A Note on Quantum-Secure PRPs

Mark Zhandry
Princeton University
mzhandry@princeton.edu

Abstract

We show how to construct pseudorandom permutations (PRPs) that remain secure even if the adversary can query the permutation on a quantum superposition of inputs. Such PRPs are called *quantum-secure*. Our construction combines a quantum-secure pseudorandom *function* together with constructions of *classical* format preserving encryption. By combining known results, we obtain the first quantum-secure PRP in this model whose security relies only on the existence of one-way functions. Previously, to the best of the author's knowledge, quantum security of PRPs had to be assumed, and there were no prior security reductions to simpler primitives, let alone one-way functions.

1 Introduction

Pseudorandom permutations (PRPs), also known as block ciphers, are one of the most widely used cryptographic building blocks. PRPs underly most symmetric key encryption used today. A PRP is a classical, efficiently computable keyed permutation that looks like a truly random permutation to anyone that is only given oracle access to the function. Additionally, given the key it is also possible to efficiently invert the permutaiton.

One of the celebrated results taught in many undergraduate and graduate cryptography courses is that PRPs can be built from a much weaker tool — in fact, the weakest of tools — a one-way function. The construction is usually described as follows. Håstad, Impagliazzo, Levin, and Luby [HILL99] show how to use one-way functions to build pseudorandom generators (PRGs). In turn, Goldreich, Goldwasser, and Micali [GGM86] show how to use pseudorandom generators to buld pseudorandom functions (PRFs), which are a relaxed notion of PRPs where the function is not required to be a permutation[1]. Finally, Luby and Rackoff [LR88] show that by plugging PRFs into a 4-round Feistel Network, one obtains a PRP.

Enter quantum computers. In this work, we consider a very strong *quantum* adversary model for PRPs. Namely, we allow the adversary to not only poses a quantum computer, but also make quantum superposition queries to the permutation. We ask whether or not such strong PRPs can still be built from (quantum resistant) one-way functions as in the classical case. There are three necessary steps toward achieving this goal:

- **Quantum immune one-way functions to quantum immune PRGs.** Fortunately, the proof of security in [HILL99] makes no assumptions about the computational model of the

[1] There is also no efficient inverse function

adversary, and hence works perfectly well for giving a quantum immune PRG from any quantum immune one-way function.

- **Quantum immune PRGs to quantum-secure PRFs.** Here, unfortunately, the classical model *does* restrict the model of the adversary, as the adversary's interactions are assumed to be classical. The classical proof [GGM86] is therefore no longer valid for proving security against quantum queries. Fortunately, Zhandry [Zha12] gives a new and very different proof that does show the security of the GGM PRF against quantum algorithms making quantum queries.

- **Quantum-secure PRFs to quantum-secure PRPs.** The classical security proof works as follows. In the first step, the PRF in the Feistel network is replaced with a truly random function. If we allow quantum queries to the PRP, we will need the PRF to be secure against quantum queries, but otherwise translating this step to the quantum setting is straightforward. The next step, however, is problematic. The next step is to show that the Feistel network, when instantiated with a truly random function, becomes indistinguishable from a truly random permutation. Unfortunately, as with the GGM security proof above, the proof in the classical setting strongly relies on the fact that the adversary only ever gets to see a polynomial number of points, and completely breaks down if the adversary gets to "see" all exponentially many points. While it is entirely possible that Feistel networks *are* indisitnguishable from random permutation under quantum queries, no quantum security proof for Feistel networks is known, and any result along these lines would likely require a substantial reworking on Luby and Rackoff's analysis.

1.1 Our Results

We give the first quantum-secure PRP whose security only relies on the existence of one-way functions.

Techniques. Our construction is a combination of prior work. Our key insight is to formally specify what is needed to convert a quantum-secure PRF into a quantum-secure PRP. We then make a simple observation that allows us results in the study of *classical* PRPs to satisfy our needs.

More precisely, we formally define an object, called a *Function-to-Permutation Converter* (FPC), that suffices for the last step above, constructing PRPs from PRFs. An FPC, roughly, is an algorithm P whose inputs and outputs belong to some domain $\mathcal{X}$, and P makes oracle queries to a function O. We require that, for any function O, the function P^O is a permutation on $\mathcal{X}$. Moreover, we need that if O is a truly random function, then P^O is indistinguishable from a truly random permutation[2]. We stress that an FPC is an information-theoretic object — we do not care if the adversary is efficient or not (though we do require that the algorithm P is efficient). To the best of the author's knowledge, FPCs have not previously been formally defined, though they are implicitly at the heart of every PRP construction from general one-way functions that we are aware of.

Notice that, since in the end we are only interested in polynomial-time adversaries, we need indistinguishability to hold only for polynomially-many queries. In the classical setting, these queries

[2]We also require an algorithm P^{-1} that also makes O queries and is the inverse of P. Security should hold with respect to adversaries that can query both P and P^{-1}

are classical. What Luby and Rackoff show is exactly that four round Feistel networks suffice as classical FPCs. Then, by setting O to be a classical PRF, one obtains a classical PRP.

The most obvious approach to constructing a quantum PRP is to show that a four round Feistel network (or more generally, a k-round network) gives a quantum FPC. As noted above, P and O are still classical objects, but now we demand security against polynomially-many quantum queries. Unfortunately, we still do not know how to prove this statement, and a proof would likely require a very different analysis from the classical setting.

Instead, we will take a much easier approach. Suppose that we actually start with a classical FPC that is secure against $|\mathcal{X}|$ queries, meaning that P^O remains indistinguishable from random even if the adversary can query P^O on its entire domain[3]. Then this classical FPC is *also* a *quantum-secure* FPC secure against up to $|\mathcal{X}|$ queries via a simple reduction. Given a quantum FPC adversary, we construct a classical FPC adversary that queries on the entire domain so that it knows the entire function, and then answers the quantum FPC adversary's queries using this knowledge. Plugging in a quantum-secure PRF as O into q quantum-secure FPC, we therefore immediately obtain a quantum secure PRP.

How do we build such full-domain FPCs? After all, despite FPCs being information-theoretic objects, they are used in a computational security setting where all adversaries are efficient. In the classical setting, there was no need to have an FPC secure beyond a polynomial number of queries, so there was little effort in the community targeted specifically at producing such an object. Note that basic four round Feistel network as investigated by [LR88] actually fails to be FPCs in this regime[4].

Fortunately for us, full-domain FPCs have been studied implicitly in the context of *format preserving encryption* [BRRS09]. A core tool in format preserving encryption is a PRP where the domain can be very small, possibly even polynomial. Note that four round Feistel networks require the domain size to exponential (or at least superlogairthmic) in order to be secure.

Once the domain is polynomial, it is conceivable that the adversary can query the function on the entire domain. A fascinating line of work [Mor05, GP07, BRRS09, SS12, HMR12, RY13, MR14] has shown how to build PRPs that (1) support small potentially polynomial-sized domains, and (2) support adversaries querying on the entire domain. What these works implicitly show are constructions of FPCs supporting these domains and query numbers.

These works had an entirely different focus that ours, as the primary motivation was the small domain setting, and full domain security was only studied as a result of considering small domains. Nonetheless, the constructions of FPCs implicit in the state-of-the-art works on format preserving encryption [RY13, MR14] remain efficient and maintain full-domain security even when the domain is exponentially large[5].

Putting everything together, by combining the FPCs implicit in the format preserving encryption literature with the quantum-secure PRF of Zhandry, we obtain quantum secure PRPs from any quantum immune one-way function.

[3]Note that since FPCs are information-theoretic objects, it makes sense to consider them even when the adversary can make exponentially-many queries

[4]Though it may be that larger k does give a full-domain FPC. We are unaware of any results ruling this out.

[5]It is quite likely that other constructions achieve what is needed as well, but we only checked these two constructions.

2 Preliminaries

A function $\epsilon = \epsilon(\lambda)$ is negligible if it is smaller than any inverse polynomial. A *classical* algorithm is said to be efficient if it runs in probabilistic polynomial time.

This work will make minimal use of quantum formalism. Nonetheless, for completeness we recall the basics of quantum computation and quantum oracle queries. The following is taken more or less verbatim from [BZ13].

Quantum Computation. We give a short introduction to quantum computation. A quantum system A is a complex Hilbert space $\mathcal{H}$ together with and 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 the value i with probability $|\langle b_i|\psi\rangle|^2$, and the quantum state collapses to the basis vector $|b_i\rangle$. If $|\psi\rangle$ is actually a state in a joint system $\mathcal{H} \otimes \mathcal{H}'$, then $|\psi\rangle$ can be written as

$$|\psi\rangle = \sum_{i=0}^{d-1} \alpha_i \, |b_i\rangle|\psi_i'\rangle$$

for some complex values α_i and states $|\psi_i'\rangle$ over $\mathcal{H}'$. Then, the measurement over $\mathcal{H}$ obtains the value i with probability $|\alpha_i|^2$ and in this case the resulting quantum state is $|b_i\rangle|\psi_i'\rangle$.

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 then measured, obtaining the tuple (a, b, c) with probability $|\langle a, b, c|\psi_x\rangle|^2$. The output of the algorithm is b. We say that a quantum algorithm is efficient if each of the unitary matrices $\mathbf{U}_i$ come from some fixed basis set, and n, the number of unitary matrices, is polynomial in the size of the input.

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 input state of the algorithm, and let $\mathbf{U}_0, ..., \mathbf{U}_q$ be the unitary transformations applied between queries. Note that the transformations $\mathbf{U}_i$ are themselves possibly the products of many simpler unitary transformations. The final state of the algorithm will be

$$\mathbf{U}_q \mathbf{O}_q ... \mathbf{U}_1 \mathbf{O}_1 \mathbf{U}_0 |\psi_0\rangle$$

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 $\mathbf{O}_i$. We call a quantum oracle algorithm efficient if the number of queries q is a polyomial, and each of the transformations $\mathbf{U}_i$ between queries can be written as the product polynomially many unitary transformations from some fixed basis set.

Conventions for this paper. For this paper, all cryptographic primitives discussed will be implemented by efficient *classical* algorithms. We will use CComp and QComp to distinguish between classical and quantum adversaries, and we will use CQuery and QQuery to distinguish between adversaries making classical or quantum queries.

2.1 Cryptographic Primitives

All algorithms and adversaries will take as input a security parameter λ. We will use the convention that λ is specified in binary, and algorithms are efficient if they run in time polynomial in λ (which is exponential in the bit-length of λ).

Definition 2.1. A CComp- (resp. QComp-)*one-way* function is an efficient classical function $\mathsf{OWF} : \{0,1\}^\lambda \to \{0,1\}^*$ such that, for any efficient classical (resp. quantum) adversary $\mathcal{A}$, the probability that $\mathcal{A}$ inverts OWF on a random input is negligible. That is, there exists a negligible $\mathsf{negl}(\lambda)$ such that

$$\Pr[\mathsf{OWF}(\ \mathcal{A}(\lambda, \mathsf{OWF}(x))\ ) = \mathsf{OWF}(x) : x \leftarrow \{0,1\}^\lambda] < \mathsf{negl}(\lambda)$$

Definition 2.2. A (C,Q)-*pseudorandom function* (PRF), for pair $(\mathsf{C},\mathsf{Q}) \in \{(\mathsf{CComp},\mathsf{CQuery}), (\mathsf{QComp},\mathsf{CQuery}), (\mathsf{QComp},\mathsf{QQuery})\}$[6] is a family of efficient classical functions $\mathsf{PRF}_{m,n} : \{0,1\}^\lambda \times \{0,1\}^m \to \{0,1\}^n$ such that the following holds. For any polynomially bounded $m = m(\lambda)$ and $n = n(\lambda)$, and any efficient adversary $\mathcal{A}$, $\mathcal{A}$ cannot distinguish $\mathsf{PRF}_{m,n}(k,\cdot)$ for a random $k \leftarrow \{0,1\}^\lambda$ from a truly random function $O : \{0,1\}^m \to \{0,1\}^n$. That is, there exists a negligible $\mathsf{negl}(\lambda)$ such that

$$\left|\Pr[\mathcal{A}^{\mathsf{PRF}_{m,n}(k,\cdot)}(\lambda) = 1 : k \leftarrow \{0,1\}^\lambda] - \Pr[\mathcal{A}^{O(\cdot)}(\lambda) = 1]\right| < \mathsf{negl}(\lambda)$$

Here, O is chosen at random from the set of all functions from $\{0,1\}^m$ into $\{0,1\}^n$. If $(\mathsf{C},\mathsf{Q}) = (\mathsf{CComp},\mathsf{CQuery})$, $\mathcal{A}$ is restricted to being an efficient classical algorithm making classical queries to O. If $\mathsf{C} = \mathsf{QComp}$, then $\mathcal{A}$ is allowed to be a quantum algorithm. In this case, if $\mathsf{Q} = \mathsf{CQuery}$, $\mathcal{A}$, despite being quantum, is still required to make classical queries. If $\mathsf{Q} = \mathsf{QQuery}$, $\mathcal{A}$ is allowed to make quantum queries to O.

We will often abuse notation when m and n are clear from context and write PRF to denote $\mathsf{PRF}_{m,n}$.

We note that PRFs domains and co-domains are *monotone*, in the sense that a PRF on for large domain and range gives a PRF for small domain and range, simply by hard-coding some bits of the inputs and discarding some bits of the output. Therefore, if a PRF is secure for large domain/range sizes (say, polynomial m and n), it can also easily be made secure for small domain and range sizes (say, logarithmic or polylogarithmic m and n).

We recall the following theorem, combining the results [GGM86, HILL99, Zha12].

[6] Note that it does not make sense to consider classical adversary's that make quantum queries

Theorem 2.3. *If* CComp*-one-way functions exist, then so do* $(\mathsf{CComp}, \mathsf{CQuery})$*-PRFs. Moreover, if* QComp*-one-way functions exist, then so do* $(\mathsf{QComp}, \mathsf{QQuery})$*-PRFs.*

The classical version follows from [GGM86, HILL99]. The reductions in those works are actually independent of the computational model, so the proofs also show that QComp-one-way functions imply $(\mathsf{QComp}, \mathsf{CQuery})$-PRFs. The analysis does not extend to the QQuery setting. Instead, the work of [Zha12] shows how to modify the proof to get the quantum part of Theorem 2.3.

3 Pseudorandom Permutations

Definition 3.1. A $(\mathsf{C}, \mathsf{Q}, \mathsf{D})$*-pseudorandom permutation* (PRP), for $(\mathsf{C}, \mathsf{Q}) \in \{(\mathsf{CComp}, \mathsf{CQuery}), (\mathsf{QComp}, \mathsf{CQuery}), (\mathsf{QComp}, \mathsf{QQuery})\}$ and $\mathsf{D} \in \{\mathsf{LargeD}, \mathsf{AnyD}\}$ is a family of efficient classical function pairs $\mathsf{PRP}_o : \{0,1\}^\lambda \times \{0,1\}^o \to \{0,1\}^o$ and $\mathsf{PRP}_o^{-1} : \{0,1\}^\lambda \times \{0,1\}^o \to \{0,1\}^o$ such that the following holds. First, for every key k and integer o, the functions $\mathsf{PRP}_o(k,\cdot)$ and $\mathsf{PRP}_o^{-1}(k,\cdot)$ are inverses of each other. That is, $\mathsf{PRP}_o^{-1}(k, \mathsf{PRP}_o(k,x)) = x$ for all o, k, x. This implies that $\mathsf{PRP}_o(k,\cdot)$ is a permutation.

Second, for any polynomially-bounded $o = o(\lambda)$ and any efficient adversary $\mathcal{A}$, $\mathcal{A}$ cannot distinguish $\mathsf{PRP}_o(k,\cdot)$ for a random $k \leftarrow \{0,1\}^\lambda$ from a truly random *permutation* $P : \{0,1\}^o \to \{0,1\}^o$. We consider the strong variant where $\mathcal{A}$ has access to both P and P^{-1}. That is, there exists a negligible $\mathsf{negl}(\lambda)$ such that

$$\left|\Pr[\mathcal{A}^{\mathsf{PRP}_o(k,\cdot),\mathsf{PRP}_o^{-1}(k,\cdot)}(\lambda) = 1 : k \leftarrow \{0,1\}^\lambda] - \Pr[\mathcal{A}^{P(\cdot),P^{-1}(\cdot)}(\lambda) = 1]\right| < \mathsf{negl}(\lambda)$$

Here, P is chosen at random from the set of all permutations on $\{0,1\}^o$. If $(\mathsf{C}, \mathsf{Q}) = (\mathsf{CComp}, \mathsf{CQuery})$, $\mathcal{A}$ is restricted to being an efficient classical algorithm making classical queries to P, P^{-1}. If $\mathsf{C} = \mathsf{QComp}$, then $\mathcal{A}$ is allowed to be a quantum algorithm. In this case, if $\mathsf{Q} = \mathsf{CQuery}$, $\mathcal{A}$, despite being quantum, is still required to make classical queries. If $\mathsf{Q} = \mathsf{QQuery}$, $\mathcal{A}$ is allowed to make quantum queries to P, P^{-1}.

Finally, if $\mathsf{D} = \mathsf{LargeD}$, we only require security to hold for o that are also lower-bounded by a polynomial. If $\mathsf{D} = \mathsf{AnyD}$, we allow for arbitrary polynomially-bounded o (so o could be, for example, logarithmic in this case). Note that a PRP for domain $\{0,1\}^o$ does not give a PRP for domain $\{0,1\}^{o'}$ for $o' < o$ by fixing bits, since PRP_o is not guaranteed to be a permutation on $\{0,1\}^{o'}$. Therefore, we make a distinction between PRPs that only work for sufficiently large domains (LargeD), and those that remain secure for both large and small domains (AnyD).

We will often abuse notation when o is clear from context and write PRP to denote PRP_o.

3.1 Function to Permutation Converters

In this section, we define a new information theoretic object called a *function to permutation converter* (FPC), which roughly takes a random function and transforms it into a random permutation. Such FPCs are implicitly used in many constructions of PRPs (and basically every construction of a PRP from general one-way functions).

Definition 3.2. Let $\mathsf{Q} \in \{\mathsf{CQuery}, \mathsf{QQuery}\}$, $\mathsf{D} \in \{\mathsf{LargeD}, \mathsf{AnyD}\}$. Fix a family of functions $\mathcal{Q}$ in o, λ. An $(\mathsf{Q}, \mathsf{D}, \mathcal{Q})$-FPC is a sequence of pairs of efficient classical oracle algorithms $\mathsf{F}_o, \mathsf{R}_o$ where:

- $\mathsf{F}_o, \mathsf{R}_o$ take as input λ and string $x \in \{0,1\}^o$, and output a string $y \in \{0,1\}^o$

- There exist polynomials $m(o,\lambda), n(o,\lambda)$ such that $\mathsf{F}_o, \mathsf{R}_o$ make queries to a function $O : \{0,1\}^m \to \{0,1\}^n$.
- $\mathsf{F}_o, \mathsf{R}_o$ are efficient, in that they make at most $\mathsf{poly}(o,\lambda)$ queries to O.
- $\mathsf{F}_o, \mathsf{R}_o$ are inverses: $\mathsf{R}_o^O(\lambda, \mathsf{F}_o^O(\lambda, x)) = x$ for all $x \in \{0,1\}^o$ and all oracles O.
- $\mathsf{F}_o, \mathsf{R}_o$ are indistinguishable from a random permutation and its inverse, given query access. That is, let $o(\lambda)$ be any polynomially bounded function. If $\mathsf{D} = \mathsf{LargeD}$, we will restrict to o being lower-bounded by a polynomial as well. Let $q(\lambda) = q(o(\lambda), \lambda)$ be any function in $\mathcal{Q}$. Let $\mathcal{A}$ be any (possibly inefficient) adversary that makes at most q queries to its oracles, where if $\mathsf{Q} = \mathsf{QQuery}$ the queries are allowed to quantum, and if $\mathsf{Q} = \mathsf{CQuery}$, the queries are restricted to being classical. Then there exists a negligible $\mathsf{negl}(\lambda)$ such that:

$$\left|\Pr[\mathcal{A}^{\mathsf{F}_o^O(\lambda,\cdot),\mathsf{R}_o^O(\lambda,\cdot)}(\lambda) = 1] - \Pr[\mathcal{A}^{P,P^{-1}}(\lambda) = 1]\right| < \mathsf{negl}(\lambda)$$

3.2 The Main Lemma

Here we prove that FPCs are sufficient to build PRPs.

Lemma 3.3. *Let $\mathcal{Q}$ be the set of all polynomials. If (C,Q)-PRFs exist and $(\mathsf{Q},\mathsf{D},\mathcal{Q})$-FPCs exist, then $(\mathsf{C},\mathsf{Q},\mathsf{D})$-PRPs exist.*

We note that the $(\mathsf{C},\mathsf{Q},\mathsf{D}) = (\mathsf{CComp},\mathsf{CQuery},\mathsf{D})$ version of this lemma is implicit in essentially all constructions of classically secure PRPs from general PRFs. The proof is a straightforward adaptation to handle quantum queries.

Proof. We prove the $(\mathsf{QComp},\mathsf{QQuery},\mathsf{AnyD})$ version, the other versions being similar. The construction is simple: $\mathsf{PRP}_o(k,x) = \mathsf{F}_o^{\mathsf{PRF}_{m,n}(k,\cdot)}(\lambda,x)$ and $\mathsf{PRP}_o^{-1}(k,x) = \mathsf{R}_o^{\mathsf{PRF}_{m,n}(k,\cdot)}(\lambda,x)$ where $\lambda = |k|$.

Security is proved by a sequence of hybrids.

Hybrid 0. This is the case where the adversary is given the oracles $P(x) = \mathsf{F}_o^{\mathsf{PRF}_{m,n}(k,\cdot)}(\lambda,x)$ and $P^{-1}(x) = \mathsf{R}_o^{\mathsf{PRF}_{m,n}(k,\cdot)}(\lambda,x)$.

Hybrid 1. In this case, we switch PRF to be random, so that the adversary is given the oracles $P(x) = \mathsf{F}_o^{O(\cdot)}(\lambda,x)$ and $P^{-1}(x) = \mathsf{R}_o^{O(\cdot)}(\lambda,x)$ for a random function $O : \{0,1\}^m \to \{0,1\}^n$.

Suppose $\mathcal{A}$ distinguishes **Hybrid 0** from **Hybrid 1** with non-negligible probability. Then we can construct a PRF adversary $\mathcal{B}^O(\lambda) = \mathcal{A}^{\mathsf{F}_o^O(\lambda,\cdot),\mathsf{R}_o^O(\lambda,\cdot)}(\lambda)$. If $O(\cdot) = \mathsf{PRF}_{m,n}(k,\cdot)$ for a random k, then the view of $\mathcal{A}$ is identical to **Hybrid 0**. Likewise, if $O(\cdot)$ is a truly random function, then the view of $\mathcal{A}$ is identical to **Hybrid 1**. Moreover, $\mathcal{B}$ can answer any quantum query made by $\mathcal{A}$ by making a polynomial number of quantum queries to its own oracle. Therefore, $\mathcal{B}$ is an efficient quantum algorithm. Moreover, its advantage in breaking the security of PRF is identical to $\mathcal{A}$'s distinguishing advantage between **Hybrid 0** and **Hybrid 1**. This contradicts our assumption that PRF is $(\mathsf{QComp},\mathsf{QQuery})$ secure.

Therefore, **Hybrid 0** and **Hybrid 1** are indistinguishable, except with negligible probability.

Hybrid 2. In this hybrid, the adversary is given truly random permutation oracles P, P^{-1}. Indistinguishability between **Hybrid 1** and **Hybrid 2** follows immediately from the $(\mathsf{QQuery}, \mathsf{AnyD}, \mathcal{Q})$ security of F, R.

Thus, **Hybrid 0** and **Hybrid 2** are indistinguishable, demonstrating the security of PRP. □

3.3 Constructions of FPCs

From Theorem 2.3, we already have PRFs from one-way functions. It remains to demonstrate a suitable FPC.

Luby and Rackoff's [LR88] proof essentially shows the following:

Lemma 3.4. *Let $\mathcal{Q}$ be the set of polynomial-bounded functions. Then $(\mathsf{CQuery}, \mathsf{LargeD}, \mathcal{Q})$-FPCs exist.*

This gives us the following corollary:

Corollary 3.5. *For $\mathsf{C} \in \{\mathsf{CComp}, \mathsf{QComp}\}$, if C-one-way functions exist, then $(\mathsf{C}, \mathsf{CQuery}, \mathsf{LargeD})$-PRPs exist.*

Luby Rackoff's construction uses Feistel networks. If we could show that Feistel networks are also $(\mathsf{QQuery}, \mathsf{LargeD}, \mathcal{Q})$-FPCs, then we would immediately get $(\mathsf{QComp}, \mathsf{QQuery}, \mathsf{LargeD})$-PRPs from any QComp-one-way function as desired. Unfortunately, we do not know how to show this, and leave it as an interesting open problem.

Instead, we observe that if $q = 2^o$, then even a classical algorithm can query the entire domain of F_o, obtaining the entire truth table for the function (as well as its inverse R_o). Then it can simulate any query, quantum or otherwise. Thus, in this regime, there is no distinction between classical and quantum FPCs. Note that this regime still makes sense even though an adversary making 2^o queries is inefficient, since FPC security is defined for computationally unbounded adversaries. This motivates the following definition:

Definition 3.6. The family $(\mathsf{F}_o, \mathsf{R}_o)$ is a D-Full Domain FPC if it is a $(\mathsf{CQuery}, \mathsf{D}, \mathcal{Q})$-FPC where $\mathcal{Q}$ contains a function $q(o, \lambda)$ such that $q(o, \lambda) \geq 2^o$.

Lemma 3.7. *Let $\mathcal{Q}$ be any class of functions. Then a D-Full Domain FPC is also a $(\mathsf{QQuery}, \mathsf{D}, \mathcal{Q})$-FPC.*

Next, we observe that the card shuffles at the heart of constructions of format preserving encryption give us Full-Domain FPCs. The following is adapted from [Mor05, GP07, SS12, HMR12, RY13, MR14].

Lemma 3.8. AnyD*-Full Domain FPCs exist.*

The goal of these works is to use Lemma 3.8 to give the following improvement to Corollary 3.5:

Corollary 3.9. *For $\mathsf{C} \in \{\mathsf{CComp}, \mathsf{QComp}\}$, if C-one-way functions exist, then $(\mathsf{C}, \mathsf{CQuery}, \mathsf{AnyD})$-PRPs exist.*

3.4 Putting it All Together

Combining Theorem 2.3 with Lemmas 3.3, 3.7, and 3.8, quantum-secure PRPs:

Theorem 3.10. *If QComp-one-way functions exist, then so do $(\mathsf{QComp}, \mathsf{QQuery}, \mathsf{AnyD})$-PRPs.*

References

[BRRS09] Mihir Bellare, Thomas Ristenpart, Phillip Rogaway, and Till Stegers. Format-preserving encryption. In Michael J. Jacobson Jr., Vincent Rijmen, and Reihaneh Safavi-Naini, editors, *SAC 2009: 16th Annual International Workshop on Selected Areas in Cryptography*, volume 5867 of *Lecture Notes in Computer Science*, pages 295–312, Calgary, Alberta, Canada, August 13–14, 2009. Springer, Heidelberg, Germany.

[BZ13] Dan Boneh and Mark Zhandry. Secure signatures and chosen ciphertext security in a quantum computing world. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 361–379, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, October 1986.

[GP07] Louis Granboulan and Thomas Pornin. Perfect block ciphers with small blocks. In Alex Biryukov, editor, *Fast Software Encryption – FSE 2007*, volume 4593 of *Lecture Notes in Computer Science*, pages 452–465, Luxembourg, Luxembourg, March 26–28, 2007. Springer, Heidelberg, Germany.

[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.

[HMR12] Viet Tung Hoang, Ben Morris, and Phillip Rogaway. An enciphering scheme based on a card shuffle. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 1–13, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Heidelberg, Germany.

[LR88] Michael Luby and Charles Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM Journal on Computing*, 17(2), 1988.

[Mor05] Ben Morris. The mixing time of the Thorp shuffle. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pages 403–412, Baltimore, Maryland, USA, May 22–24, 2005. ACM Press.

[MR14] Ben Morris and Phillip Rogaway. Sometimes-recurse shuffle - almost-random permutations in logarithmic expected time. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 311–326, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany.

[RY13] Thomas Ristenpart and Scott Yilek. The mix-and-cut shuffle: Small-domain encryption secure against N queries. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 392–409, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.

[SS12] Emil Stefanov and Elaine Shi. FastPRP: Fast pseudo-random permutations for small domains. Cryptology ePrint Archive, Report 2012/254, 2012. http://eprint.iacr.org/2012/254.

[Zha12] Mark Zhandry. How to construct quantum random functions. In *53rd Annual Symposium on Foundations of Computer Science*, pages 679–687, New Brunswick, NJ, USA, October 20–23, 2012. IEEE Computer Society Press.