11-4-2022

Quantum Key-Length Extension

Joseph Jaeger
Georgia Institute of Technology

Fang Song
Portland State University

Stefano Tessaro
University of Washington

Citation Details

Published as: Jaeger, J., Song, F., & Tessaro, S. (2021, November). Quantum key-length extension. In Theory of Cryptography Conference (pp. 209-239). Springer, Cham.

Quantum Key-length Extension

Joseph Jaeger[1], Fang Song[2], and Stefano Tessaro[3]

[1] Georgia Institute of Technology, Atlanta, Georgia, US
josephjaeger@gatech.edu
[2] Portland State University, Portland, Oregon, US
fang.song@pdx.edu
[3] Paul G. Allen School of Computer Science & Engineering
University of Washington, Seattle, Washington, US
tessaro@cs.washington.edu

Abstract. Should quantum computers become available, they will reduce the effective key length of basic secret-key primitives, such as blockciphers. To address this we will either need to use blockciphers with inherently longer keys or develop key-length extension techniques to amplify the security of a blockcipher to use longer keys.

We consider the latter approach and revisit the FX and double encryption constructions. Classically, FX was proven to be a secure key-length extension technique, while double encryption fails to be more secure than single encryption due to a meet-in-the-middle attack. In this work we provide positive results, with concrete and tight bounds, for the security of *both* of these constructions against quantum attackers in ideal models.

For FX, we consider a partially-quantum model, where the attacker has quantum access to the ideal primitive, but only classical access to FX. This is a natural model and also the strongest possible, since effective quantum attacks against FX exist in the fully-quantum model when quantum access is granted to both oracles. We provide two results for FX in this model. The first establishes the security of FX against non-adaptive attackers. The second establishes security against general adaptive attackers for a variant of FX using a random oracle in place of an ideal cipher. This result relies on the techniques of Zhandry (CRYPTO '19) for lazily sampling a quantum random oracle. An extension to perfectly lazily sampling a quantum random permutation, which would help resolve the adaptive security of standard FX, is an important but challenging open question. We introduce techniques for partially-quantum proofs without relying on analyzing the classical and quantum oracles separately, which is common in existing work. This may be of broader interest.

For double encryption, we show that it amplifies strong pseudorandom permutation security in the fully-quantum model, strengthening a known result in the weaker sense of key-recovery security. This is done by adapting a technique of Tessaro and Thiruvengadam (TCC '18) to reduce the security to the difficulty of solving the list disjointness problem and then showing its hardness via a chain of reductions to the known quantum difficulty of the element distinctness problem.

1 Introduction

The looming threat of quantum computers has inspired significant efforts to design and analyze post-quantum cryptographic schemes. In the public-key setting, polynomial-time quantum algorithms for factoring and computing discrete logarithms essentially break all practically deployed primitives [28].

In the secret-key setting, Grover's quantum search algorithm [12] will reduce the effective key length of secret-key primitives by half. Thus, a primitive like the AES-128 blockcipher which may be thought to have 128 bits of security against classical computers may provide no more than 64 bits of security against a quantum computer, which would be considered significantly lacking. Even more worrisome, it was shown relatively recent that quantum computers can break several secret-key constructions completely such as the Even-Mansour blockcipher [23] and CBC-MAC [17] if we grant the attacker fully quantum access to the cryptosystem.

This would not be the first time that we find ourselves using too short of a key. A similar issue had to be addressed when the DES blockcipher was widely used and its 56 bit keylength was considered insufficient.

Following approaches considered at that time, we can either transition to using basic primitives which have longer keys (e.g. replacing AES-128 with AES-256) or design key-length extension techniques to address the loss of concrete security due to quantum computers. In this paper we analyze the latter approach. We consider two key-length extension techniques, FX [20] and double encryption, and provide provable bounds against quantum attackers in *ideal* models.

Of broader and independent interest, our study of FX focuses on a hybrid quantum model which only allows for *classical* online access to the encrypted data, whereas offline computation is quantum. This model is sometimes referred to as the "Q1 model" in the cryptanalysis literature [17,5], in contrast to the fully-quantum, so-called "Q2 model", which allows for quantum online access. This is necessary in view of existing attacks in the Q2 model showing that FX is no more secure than the underlying cipher [24], but *also*, Q1 is arguably more realistic and less controversial than Q2. We observe that (as opposed to the plain model) ideal-model proofs in the Q1 model can be *harder* than those in the Q2 model, as we need to explicitly account for *measuring* the online queries to obtain improved bounds. In many prior ideal-model Q1 proofs, e.g. [21,22,14,4,18], this interaction is handled essentially for free because the effects of online and offline queries on an attacker's advantage can largely be analyzed separately. Our work introduces techniques to handle the interaction between classical online queries and quantum offline ideal-model queries in Q1 proofs that cannot be analyzed separately. On the other hand, our result on double encryption considers the full Q2 model – and interestingly, restricting adversaries to the Q1 model does not improve the bound. To be self-explanatory we will often refer to the Q1 and Q2 models as the partially-quantum and fully-quantum models, respectively.

The remainder of this introduction provides a detailed overview of our results for these two constructions, and of the underlying challenges and techniques.

1.1 The FX Construction

The FX construction was originally introduced by Kilian and Rogaway [20] as a generalization of Rivest's DESX construction. Consider a blockcipher E which uses a key $K \in \{0,1\}^k$ to encrypt messages $M \in \{0,1\}^n$. Then the FX construction introduces "whitening" key $K_2 \in \{0,1\}^n$ which is xor-ed into the input and output of the blockcipher. Formally, this construction is defined by $\mathsf{FX}[\mathsf{E}](K \,\|\, K_2, M) = \mathsf{E}_K(M \oplus K_2) \oplus K_2$. (Note that the Even-Mansour blockcipher [11] may be considered to be a special case of this construction where $k = 0$, i.e., the blockcipher is a single permutation.) This construction has negligible efficiency overhead as compared to using E directly.

Kilian and Rogaway proved this scheme secure against classical attacks in the ideal cipher model. In particular, they established that

$$\mathsf{Adv}^{\mathsf{sprp}}_{\mathsf{FX}}(\mathcal{A}) \leqslant pq/2^{k+n-1}\,.$$

Here $\mathsf{Adv}^{\mathsf{sprp}}$ measures the advantage of $\mathcal{A}$ in breaking the strong pseudorandom permutation (SPRP) security of FX while making at most p queries to the ideal cipher and at most q queries to the FX construction. Compared to the $p/2^k$ bound achieved by E alone, this is a clear improvement so FX can be considered a successful key-length extension technique again classical attackers.

Is this construction equally effective in the face of quantum attackers? The answer is unfortunately negative. Leander and May [24], inspired by a quantum attack due to Kuwakado and Morii [23] that completely breaks Even-Mansour blockcipher, gave a quantum attack against FX, which shows that the whitening keys provide essentially no additional security over that achieved by E in isolation. Bonnetain, et al. [5] further reduced the number of *online* quantum queries in the attack. Roughly speaking, $O(n)$ quantum queries to FX construction and $O(n2^{k/2})$ local quantum computations of the blockcipher suffice to recover the secret encryption key. Note that, however, such attacks require full quantum access to both the ideal primitive and to the instance FX that is under attack, i.e. they are attacks in the fully-quantum model. The latter is rather strong and may be considered unrealistic. While we cannot prevent a quantum attacker from locally

evaluating a blockcipher in quantum superposition, honest implementations of encryption will likely continue to be classical.[4]

PARTIALLY-QUANTUM MODEL. Because of the realistic concern of the fully-quantum model and the attacks therein that void key extension in FX, we turn to the partially-quantum model in which the attacker makes quantum queries to ideal primitives, but only *classical* queries to the cryptographic constructions.

In this model there has been extensive quantum cryptanalysis on FX and related constructions [13,5]. The best existing attack [5] recovers the key of FX using roughly $2^{(k+n)/3}$ classical queries to the construction and $2^{(k+n)/3}$ quantum queries to the ideal cipher. However, to date, despite the active development in provable quantum security, we are not aware of SPRP or just PRF security analysis, which gives stronger security guarantees. Namely, it should not just be infeasible to retrieve a key, but also to merely distinguish the system from a truly random permutation (or function). We note that in the special case where the primitives are plain-model instantiations (e.g., non-random-oracle hash functions), with a bit of care many security reductions carry over to the quantum setting [25]. This is because the underlying primitives are hidden from the adversary, and hence the difficulty arising from the interaction of classical and quantum queries to two correlated oracles becomes irrelevant.

Our main contribution on FX is to prove, for the first time, indistinguishability security in the partially-quantum model, in two restricted ways. Although they do not establish the complete security, our security bounds are tight in their respective settings.[5]

NON-ADAPTIVE SECURITY. We first consider non-adaptive security where we restrict the adversary such that its classical queries to the FX construction (but not to the underlying ideal cipher) must be specified before execution has begun. We emphasize that non-adaptive security of a blockcipher suffices to prove adaptive security for many practical uses of blockciphers such as the various randomized or stateful encryption schemes (e.g. those based on counter mode or output feedback mode) in which an attacker would have no control over the inputs to the blockcipher.

In this setting the bound we prove is of the form

$$\mathsf{Adv}^{\mathsf{sprp\text{-}na}}_{\mathsf{FX}}(\mathcal{A}) \leqslant O\left(\sqrt{p^2q/2^{k+n}}\right).$$

Supposing $k = n = 128$ (as with AES-128), an attacker able to make $p \approx 2^{64}$ queries to the ideal cipher could break security of E in isolation. But to attack FX, such an attacker with access to $q \approx 2^{64}$ encryptions would need to make $p \approx 2^{96}$ queries to the ideal cipher. In fact we can see from our bound that breaking the security with constant probability would require the order of $\Omega(2^{(k+n)/3})$ queries in total, matching the bound given in the attacks mentioned above [5]. Hence our bound is tight.

To prove this bound we apply a one-way to hiding (O2H) theorem of Ambainis, Hamburg, and Unruh [3], an improved version of the original one in [32]. This result provides a clean methodology for bounding the probability that an attacker can distinguish between two functions drawn from closely related distributions given quantum access. The non-adaptive setting allows us to apply this result by sampling the outputs of FX ahead of time and then considering the ideal world in which the ideal cipher is chosen independently of these outputs and the real world in which we very carefully reprogram this ideal cipher to be consistent with the outputs chosen for FX. These two ideal ciphers differ only in the $O(q)$ places where we need to reprogram.

ADAPTIVE SECURITY OF FFX. As a second approach towards understanding fully adaptive security of FX, we consider a variant construction (which we call FFX for "function FX") that replaces the random permutation with a random function. In particular, suppose F is a function family which uses a key $K \in \{0,1\}^k$ on input

[4] Some may argue that maintaining purely classical states, e.g., enforcing perfect measurements is also non-trivial physically. However, we deem maintaining coherent quantum superposition significantly more challenging.

[5] Throughout, when mentioning tightness, we mean it with respect to the resources required to achieve advantage around one. The roots in our bounds make them weaker for lower resource regimes. Removing these is an interesting future direction.

messages $M \in \{0,1\}^n$ to produce outputs $C \in \{0,1\}^m$. Then we define $\mathsf{FFX}[\mathsf{F}](K \,\|\, K_2, M) = \mathsf{F}_K(M \oplus K_2)$.[6] For this construction we prove a bound of the form

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{FFX}}(\mathcal{A}) \leqslant O\left(\sqrt{p^2 q / 2^{k+n}}\right).$$

in the partially-quantum random oracle model. Note that this matches the bound we obtained for the non-adaptive security of FX. Since the same key-recovery attack [5] also applies here, it follows that our bound is tight as well. Our proof combines two techniques of analyzing a quantum random oracle, the O2H theorem above and a simulation technique by Zhandry [35]. The two techniques usually serve distinct purposes. O2H is helpful to program a random oracle, whereas Zhandry's technique is typically convenient for (compactly) maintaining a random oracle and providing some notion of "recording" the queries. In essence, in the two function distributions of O2H for which we aim to argue indistinguishability, we apply Zhandry's technique to simulate the functions in a compact representation. As a result, analyzing the guessing game in O2H, which implies indistinguishability, becomes intuitive and much simplified. This way of combining them could also be useful elsewhere.

To build intuition for the approach of our proof, let us first consider one way to prove the security of this construction classically. The core idea is to use lazy sampling. In the ideal world, we can independently lazily sample a random function $F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^m$ to respond to F queries and a random function $T : \{0,1\}^n \to \{0,1\}^m$ to respond to FFX queries. These lazily random functions are stored in tables.

The real world can similarly be modeled by lazily sampling F and T to respond to the separate oracles. However, these oracles need to be kept consistent. So if the adversary ever queries M to FFX and $(K, M \oplus K_2)$ to F, then the game should copy values between the two tables such that the same value is returned by both oracles. (Here K and K_2 are the keys honestly sampled by the game.) Alternatively, we can think of the return value being stored only in the T table when such queries occur (rather than being copied into both tables) as long as we remember that this has happened. When represented in this manner, the two games only differ if the adversary makes such a pair of queries, where we think of the latter one as being "bad". Thus a simple $O(pq/2^{k+n})$ bound on the probability of making such a query bounds the advantage of the adversary.

In our quantum security proof we wish to proceed analogously. First, we find a way to represent the responses to oracle queries with two (superpositions over) tables that are independent in the ideal world and dependent in the real world (the dependency occurs only for particular "bad" inputs). Then (using the O2H theorem of Ambainis, Hamburg, and Unruh) we can bound the distinguishing advantage by the probability of an attacker finding a "bad" input. In applying this theorem we will jointly think of the security game and its adversary $\mathcal{A}$ as a combined adversary $\mathcal{A}'$ making queries to an oracle which takes in both the input of $\mathcal{A}$ *and* the tables being stored by the game – processing them appropriately.

The required representation of the oracles via two tables is a highly non-trivial step in the quantum setting. For starters, the no-cloning theorem prevents us from simply recording queries made by the adversary. This has been a recurring source of difficulty for numerous prior papers such as [33,30,9]. We make use of the recent elegant techniques of Zhandry [35] which established that, by changing the perspective (e.g., to the Fourier domain), a random function can be represented by a table which is initialized to all zeros and then xor-ed into with each oracle query made by the adversary. This makes it straightforward to represent the ideal world as two separate tables. To represent the real world similarly, we exploit the fact that the queries to FFX are classical. To check if an input to FFX is "bad" we simply check if the corresponding entry of the random oracle's table is non-zero. To check if an input to the random oracle is "bad" we check if it overlaps with prior queries to FFX which we were able to record because they were classical. For a "bad" input we then share the storage of the two tables and this is the only case where the behavior of the real world differs from that of the ideal world. These "bad" inputs may of course be part of a superposition query and it is only for the bad components of the superposition that the games differ.

[6] Note we have removed the external xor with K_2. In FX this xor is necessary, but in our analysis it would not provide any benefit for FFX.

Difficulty of extending to FX. It is possible that this proof could be extended to work for normal FX given an analogous way to lazily represent a random permutation. Unfortunately, no such representation is known.

Czajkowski, et al. [8] extended Zhandry's lazy sampling technique to a more general class of random functions, but this does not include permutations because of the correlation between the different outputs of a random permutation. Chevalier, et al. [6] provided a framework for recording queries to quantum oracles, which enables succinctly recording queries made to an externally provided function for purposes of later responding to inverse queries. This is distinct from the lazy sampling of a permutation that we require. Rosmanis [27] introduced a new technique for analyzing random permutations in a compressed manner and applied it to the question of inverting a permutation (given only forward access to it). Additional ideas seem needed to support actions based on the oracle queries that have been performed so far. This is essential in order to extend our proof for the function variant of FX to maintain consistency for the real world in the face of "bad" queries.

Recent work of Czajkowski [7] provided an *imperfect* lazy sampling technique for permutations and used it to prove indifferentiability of SHA3. They claim that their lazy sampling strategy cannot be distinguished from a random permutation with advantage better than $O(q^2/2^n)$. Unfortunately, this bound is too weak to be useful to our FX proof. For example, if $k \geqslant n$ we already have $O(q^2/2^n)$ security without key-length extension. Determining if it is possible to *perfectly* lazily sample a random permutation remains an interesting future direction.

1.2 Double Encryption

The other key-extension technique we consider is double encryption. Given a blockcipher $\mathsf{E} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ this is defined by $\mathsf{DE}[\mathsf{E}](K_1 \,\|\, K_2, M) = \mathsf{E}_{K_2}(\mathsf{E}_{K_1}(M))$. This construction requires more computational overhead than FX because it requires two separate application of the blockcipher with different keys. Classically, this construction is not considered to be a successful key-length extension technique because the meet-in-the-middle attack [10,26] shows that it can be broken in essentially the same amount of time as E alone.

However, this does not rule out that double encryption is an effective key-length extension method in the quantum setting, as it is not clear that the Grover search algorithm [12] used to halve the effective keylength of blockciphers can be composed with the meet-in-the middle attack to unify their savings. The security of double encryption in the quantum setting was previously considered by Kaplan [16]. They related the key-recovery problem in double encryption to the claw-finding problem, and gave the tight quantum query bound $\Theta(N^{2/3})$ for solving key recovery (here $N = 2^k$ is the length of the lists in the claw-finding problem). This indicates that in the quantum setting double encryption is in fact useful (compare to $N^{1/2}$), although key-recovery security is fairly weak.

We strengthen their security result by proving the SPRP security, further confirming double encryption as an effective key-extension scheme against quantum attacks. This is proven in the *fully-quantum* model, and the bound we obtain matches the attack in [16] which works in the partially-quantum model. Namely restricting to the weaker partially-quantum model would not improve the bound. Our result is obtained by a reduction to list disjointness. This is a worst-case decision problem measuring how well an algorithm can distinguish between a pair of lists with zero or *exactly* one element in common, which can be viewed as a decision version of the claw-finding problem. This reduction technique was originally used by Tessaro and Thiruvengadam [31] to establish a classical time-memory trade-off for double encryption. We observe that their technique works for a quantum adversary.

We then construct a chain of reductions to show that the known quantum hardness of element distinctness [1,34] (deciding if a list of N elements are all distinct) can be used to establish the quantum hardness of solving list disjointness. Our result (ignoring log factors) implies that a highly successful attacker must make $\Omega(2^{2k/3})$ oracle queries which is more than the $\Omega(2^{k/2})$ queries needed to attack E used in isolation.

Our proof starts by observing that Zhandry's [34] proof of the hardness of the search version of element distinctness (finding a collision in a list) in fact implies that a promise version of element distinctness (promising that there is exactly one collision) is also hard. Then a simple reduction (randomly splitting the

element distinctness list into two lists) shows the hardness of the search version of list disjointness. Next we provide a binary-search inspired algorithm showing that the decision version of list disjointness can be used to solve the search version, implying that the decision version must be hard. During our binary search we pad the lists we are considering with random elements to ensure that our lists maintain a fixed size which is necessary for our proof to go through.

The final bound we obtain for double encryption is of the form

$$\mathsf{Adv}^{\mathsf{sprp}}_{\mathsf{DE}}(\mathcal{A}) \leqslant O\left(\sqrt[6]{(q \cdot k \lg k)^3 / 2^{2k}}\right).$$

The sixth root arises in this bound from the final step in our chain of results analyzing list disjointness. The binary search algorithm requires its underlying decision list disjointness algorithm to have relatively high advantage. To obtain this from a given algorithm with advantage δ we need to amplify its advantage by running in on the order of $1/\delta^2$ times. The number of queries depending on the square of δ causes the root to arise in the proof.

1.3 Overview

In Section 2, we introduce preliminaries such as notation, basic cryptographic definitions, and some background on quantum computation that we will use throughout the paper. Following this, in Section 3 we consider the security of FX in the partially quantum setting. Non-adaptive SPRP security of FX is proven in Section 3.1 and adaptive PRF security of FFX is proven in Section 3.2. We conclude with Section 4 in which we prove the SPRP security of double encryption against fully quantum adaptive attacks.

2 Preliminaries

For $n, m \in \mathbb{N}$, we let $[n] = \{1, \ldots, n\}$ and $[n..m] = \{n, n+1, \ldots, m\}$. The set of length n bit strings is denoted $\{0,1\}^n$. We use $\|$ to denote string concatenation. We let $\mathsf{Inj}(n, m)$ denote the set of injections $f : [n] \to [m]$.

We let $y \leftarrow\!\!\$\ \mathcal{A}[O_1, \ldots](x_1, \ldots)$ denote the (randomized) execution of algorithm $\mathcal{A}$ with input $x_1, \ldots$ and oracle access to $O_1, \ldots$ which produces output y. For different $\mathcal{A}$ we will specify whether it can access its oracles in quantum superposition or only classically. If $\mathcal{S}$ is a set, then $y \leftarrow\!\!\$\ \mathcal{S}$ denotes randomly sampling y from $\mathcal{S}$.

We express security notions via pseudocode games. See Fig. 1 for some example games. In the definition of games, oracles will sometimes be specified by pseudocode with the following form.

$$\begin{array}{l}\underline{\text{Oracle O}(X_1, \cdots : Z_1, \ldots)} \\ \text{//Code defining } X'_1, \ldots \text{ and } Z'_1, \ldots \\ \text{Return } (X'_1, \cdots : Z'_1, \ldots)\end{array}$$

This notation indicates that $X_1, \ldots$ are variables controlled by the adversary prior to the oracle query and $Z_1, \ldots$ are variables controlled by the game itself which the adversary cannot access. At the end of the execution of the oracle, these variables are overwritten with the values indicated in the return statement. Looking ahead, we will be focusing on quantum computation so this notation will be useful to make it explicit that O can be interpreted as a unitary acting on the registers $X_1, \ldots$ and $Z_1, \ldots$ (because O will be an efficiently computable and invertible permutation over these values). If $\mathbf{H}$ is a function stored by the game, then oracle access to $\mathbf{H}$ represents access to the oracle that on input $(X, Y : \mathbf{H})$ returns $(X, \mathbf{H}(X) \oplus Y : \mathbf{H})$.

We define games as outputting boolean values and let $\mathsf{Pr}[\mathsf{G}]$ denote the probability that game G returns `true`. When not otherwise indicated, variables are implicitly initialized to store all 0's.

If $\mathcal{A}$ is an adversary expecting access to multiple oracles we say that it is *order consistent* if the order it will alternate between queries to these different oracles is a priori fixed before execution. Note that order consistency is immediate if, e.g., $\mathcal{A}$ is represented by a circuit where each oracle is modeled by a separate oracle gate, but is not immediate for other possible representations of an adversary.

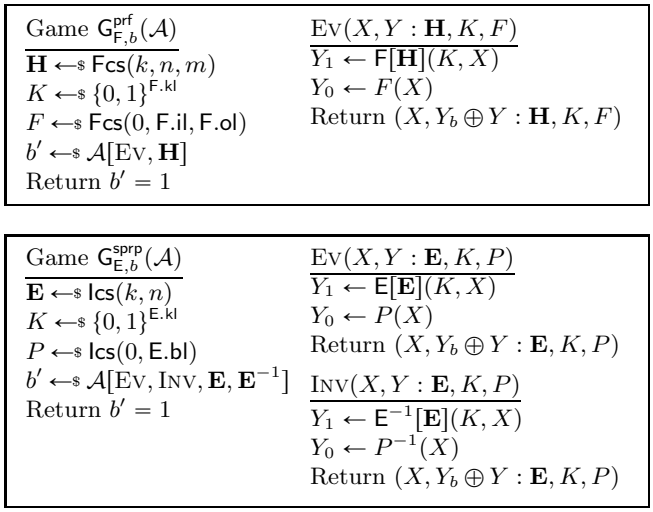

Fig. 1. Security games measuring PRF security of a family of functions F and SPRP security of a blockcipher E.

Ideal Models. In this work we will work in ideal models – specifically, the random oracle model or the ideal cipher model. Fix $k, n, m \in \mathbb{N}$ (throughout this paper we will treat these parameters as having been fixed already). We let $\mathsf{Fcs}(k,n,m)$ be the set of all functions $\mathbf{H} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^m$ and $\mathsf{Ics}(k,n) \subset \mathsf{Fcs}(k,n,n)$ be the set of all functions $\mathbf{E} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ such that $\mathbf{E}(K,\cdot)$ is a permutation on $\{0,1\}^n$. When convenient, we will write $\mathbf{H}_K(x)$ in place of $\mathbf{H}(K,x)$ for $\mathbf{H} \in \mathsf{Fcs}(k,n,m)$. Similarly, we will write $\mathbf{E}_K(x)$ for $\mathbf{E}(K,x)$ and $\mathbf{E}_K^{-1}(\cdot)$ for the inverse of $\mathbf{E}_K(\cdot)$ when $\mathbf{E} \in \mathsf{Ics}(k,n)$. When $K = \varepsilon$ we omit the subscript to $\mathbf{H}$ or $\mathbf{E}$.

In the random oracle model, honest algorithms and the adversary are given oracle access to a randomly chosen $\mathbf{H} \in \mathsf{Fcs}(k,n,m)$. In the ideal cipher model, they are given oracle access to $\mathbf{E}$ and $\mathbf{E}^{-1}$ for $\mathbf{E}$ chosen at random from $\mathsf{Ics}(k,n)$. We refer to queries to these oracles as *primitive queries* and queries to all other oracles as *construction queries*.

Function family and pseudorandomness. A function family F is an efficiently computable element of $\mathsf{Fcs}(\mathsf{F.kl},\mathsf{F.il},\mathsf{F.ol})$. If, furthermore, $\mathsf{F} \in \mathsf{Ics}(\mathsf{F.kl},\mathsf{F.il})$ and F^{-1} is efficiently computable then we say F is a blockcipher and let $\mathsf{F.bl} = \mathsf{F.il}$.

If F is a function family (constructed using oracle access to a function $\mathbf{H} \in \mathsf{Fcs}(k,n,m)$), then its security (in the random oracle model) as a *pseudorandom function* (PRF) is measured by the game $\mathsf{G}^{\mathsf{prf}}$ shown in Fig. 1. In it, the adversary $\mathcal{A}$ attempts to distinguish between a *real* world ($b=1$) where it is given oracle access to F with a random key K and an *ideal* world ($b=0$) where it is given access to a random function. We define the advantage function $\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F}}(\mathcal{A}) = \mathsf{Pr}\left[\mathsf{G}^{\mathsf{prf}}_{\mathsf{F},1}(\mathcal{A})\right] - \mathsf{Pr}\left[\mathsf{G}^{\mathsf{prf}}_{\mathsf{F},0}(\mathcal{A})\right]$.

If E is a blockcipher (constructed using oracle access to a function $\mathbf{E} \in \mathsf{Ics}(k,n)$ and its inverse), then its security (in the ideal cipher model) as a *strong pseudorandom permutation* (SPRP) is measured by the game $\mathsf{G}^{\mathsf{sprp}}$ shown in Fig. 1. In it, the adversary $\mathcal{A}$ attempts to distinguish between a real world ($b=1$) where it is given oracle access to E, E^{-1} with a random key K and an ideal world ($b=0$) where it is given access to a random permutation. We define the advantage function $\mathsf{Adv}^{\mathsf{sprp}}_{\mathsf{F}}(\mathcal{A}) = \mathsf{Pr}\left[\mathsf{G}^{\mathsf{sprp}}_{\mathsf{F},1}(\mathcal{A})\right] - \mathsf{Pr}\left[\mathsf{G}^{\mathsf{sprp}}_{\mathsf{F},0}(\mathcal{A})\right]$.

In some examples, we will restrict attention to *non-adaptive* SPRP security. In such cases our attention is restricted to attackers whose queries to Ev and Inv when relevant are a priori fixed before execution. That is, $\mathcal{A}$ is a non-adaptive attacker which makes at most q classical, non-adaptive queries to Ev, Inv if there exists $M_1, \ldots, M_{q'}, Y_{q'+1}, \ldots, Y_q \in \{0,1\}^n$ such that $\mathcal{A}$ only ever queries Ev on M_i for $1 \leqslant i \leqslant q'$ and Inv on Y_i for $q'+1 \leqslant i \leqslant q$. Then we write $\mathsf{Adv}^{\mathsf{sprp-na}}(\mathcal{A})$ in place of $\mathsf{Adv}^{\mathsf{sprp}}(\mathcal{A})$.

Game $\mathsf{G}^{\mathsf{dist}}_{\mathbf{D},b}(\mathcal{A})$	Game $\mathsf{G}^{\mathsf{guess}}_{\mathbf{D}}(\mathcal{A})$
$(S, S', P_0, P_0', P_1, P_1', z) \leftarrow\!\!\$\ \mathbf{D}$	$(S, S', P_0, P_0', P_1, P_1', z) \leftarrow\!\!\$\ \mathbf{D}$
$b' \leftarrow\!\!\$\ \mathcal{A}[P_b, P_b'](z)$	$i \leftarrow\!\!\$\ \{1, \ldots, q\}$
Return $b' = 1$	Run $\mathcal{A}[P_0, P_0']$ until its i-th query
	Measure the input x to this query
	If the query is to P_0 then
	Return $x \in S$
	Else (the query is to P_0')
	Return $x \in S'$

Fig. 2. Games used for O2H Theorem 1.

2.1 Quantum Background

We assume the reader has basic familiarity with quantum computation. Quantum computation proceeds by performing unitary operations on registers which each contain a fixed number of qubits. We sometimes use $\circ$ to denote composition of unitaries. Additionally, qubits may be measured in the computational basis. We will typically use the principle of deferred measurements to without loss of generality think of such measurements as being deferred until the end of computation.

The Hadamard transform $\mathcal{H}$ acts on a bitstring $x \in \{0,1\}^n$ (for some $n \in \mathbb{N}$) via $\mathcal{H}\left|x\right\rangle = 1/\sqrt{2^n} \cdot \sum_{x'} (-1)^{x \cdot x'} \left|x'\right\rangle$. Here $\cdot$ denotes inner product modulo 2 and the summation is over $x' \in \{0,1\}^n$. The Hadamard transform is its own inverse. We sometimes use the notation $\mathcal{H}^{X_1, X_2, \cdots}$ to denote the Hadamard transform applied to registers $X_1, X_2, \ldots$.

We make use of the fact that if P is a permutation for which both P and P^{-1} can be efficiently implemented classically, then there is a comparable efficient quantumly computable unitary U_P which maps according to $U_P \left|x\right\rangle = \left|P(x)\right\rangle$ for $x \in \{0,1\}^n$. For simplicity, we often write P in place of U_P. If $f : \{0,1\}^n \to \{0,1\}^m$ is a function, we define the permutation $f[\oplus](x, y) = (x, f(x) \oplus y)$.

One-way to hiding. We will make use of (a slight variant of) a one-way to hiding (O2H) theorem of Ambainis, Hamburg, and Unruh [3]. The theorem will consider an adversary given oracle access either to permutations (P_0, P_0') or permutations (P_1, P_1'). It relates the advantage of the adversary in distinguishing between these two cases to the probability that the adversary can be used to find one of those points on which P_0 differs from P_1 or P_0' differs from P_1'. The result considers a distribution $\mathbf{D}$ over $(S, S', P_0, P_0', P_1, P_1', z)$ where S, S' are sets, P_0, P_1 are permutations on the same domain, P_0', P_1' are permutations on the same domain, and $z \in \{0,1\}^*$ is some auxiliary information. Such a $\mathbf{D}$ is *valid* if $P_0(x) = P_1(x)$ for all $x \notin S$ and $P_0'(x) = P_1'(x)$ for all $x \notin S'$. Now consider the game $\mathsf{G}^{\mathsf{dist}}_{\mathbf{D},b}$ shown in Fig. 2. In it, an adversary $\mathcal{A}$ is given z and tries to determine which of the oracle pairs it has access to. We define $\mathsf{Adv}^{\mathsf{dist}}_{\mathbf{D}}(\mathcal{A}) = \mathsf{Pr}\left[\mathsf{G}^{\mathsf{dist}}_{\mathbf{D},1}(\mathcal{A})\right] - \mathsf{Pr}\left[\mathsf{G}^{\mathsf{dist}}_{\mathbf{D},0}(\mathcal{A})\right]$.

The game $\mathsf{G}^{\mathsf{guess}}_{\mathbf{D}}(\mathcal{A})$ in the same figure measures the ability of $\mathcal{A}$ to query its oracles on inputs at which P_0 and P_1 (or P_0' and P_1') differ. It assumes that the adversary makes at most q oracle queries. The adversary is halted in its execution on making a random one of these queries and the input to this query is measured. If the input falls in the appropriate set S or S', then the game returns `true`. Thus we can roughly think of this as a game in which $\mathcal{A}$ is trying to guess a point on which the two oracles differ. We define $\mathsf{Adv}^{\mathsf{guess}}_{\mathbf{D}}(\mathcal{A}) = \mathsf{Pr}[\mathsf{G}^{\mathsf{guess}}_{\mathbf{D}}(\mathcal{A})]$, which leads to a bound on $\mathsf{Adv}^{\mathsf{dist}}_{\mathbf{D}}(\mathcal{A})$.

Theorem 1 ([3], Thm.3). *Let $\mathbf{D}$ be a valid distribution and $\mathcal{A}$ be an adversary making at most q oracle queries. Then $\mathsf{Adv}^{\mathsf{dist}}_{\mathbf{D}}(\mathcal{A}) \leqslant 2q\sqrt{\mathsf{Adv}^{\mathsf{guess}}_{\mathbf{D}}(\mathcal{A})}$.*

Our statement of the theorem differs from the result as given in [3] in that we consider arbitrary permutations, rather than permutations of the form $f[\oplus]$ for some function f, and we provide the attacker

with access to two oracles rather than one.[7] These are simply notational conveniences to match how we will be applying the theorem. The proof given in [3] suffices to establish this variant without requiring any meaningful modifications.

The most natural applications of this theorem would apply it to distributions $\mathbf{D}$ for which the guessing advantage $\mathsf{Adv}^{\mathsf{guess}}_{\mathbf{D}}(\mathcal{A})$ is small for *any* efficient adversary $\mathcal{A}$. This will indeed be the case for our use of it in our Theorem 2. However, note that it can also be applied more broadly with a distribution $\mathbf{D}$ where it is not necessarily difficult to guess inputs on which the oracles differ. We will do so at the end of our proof of Theorem 3. Here we will use a *deterministic* $\mathbf{D}$ so, in particular, the sets S and S' are a priori fixed and not hard to query. The trick we will use to profitably apply the O2H result is to exploit knowledge of the particular form that $\mathcal{A}$ will take (it will be a reduction adversary internally simulating the view of another adversary) to provide a useful bound on its guessing advantage $\mathsf{Adv}^{\mathsf{guess}}_{\mathbf{D}}(\mathcal{A})$.

3 The FX Construction

The FX construction (originally introduced by Kilian and Rogaway [20] as a generalization of Rivest's DESX construction) is a keylength extension for blockciphers. In this construction, an additional key is used which is xor-ed with input and the output of the blockcipher.[8] Formally, given a blockcipher $\mathsf{E} \in \mathsf{Ics}(\mathsf{E.kl}, \mathsf{E.bl})$, the blockcipher $\mathsf{FX}[\mathsf{E}]$ is defined by $\mathsf{FX}[\mathsf{E}](K_1 \,\|\, K_2, x) = \mathsf{E}_{K_1}(x \oplus K_2) \oplus K_2$. Here $|K_1| = \mathsf{E.kl}$ and $|K_2| = \mathsf{E.bl}$ so $\mathsf{FX}[\mathsf{E}].\mathsf{kl} = \mathsf{E.kl} + \mathsf{E.bl}$ and $\mathsf{FX}[\mathsf{E}].\mathsf{bl} = \mathsf{E.bl}$. Its inverse can similarly be computed as $\mathsf{FX}[\mathsf{E}]^{-1}(K_1 \,\|\, K_2, x) = \mathsf{E}^{-1}_{K_1}(x \oplus K_2) \oplus K_2$. Let $k = \mathsf{E.kl}$ and $n = \mathsf{E.bl}$.

Kilian and Rogaway [19] analyzed the PRP security of FX against classical attacks, showing that $\mathsf{Adv}^{\mathsf{sprp}}_{\mathsf{FX}}(\mathcal{A}) \leqslant 2pq/2^{k+n}$ where q is the number of $\textsc{Ev}, \textsc{Inv}$ queries and p is the number of $\mathbf{E}, \mathbf{E}^{-1}$ queries made by $\mathcal{A}$ (with E modeled as an ideal cipher). In [24], Leander and May showed a quantum attack against the FX construction – establishing that the added whitening keys did not provide additionally security. This attack uses a clever combination of the quantum algorithms of Grover [12] and Simon [29]. It was inspired by an attack by Kuwakado and Morii [23] showing that the Even-Mansour blockcipher [11] provides no quantum security. Thus, it seems that $\mathsf{FX}[\mathsf{E}]$ does not provide meaningfully more security than E against quantum attackers.

However, the attack of Leander and May requires quantum access to both the FX construction and the underlying blockcipher E. This raises the question of whether the FX is actually an effective key-length extension technique in the partially-quantum setting where the adversary performs only classical queries to the construction oracles. In this section, we approach this question from two directions. First, in Section 3.1 we apply Theorem 1 with a careful representation of the real and ideal worlds to show that FX does indeed achieve improved security against non-adaptive attacks.

Analyzing the full adaptive security of FX against classical construction queries seems beyond the capabilities of current proof techniques. Accordingly, in Section 3.2, we consider a variant of FX in which a random oracle is used in place of the ideal cipher and prove its quantum PRF security. Here we apply a new reduction technique (built on the "sparse" quantum representation of a random function introduced by Zhandry [35] and Theorem 1, the O2H theorem from Ambainis, Hamburg, and Unruh [3]) to prove that this serves as an effective key-length extension technique in our setting. It seems likely that our technique could be extended to the normal FX construction, should an appropriate sparse quantum representation of random permutations be discovered.

3.1 Security of FX Against Non-Adaptive Attacks

The following theorem bounds the security of the FX construction against non-adaptive attacks (in which the non-adaptive queries are all classical). This result is proven via a careful use of Theorem 1 in which the

[7] Their result additionally allows the adversary to make oracle queries in parallel and bounds its advantage in terms of the "depth" of its oracle queries rather than the total number of queries. We omit this for simplicity.

[8] Technically, the original definition of FX [20] uses distinct keys for xor-ing with the input and the output, but this would not provide any benefit in our concrete security analysis so we focus on the simplified construction.

Distribution $\mathbf{D}$

// **Step 1: Sample responses to construction queries**

For $i = 1, \ldots, q'$ do

  $Y_i \leftarrow\!\!\$\ \{0,1\}^n \backslash \{Y_1, \ldots, Y_{i-1}\}$

  $T[M_i] \leftarrow Y_i$; $T^{-1}[Y_i] \leftarrow M_i$

For $i = q'+1, \ldots, q$ do

  If $T^{-1}[Y_i] \neq \perp$ then $M_i \leftarrow T^{-1}[Y_i]$

  Else $M_i \leftarrow\!\!\$\ \{0,1\}^n \backslash \{M_1, \ldots, M_{i-1}\}$

  $T[M_i] \leftarrow Y_i$; $T^{-1}[Y_i] \leftarrow M_i$

$z \leftarrow (T, T^{-1})$

// **Step 2: Sample f_0 as independent ideal cipher**

$f_0 \leftarrow\!\!\$\ \mathsf{Ics}(k, n)$

// **Step 3: Reprogram f_1 for consistency with construction queries**

$K_1 \leftarrow\!\!\$\ \{0,1\}^k$; $K_2 \leftarrow\!\!\$\ \{0,1\}^n$

$\mathcal{I} \leftarrow \{M_i \oplus K_2 : 1 \leqslant i \leqslant q\}$; $\mathcal{O} \leftarrow \{Y_i \oplus K_2 : 1 \leqslant i \leqslant q\}$

$\mathcal{I}' \leftarrow \{f_0^{-1}(K_1, y) : y \in \mathcal{O}\}$; $\mathcal{O}' \leftarrow \{f_0(K_1, x) : x \in \mathcal{I}\}$

$S = \{(K_1, x) : x \in \mathcal{I} \cup \mathcal{I}'\}$; $S' = \{(K_1, y) : y \in \mathcal{O} \cup \mathcal{O}'\}$

For $(K, x) \notin S$ do

  $f_1(K, x) \leftarrow f_0(K, x)$

For $i = 1, \ldots, q$ do

  $f_1(K_1, M_i \oplus K_2) \leftarrow Y_i \oplus K_2$

For $x \in \mathcal{I}' \backslash \mathcal{I}$ do

  $f_1(K_1, x) \leftarrow\!\!\$\ \mathcal{O}' \backslash \{f_1(K_1, x) : x \in \mathcal{I} \cup \mathcal{I}', f_1(K_1, x) \neq \perp\}$

Return $(S, S', f_0[\oplus], f_0^{-1}[\oplus], f_1[\oplus], f_1^{-1}[\oplus], z)$

Fig. 3. Distribution of oracles used in proof of Theorem 2.

distribution $\mathbf{D}$ is defined in terms of the non-adaptive queries that the adversary will make and defined so as to perfectly match the two worlds that $\mathcal{A}$ is attempting to distinguish between.

Theorem 2. *Let $\mathcal{A}$ be a quantum adversary which makes at most q classical, non-adaptive queries to* Ev, Inv *and consider* $\mathsf{FX}[\cdot]$ *with the underlying blockcipher modeled by an ideal cipher drawn from* $\mathsf{Ics}(k, n)$*. Then*

$$\mathsf{Adv}^{\mathsf{sprp\text{-}na}}_{\mathsf{FX}}(\mathcal{A}) \leqslant \sqrt{8p^2q/2^{k+n}},$$

where p is the number of quantum oracle queries that $\mathcal{A}$ makes to the ideal cipher.

Proof. We will use Theorem 1 to prove this result, so first we define a distribution $\mathbf{D}$. Suppose that $M_1, \ldots, M_{q'} \in \{0,1\}^n$ are the distinct queries $\mathcal{A}$ will make to Ev and $Y_{q'+1}, \ldots, Y_q \in \{0,1\}^n$ are the distinct queries that $\mathcal{A}$ will make to Inv. The order in which these queries will be made does not matter. Then we define $\mathbf{D}$ as shown in Fig. 3. This distribution is valid (as required for Theorem 1) because G_1 is reprogrammed to differ from G_0 by making inputs in S map to different values in S'.

We will show that the oracles output by this distribution (described in words momentarily) can be used to perfectly simulate the views expected by $\mathcal{A}$. In particular, let $\mathcal{A}'$ be an adversary (for $\mathsf{G}^{\mathsf{dist}}_{\mathbf{D}}$) which runs $\mathcal{A}$, responding to $\textsc{Ev}(M_i)$ queries with $T[M_i]$, responding to $\textsc{Inv}(Y_i)$ queries with $T^{-1}[Y_i]$, and simulating $\mathbf{E}, \mathbf{E}^{-1}$ with its own oracles $f_b[\oplus], f_b^{-1}[\oplus]$. When $\mathcal{A}$ halts with output b, this adversary halts with the same output. We claim that (i) $\mathsf{Pr}\left[\mathsf{G}^{\mathsf{sprp}}_{\mathsf{F},1}(\mathcal{A})\right] = \mathsf{Pr}\left[\mathsf{G}^{\mathsf{dist}}_{\mathbf{D},1}(\mathcal{A}')\right]$ and (ii) $\mathsf{Pr}\left[\mathsf{G}^{\mathsf{sprp}}_{\mathsf{F},0}(\mathcal{A})\right] = \mathsf{Pr}\left[\mathsf{G}^{\mathsf{dist}}_{\mathbf{D},0}(\mathcal{A}')\right]$. This gives $\mathsf{Adv}^{\mathsf{sprp\text{-}na}}_{\mathsf{F}}(\mathcal{A}) = \mathsf{Adv}^{\mathsf{dist}}_{\mathbf{D}}(\mathcal{A}')$.

Claim (ii) follows by noting that the view of $\mathcal{A}$ is identical when run by $\mathsf{G}^{\mathsf{sprp}}_0$ or by $\mathcal{A}'$ in $\mathsf{G}^{\mathsf{dist}}_{\mathbf{D},0}$. In $\mathsf{G}^{\mathsf{sprp}}_0$, its construction queries are answered with the random permutation F. When it is run by $\mathcal{A}'$ in $\mathsf{G}^{\mathsf{dist}}_{\mathbf{D},0}$, these queries are answered with the tables T and T^{-1} which can be viewed as having just lazily sampled enough of

a random permutation to respond to the given queries. In both cases, its primitive oracle is an independently chosen ideal cipher.

Claim (i), follows by noting that the view of $\mathcal{A}$ is identical when run by $\mathsf{G}_1^{\mathsf{sprp}}$ or by $\mathcal{A}'$ in $\mathsf{G}_{\mathbf{D},1}^{\mathsf{dist}}$. In $\mathsf{G}_1^{\mathsf{sprp}}$, construction queries are answered by the FX construction using the ideal cipher and keys K_1, K_2. In the distribution, we first sample the responses to the construction queries and then construct the ideal cipher f_1 by picking K_1 and K_2 and setting f_1 to equal f_0 except for places where we reprogram it to be consistent with these construction queries. The construction will map M_i to Y_i for each i, which means that the condition $f_1(K_1, M_i \oplus K_2) = Y_i \oplus K_2$ should hold. These inputs and outputs are stored in the sets $\mathcal{I}$ and $\mathcal{O}$. The sets $\mathcal{I}'$ and $\mathcal{O}'$ store the inputs mapping to $\mathcal{O}$ and outputs mapped to $\mathcal{I}$ by $f_0(K_1, \cdot)$, respectively. Thus while making the above condition hold, we additionally reprogram f_1 so that elements of $\mathcal{I}'\backslash\mathcal{I}$ map to (random, non-repeating) elements of $\mathcal{O}'\backslash\mathcal{O}$.

In particular, the uniformity of f_0 ensures that the map induced by $f_1(K_1, \cdot)$ between $\{0,1\}^n\backslash(\mathcal{I} \cup \mathcal{I}')$ and $\{0,1\}^n\backslash(\mathcal{O} \cup \mathcal{O}')$ is a random bijection. The last for loop samples a random bijection between $\mathcal{I}'\backslash\mathcal{I}$ and $\mathcal{O}'\backslash\mathcal{O}$. Because there are no biases in which values fall into these two cases among those of $\{0,1\}^n\backslash\mathcal{I}$ and $\{0,1\}^n\backslash\mathcal{O}$, this means the map between these two sets is a uniform bijection as desired. A more detailed probability analysis of Claim (i) is given in Appendix A.

Applying the bound on $\mathsf{Adv}_{\mathbf{D}}^{\mathsf{dist}}(\mathcal{A}')$ from Theorem 1 gives us

$$\mathsf{Adv}_{\mathsf{F}}^{\mathsf{sprp\text{-}na}}(\mathcal{A}) \leqslant 2p\sqrt{\mathsf{Pr}[\mathsf{G}_{\mathbf{D}}^{\mathsf{guess}}(\mathcal{A}')]}$$

so we complete the proof by bounding this probability. Let fw denote the event that the i-th query of $\mathcal{A}'$ in $\mathsf{H}_0^{\mathbf{D}}$ is to f_0 and let (K, x) denote the measured value of this query so that

$$\mathsf{Pr}\big[\mathsf{H}_{\mathbf{D}}^{\mathsf{guess}}(\mathcal{A}')\big] = \mathsf{Pr}[\mathsf{fw}] \cdot \mathsf{Pr}[(K,x) \in S \mid \mathsf{fw}] + \mathsf{Pr}[\neg\mathsf{fw}] \cdot \mathsf{Pr}\big[(K,x) \in S' \mid \neg\mathsf{fw}\big].$$

A union bound over the different elements of S gives

$$\begin{aligned}\mathsf{Pr}[(K,x) \in S \mid \mathsf{fw}] \leqslant &\sum_{j=1}^{q} \mathsf{Pr}[K = K_1 \wedge x \oplus M_j = K_2 \mid \mathsf{fw}] \\ &+ \sum_{j=1}^{q} \mathsf{Pr}[K = K_1 \wedge G_0(K_1, x) \oplus Y_j = K_2 \mid \mathsf{fw}].\end{aligned}$$

However, note that the view of $\mathcal{A}$ in $\mathsf{H}_0^{\mathbf{D}}$ is independent of K_1 and K_2 so we get that

$$\mathsf{Pr}[(K,x) \in S \mid \mathsf{fw}] \leqslant 2q/2^{k+n}.$$

Applying analogous analysis to the $\neg\mathsf{fw}$ case gives

$$\mathsf{Pr}\big[(K,x) \in S' \mid \neg\mathsf{fw}\big] \leqslant 2q/2^{k+n}$$

and hence $\mathsf{Pr}\Big[\mathsf{H}_0^{\mathbf{D}}(\mathcal{A}')\Big] \leqslant 2q/2^{k+n}$. Plugging this into our earlier inequality gives the stated bound. □

3.2 Adaptive Security of FFX

In this section we will prove the security of FFX (a variant of FX using a random oracle in place of the ideal cipher) against quantum adversaries making strictly classical queries to the construction.

Formally, given a function family $\mathsf{F} \in \mathsf{Fcs}(\mathsf{F.kl}, \mathsf{F.il}, \mathsf{F.ol})$, we model the FFX construction by the function family $\mathsf{FFX[F]}$ by $\mathsf{FFX[F]}(K_1 \,\|\, K_2, x) = \mathsf{F}_{K_1}(x \oplus K_2)$.[9] Here $|K_1| = \mathsf{F.kl}$ and $|K_2| = \mathsf{F.il}$ so $\mathsf{FFX[F].kl} = \mathsf{F.kl} + \mathsf{F.il}$, $\mathsf{FFX[F].il} = \mathsf{F.il}$, and $\mathsf{FFX[F].ol} = \mathsf{F.ol}$. Let $k = \mathsf{F.kl}$, $n = \mathsf{F.il}$, and $m = \mathsf{F.ol}$.

[9] The outer xor by K_2 used in FX is omitted because it is unnecessary for our analysis.

Theorem 3. *Let $\mathcal{A}$ be an order consistent quantum adversary which makes classical queries to* Ev *and consider* $\mathsf{FFX}[\cdot]$ *with the underlying function family modeled by a random oracle drawn from* $\mathsf{Fcs}(k, n, m)$*. Then*

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{FFX}}(\mathcal{A}) \leqslant \sqrt{\frac{8(p+q)pq}{2^{k+n}}},$$

where p is the number of quantum oracle queries that $\mathcal{A}$ makes to the random oracle and q is the number of queries it makes to Ev*.*

We can reasonably assume that $p > q$ so the dominant behavior of the above expression is $O\left(\sqrt{p^2q/2^{k+n}}\right)$.

The proof of this result proceeds via a sequence of hybrids which gradually transition from the real world of $\mathsf{G}^{\mathsf{prf}}_{\mathsf{FFX}}$ to the ideal world. Crucial to this sequence of hybrids are the technique of Zhandry [35] which, by viewing a space under dual bases, allows one to simulate a random function using a sparse representation table and to "record" the queries to the function. For the ideal world, we can represent the random oracle and the random function underlying Ev independently using such sparse representation tables. With some careful modification, we are also able to represent the real world's random oracle using a similar pair of sparse representation tables as if it were two separate functions. However, in this case, the tables will be slightly non-independent in that if the adversary queries Ev on an input x and the random oracle on $(K_1, x \oplus K_2)$ then the results of the latter query is stored in the Ev table, rather than the random oracle table. Beyond this minor consistency check (which we are only able to implement because the queries to Ev are classical and so can be stored by simulation), the corresponding games are identical. Having done this rewriting, we can carefully apply Theorem 1 to bound the ability of an adversary to distinguish between the two worlds by its ability to trigger this consistency check.

As mentioned in Section 2.1, our application of Theorem 1 here is somewhat atypical. Our distribution over functions $\mathbf{D}$ will be deterministic, but we are able to still extract a meaningful bound from this by taking advantage of our knowledge of the particular behavior of the adversary we apply Theorem 1 with.

Proof. In this proof we will consider a sequence of hybrid games H_0 through H_9. Of these games we will establish the following claims.

1. $\mathsf{Pr}\left[\mathsf{G}^{\mathsf{prf}}_{\mathsf{FFX},0}(\mathcal{A})\right] = \mathsf{Pr}[\mathsf{H}_0] = \mathsf{Pr}[\mathsf{H}_1] = \mathsf{Pr}[\mathsf{H}_2] = \mathsf{Pr}[\mathsf{H}_3]$
2. $\mathsf{Pr}\left[\mathsf{G}^{\mathsf{prf}}_{\mathsf{FFX},1}(\mathcal{A})\right] = \mathsf{Pr}[\mathsf{H}_9] = \mathsf{Pr}[\mathsf{H}_8] = \mathsf{Pr}[\mathsf{H}_7] = \mathsf{Pr}[\mathsf{H}_6] = \mathsf{Pr}[\mathsf{H}_5] = \mathsf{Pr}[\mathsf{H}_4]$
3. $\mathsf{Pr}[\mathsf{H}_4] - \mathsf{Pr}[\mathsf{H}_3] \leqslant \sqrt{8(p+q)pq/2^{\mathsf{F.kl}+\mathsf{F.il}}}$

Combining these claims gives the desired result.

In formally defining our hybrids we write the computation to be performed using the following quantum registers.

- W: The workspace of $\mathcal{A}$. The adversary's final output is written into $W[1]$.
- K: The k-qubit register (representing the function *key/index*) $\mathcal{A}$ uses when making oracle queries to the random oracle.
- X: The n-qubit register (representing function inputs) used when making oracle queries to the random oracle or Ev.
- Y: The m-qubit register (representing function outputs) into which the results of oracle queries are written.
- H: The $2^{k+n} \cdot m$-qubit register which stores the function defining the random oracle (initially via its truth table).
- F: The $2^n \cdot m$-qubit register which stores the function defining Ev.
- K_1: The k-qubit register which stores the first key of the construction.
- K_2: The n-qubit register which stores the second key of the construction.
- I: The $\lceil \log p \rceil$-qubit register which tracks how many Ev queries $\mathcal{A}$ has made.
- $\vec{X} = (\vec{X}_1, ..., \vec{X}_p)$: The p n-qubit registers used to store the classical queries that $\mathcal{A}$ makes to Ev.

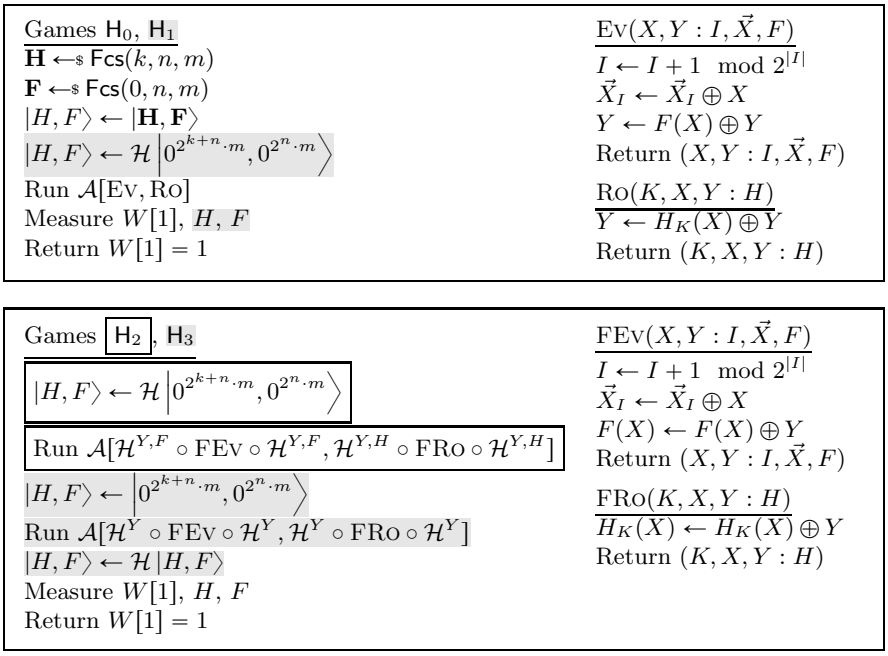

Fig. 4. Hybrid games H_0 through H_3 for the proof of Theorem 3 which are equivalent to the ideal world of $\mathsf{G}^{\mathsf{prf}}_{\mathsf{FFX}}$. Highlighted or boxed code is only included in the correspondingly highlighted or boxed game.

We start by changing our perspective. A quantum algorithm that makes classical queries to Ev can be modeled by thinking of a quantum algorithm that measures its X register immediately before the query. (Because the behavior of Ev is completely classical at this point, we do not need to measure the Y register as well.) Measuring the register X is indistinguishable from using a CNOT operation to copy it into a separate register (i.e. xor-ing X into the previously empty register $\vec{X}_I$ that will never again be modified). By incorporating this CNOT operation into the behavior of our hybrid game, we treat $\mathcal{A}$ as an attacker that makes fully quantum queries to its oracles in the hybrid game. We think of $\mathcal{A}$ as deferring all of measurements until the end of its computation. Because all that matters is its final output $W[1]$ we can have the game measure just that register and assume that $\mathcal{A}$ does not internally make any measurements. The principle of deferred measurement ensures that the various changes discussed here do not change the behavior of $\mathcal{A}$. This perspective change lets us use purely quantum analysis, rather than mixing quantum and classical.

Claim 1. We start by considering the hybrids H_0 through H_3, defined in Fig. 4 which are all identical to the ideal world of $\mathsf{G}^{\mathsf{prf}}_{\mathsf{FFX}}$. In these transitions we are applying the ideas of Zhandry [35] to transition to representing the random functions stored in H and F by an all zeros table which is updated whenever the adversary makes a query.

The hybrid H_0 is mostly just $\mathsf{G}^{\mathsf{prf}}_{\mathsf{FFX}}$ rewritten to use the registers indicated above. So $\mathsf{Pr}\left[\mathsf{G}^{\mathsf{prf}}_{\mathsf{FFX},0}(\mathcal{A})\right] = \mathsf{Pr}[\mathsf{H}_0]$ holds.

Next consider H_1 which differs from H_0 only in the grey highlighted code which initializes H and F in the uniform superposition and then measures them at the end of execution. (Recall that the Hadamard transform applied to the all zeros state gives the uniform superposition.) Note that these register control, but are unaffected by the oracles Ev and Ro. Because they are never modified while $\mathcal{A}$ is executing, the principle of deferred measurement tells us that this modification is undetectable by $\mathcal{A}$, giving $\mathsf{Pr}[\mathsf{H}_0] = \mathsf{Pr}[\mathsf{H}_1]$

Next consider H_2 which contains the boxed, but not the highlighted, code. This game uses the oracles FEv and FRo, the Fourier versions of Ev and Ro, which xor the Y value of $\mathcal{A}$'s query into the the register F or H. Note that $\mathcal{A}$'s access to these oracles is mitigated by $\mathcal{H}^{Y,F}$ on each query. The superscript here indicate that

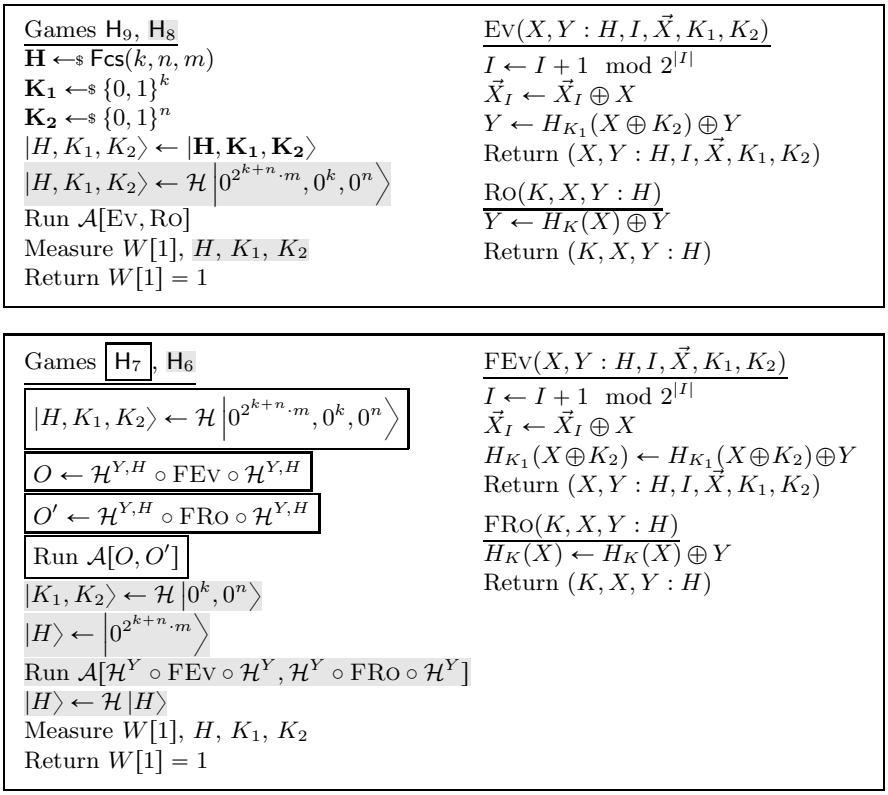

Fig. 5. Hybrid games H_9 through H_6 for the proof of Theorem 3 which are equivalent to the real world of $\mathsf{G}^{\mathsf{prf}}_{\mathsf{FFX}}$. Highlighted or boxed code is only included in the correspondingly highlighted or boxed game.

the Hadamard transform is being applied to the registers Y and F. We have that $\mathcal{H}^{Y,F}\circ\textsc{FEv}\circ\mathcal{H}^{Y,F} = \textsc{Ev}$ and $\mathcal{H}^{Y,H}\circ\textsc{FRo}\circ\mathcal{H}^{Y,H} = \textsc{Ro}$ both hold.[10] So $\mathsf{Pr}[\mathsf{H}_1] = \mathsf{Pr}[\mathsf{H}_2]$ because the adversary's oracles are identical.

Next consider H_3 which contains the highlighted, but not the boxed, code. For this transition, recall that $\mathcal{H}\circ\mathcal{H}$ is the identity operator. So to transition to this game we can cancel the $\mathcal{H}$ operator used to initialize H with the $\mathcal{H}^H$ operator applied before $\mathcal{A}$'s first FRO oracle query. Similarly, we can cancel the $\mathcal{H}^H$ operation performed after any (non-final) FRO query with the $\mathcal{H}^H$ operation performed before the next FRO query. Finally, the $\mathcal{H}^H$ operation that would be performed after the final FRO query is instead delayed to be performed immediately before H is measured. (We could have omitted this operation and measurement entirely because all that matters at that point is the measurement of $W[1]$.) The $\mathcal{H}$ operators on F are similarly changed. Because $\mathcal{A}$ does not have access to the H and F registers, we can indeed commute the $\mathcal{H}$ operators with $\mathcal{A}$ in this manner without changing behavior. Hence $\mathsf{Pr}[\mathsf{H}_2] = \mathsf{Pr}[\mathsf{H}_3]$, as desired. Note that H and F independently store tables which are initialized to all zeros and then written into by the adversary's queries.

CLAIM 2. We now consider the hybrids H_4 through H_9 (starting from H_9), which are defined in Fig. 5 and Fig. 6 using similar ideas as in the transition from H_0 through H_3. As we will justify, these games are all equivalent to the real world of $\mathsf{G}^{\mathsf{prf}}_{\mathsf{FFX}}$.

First H_9 rewrote the real world of $\mathsf{G}^{\mathsf{prf}}_{\mathsf{FFX}}$ to use our specified registers and to record queries into $\vec{X}$ in EV. Then in H_8, rather than sampling H, K_1, and K_2 uniformly at the beginning of the game we put them in the uniform superposition and measure them at the end of the game. For H_7 we replace our oracles that xor

[10] This follows as a consequence of the following. Let $U_\oplus$ and $U'_\oplus$ be the unitaries which for $y,z\in\{0,1\}$ are defined by $U_\oplus|y,z\rangle = |y\oplus z,z\rangle$ and $U'_\oplus|y,z\rangle = |y,y\oplus z\rangle$. Then $\mathcal{H}\circ U_\oplus\circ\mathcal{H} = U'_\oplus$.

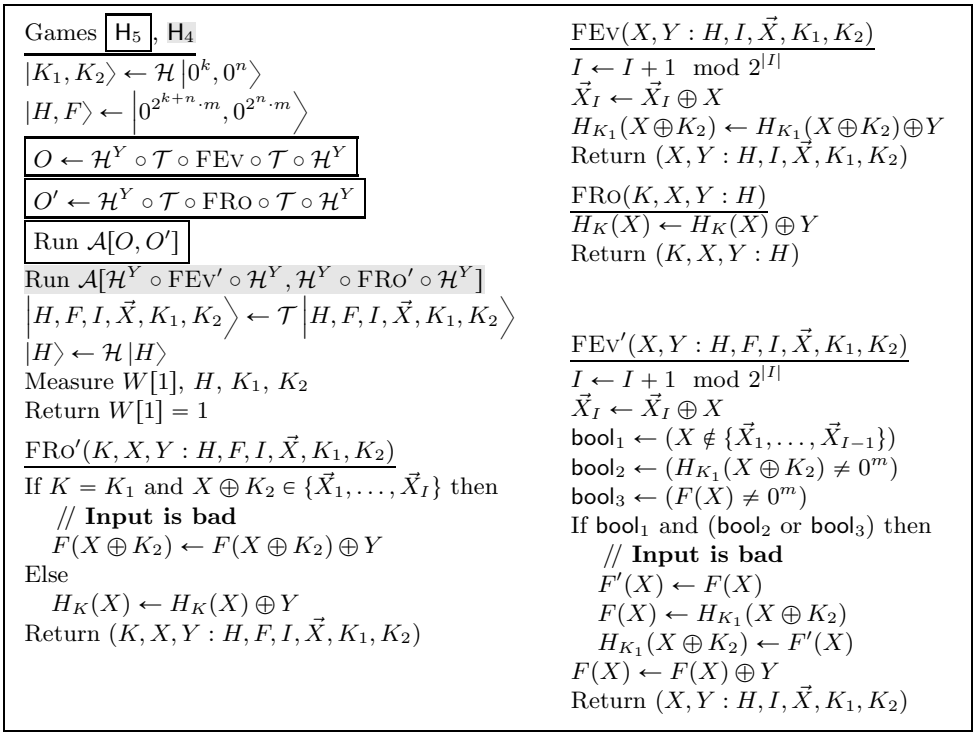

Fig. 6. Hybrid games H_5 and H_4 for the proof of Theorem 3 which are equivalent to the real world of $\mathsf{G}^{\mathsf{prf}}_{\mathsf{FFX}}$. Highlighted or boxed code is only included in the correspondingly highlighted or boxed game. Unitary $\mathcal{T}$ is define in the text.

into the adversary's Y register with oracles that xor into the H register using Hadamard operations, some of which we then cancel out to transition to H_6. The same arguments from Claim 1 of why these sorts of modifications do not change the behavior of the game apply here and so $\mathsf{Pr}\left[\mathsf{G}^{\mathsf{prf}}_{\mathsf{FFX},1}(\mathcal{A})\right] = \mathsf{Pr}[\mathsf{H}_9] = \mathsf{Pr}[\mathsf{H}_8] = \mathsf{Pr}[\mathsf{H}_7] = \mathsf{Pr}[\mathsf{H}_6]$.

Our next transitions are designed to make the current game's oracles identical with those of H_3, except on some "bad" inputs. In H_6 we have a single all zeros table H which gets written into by queries that $\mathcal{A}$ makes to either of its oracles, while in H_3 the oracles separately wrote into either H or F. For H_5 we will similarly separate the single table H into separate tables H and F. However, we cannot keep them completely independent, because if the adversary queries FEv with $X = x$ and FRo with $(K, X) = (K_1, x \oplus K_2)$ then both of these operations would be writing into the same table location in H_6. Consider the following unitary $\mathcal{T}$ which acts on registers H and F and is controlled by the registers I, $\vec{X}$, K_1, and K_2. We will think of this unitary as transitioning us between a representation of H as a single table (with an all-zero F table) and a representation of it divided between H and F.

$\mathcal{T}(H, F, I, \vec{X}, K_1, K_2)$
For $x \in \{\vec{X}_1, \ldots, \vec{X}_I\}$ do
$F'(x) \leftarrow F(x)$
$F(x) \leftarrow H_{K_1}(x \oplus K_2)$
$H_{K_1}(x \oplus K_2) \leftarrow F'(x)$
Return $(H, F, I, \vec{X}, K_1, K_2)$

In words, $\mathcal{T}$ swaps $F(x)$ and $H_{K_1}(x \oplus K_2)$ for each x that has been previous queried to FEv (as stored by $\vec{X}$ and I). Note that $\mathcal{T}$ is its own inverse. In H_5 we (i) initialize the table F as all zeros, (ii) perform $\mathcal{T}$ before and after each oracle query, and (iii) perform $\mathcal{T}$ after $\mathcal{A}$ has executed. We verify that H has the same

value in H_5 that it would have had in H_6 during each oracle query and at the end before measurement. The application of $\mathcal{T}$ before the first oracle query does nothing (because $I = 0$) so H is all zeros for this query as required. As we've seen previously with $\mathcal{H}$, we can commute $\mathcal{T}$ with the operations of $\mathcal{A}$ because $\mathcal{T}$ only acts on registers outside of the adversary's control. We can similarly commute $\mathcal{T}$ with $\mathcal{H}^Y$. Hence the $\mathcal{T}$ operation after every non-final oracle query can be seen to cancel with the $\mathcal{T}$ operation before the following oracle query. The $\mathcal{T}$ operation after the final oracle query cancels with the $\mathcal{T}$ operation performed after $\mathcal{A}$ halts execution. Hence, $\mathsf{Pr}[\mathsf{H}_6] = \mathsf{Pr}[\mathsf{H}_5]$ as claimed.

For the transition to H_4 let us dig into how our two-table representation in H_5 works so that we can incorporate the behavior of $\mathcal{T}$ directly into the oracles. For simplicity of notation in discussion, let $\widetilde{H}(x)$ denote $H_{K_1}(x \oplus K_2)$. First note that in between oracle queries the two tables representation of H_5 will satisfy the property that for each $x \in \{\vec{X}_1, \ldots, \vec{X}_I\}$ we will have $\widetilde{H}(x) = 0^{\mathsf{F.ol}}$ and for all other x we will have that $F(x) = 0^{\mathsf{F.ol}}$.[11] This is the case because after each query we have applied $\mathcal{T}$ to an H which contains the same values it would have in H_6 and an F which is all zeros.

Now consider when a $\mathcal{T} \circ \mathrm{FEv} \circ \mathcal{T}$ query is executed with some X and Y. If $X \in \{\vec{X}_1, \ldots, \vec{X}_I\}$, then $F(X)$ and $\widetilde{H}(X)$ are swapped, Y is xored into $\widetilde{H}(X)$, then finally $F(X)$ and $\widetilde{H}(X)$ are swapped back. Equivalently, we could have xored Y directly into $F(X)$ and skipped the swapping around. If $X \notin \{\vec{X}_1, \ldots, \vec{X}_I\}$, then Y is xored into $\widetilde{H}(X)$ before $F(X)$ and $\widetilde{H}(X)$ are swapped. If $\widetilde{H}(X) = 0^{F.\mathsf{ol}}$ beforehand, then we could equivalently have xored Y directly into $F(X)$ and skipped the swapping around (because $F(X) = 0^{\mathsf{F.ol}}$ must have held from our assumption on X). If $\widetilde{H}(X) \neq 0^{F.\mathsf{ol}}$ beforehand, then we could equivalently could have swapped $\widetilde{H}(X)$ and $F(X)$ first, then xored Y into $F(X)$. The equivalent behavior we described is exactly the behavior captured by the oracle FEv' which is used in H_4 in place of $\mathcal{T} \circ \mathrm{FEv} \circ \mathcal{T}$. It checks if $X \notin \{\vec{X}_1, \ldots, \vec{X}_I\}$ (bool_1) and $\widetilde{H}(X) \neq 0^m$ (bool_2), performing a swap if so. Then Y is xored into $F(X)$. A swap is also performed if $X \notin \{\vec{X}_1, \ldots, \vec{X}_{I-1}\}$ and $F(X) \neq 0^m$, however this case is impossible from our earlier observation that $F(x) = 0^m$ when X is not in $\vec{X}$. This second case was added only to ensure that FEv' is a permutation.

Similarly, consider when a $\mathcal{T} \circ \mathrm{FRo} \circ \mathcal{T}$ query is executed with some K, X, Y. Any swapping done by $\mathcal{T}$ uses H_{K_1}, so when $K \neq K_1$ this just xors Y into $H_K(X)$. If $X \oplus K_2$ is not in $\{\vec{X}_1, \ldots, \vec{X}_I\}$, then $H_K(X)$ is unaffected by the swapping so again this just xors Y into $H_K(X)$. When $K = K_1$ and $X \oplus K_2 \in \{\vec{X}_1, \ldots, \vec{X}_I\}$, first $H_K(X)$ would have been swapped with $F(X \oplus K_2)$, then Y would be xored into $H_K(X)$, then $H_K(X)$ and $F(X \oplus K_2)$ would be swapped back. Equivalently, we could just have xored Y into $F(X \oplus K_2)$ and skipped the swapping. This behavior we described is exactly the behavior captured by the oracle FRo' which is used in H_4 in place of $\mathcal{T} \circ \mathrm{FRo} \circ \mathcal{T}$.

We have just described that on the inputs we care about FEv' behaves identically to $\mathcal{T} \circ \mathrm{FEv} \circ \mathcal{T}$ and FRo' behaves identically to $\mathcal{T} \circ \mathrm{FRo} \circ \mathcal{T}$. Hence $\mathsf{Pr}[\mathsf{H}_5] = \mathsf{Pr}[\mathsf{H}_4]$, completing this claim.

Claim 3. To compare hybrids H_3 and H_4 we will note their oracles only differ on a small number of inputs (in particular those labelled as bad by comments in our code) and then apply Theorem 1 to bound the difference between them. To aid in this we have rewritten them as $\widetilde{\mathsf{H}}_3$ and $\widetilde{\mathsf{H}}_4$ in Fig. 7. For both we have removed some operations performed after $\mathcal{A}$ halted its execution for cleanliness because these operations on registers other than $W[1]$ cannot affect the probability that it is measured to equal 1. So we have $\mathsf{Pr}\left[\widetilde{\mathsf{H}}_3\right] = \mathsf{Pr}[\mathsf{H}_3]$ and $\mathsf{Pr}\left[\widetilde{\mathsf{H}}_4\right] = \mathsf{Pr}[\mathsf{H}_4]$.

Let FEv_3 and FRo_3 be the permutations defining the corresponding oracle in $\widetilde{\mathsf{H}}_3$. Define FEv_4 and FRo_4 analogously. These permutations differ only on the inputs we referred to as bad. So let S denote the set of bad $X, H, F, I, \vec{X}, K_1, K_2$ for FEv (i.e. those for which bool_1 and either bool_2 or bool_3 hold). Let S' denote the set of bad $K, X, H, F, I, \vec{X}, K_1, K_2$ for FRo (i.e. those for which $K = K_1$ and $X \oplus K \in \{\vec{X}_1, \ldots, \vec{X}_I\}$). Let $\mathbf{D}$ denote the distribution which always outputs $(S, S', \mathrm{FEv}_3, \mathrm{FRo}_3, \mathrm{FEv}_4, \mathrm{FRo}_4, \varepsilon)$. Clearly this is a valid distribution for Theorem 1 by our choice of S and S'.

Now we can define an adversary $\mathcal{A}'$ for $\mathsf{G}^{\mathsf{dist}}_{\mathbf{D},b}$ which simulates the view of $\mathcal{A}$ in $\widetilde{\mathsf{H}}_{3+b}$ by locally running the code of that hybrid except for during oracle queries when it uses its f_b oracle to simulate FEv and f'_b

[11] More precisely, the corresponding registers hold superpositions over tables satisfying the properties we discuss.

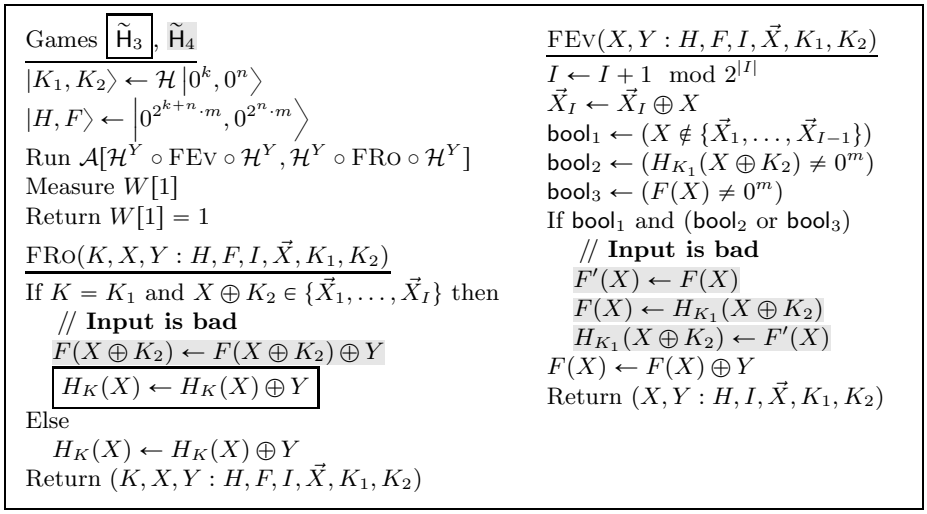

Fig. 7. Hybrid games $\widetilde{\mathsf{H}}_3$ and $\widetilde{\mathsf{H}}_4$ for the proof of Theorem 3 which are rewritten versions of H_3 and H_4 to emphasize that their oracles are identical-until-bad. Highlighted or boxed code is only included in the correspondingly highlighted or boxed game.

oracle to simulate FRo. Because the simulation of these views are perfect we have that

$$\mathsf{Pr}\left[\widetilde{\mathsf{H}}_3\right] - \mathsf{Pr}\left[\widetilde{\mathsf{H}}_4\right] = \mathsf{Adv}^{\mathsf{dist}}_{\mathbf{D}}(\mathcal{A}') \leqslant 2(p+q)\sqrt{\mathsf{Adv}^{\mathsf{guess}}_{\mathbf{D}}(\mathcal{A}')}$$

where the inequality follows from Theorem 1, noting that $\mathcal{A}'$ makes $p+q$ oracle queries.

To complete the proof we bound $\mathsf{Adv}^{\mathsf{guess}}_{\mathbf{D}}(\mathcal{A}')$. In the following probability calculation we use x and i to denote random variables taking on the values the corresponding variables have at the end of an execution of $\mathsf{G}^{\mathsf{guess}}_{\mathbf{D}}(\mathcal{A})$. Let $\mathcal{S}$ denote a random variable which equals S if the measured query is to f_0 and equals S' otherwise. Then conditioning over each possible value of i gives

$$\begin{aligned}\mathsf{Adv}^{\mathsf{guess}}_{\mathbf{D}}(\mathcal{A}') = \mathsf{Pr}[x \in \mathcal{S}] &= \sum_{j=1}^{p+q} \mathsf{Pr}[x \in \mathcal{S} \mid i = j]\mathsf{Pr}[i = j] \\ &= (p+q)^{-1} \sum_{j=1}^{p+q} \mathsf{Pr}[x \in \mathcal{S} \mid i = j].\end{aligned}$$

Because $\mathcal{A}$ is order consistent we can pick disjoint sets E and R with $E \cup R = \{1, \ldots, p+q\}$ such that $i \in E$ means the i-th query is to $\mathcal{A}$'s FEv oracle and $i \in R$ means that the i-th query is to its FRo oracle. Note that $|E| = q$ and $|R| = p$. The view of $\mathcal{A}$ (when run by $\mathcal{A}'$) in $\mathsf{G}^{\mathsf{guess}}_{\mathbf{D}}$ matches its view in $\widetilde{\mathsf{H}}_3$ so, in particular, it is independent of K_1 and K_2. Hence we can think of these keys being chosen at random at the end of execution when analyzing the probability of $x \in \mathcal{S}$.

For a FRo query, the check for bad inputs is if $K_1 = K$ and $K_2 \in \{X \oplus \vec{X}_1, \ldots, X \oplus \vec{X}_I\}$. The variable I is counting the number of FEv queries made so far, so $I < q$. By a union bound,

$$\mathsf{Pr}\left[x \in S' \mid i = j\right] \leqslant q/2^{\mathsf{F.kl}+\mathsf{F.il}}$$

when $j \in R$.

For a FEv query, the check for bad inputs is if $X \notin \{\vec{X}_1, \ldots, \vec{X}_{I-1}\}$ and $H_{K_1}(X \oplus K_2)$ is non-zero.[12] In $\widetilde{\mathsf{H}}_3$, each query to FRo can make a single entry of H non-zero so it will never have more than p non-zero

[12] Note that the other bad inputs, for which $X \notin \{\vec{X}_1, \ldots, \vec{X}_{I-1}\}$ and $F(X)$ is non-zero, will never occur.

entries. By a union bound,

$$\mathsf{Pr}[x \in S \mid i = j] \leqslant p/2^{\mathsf{F.kl}+\mathsf{F.il}}$$

when $j \in E$.

The proof is then completed by noting

$$\begin{aligned}\sum_{j=1}^{p+q} \mathsf{Pr}[x \in \mathcal{S} \mid i = j] &= \sum_{j \in R} \mathsf{Pr}\big[x \in S' \mid i = j\big] + \sum_{j \in E} \mathsf{Pr}[x \in S \mid i = j] \\ &\leqslant p(q/2^{\mathsf{F.kl}+\mathsf{F.il}}) + q(p/2^{\mathsf{F.kl}+\mathsf{F.il}}) = 2pq/2^{\mathsf{F.kl}+\mathsf{F.il}}\end{aligned}$$

and plugging in to our earlier expression. □

4 Double Encryption

In this section we prove the security of the double encryption key-length extension technique against fully quantum attacks. Our proof first reduces this to the ability of a quantum algorithm to solve the list disjointness problem and then extends known query lower bounds for element distinctness to list disjointness (with some modifications).

The double encryption blockcipher is constructed via two sequential application of an underlying blockcipher. Formally, given a blockcipher $\mathsf{E} \in \mathsf{lcs}(\mathsf{E.kl}, \mathsf{E.bl})$, we define the double encryption blockcipher $\mathsf{DE}[\mathsf{E}]$ by $\mathsf{DE}[\mathsf{E}](K_1 \,\|\, K_2, x) = \mathsf{E}_{K_2}(\mathsf{E}_{K_1}(x))$. Here $|K_1| = |K_2| = \mathsf{E.kl}$ so $\mathsf{DE}[\mathsf{E}].\mathsf{kl} = 2\mathsf{E.kl}$ and $\mathsf{DE}[\mathsf{E}].\mathsf{bl} = \mathsf{E.bl}$. Its inverse can be computed as $\mathsf{DE}[\mathsf{E}]^{-1}(K_1 \,\|\, K_2, x) = \mathsf{E}^{-1}_{K_1}(\mathsf{E}^{-1}_{K_2}(x))$.

Classically, the meet-in-the-middle attack [10,26] shows that this construction achieves essentially the same security a single encryption. In the quantum setting, this construction was recently considered by Kaplan [16]. They gave an attack and matching security bound for the key-recovery security of double encryption. This leaves the question of whether their security result can be extended to cover full SPRP security, which we resolve by the main theorem of this section. This theorem is proven via a reduction technique of Tessaro and Thiruvengadam [31] which they used to establish a (classical) time-memory tradeoff for the security of double encryption, by reducing its security to the list disjointness problem and conjecturing a time-memory tradeoff for that problem.

Problems and languages. In addition to the list disjointness problem (1LD), we will also consider two versions of the element distinctness problem (ED, 1ED). In general, a problem PB specifies a relation $\mathcal{R}$ on set on instances $\mathcal{I}$ (i.e. $\mathcal{R}$ is function which maps an instance $L \in \mathcal{I}$ and witness w to a decision $\mathcal{R}(L, w) \in \{0, 1\}$). This relation induces a language $\mathcal{L} = \{L \in \mathcal{I} : \exists w, \mathcal{R}(L, w) = 1\}$. Rather than think of instances as bit strings, we will think of them as functions (to which decision and search algorithms are given oracle access). To restrict attention to functions of specific sizes we let $\mathcal{L}(D, R) = \mathcal{L} \cap \mathcal{I}(D, R)$ where $\mathcal{I}(D, R)$ denotes the restriction of $\mathcal{I}$ to functions $L : [D] \to [R]$, where $D \leqslant R$. To discuss instances not in the language we let $\mathcal{L}' = \mathcal{I} \backslash \mathcal{L}$ and $\mathcal{L}'(D, R) = \mathcal{I}(D, R) \backslash \mathcal{L}$.

Problems have decision and search versions. The goal of a decision algorithm is to output 1 (representing "acceptance") on instances in the language and 0 (representing "rejection") otherwise. Relevant quantities are the minimum probability P^1 of accepting an instance in the language, the maximum probability P^0 of accepting an instance not in the language, and the error rater E which are formally defined by

$$\begin{aligned}&P^1_{D,R}(\mathcal{A}) = \min_{L \in \mathcal{L}(D,R)} \mathsf{Pr}[\mathcal{A}[L] = 1],\ P^0_{D,R}(\mathcal{A}) = \max_{L \in \mathcal{L}'(D,R)} \mathsf{Pr}[\mathcal{A}[L] = 1] \\ &E_{D,R}(\mathcal{A}) = \max\{1 - P^1_{D,R}(\mathcal{A}), P^0_{D,R}(\mathcal{A})\}.\end{aligned}$$

We define the decision PB advantage of $\mathcal{A}$ by $\mathsf{Adv}^{\mathsf{PB}}_{D,R}(\mathcal{A}) = P^1_{D,R}(\mathcal{A}) - P^0_{D,R}(\mathcal{A})$. In non-cryptographic contexts, instead of looking at the difference in probability that inputs that are in or out of the language are accepted, one often looks at how far these are each from 1/2. This motivates the definition $\mathsf{Adv}^{\mathsf{PB\text{-}d}}_{D,R}(\mathcal{A}) = \min\{2P^1_{D,R}(\mathcal{A}) - 1, 1 - 2P^0_{D,R}(\mathcal{A})\} = 1 - 2E_{D,R}(\mathcal{A})$.

The goal of a search algorithm is to output a witness for the instance. We define its advantage to be the minimum probability it succeeds, i.e., $\mathsf{Adv}^{\mathsf{PB\text{-}s}}_{D,R}(\mathcal{A}) = \min_{L \in \mathcal{L}(D,R)} \mathsf{Pr}[\mathcal{R}(L, \mathcal{A}[L]) = 1]$.

Problem	Witness	Promise
ED	$x \neq y$ s.t. $L(x) = L(y)$	-
1ED	$x \neq y$ s.t. $L(x) = L(y)$	At most one witness.
1LD	x, y s.t. $L_0(x) = L_1(y)$	At most one witness. Injective L_0, L_1.

Fig. 8. Summary of the element distinctness and list disjointness problems we consider.

EXAMPLE PROBLEMS. The *list disjointness* problem asks how well an algorithm can distinguish between the case that is give (oracle access to) two lists which are disjoint or have one element in common. In particular, we interpret an instance L as the two functions $L_0, L_1 : [\lfloor D/2 \rfloor] \to [R]$ defined by $L_b(x) = L(x + b\lfloor D/2 \rfloor)$. Let $\mathcal{S}_n$ denote the set of L for which L_0 and L_1 are injective and which have n elements in common, i.e. for which $|\{L_0(1), \ldots, L_0(\lfloor D/2 \rfloor)\} \cap \{L_1(1), \ldots, L_1(\lfloor D/2 \rfloor)\}| = n$. Then 1LD is defined by the relation $\mathcal{R}$ which on input $(L, (x, y))$ returns 1 iff $L_0(x) = L_1(y)$ and the instance set $\mathcal{I} = \mathcal{S}_0 \cup \mathcal{S}_1$ (i.e., the promise that there is at most one element in common and that the lists are individually injective). The search version of list disjointness is sometimes referred to as claw-finding.

The *element distinctness* problem asks how well an algorithm can detect whether all the elements in a list are distinct. Let $\mathcal{S}'_n$ denote the set of L which have n collision pairs, i.e. for which $|\{\{x, y\} : x \neq y, L(x) = L(y)\}| = n$. Then ED is defined by the relation $\mathcal{R}$ which on input $(L, (x, y))$ returns 1 iff $x \neq y$ and $L(x) = L(y)$ with the instance set $\mathcal{I} = \bigcup_{n=0}^{\infty} S'_n$ consisting of all functions. We let 1ED denote restricting ED to $\mathcal{I} = \mathcal{S}'_0 \cup \mathcal{S}'_1$ (i.e., the promise that there is at most one repetition in the list).

4.1 Security result

The following theorem shows that an attacker achieving constant advantage must make $\Omega(2^{2k/3})$ oracle queries (ignoring log terms). Our bound is not tight for a lower parameter regimes, though future work may establish better bounds for list disjointness in these regimes.

Theorem 4. *Consider* $\mathsf{DE}[\cdot]$ *with the underlying blockcipher modeled by an ideal cipher drawn from* $\mathsf{Ics}(k, n)$. *Let* $\mathcal{A}$ *be a quantum adversary which makes at most* q *queries to the ideal cipher. Then*

$$\mathsf{Adv}^{\mathsf{sprp}}_{\mathsf{DE}}(\mathcal{A}) \leqslant 11\sqrt[6]{(q \cdot k \lg k)^3/2^{2k}} + 1/2^k.$$

As mentioned earlier, our proof works by first reducing the security of double encryption against quantum queries to the security of the list disjointness problem against quantum queries. This is captured in the following theorem which we prove now.

Theorem 5. *Consider* $\mathsf{DE}[\cdot]$ *with the underlying blockcipher modeled by an ideal cipher drawn from* $\mathsf{Ics}(k, n)$. *Let* $\mathcal{A}$ *be a quantum adversary which makes at most* q *queries to the ideal cipher. Then for any* $R \geqslant 2^k$ *we can construct* $\mathcal{A}'$ *making at most* q *oracle queries such that*

$$\mathsf{Adv}^{\mathsf{sprp}}_{\mathsf{DE}}(\mathcal{A}) \leqslant \mathsf{Adv}^{\mathit{1LD\text{-}d}}_{2^k,R}(\mathcal{A}') + 1/2^k.$$

We state and prove a bound on $\mathsf{Adv}^{\mathsf{1LD\text{-}d}}$ in Section 4.2. Our proof applies the same reduction technique as Tessaro and Thiruvengadam [31], we are verifying that it works quantumly as well.

Proof. For $b \in \{0, 1\}$, let H_b be defined to be identical to $\mathsf{G}^{\mathsf{sprp}}_{\mathsf{DE},b}$ except that K_2 is chosen uniformly from $\{0,1\}^k \backslash \{K_1\}$ rather than from $\{0,1\}^k$. This has no effect when $b = 0$ because the keys are not used, so $\mathsf{Pr}[\mathsf{G}^{\mathsf{sprp}}_{\mathsf{DE},0}(\mathcal{A})] = \mathsf{Pr}[\mathsf{H}_0]$. When $b = 1$ there was only a $1/2^k$ chance that K_2 would have equalled K_1 so $\mathsf{Pr}[\mathsf{G}^{\mathsf{sprp}}_{\mathsf{DE},1}(\mathcal{A})] \leqslant \mathsf{Pr}[\mathsf{H}_1] + 1/2^k$.

Now we define a decision algorithm $\mathcal{A}'$ for 1LD which uses its input lists to simulate a view for $\mathcal{A}$. When the lists are disjoint $\mathcal{A}$'s view will perfectly match that of H_0 and when the lists have exactly one element in

Adversary $\mathcal{A}'[L]$	$\textsc{Ic}(K, X, Y : \rho, \pi, F)$	$\textsc{Inv}(K, X, Y : \rho, \pi, F)$
$\rho \leftarrow\!\!\$\ \mathsf{Ics}(0, k)$	$(i, j) \leftarrow \rho(K)$	$(i, j) \leftarrow \rho(K)$
$\pi \leftarrow\!\!\$\ \mathsf{Ics}(0, n)$	If $i = 0$ then	If $i = 0$ then
$F \leftarrow\!\!\$\ \mathsf{Ics}([\lg R], n)$	$\quad Y \leftarrow Y \oplus F_{L_0(j)}(X)$	$\quad Y \leftarrow Y \oplus F^{-1}_{L_0(j)}(X)$
$b' \leftarrow\!\!\$\ \mathcal{A}[\pi, \pi^{-1}, \textsc{Ic}, \textsc{Inv}]$	Else	Else
Return b'	$\quad Y \leftarrow Y \oplus \pi(F^{-1}_{L_1(j)}(X))$	$\quad Y \leftarrow Y \oplus F_{L_1(j)}(\pi^{-1}(X))$
	Return $(K, X, Y : \rho, \pi, F)$	Return $(K, X, Y : \rho, \pi, F)$

Fig. 9. Reduction adversary used in proof of Theorem 5.

common $\mathcal{A}$'s view will perfectly match that of H_1. Hence we have $\mathsf{Pr}[\mathsf{H}_1] = \min_{(L,L') \in \mathcal{L}_1^{\kappa,k'}} \mathsf{Pr}\left[\mathsf{G}^{\mathsf{Id}}_{L,L'}(\mathcal{A})\right]$ and $\mathsf{Pr}[\mathsf{H}_0] = \max_{(L,L') \in \mathcal{L}_0^{\kappa,k'}} \mathsf{Pr}\left[\mathsf{G}^{\mathsf{Id}}_{L,L'}(\mathcal{A})\right]$, so $\mathsf{Pr}[\mathsf{H}_1] - \mathsf{Pr}[\mathsf{H}_0] = \mathsf{Adv}^{\mathsf{Id}}_{2^k,k'}(\mathcal{A}')$ which gives the claimed bound.

The adversary $\mathcal{A}'$ is defined in Fig. 9. It samples a permutation ρ on $\{0,1\}^k$, a permutation π on $\{0,1\}^n$, and a cipher F. The permutation ρ is used to provide a random map from the keys $K \in \{0,1\}^k$ to the elements of the lists L_0 and L_1. We will interpret $\rho(K)$ as a tuple (i, j) with $i \in \{0, 1\}$ and $j \in [2^k/2]$. Then K gets mapped to $L_i(j)$. Therefore either none of the keys map to the same element or a single pair of them maps to the same element.

Adversary $\mathcal{A}$'s queries to Ev and Inv are answered using π and π^{-1}. Its queries to the ideal cipher are more complicated and are handled by the oracles Ic and Inv. We can verify that these oracles define permutations on bitstring inputs, so $\mathcal{A}'$ is a well defined quantum adversary. Consider a key K and interpret $\rho(K)$ as a tuple (i, j) as described above. If $i = 0$, then ideal cipher queries for it are answered as if $\mathbf{E}_K(\cdot) = F_{L_0(j)}(\cdot)$. If $i = 1$, then ideal cipher queries for it are answered as if $\mathbf{E}_K(\cdot) = \pi(F^{-1}_{L_1(j)}(\cdot))$.[13] If the list element K is not mapped to by any other keys, then the indexing into F ensures that $\mathbf{E}_K(\cdot)$ is independent of π and $\mathbf{E}_{K'}$ for all other K'. If K and K' map to the same list element (and $i = 0$ for K), then $\mathbf{E}_K(\cdot)$ and $\mathbf{E}'_K(\cdot)$ are random permutations conditioned on $\mathbf{E}'_K(\mathbf{E}_K(\cdot)) = \pi(\cdot)$ and independent of all other $\mathbf{E}_{K''}(\cdot)$.

In H_0, the permutation of Ev and Inv is independent of each $\mathbf{E}_K(\cdot)$ which are themselves independent of each other. So this perfectly matches the view presented to $\mathcal{A}$ by $\mathcal{A}'$ when the lists are disjoint. In H_1, each $\mathbf{E}_K(\cdot)$ is pairwise independent and the permutation of Ev and Inv is defined to equal $\mathsf{E}_{K_2}(\mathsf{E}_{K_1}(\cdot))$. This perfectly matches the view presented to $\mathcal{A}$ by $\mathcal{A}'$ when the lists have one element in common because we can think of it as just having changed the order in which the permutations $\mathsf{E}_{K_2}(\mathsf{E}_{K_1}(\cdot))$, $\mathsf{E}_{K_1}(\cdot)$, and $\mathsf{E}_{K_2}(\cdot)$ were sampled. □

4.2 The Hardness of List Disjointness

If $\mathcal{A}$ is an algorithm making at most q classical oracle queries, then it is not hard to prove that $\mathsf{Adv}^{\mathsf{1LD}}_{D,R}(\mathcal{A}) \leqslant q/D$. If, instead, $\mathcal{A}$ makes as most q quantum oracle queries, the correct bound is less straightforward. In this section, we will prove the following result.

Theorem 6. *If $\mathcal{A}$ is a quantum algorithm making at most q queries to its oracle and $D \geqslant 32$ is a power of 2, then*

$$\mathsf{Adv}^{\mathsf{1LD}}_{D,3D^2}(\mathcal{A}) \leqslant 11 \sqrt[6]{(q \cdot \lg D \cdot \lg \lg D)^3 / D^2}.$$

We restrict attention to the case that D is a power of 2 only for notational simplicity in the proof. Essentially the same bound for more general D follows from the same techniques.

Ambanis's $\mathcal{O}(N^{2/3})$ query algorithm for element distinctness [2] can be used to solve list disjointness and hence shows this is tight (up to logarithmic factors) for attackers achieving constant advantage. The sixth root degrades the quality of the bound for lower parameter regimes. An interesting question we leave open is whether this could be proven without the sixth root or the logarithmic factors.

[13] When using output of L as keys for F we are identifying the elements of $[R]$ with elements of $\{0,1\}^{\lceil \lg R \rceil}$ in the standard manner.

PROOF SKETCH. The starting point for our reduction is that $\Omega(N^{2/3})$ lower bounds are known both the search and decision versions of ED [1,34].[14] By slightly modifying Zhandry's [34] technique for proving this, we instead get a bound on the hardness of 1ED-s. Next, a simple reduction (split the list in half at random) shows that 1LD-s is as hard as 1ED-s.

Then a "binary search" style reduction shows that 1LD-d is as hard as 1LD-s. In the reduction, the 1LD-s algorithm repeatedly splits its lists in half and uses the 1LD-d algorithm to determine which pair of lists contains the non-disjoint entries. However, we need our reduction to work by running the 1LD-d algorithm on a particular fixed size of list (the particular size we showed 1LD-d is hard for) rather than running it on numerous shrinking sizes. We achieve this by padding the lists with random elements. The choice of $R = 3D^2$ was made so that with good probability these random elements do not overlap with the actual list. This padding adds the $\lg D$ term to our bound.

Finally a generic technique allows us to relate the hardness of 1LD and 1LD-d. Given an algorithm with high 1LD advantage we can run it multiple times to get a precise estimate of how frequently it is outputting 1 and use that to determine what we want to output. This last step is the primary cause of the sixth root in our bound; it required running the 1LD algorithm on the order of $1/\delta^2$ times to get a precise enough estimate, where δ is the advantage of the 1LD algorithm. This squaring of δ in the query complexity of our 1LD-d algorithm (together with the fact that the query complexity is cubed in our 1ED-s bound) ultimately causes the sixth root.

CONSTITUENT LEMMAS. In the rest of the section we will state and prove lemmas corresponding to each of the step of the proof described above. In Section 4.3 we apply them one at a time to obtain the specific bound claimed by Theorem 6.

Lemma 1 (1ED-s is hard). *If $\mathcal{A}_{\textit{1ED-s}}$ is a quantum algorithm for 1ED-s making at most q queries to its oracle and $D \geqslant 32$, then $\mathsf{Adv}^{\textsf{ED-s}}_{D,3D^2}(\mathcal{A}_{\textit{1ED-s}}) \leqslant 9 \cdot (q+2)^3/D^2$.*

Proof. In [34], Zhandry shows that no Q-query algorithm can distinguish between a random function and a random injective function with domain $[D]$ and codomain $[R]$ with advantage better than $(\pi^2/3)Q^3/R$, as long as $D \leqslant R$. We could build a distinguisher $\mathcal{B}$ that on input $L : [D] \to [R]$ runs $(x, y) \leftarrow \mathcal{A}[L]$. If (x, y) is a collision (i.e., $x \neq y$ and $L(x) = L(x)$), then $\mathcal{B}$ outputs 1. It checks this by making two additional L queries, so $Q = q + 2$. Otherwise it outputs zero. Clearly, (x, y) cannot be a collision when L is an injection so it will always output zero. Hence, if 1coll denotes the event that L contains exactly one collision we obtain a bound of

$$\Pr[\textsf{1coll}] \cdot \mathsf{Adv}^{\textsf{ED-s}}_{D,3D^2}(\mathcal{A}_{\textsf{1ED-s}}) \leqslant (\pi^2/3)(q+2)^3/R. \tag{1}$$

It remains to lower bound Pr[1coll]. Note there are $\binom{D}{2}$ possible pairs of inputs that could be collisions. Each pair has a $1/R$ chance of colliding. By a union bound, the probability any other input has the same output as the collision is at most $(D-2)/R$, so there is at least a $1-(D-2)/R$ probability this does not occur. Given the above there are $(D-2)$ inputs sampled from $R-1$ possible values, so by a union bound the probability none of them collide is at least $1-\binom{D-2}{2}/(R-1)$. This gives

$$\Pr[\textsf{1coll}] \geqslant \frac{D(D-1)}{2R} \cdot \left(1 - \frac{D-2}{R}\right) \cdot \left(1 - \frac{(D-2)(D-3)}{2(R-1)}\right).$$

Now setting $R = 3D^2$ and applying simple bounds (e.g. $D-2 < D-1 < D$ and $(D-3)/(R-1) < (D-2)/R$) gives,

$$\Pr[\textsf{1coll}] \geqslant \frac{(1-1/D)}{6} \cdot \left(1 - \frac{1}{3D}\right) \cdot \left(1 - \frac{1}{6}\right)$$

Plugging this lower bound into equation 1, re-arranging, applying the bound $D \geqslant 32$, and rounding up to the nearest whole number gives the claimed bound $\mathsf{Adv}^{\textsf{ED-s}}_{D,3D^2}(\mathcal{A}_{\textsf{1ED-s}}) \leqslant 9(q+2)^3/D^3$. □

[14] In the proof we actually work with the advantage upper bounds, rather than the corresponding query lower bounds.

Algorithm $\mathcal{A}_{\mathsf{1LD\text{-}s}}[L_0, L_1]$
$L' \xleftarrow{\$} \mathsf{Inj}(D, R)$; $L'_0 \,\|\, L'_1 \leftarrow L$
$i \leftarrow 0$; $L_0^0 \leftarrow L_0$; $L_1^0 \leftarrow L_1$
Repeat
  $i \leftarrow i + 1$
  $L_{0,l} \,\|\, L_{0,r} \leftarrow L_0^{i-1}$
  $L_{1,l} \,\|\, L_{1,r} \leftarrow L_1^{i-1}$
  $(j^*, k^*) \leftarrow (r, r)$
  For $(j, k) \in \{(l, l), (l, r), (r, l)\}$ do
    // $f \,\square\, g(x) = f(x)$ **for** $x \in \mathsf{Dom}(f)$ **else** $g(x)$
    $b \xleftarrow{\$} \mathcal{A}_{\mathsf{1LD\text{-}d}}[L_{0,j} \,\square\, L'_0, L_{1,k} \,\square\, L'_1]$
    If $b = 1$ then $(j^*, k^*) \leftarrow (j, k)$
  $L_0^i \leftarrow L_{0,j^*}$; $L_1^i \leftarrow L_{1,k^*}$
Until $|\mathsf{Dom}(L_0^i)| = |\mathsf{Dom}(L_1^i)| = 1$
Pick $(x, y) \in \mathsf{Dom}(L_0^i) \times \mathsf{Dom}(L_1^i)$
Return (x, y)

Fig. 10. Reduction algorithm $\mathcal{A}_{\mathsf{1LD\text{-}s}}$ for Lemma 3. For notational convenience we write the two lists as separate input, rather than combined into a single list.

Lemma 2 (1ED-s hard ⇒ 1LD-s hard). *Let D be even. If $\mathcal{A}_{\mathit{1LD}\text{-}\mathsf{s}}$ is a quantum algorithm for 1LD-s making at most q queries to its oracle, then there is an algorithm $\mathcal{A}_{\mathit{1ED}\text{-}\mathsf{s}}$ (described in the proof) such that $\mathsf{Adv}^{\mathit{1LD}\text{-}\mathsf{s}}_{D,R}(\mathcal{A}_{\mathit{1LD}\text{-}\mathsf{s}}) \leqslant 2\mathsf{Adv}^{\mathit{1ED}\text{-}\mathsf{s}}_{D,R}(\mathcal{A}_{\mathit{1ED}\text{-}\mathsf{s}})$. Algorithm $\mathcal{A}_{\mathit{1ED}\text{-}\mathsf{s}}$ makes at most q queries to its oracle.*

Proof. On input a list $L : [D] \to [R]$, the algorithm $\mathcal{A}_{\mathsf{1ED\text{-}s}}$ will pick a random permutation $\pi : [D] \to [D]$. It runs $\mathcal{A}_{\mathsf{1LD\text{-}s}}[L \circ \pi]$ and then, on receiving output $(x, y) \in [D/2]^2$, returns $(\pi^{-1}(x), \pi^{-1}(y + D/2))$. The permutation π serves the role of splitting L into two sublists for $\mathcal{A}_{\mathsf{1LD\text{-}s}}$ at random. As long as the original collision of L doesn't end up being put into the same sublist (which has probability less than $1/2$), $\mathcal{A}_{\mathsf{1LD\text{-}s}}$ will be run on a valid 1LD-s instance and $\mathcal{A}_{\mathsf{1ED\text{-}s}}$ will succeed whenever $\mathcal{A}_{\mathsf{1LD\text{-}s}}$ does. The claim follows. □

Lemma 3 (1LD-s hard ⇒ 1LD-d hard). *Let D be a power of two. If $\mathcal{A}_{\mathit{1LD}}$ is a quantum algorithm for 1LD making at most q queries to its oracle, then there is an 1LD-s algorithm $\mathcal{A}_{\mathit{1LD}\text{-}\mathsf{s}}$ (described in the proof) such that*

$$\mathsf{Adv}^{\mathit{1LD}\text{-}\mathsf{s}}_{D,R}(\mathcal{A}_{\mathit{1LD}\text{-}\mathsf{s}}) \geqslant 1 - D^2/R - 1.5(\lg D - 2)(1 - \mathsf{Adv}^{\mathit{1LD}\text{-}\mathsf{d}}_{D,R}(\mathcal{A}_{\mathit{1LD}\text{-}\mathsf{d}})).$$

Algorithm $\mathcal{A}_{\mathit{1LD}\text{-}\mathsf{s}}$ makes at most $3q \lg D$ queries to its oracle.

This theorem's bound is vacuous if $\mathsf{Adv}^{\mathsf{1LD\text{-}d}}_{D,R}(\mathcal{A}_{\mathsf{1LD\text{-}d}})$ is too small. When applying the result we will have obtained $\mathcal{A}_{\mathsf{1LD\text{-}d}}$ by amplifying the advantage of another adversary to ensure it is sufficiently large.

Proof. The algorithm $\mathcal{A}_{\mathsf{1LD\text{-}s}}$ is given in Fig. 10. The intuition behind this algorithm is as follows. It wants to use the decision algorithm $\mathcal{A}_{\mathsf{1LD\text{-}d}}$ to perform a binary search to find the overlap between L_0 and L_0. It runs for $\lg D/2 - 1$ rounds. In the i-th round, it splits the current left list L_0^{i-1} into two sublists $L_{0,l}$, $L_{0,r}$ and the current right lists L_1^{i-1} into two sublists $L_{1,l}$, $L_{1,r}$.[15] If L_0^{i-1} and L_1^{i-1} have an element in common, then one of the pairs of sublists $L_{0,j}$ and $L_{1,k}$ for $j, k \in \{l, r\}$ must have an element in common. The decision algorithm $\mathcal{A}_{\mathsf{1LD\text{-}d}}$ is run on different pairs to determine which contains the overlap. We recurse with chosen pair, until we are left with lists that have singleton domains at which point we presume the entries therein give the overlap.

[15] In code, $f \,\|\, g \leftarrow h$ for h with domain $\mathsf{Dom}(h) = \{n, n+1, \ldots, m\}$ defines f to be the restriction of h to domain $\mathsf{Dom}(f) = \{n, n+1, \ldots, \lfloor (n+m)/2 \rfloor\}$ and g to be the restriction of h to domain $\mathsf{Dom}(g) = \mathsf{Dom}(h) \backslash \mathsf{Dom}(f)$.

Because we need to run the decision algorithm $\mathcal{A}_{\mathsf{1LD\text{-}d}}$ on fixed size inputs we pad the sublists to be of a fixed size using lists L'_0 and L'_1 (the two halves of an injection L' that we sampled locally). As long as the image of L' does not overlap with the images of L_0 and L_1, this padding does not introduce any additional elements repetitions between or within the list input to $\mathcal{A}_{\mathsf{1LD\text{-}d}}$. By a union bound, overlaps occurs with probability at most D^2/R.

Conditioned on such overlaps not occurring, $\mathcal{A}_{\mathsf{1LD\text{-}s}}$ will output the correct result if $\mathcal{A}'_{\mathsf{1LD\text{-}d}}$ always answers correctly. To bound the probability of $\mathcal{A}'_{\mathsf{1LD\text{-}d}}$ erring we can note that it is run $3 \cdot (\lg D - 2)$ times and, each time, has at most a $(1 - \mathsf{Adv}^{\mathsf{1LD\text{-}d}}_{D,R}(\mathcal{A}'_{\mathsf{1LD\text{-}d}}))/2$ chance of error and apply a union bound.

Put together we have

$$\mathsf{Adv}^{\mathsf{1LD\text{-}s}}_{D,R}(\mathcal{A}_{\mathsf{1LD\text{-}s}}) \geqslant 1 - D^2/R - 1.5(\lg D - 2)(1 - \mathsf{Adv}^{\mathsf{1LD\text{-}d}}_{D,R}(\mathcal{A}_{\mathsf{1LD\text{-}d}}).$$

That $\mathcal{A}_{\mathsf{1LD\text{-}s}}$ makes at most $3q(\lg D - 2)$ oracle queries is clear. □

Lemma 4 (PB-d hard ⇒ PB hard). *Let PB be any problem. Suppose $\mathcal{A}_{PB}$ is a quantum algorithm for PB making at most q queries to its oracle, with $\mathsf{Adv}^{PB}_{D,R}(\mathcal{A}_{PB}) = \delta > 0$. Then for any $t \in \mathbb{N}$ there is an algorithm $\mathcal{A}_{PB\text{-}\mathsf{d}}$ (described in the proof) such that $\mathsf{Adv}^{PB\text{-}\mathsf{d}}_{D,R}(\mathcal{A}_{PB\text{-}\mathsf{d}}) > 1 - 2/2^t$. Algorithm $\mathcal{A}_{PB\text{-}\mathsf{d}}$ makes $q \cdot \lceil 4.5(t+1)\ln 2/\delta^2 \rceil$ queries to its oracle.*

Proof. For compactness, let $p^1 = P^1_{D,R}(\mathcal{A}_{\mathsf{PB}})$ denote the minimum probability that $\mathcal{A}_{\mathsf{PB}}$ outputs 1 on instances in the language, $p^0 = P^0_{D,R}(\mathcal{A}_{\mathsf{PB}})$ denote the maximum probability that $\mathcal{A}_{\mathsf{PB}}$ outputs 1 on instances not in the language, and $\delta = \mathsf{Adv}^{\mathsf{PB}}_{D,R}(\mathcal{A}_{\mathsf{PB}}) = p^1 - p^0$. We define $\mathcal{A}_{\mathsf{PB\text{-}d}}$ to be the algorithm that, on input L, runs n independent copies of $\mathcal{A}_{\mathsf{PB}}[L]$ (with $n = \lceil 4.5(t+1) * \ln 2/\delta^2 \rceil$) and calculates the average p of all the values output by $\mathcal{A}_{\mathsf{PB}}[L]$. (Think of this as an estimate of the probability $\mathcal{A}_{\mathsf{PB}}$ outputs 1 on this input.) If $p < p^0 + \delta/2$, $\mathcal{A}_{\mathsf{PB\text{-}d}}$ outputs 0, otherwise it outputs 1.

Let X_i denote the output of the i-th execution of $\mathcal{A}_{\mathsf{PB\text{-}d}}$. An inequality of Hoeffding [**?**] bounds how far the average of independent random variables $0 \leqslant X_i \leqslant 1$ can differ from the expectation by

$$\mathsf{Pr}\left[\left|\sum_{i=1}^{n} X_i/n - \mathsf{E}\left[\sum_{i=1}^{n} X_i/n\right]\right| > \varepsilon\right] < 2e^{-2\varepsilon^2 n}$$

for any $\varepsilon > 0$. Let p' denote the expected value of p (which is also the expected value of X_i). If L is in the language, then $p^1 \leqslant p$. Otherwise $p^0 \geqslant p$. In either case, $\mathcal{A}_{\mathsf{PB\text{-}d}}$ will output the correct answer if p does not differ from p' by more than $\delta/3$ (because this is strictly less than $\delta/2$). Applying the above inequality tells us that $\mathsf{Pr}[|p - p'| > \delta/3] < 2e^{-2\delta^2 n/9} \leqslant 2^{-t}$. (The value of n was chosen to make this last inequality hold.)

Hence $P^1_{D,R}(\mathcal{A}_{\mathsf{PB\text{-}d}}) > 1 - 2^{-t}$, $P^0_{D,R}(\mathcal{A}_{\mathsf{PB\text{-}d}}) < 2^{-t}$, and $\mathsf{Adv}^{\mathsf{PB\text{-}d}}_{D,R}(\mathcal{A}_{\mathsf{PB\text{-}d}}) > 1 - 2/2^t$. The bound on $\mathcal{A}_{\mathsf{PB\text{-}d}}$'s number of queries is clear. □

4.3 Proof of Theorem 6

Let $\mathcal{A}_{\mathsf{1LD}}$ be our given list disjointness adversary which makes q oracle queries and has advantage $\delta = \mathsf{Adv}^{\mathsf{1LD}}_{D,R}(\mathcal{A}_{\mathsf{1LD}}) > 0$. If we apply Lemma 4 with $t = \lg(12 \lg D)$, then we get an adversary $\mathcal{A}_{\mathsf{1LD\text{-}d}}$ which makes $q_{\mathsf{1LD\text{-}d}} = q \cdot \lceil 4.5(t+1)\ln 2/\delta^2 \rceil < (10q \lg\lg D)/\delta^2$ oracle queries. (We used here the assumption $D \geqslant 32$ to simplify constants.) This adversary has advantage

$$\mathsf{Adv}^{\mathsf{1LD\text{-}d}}_{D,3D^2}(\mathcal{A}_{\mathsf{1LD\text{-}d}}) > 1 - 2/2^t = 1 - 1/(6\lg(D)).$$

Next applying Lemma 3, gives us $\mathcal{A}_{\mathsf{1LD\text{-}s}}$ which makes fewer than $q_{\mathsf{1LD\text{-}s}} = 30q(\lg D)(\lg\lg D)/\delta^2$ oracle queries and has advantage

$$\begin{aligned}\mathsf{Adv}^{\mathsf{1LD\text{-}s}}_{D,R}(\mathcal{A}_{\mathsf{1LD\text{-}s}}) &\geqslant 1 - D^2/R - 1.5(\lg D - 2)(1 - \mathsf{Adv}^{\mathsf{1LD\text{-}d}}_{D,R}(\mathcal{A}_{\mathsf{1LD\text{-}d}})) \\ &> 1 - 1/3 - 1.5(\lg D - 2)(6\lg D)^{-1} > 1 - 1/3 - 1/4 = 5/12.\end{aligned}$$

Lemmas 1 and 2 together bound this advantage from the other direction. In particular, we get that

$$5/12 < \mathsf{Adv}^{\mathsf{1LD\text{-}s}}_{D,R}(\mathcal{A}_{\mathsf{1LD\text{-}s}}) \leqslant 18 \cdot (q_{\mathsf{1LD\text{-}s}} + 2)^3/D^2.$$

Using the assumption $D \geqslant 32$ we can bound $q_{\mathsf{1LD\text{-}s}} + 2$ by $31q(\lg D)(\lg\lg D)/\delta^2$. Plugging this in and solving for δ gives our claimed bound of

$$\delta < 11\sqrt[6]{(q \cdot \lg D \cdot \lg\lg D)^3/D^2}.$$

Acknowledgements

Joseph Jaeger and Stefano Tessaro were partially supported by NSF grants CNS-1930117 (CAREER), CNS-1926324, CNS-2026774, a Sloan Research Fellowship, and a JP Morgan Faculty Award. Joseph Jaeger's work done while at the University of Washington. Fang Song thanks Robin Kothari for helpful discussion on the element distinctness problem. Fang Song was supported by NSF grants CCF-2041841, CCF-2042414, and CCF-2054758 (CAREER).

References

1. S. Aaronson and Y. Shi. Quantum lower bounds for the collision and the element distinctness problems. *Journal of the ACM (JACM)*, 51(4):595–605, 2004.
2. A. Ambainis. Quantum walk algorithm for element distinctness. *SIAM Journal on Computing*, 37(1):210–239, 2007.
3. A. Ambainis, M. Hamburg, and D. Unruh. Quantum security proofs using semi-classical oracles. In A. Boldyreva and D. Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 269–295. Springer, Heidelberg, Aug. 2019.
4. N. Bindel, M. Hamburg, K. Hövelmanns, A. Hülsing, and E. Persichetti. Tighter proofs of cca security in the quantum random oracle model. In *Theory of Cryptography Conference*, pages 61–90. Springer, 2019.
5. X. Bonnetain, A. Hosoyamada, M. Naya-Plasencia, Y. Sasaki, and A. Schrottenloher. Quantum attacks without superposition queries: the offline Simon's algorithm. In *Advances in Cryptology – ASIACRYPT*, pages 552–583. Springer, 2019.
6. C. Chevalier, E. Ebrahimi, and Q.-H. Vu. On security notions for encryption in a quantum world. Cryptology ePrint Archive, Report 2020/237, 2020. https://ia.cr/2020/237.
7. J. Czajkowski. Quantum indifferentiability of sha-3. Cryptology ePrint Archive, 2021. http://eprint.iacr.org/2021/192.
8. J. Czajkowski, C. Majenz, C. Schaffner, and S. Zur. Quantum lazy sampling and game-playing proofs for quantum indifferentiability. Cryptology ePrint Archive, Report 2019/428, 2019. https://eprint.iacr.org/2019/428.
9. Ö. Dagdelen, M. Fischlin, and T. Gagliardoni. The Fiat-Shamir transformation in a quantum world. In K. Sako and P. Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 62–81. Springer, Heidelberg, Dec. 2013.
10. W. Diffie and M. E. Hellman. Exhaustive cryptanalysis of the nbs data encryption standard. *IEEE Computer*, 10(6), 1977.
11. S. Even and Y. Mansour. A construction of a cipher from a single pseudorandom permutation. *Journal of Cryptology*, 10(3):151–162, June 1997.
12. L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pages 212–219, New York, NY, USA, 1996. Association for Computing Machinery.
13. A. Hosoyamada and Y. Sasaki. Cryptanalysis against symmetric-key schemes with online classical queries and offline quantum computations. In *Cryptographers' Track at the RSA Conference*, pages 198–218. Springer, 2018.
14. K. Hövelmanns, E. Kiltz, S. Schäge, and D. Unruh. Generic authenticated key exchange in the quantum random oracle model. In *IACR International Conference on Public-Key Cryptography*, pages 389–422. Springer, 2020.
15. J. Jaeger, F. Song, and S. Tessaro. Quantum key-length extension. Cryptology ePrint Archive, 2021. http://eprint.iacr.org/2021/579.
16. M. Kaplan. Quantum attacks against iterated block ciphers. *arXiv preprint arXiv:1410.1434*, 2014.

17. M. Kaplan, G. Leurent, A. Leverrier, and M. Naya-Plasencia. Quantum differential and linear cryptanalysis. *IACR Trans. Symm. Cryptol.*, 2016(1):71–94, 2016. https://tosc.iacr.org/index.php/ToSC/article/view/536.
18. S. Katsumata, S. Yamada, and T. Yamakawa. Tighter security proofs for GPV-IBE in the quantum random oracle model. *Journal of Cryptology*, 34(1):1–46, 2021.
19. J. Kilian and P. Rogaway. How to protect DES against exhaustive key search. In N. Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 252–267. Springer, Heidelberg, Aug. 1996.
20. J. Kilian and P. Rogaway. How to protect DES against exhaustive key search (an analysis of DESX). *Journal of Cryptology*, 14(1):17–35, Jan. 2001.
21. E. Kiltz, V. Lyubashevsky, and C. Schaffner. A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle model. In J. B. Nielsen and V. Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 552–586. Springer, Heidelberg, Apr. / May 2018.
22. V. Kuchta, A. Sakzad, D. Stehlé, R. Steinfeld, and S.-F. Sun. Measure-rewind-measure: tighter quantum random oracle model proofs for one-way to hiding and cca security. In *Advances in Cryptology – EUROCRYTPT 2020*, pages 703–728. Springer, 2020.
23. H. Kuwakado and M. Morii. Security on the quantum-type even-mansour cipher. In *2012 International Symposium on Information Theory and its Applications*, pages 312–316, 2012.
24. G. Leander and A. May. Grover meets simon - quantumly attacking the FX-construction. In T. Takagi and T. Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 161–178. Springer, Heidelberg, Dec. 2017.
25. B. Mennink and A. Szepieniec. XOR of PRPs in a quantum world. In T. Lange and T. Takagi, editors, *Post-Quantum Cryptography - 8th International Workshop, PQCrypto 2017*, pages 367–383. Springer, Heidelberg, 2017.
26. R. C. Merkle and M. E. Hellman. On the security of multiple encryption. *Communications of the ACM*, 24(7):465–467, 1981.
27. A. Rosmanis. Tight bounds for inverting permutations via compressed oracle arguments. *arXiv preprint arXiv:2103.08975*, 2021.
28. P. W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th FOCS*, pages 124–134. IEEE Computer Society Press, Nov. 1994.
29. D. R. Simon. On the power of quantum computation. *SIAM journal on computing*, 26(5):1474–1483, 1997.
30. E. E. Targhi and D. Unruh. Post-quantum security of the Fujisaki-Okamoto and OAEP transforms. In M. Hirt and A. D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 192–216. Springer, Heidelberg, Oct. / Nov. 2016.
31. S. Tessaro and A. Thiruvengadam. Provable time-memory trade-offs: Symmetric cryptography against memory-bounded adversaries. In A. Beimel and S. Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 3–32. Springer, Heidelberg, Nov. 2018.
32. D. Unruh. Revocable quantum timed-release encryption. *J. ACM*, 62(6), Dec. 2015.
33. M. Zhandry. How to construct quantum random functions. In *53rd FOCS*, pages 679–687. IEEE Computer Society Press, Oct. 2012.
34. M. Zhandry. A note on the quantum collision and set equality problems. *Quantum Information and Computation*, 15(7& 8), 2015.
35. M. Zhandry. How to record quantum queries, and applications to quantum indifferentiability. In A. Boldyreva and D. Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 239–268. Springer, Heidelberg, Aug. 2019.

A Probabilistic Analysis for Proof of Theorem 2

In this appendix we give a detailed probability analysis of Claim (i) from the proof of Theorem 2.

The view of $\mathcal{A}$ when run by $\mathcal{A}'$ in $\mathsf{G}^{\mathsf{dist}}_{\mathbf{D},1}$ is determined by f_1, f_1^{-1}, and $z = (T, T^{-1})$. These are chosen such that $T[M_i] = f_1(K_1, M_i \oplus K_2) \oplus K_2$ for $i = 1, \ldots, q'$ and $T^{-1}[Y_i] = f_1^{-1}(K_1, Y_i \oplus K_2) \oplus K_2$ for $i = q'+1, \ldots, q$. Consequently to ensure this matches the view from $\mathsf{G}^{\mathsf{sprp}}_{\mathsf{F},1}$ we need to show that $\mathbf{D}$ choses (f_1, K_1, K_2) uniformly from $\mathsf{Ics}(k,n) \times \{0,1\}^k \times \{0,1\}^n$. It is clear that K_1 and K_2 are uniform. Furthermore, $f_1(K, \cdot) = f_0(K, \cdot)$ for $K \neq K_1$ so these are sampled correctly. Hence we can think of these values as fixed and argue that the distribution induced over $f_1(K_1, \cdot)$ is uniform over all permutations on $\{0,1\}^n$.

Let $\mathbf{f} \in \mathsf{lcs}(k,n)$, $N = 2^n$, and E denote the event that $f_1(K_1,\cdot) = \mathbf{f}(\cdot)$. Let E_1 denote the event that $T[M_i] = \mathbf{f}(M_i \oplus K_2) \oplus K_2$ for $i = 1,\ldots,q$ after Step 1. The second for loop of Step 3 programs f_1 to satisfy $f_1(K_1, M_i \oplus K_2) = T[M_i] \oplus K_2$ so E_1 is a necessary condition for E. Note that E_1 requires Q values to have been sampled correctly in Step 1's for loops where $Q = |\{\mathbf{f}(M_i \oplus K_2) \oplus K_2 : 1 \leqslant i \leqslant q'\} \cup \{\mathbf{f}(Y_i \oplus K_2) \oplus K_2 : q' \leqslant i \leqslant q\}|$. Hence,

$$\mathsf{Pr}[E] = \mathsf{Pr}[E|E_1] \cdot \frac{(N-Q)!}{N!}.$$

Let E_2 denote the event that Step 2 samples an f_0 which is consistent with $\mathbf{f}$. That is to say, that f_0 is chosen such that $\mathbf{f}(\mathcal{I} \cup \mathcal{I}') = \mathcal{O} \cup \mathcal{O}'$ and $\mathbf{f}(x) = f_0(K_1,x)$ for $x \notin \mathcal{I} \cup \mathcal{I}'$. The first for loop in Step 3 programs $f_1(K_1,x) = f_0(K_1,x)$ for $x \notin \mathcal{I} \cup \mathcal{I}'$. The second and third for loop in Step 3 programs $f_1(K_1, \mathcal{I} \cup \mathcal{I}')$ to have values in $\mathcal{O} \cup \mathcal{O}'$ so E_2 is a necessary condition for E. Let $M(f_0) = |\mathcal{I} \backslash \mathcal{I}'| = |\mathcal{O} \backslash \mathcal{O}'|$ and let E_2^m denote the event that $M(f_0) = m$.

$$\mathsf{Pr}[E|E_1] = \sum_m \mathsf{Pr}[E_2^m|E_1] \cdot \mathsf{Pr}[E_2|E_1,E_2^m] \cdot \mathsf{Pr}[E|E_1,E_2^m,E_2].$$

We can think of Step 2 as lazily sampling $f_0(K_1,\cdot)$ by first sampling $f_0(K_1,x)$ for $x \in \mathcal{I}$, then sampling $f_0^{-1}(K_1,y)$ for $y \in \mathcal{O} \backslash \mathcal{O}'$, and then sampling $f_0(K_1,x)$ for $x \notin \mathcal{I} \cup \mathcal{I}'$. The event E_2 requires that the m values sampled in this second step are the elements of $\mathbf{f}^{-1}(\mathcal{O}' \backslash \mathcal{O})$ and that on the third step $\mathbf{f}(x)$ is sampled for $f_0(K_1,x)$ for each $x \notin \mathcal{I} \cup \mathcal{I}'$. Hence,

$$\mathsf{Pr}[E_2|E_1,E_2^m] = \frac{1}{\binom{N-Q}{m}} \cdot \frac{1}{(N-Q-m)!}.$$

For E to occur (conditioned on E_1, E_2^m, and E_2) the third for loop in Step 3 must sample m values correctly, so $\mathsf{Pr}[E|E_1,E_2^m,E_2] = 1/m!$. Putting everything together we have

$$\mathsf{Pr}[E] = \frac{(N-Q)!}{N!} \sum_m \mathsf{Pr}[E_2^m|E_1] \cdot \frac{m! \cdot (N-Q-m)!}{(N-Q)!} \cdot \frac{1}{(N-Q-m)!} \cdot \frac{1}{m!} = \frac{1}{N!}.$$

So f_1 is uniformly distributed, as desired.