguishable by quantum queries.

Theorem 4.18. *Let $\mathcal{X}$ and $\mathcal{Y}$ be sets, and for each $x \in \mathcal{X}$, let D_x and D'_x be distributions on $\mathcal{Y}$ such that $|D_x - D'_x| \leq \epsilon$ for some value ϵ that is independent of x. Let $O : \mathcal{X} \to \mathcal{Y}$ be a function where, for each x, $O(x)$ is drawn from D_x, and let $O'(x)$ be a function where, for each x, $O'(x)$ is drawn form D'_x. Then any quantum algorithm making at most q queries to either O or O' cannot distinguish the two, except with probability at most $\sqrt{216q^3\epsilon}$.*

We note that if all the D_x are the same and all the D'_x are the same, then Theorem 4.18 follows immediately form the statistical case of Theorem 4.17. We now explain how to prove the more general case..

We first suppose that each of the probabilities in each of the distributions D_x and D'_x are rational.

Claim 4.19. *If each of the probabilities in D_x and D'_x are rational, then any quantum algorithm making q quantum queries can only distinguish O from O' with probability $\sqrt{216q^3\epsilon}$.*

Before proving this claim, we explain how it proves Theorem 4.18. Fix any quantum algorithm $\mathcal{A}$. The distinguishing probability for any rational collection of distributions D_x and D'_x is bounded by $\sqrt{216q^3 \max_x |D_x - D'_x|}$. But the distinguishing probability of $\mathcal{A}$ is a continuous function of the probabilities in the distributions D_x and D'_x, and the pairs of rational distributions are dense in the set of all pairs of distributions. Therefore, the bound of $\sqrt{216q^3 \max_x |D_x - D'_x|}$ applies for all pairs of distributions.

Now we prove the claim:

Proof. Let r be the smallest integer such that each of the probabilities in each of the distributions D_x and D'_x can be represented as a rational number with denominator r. Observe that we can take ϵ to be an integer times $2/r$, say $2s/r$. Let $\mathcal{Z} = [s+r]$. Let E be the uniform distribution on $[r]$ and E' the uniform distribution on $[r]+s = \{s+1, ..., s+r\}$. The probabilities in E and E' are the same on $[s]+(r-s) = \{s, ..., r\}$, and are $1/r$ on$[s]$ and $[s]+r$ respectively. Therefore $|E - E'| = 2s/r = \epsilon$. We now construct functions f_x such that if $z \leftarrow E$, $f_x(y)$ is distributed according to D_x and if $z \leftarrow E'$, $f_x(y)$ is distributed according to D'_x. For each $y \in \mathcal{Y}$, let p/r be the probability under D_x and p'/r the probability under D'_x. Suppose $p \leq p'$. Then we will choose p elements of $[s] + (r - s)$ that have not been chosen before, and let f_x evaluate to y on those elements. We will also choose $p' - p$ elements of $[s] + r$ and let f_x be y on those elements as well. We treat the $p' < p$ case similarly. Then f_x evaluates to y with the desired probabilities, so it remains to show that we never run out elements. Since $|D_x - D'_x| \leq 2s/r$, we will never run out of elements in $[s] + r$ or $[s]$. If

$|D_x - D'_x| < 2s/r$, we will run out of elements in $[s] + (r - s)$. When we run out, however, instead of picking an element in $[s] + (r - s)$, we can pick two elements, one in each of $[s]$ and $[s] + r$, and still have the correct probability.

Now that we can generate $D_x = f_x \circ E$ and $D'_x = f_x \circ E'$, we can generate O and O' differently. Let P be the set of oracles from $\mathcal{X}$ to $\mathcal{Z}$ where each output is drawn according to E, and let P' be the set of oracles where each output is drawn from E'. Then letting $O(x) = f_x(P(x))$ and $O'(x) = f_x(P'(x))$ gives the correct distributions for O and O'. Suppose A distinguishes O from O' with probability σ. Then we can easily construct an algorithm B that distinguishes P and P' with probability σ.

Let ℓ be some integer to be chosen later. We replace P and P' with small-range distributions on ℓ samples of E and E' respectively. Applying Corollary 4.15 twice shows that B must still distinguish P and P' with probability at least $\sigma - 2 \times 27q^3/\ell$. But now the difference between the distribution P and P' is only ℓ samples of either E or E', so the distinguishing probability is at most $\ell\epsilon$. Thus $\sigma \leq \ell\epsilon + 2 \times 27q_4/\ell$ for any ℓ. Setting $\ell = \sqrt{2 \times 27q^3/\epsilon}$ minimizes this quantity, yielding $\sqrt{216q^3\epsilon}$ as desired. □

Chapter 5

The Quantum Random Oracle Model

In this chapter, we explore the *quantum random oracle model*.

The classical random oracle model. The random oracle model was introduced by Bellare and Rogaway [BR93] as a way to analyze protocols that had so far resisted security proof. In the random oracle model, a hash function h is modeled as a truly random function H that can only be evaluated — either in the protocol or by the adversary — by making oracle queries to H. Most protocols using hash functions only make black-box use of the function[1], and so these protocols can be rewritten as protocols making oracle queries to H. Then the security of the scheme is analyzed in this setting. The random oracle model intuitively captures adversaries that do not exploit any particular structure of the hash function h, and instead only evaluate h on arbitrarily chosen inputs.

The advantage of the random oracle model is that many of the most efficient schemes *only* can be proved secure in this model. The disadvantage is that a security proof in the random oracle model does not give a formal proof in the plain model (without random oracles), and therefore only provides a heuristic security argument.

Modeling random oracles in a quantum world. When the adversary is equipped with a quantum computer, the most efficient schemes will still likely still lack standard model proofs, and therefore the random oracle heuristic remains important. How should the oracle be modeled? One solution is to simply model the random oracle as a *classical* oracle. In this setting, many classical security proofs carry through, assuming the underlying primitives are quantum resistant. However, when implementing the protocol in the standard model, the random oracle H is replaced with a

[1] Some exceptions would be protocols using non-black-box tools such as zero knowledge proofs and program obfuscation

hash function h, which the adversary knows the code for. Thus, in the real world, the adversary can evaluate h on a quantum superposition of inputs. Such an evaluation is not exploiting any particular structure of h: it is simply using the fact that the adversary can evaluate any classical function on a superposition. Therefore, to maintain the intuition of capturing quantum attacks that do not exploit the structure of h, we need to model the random oracle H as a quantum accessible oracle.

Unfortunately, many classical random oracle security proofs fail when quantum queries are allowed. A couple potential issues arise:

- In a security reduction in the classical random oracle model, the reduction algorithm can see the adversary's queries, and try to use the query inputs to solve some underlying problem. If the adversary makes a quantum query, the reduction algorithm can only learn anything about the query by measuring it, which destroys the state. As a consequence of the no-cloning theorem in quantum physics, the reduction will then be unable to give the correct state back to the adversary when it answers the query. A quantum adversary can, for example, then detect that the state was tampered with and abort, preventing the reduction from making further use of the adversary.

- An efficient classical algorithm can only make a polynomial number of queries, and thus can only see a polynomial number of outputs of the oracle. This fact is crucial in many classical arguments. For example, if the reduction answers the queries with a distribution that are only ϵ-close to random, the success probability of an algorithm that makes q classical queries will only be affected by $q\epsilon$. If ϵ is negligible, so is $q\epsilon$.

  However, a quantum adversary, given even one query, can query on a quantum superposition of all possible inputs, and therefore "see" the entire oracle. In the ϵ-close setting above, the straightforward argument bounds the affect on the adversary to $2^n\epsilon$, where n is the number of input bits to the oracle. Even if ϵ is negligible, $2^n\epsilon$ could be greater than 1, resulting in a meaningless bound.

- A similar problem arises when a reduction embeds a challenge into a random oracle query. In many classical proofs, the reduction picks a random oracle query, and embeds some challenge into the answer for that query, and hopes that the adversary "uses" that query, which happens with probability $1/q$ where q is the number of oracle queries made.

  It is not clear how to translate this strategy to the case where the adversary can query on a superposition. If the reduction embeds the challenge into all answers for a random query, the answers will be inconsistent with previous and future queries. Moreover, the answer itself will not consist of independent random responses. If the reduction embeds the challenge into just a single output consistently across all queries, the adversary is exponentially unlikely to use that output. The result is that the reduction is exponentially unlikely to succeed.

Thus proving security in the quantum random oracle model will often require new proof strategies specifically tailored to quantum adversaries that can make quantum queries to the oracle. In this section, we present several such proofs. First, however, we show that the quantum random oracle model is "stronger" that the classical model by exhibiting a scheme that is secure in the classical random oracle mode (against quantum adversaries) but is insecure under quantum queries. Moreover, implementing the random oracle with a specific type of concrete hash function results in a classically secure protocol, but *any* instantiation of the random oracle results in a quantum insecure protocol. Our separation gives concrete support for the claim that the quantum random oracle model is the "right" model for quantum adversaries.

5.1 A Separation

Here, we formalize the above arguments. In particular, under appropriate computational assumptions, we give a two party protocol in the random oracle model such that:

- The protocol is secure against classical adversaries when the random oracle is instantiated with an appropriate hash function.
- The protocol is secure against quantum and classical adversaries that only have *classical* access to the random oracle.
- The protocol is insecure against quantum adversaries that are given *quantum* access to the random oracle.
- The protocol is insecure against quantum adversaries when the random oracle is instantiated with *any* hash function.

This highlights the need for defining allowing quantum access to random oracles in the quantum setting, as classical access does not model the types of attacks the adversary can perform.

We note that our results rely on assumptions that are somewhat non-standard in the cryptographic community. Nonetheless, we argue below that the assumptions are reasonable.

5.1.1 Preliminaries

Identification Schemes. An identification scheme consists of three efficient algorithms (Gen, $\mathsf{Prove}, \mathsf{Ver}$) where Gen outputs a key pair $(\mathsf{sk}, \mathsf{vk})$. The joint execution of $\mathsf{Prove}(\mathsf{sk})$ and $\mathsf{Ver}(\mathsf{vk})$ then defines an interactive protocol between the prover Prove and the verifier Ver. At the end of the protocol Ver outputs a decision bit $b \in \{0, 1\}$. We assume completeness in the sense that for any honest prover the verifier accepts the interaction with output $b = 1$. Security of identification schemes is usually defined by considering an adversary $\mathcal{A}$ that first interacts with the honest prover to obtain some information about the secret key. In a second stage, the adversary then plays the

role of the prover and has to make a verifier accept the interaction. We say that an identification scheme is *sound* if the adversary can convince the verifier with negligible probability only.

Assumptions. We will need to make several assumptions for our results. First, we will need an identification protocol that is secure against quantum adversaries. Such protocols can be built from any EUF-CMA secure signature scheme (that is, any signature scheme secure against quantum adversaries making classical signing queries), which can in turn be built from any quantum-immune one-way function by translating [Rom90] to the quantum setting.

We will also need to make some slightly non-standard assumptions. The first says that parallel computation can only result in a bounded speedup.

Assumption 1 (Bounded Parallel Speed-Up)**.** There is a constant α such that any computation running in parallel time t can be computed sequentially using time αt.

Assumption 1 reflects the fact that in the real world there is only a concrete and finite amount of equipment available that can contribute to such a performance gain. Indeed, if there are only α processors available, any parallel-time t-time computation on the processors can be carried out in time αt on a single processor.

For any integer k, let $t = C2^{k/2}$ be the number of queries that Grover's algorithm takes to find pre-images with probability $1 - O(2^{-k/2})$ on functions of output size k. We will say a function f is a *quantum-gap* one-way function with output size k if no classical algorithm running in parallel time $t \cdot \lambda^{O(1)}$ can invert f on a random input, except with probability at most $1/2$, where $\lambda^{O(1)}$ represents the running time of f.

Assumption 2 (Quantum-gap hash functions)**.** For some $k = O(\log \lambda)$, there exist quantum-gap gap one-way functions whose evaluation time is $\mathsf{poly}(\lambda)$ and have output length k.

To see why Assumption 2 is reasonable, consider an "ideal" one-way function f where finding pre-images classically requires time essentially equal to a brute-force search. More precisely, for output size of k bits (and input size much larger than k bits), and given classical (sequential) time t, the probability of inverting f on a random input is at most approximately $t/2^k$. Such ideal behavior is commonly assumed from existing hash function candidates such as SHA-256. Using Assumption 1, the probability of inverting f in parallel time t is at most $\alpha t/2^k$. Setting $t = C2^{k/2}$, we get a probability of $C\alpha/2^{k/2}$, which is less than $1/2$ for $k \geq 2\log(C\alpha) + 2$, a constant (whereas we allow $k = O(\log \lambda)$). Of course, hash functions usually have k as big as the security parameter (for example, SHA-256 has $k = 256$), whereas we need such hash functions even for very small k. Such functions can be achieved by simply truncating the output of a longer function to k bits.

5.1.2 Construction

We now present our identification scheme between a prover Prove and a verifier Ver. The main idea is to augment a (quantum) secure identification protocol $(\mathsf{Gen}, \mathsf{Prove}', \mathsf{Ver}')$ by an inverting stage for a function f. In the first stage, the verifier checks if the prover is able to invert f on random inputs in a given amount of time. If inversion is successful, the verifier accepts. Otherwise, it reverts to running protocol between Prove' and Ver', and accepts if only if the $(\mathsf{Prove}', \mathsf{Ver}')$ accepts.

If f is a quantum-gap one-way function, and the verifier allows time $t = C2^{k/2} \cdot \lambda^{O(1)}$ (where the $\lambda^{O(1)}$ represents the time required to evaluate the function) for inversion, then any classical adversary has probability $1/2$ of inverting. By having multiple inversion challenges, we can make a classical adversary's success probability arbitrarily small, and hence the soundness of the augmented protocol reduces to the soundness of the underlying protocol. However, regardless of how f is instantiated, a quantum adversary can, in time t, invert f with high probability, causing the verifier to accept and thus breaking the protocol. Therefore, $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver})$ is classically secure for some f, but quantum insecure for any f.

We can extend this to the case where f is modeled as a random oracle H. Here, in time t, only t sequential queries can be made to H, and by Assumption 1, only αt parallel queries can be made. Then it is easy to see that with classical queries, an adversary can only invert f with probability at most $\alpha t/2^k$. Again, setting $t = C2^{k/2}$, this value is less than $1/2$ for sufficiently large k, and by having multiple inversion challenges, a classical adversary will fail with overwhelming probability. However, using t quantum queries, H can be inverted with near certainty. Thus, the protocol is secure in the classical random oracle model (even against quantum adversaries) but insecure in the quantum random oracle model.

Construction 5.1. Let $(\mathsf{Gen}, \mathsf{Prove}', \mathsf{Ver}')$ be a quantumly secure identification protocol. Define $\mathsf{Prove}^H, \mathsf{Ver}^H$ as the following augmented random oracle protocols, where the domain of H is $\mathcal{X}$ and the output size of H is $k = O(\log \lambda)$ bits for a security parameter λ. Let $t = C2^{k/2} = \lambda^{O(1)}$ be the number of queries required by Grover's algorithm. Let $n = O(\lambda)$ be some parameter.

Verification. The algorithm Ver^H works as follows.

- Ver^H first initializes a counter c.
- For $i = 1, ..., n$,
  - choose a random $x_i \in \mathcal{X}$, compute $y_i = H(x_i)$, and send y_i to the prover.
  - For $j = 1, \ldots, t$, compute $H(it + j)$, and discard the result.
  - If at any point prior to computing $H(it + t)$, Ver^H receives an $x_i' \in \mathcal{X}$, check if $H(x_i') = H(x_i)$. If equality holds, set $c \leftarrow c + 1$.
  - Otherwise, continue.

- If $c = n$, terminate and accept.
- Otherwise, initiate the Ver' protocol with the prover.
- If Ver' accepts, Ver^H accepts. If Ver' rejects, Ver^H rejects.

Proving. The algorithm Prove^H works as follows. Upon receiving each y_i from the verifier, Prove^H does nothing and waits. Then when the verifier moves to initiating Ver', the prover initiates the Prove' protocol.

Completeness of the $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver})$ protocol follows easily from the completeness of the underlying $(\mathsf{Gen}, \mathsf{Prove}', \mathsf{Ver}')$ scheme. We now move to arguing (in)security. For simplicity, we will consider evaluating H as taking unit time. Then each iteration of the inversion stage allows time $t = C2^{k/2}$. In time t, Grover's algorithm can invert H on one of the y_i with probability $1-O(2^{-k/2})$, regardless of whether f is a quantum random oracle or implemented by a concrete hash function. Thus it can invert *all* of the y_i with probability at least $1 - O(n2^{-k/2}) = 1 - O(n/\lambda^O(1))$. Choosing n to be a sufficiently small polynomial in λ, the next two Lemmas easily follow:

Lemma 5.2. $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver})$ *is insecure in the quantum random oracle model.*

Lemma 5.3. $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver})$ *is insecure against quantum adversaries for any concrete instantiation of the oracle H by a function f.*

On the positive side, we consider two possible scenarios. In the first, we implement H with any quantum-gap one-way function f, and consider security against only *classical* adversaries. The adversary has parallel time t to for each y_i to invert f. By the quantum-gap property, the adversary succeeds for each y_i with probability at most $1/2$. Therefore it inverts f on all the y_i with probability $1/2^{\lambda^{O(1)}} = \mathsf{negl}$. In the second case, we leave H as an oracle, consider both quantum and classical adversaries, but only allow classical access to the random oracle. In this setting as well, the adversary only succeeds at inverting every challenge with negligible probability. Therefore, the security of $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver})$ reduces to the security of $(\mathsf{Gen}, \mathsf{Prove}', \mathsf{Ver}')$. Thus, we obtain the following:

Lemma 5.4. *If* $(\mathsf{Gen}, \mathsf{Prove}', \mathsf{Ver}')$ *is secure, then* $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver})$ *is secure against classical adversaries when H is implemented by a quantum-gap one-way function f.*

Lemma 5.5. *If* $(\mathsf{Gen}, \mathsf{Prove}', \mathsf{Ver}')$ *is secure, then* $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver})$ *is secure against both classical and quantum adversaries that are limited to making classical queries to H.*

Therefore, if we want the random oracle model to best represent the real world (in which quantum algorithms can always break the scheme), we need the random oracle to be quantum accessible.

5.2 Signatures in the QROM

Now we discuss how to build digital signatures that are secure in the quantum random oracle model (QROM).

5.2.1 GPV Signatures

Here we show that the signature scheme of Gentry, Peikert, and Vaikuntanathan [GPV08] is secure in the QROM. The GPV signature scheme is closely related to of the Full Domain Hash [BR93] scheme, and is build from pre-image sampleable functions (PSFs), which can in turn be build from lattices. We give a stateless deterministic variant — [GPV08] give a *stateful* randomized variant, and the stateless randomized protocol is insecure.

Construction 5.6. Let $\mathsf{PSF} = (\mathsf{Gen}_{psf}, \mathsf{Sample}, F, F^{-1})$ be a pre-image sampleable function, PRF be a pseudorandom function, and H a hash function. Let $\mathcal{S} = (\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ where

$$\mathsf{Gen}(\lambda) : (\mathsf{ik}, \mathsf{fk}) \xleftarrow{R} \mathsf{Gen}_{psf}(\lambda),\ k \xleftarrow{R} \{0,1\}^{\lambda}$$
$$\text{output } \mathsf{sk} = (\mathsf{ik}, k), \mathsf{vk} = \mathsf{fk}$$

$$\mathsf{Sig}((\mathsf{ik}, k), m) : r \leftarrow \mathsf{PRF}(k, m)\ h \leftarrow H(m),\ \text{output } \sigma = F^{-1}(\mathsf{ik}, h; r)$$

$$\mathsf{Ver}(\mathsf{fk}, m, \sigma) : h \leftarrow H(m),\ h' \leftarrow F(\mathsf{fk}, \sigma),\ \text{accept if and only if } h = h'$$

Correctness of the scheme follows from the correctness of the PSF. For security, we have the following theorem:

Theorem 5.7. *If $(\mathsf{Gen}, F, F^{-1})$ is a PSF with pre-image min-entropy at least 1, and PRF is a secure PRF, then $(\mathsf{Gen}, \mathsf{Sig}^H, \mathsf{Ver}^H)$ in Construction 5.6 is strongly EUF-CMA secure in the quantum random oracle model.*

Proof. Let $\mathcal{A}$ be an EUF-CMA adversary for $(\mathsf{Gen}, \mathsf{Sig}^H, \mathsf{Ver}^H)$ with non-negligible advantage ϵ. First, notice that we can replace the PRF PRF with a random oracle R, and implement the signing oracle as $m \longmapsto F^{-1}(\mathsf{ik}, H(m); R(m))$. The security of PRF shows that $\mathcal{A}$ still has non-negligible advantage ϵ'.

We use $\mathcal{A}$ to build an adversary $\mathcal{B}$ for the collision resistance of $(\mathsf{Gen}, F, F^{-1})$. Let q_H be the number of random oracle queries made by $\mathcal{A}$, and let q_S be the number of signing queries. We will assume $\mathcal{B}$ as access to a random oracle $O : \mathcal{M} \to \mathcal{X}$ to which it can make $2q_H$ quantum queries and $q_S + 1$ classical queries. Using Lemma 4.1 and the techniques of Section 4.1, we can simulate O using a $(4q_H + q_S + 1)$-wise independent function. $\mathcal{B}$ works as follows:

- $\mathcal{B}$, on input fk, sends fk to $\mathcal{A}$.

- $\mathcal{B}$ responds to signing queries on messages m with $O(m)$. In other words, it treats O as the signing oracle.
- $\mathcal{B}$ implements the random oracle $H(m)$ as $F(\mathsf{fk}, H(m))$. That is, when $\mathcal{A}$ queries H on the superposition

$$\sum_{m\in\mathcal{M},y\in\mathcal{Y},z\in\mathcal{Z}} \alpha_{m,y,z}|m,y,z\rangle$$

  $\mathcal{B}$ responds with

$$\sum_{m\in\mathcal{M},y\in\mathcal{Y},z\in\mathcal{Z}} \alpha_{m,y,z}|m,y\oplus F(\mathsf{fk},H(m)),z\rangle$$

  $\mathcal{B}$ can compute the response state using 2 quantum queries to H: first it queries H, XORing the response into a new register initialized to 0. Then is compute $F(\mathsf{fk},\cdot)$ in superposition over the new register, XORing the response into the y register. Finally, it uncomputes the new register my making an additional query to H.
- When $\mathcal{A}$ outputs m^*, σ^*, $\mathcal{B}$ outputs $(H(m^*), \sigma^*)$ as a collision for F.

We now analyze the success probability of $\mathcal{B}$. Let (m_i, σ_i) be the message/signature pairs $\mathcal{A}$ obtains through its signing queries (so $\sigma_i = O(m_i)$). First, notice that the oracle $H(m) = F(\mathsf{fk}, O(m))$ seen by $\mathcal{A}$ is a truly random oracle. Thus, H has the correct distribution. Next, since signatures are computed as $\sigma = O(m)$, $F(\mathsf{fk},\sigma) = H(m)$, so the signatures computed by $\mathcal{B}$ are valid. Moreover, the signatures are uniform, conditioned on $F(\mathsf{fk},\sigma) = H(m)$, so the distribution on signatures is correct. Therefore, te view of $\mathcal{A}$ as a subroutine of $\mathcal{B}$ is identical to that of $\mathcal{A}$ in the EUF-CMA experiment. This means that $F(\mathsf{fk},\sigma^*) = H(m^*) = F(\mathsf{fk}, O(m^*))$ and $(m^*,\sigma^*) \notin \{(m_i, O(m_i))\}_{i\in[q_S]}$ with probability ϵ. Thus, we have a collision as long as $O(m^*) \neq \sigma^*$. Notice that since $(m^*,\sigma^*) \notin \{(m_i, O(m_i))\}_{i\in[q_S]}$, $\mathcal{A}$ never made a signing query on m_i. This means that the only information $\mathcal{A}$ has about $O(m^*)$ is through the value $H(m^*) = F(\mathsf{fk}, O(m^*))$. Therefore from $\mathcal{A}$'s perspective, $O(m^*)$ is random conditioned on $F(\mathsf{fk}, O(m^*)) = H(m^*) = F(\mathsf{fk},\sigma^*)$. Since F has pre-image min-entropy $H_\infty = 1$, we therefore have that the probability $O(m^*) = \sigma^*$ is $1/2$. Therefore, the probability that $\mathcal{B}$ outputs a collision is $\epsilon/2$, which is non-negligible.

It remains to show that $\mathcal{B}$ did not make too many queries to O. Indeed, $\mathcal{B}$ makes 2 quantum queries to O for every random oracle query $\mathcal{A}$ makes to H, giving $2q_H$ in total. $\mathcal{B}$ additionally makes a classical query for every signing query, as well as one final query at the end. Therefore, $\mathcal{B}$ makes $2q_H$ quantum queries and $q_S + 1$ classical queries, as desired.

□

Remark 5.8. The PSF constructed by Gentry, Peikert, and Vaikuntanathan [GPV08] actually departs somewhat from the ideal notion described in Section 2.2. Namely, there is a distribution D such that inputs to the PSF are sampled from D, and one-wayness is defined with respect to inputs chosen from this distribution. Moreover, correctness is slightly relaxed - the pre-image sampler only

outputs a distribution that is statistically-close to being the correct distribution. It is straightforward to adapt the proof of Theorem 5.7 to handle non-uniform D. The fact that pre-image sampling is only statistically correct is more challenging to handle. The result is that $\mathcal{B}$'s answers to $\mathcal{A}$'s queries are only statistically close to the correct distribution, rather than being identically distributed (though all signatures are still valid). We need to argue that $\mathcal{A}$ cannot tell the difference between the two distributions. Fortunately, we have already proved Theorem 4.18, which shows that this is the case. Therefore, using Theorem 4.18, it is straightforward to adapt the proof of Theorem 5.7 to handle non-ideal PSFs.

In Section 6.3.3, we show that the GPV signatures are actually secure in an even stronger model, where the adversary is allowed quantum signing queries.

Notice that the proof of Theorem 5.19 crucially relies on the PSF having high pre-image min-entropy, and therefore is insufficient for proving that the Full Domain Hash signature scheme from trapdoor permutations is secure. In Section 5.2.3 we show that the Full Domain Hash signature scheme using trapdoor permutations is secure using more advanced techniques.

5.2.2 Signatures from Claw-Free Permutations

We show that two classical signature constructions using *claw-free* permutations are secure in the quantum random oracle model. A claw-free permutation is a pair $(\mathsf{Gen}_0, F_0, F_0^{-1}), (\mathsf{Gen}_1, F_1, F_1^{-1})$ with the following properties:

- $\mathsf{Gen}_0 = \mathsf{Gen}_1$. Define $\mathsf{Gen} = \mathsf{Gen}_0 = \mathsf{Gen}_1$.
- F_0, F_1 have the same domain (and range).
- Given only fk, no efficient quantum adversary can find a claw for F_0, F_1, that is, a pair x_0, x_1 such that $F(\mathsf{fk}, x_0) = F(\mathsf{fk}, x_1)$.

We currently do not know of any trapdoor permutations secure against quantum adversaries, so the results of this section are mostly of theoretical interest.

As noted by GPV [GPV08], claw-free permutations can be seen as a special case of pre-image sampleable functions. That is, define $F(\mathsf{fk}, (b, x)) = F_b(\mathsf{fk}, x)$. Then any claw (x_0, x_1) for F_0, F_1 corresponds to a collision $((0, x_0), (1, x_1))$ for F. Moreover, F is exactly two-to-one, so the pre-image min-entropy of F is exactly 1. Therefore, plugging into Construction 5.6, we obtain the following signature scheme:

Construction 5.9. Let $(\mathsf{Gen}, F_0, F_0^{-1}), (\mathsf{Gen}, F_1, F_1^{-1})$ be a pair of claw-free permutations with domain $\mathcal{X}$. Let $\mathcal{M}$ be the desired message space, and let $H : \mathcal{M} \to \mathcal{X}$ be a random oracle. We

define the signature scheme $(\mathsf{Gen}, \mathsf{Sig}^H, \mathsf{Ver}^H)$ where:

$$\mathsf{Sig}^H(\mathsf{ik}, m) : y \leftarrow H(m),\ b \xleftarrow{R} \{0,1\},\ \sigma \leftarrow F_b^{-1}(\mathsf{ik}, y)$$
$$\text{Output } (b, \sigma)$$

$$\mathsf{Ver}^H(\mathsf{fk}, m, (b, \sigma)) : y \leftarrow H(m)$$
$$\text{Accept if and only if } y = F_b(\mathsf{fk}, \sigma)$$

Security follows immediately from Theorem 5.19:

Corollary 5.10. *If* $(\mathsf{Gen}, F_0, F_0^{-1}), (\mathsf{Gen}, F_1, F_1^{-1})$ *is a pair of claw-free permutations, then* $(\mathsf{Gen}, \mathsf{Sig}^H, \mathsf{Ver}^H)$ *in Construction 5.9 is strongly EUF-CMA secure in the quantum random oracle model.*

Katz-Wang signatures. The following signature scheme, due to Katz and Wang [KW03], can be seen as a variant of Construction 5.9. An interesting feature of their scheme is that in the actual protocol, only one of the permutations, say F_0, is used. F_1 only comes up in the analysis.

Construction 5.11. Let $(\mathsf{Gen}, F_0, F_0^{-1}), (\mathsf{Gen}, F_1, F_1^{-1})$ be a pair of claw-free permutations with domain $\mathcal{X}$. Let $\mathcal{M}$ be the desired message space, and let $H : \mathcal{M} \times \{0,1\} \to \mathcal{X}$ be a random oracle. We define the signature scheme $(\mathsf{Gen}, \mathsf{Sig}^H, \mathsf{Ver}^H)$ where:

$$\mathsf{Sig}^H(\mathsf{ik}, m) : b \xleftarrow{R} \{0,1\},\ y \leftarrow H(m, b),\ \sigma \leftarrow F_0^{-1}(\mathsf{ik}, y)$$
$$\text{Output } \sigma$$

$$\mathsf{Ver}^H(\mathsf{fk}, m, \sigma) : y_0 \leftarrow H(m, 0),\ y_1 \leftarrow H(m, 1)$$
$$\text{Accept if and only if } F_0(\mathsf{fk}, \sigma) \in \{y_0, y_1\}$$

Theorem 5.12. *If* $(\mathsf{Gen}, F_0, F_0^{-1}), (\mathsf{Gen}, F_1, F_1^{-1})$ *is a pair of claw-free permutations, then* $(\mathsf{Gen}, \mathsf{Sig}^H, \mathsf{Ver}^H)$ *in Construction 5.11 is strongly EUF-CMA secure in the quantum random oracle model.*

The proof is extremely similar to the proof of Theorem 5.19.

Full-Domain Hash with a Claw-Free pair. Coron [Cor00] shows that the Full-Domain Hash (FDH) signature scheme, when implemented with a permutation that is part of a claw-free pair, admits a tighter security reduction that FDH when implemented with a generic trapdoor permutation. Coron's reduction has a form that permits us to prove security in the quantum random oracle model. Like the Katz-Wang scheme above, the second permutation in the pair is only used for analysis. We therefore describe the protocol in terms of an arbitrary trapdoor permutation, and only bring in the claw-free property in the security statement and proof.

Construction 5.13. Let $(\mathsf{Gen}, F, F^{-1})$ be a trapdoor permutation with domain $\mathcal{X}$. Let $\mathcal{M}$ be the desired message space, and let $H : \mathcal{M}\times \to \mathcal{X}$ be a random oracle. We define the signature scheme $(\mathsf{Gen}, \mathsf{Sig}^H, \mathsf{Ver}^H)$ where:

$$\mathsf{Sig}^H(\mathsf{ik}, m) : \text{Output } \sigma \leftarrow F^{-1}(\mathsf{ik}, H(m))$$

$$\mathsf{Ver}^H(\mathsf{fk}, m, \sigma) : \text{Accept if and only if } F(\mathsf{fk}, \sigma) = H(m)$$

Theorem 5.14. *If $(\mathsf{Gen}, F, F^{-1}) = (\mathsf{Gen}, F_0, F_0^{-1})$ is part of a claw-free pair of trapdoor permutations $(\mathsf{Gen}, F_0, F_0^{-1}), (\mathsf{Gen}, F_1, F_1^{-1})$, then $(\mathsf{Gen}, \mathsf{Sig}^H, \mathsf{Ver}^H)$ in Construction 5.13 is strongly EUF-CMA secure in the quantum random oracle model.*

Proof. Let $\mathcal{A}$ be an adversary with advantage ϵ. We prove security through a sequence of hybrid games.

Game 0. This is the standard attack game for $(\mathsf{Gen}, \mathsf{Sig}^H, \mathsf{Ver}^H)$, where $\mathcal{A}$ has advantage ϵ.

Game 1. Here, we modify the random oracle H and signing oracle are generated. Let $O : \mathcal{M} \to \mathcal{X} \times [p]$ for some integer p to be chosen later. Then we implement the oracles as follows:

- H, on input m, runs $(a, b) \leftarrow O(m)$. If $b = 1$, it returns $F_1(\mathsf{fk}, a)$. If $b > 1$, it returns $F_0(\mathsf{fk}, a)$.
- To sign a message m, compute $(a, b) \leftarrow O(m)$. If $b = 1$, return $F_0^{-1}(\mathsf{ik}, F_1(\mathsf{fk}, a))$. If $b > 1$, return $a = F_0^{-1}(\mathsf{ik}, F_0(\mathsf{fk}, a))$

Notice that the oracle H is a truly random oracle, and the signing oracle correctly computes the pre-image under F_0 of the oracle value. Therefore the view of $\mathcal{A}$ is unchanged, and $\mathcal{A}$ has advantage ϵ in **Game 1**. Notice that, for all m, the b part of the output of O is information-theoretically hidden from $\mathcal{A}$.

Game 2. Now change the game so that the challenger aborts and reports failure if $\mathcal{A}$ ever asks for a signature on m such that $(a, b) \leftarrow O(m)$ with $b = 1$. Also, when $\mathcal{A}$ produces a forgery (m^*, σ^*), the challenger computes $(a^*, b^*) \leftarrow O(m^*)$, and aborts if $b^* \neq 1$.

The view of $\mathcal{A}$ is independent of b, b^*, so these abort conditions are independent of $\mathcal{A}$'s success. The probability of no abort is $(1-1/p)^{q_S}/p$. Setting $p = q_S$, this probability is at least $(e^{-1}-o(1))/p$. Thus, $\mathcal{A}$ still outputs a valid forgery with no abort with probability at least $\epsilon(e^{-1} - o(1))/p$.

Now notice that in **Game 2**, the challenger never needs to compute F_1^{-1} during a signing query, because it aborts exactly when it would need to compute the inverse. Therefore, the challenger does not need the master secret key.

This allows us to build the following claw-finding adversary $\mathcal{B}$, which we assume has access to a quantum accessible oracle O:

- On input fk, simulate $\mathcal{A}$ on input fk.
- Simulate the oracle H and the signing oracle as in **Game 2**, aborting when the challenger would abort.
- When $\mathcal{A}$ outputs a forgery m^*, σ^*, compute $(a^*, 1) \leftarrow O(m^*)$, and output (σ^*, a^*) as a claw.

With probability $\epsilon(e^{-1} - o(1))/p$, $\mathcal{B}$ will succeed in outputting such a (σ^*, a^*), where $(a^*, 1) \leftarrow O(m^*)$, and $\mathsf{Ver}^H(\mathsf{fk}, m^*, \sigma^*)$ accepts. But verification passing means that $H(m^*) = F_0(\mathsf{fk}, \sigma^*)$. Since $O(m^*) = (a^*, 1)$, we have that $H(m^*) = F_1(\mathsf{fk}, a^*)$. Thus $F_0(\mathsf{fk}, \sigma^*) = F_1(\mathsf{fk}, a^*)$, meaning (σ^*, a^*) is a claw. Thus $\mathcal{B}$ has non-negligible advantage, as desired. □

5.2.3 Signatures from General Trapdoor Permutations

We now demonstrate that the plain Full-Domain Hash signature scheme (Construction 5.13) is secure, given any arbitrary trapdoor permutation. Theorem 5.14 proves security assuming the permutation is part of a claw-free pair, which are not known to exist in the quantum setting. In contrast, this section shows that any trapdoor permutation suffices, such as the obfuscation-based construction of Bitansky, Paneth, and Wichs [BPW15].

Theorem 5.15. *If* $(\mathsf{Gen}, F, F^{-1})$ *is a trapdoor permutation, then* $(\mathsf{Gen}, \mathsf{Sig}^H, \mathsf{Ver}^H)$ *in Construction 5.13 is strongly EUF-CMA secure in the quantum random oracle model.*

Proof. Let $\mathcal{A}$ be a quantum adversary making q_H hash queries, q_S signing queries queries, that breaks $(\mathsf{Gen}, \mathsf{Sig}^H, \mathsf{Ver}^H)$ with advantage ϵ. We prove security through a sequence of hybrid games.

Game 0. This is the standard attack game for $(\mathsf{Gen}, \mathsf{Sig}^H, \mathsf{Ver}^H)$: the challenger generates $(\mathsf{ik}, \mathsf{fk})$ from Gen, and sends fk to the adversary. The adversary can make (classical) signing queries on messages m_i, to which the challenger responds with $\sigma_i = \mathsf{Sig}^H(\mathsf{ik}, m_i) = F^{-1}(\mathsf{ik}, H(m_i))$, and (quantum) hash queries to the random oracle H. $\mathcal{A}$ then produces an message m^*, and forgery σ^*. We will say that $\mathcal{A}$ "wins" if $(m^*, \sigma^*) \notin \{(m_i, \sigma_i)\}_{i \in [q_S]}$ and $H(m^*) = F(\mathsf{fk}, \sigma^*)$ (which means (m^*, σ^*) is a valid message/signature pair). By definition, this happens with probability ϵ.

Game 1. Here, we modify the adversary's oracles as follows. The signing oracle is implemented by a random oracle $O : \mathcal{M} \to \mathcal{X}$, and H is implemented as $H(m) = F(\mathsf{fk}, O(m))$. Note that the oracle H is a truly random oracle, and given H, the outputs of O are the correct pre-images of H under F. Therefore, the distribution on oracles is identical to **Game 1**, and so $\mathcal{A}$ still wins with probability ϵ.

Game 2. In this game, we modify the random oracle O to be a small-range function. Let $r = 2\ell(q_H + q_S + 1)/\epsilon'$, where $\ell(\cdot)$ is the polynomial from Corollary 4.15. At the beginning of the game, choose r random values $x_i \xleftarrow{R} \mathcal{X}$. Also choose a random oracle $P : \mathcal{M} \to [r]$. Then let O be the function $O(m) = x_{P(m)}$. O is then distributed as a small-range distribution on r uniformly random samples. Notice that only $q_H + q + S + 1$ queries are made to O (q_H direct queries, q_S queries to answer $\mathcal{A}$'s signing queries, and one additional query to check the forgery). By Corollary 4.15, the advantage of $\mathcal{A}$ only decreases by at most $\ell/r = \epsilon'/2$. Therefore, the adversary wins with probability at least $\epsilon'/2$.

Set $y_i = F(\mathsf{fk}, x_i)$. Notice that we can implement the oracle H as $H(x) = y_{P(x)}$.

Game 3. Here, we modify **Game 2** as follows. At the beginning of the game, a random index $i^* \in [r]$ is chosen. Then, for every signing query in message m_j, if $P(m_j) = i^*$, the game declares $\mathcal{A}$ loses, and aborts. When $\mathcal{A}$ produces (m^*, σ^*), if $P(m^*) \neq i^*$, then the game declares $\mathcal{A}$ loses, and aborts. Notice that the probability $P(m_j) = i^*$ is $1/r$, and the probability $P(m^*) \neq i^*$ is $1 - 1/r$. Therefore, the probability of no aborts is $(1 - 1/r)^{q_S}(1/r) \geq 1/r - q_S/r^2$. The abort condition is independent of $\mathcal{A}$ success probability in **Game 1**. Therefore, $\mathcal{A}$ still wins with probability $(\epsilon'/2)(1/r - q_S/r^2)$. Since $r > 2q_S$, $\mathcal{A}$ wins with probability at least $(\epsilon'/4r) = (\epsilon')^2/4\ell$ where $\ell = \ell(q_H + q_S + 1)$.

Notice that the value x_{i^*} is never needed in **Game 3**. This observation allows to build an adversary $\mathcal{B}$ for $(\mathsf{Gen}, F, F^{-1})$. $\mathcal{B}$ works as follows:

- On input a public fk, y, $\mathcal{B}$ chooses a random $i^* \in [r]$, and sets $y_{i^*} = y$. For $i \neq i^*$, $\mathcal{B}$ chooses random $x_i \in \mathcal{X}$ and computes $y_i = F(\mathsf{fk}, x_i)$. Then $\mathcal{B}$ simulates $\mathcal{A}$ on input fk.
- $\mathcal{B}$ implements random oracle as $H(m) = y_{P(m)}$ for a random oracle $P : \mathcal{M} \to [r]$.
- When $\mathcal{A}$ makes a signing query on message m, $\mathcal{B}$ checks that $P(m) \neq i^*$. If the check fails, $\mathcal{B}$ aborts. If the check passes, $\mathcal{B}$ responds with $x_{P(m)}$.
- When $\mathcal{A}$ produces a forgery pair (m^*, σ^*), $\mathcal{B}$ checks that $P(m^*) = i^*$ and $F(\mathsf{fk}, \sigma^*) = H(m^*)$. If the check fails, $\mathcal{B}$ aborts. If the check passes, $\mathcal{B}$ outputs σ^*. Notice that if the checks pass, then $F(\mathsf{fk}, \sigma^*) = H(m^*) = y_{i^*} = y$. Therefore, σ^* is the correct pre-image and $\mathcal{B}$ wins.

Notice that $\mathcal{B}$ perfectly simulates the view of $\mathcal{A}$ in **Game 4**. Therefore, $\mathcal{B}$ wins with the same probability as $\mathcal{A}$ in **Game 4**, namely $(\epsilon')^2/4\ell$, which is non-negligible. □

5.3 Encryption in the QROM

We now show how to build encryption in the quantum random oracle model.

Construction 5.16. Let $(\mathsf{Gen}_{PSF}, F, F^{-1})$ be an injective trapdoor function. Let $(\mathsf{Enc}_S, \mathsf{Dec}_S)$ be a symmetric-key encryption algorithm. Define $(\mathsf{Gen}, \mathsf{Enc}^H, \mathsf{Dec}^H)$ as the following public key encryption algorithm:

$$\mathsf{Gen}() : (\mathsf{ik}, \mathsf{fk}) \xleftarrow{R} \mathsf{Gen}_{PSF}(), \text{ Output } (\mathsf{dk} = \mathsf{ik}, \mathsf{ek} = \mathsf{fk})$$

$$\mathsf{Enc}^H(\mathsf{fk}, m) : x \xleftarrow{R} \mathcal{X},\ y \leftarrow F(\mathsf{fk}, x),\ k \leftarrow H(x),\ c \xleftarrow{R} \mathsf{Enc}_S(k, m), \text{ Output } (y, c)$$

$$\mathsf{Dec}^H(\mathsf{ik}, (y, c)) : x \leftarrow F^{-1}(\mathsf{ik}, y),\ k \leftarrow H(x),\ m \leftarrow \mathsf{Dec}_S(k, c), \text{ Output } m$$

Theorem 5.17. *If $(\mathsf{Gen}, F, F^{-1})$ is an injective trapdoor function, and $(\mathsf{Enc}_S, \mathsf{Dec}_S)$ is a one-ciphertext CCA-secure (resp. semantically secure) symmetric-key encryption scheme, then $(\mathsf{Gen}, \mathsf{Enc}^H, \mathsf{Dec}^H)$ is a CCA-secure (resp. CPA-secure) public key encryption scheme in the quantum random oracle model.*

We note that $\mathsf{Enc}_S(k, m) = k \oplus m, \mathsf{Dec}_S(k, c) = k \oplus c$ is a semantically secure encryption scheme

Proof. We prove the CCA case, the CPA case being similar. Let $\mathcal{A}$ be a CCA adversary for $(\mathsf{Gen}, \mathsf{Enc}^H, \mathsf{Dec}^H)$ with non-negligible advantage ϵ. We prove security through a sequence of hybrid games:

Game 0. This is the standard attack game. The challenger runs $(\mathsf{ik}, \mathsf{fk}) \xleftarrow{R} \mathsf{Gen}()$, and gives fk to $\mathcal{A}$. The challenger also instantiates a random oracle H, which $\mathcal{A}$ can make queries to. When $\mathcal{A}$ makes a challenge query on messages (m_0, m_1), the challenger chooses a random $b \in \{0, 1\}$, $x^* \xleftarrow{R} \mathcal{X}$, computes $y^* \leftarrow F(\mathsf{fk})$, $k^* \leftarrow H(x^*)$, and $c^* \xleftarrow{R} \mathsf{Enc}(k^*, m_b)$, and responds with (y^*, c^*). When $\mathcal{A}$ makes a decryption query on ciphertext (y, c), if $(y, c) \neq (y^*, c^*)$ the challenger computes $x \leftarrow F^{-1}(\mathsf{ik}, y)$, $k \leftarrow H(x)$, and $m \leftarrow \mathsf{Dec}_S(k, c)$, and responds with m. If $(y, c) = (y^*, c^*)$, the challenger responds with $\perp$. By definition, $\mathcal{A}$ has advantage ϵ, meaning it outputs $b' = b$ with probability $1/2 + \epsilon$.

Game 1. Here the challenger implements the oracle H as $O(F(\mathsf{pk}, \cdot))$ for a random oracle $O : \mathcal{Y} \to \mathcal{K}$. Since F is injective, H is still a random oracle, and so the view of $\mathcal{A}$ is unchanged. Therefore, $\mathcal{A}$ still has advantage ϵ.

Notice that in **Game 1**, we can now answer decryption queries on (y, c) by computing $k = O(y)$ and decrypting c using k. Thus we do not need ik to simulate the view of $\mathcal{A}$. Notice that we can also determine $x^*, b, y^*, k^* = O(x^*)$ are the beginning of the experiment.

Let δ_i be the probability of obtaining x^* in the following process: when $\mathcal{A}$ makes the ith random oracle query in **Game 1**, halt, measure the query input, and output the result. Let $\delta = \sum i \in [q_H]\delta_i$ be the sum of the probabilities. We refer to δ as the total query magnitude of x^*.

We claim that δ is negligible. Indeed, we can construct an adversary $\mathcal{B}$ for $(\mathsf{Gen}, F, F^{-1})$ with advantage δ: on input (fk, y), $\mathcal{B}$ simulates $\mathcal{A}$ in **Game 1** setting $y^* = y$ (which it can do since simulating **Game 1** does not require ik). Then, at a randomly chosen oracle query $i \in [q_H]$, $\mathcal{B}$ halts the execution of $\mathcal{A}$, measures the input register for the query, and outputs the resulting string x. The probability $x = x^* = F^{-1}(\mathsf{ik}, y^*)$ is exactly δ. The security of $(\mathsf{Gen}, F, F^{-1})$ shows that δ is negligible.

Game 2. Here we change **Game 1** so that k^* is chosen at random, independent of $H(x^*) = O(y^*)$. Alternatively, think of this change as changing the value of $H(y^*)$ to a random value independent of $k^* = O(y^*)$. For decryption queries of the form (y^*, c), decrypt c using key k^* instead of $H(x^*)$. Queries of the form (y, c) for $y \neq y^*$ are still decrypted using key $O(y)$.

Since the total query magnitude of x^*, the only point where we changed the output of H, is negligible, it is known that the advantage of $\mathcal{A}$ is only changed by a negligible amount ([BBBV97] Theorems 3.1 and 3.3). Thus $\mathcal{A}$ still has non-negligible advantage in **Game 2**.

We now build an adversary $\mathcal{C}$ for $(\mathsf{Enc}_S, \mathsf{Dec}_S)$. $\mathcal{B}$ works as follows:

- Run $(\mathsf{ik}, \mathsf{fk}) \leftarrow \mathsf{Gen}()$, and initialize a random oracle $O : \mathcal{Y} \to \mathcal{X}$. Choose random $x^* \xleftarrow{R} \mathcal{X}$ and set $y^* = F(\mathsf{pk}, y^*)$. Simulate $\mathcal{A}$ on input fk.
- Implement the (quantum accessible) random oracle H seen by $\mathcal{A}$ as $O(F(\mathsf{pk}, \cdot))$.
- When $\mathcal{A}$ makes a challenge query on message pair (m_0, m_1), $\mathcal{C}$ sends (m_0, m_1) as its own challenge messages for Enc_S. In response, it receives c^*. $\mathcal{C}$ sends (y^*, c^*) to $\mathcal{A}$.
- When $\mathcal{A}$ makes a decryption query in (y, c), there are two cases. If $y = y^*$, then to be a valid decryption query we must have $c \neq c^*$, and therefore $\mathcal{C}$ can make a decryption query to its challenger, obtaining m, which it sends to $\mathcal{A}$. If $y \neq y^*$, then compute $k = O(y)$, $m \leftarrow \mathsf{Dec}(k, c)$, and return m to $\mathcal{A}$.
- Finally, when $\mathcal{A}$ outputs b', $\mathcal{C}$ outputs b'.

$\mathcal{C}$ perfectly simulates the view of $\mathcal{A}$ in **Game 2** with b being the same b as $\mathcal{C}$'s challenger. Therefore $b' = b$ with probability $1/2 + \epsilon - \mathsf{negl}$, and so $\mathcal{C}$ has non-negligible advantage in breaking $(\mathsf{Enc}_S, \mathsf{Dec}_S)$. □

5.4 Identity-based Encryption in the QROM

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

More specifically, let $(\mathsf{Gen}_{PSF}, F, F^{-1})$ be a pre-image sampleable function (PSF) with domain $\mathcal{X}$. Suppose Gen_E works as follows: generate $(\mathsf{ik}, \mathsf{fk}) \leftarrow \mathsf{Gen}_{PSF}$. Then, sample $\mathsf{dk} \xleftarrow{R} \mathcal{X}$, and compute $\mathsf{ek} = f(\mathsf{mpk}, \mathsf{sk})$. Output $(\mathsf{dk}, (\mathsf{ek}, \mathsf{fk}))$.

Gentry, Peikert, and Vaikuntanathan 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 following IBE scheme:

Construction 5.18. Let $(\mathsf{Gen}_{PSF}, F, F^{-1})$ be a PSF, and let $(\mathsf{Gen}_E, \mathsf{Enc}_E, \mathsf{Dec}_E)$ be an encryption scheme with Gen_E as above. Let $H : \mathcal{ID} \to \mathcal{Y}$ be a random oracle. Let PRF be a pseudorandom function with key space $\mathcal{K}$. Define the IBE scheme $(\mathsf{Gen}, \mathsf{KeyGen}^H, \mathsf{Enc}^H, \mathsf{Dec})$ where:

$$\mathsf{Gen}() : (\mathsf{ik}, \mathsf{fk}) \xleftarrow{R} \mathsf{Gen}_{PSF}(),\ k \xleftarrow{R} \mathcal{K},\ \text{Output } (\mathsf{msk} = (\mathsf{ik}, k), \mathsf{mek} = \mathsf{fk})$$

$$\mathsf{KeyGen}^H((\mathsf{ik}, k), \mathsf{id}) : y \leftarrow H(\mathsf{id}),\ r \leftarrow \mathsf{PRF}(k, r),\ \mathsf{dk}_{\mathsf{id}} \leftarrow F^{-1}(\mathsf{ik}, y; r),\ \text{Output } \mathsf{dk}_{\mathsf{id}}$$

$$\mathsf{Enc}^H(\mathsf{fk}, \mathsf{id}, m) : y \leftarrow H(\mathsf{id}),\ \text{Output } \mathsf{Enc}_E(y, m)$$

$$\mathsf{Dec}^H(\mathsf{dk}_{\mathsf{id}}, c) : \mathsf{Dec}_E(\mathsf{dk}_{\mathsf{id}}, c)$$

Theorem 5.19. *Let* $(\mathsf{Gen}_{PSF}, F, F^{-1})$ *and* $(\mathsf{Gen}_E, \mathsf{Enc}_E, \mathsf{Dec}_E)$ *be as above, and suppose that* $(\mathsf{Gen}_E, \mathsf{Enc}_E, \mathsf{Dec}_E)$ *is quantum IND-CPA-secure. Then* $(\mathsf{Gen}, \mathsf{KeyGen}^H, \mathsf{Enc}^H, \mathsf{Dec})$ *in Construction 5.18 is IND-ID-CPA-secure in the quantum random oracle model.*

Proof. Let $\mathcal{A}$ be a quantum adversary making q_H hash queries, q_E keygen queries queries, that breaks $(\mathsf{Gen}, \mathsf{KeyGen}^H, \mathsf{Enc}^H, \mathsf{Dec})$ with advantage ϵ. We prove security through a sequence of hybrid games.

Game 0. This is the standard attack game for $(\mathsf{Gen}, \mathsf{KeyGen}^H, \mathsf{Enc}^H, \mathsf{Dec})$: the challenger generates $((\mathsf{ik}, k), \mathsf{fk})$ from Gen, and sends fk to the adversary. The adversary can make (classical) keygen queries on identities id_i, to which the challenger responds with $\mathsf{KeyGen}^H(\mathsf{msk}, \mathsf{id}_i)$, and (quantum) hash queries to the random oracle H. $\mathcal{A}$ then produces an identity id^*, along with messages m_0 and m_1. The challenger chooses a random bit b, and responds with $\mathsf{Enc}^H(\mathsf{mek}, \mathsf{id}^*, m_b)$. $\mathcal{A}$ is allowed to make more keygen and hash queries, and produces a bit b'. We will say that $\mathcal{A}$ "wins" if $b' = b$ and for all i, $\mathsf{id}_i \neq \mathsf{id}^*$. By definition, this happens with probability $\frac{1}{2} + \epsilon$.

Assume without loss of generality that $\mathcal{A}$ never asks for the secret key corresponding to id^*. We can make this assumption because, in the event that $\mathcal{A}$ asks for a secret key for id^*, we abort the game and the adversary has no advantage in this case. We also assume without loss of generality that the adversary never performs an keygen query on the same identity twice: since the key generation procedure is deterministic, $\mathcal{A}$ can simulate subsequent queries on the same identity for itself.

Game 1. In this game, we modify keygen queries to be generated with fresh randomness. The security of PRF shows that the adversary still wins with probability $1/2 + \epsilon'$ where $\epsilon' = \epsilon - \mathsf{negl}$.

Game 2. Here, we modify the adversary's oracles as follows. The keygen oracle is implemented by a random oracle $O : \mathcal{ID} \to \mathcal{X}$, and H is implemented as $H(\mathsf{id}) = F(\mathsf{fk}, O(\mathsf{id}))$. Note that the oracle H is a truly random oracle, and given H, the outputs of O are random pre-images of H. Therefore, the distribution on oracles is identical to **Game 1**, and so $\mathcal{A}$ still wins with probability $1/2 + \epsilon$.

Game 3. In this game, we modify the random oracle O to be a small-range function. Let $r = 2\ell(q_H + q_E + 1)/\epsilon'$, where $\ell(\cdot)$ is the polynomial from Corollary 4.15. At the beginning of the game, choose r random values $x_i \xleftarrow{R} \mathcal{X}$. Also choose a random oracle $P : \mathcal{ID} \to [r]$. Then let O be the function $O(\mathsf{id}) = x_{P(\mathsf{id})}$. O is then distributed as a small-range distribution on r uniformly random samples. Notice that only $q_H + q + E + 1$ queries are made to O (q_H direct queries, q_E queries to answer $\mathcal{A}$'s keygen queries, and one additional query for the challenge query). By Corollary 4.15, the advantage of $\mathcal{A}$ only decreases by at most $\ell/r = \epsilon'/2$. Therefore, the adversary wins with probability at least $\frac{1}{2} + \epsilon'/2$.

Set $y_i = F(\mathsf{fk}, x_i)$. Notice that we can implement the oracle H as $H(\mathsf{id}) = y_{P(\mathsf{id})}$.

Game 4. Here, we modify **Game 3** as follows. At the beginning of the game, a random index $i^* \in [r]$ is chosen. Then, for every keygen query in identity id_j, if $P(\mathsf{id}_j) = i^*$, the game declares $\mathcal{A}$ wins with probability $1/2$ and $\mathcal{A}$ loses with probability $1/2$, and then aborts. For the challenge query on id^*, if $P(\mathsf{id}^*) \neq i^*$, then similarly the game declares $\mathcal{A}$ wins/loses with probability $1/2$, and then aborts. Notice that the probability $P(\mathsf{id}_j) = i^*$ is $1/r$, and the probability $P(\mathsf{id}^*) \neq i^*$ is $1 - 1/r$. Therefore, the probability of no aborts is $(1 - 1/r)^{q_E}(1/r) \geq 1/r - q_E/r^2$. The abort condition is independent of $\mathcal{A}$ success probability in **Game 1**. Therefore, $\mathcal{A}$ still wins with probability $\frac{1}{2} + (\epsilon'/2)(1/r - q_E/r^2)$. Since $r > 2q_E$, $\mathcal{A}$ wins with probability at least $\frac{1}{2} + (\epsilon'/4r) = \frac{1}{2} + (\epsilon')^2/4\ell$ where $\ell = \ell(q_H + q_E + 1)$.

Notice that the value x_{i^*} is never needed in **Game 4**. This observation allows to build an adversary $\mathcal{B}$ for $(\mathsf{Gen}_E, \mathsf{Enc}_E, \mathsf{Dec}_E)$. $\mathcal{B}$ works as follows:

- On input a public key $(\mathsf{ek}, \mathsf{fk})$, $\mathcal{B}$ chooses a random $i^* \in [r]$, sets fk to be the master public key and $y_{i^*} = \mathsf{ek}$. For $i \neq i^*$, $\mathcal{B}$ chooses random $x_i \in \mathcal{X}$ and computes $y_i = F(\mathsf{fk}, x_i)$. Then $\mathcal{B}$ simulates $\mathcal{A}$ on input fk.
- $\mathcal{B}$ implements random oracle as $H(\mathsf{id}) = y_{P(\mathsf{id})}$.
- When $\mathcal{A}$ makes an extraction query on identity id, $\mathcal{B}$ checks that $P(\mathsf{id}) \neq i^*$. If the check fails, $\mathcal{B}$ outputs a random bit and aborts. If the check passes, $\mathcal{B}$ responds with $x_{P(\mathsf{id})}$.

- When $\mathcal{A}$ makes the challenge query on identity id^* and messages m_0^*, m_1^*, $\mathcal{B}$ checks that $P(\mathsf{id}) = i^*$. If the check fails, $\mathcal{B}$ outputs a random bit and aborts. If the check passes, $\mathcal{B}$ makes a challenge query to the CPA challenger on m_0^*, m_1^*. Upon receiving c^* from the challenger, $\mathcal{B}$ sends c^* to $\mathcal{A}$.
- When $\mathcal{A}$ outputs a bit b', $\mathcal{B}$ outputs b'.

Notice that $\mathcal{B}$ perfectly simulates the view of $\mathcal{A}$ in **Game 4**. Therefore, $b' = b$ with probability $\frac{1}{2} + (\epsilon')^2/4\ell$. Thus, $\mathcal{B}$ has advantage $(\epsilon')^2/4\ell$, which is non-negligible, as desired. □

Remark 5.20. Similar to Construction 5.6, Gentry, Peikert, and Vaikuntanathan [GPV08] use PSFs that are non-ideal to instantiate Construction 5.18. As in the proof of security for Construction 5.6 (namely, Theorem 5.7), it is straightforward to adapt the security proof proof in Theorem 5.19 above to non-ideal PSFs by using Theorem 4.18.

5.4.1 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) schemes of Agrawal et al. [ABB10b] and Cash et al. [CHKP10]. Unfortunately, describing these schemes completely and giving a full security proof would require a somewhat detailed explanation of lattices, the algorithms involved, and the computational assumptions being made. Instead, we give a high-level overview of the proof technique, and explain how our techniques can be applied to this setting.

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 adversary's choice but not under its control.

In [ABB10b], 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. Also 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 $H(\mathsf{id}_i) = R_i$ for all 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 5.19. Basically, we iterate over levels i, and for identities at the ith level, we replace H on those identities with a small-range distribution on r_i samples. Then, choose one of the r_i samples at random, and set the sample to be R_i. 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 extraction query is. Let $\ell = \ell(2q_H + q_E + 1)$ where ℓ is the polynomial from Corollary 4.15. If the iteration i advantage is ϵ_i, then using the same techniques as in Theorem 5.19, we can set r_i so that

$$\epsilon_i \geq \frac{\epsilon_{i-1}^2}{4\ell}$$

In iteration 0, we say the adversary wins if it wins the standard HIBE game and the reduction correctly guessed which level id^* belonged to. Since the reduction guesses the level at random, independent of the adversary's view, we have that $\epsilon_0 = \epsilon/d$, where ϵ is the adversary's advantage in the standard game. This gives us a total advantage after iteration d of at least

$$\frac{(\epsilon/d)^{2^d}}{(4\ell)^{(2^d-1)}} = 4\ell\Big(\frac{\epsilon}{4d\ell}\Big)^{2^d}$$

Notice that the dependence on d is doubly-exponential, whereas in the classical proof 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.

Chapter 6

Fully Quantum Security Notions

In this chapter, we explore new quantum security definitions where the entire security game is a quantum experiment. This models attacks where the adversary can interact with the cryptosystem over a quantum channel. Even in the near future dominated by classical devices, such channels may become relevant as microprocessor feature sizes continue to shrink and quantum affects become more prevalent. Moreover, looking further out, it seems inevitable that quantum computers will be ubiquitous, and such computers may intentionally or unintentionally support quantum interaction channels. Implementing classical systems on a quantum device thus potentially exposes the system to attacks over these quantum channels, which are not modeled by conventional security notions. We therefore give new security definitions that capture these attacks and give new constructions and security analyses for these enhanced security models.

6.1 Quantum Pseudorandom Functions

In this section, we define PRF security against quantum channel attacks, and show how to build and analyze schemes satisfying this security definition. The definition is a natural extension of the classical security definition:

Definition 6.1 (Quantum security for PRFs)**.** A PRF PRF is *quantum-secure* if quantum queries cannot distinguish it from a truly random function. That is, for any efficient quantum adversary $\mathcal{A}$ with *quantum* access to an oracle,

$$\Pr[\mathcal{A}^{|\mathsf{PRF}(k,\cdot)\rangle}() = 1] - \Pr[\mathcal{A}^{|O(\cdot)\rangle}() = 1] < \mathsf{negl}$$

How do we build quantum-secure PRFs? Ideally, classical constructions of PRFs that are based on quantum resistant primitives can be used. For example, one may ask if the construction of Goldreich, Goldwasser, and Micali [GGM86] is quantum-secure if the underlying pseudorandom

generator is secure against quantum adversaries. Unfortunately, as discussed in the introduction, the classical security proof is insufficient for arguing quantum security. In this section, we rectify this situation by giving the first security proof for the GGM construction that works when the adversary is allowed quantum queries. We also show that two other classical constructions [NR95, BPR12] are secure, assuming the quantum hardness of the underlying problems. We accomplish this using the techniques developed in Chapter 4, specifically the small-range distributions of Section 4.5.

Before give our positive results, we first show that quantum security is strictly stronger than standard (non-quantum) security. This demonstrates that quantum security is more difficult to achieve, and therefore that the need for new proof techniques may be inherent.

6.1.1 Separation

Theorem 6.2. *If standard secure PRFs exist, then there are standard secure PRFs that are not QPRFs.*

We will actually prove a slightly stronger theorem. We define a *quantum-gap* PRF as a standard secure PRF that is quantum insecure in a very strong sense. Namely, a single quantum oracle query suffices to distinguish the PRF from a random oracle with only negligible distinguishing error.

Theorem 6.3. *If quantum secure PRFs exist, then so do quantum-gap PRFs*

This theorem will be useful in proving further separation results in Sections 6.3 and 6.4.

Proof. Let PRF be a quantum 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^9 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 $\mathsf{PRF}'((k,p),x) = \mathsf{PRF}(k, x \mod p)$ where:

- $x \mod p$ means the unique integer x' in $[p]$ such that $x - x'$ is a multiple of p[1].
- The key space of PRF' is $\mathcal{K}' = \mathcal{K} \times \mathcal{P}$ where $\mathcal{P}$ is the set of primes between $N - \sqrt{N}$ and N. That is, a key for PRF' is a pair (k,p) where k is a key for PRF, and p is a prime in the range $(N - \sqrt{N}, N]$.
- The domain is $\mathcal{X}' = [N']$ where N' is the smallest power of 2 greater than N^4.

[1]Note that this differs from the usual definition of mod where x' is required to be in $\{0, \ldots, p-1\}$

The following two claims together prove Theorem 6.3:

Claim 6.4. *If* PRF *is standard secure, then so is* PRF'.

Claim 6.5. *If* PRF *is quantum secure, then* PRF' *can be distinguished from random with overwhelming probability using only a single quantum query.*

Thus one of PRF and PRF' is standard secure but not quantum secure, as desired. In particular, if PRF is quantum secure, then PRF' is a quantum-gap PRF. We now prove Claims 6.4 and 6.5.

Proof of Claim 6.4. We prove that if PRF is standard-secure, so is PRF'. The idea is that, 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.

Suppose we have a quantum adversary $\mathcal{A}$ making classical queries that distinguishes PRF' from a random function with non-negligible probability ϵ. That is,

$$\left| \Pr_{k\leftarrow\mathcal{K},p\leftarrow\mathcal{P}}[\mathcal{A}^{\mathsf{PRF}'((k,p),\cdot)}() = 1[] - \Pr_{O\leftarrow\mathcal{Y}^{\mathcal{X}}}[\mathcal{A}^{O}() = 1] \right| = \epsilon$$

This is equivalent to

$$\left| \Pr_{k\leftarrow\mathcal{K},p\leftarrow\mathcal{P}}[\mathcal{A}^{\mathsf{PRF}(k,\cdot \mod p)}() = 1] - \Pr_{O\leftarrow\mathcal{Y}^{\mathcal{X}}}[\mathcal{A}^{O}() = 1] \right| = \epsilon$$

Consider the quantity

$$\left| \Pr_{O\leftarrow\mathcal{Y}^{\mathcal{X}},p\leftarrow\mathcal{P}}[\mathcal{A}^{O(\cdot \mod p)}() = 1] - \Pr_{O\leftarrow\mathcal{Y}^{\mathcal{X}}}[\mathcal{A}^{O}() = 1] \right|$$

The left hand side is the case where O is a random function in $\mathcal{Y}^{\mathcal{X}}$, a is a random prime in $(N-\sqrt{N}, N]$, and we give $\mathcal{A}$ the oracle $O'(x) = O(x \mod p)$. As long as $\mathcal{A}$ never queries its oracle on two points x and x' such that $x \equiv x' \mod p$, this oracle will look random. If $\mathcal{A}$ makes q queries, there are $\binom{q}{2}$ possible differences between query points. Each difference is at most $2N^4$, so for large N it can only be divisible by at most 4 different moduli p. Notice that $|\mathcal{P}| \in \Omega(\sqrt{N}/\log N)$. Each difference thus has a probability at most $4/|\mathcal{P}| \in O(\log N/\sqrt{N})$ of being divisible by p, so the total probability of querying x and x' such that $x \equiv x' \mod p$ is at most $O(q^2 \log N/\sqrt{N})$. Thus this probability, and hence the ability of $\mathcal{A}$ to distinguish O' from a random oracle, is negligible.

A simple hybrid argument then shows that

$$\left| \Pr_{k\leftarrow\mathcal{K},p\leftarrow\mathcal{P}}[\mathcal{A}^{\mathsf{PRF}(k,\cdot \mod p)}() = 1] - \Pr_{O\leftarrow\mathcal{Y}^{\mathcal{X}},p\leftarrow\mathcal{P}}[\mathcal{A}^{O(\cdot \mod p)}() = 1] \right| \geq \epsilon - O(q^2 \log N/\sqrt{N})$$

Define a quantum algorithm $\mathcal{B}$ which distinguishes PRF from a random oracle. $\mathcal{B}$ has an oracle O, chooses a random prime $p \in (N/2, N]$, and simulates $\mathcal{A}$ with the oracle $O'(x) = O(x \mod p)$. When $O(\cdot) = \mathsf{PRF}(k, \cdot)$, we get the left side, and when O is random, we get the right side. Thus,

$$\left| \Pr_{k\leftarrow\mathcal{K}}[\mathcal{B}^{\mathsf{PRF}(k,\cdot)}() = 1] - \Pr_{O\leftarrow\mathcal{Y}^{\mathcal{X}}}[\mathcal{B}^{O}() = 1] \right| \geq \epsilon - O(q^2 \log N/\sqrt{N})$$

Since N is exponential, B breaks the standard security of PRF. □

Proof of Claim 6.5. We now show that if PRF is quantum-secure, then we can distinguish PRF' from random with overwhelming probability using a single quantum query. The idea is that, if we allow quantum queries to PRF', we can use quantum period finding [Sho94, BL95] to find p. With p, 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.

Also unfortunately, the analyses from Shor [Sho94] and Boneh and Lipton [BL95] only recover the period with non-negligible probability with one query, and thus require many queries to achieve overwhelming probability. As we wish to obtain overwhelming probability with a single query, we present a more refined analysis for our setup.

We first consider the case of distinguishing a truly random function from a periodic function that is truly random on its period. That is, distinguishing a truly random oracle $O : [N'] \to \mathcal{Y}$ from an oracle $O(x) = O'(x \mod p)$ where p is a random prime in $\mathcal{P}$ and $O' : [N] \to \mathcal{Y}$ is a random oracle. Our distinguisher is as follows:

1. Initialize quantum registers to the state

$$|\psi_1\rangle = \frac{1}{\sqrt{N'}} \sum_{x\in[N']} |x, 0\rangle$$

2. Now make a quantum PRF query on $|\psi_1\rangle$. The result is the state

$$|\psi_2\rangle = \frac{1}{\sqrt{N'}} \sum_{x\in[N']} |x, O(x)\rangle$$

3. Now measure the second register, obtaining a value y and the state

$$|\psi_y\rangle = \frac{1}{\sqrt{|C_{O,y}|}} \sum_{x \in C_{O,y}} |x\rangle$$

where $C_{O,y}$ is the set of x such that $O(x) = y$. The probability of obtaining state $|\psi_y\rangle$ is $|C_{O,y}|/N'$

4. Now perform the quantum Fourier Transform on $|\psi_y\rangle$. The result is the state

$$|\psi'_y\rangle = \frac{1}{\sqrt{N'|C_{O,y}|}} \sum_{x \in C_{O,y}, z \in [N']} \omega^{xz} |z\rangle$$

where ω is a primitive N'th root of 1.

5. Measure the first register, obtaining an integer $q \in [N']$.

6. Determine the (reduced) fraction d/p' that is closest to q/N' with denominator $p' \leq N$. This can be done using continued fractions. If p' lies in the range $(N - \sqrt{N}, N]$, output 1. Otherwise, output 0.

To analyze the success probability of our distinguisher, first consider the case where O is random. Then since $|\mathcal{Y}| \geq N^9 \geq (N')^2 O(N)$, O is with overwhelming probability injective. This means $C_{O,y}$ is either 1 or 0 for all $y \in \mathcal{Y}$. Therefore, the state $|\psi_y\rangle$ obtained in Step 3 above is equal to $|x\rangle$ for some $x \in [N']$. Then a Fourier transform and measurement gives a random integer in $[N']$. It is straightforward to argue that in this case the probability the algorithm outputs a $p' \in (N - \sqrt{N}, N]$ is $O(1/\sqrt{N})$, which is negligible. Therefore, our algorithm outputs 0 with overwhelming probability in this case.

Next, consider the case where $O(x) = O'(x \mod p)$ for a random oracle $O' : [p] \to \mathcal{Y}$. We wish to show that our attack computes $p' = p$ with overwhelming probability, meaning it almost always outputs 1.

Since $|\mathcal{Y}| \geq N^9 \geq (p)^2 \cdot \Omega(N)$, O is injective on the interval $[p]$ with overwhelming probability. Moreover, $C_{O,y}$ is non-zero only for $y = O'(x)$, $x \in [p]$, and in this case, we can write $C_{O,y} = C_{p,x}$ where $C_{p,x}$ is the set of $x' \in [N']$ such that $x' - x$ is divisible by p. The state $|\psi_y\rangle$ in Step 3 can be written as

$$|\psi_x\rangle = \frac{1}{\sqrt{|C_{p,x}|}} \sum_{x' \in C_{p,x}} |x'\rangle = \frac{1}{\sqrt{|C_{p,x}|}} \sum_{j \in [C_{p,x}]} |x - 1 + jp\rangle$$

The quantum Fourier transform gives us

$$|\psi'_x\rangle = \frac{1}{\sqrt{N'|C_{p,x}|}} \omega^{x-1} \sum_{z \in [N']} \left(\sum_{j \in C_{p,x}} \omega^{jpz} \right) |z\rangle$$

Measuring gives z with probability $\frac{1}{N'C_{p,x}}\left|\sum_{j\in C_{p,x}}\omega^{jz}\right|^2$. Notice that $|C_{p,x}| = \left\lfloor \frac{N'-x}{p}+1 \right\rfloor$ and therefore satisfies $|C_{p,x}| - \lfloor N'/p \rfloor \in \{0,1\}$. It is straightforward to show that the probability of obtaining z is

$$\frac{1}{N'C_{p,x}}\frac{\sin(\frac{\pi pC_{p,x}}{N'}z)^2}{\sin(\frac{\pi p}{N'}z)^2}$$

Now, an output when measuring z gives the right answer p when d/p for some d is the closest rational approximation to z/N' with denominator at most N, and $d \neq 0$. This is happens if

$$\left|\frac{z}{N'} - \frac{d}{p}\right| \leq \frac{1}{2N^2} \leq \frac{1}{2p^2}$$

We bound the total probability mass of z such that this inequality is unsatisfied. That is, z for which z/N' is at least $1/2p^2$ away from d/p. For such z, we can bound their probability as

$$\frac{1}{N'C_{p,x}}\frac{1}{\sin(\pi/2p)^2} \leq \frac{16p^2}{\pi^2 N'C_{p,x}} \leq \frac{16p^3}{\pi^2(N')^2} \leq \frac{16N^3}{\pi^2(N')^2}$$

The total probability of such z is at most N' times this amount, or $\frac{16N^3}{\pi^2 N'}$, which is negligible by our choice of $N' \geq N^4$. It is also possible to show that the total probability mass of z near $0 = 0/p$ is at most $O(1/p)$, which is negligible as well.

Therefore, the total probability of z for which our algorithm fails to output 1 is negligible. Thus, our distinguishes $O(x)$ from $O'(x \mod p)$ with overwhelming advantage. By the quantum security of PRF, $O'(x \mod p)$ is indistinguishable from $\mathsf{PRF}(k, x \mod p)$, and therefore our algorithm successfully distinguishes $\mathsf{PRF}'((k,p),x) = \mathsf{PRF}(k, x \mod p)$ from $O(x)$ with overwhelming probability. □

This completes the proof of Theorem 6.3. □

6.1.2 Construction 1: The GGM Construction

We now prove the quantum security of the Goldriech, Goldwasser, and Micali [GGM86] PRF construction in a new way that makes sense in the quantum setting. Recall that the GGM construction builds a PRF out of any length-doubling pseudorandom generator (PRG). Our approach is as follows. We re-interpret the classical GGM proof as consisting of two stages: in the first, the security of the PRF is proved assuming a seemingly stronger security notion for the underlying PRG. In the second stage, the seemingly stronger PRG security notion is proved to be equivalent to the standard PRG security notion.

In the quantum setting, the for an appropriate strengthening of the PRG security notion — namely that the distribution of PRG outputs is *oracle indistinguishable* from uniform, in the sense

of Section 4.6 — the first phase of the GGM security proof caries through to the quantum setting. The difficulty is then in proving the equivalence of the stronger notion and weaker notion of PRG security. However, this is now easily handled by Theorem 4.17. Thus, we obtain the first proof of security for the GGM construction under quantum queries.

Construction 6.6 (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 $\mathsf{PRF} : \mathcal{K} \times \{0,1\}^n \to \mathcal{K}$ 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. We first define a stronger notion of security for pseudorandom generators, which we call oracle-security:

Definition 6.7 (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 4.17 gives:

Corollary 6.8. *If G is a secure PRG, then it is also oracle-secure.*

We now can prove the security of Construction 6.6.

Theorem 6.9. *If G is a standard-secure PRG, then* PRF *from Construction 6.6 is a QPRF.*

Proof. We adapt the security proof of Goldreich, Goldwasser, and Micali [GGM86] to convert any adversary for PRF into an adversary for the oracle-security of G. Then Corollary 6.8 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 $P \leftarrow \mathcal{K}^{\{0,1\}^{n-i}}$ (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 $i = n$, $P \leftarrow \mathcal{K}^{\{0,1\}^{n-i}}$ 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,

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

Notice that there is no absolute value in the definition of ϵ_i. 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 $P : \{0,1\}^{n-1} \to \mathcal{K}^2$, and distinguishes $P \leftarrow \left(\mathcal{K}^2\right)^{\{0,1\}^{n-1}}$ from $P \leftarrow G \circ \mathcal{K}^{\{0,1\}^{n-1}}$. 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 $P^{(i)} : \{0,1\}^{n-i-1} \to \mathcal{K}^2$ be the oracle $P^{(i)}(x) = P(0^i x)$
- Write $P^{(i)}$ as $(P_0^{(i)}, P_1^{(i)})$ where $P_b^{(i)} : \{0,1\}^{n-i-1} \to \mathcal{K}$ are the left and right halves of the output of $P^{(i)}$.
- Construct the oracle $O : \{0,1\}^n \to \mathcal{K}$ 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 $G \circ \mathcal{K}^{\{0,1\}^{n-1}}$, then $P^{(i)}$ is distributed according to $G \circ \mathcal{K}^{\{0,1\}^{n-i-1}}$, and so $P_b \leftarrow G_b \circ \mathcal{K}^{\{0,1\}^{n-i-1}}$. Thus $O = O_{i+1}$, the oracle in hybrid H_{i+1}. For fixed i, we then have that the quantity

$$\Pr_{P \leftarrow (\mathcal{K}^2)^{\{0,1\}^{n-1}}}\left[B_i^{|P\rangle}() = 1\right] - \Pr_{P \leftarrow G \circ \mathcal{K}^{\{0,1\}^{n-1}}}\left[B_i^{|P\rangle}() = 1\right]$$

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

$$\left|\Pr_{P \leftarrow (\mathcal{K}^2)^{\{0,1\}^{n-1}}}\left[B^{|P\rangle}() = 1\right] - \Pr_{P \leftarrow G \circ \mathcal{K}^{\{0,1\}^{n-1}}}\left[B^{|P\rangle}() = 1\right]\right| \ ,$$

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. □

6.1.3 Construction 2: The Synthesizer Construction

In this section, we show that the construction of pseudorandom functions from pseudorandom synthesizers due to Naor and Reingold [NR95] is quantum-secure.

Definition 6.10 (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 6.11 (Standard-Security)**.** A pseudorandom 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

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

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 6.12 (NR-PRF)**.** Given a pseudorandom synthesizer $S : \mathcal{X}^2 \to \mathcal{X}$, let ℓ be an integer and $n = 2^\ell$. We let $\mathsf{PRF}_k(x) = \mathsf{PRF}^{(\ell)}_k(x)$ where $\mathsf{PRF}^{(i)} : \left(\mathcal{X}^{2\times 2^i}\right) \times \{0,1\}^{2^i} \to \mathcal{X}$ 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.

Theorem 6.13. *If S is a standard-secure synthesizer, then* PRF *from Construction 6.12 is a QPRF.*

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 6.12 is quantum secure. Unlike the GGM case, the equivalence of quantum- and standard-security for synthesizers is not an immediate consequence of Theorem 4.17. Nevertheless, we prove the equivalence, completing the proof of security for Construction 6.12.

Recall the definition of standard-security for a synthesizer $S : \mathcal{X}^2 \to \mathcal{Y}$ from Definition 6.11: for all sets $\mathcal{Z}$, no efficient quantum algorithm A making classical queries to an oracle O from $\mathcal{Z}^2 \to \mathcal{Y}$ can tell if $O(z_1, z_2) = S(O_1(z_1), O_2(z_2))$ for random oracles $O_i \leftarrow \mathcal{X}^{\mathcal{Z}}$ or if O is truly random.

Since all queries are classical, and only a polynomial number of queries are possible, a simple argument shows that Definition 6.11 is equivalent to the case where $|\mathcal{Z}| \in n^{O(1)}$. Further, if $\mathcal{Z}$ is polynomial in size, we can query the entire set classically, so there is no advantage in having quantum queries. Therefore, Definition 6.11 is equivalent to the following:

Definition 6.14 (Standard-Security)**.** A pseudorandom synthesizer $S : \mathcal{X}^2 \to \mathcal{Y}$ is standard-secure if, for any set $\mathcal{Z}$ where $|\mathcal{Z}| \in n^{O(1)}$, no efficient quantum algorithm A making quantum queries can distinguish $O(z_1, z_2) = S(O_1(z_1), O_2(z_2))$ where $O_b \leftarrow \mathcal{X}^{\mathcal{Z}}$ from $O \leftarrow \mathcal{Y}^{\mathcal{Z}\times\mathcal{Z}}$.

Before proving security, we define the quantum-security of a pseudorandom synthesizer. The definition is similar to Definition 6.14, except that there is no bound on the size of $\mathcal{Z}$:

Definition 6.15 (Quantum-Security)**.** A pseudorandom synthesizer $S : \mathcal{X}^2 \to \mathcal{Y}$ is quantum-secure if, for any set $\mathcal{Z}$, no efficient quantum algorithm A making quantum queries can distinguish $O(z_1, z_2) = S(O_1(z_1), O_2(z_2))$ where $O_b \leftarrow \mathcal{X}^{\mathcal{Z}}$ from $O \leftarrow \mathcal{Y}^{\mathcal{Z}\times\mathcal{Z}}$

We now show that the two definitions are equivalent:

Lemma 6.16. *If S is standard-secure, then it is also quantum-secure.*

Proof. Let's define a new oracle distribution, which we will denote $\mathtt{AR}_s$, which stands for *almost random*. $\mathtt{AR}_s$ is defined as follows:

- Pick random oracles P_1 and P_2 from $[s]^{\mathcal{Z}}$.
- Pick a random oracle Q from $\mathcal{Z}^{[s]^2}$.
- Output the oracle $O(z_1, z_2) = Q(P_1(z_1), P_2(z_2))$.

Notice that as s goes to ∞, P_1 and P_2 become injective with probability approaching 1, and thus $\mathtt{AR}_\infty$ is the uniform distribution.

Now, let B be an adversary breaking the oracle-security of S with non-negligible probability ϵ. Define $\epsilon(s)$ as the following quantity:

$$\epsilon(s) = \left| \Pr_{O_1 \leftarrow \mathcal{X}^{\mathcal{Z}}, O_2 \leftarrow \mathcal{X}^{\mathcal{Z}}}[B^{S(O_1,O_2)}() = 1] - \Pr_{O \leftarrow \mathtt{AR}_s}[B^{O}() = 1] \right|$$

Then $\epsilon = \lim_{s\to\infty} \epsilon(s)$. Let r be an integer such that $\ell(q)/r = \epsilon/8$ where q in the number of queries made by B. We now replace O_i with $\mathsf{SR}_r^{\mathcal{X}}$, and the P_i (as a part of $\mathtt{AR}_s$) with $\mathsf{SR}_r^{[s]}$. Each of these changes will only change the behavior of A by $\epsilon/8$. Thus, a simple argument shows that

$$\left| \Pr_{O_i \leftarrow \mathsf{SR}_r^{\mathcal{X}}}[B^{S(O_1,O_2)}() = 1] - \Pr_{P_i \leftarrow \mathsf{SR}_r^{[s]}, Q \leftarrow \mathcal{Y}^{[s]^2}}[B^{Q(P_1,P_2)}() = 1] \right| \geq \epsilon(s) - \epsilon/2$$

Notice that we can think of the oracle $Q(P_1, P_2)$ as the oracle

$$O'(z_1, z_2) = Q(S_1 \circ R_1(z_1), S_2 \circ R_2(z_2)) = O(R_1(z_1), R_2(z_2))$$

Where $S_i \leftarrow \{0,1\}[s]^{\{0,1\}[r]}$, $R_i \leftarrow [r]^{\mathcal{Z}}$, and $O(w_1, w_2) = Q(S_1(w_1), S_2(w_2))$. As s goes to ∞, S_i become injective with probability converging to one, so O approaches a random function from $[r]^2 \to \mathcal{Y}$.

We now describe a new algorithm A which tries to break the standard-security of S according to Definition 6.14. A takes as input a quantum-accessible oracle O from $\{0,1\}[r]^2$ to $\mathcal{Y}$. A constructs two random oracles $R_1 \leftarrow \{0,1\}[r]^{\mathcal{Z}}$ and $R_2 \leftarrow [r]^{\mathcal{Z}}$, gives B the oracle $O'(z_1, z_2) = O(R_1(z_1), R_2(z_2))$, and simulates B. If $O = S(T_1, T_2)$ for random oracles $T_i \leftarrow \mathcal{X}^{\{0,1\}[r]}$, then the oracle seen by A is $O'(z_1, z_2) = S(O_1(z_1), O_2(z_2))$, where O_1 and O_2 are drawn from $\mathsf{SR}_r^{\mathcal{X}}$. If O is a random oracle, then the oracle seen by A is $O'(z_1, z_2) = O(R_1(z_1), R_2(z_2))$, where $R_i[r]^{\mathcal{Z}}$. This corresponds to the case where $s = \infty$, and thus the advantage of A in distinguishing these two cases is $\epsilon(\infty) - \epsilon/2 = \epsilon/2$. If ϵ is non-negligible, then there is a polynomial bounding r infinitely often, and in these cases, A breaks the standard-security of S.

□

We are now ready to prove that Construction 6.12 is quantum-secure:

Proof of Theorem 6.13. Let A be a quantum adversary breaking the quantum-security of PRF with probability ϵ. That is,

$$\left| \Pr_{k \leftarrow \mathcal{X}^{2^n}}[A^{\mathsf{PRF}_k}() = 1] - \Pr_{O \leftarrow \mathcal{X}^{\{0,1\}^n}}[A^{O}() = 1] \right| = \epsilon$$

Let hybrid H_i be the game where the oracle seen by A is PRF, except that each instance of $\mathsf{PRF}^{(i)}$

is replaced with a truly random function from $\{0,1\}^{2^i}$ into $\mathcal{X}$. Since $\mathsf{PRF}^{(0)}$ is already a random function, H_0 is equivalent to the case where the oracle is PRF. Similarly, H_ℓ is by definition the case where the oracle is truly random. Thus a simple hybrid argument shows that there is an i such that A can distinguish H_i from H_{i-1} with probability at least ϵ/ℓ.

We now describe an algorithm B which breaks the quantum-security of S. B is given an oracle from P from $\left(\mathcal{X} \times [2^{\ell-i}]\right)^2$ into $\mathcal{X}$, which is either $S(Q_1, Q_2)$ for random oracles $Q_b \leftarrow \mathcal{X}^{\mathcal{X}\times[2^{\ell-i}]}$ or a truly random oracle. It then constructs oracles

$$P_y(x_1, x_2) = P((x_1, y), (x_2, y))$$

Notice that there are $2^{\ell-i}$ possible y values, and that for fixed y, P_y is either a random oracle from $\mathcal{X}^2$ into $\mathcal{X}$, or it is $S(Q_{y,1}, Q_{y,2})$ for random oracles $Q_{y,b}$ from $\mathcal{X}$ to $\mathcal{X}$. We then construct the oracle O which is PRF, except that we stop the recursive construction at $\mathsf{PRF}^{(i)}$. There are $2^{\ell-i}$ different instances of $\mathsf{PRF}^{(i)}$, so we use the $2^{\ell-i}$ P_y oracles in their place. If P is $S(Q_1, Q_2)$, this corresponds to hybrid H_{i-1}, whereas if P is a random oracle, this corresponds to H_i. Thus, B distinguishes the two cases with probability ϵ/ℓ.

However, under the assumption that S is standard-secure, Lemma 6.16 shows that it is quantum-secure, meaning the algorithm B is impossible. Therefore, PRF is quantum-secure. □

6.1.4 Construction 3: A direct construction from lattices

In this section, we present the construction of pseudorandom functions from Banerjee, Peikert, and Rosen [BPR12]. 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 6.17. 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 $\mathsf{PRF} : \mathcal{K} \times \{0,1\}^\ell \to \mathbb{Z}_p^{m\times n}$ 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 $\mathbf{A}$, whose result is rounded mod p.

Next we prove that Construction 6.17 is secure when the secret key $(\mathbf{A}, \{\mathbf{S}_i\})$ is drawn from an appropriate distribution. First, we need to define the Learning With Errors (LWE) problem:

Definition 6.18 (Learning With Errors)**.** Let $q \geq 2$ an integer, n a security parameter, and $m = poly(n)$ and $w = poly(n)$ be integers. For a distribution χ over $\mathbb{Z}$ and a secret matrix $\mathbf{S} \in \mathbb{Z}_q^{n \times w}$, the LWE distribution $\mathsf{LWE}_{\mathbf{S},\chi}$ is the distribution over $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{m \times w}$ defined as follows:

- Choose a random matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$.
- Choose a random error matrix $\mathbf{E} \leftarrow \chi^{m \times w}$
- Output $(\mathbf{A}^t, \mathbf{B}^t = \mathbf{A}^t\mathbf{S} + \mathbf{E} \mod q)$

The LWE problem is then to distinguish between a polynomial number of samples from $\mathsf{LWE}_{\mathbf{S},\chi}$ for a fixed $\mathbf{S} \leftarrow \chi^{n \times w} \mod q$ from the same number of samples from the uniform distribution. The LWE problem is hard if, for all efficient quantum adversaries A, the probability A distinguishes these two cases is negligible in n.

We now define the oracle-LWE problem:

Definition 6.19 (Oracle-LWE)**.** The oracle-LWE problem is to distinguish an oracle O whose outputs are generated by $\mathsf{LWE}_{\mathbf{S},\chi}$ (where the same $\mathbf{S} \leftarrow \chi^{n \times w} \mod q$ is used for all points) from a truly random oracle O. We say that LWE is oracle-hard if, for all efficient adversaries A making quantum queries, A cannot distinguish these two distributions with more than negligible probability.

Lemma 6.20. *If* LWE *is hard, it is also oracle-hard.*

Proof. The proof is very similar to that of Theorem 4.17. Let A be an adversary breaking the oracle-hardness of LWE using q quantum queries with probability ϵ. Let r be an integer such that $\ell(q)/r \approx \epsilon/4$. We then construct an algorithm B, which takes as input r pairs $(\mathbf{A}_i^t, \mathbf{B}_i^t)$, and distinguishes when the pairs come from $\mathsf{LWE}_{\mathbf{S},\chi}$ for some fixed $\mathbf{S} \leftarrow \chi^{n \times w}$ from when the pairs are random. B works as follows:

- Construct the oracle O where $O(x)$ is selected at random from $(\mathbf{A}_i^t, \mathbf{B}_i^t)$
- Simulate A with oracle O, and output the output of A.

Using the same analysis as in the proof of Theorem 4.17, we get that B distinguishes the two cases with probability $\epsilon/2$. If ϵ is non-negligible, then there is a polynomial that bounds r infinitely often, and in these cases, the number of samples received by B is a polynomial, and hence B breaks the hardness of LWE. □

Next, we need to define the discrete Gaussian distribution:

Definition 6.21 (Discrete Gaussian)**.** Let $D_{\mathbb{Z},r}$ denote the discrete Gaussian distribution over $\mathbb{Z}$, where the probability of x is proportional to $e^{-\pi x^2/r^2}$.

We are now ready to state and prove the security of Construction 6.17:

Theorem 6.22. *Let $\chi = D_{\mathbb{Z},r}$, and $q \geq p \cdot \ell(Cr\sqrt{n+\ell})^\ell n^{\omega(1)}$ for some suitable universal constant C. Let* PRF *be as in Construction 6.17, and suppose each $\mathbf{S}_i$ is drawn from $\chi^{n\times n}$. If the* LWE *problem is hard for modulus q and distribution χ, then* PRF *from Construction 6.17 is a QPRF.*

Proof. The proof is very similar to that of Banerjee et al. Notice that our theorem requires $q \geq p \cdot \ell(Cr\sqrt{n+\ell})^\ell n^{\omega(1)}$ whereas the original only requires $q \geq p \cdot \ell(Cr\sqrt{n})^\ell n^{\omega(1)}$. We will explain why this is later. We first define a class of functions $G : \mathcal{K} \times \{0,1\}^k \to \mathbb{Z}_q^{m\times n}$ to be PRF without rounding. That is,

$$G_k(x) = \mathbf{A}^t \prod_{i=1}^{\ell} \mathbf{S}_i^{x_i}$$

Then $\mathsf{PRF}_k(x) = \lfloor G_k(x) \rceil_p$. We also define a related class of functions $\tilde{G}$ where $\tilde{G} = \tilde{G}^{(\ell)}$ and

- $\tilde{G}^{(0)}$ is a function from $\{0,1\}^0$ into $\mathbb{Z}_q^{m\times n}$ defined as follows: pick a random $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, and set $\tilde{G}^{(0)}(\epsilon) = \mathbf{A}^t$.
- $\tilde{G}^{(i)}$ is a function from $\{0,1\}^i$ into $\mathbb{Z}_q^{m\times n}$ defined as follows: pick a random $\tilde{G}^{(i-1)}$, pick $\mathbf{S}_i \leftarrow \chi^{n\times n}$ and for each $x' \in \{0,1\}^{i-1}$, pick $\mathbf{E}_{x'} \leftarrow \chi^{m\times n}$. Then

$$\tilde{G}^{(i)}(x = x'x_i) = \tilde{G}^{(i-1)}(x') \cdot \mathbf{S}_i^{x_i} + x_i \cdot \mathbf{E}_{x'} \mod q$$

Let A be an adversary that distinguishes PRF from a random function with probability ϵ.

First, consider the case where A sees a truly random function $U : \{0,1\}^k \to \mathbb{Z}_p^{m\times n}$. Replace U with $\lfloor U' \rceil_p$ where U' is a truly random function from $\{0,1\}^k \to \mathbb{Z}_q^{m\times n}$. For each input, the bias introduced by this rounding is negligible because $q \geq pn^{\omega(1)}$. Thus, by Theorem 4.17, the ability of A to distinguish these two cases is negligible.

Now, let $B = \ell(Cr\sqrt{n+k})^\ell$. Let $\mathsf{BAD}(y)$ be the event that

$$\lfloor y + [-B, B]^{m\times n} \rceil_p \neq \{\lfloor y \rceil_p\}$$

That is, $\mathsf{BAD}(y)$ is the event that y is very close to another element in $\mathbb{Z}_q$ that rounds to a different value in $\mathbb{Z}_p$. Banerjee et al. show that for each x, the probability that $\mathsf{BAD}(U'(x))$ occurs is negligible. Therefore, according to Theorem 4.17, $\mathsf{BAD}(U'(x))$ as an oracle with outputs in $\{\mathsf{True}, \mathsf{False}\}$ is indistinguishable from the oracle that always outputs False. Hence, it is impossible for an algorithm making quantum queries to U' to find an x such that $\mathsf{BAD}(U'(x))$ occurs, except with negligible probability.

The next step is to prove that U' and $\tilde{G}$ are oracle-indistinguishable. Once we have accomplished this, we replace U' with $\tilde{G}$. Then the probability that A detects this change is negligible. Additionally, it is also impossible to find an x such that $\mathsf{BAD}(\tilde{G}(x))$ occurs, except with negligible probability.

Lastly, we replace $\tilde{G}$ with G. Banerjee et al. show that as long as $\mathsf{BAD}(\tilde{G}(x))$ does not occur, $\left\lfloor \tilde{G}(x) \right\rceil_p = \lfloor G_k(x) \rceil_p = \mathsf{PRF}_k(x)$ with all but negligible probability. Our modification to the parameters of the theorem (replacing $\sqrt{n}$ with $\sqrt{n+k}$) allows us to choose C so that this probability is actually $2^{-\ell}\sigma$ for some negligible σ. Summing over all 2^ℓ different x, we get that, except with negligible overall probability, $\mathsf{PRF}_k(x) = \left\lfloor \tilde{G}(x) \right\rceil_p$ whenever $\mathsf{BAD}(\tilde{G}(x))$ does not occur.

Thus, if A distinguishes $\mathsf{PRF}_k(x)$ from $\left\lfloor \tilde{G}(x) \right\rceil_p$ with non-negligible probability, it must be that the sum over all queries made by A of the sum of the query magnitudes of all the x such that $\mathsf{BAD}(\tilde{G}(x))$ occurs is non-negligible (Theorems 3.1 and 3.3 of [BBBV97]). But this means we can find an x such that $\mathsf{BAD}(\tilde{G}(x))$ occurs with non-negligible probability (simply run A, and at a randomly chosen query, halt and sample the query). But, as we have already shown, this is impossible.

Hence, we have shown that PRF is indistinguishable from a random function.

It remains to show that U' and $\tilde{G}$ are oracle-indistinguishable. We show that this is true given that the LWE problem is oracle-hard. Using Lemma 6.20, we reach the same conclusion assuming LWE is hard, thus completing the theorem.

Let B be an adversary that distinguishes U' from $\tilde{G}$ with probability ϵ. Define hybrid H_i as the case where B is given the oracle O_i where $O_i = \tilde{G}$, except that, in the recursive definition of $\tilde{G}$, $\tilde{G}^{(i)}$ is replaced with a truly random function. H_0 corresponds to the correct definition of $\tilde{G}$, and H_k corresponds to U'. Thus, there exists an i such that B distinguishes H_i from H_{i-1} with probability ϵ/ℓ.

Construct an adversary C with access to an oracle $P : \{0,1\}^{i-1} \to \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^{m\times n}$. P is either a random function or each output is chosen according to the LWE distribution. In other words, $P(x) = (\mathbf{A}^t, \mathbf{B}^t)$, where either $\mathbf{A}(x)$ and $\mathbf{B}(x)$ are chosen at random for all x, or there is a secret $\mathbf{S} \leftarrow \chi^{n\times n}$ and $\mathbf{B}(x)^t = \mathbf{A}(x)^t\mathbf{S} + \mathbf{E}(x) \mod q$ where $\mathbf{E}(x) \leftarrow \chi^{m\times n}$.

For each $j > i$, C constructs random oracles $Q_j : \{0,1\}^{j-1} \to \mathbb{Z}^{m\times n}$ where $Q_j(x) \leftarrow \chi^{m\times n}$. C also generates $\mathbf{S}_j \leftarrow \chi^{n\times n}$ for $j > i$. Then C works as follows:

- Let $\tilde{G}^{(i)}(x = x'x_i) = \begin{cases} \mathbf{A}(x')^t & \text{if } x_i = 0 \\ \mathbf{B}(x')^t & \text{if } x_i = 1 \end{cases}$
- Let $\tilde{G}^{(j)}(x = x'x_j) = \tilde{G}^{(j-1)}(x') \cdot \mathbf{S}_j^{x_j} + x_j \cdot Q_j(x') \mod q$ for $j > i$.
- Let $O(x) = \tilde{G}^{(k)}(x)$
- Run B with oracle O.

When P is a random oracle, this corresponds to H_i. When P is the LWE oracle, this corresponds to H_{i-1}. Thus, C distinguishes these two cases with probability at least ϵ/ℓ. Under the assumption that LWE is oracle hard, this quantity, and hence ϵ, are negligible. We then use Lemma 6.20 to complete the theorem.

□

6.2 Quantum-secure MACs

In this section, we define quantum security for message authentication codes (MAC), and show how to build schemes meeting this definition. Informally, we wish to capture *quantum* chosen message attacks, where the adversary submits a superposition of messages

$$\sum_m \alpha_m |m\rangle$$

and the challenger responds with a superposition of tags on those messages:

$$\sum_m \alpha_m |m, \mathsf{Sig}(K, m)\rangle$$

However, there are several subtleties with actually defining security under this type of attack:

- **Randomness.** When using a randomized MAC scheme, there are several choices for how the randomness is used. One option is to choose a single randomness value for each chosen message query, and sign every message in the superposition with that randomness. Another approach is to choose fresh randomness for each message in the superposition. The drawback of the second approach is that whomever is implementing the scheme on a quantum device needs to guarantee that every message in the superposition is signed with fresh independent randomness.

  The first approach, where the same randomness is used to sign all messages in a superposition, is much simpler for implementers and we therefore design MAC schemes secure in this setting. Fortunately, there is a simple transformation that converts a scheme requiring independent randomness for every message into a scheme that is secure when a single randomness value is used for an entire query: when signing, choose a fresh random key k for a quantum pseudorandom function (QPRF). This will be the single per-query randomness value. To sign a superposition of messages, sign each message m in the superposition using randomness obtained by applying the QPRF to m using the key k. From the adversary's point of view, this is indistinguishable from choosing independent randomness for each message. Using Lemma 4.1, we can replace the QPRF with a function drawn from a pairwise independent function family, which is far more efficient than using a QPRF. Hence, requiring global randomness per query does not complicate the MAC scheme much, but greatly simplifies its implementation.

- **Forgeries.** Each quantum chosen message query can be a superposition of every message in the message space. Sampling the returned superposition will result in a single message/tag pair for a random message. Therefore, the classical notion of existential forgery being a tag on a *new* message is ill-defined when we allow quantum access. Instead, for security we require that the adversary cannot produce $q + 1$ valid message/tag pairs with q quantum chosen

message queries. Security definitions in this style were previously used in the context of blind signatures [PS96].

Definition 6.23 (Quantum security for message authentication codes)**.** We define several variants of security for a MAC system $(\mathsf{Sig}, \mathsf{Ver})$:

- $(\mathsf{Sig}, \mathsf{Ver})$ is strongly existentially unforgeable under a quantum chosen message attack (strongly EUF-qCMA secure) if all efficient quantum adversaries $\mathcal{A}$ have negligible advantage in the following experiment. A random key $k \in \mathcal{K}$ is chosen. $\mathcal{A}$ makes a polynomial number q of adaptive *quantum* signing queries on superpositions

$$\sum_{m,x,z} \alpha_{m,x,z} |m, x, z\rangle$$

and in response the challenger chooses a random string r and applies the deterministic procedure $m \longmapsto \mathsf{Sig}(k, m; r)$ to the superposition. That is, it returns the state

$$\sum_{m,x,z} \alpha_{m,x,z} |m, x \oplus \mathsf{Sig}(k, m; r), z\rangle \ .$$

Then, the adversary produces $q+1$ message/tag pairs $\{(m_i^*, \sigma_i^*)\}_{i \in [q+1]}$. The advantage of $\mathcal{A}$ is defined as the probability that $\mathsf{Ver}(k, m_i^*, \sigma_i^*)$ accepts for all i, and that $(m_i^*, \sigma_i^*) \neq (m_j^*, \sigma_j^*)$ for all $i \neq j$.

- $(\mathsf{Sig}, \mathsf{Ver})$ is q-time strongly EUF-qCMA secure if it is strongly EUF-CMA secure for adversaries limited to making q signing queries.

- We can also make weak variants of the above definitions by requiring that the forged messages are all distinct: that is $m_i^* \neq m_j^*$ for all $i \neq j$.

Relationship between classical and quantum security. Changing classical queries to quantum queries only enhances the adversaries power, and therefore one would naturally expect that quantum security is stronger than standard security against quantum adversaries. Indeed, in the following sections, we demonstrate protocols that are EUF-CMA secure against quantum adversaries, but EUF-qCMA insecure. However, our definition of quantum security also makes a stronger requirement on the adversary's output: namely he has to produce $q+1$ forgeries, whereas a standard security adversary only needs to produce one. Hence, it may be that the two definitions are incompatible.

We can "classicalize" the EUF-qCMA security definition by allowing only classical queries, but still requiring the adversary to output $q+1$ valid message/tag pairs. The resulting security definition, which we call EUF-CMA' security, is clearly implied by both EUF-qCMA and EUF-CMA security.

For several settings, the two classical security definitions, EUF-CMA and EUF-CMA', are actually equivalent. Combined with the results in the following sections, for these settings quantum security *is* strictly stronger than classical security. The settings are the following:

- **Deterministic schemes.** For deterministic schemes, we can assume without loss of generality that an adversary making classical queries never queries the same message twice, as he could just have recorded the signature on that message for later use. In this case, the q message/tag pairs received from signing queries plus the forgery message/tag pair will constitute $q+1$ distinct message/tag pairs. Thus, for deterministic schemes, EUF-CMA and EUF-CMA' are equivalent, and therefore quantum security is strictly stronger than classical security.
- **High min-entropy schemes.** For schemes where the signing algorithm has high min-entropy on each message, repeated queries on the same message will most likely give different tags. As a result, q message/tag pairs received from signing queries will be distinct, and therefore the strong notions of the two classical security definitions are again equivalent.

The two cases above handle most reasonable schemes, and therefore for most schemes quantum security is strictly stronger than classical security.

Constructing quantum-secure MACs. Just like the the PRF setting, we may hope that classical MAC constructions are actually quantum-secure. Classically, for example, any PRF with sufficiently large output is a secure MAC. What about quantumly? It turns out that, even assuming a quantum-secure PRF, proving its quantum security is non-trivial. In the following sections, using the techniques built in Chapters 3, we show that quantum-secure PRFs actually do give quantum-secure MACs. We also analyze several other classical MAC constructions; in some cases the classical constructions are actually quantum insecure, and in others the classical scheme or a slight variant can be proved quantum secure.

6.2.1 q-time MACs

We begin by discussing quantum one-time MACs, which are MACs secure against a single quantum query. More generally, we will discuss quantum q-time MACs.

Classically, any pairwise independent function is a one-time MAC. In the quantum setting, Corollary 3.4 shows that when the range is much larger than the domain, this still holds. However, such MACs are not useful since we want the tag to be short. We first show that when the range is not larger than the domain, pairwise independence is not enough to ensure security:

Theorem 6.24. *For any set $\mathcal{Y}$ of prime-power size, and any set $\mathcal{X}$ with $|\mathcal{X}| \geq |\mathcal{Y}|$, there exist $(q+1)$-wise independent functions from $\mathcal{X}$ to $\mathcal{Y}$ that are not q-time MACs.*

Thus there are classical one-time MACs that are not quantum 1-time MACs, giving an unconditional separation between the two notions. To prove this theorem, we treat $\mathcal{Y}$ as a finite field, and assume $\mathcal{X} = \mathcal{Y}$, as our results are easy to generalize to larger domains. We use random degree q polynomials as our $(q+1)$-wise independent family, and show in Theorem 6.25 below that such polynomials can be completely recovered using only q quantum queries. It follows that the derived MAC cannot be q-time secure since once the adversary has the polynomial it can easily forge tags on new messages.

Theorem 6.25. *For any prime power n, there is an efficient quantum algorithm that makes only q quantum queries to an oracle implementing a degree-q polynomial $F : \mathbb{F}_n \to \mathbb{F}_n$, and completely determines F with probability $1 - O(qn^{-1})$.*

The theorem shows that a $(q+1)$-wise independence family is not necessarily a secure quantum q-time MAC since after q quantum chosen message queries the adversary extracts the entire secret key. The case $q = 1$ of Theorem 6.25 is particularly interesting.

The problem of recovering the coefficients of a polynomial by quantum queries was previously studied by Kane and Kutin [KK11], where they show that q quantum queries cannot recover all the coefficients of a degree $2q$ polynomial, and conjecture that q queries cannot even recover the coefficients of a degree q polynomial. Theorem 6.25 refutes this conjecture.

The following lemma will be used to prove Theorem 6.25:

Lemma 6.26. *For any prime power n, and any subset $\mathcal{X} \subseteq \mathbb{F}_n$ of size $n - k$, there is an efficient quantum algorithm that makes a single quantum query to any degree-1 polynomial $F : \mathcal{X} \to \mathbb{F}_n$, and completely determines F with probability $1 - O(kn^{-1})$.*

Proof. Write $F(x) = ax + b$ for values $a, b \in \mathbb{F}_n$, and write $n = p^t$ for some prime p and integer t. We design an algorithm to recover a and b.

Initialize the quantum registers to the state

$$|\psi_1\rangle = \frac{1}{\sqrt{n-k}} \sum_{x \in \mathcal{X}} |x, 0\rangle$$

Next, make a single oracle query to F, obtaining

$$|\psi_2\rangle = \frac{1}{\sqrt{n-k}} \sum_{x \in \mathcal{X}} |x, ax + b\rangle$$

Note that we can interpret elements $z \in \mathbb{F}_n$ as vectors $\mathbf{z} \in \mathbb{F}_p^t$. Let $\langle \mathbf{y}, \mathbf{z} \rangle$ be the inner product of vectors $\mathbf{y}, \mathbf{z} \in \mathbb{F}_p^t$. Multiplication by a in $\mathbb{F}_n$ is a linear transformation over the vector space $\mathbb{F}_p^t$, and can therefore be represented by a matrix $\mathbf{M}_a \in \mathbb{F}_p^{t \times t}$. Thus, we can write

$$|\psi_2\rangle = \frac{1}{\sqrt{n-k}} \sum_{\mathbf{x} \in \mathcal{X}} |\mathbf{x}, \mathbf{M}_a \mathbf{x} + \mathbf{b}\rangle$$

Note that in the case $t = 1$, a is a scalar in $\mathbb{F}_p$, so $\mathbf{M}_a$ is just the scalar a.

Now, the algorithm applies the Fourier transform to both registers, to obtain

$$|\psi_3\rangle = \frac{1}{n\sqrt{n-k}} \sum_{\mathbf{y},\mathbf{z}} \left(\sum_{\mathbf{x}\in\mathcal{X}} \omega_p^{\langle \mathbf{x},\mathbf{y}\rangle + \langle \mathbf{M}_a\mathbf{x}+\mathbf{b},\mathbf{z}\rangle} \right) |\mathbf{y},\mathbf{z}\rangle$$

where ω_p is a complex primitive pth root of unity.

The term in parenthesis can be written as

$$\left(\sum_{\mathbf{x}\in\mathcal{X}} \omega_p^{\langle \mathbf{x},\mathbf{y}+\mathbf{M}_a^T\mathbf{z}\rangle} \right) \omega_p^{\langle \mathbf{b},\mathbf{z}\rangle}$$

We will then do a change of variables, setting $\mathbf{y}' = \mathbf{y} + \mathbf{M}_a^T\mathbf{z}$.

Therefore, we can write the state as

$$|\psi_3\rangle = \frac{1}{n\sqrt{n-k}} \sum_{\mathbf{y}',\mathbf{z}} \left(\sum_{\mathbf{x}\in\mathcal{X}} \omega_p^{\langle \mathbf{x},\mathbf{y}'\rangle} \right) \omega_p^{\langle \mathbf{b},\mathbf{z}\rangle} |\mathbf{y}' - \mathbf{M}_a^T\mathbf{z},\mathbf{z}\rangle$$

For $\mathbf{z} \neq 0$ and $\mathbf{y}' = 0$, we will now explain how to recover a from $(-\mathbf{M}_a^T\mathbf{z},\mathbf{z})$. Notice that the transformation that takes $\mathbf{a}$ and outputs $-\mathbf{M}_a^T\mathbf{z}$ is a linear transformation. Call this transformation $\mathbf{L}_z$. The coefficients of $\mathbf{L}_z$ are easily computable, given $\mathbf{z}$, by applying the transformation to each of the unit vectors. Notice that if $t = 1$, $\mathbf{L}_z$ is just the scalar $-z$. We claim that $\mathbf{L}_z$ is invertible if $\mathbf{z} \neq 0$. Suppose there is some $\mathbf{a}$ such that $\mathbf{L}_z\mathbf{a} = -\mathbf{M}_a^T\mathbf{z} = 0$. Since $\mathbf{z} \neq 0$, this means the linear operator $-\mathbf{M}_a^T$ is not invertible, so neither is $-\mathbf{M}_a$. But $-\mathbf{M}_a$ is just multiplication by $-a$ in the field $\mathbb{F}_n$. This multiplication is only non-invertible if $-a = 0$, meaning $\mathbf{a} = 0$, a contradiction. Therefore, the kernel of $\mathbf{L}_z$ is just 0, so the map is invertible.

Therefore, to compute a, compute the inverse operator $\mathbf{L}_z^{-1}$ and apply it to $-\mathbf{M}_a^T\mathbf{z}$, interpreting the result as a field element in $\mathbb{F}_n$. The result is a. More specifically, for $\mathbf{z} \neq 0$, apply the computation mapping $(\mathbf{y},\mathbf{z})$ to $(\mathbf{L}_z^{-1}\mathbf{y},\mathbf{z})$, which will take $(-\mathbf{M}_a^T\mathbf{z},\mathbf{z})$ to $(a,\mathbf{z})$. For $\mathbf{z} = 0$, we will just apply the identity map, leaving both registers as is. This map is now reversible, meaning this computation can be implemented as a quantum computation. The result is the state

$$|\psi_4\rangle = \frac{1}{n\sqrt{n-k}} \sum_{\mathbf{y}'} \left(\sum_{\mathbf{x}\in\mathcal{X}} \omega_p^{\langle \mathbf{x},\mathbf{y}'\rangle} \right) \left(\sum_{\mathbf{z}\neq 0} \omega_p^{\langle \mathbf{b},\mathbf{z}\rangle} |\mathbf{L}_z^{-1}\mathbf{y}' + a,\mathbf{z}\rangle + |\mathbf{y}',0\rangle \right)$$

We will now get rid of the $|\mathbf{y}',0\rangle$ terms by measuring whether $\mathbf{z} = 0$. The probability that $\mathbf{z} = 0$ is $1/n$, and in this case, we abort. Otherwise, we are left if the state

$$|\psi_5\rangle = \frac{1}{\sqrt{n(n-1)(n-k)}} \sum_{\mathbf{z}\neq 0,\mathbf{y}'} \left(\sum_{\mathbf{x}\in\mathcal{X}} \omega_p^{\langle \mathbf{x},\mathbf{y}'\rangle} \right) \omega_p^{\langle \mathbf{b},\mathbf{z}\rangle} |\mathbf{L}_z^{-1}\mathbf{y}' + a,\mathbf{z}\rangle$$

The algorithm then measures the first register. Recall that $\mathcal{X}$ has size $n-k$. The probability the outcome of the measurement is a is then $(1-k/n)$. In this case, we are left in the state

$$|\psi_6\rangle = \frac{1}{\sqrt{n-1}} \sum_{\mathbf{z}\neq 0} \omega_p^{\langle \mathbf{b},\mathbf{z}\rangle} |\mathbf{z}\rangle$$

Next, the algorithm performs the inverse Fourier transform to the second register, arriving at the state

$$|\psi_7\rangle = \frac{1}{\sqrt{n(n-1)}} \sum_{\mathbf{w}} \left(\sum_{\mathbf{z}\neq 0} \omega_p^{\langle \mathbf{b}-\mathbf{w},\mathbf{z}\rangle} \right) |\mathbf{w}\rangle$$

Now the algorithm measures again, and interpret the resulting vector as a field element. The probability that the result is b is $1-1/n$. Therefore, with probability $(1-k/n)(1-1/n)^2 = 1-O(k/n)$, the algorithm outputs both a and b.

□

Now we use this attack to obtain an attack on degree-d polynomials, for general d:

Proof of Theorem 6.25. We show how to recover the $q+1$ different coefficients of any degree-q polynomial, using only $q-1$ classical queries and a single quantum query.

Let a be the coefficient of x^q, and b the coefficient of x^{q-1} in $F(x)$. First, make $q-1$ *classical* queries to arbitrary distinct points $\{x_1, ..., x_{q-1}\}$. Let $Z(x)$ be the unique polynomial of degree $q-2$ such that $r(x_i) = F(x_i)$, using standard interpolation techniques. Let $G(x) = F(x) - Z(x)$. $G(x)$ is a polynomial of degree q that is zero on the x_i, so it factors, allowing us to write

$$F(x) = Z(x) + (a'x + b') \prod_{i=1}^{q-1} (x - x_i)$$

By expanding the product, we see that $a = a'$ and $b = b' - a\sum x_i$. Therefore, we can implement an oracle mapping x to $a(x + \sum x_i) + b$ as follows:

- Query F on x, obtaining $F(x)$.
- Compute $Z(x)$, and let $G(x) = F(x) - Z(x)$.
- Output $G(x)/\prod(x - x_i) = a(x + \sum x_i) + b$.

This oracle works on all inputs except the $q-1$ different x_i values. We run the algorithm from Lemma 6.26 on $\mathcal{X} = \mathbb{F}_n \setminus \{x_i\}$, we will recover with probability $1 - O(q/n)$ both a and $b + a\sum x_i$ using a single quantum query, from which we can compute a and b. Along with the $F(x_i)$ values, we can then reconstruct the entire polynomial. □

Sufficient Conditions for a One-Time Mac

We use Lemma 4.1 to show that $(3q+1)$-wise independence is sufficient for q-time MACs. We note that finding $q+1$ input/output pairs is an easier problem than recovering the coefficients for a degree d polynomial in the case $d > q$ (and in the case $d \leq q$, both problems are easy given q quantum queries). Therefore, the results of Kane and Kutin [KK11] are insufficient for proving MAC security.

Theorem 6.27. *Any $(3q+1)$-wise independent family with domain $\mathcal{X}$ and range $\mathcal{Y}$ is a quantum q-time secure MAC provided $(q+1)/|\mathcal{Y}|$ is negligible.*

Proof. Let D be some $(3q+1)$-wise independent function. Suppose we have an adversary $\mathcal{A}$ that makes q quantum queries to an oracle H, and attempts to produces $q+1$ input/output pairs. Let ϵ_R be the probability of success when H is a random oracle, and let ϵ_D be the probability of success when H is drawn from D. We construct an algorithm $\mathcal{B}$ with access to H as follows: simulate $\mathcal{A}$ with oracle access to H. When $\mathcal{A}$ outputs $q+1$ input/output pairs, simply make $q+1$ queries to H to check that these are valid pairs. Output 1 if and only if all pairs are valid. Therefore, $\mathcal{B}$ makes q quantum queries and $c = q+1$ classical queries to H, and outputs 1 if and only if $\mathcal{A}$ succeeds: if H is random, $\mathcal{B}$ outputs 1 with probability ϵ_R, and if H is drawn from D, $\mathcal{B}$ outputs 1 with probability ϵ_D. Now, since D is $(3q+1)$-wise independent and $3q+1 = 2q+c$, Lemma 4.1 shows that the distributions of outputs when H is drawn from D is identical to that when H is random, meaning $\epsilon_D = \epsilon_R$.

Thus, when H is drawn from D, $\mathcal{A}$ succeeds with the same probability that it would if H was random. But we already know from Theorem 3.6 that if H is truly random, $\mathcal{A}$ success probability is less than $(q+1)/|\mathcal{Y}|$. Therefore, when H is drawn from D, $\mathcal{A}$ succeeds with probability less than $(q+1)/|\mathcal{Y}|$, which is negligible. Hence, if H is drawn from D, H is a q-time MAC. □

6.2.2 Many-time MACs

In this section, we show how to build quantum many-time MACs.

Quantum-secure PRFs are Quantum-secure MACs

Using Theorem 3.6 we show that a quantum secure pseudorandom function gives rise to the quantum-secure MAC, namely $\mathsf{Sig}(k,m) = \mathsf{PRF}(k,m)$. We prove that this mac is secure.

Theorem 6.28. *If $\mathsf{PRF} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ is a quantum-secure pseudorandom function and $1/|\mathcal{Y}|$ is negligible, then $(\mathsf{Sig}, \mathsf{Ver})$ where $\mathsf{Sig}(k,m) = \mathsf{PRF}(k,m)$ and $\mathsf{Ver}(k,m,\sigma)$ checks that $\sigma = F(k,m)$ is a EUF-qCMA-secure MAC.*

Proof. Let $\mathcal{A}$ be an efficient quantum adversary that makes q quantum queries to $\mathsf{Sig}(k,\cdot)$ and produces $q+1$ valid input/output pairs with probability ϵ. Let **Game 0** be the standard quantum MAC attack game. By definition, A's success probability in this game is ϵ.

Let **Game 1** be the same as **Game 0**, except that $\mathsf{Sig}(k,\cdot)$ is replaced with a truly random function $O : \mathcal{X} \to \mathcal{Y}$, and define $\mathcal{A}$'s success probability as the probability that $\mathcal{A}$ outputs $q+1$ input/output pairs of O. Since PRF is a quantum-secure PRF, $\mathcal{A}$'s advantage in distinguishing **Game 0** from **Game 1** is negligible.

Now, in **Game 1**, $\mathcal{A}$ makes q quantum queries to a random oracle, and tries to produce $q+1$ input/output pairs. However, by Theorem 3.6 and Eq. (3.1.1) we know that $\mathcal{A}$'s success probability is bounded by $(q+1)/|\mathcal{Y}|$ which is negligible. It now follows that ϵ is negligible and therefore, $(\mathsf{Sig}, \mathsf{Ver})$ is a EUF-qCMA-secure MAC. □

Carter-Wegman MACs

In this section, we show how to modify the Carter-Wegman MAC so that it is secure in the quantum setting. Recall that H is an XOR-universal family of hash functions from $\mathcal{X}$ into $\mathcal{Y}$ if for any two distinct points x and y, and any constant $c \in \mathcal{Y}$,

$$\Pr_{h \xleftarrow{R} H}[H(x) - H(y) = c] = 1/|\mathcal{Y}|$$

The Carter-Wegman construction uses a pseudorandom function family PRF with domain $\mathcal{X}$ and range $\mathcal{Y}$, and an XOR-universal family of hash functions $\mathcal{H}$ from $\mathcal{M}$ to $\mathcal{Y}$. The key is a pair (k, H), where k is a key for PRF and H is a function drawn from $\mathcal{H}$. To sign a message, pick a random $r \in \mathcal{X}$, and return $(r,\ \mathsf{PRF}(k,r) + H(m))$.

This MAC is not, in general, secure in the quantum setting. The reason is that the same randomness is used in all slots of a quantum chosen message query. That is the signing oracle computes:

$$\sum_m \alpha_m |m\rangle \longrightarrow \sum_m \alpha_m |m, r, \mathsf{PRF}(k,r) + H(m)\rangle$$

where the same r is used for all classical states of the superposition. For example, suppose $\mathcal{H}$ is the set of functions $H(x) = ax + b$ for random a and b. With even a single quantum query, the adversary will be able to obtain a and $\mathsf{PRF}(k,r) + b$ with high probability, using the algorithm from Theorem 6.25 in Section 6.2.1. Knowing both of these will allow the adversary to forge any message.

We show how to modify the standard Carter-Wegman MAC to make it secure in the quantum setting.

Construction 6.29 (Quantum Carter-Wegman)**.** The Quantum Carter-Wegman MAC (QCW-MAC) is built from a pseudorandom function PRF, an XOR-universal set of functions $\mathcal{H}$, and a pairwise independent set of functions $\mathcal{R}$.

Keys: The secret key for QCW-MAC is a pair (k, H), where k is a key for PRF and $H : \mathcal{M} \to \mathcal{Y}$ is drawn from $\mathcal{H}$

Signing: To sign a message m choose a random $R \in \mathcal{R}$ and output the pair $\big(R(m),\ \mathsf{PRF}(k, R(m)) + H(m)\ \big)$ as the tag. When responding to a quantum chosen message query, the same R is used in all classical states of the superposition.

Verification: To verify that (r, s) is a valid tag for m, accept iff $\mathsf{PRF}(k, r) + H(m) = s$.

Theorem 6.30. *The Quantum Carter-Wegman MAC is a EUF-qCMA secure MAC.*

Proof. We start with an adversary $\mathcal{A}$ that makes q tag queries, and then produces $q + 1$ valid message/tag pairs with probability ϵ. We now adapt the classical Carter-Wegman security proof to our MAC in the quantum setting.

When the adversary makes query i on the superposition

$$\sum_{m,y,z} \alpha^{(i)}_{m,y,z} |m, y, z\rangle \ ,$$

the challenger responds with the superposition

$$\sum_{m,y,z} \alpha^{(i)}_{m,y,z} |m, y + S_i(m), z\rangle$$

where $S_i(m) = (R_i(m), \mathsf{PRF}(k, (R_i(m)) + H(m))$ for a randomly chosen $R_i \in \mathcal{R}$, where $\mathcal{R}$ is a pairwise independent set of functions.

The adversary then creates $q + 1$ triples (m_j, r_j, s_j) which, with probability ϵ, are valid message/tag tuples. That means $H(m_j) + \mathsf{PRF}(k, r_j) = s_j$ for all j.

We now prove that ϵ must be small using a sequence of games:

Game 0. Run the standard MAC game, responding to query i with the oracle that maps m to $(R_i(m), \mathsf{PRF}(k, R_i(m)) + H(m))$, where R_i is a random function from $\mathcal{R}$. The advantage of A in this game is the probability is produces $q + 1$ forgeries. Denote this advantage as ϵ_0, which is equal to ϵ.

Game 1. Replace $\mathsf{PRF}(k, \cdot)$ with a truly random function F, and denote the advantage in this game as ϵ_1. Since PRF is a quantum-secure PRF, ϵ_1 is negligibly close to ϵ_0.

Game 2. Next we change the goal of the adversary. The adversary is now asked to produce a triple (m_0, m_1, s) where $H(m_0) - H(m_1) = s$. Given an adversary $\mathcal{A}$ for Game 1, we construct an adversary $\mathcal{B}$ for Game 2 as follows: run $\mathcal{A}$, obtaining $q + 1$ forgeries (m_j, r_j, s_j) such that $H(m_j) + F(r_j) = s_j$ with probability ϵ_1. If all r_j are distinct, abort. Otherwise, assume without loss of generality that $r_0 = r_1$. Then

$$H(m_0) - H(m_1) = (s_0 - F(r_0)) - (s_1 - F(r_1)) = s_0 - s_1$$

so output $(m_0, m_1, s_0 - s_1)$. Let ϵ_2 be the advantage of $\mathcal{B}$ in this game. Let p be the probability that all r_j are distinct and $\mathcal{A}$ succeeds. Then $\epsilon_2 \geq \epsilon_1 - p$.

We wish to bound p. Define a new algorithm $\mathcal{C}$, with oracle access to F, that first generates H, and then runs A, playing the role of challenger to $\mathcal{A}$. When $\mathcal{A}$ outputs $q+1$ triples (m_j, r_j, s_j), $\mathcal{C}$ outputs $q+1$ pairs $(r_j, s_j - H(m_j))$. If $\mathcal{A}$ succeeded, then $H(m_j)+F(r_j) = s_j$, so $F(r_j) = s_j - H(m_j)$, meaning the pairs $\mathcal{C}$ outputs are all input/output pairs of F. If all the r_j are distinct, then $\mathcal{C}$ will output $q+1$ input/output pairs, which is impossible except with probability at most $(q+1)/|\mathcal{Y}|$. Therefore, $p \leq (q+1)/|\mathcal{Y}|$. Therefore, as long as $|\mathcal{Y}|$ is super-polynomial in size, p is negligible, meaning ϵ_2 is negligibly close to ϵ_1.

Game 3. Now modify the game so that we draw R_i uniformly at random from the set of all oracles. Notice that each R_i is queried only once, meaning pairwise-independent R_i look exactly like truly random R_i, so **Game 3** looks exactly like **Game 2** from the point of view of the adversary. Thus the success probability ϵ_3 is equal to ϵ_2.

Game 4. For this game, we answer query i with the oracle that maps m to $(R_i(m), F(R_i(m))$. That is, we ignore H for answering MAC queries. Let ϵ_4 be the success probability in this game. We show that ϵ_4 is negligibly-close to ϵ_3 using a sequence of sub-games. **Game 3a** is the game where we answer query i with the oracle that maps m to $(R_i(m), P_i(m) + H(m))$ where P_i is another random oracle. Notice that we can define oracles $R(i,m) = R_i(m)$ and $P(i,m) = P_i(m)$, which are random oracles. Thus we can simulate **Game 3** with the oracle $(R(i,m), F(R(i,m)))$ where F and R are random functions. Thus $F(R(i,m))$ is a small-range function from Section 4.5, and Lemma 4.16 shows that $F(R(i,m))$ is indistinguishable from a truly random function $P(i,m)$ except with probability $O(q^3/|\mathcal{X}|) = \mathsf{negl}$, even when given oracle access to $R(i,m)$. Replacing $F(R(i,m))$ with $P(i,m)$ moves us to **Game 3a**, showing that **Game 3a** is negligibly close to that of **Game 3**. Notice that since P_i is random, $P_i'(m) = P_i(m) + H(m)$ is also random, so **Game 3a** is equivalent to the game were we answer query i with the oracle that maps m to $(R_i(m), P_i(m))$. Using the above lemma again, the success probability of $\mathcal{B}$ in this game is negligibly close to that of Game 4.

Now, we claim that ϵ_4, the success probability in **Game 4** is negligible. Indeed, the view of $\mathcal{B}$ is independent of H, so the probability that $H(m_0) - H(m_1) = s$ is $1/|\mathcal{Y}|$. Since ϵ_4 is negligibly close to $\epsilon = \epsilon_0$, the advantage of $\mathcal{A}$, we have that $\mathcal{A}$'s advantage is also negligible. □

6.3 Quantum-secure Signatures

In this section, we study the quantum security of signature schemes, which are essentially a public key version of message authentication codes. Like in the MAC case, we ask for security even when the adversary can make signing queries on arbitrary superpositions of messages. Unsurprisingly, the security definitions for signatures are very similar to the MAC definitions in Section 6.2.

Definition 6.31 (Quantum security for signatures)**.** We define several variants of security for a signature scheme $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$:

- $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ is strongly existentially unforgeable under a quantum chosen message attack (strongly EUF-qCMA secure) if all efficient quantum adversaries $\mathcal{A}$ have negligible advantage in the following experiment. A signing and verification key pair $(\mathsf{sk}, \mathsf{vk}) \xleftarrow{R} \mathsf{Gen}()$ is generated. $\mathcal{A}$ receives vk, and then makes a polynomial number q of adaptive *quantum* signing queries on superpositions
$$\sum_{m,x,z} \alpha_{m,x,z} |m, x, z\rangle$$
and in response the challenger chooses a random string r and applies the deterministic procedure $m \longmapsto \mathsf{Sig}(\mathsf{sk}, m; r)$ to the superposition. That is, it returns the state
$$\sum_{m,x,z} \alpha_{m,x,z} |m, x \oplus \mathsf{Sig}(\mathsf{sk}, m; r), z\rangle \ .$$
Then, the adversary produces $q+1$ message/tag pais $\{(m_i^*, \sigma_i^*)\}_{i \in [q+1]}$. The advantage of $\mathcal{A}$ is defined as the probability that $\mathsf{Ver}(\mathsf{vk}, m_i^*, \sigma_i^*)$ accepts for all i, and that $(m_i^*, \sigma_i^*) \neq (m_j^*, \sigma_j^*)$ for all $i \neq j$.
- $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ is q-time strongly EUF-qCMA secure if it is strongly EUF-CMA secure for adversaries limited to making q signing queries.
- We can also make weak variants of the above definitions by requiring that the forged messages are all distinct: that is $m_i^* \neq m_j^*$ for all $i \neq j$.

Constructing quantum-secure MACs. As in the MAC case, we may hope that classical signature constructions from the literature actually give rise to quantum-secure constructions. Unfortunately, our techniques seem insufficient for proving the quantum security of most existing signatures. Instead, we give three generic transformations from standard signatures into quantum secure signatures. Our transformations have small computational overhead compared to the original schemes. One existing scheme that we are able to prove quantum security for is the GPV signature scheme from Section 5.2.1. First, however, we show a separation between quantum and standard security. Similarly to MACs, quantum and standard security may in general be incompatible. However, for determinstic signatures or signatures with high min-entropy, our separation shows that quantum security is strictly stronger than standard security.

6.3.1 Separation

Next we show that quantum chosen message queries give the adversary more power than classical chosen message queries. In particular, we present a signature scheme that is secure under classical queries, but completely insecure once an adversary can make quantum queries. Let $(\mathsf{Gen}_c, \mathsf{Sig}_c, \mathsf{Ver}_c)$ be a signature scheme that is secure under classical chosen message queries. We augment $(\mathsf{Gen}_c, \mathsf{Sig}_c, \mathsf{Ver}_c)$ as follows. Let t be the bit-length of the secret key sk_c. Let PRF_0 be a quantum secure PRF, and let PRF_1 be a quantum-gap PRF from Section 6.1. That is, PRF_1 is (standard) secure against quantum adversaries, but quantum queries can distinguih PRF_1 from random with all but negligible error. Assume for simplicity that PRF_0 and PRF_1 share the same key space $\mathcal{K}$, domain $\mathcal{X}$, and range $\mathcal{Y}$; the key spaces can be always be padded so that this is the case, and the domain and random can easily be modified as well. The secret key will consist of a Sig_c secret key sk_c, as well as t PRF keys $k_i \in \mathcal{K}$. The message space $\mathcal{M}$ of the signature scheme will be $\mathcal{X}^t$. Then the signature on $m = (x_1, \ldots, x_t)$ will consist of an Sig_c signature, as well as the values $\mathsf{PRF}_{\mathsf{sk}_i}(k_i, x_i)$ for each i, where sk_i is the ith bit of sk_c. For verification, the augmented part of the signature will be ignored.

Under classical queries, PRF_0 and PRF_1 are both indistinguishable from random functions, so the auxiliary parts of the signature reveal nothing about sk_c. Therefore, the scheme inherits EUF-CMA security from the underlying signature. Under quantum queries, PRF_0 and PRF_1 can be distinguished almost perfectly (since PRF_0 is indistinguishable from a random oracle, but PRF_1 is almost perfectly distinguishable from a random oracle). Thus, quantum queries can reveal bits of the secret key, leading to a total break.

Construction 6.32. Let $\mathsf{PRF}_0, \mathsf{PRF}_1$ be PRFs with key space $\mathcal{K}$, domain $\mathcal{X}$, and co-domain $\mathcal{Y}$. Let $(\mathsf{Gen}_c, \mathsf{Sig}_c, \mathsf{Ver}_c)$ be a signature scheme where secret keys have bit length t and the message space is $\mathcal{M} = \mathcal{X}^t \times \mathcal{Y}^t$[2]. We construct a new signature scheme $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ with message space $\mathcal{X}^t$ as follows:

$$\mathsf{Gen}() : (\mathsf{sk}_c, \mathsf{vk}_c) \xleftarrow{R} \mathsf{Gen}_c(),\ k = (k_1, \ldots, k_t) \xleftarrow{R} \mathcal{K}^t,\ \text{Output } (\mathsf{sk} = (\mathsf{sk}_c, k), \mathsf{vk} = \mathsf{vk}_c)$$

$$\begin{aligned}\mathsf{Sig}((\mathsf{sk}_c, k), m) : &\text{ Write } \mathsf{sk}_c = (\mathsf{sk}_1, \ldots, \mathsf{sk}_t) \in \{0,1\}^t,\ m = (m_1, \ldots, m_t),\ k = (k_1, \ldots, k_t)\\ &y_i \leftarrow \mathsf{PRF}_{\mathsf{sk}_i}(k_i, m_i) \forall i \in [t],\ \sigma = \mathsf{Sig}(\mathsf{sk}_c, (m_1, \ldots, m_t, y_1, \ldots, y_t))\\ &\text{Output } \sigma = (\sigma_c, y_1, \ldots, y_t)\end{aligned}$$

$$\begin{aligned}\mathsf{Ver}(\mathsf{vk}_c, m, \sigma) : &\text{ Write } \sigma = (\sigma_c, y_1, \ldots, y_t)\\ &\text{Output } \mathsf{Ver}_c(\mathsf{sk}_c, (m, y_1, \ldots, y_t), \sigma_c)\end{aligned}$$

[2]We can always make the message space arbitrarily large by hashing. This can be done using only one-way functions, which are implied by the existence of signature schemes. Therefore, requiring this large message space does not add any assumptions

Theorem 6.33. *If* $(\mathsf{Gen}_c, \mathsf{Sig}_c, \mathsf{Ver}_c)$ *is (strongly) existentially unforgeable under a* classical *chosen message attack,* PRF *is a quantum-secure PRF, and* PRF_1 *is a quantum-gap PRF, then* $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ *is also (strongly) existentially unforgeable under a classical chosen message attack, but a single quantum chosen message query suffices to recover the secret key with overwhelming probability.*

Using the fact that classical signatures can be build from one-way functions [Rom90], using Theorem 6.9 to build quantum-secure PRFs from any PRG, Theorem 6.3 to builds quantum-gap PRFs from quantum-secure PRFs, and finally the fact that PRGs can be built from one-way functions [HILL99], we get the following corollary:

Corollary 6.34. *Assuming the existence of one-way functions, there are (strongly) EUF-CMA signatures that for which the secret key can be recovered with a single* quantum *chosen message query with overwhelming probability. In particular, the signatures are not even one-time EUF-qCMA secure.*

Proof. We first prove that $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ is EUF-CMA secure against classical queries. Suppose we have an adversary $\mathcal{A}$ for $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ with advantage ϵ. We define $k+1$ hybrid experiments:

Game j**.** In **Game** j, signature queries are answered as in the standard game, except that PRF_0 is used for the first j of the y_i. That is, to sign a message m, the challenger sets $y_i = \begin{cases} \mathsf{PRF}_0(k_i, m_i) & \text{If } i \leq j \\ \mathsf{PRF}_{\mathsf{sk}_i}(k_i, m_i) & \text{If } i > j \end{cases}$.

Game 0 is the normal EUF-CMA security game. We wish to show that **Game 0** is indistinguishable from **Game** t. We do this by showing **Game** $j-1$ is indistinguishable from **Game** j for all $j \in [t]$. We do this through an intermediate hybrid, where $y_j = O(m)$ for a random oracle O, which is simulated on the fly. The standard security of PRF_0 or the quantum-gap security of PRF_1 (which in particular implies standard security) mean that this change is undetectable. Then we can move to $y_j = \mathsf{PRF}_0(k_j, m)$ using the standard security of PRF_0.

Thus $\mathcal{A}$ has non-neligible advantage in **Game** t. We can use $\mathcal{A}$ to build a forger $\mathcal{B}$ for $(\mathsf{Gen}_c, \mathsf{Sig}_c, \mathsf{Ver}_c)$. On input vk_c, $\mathcal{B}$ chooses t keys $k_1, \ldots, k_1$, and simulates $\mathcal{A}$ on input vk_c. When $\mathcal{A}$ makes a signing query on m, $\mathcal{B}$ computes the y_i as in **Game** t, and makes a signing query on $(m, y_1, \ldots, y_t)$, obtaining a signature σ_c. It then gives the signature $(\sigma_c, y_1, \ldots, y_t)$ to $\mathcal{A}$. When $\mathcal{A}$ produces a message/forgery pair $(m^*, (\sigma^*, y_1^*, \ldots, y_t^*))$ for Sig, $\mathcal{B}$ outputs the message/forgery pair $((m^*, y_1^*, \ldots, y_t^*), \sigma^*)$ for Sig_c. If $\mathcal{A}$'s forgery is new and valid, so will $\mathcal{B}$'s. Moreover, $\mathcal{B}$ simualtes the view of $\mathcal{A}$ in **Game** t, and therefore has non-negligible advantage in breaking the EUF-CMA security of Sig_c.

We now give a quantum attack on Sig. From the signing oracle, we can easily implement the oracle that maps $(m_1, \ldots, m_t) \longmapsto (\mathsf{PRF}_{\mathsf{sk}_1}(k_1, m_1), \mathsf{PRF}_{\mathsf{sk}_2}(k_2, m_2), \ldots, (\mathsf{PRF}_{\mathsf{sk}_t}(k_t, m_t)$ by ignoring the σ part[3]. We can then treat this as t individual oracles mapping $m \longmapsto \mathsf{PRF}_{\mathsf{sk}_i}(k_i, m)$. Since

[3]Technically, we cannot just ignore the σ part as quantum computation needs to be reversible. However, we can

PRF_0 is quantum secure, oracle access to PRF_0 is indistinguishable from a random oracle. Since PRF_1 is a quantum-gap PRF, a single quantum query can distinguish it from a random oracle, and hence from PRF_0. Therefore,we can apply the quantum gap attack on PRF_1 to each of the oracles in parallel, using only a single quantum query. The result is that we recover all of the sk_i, and hence sk_c. Now we can sign any message. □

6.3.2 Construction 1: A conversion using Chameleon Hash Functions

Now we move to actually building signature schemes that are secure against quantum chosen message attacks. In this section, we show a general transformation from classically secure signatures to quantum secure signatures. The building blocks for our construction are chameleon hash functions and signatures that are secure against a classical random message attack.

The idea behind our construction is to first hash the message with the chameleon hash function and then sign the hash. In order to be secure against quantum queries, care has to be taken in how the randomness for the hash and the signature scheme is generated. In what follows, for any randomized algorithm A, we let $A(x;r)$ denote running A on input x with randomness r.

Construction 6.35. Let $(\mathsf{Gen}_H, \mathsf{H}, \mathsf{Inv})$ be a chameleon hash function, and $(\mathsf{Gen}_c, \mathsf{Sig}_c, \mathsf{Ver}_c)$ a signature scheme. Let $\mathcal{Q}$ and $\mathcal{R}$ be families of pairwise independent functions mapping messages to randomness used by H and Sig_c, respectively. We define a new signature scheme $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ where:

$$\begin{aligned}
\mathsf{Gen}(\lambda) &: (\mathsf{ik}, \mathsf{fk}) \xleftarrow{R} \mathsf{Gen}_H(\lambda),\ (\mathsf{sk}_c, \mathsf{vk}_c) \xleftarrow{R} \mathsf{Gen}_c(\lambda) \\
&\quad \text{output } \mathsf{sk} = (\mathsf{fk}, \mathsf{sk}_c),\ \mathsf{vk} = (\mathsf{fk}, \mathsf{vk}_c) \\
\mathsf{Sig}((\mathsf{fk}, \mathsf{sk}_c), m) &: Q \xleftarrow{R} \mathcal{Q}, R \xleftarrow{R} \mathcal{R} \\
&\quad r \xleftarrow{R} (m),\ s \leftarrow Q(m),\ h \leftarrow \mathsf{H}(\mathsf{fk}, m, r) \\
&\quad \sigma \leftarrow \mathsf{Sig}(\mathsf{sk}_c, h; s),\ \text{output } (r, \sigma) \\
\mathsf{Ver}((\mathsf{fk}, \mathsf{vk}_c), m, (r, \sigma)) &: h \leftarrow \mathsf{H}(\mathsf{fk}, m, r),\ \text{output } \mathsf{Ver}(\mathsf{vk}_c, h, \sigma)
\end{aligned}$$

We note that the chameleon secret key is not used in Construction 6.35, though it will be used in the security proof. Classically, this method of hashing with a chameleon hash and then signing converts any non-adaptively secure scheme into an adaptive one. We show that the resulting scheme is actually secure against an adaptive *quantum* chosen message attack.

Theorem 6.36. *If* $(\mathsf{Gen}_c, \mathsf{Sig}_c, \mathsf{Ver}_c)$ *is weakly (resp. strongly) EUF-RMA secure and* $(\mathsf{Gen}_H, \mathsf{H}, \mathsf{Inv})$ *is a secure chameleon hash function, then* $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ *in Construction 6.35 is weakly (resp.*

initialize the response register where σ is written to be a uniform superposition over all stings. Then writing σ to the register just permutes the strings, which leaves the state unchanged. Then the state can be discarded

strongly) EUF-qCMA secure. Moreover, if $(\mathsf{Gen}_c, \mathsf{Sig}_c, \mathsf{Ver}_c)$ *is only one-time secure, then* $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ *is also one-time secure.*

Theorem 6.36 shows that we can take a classically EUF-RMA secure signature scheme, combine it with a a chameleon hash, and obtain a quantum-secure signature scheme. In particular, the following constructions will be quantum secure, assuming SIS is hard:

- A slight modification to the signature scheme of Cash et al. [CHKP10], which combines their chameleon hash function with an EUF-RMA secure signature scheme. The only difference in their scheme is that the values r and s are sampled directly, rather than setting them to be the outputs of pairwise independent functions.
- A modification of the signature scheme of Agrawal, Boneh, and Boyen [ABB10a], where we hash the message using a chameleon hash before applying the signature.

We now prove Theorem 6.36:

Proof. We first sketch the proof idea. Given an Sig_c signature σ on a random hash h, we can construct an Sig signature on any given message m: use the chameleon secret key ik to compute a randomness r such that $\mathsf{H}(\mathsf{fk}, m, r) = h$, and output the signature (r, σ). Thus, we can respond to a classical chosen message attack, given only signatures on random messages.

If the adversary issues a *quantum* chosen message query, we need to sign each of the exponentially many messages in the query superposition. Therefore, using the above technique directly would require signing an exponential number of random hashes. Instead, we use small-range distributions and Corollary 4.15 to reduce the number of signed hashes required to a polynomial. The problem is that the number of hashes signed is still a very large polynomial, whereas the number of signatures produced by our adversary is only $q+1$, so we cannot rely on the pigeon-hole principle to argue that one of the Sig forgeries is in fact a Sig_c forgery. We can, however, argue that two of the forgeries must, in some sense, correspond to the same query. If we knew which query, we could perform a measurement, observing which of the (polynomially many) random hashes were signed. Lemma 4.4 shows that the adversary's advantage is reduced by only a polynomial factor. For this query, we now only sign a single random hash, but the adversary produces two forgeries. Therefore, one of these forgeries must be a forgery for Sig_c. Of course, we cannot tell ahead of time which query to measure, so we just pick the query at random, and succeed with probability $1/q$.

We now give the complete proof. There are four variants to the theorem (one-time vs many time, strong vs weak). We will prove the many-time strong security variant, the other proofs being similar. Let $\mathcal{A}$ be an adversary breaking the EUF-qCMA security of Sig in Construction 6.35 with non-negligible probability ϵ. We prove security through a sequence of games.

Game 0. This is the standard attack experiment, where A receives vk_c and fk, and is allowed to make a polynomial number of quantum chosen message queries. For query i, the challenger

produces pairwise independent functions $R^{(i)}$ and $Q^{(i)}$, and responds to each message in the query superposition as follows:

- Let $r_m^{(i)} = R^{(i)}(m)$ and $s_m^{(i)} = Q^{(i)}(m)$.
- Compute $h_m^{(i)} = \mathsf{H}(\mathsf{fk}, m, r_m^{(i)})$
- Compute $\sigma_m^{(i)} = \mathsf{Sig}_c(\mathsf{sk}_c, h_m^{(i)}; s_m^{(i)})$
- Respond with the signature $(r_m^{(i)}, \sigma_m^{(i)})$.

In the end, $\mathcal{A}$ must produce $q+1$ distinct triples $(m_k^*, r_k^*, \sigma_k^*)$ such that $\mathsf{Ver}(\mathsf{vk}_c, \mathsf{H}(\mathsf{fk}, m_k^*, r_k^*), \sigma_k^*)$ accepts. By definition, $\mathcal{A}$ wins with probability ϵ, which is non-negligible. Thus we can take $1/\epsilon$ to be a polynomial infinitely often.

Game 1. We make two modifications: first, we choose $R^{(i)}$ and $Q^{(i)}$ as truly random functions, which amounts to generating $r_m^{(i)}$ and $s_m^{(i)}$ at random for each i, m. According to Lemma 4.1, the view of the adversary is unchanged. Second, we modify the conditions in which $\mathcal{A}$ wins by requiring that no two (m_k^*, r_k^*) pairs form a collision for H. The security of $\mathcal{H}$ implies that $\mathcal{A}$ succeeds in Game 1 with probability at least $\epsilon - \mathsf{negl}$.

Game 2. Generate $s_m^{(i)}$ as before, but now draw $h_m^{(i)}$ uniformly at random. Additionally, draw uniform randomness $t_m^{(i)}$ for the Inv algorithm. We will sample $r_m^{(i)}$ from the set of randomness that makes $\mathsf{H}(\mathsf{fk}, m, r_m^{(i)}) = h_m^{(i)}$. That is, let $r_m^{(i)} = \mathsf{Inv}(\mathsf{ik}, h_m^{(i)}, m; t_m^{(i)})$. The properties of the chameleon hash ensure that the view of $\mathcal{A}$ is unchanged. Therefore, the success probability is at least $\epsilon - \mathsf{negl}$.

Game 3. Let $w = 2\ell(1)q/\epsilon$ where $\ell(\cdot)$ is the polynomial from Corollary 4.15. Then w is a polynomial infinitely often. At the beginning of the game, for $i = 1, ..., q$ and $j = 1, ..., w$, sample values $\hat{h}_j^{(i)}$ and let $\hat{\sigma}_j^{(i)} = \mathsf{Sig}_c(\mathsf{sk}_c, \hat{h}_j^{(i)})$. Also pick q random functions O_i mapping m to $[w]$. Then let $h_m^{(i)} = \hat{h}_{O_i(m)}^{(i)}$ and $\sigma_m^{(i)} = \hat{\sigma}_{O_i(m)}^{(i)}$. Let T_i be random functions, and let $t_m^{(i)} = T_i(m)$. The only difference between **Game 2** and **Game 3** is that the $h_m^{(i)}$ and $\sigma_m^{(i)}$ values were generated by q small-range distributions on w samples. Each of the small-range distributions is only queried once, so Corollary 4.15 implies that the success probability is only affected by $\epsilon/2q$ for each query. Thus, the success probability is still at least $\epsilon/2 - \mathsf{negl}$.

Game 4. Let the O_i and T_i be pairwise independent functions. The adversary cannot tell the difference.

Notice that Game 4 can now be simulated efficiently, and $\mathcal{A}$ wins in this game with probability $\epsilon/2 - \mathsf{negl}$. Let $h_k^* = \mathsf{H}(\mathsf{fk}, m_k^*, r_k^*)$ be the hashes of the forgeries. Since we have no collisions in H, the pairs (h_k^*, σ_k^*) are distinct. Let $\mathcal{H}^{(i)} = \{\hat{h}_j^{(i)}\}$ be the set of $\hat{h}$ values used to answer query i, and $\mathcal{H}$ be the union of the $\mathcal{H}^{(i)}$. There are two possibilities:

- At least one of the h_k^* is not in $\mathcal{H}$, or two of them are equal. In this case, we can obtain a forger $\mathcal{B}_0$ for Sig_c, which is given vk_c and simulates **Game 4** exactly: To generate the $(\hat{h}_j^{(i)}, \hat{\sigma}_j^{(i)})$ pairs, $\mathcal{B}_0$ asks its own challenger for signatures on $q\ell$ random messages. When A responds with forgeries $(m_k^*, r_k^*, \sigma_k^*)$, B_0 computes $h_k^* = \mathsf{H}(\mathsf{fk}, m_k^*, r_k^*)$, and finds the k value such that $h_k^* \notin \mathcal{H}$, or the k_0, k_1 such that $h_{k_0}^* = h_{k_1}^*$. In the latter case, one of the $\sigma_{k_b}^*$ was not the result of a signing query, so let $k = k_b$. It then outputs the pair (h_k^*, σ_k^*). Then $\mathcal{B}_0$ never received the signature σ_k^* on h_k^*, so this is a valid forgery. Therefore, this event happens with negligible probability.
- All of the h_k^* values are distinct and lie in $\mathcal{H}$. In this case, there is some i such that two h_k^* values are in $\mathcal{H}^{(i)}$ for the same i. Notice that this event happens, and all the forgeries are valid, with probability $\epsilon/2 - \mathsf{negl}$.

Game 5. Now we guess a random query i^* and add a check that all the h_k^* values lie in $\mathcal{H}$, and that two of them are distinct and lie in $\mathcal{H}^{(i^*)}$. Without loss of generality, assume these two h^* values are h_0^* and h_1^*. $\mathcal{A}$ then wins in this game with probability $\epsilon/2q - \mathsf{negl}$. Let j_b^* be the j such that $h_b^* = \hat{h}_{j_b^*}^{(i^*)}$ for $b = 0, 1$.

Game 6. On query i^*, measure the value of $O_i(m)$, to get a value j^*. O_i takes values in $[w]$, so Lemma 4.4 says the adversary's success probability is still at least $\epsilon/2qw - \mathsf{negl}$. Notice now that for query i^*, the challenger only needs to sign $\hat{h}_{j^*}^{(i^*)}$, and therefore, one of the $h_b^* = \hat{h}_j^{(i^*)}$ values was never signed.

Game 7. Now guess at the beginning of the game the value of j^*, and at the end, check that the guess was correct. The adversary still wins with probability $\epsilon/2qw^2 - \mathsf{negl}$.

We now describe an adversary $\mathcal{B}_1$ that breaks the security of Sig_c. Ask the RMA challenger for $(q-1)w+1$ random messages and corresponding signatures. For $j \neq j^*$, choose $\hat{h}_j^{(i^*)}$ randomly. Set the rest of the $\hat{h}_j^{(i)}$ values to be the signed messages, and set $\hat{\sigma}_j^{(i)}$ to be the corresponding signatures. Now play the role of challenger to $\mathcal{A}$ in **Game 7** using these values for $\hat{h}_j^{(i)}$ and $\hat{\sigma}^{(i)}$. As discussed above, $\mathcal{B}_1$ will never have to sign a message it does not have a signature for. Now if $\mathcal{A}$ wins, it means that it produced an
sig_c signature for some $\hat{h}_j^{(i^*)}$ with $j \neq j^*$. Since $\mathcal{B}_1$ never saw a signature on $\hat{h}_j^{(i^*)}$, this is a valid forgery. Therefore, $\mathcal{B}_1$ breaks the security of Sig_c with non-negligible probability $\epsilon/2qw^2 - \mathsf{negl}$. □

Remark 6.37. We note that for one-time security, this security reduction signs only a single message, so we only need to rely on the one-time security of Sig_c.

Remark 6.38. We note that the chameleon hash function built by Cash et al. [CHKP10] departs from the ideal notion described in Section 2.2. Similar to the proofs of Theorems 5.7 and 5.19, this

results $\mathcal{A}$ no longer having identical views in **Games 1** and **2**. Instead, the distributions on oracle outputs seen in **Games 1** and **2** are statistically close. We can then invoke Theorem 4.18 to argue that the two views are still indistinguishable.

6.3.3 Construction 2: Quantum Random Oracle Model Conversion

In this section we present a generic conversion from any classical signature scheme to a scheme secure against quantum chosen message attacks in the quantum random oracle model.

We demonstrate a simple generic conversion from a classical signature scheme to one that is secure against an adaptive *quantum* chosen message attack in the quantum random oracle model. The construction is quite simple: use the random oracle to hash the message along with a random salt, and send the signature on the hash, together with the salt. This construction is very appealing since messages are often hashed anyway before signing. The results in this section then show that only minor modifications to existing schemes are necessary to make them quantum immune.

Construction 6.39. Let $(\mathsf{Gen}_c, \mathsf{Sig}_c, \mathsf{Ver}_c)$ be a signature scheme, H be a random function, and $\mathcal{Q}$ be a family of pairwise independent functions mapping messages to the randomness used by Sig_c, and k some polynomial in λ. Define $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ where:

$$\begin{aligned}
\mathsf{Gen}(\lambda) &= \mathsf{Gen}_c(\lambda) \\
\mathsf{Sig}(\mathsf{sk}, m) &: Q \xleftarrow{R} \mathcal{Q},\ r \xleftarrow{R} \{0,1\}^k \\
&\qquad s \leftarrow Q(m),\ h \leftarrow H(m,r),\ \sigma \leftarrow \mathsf{Sig}_c(\mathsf{sk}, h; s),\ \text{output } (r, \sigma) \\
\mathsf{Ver}(\mathsf{vk}, m, (r, \sigma)) &: h \leftarrow H(m,r),\ \text{output } \mathsf{Ver}_c(\mathsf{vk}, h, \sigma)
\end{aligned}$$

We note that Construction 6.39 is similar to Construction 6.35: instead of the chameleon hash $\mathsf{H}(\mathsf{fk}, \cdot, \cdot)$ we have a random oracle $H(\cdot, \cdot)$, and instead of generating a different r for each message in the superposition, we just generate a single r for the entire superposition. We can achieve security for Construction 6.39, assuming only a very weak form of security for Sig_c, namely, universal unforgeability under a random message attack (UUF-RMA security):

Theorem 6.40. *If* $(\mathsf{Gen}_c, \mathsf{Sig}_c, \mathsf{Ver}_c)$ *is strongly (resp. weakly) UUF-RMA secure, then the scheme* $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ *in Construction 6.39 is strongly (resp. weakly) EUF-qCMA secure in the quantum random oracle model. Moreover, if* $(\mathsf{Gen}_c, \mathsf{Sig}_c, \mathsf{Ver}_c)$ *is only one-time secure, then* $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ *is also one-time secure.*

Before proving Theorem 6.40, we explain how to realize the strong UUF-RMA notion of security. We note that any strongly EUF-RMA or EUF-CMA secure signature scheme satisfies this security notion. We also note that some weaker primitives do as well. The first is pre-image sampleable functions (PSFs). One-wayness plus collision resistance for PSFs implies strong UUF-RMA security.

Corollary 6.41. *If* PSF *is a collision resistant and one-way PSF, then Construction 6.39 instantiated with* PSF *is strongly EUF-qCMA secure in the quantum random oracle model.*

The result is a slight variant of the GPV signature scheme of Gentry, Peikert, and Vaikuntanathan [GPV08], who construct PSFs based on the hardness of SIS. Therefore, we can construct efficient quantum-secure signatures in the quantum random oracle model based on SIS. Later, we also show that the basic GPV signature scheme is secure in the quantum random oracle model, though the proof is very different.

We note also that PSFs give UUF-RMA signatures, even if they do not satisfy the pre-image minentropy requirement. Thus, trapdoor permutaitons suffice:

Corollary 6.42. *If* $\mathcal{F}$ *is a one-way trapdoor permutation, then Construction 6.39 instantiated with* $\mathcal{F}$ *is strongly EUF-qCMA secure in the quantum random oracle model.*

Next, we observe that any adversary $\mathcal{A}$ breaking the universal unforgeability of Sig_c by mounting a random message attack can easily be transformed into an adversary $\mathcal{B}$ breaking Construction 6.39 under a *classical* chosen message attack in the *classical* random oracle model:

- When $\mathcal{B}$ receives the public key vk for Sig in Consutrction 6.39, $\mathcal{B}$ forwards the public key to $\mathcal{A}$.
- $\mathcal{A}$ requests q message/signature pairs for random messages, and n additional random messages. To respond, $\mathcal{B}$ queries its signing oracle on q arbitrary distinct points m_i, obtaining q pairs (r_i, σ_i), where σ_i is a valid Sig_c signature of $h_i = H(m_i, r_i)$. $\mathcal{B}$ queries its random oracle on m_i, r_i to obtain h_i, and sends the q pairs (h_i, σ_i) as the message/tag pairs to $\mathcal{A}$. Additionally, $\mathcal{B}$ queries its random oracle on n additional arbitrary points m_i^*, r_i^*, obtaining h_i^*, and sends the h_i^* to $\mathcal{A}$ as the n additional messages.
- Finally, $\mathcal{A}$ outputs a new signature σ_i^* for one on the messages h_i^*, or potentially one of the h_i if we are interested in strong security. $\mathcal{B}$ simply figures out which pre-image (m_i^*, r_i^*) this forgery corresponds to, and outputs the tuple (m_i^*, r_i^*, σ^*).

Together with Theorem 6.40, this roughly means that quantum chosen message queries and quantum random oracle queries do not help the adversary break Construction 6.39. Therefore, if a scheme matches the form of Construction 6.39, it is only necessary to prove classical security. This is formalized by the following corollary:

Corollary 6.43. *If* $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ *in Construction 6.39 is weakly (resp. strongly) existentially unforgeable under a* classical *chosen message attack performed by a quantum adversary, then it is also weakly (resp. strongly) existentially unforgeable under a* quantum *chosen message attack.*

We now proof Theorem 6.40, which is very similar to the proof of Theorem 6.36

Proof. Suppose we have an adversary $\mathcal{A}$ that breaks the security of $\mathcal{S}$ from Construction 6.39. Let q_S be the number of signing queries made by $\mathcal{A}$, and q_H be the number of hash queries (including those used in signing). We will prove security through a sequence of games.

Game 0. This is the standard attack game. $\mathcal{A}$ makes q_S quantum chosen message queries, and succeeds if it produces q_S+1 valid message/signature tuples $(m_j^*, r_j^*, \sigma_j^*)$. Let r_i be the random value produced in the ith query, and Q_i be the pair-wise independent functions. $\mathcal{A}$'s success probability is, by assumption, some non-negligible quantity ϵ. Thus, we can take $1/\epsilon$ to be a polynomial infinitely often.

Game 1. Replace the R_i with a truly random function, and abort if any of the r_i values are identical. Then the success probability is at least $\epsilon - q^2/2^{k+1} \geq \epsilon - \mathsf{negl}$. Notice that if all the r_i are distinct, we can replace $Q_i(m)$ with $Q(m, r_i)$ for a random oracle Q that is fixed across all queries. That is, we sign the ith query with the oracle that maps m to $(r_i, \mathsf{Sig}_c(\mathsf{sk}, H(m, r_i); Q(m, r_i)))$. Notice that the function $H'(m, r) = (H(m, r), Q(m, r))$ is a random function.

Game 2. Let $w = 6\ell(q_H)/\epsilon$. Since ϵ is non-negligible, w is a polynomial infinitely often. We now change how H' is generated. Pick three random oracles U, V and W, where the codomain of U and V is $[w]$, and let $H'(m, r) = W(U(m, V(r)), V(r))$. What this distribution represents is, for each $V(r)$ value, picking a random small-range function on w samples. In essence, we have a small-range distribution on small-range distributions. A simple generalization of Corollary 4.15 shows that this is indistinguishable from **Game 1** except with probability $\epsilon/3$.

Game 3. Pick the r_i values up front, and let $\mathcal{R}$ be the set of r_i values. Abort if $V(r_i) = V(r_j)$ for any $i \neq j$. We can assume without loss of generality that $V(r_i) = i$. The probability of this abort is at most $q_S^2/2w$. We can assume that $q_S \leq q_H$. Then $q_S^2 \leq \ell(q_H)$, and so the abort probability is at most $\epsilon/3$. Therefore, $\mathcal{A}$ wins in Game 3 with probability at least $\epsilon/3$.

The following modifications are indistinguishable to the adversary: before the start of the game, draw w^2 different $\hat{h}_{i,j}$ values. Sign each of them with $i \leq q_S$ using $\mathcal{S}$ to get $\hat{\sigma}_{i,j}$. Then let $H(m, r) = \hat{h}_{V(r),U(m,V(r))}$ and sign the ith query my mapping m to $(r_i, \hat{\sigma}_{i,U(m,i)})$. We can also generate V and U from $2q_H$-wise independent functions, and the adversary cannot tell.

Game 4. Pick a random r_{i_0} from $\mathcal{R}$. Add the condition that if the r_j^* all lie in $\mathcal{R}$, that the two that are equal must be r_{i_0}. This condition is independent of the view of the adversary, so the adversary wins with probability at least $(\epsilon/3)/q_S$.

Game 5. Measure the value of $U(m, i_0)$ for the i_0th query. The adversary still wins with probability at least $(\epsilon/3)/q_S w$.

Game 6. Pick a random $j_0 \in [w]$, and abort if the result of the measurement in Game 5 does not yield j_0. We guess right with probability $1/w$, so the adversary still wins with probability at least $(\epsilon/3)/q_S w^2$. Now, if we succeed, we never need the values of $\hat{\sigma}_{i_0,j}$ except for $\hat{\sigma}_{i_0,j_0}$, so we don't need to ever sign the others.

We can now describe an adversary $\mathcal{B}$ that attacks the UUF-RMA security of Sig_c. $\mathcal{B}$ simulates the entire **Game 6**, except for generating the $\hat{h}_{i,j}$ and $\hat{\sigma}_{i,j}$. For these, $\mathcal{B}$ asks its Sig_c challenger for $q = (q_S - 1)w + 1$ random message/signature pairs, and $n = w^2 - q$ additional random messages. It assigns the q message/signature pairs to $\hat{h}_{i,j}$ and $\hat{\sigma}_{i,j}$ for $i \in [q_S] \setminus \{i_0\}$ and $\hat{\sigma}_{i_0,j_0}$. The rest of the $\hat{h}_{i,j}$ it sets to the n additional messages. When $\mathcal{A}$ outputs its $q_S + 1$ forgery candidates, there are several possibilities:

- $r^*_{j_1}$ lies outside $\mathcal{R}$ for some j_1. In this case, since there are no collisions among the (m^*_j, r^*_j), $h^*_{j_1} = H(m^*_{j_1}, r^*_{j_1})$ was never signed. Therefore, $\sigma^*_{j_1}$ is a signature on a fresh message, so $\mathcal{B}$ wins.
- All of the r^*_j lie in $\mathcal{R}$, and two of them are equal. Assume without loss of generality that $r^*_0 = r^*_1 = r_{i_0}$. If $m^*_0 = m^*_1$, then we must have $\sigma^*_0 \neq \sigma^*_1$, so one of these is a fresh signature. If $m^*_0 \neq m^*_1$, then $h^*_0 \neq h^*_1$, so one of h^*_0 and h^*_1 was never signed. Therefore, $\mathcal{B}$ also wins.

Therefore, $\mathcal{B}$ wins with non-negligible probability $(\epsilon/3)/q_S w^2$. □

Remark 6.44. We note that, for one-time security, $q = 1$, so we only need to rely on the one-time security of $\mathcal{S}_c$.

Deterministic GPV Signatures

Now we show that the basic (deterministic) signature scheme of Gentry, Peikert, and Vaikuntanathan [GPV08] (the GPV signature scheme) is secure in the quantum random oracle model. For convenience, we re-state the GPV signature scheme:

Construction 5.6. Let $\mathsf{PSF} = (\mathsf{Gen}_{psf}, \mathsf{Sample}, F, F^{-1})$ be a pre-image sampleable function, PRF be a pseudorandom function, and H a hash function. Let $\mathcal{S} = (\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ where

$$\mathsf{Gen}(\lambda) : (\mathsf{ik}, \mathsf{fk}) \xleftarrow{R} \mathsf{Gen}_{psf}(\lambda),\ k \xleftarrow{R} \{0,1\}^\lambda$$
$$\text{output } \mathsf{sk} = (\mathsf{ik}, k), \mathsf{vk} = \mathsf{fk}$$

$$\mathsf{Sig}((\mathsf{ik}, k), m) : r \leftarrow \mathsf{PRF}(k, m)\ h \leftarrow H(m),\ \text{output } \sigma = F^{-1}(\mathsf{ik}, h; r)$$

$$\mathsf{Ver}(\mathsf{fk}, m, \sigma) : h \leftarrow H(m),\ h' \leftarrow F(\mathsf{fk}, \sigma),\ \text{accept if and only if } h = h'$$

We say that PSF has large pre-image min-entropy if, for all fk,

$$\max_{y \in \mathcal{Y}} \Pr[x \leftarrow \mathsf{Sample}(\lambda) : F(\mathsf{fk}, x) = y] < 2^{-\omega(\log \lambda)}$$

We note that the PSF given by Gentry et al. [GPV08] has large pre-image min-entropy.

Theorem 6.45. *If* PSF *is collision resistant and has large pre-image min-entropy, then* $\mathcal{S}$ *from Construction 5.6 is EUF-qCMA secure.*

Proof. We prove security via a sequence of games:

Game 0. This is the standard security game. The adversary wins with probability ϵ.

Game 1. Replace PRF with a truly random function. The security of PRF implies that the adversary wins with probability at least $\epsilon - \mathsf{negl}$.

Game2. We change the way we answer signing queries and oracle queries as follows: Pick a random function J that maps messages to the randomness used by $\mathsf{Sample}(\lambda)$. We implement the signing oracle as $S(m) = \mathsf{Sample}(\lambda; J(m))$. That is, signatures are random samples from D_λ, where the randomness used in the sampling is obtained by $J(m)$. We implement the random oracle as $H(m) = F(\mathsf{fk}, S(m))$. The adversary wins if he can produce $q+1$ (m_i, σ_i) pairs where $H(m_i) = F(\mathsf{fk}, \sigma_i)$. This corresponds to $F(\mathsf{fk}, S(m_i)) = F(\mathsf{fk}, \sigma_i)$. In other words, $S(m_i)$ and σ_i form a collision. By the collision resistance of PSF, we must have $S(m_i) = \sigma_i$ for all i, except with negligible probability. This means that we make q queries to the oracle S and a polynomial number of queries to the oracle $F(\mathsf{fk}, S(\cdot))$, and output $q+1$ input/output pairs of S with probability $\epsilon - \mathsf{negl}$.

Even if the adversary is able to completely learn the oracle $H(\cdot) = F(\mathsf{fk}, S(\cdot))$, the oracle $S(\cdot)$ is unpredictable to the adversary. In particular, $S(m)$ is a random pre-image of $H(m)$, which has minentropy at least $H_\infty = \omega(\log \lambda)$. Therefore, Game 2 satisfies the conditions of Theorem 3.8, meaning the probability A wins in Game 2 is at most $(q+1)/\lfloor 2^{H_\infty} \rfloor < (q+1)2^{-\omega(\log \lambda)}$, which is negligible. Therefore, A wins in Game 0 with negligible probability, as desired. □

Remark 6.46. Again, we need to invoke Theorem 4.18 to adopt the above proof to handle non-ideal PSFs.

6.3.4 Generic Constructions of Digital Signatures

In this section, we show how to construct signatures from generic assumptions. We first construct one-time signatures from one-way functions using the basic Lamport construction [Lam79]. We then show that by combining these signatures with standard-secure signatures in an online/offline fashion [EGM90], we obtain quantum many-time signatures. As standard-secre signatures can be built from any one-way function, the result is quantum many-time signatures from any one-way function[4].

[4]In *Secure Signatures and Chosen Ciphertext Security*, we showed how to obtain many-time signatures using collision resistance, a stronger assumption that one-wayness. We would like to thank Andreas Hülsing for discussions that helped realize the stronger version of the theorem.

Theorem 6.47. *If there exists a one-way function, then there exists a strongly EUF-qCMA secure signature scheme.*

Lamport Signatures. We now give the basic Lamport scheme [Lam79] and prove its security:

Construction 6.48. Let F be a one-way function. We define the following signature scheme for n-bit messages:

$$\begin{aligned}
\mathsf{Gen}(\lambda) &: \text{for each } i \in [n], b \in \{0,1\} : x_{i,b} \xleftarrow{R} \{0,1\}^\lambda,\ y_{i,b} \xleftarrow{R} F(x_i) \\
&\quad \text{output } \mathsf{sk} = (x_{i,b})_{i\in[n],b\in\{0,1\}}, \mathsf{vk} = (y_{i,b})_{i\in[n],b\in\{0,1\}} \\
\mathsf{Sig}(\mathsf{sk}, m) &: \text{write } \mathsf{sk} = (x_{i,b})_{i\in[n],b\in\{0,1\}} \\
&\quad \text{output } (x_{i,m_i})_{i\in[n]} \\
\mathsf{Ver}(\mathsf{vk}, m, \sigma) &: \text{write } \mathsf{vk} = (y_{i,b})_{i\in[n],b\in\{0,1\}}, \sigma = (x'_i)_{i\in[n]} \\
&\quad \text{accept if and only if } F(x'_i) = y_{i,m_i} \text{ for all } i \in [n]
\end{aligned}$$

Theorem 6.49. *If F is one-way (resp. second pre-image resistant), then the Lamport signature scheme built from F is weakly (resp. strongly) one-time EUF-qCMA secure.*

Proof. We prove the weak security case; the strong security case is almost identical. Let A be an adversary that makes a single quantum query to Sig and outputs a pair of valid message/signature pairs for different messages with probability ϵ. We prove security through a sequence of games.

Game 0. This is the standard attack game, where A wins with probability ϵ.

Game 1. Pick a random value $i^* \in [n]$. Abort if both messages in A's forgery are the same for index i^*. A still wins with probability ϵ/n.

Game 2. For the quantum chosen message query, measure the bit i^* of the message superposition. Lemma 4.4 shows that A still wins with probability $\epsilon/2n$.

Game 3. At the beginning of the game, guess a bit b^* at random, and abort if the outcome of the measurement in Game 2 is b^*. A still wins with probability $\epsilon/4n$.

We can now describe an adversary B that inverts F. On input y, B guesses $i^* \in [n]$ and $b^* \in \{0,1\}$, and sets $y_{i^*,b^*} = y$. For $(i,b) \neq (i^*,b^*)$, B picks $x_{i,b}$ at random, and lets $y_{i,b} = F(x_{i,b})$. Now B simulates Game 3. With probability at least $\epsilon/4n$, B is able to answer A's query, and A produces valid forgeries whose messages differ on bit i^*. This means A produces pre-images $x'_{i^*,0}, x'_{i^*,1}$ for $y_{i^*,0}, y_{i^*,1}$. B outputs x'_{i^*,b^*}, which is a valid pre-image for $y_{i^*,b^*} = y$.

□

Obtaining many-time security. The natural approach to obtaining many-time security is to plug the construction above into the Merkle signature tree construction [Mer88]. To use in this fashion, the one-time signature must be able to sign messages twice as long as its own public key. Unfortunately, Lamport signatures necessarily can only sign messages much shorter than their own public key. In the classical setting, this problem can be rectified in two ways. The first way is to hash the message using a collision resistant hash function. While this solution is appealing due to its simplicity, collision resistance is a stronger primitive than one-wayness, so the resulting scheme requires stronger assumptions. The second option is to hash the messages with a *target collision resistant* (TGR) function. A TGR function is a keyed function $H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ with the following security property: any efficient quantum adversary $\mathcal{A}$ has negligible advantage in the following experiment. $\mathcal{A}$ first commits to arbitrary message $m_0 \in \mathcal{M}$. Then a random key $k \xleftarrow{R} \mathcal{K}$ is chosen and given to $\mathcal{A}$. Then $\mathcal{A}$ chooses an arbitrary second message $m_1 \in \mathcal{M}$. $\mathcal{A}$'s advantage is the probability that m_0, m_1 form a collision for H using key k: $H(k, m_0) = H(k, m_1)$.

Now, the modified one-time signature scheme is constructed as follows. To sign a message m, first, choose a random key $k \xleftarrow{R} \mathcal{K}$. Then compute $h \leftarrow H(k, m)$. Then construct the signature $\sigma \xleftarrow{R} \mathsf{Sig}(\mathsf{sk}, (k, h))$ by signing the pair (k, h). The overall signature is (k, σ). In the classical setting, TGR security enough to argue security. However, when quantum queries are allowed, the proof no longer works: it appears that the TGR security definition needs to be modified to allow $\mathcal{A}$ to commit to a superposition of messages. However, it is not clear how exactly to define TGR security in this setting to make the proof go through.

Online/offline signatures. We now explain our approach to obtaining many-time security from one-way functions. We will have two signature schemes: $(\mathsf{Gen}_{OT}, \mathsf{Sig}_{OT}, \mathsf{Ver}_{OT})$ will be a quantum one-time secure scheme, and $(\mathsf{Gen}_{MT}, \mathsf{Sig}_{MT}, \mathsf{Ver}_{MT})$ will be a (classical) many-time secure scheme. Then we construct $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver})$ following the online/offline signature protocol. That is, to sign a message m, a random signing/verification key $(\mathsf{sk}_{OT}, \mathsf{vk}_{OT})$ is generated for the one-time scheme, and sk_{OT} is used to sign m, obtaining the signature σ_{OT}. Then vk_{OT} is signed using the many-time schene, obtaining σ_{MT}. The full signature consists of $\mathsf{vk}_{OT}, \sigma_{OT}, \sigma_{MT}$. The point of using this signature scheme is that, when signing a superposition of messages, only a single one-time verification key vk_{OT} needs to be signed, not an entire superposition. Thus the many-time signature scheme only needs to be secure against classical signing queries. Such signature schemes can be build from one-way functions as in the classical setting. Only the one-time scheme is used to sign the actual superposition of messages, and therefore needs to be secure against quantum signing queries. Putting this together proves Theorem 6.47.

Construction 6.50. Let F be a one-way function. We define the following signature scheme for

n-bit messages:

$$\begin{aligned}
\mathsf{Gen}() &: (\mathsf{sk}_{MT}, \mathsf{vk}_{MT}) \xleftarrow{R} \mathsf{Gen}_{MT}() \\
&\quad \text{output } \mathsf{sk} = \mathsf{sk}_{MT}, \mathsf{vk} = \mathsf{vk}_{MT} \\
\mathsf{Sig}(\mathsf{sk}_{MT}, m) &: (\mathsf{sk}_{OT}, \mathsf{vk}_{OT}) \xleftarrow{R} \mathsf{Gen}_{OT}(), \sigma_{OT} \xleftarrow{R} \mathsf{Sig}_{OT}(\mathsf{sk}_{OT}, m), \sigma_{MT} \xleftarrow{R} \mathsf{Sig}_{MT}(\mathsf{sk}_{MT}, \mathsf{vk}_{OT}) \\
&\quad \text{output } \sigma = (\mathsf{vk}_{OT}, \sigma_{OT}, \sigma_{MT}) \\
\mathsf{Ver}(\mathsf{vk}_{MT}, m, \sigma) &: \text{write } \sigma = (\mathsf{vk}_{OT}, \sigma_{OT}, \sigma_{MT}) \\
&\quad \text{accept if both } \mathsf{Ver}_{MT}(\mathsf{vk}_{MT}, \mathsf{vk}_{OT}, \sigma_{MT}) \text{ and } \mathsf{Ver}_{OT}(\mathsf{vk}_{OT}, m, \sigma_{OT}) \text{ accept}
\end{aligned}$$

Theorem 6.51. *If* $(\mathsf{Gen}_{OT}, \mathsf{Sig}_{OT}, \mathsf{Ver}_{OT})$ *is strongly (resp. weakly) one-time EUF-qCMA secure and* $(\mathsf{Gen}_{MT}, \mathsf{Sig}_{MT}, \mathsf{Ver}_{MT})$ *is strongly (resp. weakly) many-time EUF-CMA secure, then the online/offline signature protocol in Construction 6.50 is strongly (resp. weakly) manytime EUF-qCMA secure.*

Proof. We prove the strong case, the weak case being similar. Let $\mathcal{A}$ be an adversary for the strong EUF-qCMA security of Construction 6.50 with non-negligible advantage ϵ. Let $\{(m_i^*, \sigma_i^*)\}_{i\in[q+1]}$ denote the $q+1$ message/signature pairs produced by $\mathcal{A}$, and let $\{(\mathsf{vk}_{j,OT}, \sigma_{j,MT})\}_{j\in[q]}$ denote the q one-time verification keys and corresponding signatures on the verification keys received during signing queries.

We define two quantities:

- ϵ_1: This is the probability that $\mathcal{A}$ produces $q+1$ valid distinct message/signature pairs $(m_i^*, \sigma_i^* = (\mathsf{vk}_{i,OT}^*, \sigma_{i,OT}^*, \sigma_{i,MT}^*))$, *and* among the $2q+1$ verification key/signature pairs $(\mathsf{vk}_{i,OT}^*, \sigma_{i,MT}^*)$ and $(\mathsf{vk}_{j,OT}, \sigma_{j,MT})$, at least $q+1$ of them are distinct. Then the $q+1$ distinct pairs are valid message/signature pairs for $(\mathsf{Gen}_{MT}, \mathsf{Sig}_{MT}, \mathsf{Ver}_{MT})$ relative to verification key vk_{MT}, and hence constitute forgeries. It is therefore straightforward to construct from this a strong EUF-CMA adversary for $(\mathsf{Gen}_{MT}, \mathsf{Sig}_{MT}, \mathsf{Ver}_{MT})$ with advantage ϵ_1. Therefore, ϵ_1 is negligible.

- ϵ_2: This is the probability that $\mathcal{A}$ produces $q+1$ valid distinct message/signature pairs $(m_i^*, \sigma_i^* = (\mathsf{vk}_{i,OT}^*, \sigma_{i,OT}^*, \sigma_{i,MT}^*))$, *and* among the $2q+1$ verification key/signature pairs $(\mathsf{vk}_{i,OT}^*, \sigma_{i,MT}^*)$ and $(\mathsf{vk}_{j,OT}, \sigma_{j,MT})$, at most q of them are distinct. In particular, this means two of the $(\mathsf{vk}_{i,OT}^*, \sigma_{i,MT}^*)$ are identical. That is, $(\mathsf{vk}_{i_0,OT}^*, \sigma_{i_0,MT}^*) = (\mathsf{vk}_{i_1,OT}^*, \sigma_{i_1,MT}^*)$ for distinct i_0, i_1. Moreover, each of the $(\mathsf{vk}_{i,OT}^*, \sigma_{i,MT}^*)$ is equal to one of the $(\mathsf{vk}_{j,OT}, \sigma_{j,MT})$, since the $(\mathsf{vk}_{j,OT}, \sigma_{j,MT})$ form q distinct pairs with overwhelming probability.

  Then since $\mathcal{A}$ produces distinct pairs, it must be that $(m_{i_0}, \sigma_{i_0,OT}) \neq (m_{i_1}, \sigma_{i_1,OT})$. Hence, $(m_{i_0}, \sigma_{i_0,OT}), (m_{i_1}, \sigma_{i_1,OT})$ constitute two valid message/signature pairs for $(\mathsf{Gen}_{OT}, \mathsf{Sig}_{OT}, \mathsf{Ver}_{OT})$ relative to verification key $\mathsf{vk}_{OT}^* = \mathsf{vk}_{i_0,OT}^* = \mathsf{vk}_{i_1,OT}^*$. It is straightforward to construct

from this a strong one-time EUF-qCMA adversary for $(\mathsf{Gen}_{OT}, \mathsf{Sig}_{OT}, \mathsf{Ver}_{OT})$ with advantage ϵ_2/q. The loss of a factor of q is from the adversary guessing which of the q queries made by $\mathcal{A}$ will use vk^*_{OT} (recall that each of the $\mathsf{vk}^*_{i,OT}$ must have come from the signing queries made by the adversary). Therefore, ϵ_2/q, and hence ϵ_2, must be negligible.

Thus $\epsilon = \epsilon_1 + \epsilon_2$ is negligible, showing that Construction 6.50 is strongly EUF-qCMA secure.

□

6.4 Quantum-secure Encryption

We now turn to encryption schemes where we first discuss an adequate notion of security under quantum queries. In what follows, we will discuss symmetric key schemes; the discussion for public key schemes is similar. At a high level, our notion of security allows quantum encryption and decryption queries, but requires challenge queries to be *classical*. One might hope for an entirely quantum game, where challenge queries are quantum as well, but we show that such fully-quantum security definitions are unsatisfiable.

We start by developing a notion of CPA security where encryption queries are allowed to be quantum. Since finding an attainable definition is non-trivial we first present a few alternatives and then converge to a workable definition (Definition 6.56). Once we arrive at a suitable definition for CPA security we will also obtain a corresponding definition for CCA security. Our first attempt at defining quantum CPA security is as follows:

Definition 6.52. A symmetric key encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$ is indistinguishable under a fully quantum chosen plaintext attack (IND-fqCPA secure) if no efficient adversary $\mathcal{A}$ can win the following game, except with probability at most $1/2 + \mathsf{negl}$:

Key Gen The challenger picks a random key k and a random bit b.

Encryption Queries $\mathcal{A}$ is allowed to make chosen message queries on superpositions of message pairs. For each such query, the challenger chooses randomness r, and encrypts the appropriate message in each pair using r as randomness:

$$\sum_{m_0,m_1,c} \psi_{m_0,m_1,c} \big| m_0,\ m_1,\ c \big\rangle \qquad \longrightarrow \qquad \sum_{m_0,m_1,c} \psi_{m_0,m_1,c} \big| m_0,\ m_1,\ c \oplus \mathsf{Enc}(k, m_b; r) \big\rangle$$

Guess $\mathcal{A}$ produces a bit b', and wins if $b = b'$.

Definition 6.52 captures a scheme where we can encrypt a superposition of messages by encrypting each message in the superposition separately, and no efficient adversary can learn anything about the plaintext superposition. Unfortunately, this definition is not achievable:

Theorem 6.53. *No encryption scheme* $(\mathsf{Enc}, \mathsf{Dec})$ *satisfies the security notion of Definition 6.52.*

Proof. We construct a generic adversary $\mathcal{A}$. $\mathcal{A}$ prepares three registers: two plaintext registers and a ciphertext register. $\mathcal{A}$ puts a uniform superposition of all messages in the first register, and 0 in the second plaintext and ciphertext registers. $\mathcal{A}$ submits these three registers as a chosen message query. If $b = 0$, the ciphertext register will contain the encryptions of the messages in the superposition. If $b = 1$, it will contain the encryption of 0. $\mathcal{A}$ then measures the ciphertext register. If $b = 0$, the resulting state will be the purely classical state $(m, 0, \mathsf{Enc}(k, m))$ for a random message m. If $b = 1$, the measurement does nothing, so the first register still contains a superposition of all messages. $\mathcal{A}$ now performs the quantum Fourier transform to the first message register and measures. If $b = 0$, the transform will place a uniform superposition of all messages in the first register, and measuring will give a random message. If $b = 1$, the transform will place 0 in the first register. Thus, $\mathcal{A}$ distinguishes $b = 0$ from $b = 1$ with probability exponentially-close to 1. □

The problem with Definition 6.52 is that the message query is entangled with the ciphertext response, and this entanglement depends on which register gets encrypted. Another reasonable idea is to encrypt *both* message registers, but flip which register each ciphertext is written to depending on the value of b:

Definition 6.54. A symmetric key encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$ is indistinguishable under a fully quantum chosen left-right plaintext attack (IND-lrCPA secure) if no efficient adversary $\mathcal{A}$ can win in the following game, except with probability at most $1/2 + \mathsf{negl}$:

Key Gen The challenger picks a random key k and a random bit b.

Encryption Queries $\mathcal{A}$ is allowed to make chosen message queries. For each such query, the challenger chooses randomness r_0, r_1, and responds with the encryptions of both messages in the pair, but in an order determined by b:

$$\sum_{m_0,m_1,c_1,c_2} \psi_{m_0,m_1,c_1,c_2} |m_0,\ m_1,\ c_1,\ c_2\rangle \qquad \longrightarrow$$
$$\sum_{m_0,m_1,c_1,c_2} \psi_{m_0,m_1,c_1,c_2} |m_0,\ m_1,\ c_1 \oplus \mathsf{Enc}(k, m_b; r_0),\ c_2 \oplus \mathsf{Enc}(k, m_{1-b}; r_1)\rangle$$

Guess $\mathcal{A}$ produces a bit b', and wins if $b = b'$.

Unfortunately, this definition turns out to be at least as strong as Definition 6.52, and so it is also unattainable:

Theorem 6.55. *No encryption scheme* $(\mathsf{Enc}, \mathsf{Dec})$ *satisfies the notion of security in Definition 6.54. In particular, any encryption scheme that is secure in the sense of Definition 6.54 is also secure in the sense of Definition 6.52.*

Proof. Suppose we have an adversary $\mathcal{A}$ for Definition 6.52. We will convert it into an adversary $\mathcal{B}$ for Definition 6.54. $\mathcal{B}$ simulates $\mathcal{A}$ forwarding encryption queries as follows: When $\mathcal{A}$ makes an

encryption query, $\mathcal{B}$ adds a second ciphertext register, and puts into it a uniform superposition over all ciphertexts. $\mathcal{B}$ then sends the resulting state to its challenger as its encryption query. The answer to this query does not affect the second ciphertext register, so $\mathcal{B}$ can uncompute it. $\mathcal{B}$ then passes the resulting state back to $\mathcal{A}$. $\mathcal{B}$ perfectly simulates $\mathcal{A}$'s view, and therefore $\mathcal{B}$ breaks the security of $(\mathsf{Enc}, \mathsf{Dec})$ under Definition 6.54. □

Our attempts to make the entire security game quantum lead to an adversary that can always win. Therefore, we must force encryption queries to be classical. We do, however, wish to allow the adversary to encrypt superpositions of messages, but not have the response depend in any way on b. Therefore, we propose separating encryption queries into classical challenge queries and quantum encryption queries. This gives the following definition:

Definition 6.56. A symmetric key encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$ is indistinguishable under a quantum chosen message attack (IND-qCPA secure) if no efficient adversary $\mathcal{A}$ can win in the following game, except with probability at most $1/2 + \mathsf{negl}$:

Key Gen The challenger picks a random key k and a random bit b.

Queries $\mathcal{A}$ is allowed to make two types of queries:

Challenge queries $\mathcal{A}$ sends two messages m_0, m_1, to which the challenger responds with $c^* = \mathsf{Enc}(k, m_b)$.

Encryption queries For each such query, the challenger chooses randomness r, and encrypts each message in the superposition using r as randomness:

$$\sum_{m,c} \psi_{m,c} \big| m,\ c \big\rangle \qquad \longrightarrow \qquad \sum_{m,c} \psi_{m,c} \big| m,\ c \oplus \mathsf{Enc}(k, m; r) \big\rangle$$

Guess $\mathcal{A}$ produces a bit b', and wins if $b = b'$.

This definition has another advantage: since challenge queries are classical, when we move to CCA security, we can check if a ciphertext was the result of a challenge query and reject decryption queries for these ciphertexts. This gives us the following notion of CCA security:

Definition 6.57. A symmetric key encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$ is indistinguishable under a quantum chosen message attack (IND-qCCA secure) if no efficient adversary $\mathcal{A}$ can win in the following game, except with probability at most $1/2 + \mathsf{negl}$:

Key Gen The challenger picks a random key k and a random bit b. It also creates a list $\mathcal{C}$ which will store challenger ciphertexts.

Queries $\mathcal{A}$ is allowed to make three types of queries:

Challenge queries $\mathcal{A}$ sends two messages m_0, m_1, to which the challenger responds with $c^* = \mathsf{Enc}(k, m_b)$. The challenger also adds c^* to $\mathcal{C}$.

Encryption queries For each such query, the challenger chooses randomness r, and encrypts each message in the superposition using r as randomness:

$$\sum_{m,c} \psi_{m,c} |m,\ c\rangle \qquad \longrightarrow \qquad \sum_{m,c} \psi_{m,\ c} |m, c \oplus \mathsf{Enc}(k, m; r)\rangle$$

Decryption queries For each such query, the challenger decrypts all ciphertexts in the superposition, except those that were the result of a challenge query:

$$\sum_{c,m} \psi_{c,m} |c,\ m\rangle \qquad \longrightarrow \qquad \sum_{c,m} \psi_{c,m} |c,\ m \oplus f(c)\rangle$$

where

$$f(c) = \begin{cases} \perp & \text{if } c \in \mathcal{C} \\ \mathsf{Dec}(k, c) & \text{otherwise} \end{cases}$$

Guess $\mathcal{A}$ produces a bit b', and wins if $b = b'$.

In the above definition, we need to define the operation $m \oplus \perp$. Since the query responses will xor $\perp$ with different messages, we need a convention that makes this operation reversible. Taking $\perp$ to be some bit string that lies outside of the message space, and $\perp \oplus m$ to be bitwise xor will suffice.

Note that we implicitly assume that the decryption algorithm is deterministic. This will be true of our encryption schemes. We note that this is not a limiting assumption since one can always make the decryption algorithm deterministic by deriving the randomness for decryption from a PRF applied to the ciphertext. Also, as in the classical case, a simple hybrid argument shows that the above definition is equivalent to the case where the number of challenge queries is limited to 1. Lastly, it is straightforward to modify the above definition for public key encryption schemes.

6.4.1 Separation

Here we show that quantum chosen ciphertext queries give the adversary more power than classical queries. In particular, we present a public key encryption scheme that is secure under classical queries, but completely insecure once an adversary can make quantum queries. Let $(\mathsf{Gen}_c, \mathsf{Enc}_c, \mathsf{Dec}_c)$ be a public key encryption scheme that is secure under classical chosen ciphertext queries. The idea of our construction is similar in spirit to that for signatures. The construction is as follows:

Construction 6.58. Let $(\mathsf{Gen}_c, \mathsf{Enc}_c, \mathsf{Dec}_c)$ be an encryption scheme and $\mathsf{PRF}_0, \mathsf{PRF}_1$ be pseudorandom functions with key space $\mathcal{K}$, domain $\mathcal{X}$, and range $\mathcal{Y}$. Let t be the bit length of decryption key for Enc_c. Assume that the message space of Enc_c is $\mathcal{Y}^t$ (potentially using one of various message-space

extending techniques). We build a new encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ as follows:

$$\begin{aligned}
\mathsf{Gen}(\lambda) &: (\mathsf{dk}_c, \mathsf{ek}_c) \xleftarrow{R} \mathsf{Gen}_c(\lambda),\ k = (k_1, \ldots, k_t) \xleftarrow{R} \mathcal{K}^t \\
&\quad \text{output } \mathsf{dk} = (\mathsf{dk}_c, k),\ \mathsf{ek} = \mathsf{ek}_c \\
\mathsf{Enc}(\mathsf{ek}_c, m) &: c \leftarrow \mathsf{Enc}_c(\mathsf{ek}, m) \\
&\quad \text{output } (c, 0 \in \{0,1\}) \\
\mathsf{Dec}((\mathsf{dk}_c, k), (c, a, x)) &: \text{Write } dk_c = (\mathsf{dk}_1, \ldots, \mathsf{dk}_t) \in \{0,1\}^t,\ y_i \leftarrow \mathsf{PRF}_{\mathsf{dk}_i}(k_i, x) \\
&\quad \text{Output } \begin{cases} \mathsf{Dec}_c(\mathsf{dk}_c, c) & \text{if } a = 0 \text{ and } x = 0 \\ \perp & \text{if } a = 0 \text{ and } x \neq 0 \\ (y_1, \ldots, y_t) & \text{if } a = 1 \end{cases}
\end{aligned}$$

Theorem 6.59. *If* $(\mathsf{Gen}_c, \mathsf{Enc}_c, \mathsf{Dec}_c)$ *is secure under a classical chosen ciphertext attack,* PRF_0 *is secure against quantum queries, and* PRF_1 *is a quantum-gap PRF, then* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *in Construction 6.58 is secure under a classical chosen ciphertext attack, but the secret key can be recovered using only a single quantum chosen ciphertext query.*

Proof. The proof is very similar to the signature case in Theorem 6.33. To argue the classical security of $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, we note that under classical queries PRF_0 and PRF_1 are both indistinguishable from a random function; in particular, they are indistinguishable from each other. Therefore, $a = 1$ decryption queries reveal no information about sk. Then the security of the protocol follows from the security of $(\mathsf{Gen}_c, \mathsf{Enc}_c, \mathsf{Dec}_c)$.

To argue the insecurity, we note that we can use a decryption query to simulate an oracle query to the oracle $(x_1, \ldots, x_t) \longmapsto (\mathsf{PRF}_{\mathsf{sk}_1}(k_1, x_1), \mathsf{PRF}_{\mathsf{sk}_2}(k_2, x_2), \ldots, (\mathsf{PRF}_{\mathsf{sk}_t}(k_t, x_t)$. Then using the attack from Theorem 6.33, we can recover the secret key sk_c. □

6.4.2 Symmetric CCA Security

In this section, we construct symmetric-key CCA secure encryption. We will follow the encrypt-then-MAC paradigm. Ideally, we would like to show that encrypt-then-MAC, when instantiated with any IND-qCPA-secure encryption scheme and any EUF-qCMA MAC, would be CCA secure. However, it is not obvious how to prove security, as the reduction algorithm has no way to tell which ciphertexts the adversary received as the result of an encryption query, and no way to decrypt the ciphertexts if it has received them. To remedy these problems, we choose a specific encryption scheme and MAC and leave the general security proof as an open question. The encryption scheme allows us to efficiently check if the adversary has seen a particular ciphertext as a result of an encryption query, and to decrypt in this case. The construction is as follows:

Construction 6.60. Let F and G be pseudorandom functions. We construct the following encryption scheme $\mathcal{E} = (\mathsf{Enc}, \mathsf{Dec})$ where:

$$\begin{aligned}
\mathsf{Enc}((k_1,k_2),m) : r &\xleftarrow{R} \{0,1\}^\lambda \\
&c_1 \leftarrow F(k_1,r) \oplus m,\ c_2 \leftarrow G(k_2,(r,m)) \\
&\text{output } (r,c_1,c_2) \\
\mathsf{Dec}((k_1,k_2),(r,c_1,c_2)) : &m \leftarrow c_1 \oplus F(k_1,r),\ c_2' \leftarrow G(k_2,(r,m)) \\
&\text{if } c_2 \neq c_2', \text{ output } \perp \\
&\text{otherwise, output } m
\end{aligned}$$

For security, we require F and G to be quantum secure — secure against queries on a superposition of inputs, such as those that we built in Section 6.3.

Theorem 6.61. *If F and G are quantum-secure pseudorandom functions, then $\mathcal{E}$ in Construction 6.60 is qCCA-secure.*

As demonstrated in Section 6.3, quantum-secure pseudorandom functions can be built from any one-way function. Therefore, Theorem 6.61 shows that quantum chosen ciphertext security can be obtained from the minimal assumption that one-way functions exist. We now give the proof of Theorem 6.61:

Proof. We first sketch the proof: we can replace F and G with random functions and only negligibly affect the success probability. Since each encryption query receives a single r for the entire query superposition, we can answer any encryption query by making a single query to F on r. It is easy to check if a ciphertext (r', c_1, c_2) was computed during an encryption query: just check if $r = r'$. We can also decrypt such a ciphertext, since we have seen $F(k_1, r)$. Including $c_2 = G(k_2, (r, m))$ in the ciphertext guarantees with overwhelming probability that the adversary can only submit valid ciphertexts if they were ciphertexts received during an encryption query, so we might as well reject all ciphertexts (r', c_1, c_2) where r' was not the randomness used in any encryption query. Now, the value of m_b in the challenge query becomes perfectly hidden, which means that the distinguishing probability is 0.

We now give the complete security proof: assume we have an adversary A that breaks the indistinguishability of $\mathcal{E}$ in Construction 6.60 with probability ϵ. We prove security through a sequence of games.

Game 0. This is the standard attack game where A makes q_e encryption queries which are answered using randomness values r_i, q_c challenge queries which are answered using randomness r_i^*, and q_d decryption queries. Let $(m_{i,0}^*, m_{i,1}^*)$ denote the ith challenger query, and (r_i^*, c_i^*, d_i^*) be the response.

Game 1. Replace F and G with truly random functions. That is, answer the ith encryption query by mapping m to $(r_i, F(r_i)\oplus m, G(r_i, m))$, the ith challenge query with $(r_i^*, F(r_i^*)\oplus m_{i,b}^*, G(r_i^*, m_{i,b}^*))$, and answer decryption queries accordingly. Since F and G are quantum-secure pseudorandom functions, the advantage of A in Game 1 is at least $\epsilon - \mathsf{negl}$.

Game 2. Now we abort if there is a collision among any of the r_i or r_i^*. The probability of a collision is at most $(q_e + q_c)^2/2|\mathcal{R}|$ where $\mathcal{R}$ is the randomness space. This quantity is negligible, so A's advantage is still $\epsilon - \mathsf{negl}$.

Notice that we can pick the r_i values and r_i^* value at the start of the game, and query F on these values. Let $\mathcal{T}_i = \{r_j : j \leq i\}$ and $\mathcal{T}_i^* = \{r_j^* : j \leq i\}$. Also let $\mathcal{T} = \mathcal{T}_{q_e}$ and $\mathcal{T}^* = \mathcal{T}_{q_c}^*$. Notice that at any point, A never gets to see $G(r, m)$ for any m if $r \notin \mathcal{T}_i \bigcup \mathcal{T}_j^*$ where i is the number of encryption queries made so far and j is the number of challenge queries made so far. Note also that A only gets to see $G(r_k^*, m)$ where $m = m_{k,b}^*$).

Game 3. For a decryption query on a superposition of ciphertexts (r, c, d), let n_e be the number of encryption queries made so far and n_c the number of challenge queries. Check that $r \in \mathcal{T}_{n_e}$, and respond with $\perp$ for that slot otherwise. We now consider the ciphertexts that would be accepted in Game 2 but rejected in Game 3. Such ciphertexts come in two forms:

- $r \in \mathcal{T}_{n_c}^*$: Then $r = r_i^*$ for some i. In order to not be rejected in Game 2, we must have $c \neq c_i^*$ or $d \neq d_i^*$. In the first case, (r, c, d) is an encryption of a message $m \neq m_{i,b}^*$, so the value of $G(r_i^*, m)$ is hidden to the adversary. Therefore, the probability (r, c, d) is a valid ciphertext is negligible. In the second case, (r, c, d) is an encryption of $m_{i,b}^*$, but then d is not a valid MAC, so decryption fails.
- $r \notin \mathcal{T}_{n_e} \bigcup \mathcal{T}_{n_c}^*$: Then the value of $G(r, m)$ is completely hidden from the adversary, so the probability d is a valid MAC is negligible.

Therefore, the probability of rejection for any ciphertext in Game 3 is only negligibly higher than that in Game 2. This means that with overwhelming probability, we only changed the decryption oracle on a negligible fraction of inputs, so A can only distinguish Games 2 and 3 with negligible probability. Therefore, A's advantage is still $\epsilon - \mathsf{negl}$.

Game 4. Now notice that F is never queried except on the points r_i and r_i^*. Therefore, at the start of the game, we can pick random values f_i and f_i^* to correspond to $F(r_i)$ and $F(r_i^*)$. We can also pick random values g_i^* that correspond to $G(r_i^*, m_{i,b}^*)$ (since we only query G on this point once). The adversary's view in this game is unchanged, so A's advantage is at least $\epsilon - \mathsf{negl}$.

Notice that we answer the ith challenge query with $(r_i^*, f_i^* \oplus m_{i,b}^*, g_i^*)$, and that the values of f_i^* and g_i^* are never used again. This means that $m_{i,b}^*$ is statistically hidden from the adversary. Therefore, A's advantage in Game 4 is identically 0, so $\epsilon = \mathsf{negl}$. □

6.4.3 Public Key CCA Security

In this section, we construct CCA-secure signatures in the public-key setting. The basic idea is to first build a selectively secure identity-based encryption scheme — whose security can be based on the Learning With Errors (LWE) Problem — and then adapt the generic transformation to CCA-security to the quantum setting:

Let $\mathcal{E}_{ibe} = (\mathsf{Gen}_{ibe}, \mathsf{Enc}_{ibe}, \mathsf{Dec}_{ibe}, \mathsf{KeyGen})$ be an IBE scheme that is selectively secure against quantum queries. It is straightforward to show that the basic IBE scheme of Agrawal, Boneh, and Boyen [ABB10a] meets this security notion assuming LWE is hard. Let $\mathcal{S} = (\mathsf{Gen}_s, \mathsf{Sig}, \mathsf{Ver})$ be a strongly EUF-CMA secure one-time signature scheme (quantum security is unnecessary). We now construct an encryption scheme using the generic transformation from IBE to CCA security due to Boneh et al. [BCHK07].

Construction 6.62. $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ where

$$
\begin{aligned}
\mathsf{Gen}(\lambda) &: \mathsf{Gen}_{ibe}(\lambda) \\
\mathsf{Enc}(\mathsf{mek}, m) &: (\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Gen}_s(\lambda) \\
&\quad\ c \leftarrow \mathsf{Enc}_{ibe}(\mathsf{mek}, \mathsf{vk}, m),\ \sigma \leftarrow \mathsf{Sig}(\mathsf{sk}, c) \\
&\quad\ \text{output } (\mathsf{vk}, c, \sigma) \\
\mathsf{Dec}(\mathsf{msk}, (\mathsf{vk}, c, \sigma)) &: \text{if } \mathsf{Ver}(\mathsf{vk}, c, \sigma) \text{ rejects, output } \perp \\
&\quad\ \mathsf{sk}_{\mathsf{vk}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{vk}),\ m \leftarrow \mathsf{Dec}_{ibe}(\mathsf{sk}_{\mathsf{vk}}, c), \quad \text{output } m
\end{aligned}
$$

It is not difficult to adapt the classical security proof to the quantum setting, showing that the above construction achieves quantum CCA security:

Theorem 6.63. *If the* LWE *problem is hard for quantum computers, then there exists a public key encryption scheme that is IND-qCCA secure.*

Chapter 7

Missing Proofs

7.1 Proof of Lemma 3.2

The probability that A outputs a w such that $R(H, w) = \texttt{True}$ is

$$\Pr_{\substack{H \leftarrow D \\ w \leftarrow |\psi_H\rangle}}[R(H,w)] = \sum_H \Pr_D[H] \sum_{w:R(H,w)} |\langle w|\psi_H\rangle|^2 = \sum_w \sum_{H:R(H,w)} \Pr_D[H] |\langle w|\psi_H\rangle|^2$$

Now, $|\langle w|\psi_H\rangle|$ is just the magnitude of the projection of $|w\rangle$ onto the space spanned by the vector $|\psi_H\rangle$, that is, $\text{proj}_{\text{span}|\psi_H\rangle}(|w\rangle)$. This is at most the magnitude of the projection of $|w\rangle$ onto the space spanned by all of the $|\psi_{H'}\rangle$ for $H' \in \mathcal{H}$, or $\text{proj}_{\text{span}\{|\psi_{H'}\rangle\}}(|w\rangle)$. Thus,

$$\Pr_{\substack{H \leftarrow D \\ w \leftarrow |\psi_H\rangle}}[R(z,w)] \leq \sum_w \left(\sum_{H:R(H,w)} \Pr_D[H] \right) \left| \text{proj}_{\text{span}\{|\psi_{H'}\rangle\}}(|w\rangle) \right|^2$$

Now we can perform the sum over H, which gives $\Pr_{H \leftarrow D}[R(H,w)]$. We can bound this by the maximum it attains over all w, giving us

$$\Pr_{\substack{H \leftarrow D \\ w \leftarrow |\psi_H\rangle}}[R(H,w)] \leq \left(\max_w \Pr_{H \leftarrow D}[R(H,w)] \right) \sum_w \left| \text{proj}_{\text{span}\{|\psi_{H'}\rangle\}}(|w\rangle) \right|^2$$

Now, let $\{|b_i\rangle\}$ be an orthonormal basis for $\text{span}\{|\phi_{H'}\rangle\}$. Then

$$\left| \text{proj}_{\text{span}\{|\psi_{H'}\rangle\}}(|w\rangle) \right|^2 = \sum_i |\langle b_i|w\rangle|^2$$

Summing over all w gives

$$\sum_w \left|\mathrm{proj}_{\mathrm{span}\{|\psi_{H'}\rangle\}}(|w\rangle)\right|^2 = \sum_w \sum_i |\langle b_i|w\rangle|^2 = \sum_i \sum_w |\langle b_i|w\rangle|^2$$

Since the w are the possible results of measurement, the vectors $|w\rangle$ form an orthonormal basis for the whole space, meaning $\sum_w |\langle b_i|w\rangle|^2 = |\ |b_i\rangle\ |^2 = 1$. Hence, the sum just becomes the number of $|b_i\rangle$, which is just the dimension of the space spanned by the $|\psi_{H'}\rangle$. Thus,

$$\Pr_{\substack{H\leftarrow D\\ w\leftarrow|\psi_H\rangle}}[R(H,w)] \leq \left(\max_{w\in\mathcal{W}} \Pr_{H\leftarrow D}[R(H,w)]\right)(\dim \mathrm{span}\{|\psi_{H'}\rangle\}) \ .$$

But $\dim \mathrm{span}\{|\psi_{H'}\rangle\}$ is exactly $\mathrm{rank}(\mathbf{\Psi}_{\mathcal{A},\mathcal{H}}) = \mathrm{rank}(\mathcal{A},\mathcal{H})$, which finishes the proof of the theorem.

7.2 Proof of Theorem 3.3

Let $|\psi_H^q\rangle$ be the final state of a quantum algorithm after q quantum oracle calls to an oracle $H \in \mathcal{H}$. We wish to bound the dimension of the space spanned by the vectors $|\psi_H^q\rangle$ for all $H \in \mathcal{H}$. We accomplish this by exhibiting a spanning set for this space. Our basis consists of $|\psi_{H'}^q\rangle$ vectors where H' only differs from H_0 at a maximum of q points in $\mathcal{S}$. We need to show that two things: that our basis consists of $C_{k,q,n}$ vectors, and that our basis does in fact span the whole space.

We first count the number of basis vectors by counting the number of H' oracles. For each r, there are $\binom{k}{r}$ ways of picking the subset $\mathcal{T}$ of size r from $\mathcal{S}$ where H' will differ from H_0. For each subset $\mathcal{T}$, there are n^r possible functions H'. However, if any value $x \in \mathcal{T}$ satisfies $F(x) = H_0(x)$, then this is equivalent to a case where we remove x from $\mathcal{T}$, and we would have already counted this case for a smaller value of r. Thus, we can assume $H'(x) \neq H_0(x)$ for all x in $\mathcal{T}$. There are $(n-1)^r$ such functions. Summing over all r, we get that the number of distinct H' oracles is

$$\sum_{r=0}^{q} \binom{k}{r}(n-1)^r = C_{k,q,n} \ .$$

Next, we need to show that the $|\psi_{H'}^q\rangle$ vectors span the entire space of $|\psi_H^q\rangle$ vectors. We first introduce some notation: let $|\psi^0\rangle$ be the state of a quantum algorithm before any quantum queries. Let $|\psi_H^q\rangle$ be the state after q quantum oracle calls to the oracle H. Let

$$\mathbf{M}_H^q = \mathbf{U}_q\mathbf{H}\mathbf{U}_{q-1}\mathbf{H}\cdots\mathbf{U}_1\mathbf{H} \ .$$

Then $|\psi_H^q\rangle = \mathbf{M}_H^q|\psi^0\rangle$.

We note that since $|\psi^0\rangle$ is fixed for any algorithm, it is sufficient to prove that the $\mathbf{M}_H^q$ matrices

are spanned by the $\mathbf{M}^q_{H'}$.

For any subset $\mathcal{T}$ of $\mathcal{S}$, and a function $F : \mathcal{T} \to \mathcal{Y}$, let $J_{\mathcal{T},F}$ be the oracle such that

$$J_{\mathcal{T},F}(x) = \begin{cases} F(x) & \text{if } x \in \mathcal{T} \\ H_0(x) & \text{otherwise} \end{cases}.$$

Let $\mathbf{M}^q_{\mathcal{T},H}$ denote $\mathbf{M}^q_{J_{\mathcal{T},H}}$. In other words, $\mathbf{M}_{\mathcal{T},H}$ is the transformation matrix corresponding to the oracle that is equal to H on the set $\mathcal{T}$, and equal to H_0 elsewhere. We claim that any $\mathbf{M}^q_H$ for $H \in \mathcal{H}_{\mathcal{S}}$ is a linear combination of the matrices $\mathbf{M}^q_{\mathcal{T},H}$ for subsets $\mathcal{T}$ of $\mathcal{S}$ of size at most q. We will fix a particular H, and for convenience of notation, we will let $J_{\mathcal{T}} = J_{\mathcal{T},H}$. That is, $J_{\mathcal{T}}$ is the oracle that is equal to H on the set $\mathcal{T}$ and H_0 otherwise. We will also let $\mathbf{M}^q_{\mathcal{T}} = \mathbf{M}^q_{\mathcal{T},H}$ and $\mathbf{M}^q = \mathbf{M}^q_H$. That is, $\mathbf{M}^q$ is the transition matrix corresponding to the oracle H, and $\mathbf{M}_{\mathcal{T}}$ is the transition matrix corresponding to using the oracle $J_{\mathcal{T}}$. For the singleton set $\{x\}$, we will also let $J_x = J_{\{x\}}$.

We make the following observations:

$$\mathbf{H} = \left(\sum_{x \in \mathcal{S}} \mathbf{J}_x\right) - (k-1)\mathbf{H}_0 \tag{7.2.1}$$

$$\mathbf{J}_{\mathcal{T}} = \left(\sum_{x \in \mathcal{T}} \mathbf{J}_x\right) - (|\mathcal{T}|-1)\mathbf{H}_0 \tag{7.2.2}$$

These identities can be seen by applying each side to the different inputs. Next, we take $\mathbf{M}^q_H$ and $\mathbf{M}^q_T$ and expand out the $\mathbf{H}$ and $\mathbf{J}_{\mathcal{T}}$ terms using Equations 7.2.1 and 7.2.2:

$$\mathbf{M}^q = \mathbf{U}_q\left(\left(\sum_{x \in \mathcal{S}} \mathbf{J}_x\right) - (k-1)\mathbf{H}_0\right)\mathbf{U}_{q-1}\cdots\mathbf{U}_1\left(\left(\sum_{x \in \mathcal{S}} \mathbf{J}_x\right) - (k-1)\mathbf{H}_0\right) \tag{7.2.3}$$

$$\mathbf{M}^q_{\mathcal{T}} = \mathbf{U}_q\left(\left(\sum_{x \in \mathcal{T}} \mathbf{J}_x\right) - (|\mathcal{T}|-1)\mathbf{H}_0\right)\mathbf{U}_{q-1}\cdots\mathbf{U}_1\left(\left(\sum_{x \in \mathcal{T}} \mathbf{J}_x\right) - (|\mathcal{T}|-1)\mathbf{H}_0\right) \tag{7.2.4}$$

Let $\mathbf{J}_\perp = \mathbf{H}_0$. For a vector $\mathbf{r} \in (\mathcal{S} \cup \{\perp\})^q$, let

$$\mathbf{P}_{\mathbf{r}} = \mathbf{U}_q\mathbf{J}_{r_q}\mathbf{U}_{q-1}\cdots\mathbf{J}_{r_2}\mathbf{U}_1\mathbf{J}_{r_1}$$

For a particular $\mathbf{r}$, we wish to expand the $\mathbf{M}^q$ and $\mathbf{M}^q_{\mathcal{T}}$ matrices in terms of the $\mathbf{P}_{\mathbf{r}}$ matrices. If d of the components of $\mathbf{r}$ are $\perp$, then the coefficient of $\mathbf{P}_{\mathbf{r}}$ in the expansion of $\mathbf{M}^q$ is $(-1)^d(k-1)^d$. If, in addition, all of the other components of $\mathbf{r}$ lie in $\mathcal{T}$, then the coefficient in the expansion of $\mathbf{M}^q_{\mathcal{T}}$ is $(-1)^d(|\mathcal{T}|-1)^d$ (if any of the components of $\mathbf{r}$ lie outside of $\mathcal{T}$, the coefficient is 0).

Now, we claim that, for some values a_ℓ, we have

$$\mathbf{M}^q = \sum_{\ell=0}^{q} a_\ell \sum_{\mathcal{T}\subseteq\mathcal{S}:|\mathcal{T}|=\ell} \mathbf{M}^q_{\mathcal{T}}$$

To accomplish this, we look for the coefficient of $\mathbf{P}_\mathbf{r}$ in the expansion of the right hand side of this equation. Fix an ℓ. Let d be the number of components of $\mathbf{r}$ equal to $\perp$, and let p be the number of distinct component values other than $\perp$. Notice that $p + d \leq q$. Then there are $\binom{k-p}{\ell-p}$ different sets $\mathcal{T}$ of size ℓ for which all of the values of the components lie in $\mathcal{T}$. Thus, the coefficient of $\mathbf{P}_\mathbf{r}$ is

$$\sum_{\ell=p}^{q} a_\ell \binom{k-p}{\ell-p} (-1)^i (\ell-1)^d$$

Therefore, we need values a_ℓ such that

$$\sum_{\ell=p}^{q} a_\ell \binom{k-p}{\ell-p} (\ell-1)^d = (k-1)^d \tag{7.2.5}$$

for all d, p. Notice that we can instead phrase this problem as a polynomial interpolation problem. The right hand side of Equation 7.2.5 is a polynomial P of degree $d \leq q-p$, evaluated at $k-1$. We can interpolate this polynomial using the points $\ell = p, ..., q$, obtaining

$$P(k-1) = \sum_{\ell=p}^{q} P(\ell-1) \prod_{j=p, j\neq\ell}^{q} \frac{k-j}{\ell-j} \ .$$

The numerator of the product evaluates to

$$\frac{(k-p)!}{(k-\ell)(k-q-1)!}$$

while to evaluate the bottom, we split it into two parts: $j = p, ..., \ell-1$ and $j-\ell+1, ..., q$. The first part evaluates to $(\ell-p)!$, and the second part evaluates to $(-1)^{q-\ell}(q-\ell)!$. With a little algebraic manipulation, we have that

$$P(k-1) = \sum_{\ell=p}^{q} P(\ell-1) \left(\binom{k-\ell-1}{k-q-1} (-1)^{q-\ell} \right) \binom{k-p}{\ell-p}$$

for all polynomials $P(x)$ of degree at most $q-p$. Setting $P(x) = x^d$ for $d = 0, ..., q-\ell$, we see that Equation 7.2.5 is satisfied if

$$a_\ell = \binom{k-1-\ell}{k-1-q} (-1)^{q-\ell} \ .$$

7.3 Proof of Lemma 4.1

Recall the setup: $\mathcal{A}$ is a quantum algorithm making q quantum queries to an oracle $H : \mathcal{X} \to \mathcal{Y}$, and c classical queries, where H is drawn from an arbitrary distribution D. We wish to show that, for every z, the quantity $\Pr_{H \leftarrow D}[\mathcal{A}^H() = z]$ is a linear combination of the quantities $\Pr_{H \leftarrow D}[H(x_i) = r_i \forall i \in [2q + c]]$ for all possible settings of the x_i and r_i.

Our approach is similar to the polynomial method, but needs to be adapted to handle classical queries correctly. Let $k = c + q$ be the total number of queries made by $\mathcal{A}$, $Q \subseteq [k]$ be the set of queries that are quantum, and $C \subseteq [k]$ be the set of queries that are classical.

Fix an oracle H. Let $\delta_{x,y}$ be 1 if $H(x) = y$ and 0 otherwise. Let $\boldsymbol{\rho}^{(i)}$ be the density matrix after the ith query, and $\boldsymbol{\rho}^{(i-1/2)}$ be the density matrix before the ith query. $\rho^{(k+1/2)}$ is the final density matrix of the algorithm.

After any query i, let k_i be twice the number of quantum queries made so far, plus the number of classical queries made so far. We make the following claim:

Claim 7.1. *$\boldsymbol{\rho}^{(i)}$ and $\boldsymbol{\rho}^{(i+1/2)}$ are polynomials of the $\delta_{x,y}$ of degree k_i*

We first show that Claim 7.1 implies Lemma 4.1. The probability of a particular output z is obtained from the zth diagonal entry in $\boldsymbol{\rho}^{(k+1/2)}$. For a fixed H, we know this to be a polynomial p of degree at most $k = 2q + c$ in the $\delta_{x,y}$. First, observe that $\delta_{x,y}^2 = \delta_{x,y}$, so p is multilinear. Thus p can be written as

$$\sum_{\substack{S \subseteq \mathcal{X} \times \mathcal{Y} \\ |S| \leq k}} \alpha_S \prod_{(x,r) \in S} \delta_{x,r}$$

for some constants α_S. The probability of outputting z when H is drawn from D is the expected value of p for $H \xleftarrow{R} D$. Thus the probability is

$$\mathop{\mathbb{E}}_{H \xleftarrow{R} D} p = \sum_{\substack{S \subseteq \mathcal{X} \times \mathcal{Y} \\ |S| \leq k}} \alpha_S \mathop{\mathbb{E}}_{H \xleftarrow{R} D} \left[\prod_{(x,r) \in S} \delta_{x,r} \right]$$

Notice that $\mathbb{E}_{H \xleftarrow{R} D} \prod_{(x,r) \in S} [\delta_{x,r}]$ is identical to $\Pr_{H \leftarrow D}[H(x_i) = r_i \forall i \in [k]]$ where $S = \{(x_i, r_i)\}$. Thus the probability of outputting z is a linear combination of the $\Pr_{H \leftarrow D}[H(x_i) = r_i \forall i \in [k]]$ as desired.

It remains to prove Claim 7.1. For $i = 0$, $\boldsymbol{\rho}^{(0)}$ and $\boldsymbol{\rho}^{(0+1/2)}$ are independent of H, so they are not a function of the $\delta_{x,y}$ at all, meaning the degree is $0 = k_0$.

We now inductively assume our claim is true for $i-1$. That is, the entries of $\boldsymbol{\rho}^{(i-1)}$ are polynomials in the $\delta_{x,y}$ of degree at most k_{i-1}. Notice that $\boldsymbol{\rho}^{(i-1/2)}$ is obtained from $\boldsymbol{\rho}^{(i-1)}$ by left- and right-multiplying by matrices whose entries are independent of H. Thus, the entries of $\boldsymbol{\rho}^{(i-1/2)}$ are linear combinations of the entries in $\boldsymbol{\rho}^{(i-1)}$ and are hence also polynomials of degree at most k_{i-1} in the $\delta_{x,y}$.

Now we look at $\boldsymbol{\rho}^{(i)}$. There are two cases:

- i is a quantum query. In this case, $k_i = k_{i-1} + 2$. We can write

$$\boldsymbol{\rho}^{(i)}_{x,y,z,x',y',z'} = \boldsymbol{\rho}^{(i-1/2)}_{x,y-H(x),z,x',y'-H(x'),z}$$

  An alternative way to write this is as

$$\boldsymbol{\rho}^{(i)}_{x,y,z,x',y',z'} = \sum_{r,r'} \delta_{x,y-r}\delta_{x',y'-r'}\boldsymbol{\rho}^{(i-1/2)}_{x,r,z,x',r',z}$$

  By induction, each of the $\boldsymbol{\rho}^{(i-1/2)}_{x,r,z,x',r',z}$ are polynomials of degree k_{i-1} in the $\delta_{x,y}$ values, so $\boldsymbol{\rho}^{(i)}_{x,y,z,x',y',z'}$ is a polynomial of degree $k_{i-1} + 2 = k_i$.

- i is a classical query. This means $k_i = k_{i-1} + 1$. Let $\boldsymbol{\rho}^{(i-1/4)}$ representing the state after measuring the x register, but before making the actual query. This is identical to $\boldsymbol{\rho}^{(i-1/2)}$, except the entries where $x \neq x'$ are zeroed out. We can then write

$$\boldsymbol{\rho}^{(i)}_{x,y,z,x',y',z'} = \sum_{r,r'} \delta_{x,y-r}\delta_{x',y'-r'}\boldsymbol{\rho}^{(i-1/4)}_{x,r,z,x',r',z} = \sum_{r,r'} \delta_{x,y-r}\delta_{x,y'-r'}\boldsymbol{\rho}^{(i-1/2)}_{x,r,z,x,r',z}$$

  Now, notice that $\delta_{x,y-r}\delta_{x,y'-r'}$ is zero unless $y - r = y' - r'$ (since $\delta_{x,y} = 1$ means $H(x) = y$ and H is a function), in which case it just reduces to $\delta_{x,y-r}$. Therefore, we can simply further:

$$\boldsymbol{\rho}^{(i)}_{x,y,z,x',y',z'} = \sum_{r} \delta_{x,y-r}\boldsymbol{\rho}^{(i-1/2)}_{x,r,z,x,(y-y')+r,z}$$

  By induction, each of the $\boldsymbol{\rho}^{(i-1/2)}_{x,r,z,x,(y-y')+r,z}$ values are polynomials of degree k_{i-1} in the $\delta_{x,y}$ values, so $\boldsymbol{\rho}^{(i)}_{x,y,z,x',y',z'}$ is a polynomial of degree $k_{i-1} + 1 = k_i$

Therefore, after all q queries, final matrix $\boldsymbol{\rho}^{(k+1/2)}$ is a polynomial in the $\delta_{x,y}$ of degree at most $k = 2q + c$.

7.4 Proof of Lemma 4.2

By the assumptions of the theorem, for any $2q$ pairs (x_i, r_i), the quantity $\Pr_{H\leftarrow D_\lambda}[H(x_i) = r_i \forall i \in [2q]]$ is a polynomial of degree d in λ with the first $\Delta - 1$ derivatives at 0 being 0. By Theorem 4.1, for a q-query quantum algorithm A, $\Pr_{H\leftarrow D_\lambda}[A^H() = z]$ is a linear combination of these values. Thus, for any z, $\Pr_{H\leftarrow D_\lambda}[A^H() = z]$ is also a polynomial in λ of degree d with the first $\Delta - 1$ derivatives at 0 being 0.

Now, suppose that A distinguishes D_λ from D_0 with probability $\epsilon(\lambda)$. That is

$$\sum_z |\Pr_{H\leftarrow D_\lambda}[A^H() = z] - \Pr_{H\leftarrow D_0}[A^H() = z]| = \epsilon(\lambda) \ .$$

Let $\mathcal{Z}_\lambda$ be the set of z such that z is a more likely output under D_λ than D_0. That is, $\Pr_{H\leftarrow D_\lambda}[A^H() = z] > \Pr_{H\leftarrow D_0}[A^H() = z]$. It is not difficult to show that

$$\Pr_{H\leftarrow D_\lambda}[A^H() \in \mathcal{Z}_\lambda] - \Pr_{H\leftarrow D_0}[A^H() \in \mathcal{Z}_\lambda] = \epsilon(\lambda)/2 \ .$$

Fix λ_0, and consider the quantity

$$p_{\lambda_0}(\lambda) \equiv \Pr_{H\leftarrow D_\lambda}[A^H() \in \mathcal{Z}_{\lambda_0}] = \sum_{z\in\mathcal{Z}_{\lambda_0}} \Pr_{H\leftarrow D_\lambda}[A^H() = z] \ .$$

Then $p_\lambda(\lambda) - p_\lambda(0) = \epsilon(\lambda)/2$. Further, for each λ_0, p_{λ_0} is a degree-d polynomial in λ such that $p^{(i)}_{\lambda_0}(0) = 0$ for $i \in [\Delta - 1]$. It also lies in the range $[0, 1]$ for all $\lambda \in [0, 1]$, so we can use an inequality by Duffin and Schaffer [DS41] to bound the Δ-th derivative for all $\lambda \in [0, 1]$:

$$|p^{(\Delta)}_{\lambda_0}(\lambda)| \leq \frac{4^\Delta \Delta!}{2(2\Delta)!} d^{2\Delta}$$

Thus we can bound $p_{\lambda_0}(\lambda) \leq p_{\lambda_0}(0) + \frac{4^\Delta}{2(2\Delta)!}\lambda^\Delta d^{2\Delta}$. Setting $\lambda_0 = \lambda$, we get

$$\epsilon(\lambda) = 2(p_\lambda(\lambda) - p_\lambda(0)) \leq \frac{4^\Delta}{(2\Delta)!}\lambda^\Delta d^{2\Delta} \ .$$

7.5 Proof of Lemma 4.3

Recall that we have a family of distributions E_r over $\mathcal{Y}^\mathcal{X}$ parametrized by $r \in \mathbb{Z}^+ \bigcup \{\infty\}$. For any $2q$ pairs (x_i, r_i), suppose the function $p(\lambda) = \Pr_{H\leftarrow E_{1/\lambda}}[H(x_i) = r_i \forall i \in [2q]]$ satisfies:

- p is represented by a polynomial in λ of degree at most d.
- $p^{(i)}(0)$, the ith derivative of p at 0, is 0 for each $i \in [\Delta - 1]$.

We will show that any q query quantum algorithm can only distinguish E_r from E_∞ with probability at most $2^{2-\Delta}\zeta(2\Delta)(1/r)^\Delta(d)^{3\Delta}$.

Let $\lambda = 1/r$. By Lemma 4.1, for a q-query quantum algorithm A, $\Pr_{H\leftarrow E_r}[A^H() = z]$ is a linear combination of the $\Pr_{H\leftarrow E_r}[H(x_i) = r_i \forall i \in [2q]]$. Thus, for any z, $\Pr_{H\leftarrow E_{1/\lambda}}[A^H() = z]$ is a polynomial in λ of degree d with the first $\Delta - 1$ derivatives at $\lambda = 0$ being 0.

Now, suppose that A distinguishes $E_{1/\lambda}$ from E_∞ with probability $\epsilon(\lambda)$. That is

$$\sum_z |\Pr_{H\leftarrow E_{1/\lambda}}[A^H() = z] - \Pr_{H\leftarrow E_\infty}[A^H() = z]| = \epsilon(\lambda) \ .$$

Let $\mathcal{Z}_\lambda$ be the set of z such that z is a more likely output under $E_{1/\lambda}$ than E_∞. That is, $\Pr_{H\leftarrow E_{1/\lambda}}[A^H() = z] > \Pr_{H\leftarrow E_\infty}[A^H() = z]$. It is not difficult to show that

$$\Pr_{H\leftarrow E_{1/\lambda}}[A^H() \in \mathcal{Z}_\lambda] - \Pr_{H\leftarrow E_\infty}[A^H() \in \mathcal{Z}_\lambda] = \epsilon(\lambda)/2 \ .$$

Fix λ_0, and consider the quantity

$$p_{\lambda_0}(\lambda) \equiv \Pr_{H\leftarrow E_{1/\lambda}}[A^H() \in \mathcal{Z}_{\lambda_0}] = \sum_{z\in\mathcal{Z}_{\lambda_0}} \Pr_{H\leftarrow E_{1/\lambda}}[A^H() = z] \ .$$

Then $p_\lambda(\lambda) - p_\lambda(0) = \epsilon(\lambda)/2$. Further, for each λ_0, p_{λ_0} is a degree-d polynomial in λ such that $p^{(i)}_{\lambda_0}(0) = 0$ for $i \in [\Delta - 1]$. It also lies in the range $[0, 1]$ when $\lambda = 0$ or $1/\lambda \in \mathbb{Z}^+$. Thus, we make use of the following theorem:

Theorem 7.2. *Let $p(\lambda)$ be a polynomial in λ of degree d such that $p^{(i)}(0) = 0$ for $i \in [\Delta - 1]$, $0 \le p(0) \le 1$, and $0 \le p(1/r) \le 1$ for all $r \in \mathbb{Z}^+$. Then $|p(1/r) - p(0)| < 2^{1-\Delta}\zeta(2\Delta)(1/r)^\Delta d^{3\Delta}$ for all $r \in \mathbb{Z}^+$, where ζ is the Riemann Zeta function.*

Before proving this theorem, we use it to finish the proof of Lemma 4.3. For each λ_0, p_{λ_0} satisfies the conditions of Theorem 7.2, so we must have that $p_{\lambda_0}(\lambda) - p_{\lambda_0}(0) < 2^{1-\Delta}\zeta(2\Delta)\lambda^\Delta d^{3\Delta}$. But then setting $\lambda_0 = \lambda$, we get that

$$\epsilon(\lambda) = 2(p_\lambda(\lambda) - p_\lambda(0)) < 2^{2-\Delta}\zeta(2\Delta)\lambda^\Delta d^{3\Delta} \ .$$

Replacing $1/\lambda$ with r, we have shown that the output distributions of any q query quantum algorithm A under E_r and E_∞ are $2^{2-\Delta}\zeta(2\Delta)(1/r)^\Delta d^{3\Delta}$-close, as desired.

Proof of Theorem 7.2. We have a polynomial p of degree d with $p^{(i)}(0) = 0$ for $i \in [\Delta - 1]$. Further, for $r \in \mathbb{Z}^+ \cap \{\infty\}$, $0 \le p(1/r) \le 1$. Now, let $s(\lambda) = \frac{p(\lambda)-p(0)}{\lambda^\Delta}$. Then s is a $d - \Delta$-degree polynomial. We will now interpolate this polynomial at $d - \Delta + 1$ points: let

$$\lambda_i = \frac{1}{\left\lfloor \frac{(d-\Delta+1)^3}{2i^2} \right\rfloor} \ .$$

Then we can use the Lagrange interpolating polynomials to interpolate $s(\lambda)$. Let $s_i = s(\lambda_i)$. Then:

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

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

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

Then we get

$$\begin{aligned} p(\lambda) - p(0) &= \lambda^\Delta \sum_{i=1}^{d-\Delta+1} \frac{p(\lambda_i) - p(0)}{\lambda_i^\Delta} \ell_i(\lambda) \\ &= \sum_{i-1}^{d-\Delta+1} a_i(\lambda)(p(\lambda_i) - p(0)) \end{aligned}$$

where

$$a_i(\lambda) = \left(\frac{\lambda}{\lambda_i}\right)^\Delta \ell_i(\lambda) \ .$$

Now, observe that $1/\lambda_i$ are integers, so $0 \leq p(\lambda_i) \leq 1$ by assumption. Since $0 \leq p(0) \leq 1$ as well, we must have that $|p(\lambda_i) - p(0)| \leq 1$. Therefore,

$$|p(\lambda) - p(0)| = \sum_{i=1}^{d-\Delta+1} |a_i(\lambda)| \ .$$

We now need to bound this sum.

Claim 7.3. *If $\lambda \leq \lambda_i$ for all i, then $\sum_i |a_i(\lambda)| < 2^{1-\Delta}\zeta(2\Delta)\lambda^\Delta d^{3\Delta}$*

Before proving this claim, we note that it proves Theorem 7.2 when $\lambda \leq \lambda_i$ for all i (equivalently, $\lambda \leq \lambda_1$). If $\lambda > \lambda_1$, then the bound we are trying to prove is at least

$$2^{1-\Delta}\zeta(2\Delta)\left(\frac{(d-\Delta+1)^3}{\lfloor (d-\Delta+1)^3/2 \rfloor}\right)^\Delta > 2\zeta(2\Delta) > 2 \ .$$

Which is already trivially satisfied by the assumption that $p(1/r) \in [0, 1]$. □

Proof of Claim 7.3. First, notice that

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

Now, observe that $\lambda_i \geq \frac{2i^2}{(d-\Delta+1)^3}$ and that

$$\begin{aligned} |\lambda_i - \lambda_j| &= \lambda_i \lambda_j \left| \left\lfloor \frac{(d-\Delta+1)^3}{2i^2} \right\rfloor - \left\lfloor \frac{(d-\Delta+1)}{2j^2} \right\rfloor \right| \\ &\geq \frac{2i^2}{(d-\Delta+1)^3} \lambda_j \left(\left| \frac{(d-\Delta+1)^3}{2i^2} - \frac{(d-\Delta+1)^3}{2j^2} \right| - 1 \right) \end{aligned}$$

Which can be simplified to

$$|\lambda_i - \lambda_j| \geq \lambda_j \frac{\left|i^2 - j^2\right| - \frac{2i^2 j^2}{(d-\Delta+1)^3}}{j^2}$$

We notice that the numerator is minimized by making i and j as large as possible, which is when they are $d-\Delta+1$ and $d-\Delta$. In this case, the quantity becomes $\lambda_j \left(3 - \frac{2}{d-\Delta+1}\right)/j^2$, which is greater than 0 as long as $d - \Delta + 1 \geq 1$ (if $d - \Delta < 0$, then $p(\lambda)$ is a constant, so the theorem is trivial).

Thus

$$\left| \frac{a_i(\lambda)}{\lambda^\Delta} \right| \leq \left(\frac{1}{\lambda_i} \right)^\Delta \prod_{j=1, j\neq i}^{d-\Delta+1} \left(\frac{j^2}{|i^2 - j^2| - \frac{2i^2 j^2}{(d-\Delta+1)^3}} \right)$$

The $(1/\lambda_i)^\Delta$ term is bounded by $\left(\frac{(d-\Delta+1)^3}{2i^2}\right)^\Delta$. We now bound the other term:

Claim 7.4. *For all integers D and i such that $i \leq D$, $\alpha_{D,i} \leq 2$ where*

$$\alpha_{D,i} = \prod_{j=1, j\neq i}^{D} \left(\frac{j^2}{|i^2 - j^2| - \frac{2i^2 j^2}{D^3}} \right)$$

Proof. First, rewrite $\alpha_{D,i}$ as

$$\alpha_{D,i} = \prod_{j=1, j\neq i}^{D} \frac{j^2}{|i^2 - j^2|} \prod_{j=1, j\neq i}^{D} \frac{1}{1 - \frac{2i^2 j^2}{|i^2 - j^2| D^3}} = \beta_{D,i} \gamma_{D,i}$$

Where

$$\beta_{D,i} = \prod_{j=1, j\neq i}^{D} \frac{j^2}{|i^2 - j^2|} \text{ and } \gamma_{D,i} = \prod_{j=1, j\neq i}^{D} \frac{1}{1 - \frac{2i^2 j^2}{|i^2 - j^2| D^3}}$$

We first bound $\beta_{D,i}$:

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

Now we compute the Taylor series expansion of $\ln\left(\frac{\beta_{D,i}}{2}\right) = \ln\binom{2D}{D+i} - \ln\binom{2D}{D}$ around $i = 0$:

$$\ln\left(\frac{\beta_{D,i}}{2}\right) = \sum_{m=1}^{\infty} \frac{i^m}{m!}\left(\left(\frac{d}{dj}\right)^m \ln\binom{2D}{D+j}\right)\Bigg|_{j=0}$$

We recall that

$$\left(\frac{d}{dk}\right)^m \ln\binom{n}{k} = -\psi^{(m-1)}(k+1) - (-1)^m\psi^{(m-1)}(n-k+1)$$

where $\psi^{(m)}(x) = \left(\frac{d}{dx}\right)^m \ln\Gamma(x)$ is the polygamma function. This allows us to write:

$$\ln\left(\frac{\beta_{D,i}}{2}\right) = \sum_{m=1}^{\infty} (-1-(-1)^m)\psi^{(m-1)}(D+1)\frac{i^m}{m!} = -2\sum_{\ell=1}^{\infty} \psi^{(2\ell-1)}(D+1)\frac{i^{2\ell}}{(2\ell)!}$$

Next, we use the fact that $\psi^{(m)}(x) \geq 0$ for all odd m and non-negative x, and $\psi^{(1)}(x+1) \geq \frac{1}{x} - \frac{1}{2x^2}$ to bound

$$\ln\left(\frac{\beta_{D,i}}{2}\right) \leq -\psi^{(1)}(D+1)i^2 = -i^2\left(\frac{1}{D} - \frac{1}{2D^2}\right)$$

We now bound the second term $\gamma_{D,i} = \prod_{j=1,j\neq i}^{D} \frac{1}{1-\frac{2i^2j^2}{|i^2-j^2|D^3}}$

First, let $x_{i,j} = \frac{2i^2j^2}{|i^2-j^2|D^3}$. Notice that $x_{i,j} = x_{j,i}$, and that for $j < i$, $x_{i,j} \leq x_{i,i-1}$. Lastly, notice that $x_{i,i-1} \leq x_{D,D-1}$. Therefore, for all i, j,

$$x_{i,j} \leq \frac{2(D-1)^2}{D(2D-1)} \leq 1 - \frac{1}{D}$$

Now, we use the fact that $\frac{1}{1-x} \leq e^{x+x^2 \ln D}$ for all $x \leq 1 - 1/D$ to bound

$$\ln \gamma_{D,i} \leq \sum_{j=1, j\neq i}^{D} x_{i,j} + x_{i,j}^2 \ln D$$

Much the same way as we analyzed the first term by breaking the product into cases where $j < i$ and $j > i$, we can do the same for this sum. To simplify notation, we use the generalized harmonic numbers:

$$H_n^{(r)} = \sum_{j=1}^{n} \frac{1}{j^r}$$

and $H_n = H_n^{(1)}$. Evaluating the sum in terms of the harmonic number, we can write

$$\begin{aligned}\ln \gamma_{D,i} \leq & \frac{i^2}{2D^3}(1 + 4D - 8i + 2i(H_{D-i} - H_{D+i} + 2H_{2i})) \\ & + \frac{i^4 \ln D}{4D^6}(-3 + 16D + 12i(H_{D-i} - H_{D+i}) + 4i^2(H_{D-i}^{(2)} + H_{D+i}^{(2)}))\end{aligned}$$

Putting together with $\beta_{D,i}$, we have that

$$\begin{aligned}\ln\left(\frac{\alpha_{D,i}}{2}\right) \leq & \frac{i^2}{2D^3}(1 + 5D - 2D^2 - 8i + 2i(H_{D-i} - H_{D+i} + 2H_{2i})) \\ & + \frac{i^4 \ln D}{4D^6}(-3 + 16D + 12i(H_{D-i} - H_{D+i}) + 4i^2(H_{D-i}^{(2)} + H_{D+i}^{(2)})) \\ = & \Big[-4D^5 + \Big(4D^3 i(H_{D-i} - H_{D+i} + 2H_{2i}) + 10D^4 - 16D^3 i + 4i^4(H_{D-i}^{(2)} + H_{D+i}^{(2)}) \ln D\Big) \\ & + \big(2D^3 + 16Di^2 \ln D + 12i^3(H_{D-i} - H_{D+i}) \ln D\big) - 3i^2 \ln D\Big] \times \frac{i^2}{4D^6}\end{aligned}$$

In order to prove the claim, we need to show that the quantity $\ln(\alpha_{D,i}/2)$ on the left is less than 0. We can already see that, for large enough D, since $1 \leq i \leq D$ and $H_n = O(\log n)$ and $H_n^{(2)} = O(1)$, that the $-4D^5$ term will dominate, and therefore the whole quantity will indeed be less than 0. However, this only shows that the claim holds for sufficiently large D, whereas we want to prove the claim for all D. First, we will find for what D the expression on the right is less than 0, and then calculate by hand the rest of the values to show that $\alpha_{D,i} \leq 2$ for all D, i.

Let's first look at the quartic term of the right side (where powers of i and D sum to 4):

$$4D^3 i(H_{D-i} - H_{D+i} + 2H_{2i}) + 10D^4 - 16D^3 i + 4i^4(H_{D-i}^{(2)} + H_{D+i}^{(2)}) \ln D$$

Let $r = i/D$. We use the fact that, $\log n \leq H_n \leq \log n + 1$ and $H_n^{(2)} \leq \pi^2/6$ to bound this above by:

$$D^4\left(4r(3 + \ln(1-r) - \ln(1+r) + 2\ln(2r) + 2\ln(2D)) + 10 - 16r + \frac{4}{3}\pi^2 r^4 \ln D\right)$$

Rearranging gives

$$D^4\left(4r(-1+\ln\left(\frac{16(1-r)r^2}{1+r}\right)+2\ln D)+10+\frac{4}{3}\pi^2 r^4\ln D\right)$$

In the range $r\in[0,1]$, $16(1-r)r^2/(1+r)$ obtains a maximum value of $8(5\sqrt{5}-11)<1.443<1.649<e^{1/2}$. This means $\ln\left(\frac{16(1-r)r^2}{1+r}\right)<\frac{1}{2}$. We also use the fact that $\pi^2\approx 9.87<99/10$. Therefore, we upper bound the quantity as

$$D^4\left(\left(8+\frac{4}{3}\pi^2\right)\ln D+8\right)\leq D^4\left(\frac{106}{5}\ln D+8\right)$$

We now bound the cubic term

$$2D^3+16Di^2\ln D+12i^3(H_{D-i}-H_{D+i})\ln D\leq D^3(2+16r^2\ln D+12r^3(1+\ln(1-r)-\ln(1+r))\ln D)$$

Now, using Taylor expansions, we see that $1+\ln(1-r)-\ln(1+r)\leq 1-2r-\frac{2}{3}r^3$, allowing us to bound this as

$$2D^3(1+2r^2(4+3r-6r^2-2r^4)\ln D)$$

$2r^2(4+3r-6r^2-2r^4)$ obtains a maximum value less than 3, meaning we can bound this as $2D^3(1+3\ln D)$.

Putting this all together, we have that

$$\ln\left(\frac{\alpha_{D,i}}{2}\right)\leq\frac{r^2}{4D^2}\left(-4D^3+D^2\left(\frac{106}{5}\ln D+8\right)+2D(3\ln D+1)\right)$$

Define the function

$$f(D)=-4D^3+D^2(\frac{106}{5}\ln D+8)+2D(3\ln D+1)$$

so that $\ln\left(\frac{\alpha_{D,i}}{2}\right)\leq\frac{r^2}{4D^2}f(D)$. We now show that $f(D)<0$ for $D\geq 18$. We have the following derivatives:

$$\begin{aligned}f'(D)&=\frac{-60D^2+D(186+212\ln D)+(30\ln D+40)}{5}\\f''(D)&=\frac{-120D^2+D(212\ln D+398)+30}{5D}\\f'''(D)&=\frac{-120D^2+212D-30}{5D^2}\end{aligned}$$

Notice that $-120D^2+212D-30=0$ at $\frac{1}{60}(53\pm\sqrt{1909})$ which is approximately $0.155, 1.612$. For D between these two values, $-120D^2+212D-30$, and hence $f'''(D)$, is greater than zero. For D outside of these values (in particular, for $D\geq 2$), $f'''(D)<0$.

Now notice that $f''(7) \approx -5.04 < 0$. Since $f'''(D) < 0$ for $D \geq 2$, this means $f''(D) < 0$ for $D \geq 7$. Next, notice that $f'(13) \approx -107.2 < 0$. Since $f''(D) < 0$ for $D \geq 8$, this means $f'(D) < 0$ for $D \geq 13$. Finally, notice that $f(18) \approx -534.5 < 0$. Since $f'(D) < 0$ for $D \geq 13$, we therefore have that $f(D) < 0$ for $D \geq 18$.

Therefore, we have proved $\alpha_{D,i} \leq 2$ for $D \geq 18$. The remaining 153 cases for $D < 18$ can easily (though tediously) be verified manually. For completeness, the values are included in Table 7.1. This completes the proof of Claim 7.4.

Table 7.1: The values of $\alpha_{D,i}$ for $D < 18$, rounded to the nearest 0.01. Values in bold are exact.

	i												
D	1	2	3	4	5	6	7	8	9	10	11	12	13
17	1.9	1.64	1.27	0.9	0.58	0.34	0.18	0.09	0.04	0.01	0.01	0.00	0.00
16	1.9	1.62	1.25	0.87	0.54	0.31	0.16	0.07	0.03	0.01	0.00	0.00	0.00
15	1.89	1.6	1.21	0.83	0.51	0.28	0.14	0.06	0.02	0.01	0.00	0.00	0.00
14	1.89	1.58	1.18	0.79	0.47	0.25	0.12	0.05	0.02	0.01	0.00	0.00	0.00
13	1.88	1.56	1.15	0.75	0.44	0.23	0.10	0.04	0.02	0.00	0.00	0.00	0.00
12	1.87	1.53	1.11	0.71	0.40	0.20	0.09	0.03	0.01	0.00	0.00	0.00	
11	1.86	1.51	1.07	0.66	0.36	0.17	0.07	0.03	0.01	0.00	0.00		
10	1.85	1.48	1.02	0.61	0.32	0.15	0.06	0.02	0.01	0.00			
9	1.84	1.45	0.97	0.57	0.28	0.12	0.05	0.02	0.00				
8	1.83	1.41	0.92	0.52	0.25	0.10	0.04	0.00					
7	1.82	1.37	0.87	0.47	0.22	0.10	0.01						
6	1.81	1.33	0.82	0.43	0.22	0.01							
5	1.79	1.30	0.79	0.46	0.04								
4	1.79	1.29	0.86	0.10									
3	1.82	1.43	0.23										
2	**2**	**0.5**											
1	**1**												

	i			
D	14	15	16	17
17	0.00	0.00	0.00	0.00
16	0.00	0.00	0.00	
15	0.00	0.00		
14	0.00			

□

With Claim 7.4 proved, we can now complete the proof of Claim 7.3.

$$\left|\frac{a_i(\lambda)}{\lambda^\Delta}\right| \leq \left(\frac{1}{\lambda_i}\right)^\Delta \prod_{j=1, j\neq i}^{d-\Delta+1} \left(\frac{j^2}{|i^2-j^2| - \frac{2i^2j^2}{(d-\Delta+1)^3}}\right) \leq \left(\frac{(d-\Delta+1)^3}{2i^2}\right)^\Delta \times 2 \leq 2\left(\frac{d^3}{2i^2}\right)^\Delta$$

This gives

$$|a_i(\lambda)| \leq \lambda^\Delta d^{3\Delta} 2^{1-\Delta} \frac{1}{i^{2\Delta}}$$

Summing over all i from 1 to $d-\Delta+1$ gives

$$\sum_{i=1}^{d-\Delta+1} |a_i(\lambda)| \leq \lambda^\Delta d^{3\Delta} 2^{1-\Delta} \sum_{i=1}^{d-\Delta+1} \frac{1}{i^{2\Delta}}$$

The sum on the right hand side is the truncated p series for $p = 2\Delta$. This series sums to $\zeta(p)$ where ζ is the Riemann Zeta function, so the truncation is strictly less than this value. Therefore,

$$\sum_{i=1}^{d-\Delta+1} |a_i(\lambda)| < \lambda^{\Delta} d^{3\Delta} 2^{1-\Delta} \zeta(2\Delta)$$

□

7.6 Proof of Lemma 4.4

We prove Lemma 4.4, which states that performing a partial measurement obtaining one of k outcomes during a computation only decreases any output's probability by at most a factor of k.

Proof. Let $|\psi\rangle$ be the final state of $\mathcal{A}$, and let $|\psi_y\rangle$ be the final state of $\mathcal{A}'$ when the outcome of the partial measurement is y. Let $\Pr[y]$ be the probability that the partial measurement obtains y. It is straightforward to show that $|\psi\rangle = \sum_y \sqrt{\Pr[y]}\alpha_y|\psi_y\rangle$ for some α_y of unit norm. Then we have

$$\Pr[x] = |\langle x|\psi\rangle|^2 = \left|\sum_y \sqrt{\Pr[y]}\alpha_y\langle x|\psi_y\rangle\right|^2 \leq k\sum_y \Pr[y]|\langle x|\psi_y\rangle|^2 = k\Pr{}'[x]$$

□

Bibliography

[AA11] Scott Aaronson and Andris Ambainis. The need for structure in quantum speedups. In Bernard Chazelle, editor, *ICS 2011: 2nd Innovations in Computer Science*, pages 338–352, Tsinghua University, Beijing, China, January 7–9, 2011. Tsinghua University Press.

[Aar09] Scott Aaronson. Quantum Copy-Protection and Quantum Money. *Proceedings of the 24th Annual IEEE Conference on Computaitonal Complexity (CCC)*, 2009.

[ABB10a] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 553–572, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany.

[ABB10b] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 98–115, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Berlin, Germany.

[ABI86] Noga Alon, László Babai, and Alon Itai. A Fast and Simple Randomized Parallel Algorithm for the Maximal Independent Set Problem. *Journal of algorithms*, 7(4):567–583, 1986.

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *29th Annual ACM Symposium on Theory of Computing*, pages 284–293, El Paso, Texas, USA, May 4–6, 1997. ACM Press.

[Amb03] Andris Ambainis. Quantum walk algorithm for element distinctness. Available at `http://arxiv.org/abs/quantph/0311001`, 2003.

[Amb05] Andris Ambainis. Polynomial degree and lower bounds in quantum complexity: Collision and element distinctness with small range. *Theory of Computing*, 1(1):37–46, 2005.

[ARU14] Andris Ambainis, Ansis Rosmanis, and Dominique Unruh. Quantum attacks on classical proof systems: The hardness of quantum rewinding. In *55th Annual Symposium on Foundations of Computer Science*, pages 474–483, Philadelphia, PA, USA, October 18–21, 2014. IEEE Computer Society Press.

[AS04] Scott Aaronson and Yaoyun Shi. Quantum lower bounds for the collision and the element distinctness problems. *J. ACM*, 51(4):595–605, July 2004.

[BB84] C. H. Bennett and G. Brassard. Quantum Cryptography: Public Key Distribution and Coin Tossing. In *Proceedings of the IEEE International Conference on Computers, Systems and Signal Processing*, pages 175–179, New York, 1984. IEEE Press.

[BBBV97] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and Weaknesses of Quantum Computing. *SIAM Journal on Computing*, 26:1510–1523, 1997.

[BCG+02] Howard Barnum, Claude Crépeau, Daniel Gottesman, Adam Smith, and Alain Tapp. Authentication of quantum messages. In *43rd Annual Symposium on Foundations of Computer Science*, pages 449–458, Vancouver, British Columbia, Canada, November 16–19, 2002. IEEE Computer Society Press.

[BCHK07] Dan Boneh, Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. *SIAM Journal on Computing*, 36(5):1301–1328, 2007.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Berlin, Germany.

[BHK+11] Gilles Brassard, Peter Høyer, Kassem Kalach, Marc Kaplan, Sophie Laplante, and Louis Salvail. Merkle puzzles in a quantum world. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 391–410, Santa Barbara, CA, USA, August 14–18, 2011. Springer, Berlin, Germany.

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

[BL95] Dan Boneh and Richard J. Lipton. Quantum cryptanalysis of hidden linear functions (extended abstract). In Don Coppersmith, editor, *Advances in Cryptology – CRYPTO'95*, volume 963 of *Lecture Notes in Computer Science*, pages 424–437, Santa Barbara, CA, USA, August 27–31, 1995. Springer, Berlin, Germany.

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 719–737, Cambridge, UK, April 15–19, 2012. Springer, Berlin, Germany.

[BPW15] Nir Bitansky, Omer Paneth, and Daniel Wichs. Perfect structure on the edge of chaos. Cryptology ePrint Archive, Report 2015/126, 2015. `http://eprint.iacr.org/2015/126`.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press.

[BS08] Gilles Brassard and Louis Salvail. Quantum merkle puzzles. In *Proceedings of the Second International Conference on Quantum, Nano and Micro Technologies (ICQNM 2008)*, ICQNM '08, pages 76–79, Washington, DC, USA, 2008. IEEE Computer Society.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd Annual Symposium on Foundations of Computer Science*, pages 136–145, Las Vegas, Nevada, USA, October 14–17, 2001. IEEE Computer Society Press.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 523–552, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany.

[Cor00] Jean-Sébastien Coron. On the exact security of full domain hash. In Mihir Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 229–235, Santa Barbara, CA, USA, August 20–24, 2000. Springer, Berlin, Germany.

[DFG13] Özgür Dagdelen, Marc Fischlin, and Tommaso Gagliardoni. The Fiat-Shamir transformation in a quantum world. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 62–81, Bengalore, India, December 1–5, 2013. Springer, Berlin, Germany.

[DFNS14] Ivan Damgård, Jakob Funder, Jesper Buus Nielsen, and Louis Salvail. Superposition attacks on cryptographic protocols. In Carles Padró, editor, *ICITS 13: 7th International Conference on Information Theoretic Security*, volume 8317 of *Lecture Notes in Computer Science*, pages 142–161, Singapore, 2014. Springer, Berlin, Germany.

[DS41] R J Duffin and A C Schaffer. A refinement of an inequality of the brothers markoff. *Trans. Amer. Math. Soc*, 44(3):289–297, 1941.

[EGM90] Shimon Even, Oded Goldreich, and Silvio Micali. On-line/off-line digital schemes. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 263–275, Santa Barbara, CA, USA, August 20–24, 1990. Springer, Berlin, Germany.

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual Symposium on Foundations of Computer Science*, pages 40–49, Berkeley, CA, USA, October 26–29, 2013. IEEE Computer Society Press.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, October 1986.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th Annual ACM Symposium on Theory of Computing*, pages 197–206, Victoria, British Columbia, Canada, May 17–20, 2008. ACM Press.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *28th Annual ACM Symposium on Theory of Computing*, pages 212–219, Philadephia, Pennsylvania, USA, May 22–24, 1996. ACM Press.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[HSS11] Sean Hallgren, Adam Smith, and Fang Song. Classical cryptographic protocols in a quantum world. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 411–428, Santa Barbara, CA, USA, August 14–18, 2011. Springer, Berlin, Germany.

[Jof74] A. Joffe. On a Set of Almost Deterministic k-Independent Random Variables. *The Annals of Probability*, 2(1):161–162, 1974.

[KK11] Daniel M. Kane and Samuel A. Kutin. Quantum interpolation of polynomials. *Quantum Information & Computation*, 11(1&2):95–103, 2011. First published in 2009.

[KM93] Daphne Koller and Nimrod Megiddo. Constructing small sample spaces satisfying given constraints. In *25th Annual ACM Symposium on Theory of Computing*, pages 268–277, San Diego, California, USA, May 16–18, 1993. ACM Press.

[KM94] Howard Karloff and Yishay Mansour. On Construction of k -Wise Independent Random Variables. *Proceedings of the 26th Annual ACM Symposium on Theory of Computing (STOC)*, 17:564–573, 1994.

[Kut05] Samuel Kutin. Quantum lower bound for the collision problem with small range. pages 29–36, 2005.

[KW03] Jonathan Katz and Nan Wang. Efficiency improvements for signature schemes with tight security reductions. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *ACM CCS 03: 10th Conference on Computer and Communications Security*, pages 155–164, Washington D.C., USA, October 27–30, 2003. ACM Press.

[Lam79] Leslie Lamport. Constructing digital signatures from a one-way function. Technical Report SRI-CSL-98, SRI International Computer Science Laboratory, October 1979.

[Lub85] Michael Luby. A Simple Parallel Algorithm for the Maximal Independent Set Problem. In *Proceedings of the 17th Annual ACM symposium on Theory of computing (STOC)*, pages 1–10. ACM, 1985.

[Mer88] Ralph C. Merkle. A digital signature based on a conventional encryption function. In Carl Pomerance, editor, *Advances in Cryptology – CRYPTO'87*, volume 293 of *Lecture Notes in Computer Science*, pages 369–378, Santa Barbara, CA, USA, August 16–20, 1988. Springer, Berlin, Germany.

[Mid04] Gatis Midrijanis. A polynomial quantum query lower bound for the set equality problem. In Josep Díaz, Juhani Karhumäki, Arto Lepistö, and Donald Sannella, editors, *ICALP 2004: 31st International Colloquium on Automata, Languages and Programming*, volume 3142 of *Lecture Notes in Computer Science*, pages 996–1005, Turku, Finland, July 12–16, 2004. Springer, Berlin, Germany.

[NR95] Moni Naor and Omer Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. In *36th Annual Symposium on Foundations of Computer Science*, pages 170–181, Milwaukee, Wisconsin, October 23–25, 1995. IEEE Computer Society Press.

[PS96] David Pointcheval and Jacques Stern. Provably secure blind signature schemes. In Kwangjo Kim and Tsutomu Matsumoto, editors, *Advances in Cryptology – ASIACRYPT'96*, volume 1163 of *Lecture Notes in Computer Science*, pages 252–265, Kyongju, Korea, November 3–7, 1996. Springer, Berlin, Germany.

[PW08] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Richard E. Ladner and Cynthia Dwork, editors, *40th Annual ACM Symposium on Theory*

of Computing, pages 187–196, Victoria, British Columbia, Canada, May 17–20, 2008. ACM Press.

[Reg02] Oded Regev. Quantum computation and lattice problems. In *43rd Annual Symposium on Foundations of Computer Science*, pages 520–529, Vancouver, British Columbia, Canada, November 16–19, 2002. IEEE Computer Society Press.

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *22nd Annual ACM Symposium on Theory of Computing*, pages 387–394, Baltimore, Maryland, USA, May 14–16, 1990. ACM Press.

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th Annual Symposium on Foundations of Computer Science*, pages 124–134, Santa Fe, New Mexico, November 20–22, 1994. IEEE Computer Society Press.

[Unr10] Dominique Unruh. Universally composable quantum multi-party computation. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 486–505, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany.

[Unr14] Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. Cryptology ePrint Archive, Report 2014/587, 2014. `http://eprint.iacr.org/2014/587`.

[van98] Wim van Dam. Quantum oracle interrogation: Getting all information for almost half the price. In *39th Annual Symposium on Foundations of Computer Science*, pages 362–367, Palo Alto, California, USA, November 8–11, 1998. IEEE Computer Society Press.

[Yue14] Henry Yuen. A quantum lower bound for distinguishing random functions from random permutations. *Quantum Info. Comput.*, 14(13-14):1089–1097, October 2014.