Simpler Proofs of Quantumness

Zvika Brakerski
Weizmann Institute of Science, Rehovot, Israel
zvika.brakerski@weizmann.ac.il

Venkata Koppula
Weizmann Institute of Science, Rehovot, Israel
venkata.koppula@weizmann.ac.il

Umesh Vazirani
University of California, Berkeley, CA, USA
vazirani@cs.berkeley.edu

Thomas Vidick
California Institute of Technology, Pasadena, CA, USA
vidick@cms.caltech.edu

Abstract

A proof of quantumness is a method for provably demonstrating (to a classical verifier) that a quantum device can perform computational tasks that a classical device with comparable resources cannot. Providing a proof of quantumness is the first step towards constructing a useful quantum computer.

There are currently three approaches for exhibiting proofs of quantumness: (*i*) Inverting a classically-hard one-way function (e.g. using Shor's algorithm). This seems technologically out of reach. (*ii*) Sampling from a classically-hard-to-sample distribution (e.g. BosonSampling). This may be within reach of near-term experiments, but for all such tasks known verification requires exponential time. (*iii*) Interactive protocols based on cryptographic assumptions. The use of a trapdoor scheme allows for efficient verification, and implementation seems to require much less resources than (*i*), yet still more than (*ii*).

In this work we propose a significant simplification to approach (*iii*) by employing the random oracle heuristic. (We note that we *do not* apply the Fiat-Shamir paradigm.)

We give a two-message (challenge-response) proof of quantumness based on any trapdoor claw-free function. In contrast to earlier proposals we do not need an adaptive hard-core bit property. This allows the use of smaller security parameters and more diverse computational assumptions (such as Ring Learning with Errors), significantly reducing the quantum computational effort required for a successful demonstration.

2012 ACM Subject Classification Theory of computation → Cryptographic protocols; Theory of computation → Quantum complexity theory

Keywords and phrases Proof of Quantumness, Random Oracle, Learning with Errors

Digital Object Identifier 10.4230/LIPIcs.TQC.2020.8

Funding *Zvika Brakerski*: Supported by the Binational Science Foundation (Grant No. 2016726), and by the European Union Horizon 2020 Research and Innovation Program via ERC Project REACT (Grant 756482) and via Project PROMETHEUS (Grant 780701).
Venkata Koppula: Supported by the Binational Science Foundation (Grant No. 2016726), and by the European Union Horizon 2020 Research and Innovation Program via ERC Project REACT (Grant 756482) and via Project PROMETHEUS (Grant 780701).
Umesh Vazirani: Supported in part by ARO Grant W911NF-12-1-0541, NSF Grant CCF1410022, a Vannevar Bush faculty fellowship, and the Miller Institute at U.C. Berkeley through a Miller Professorship.
Thomas Vidick: Supported by NSF CAREER Grant CCF-1553477, AFOSR YIP award number FA9550-16-1-0495, a CIFAR Azrieli Global Scholar award, MURI Grant FA9550-18-1-0161, and the IQIM, an NSF Physics Frontiers Center (NSF Grant PHY-1125565).

15th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2020).
Editor: Steven T. Flammia; Article No. 8; pp. 8:1–8:14

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Quantum computing holds a promise of a qualitative leap in our ability to perform important computational tasks. These tasks include simulation of chemical and physical systems at the quantum level, generating true randomness, algorithmic tasks such as factoring large numbers, and more. However, constructing a quantum computer with capabilities beyond those of existing classical computers is technologically challenging. Indeed, whether it is possible or not remains to be proven; such a "proof" is the focus of the ongoing race to construct a useful quantum device, with records for device size and functionality set at an increasing rate by the likes of Google, IBM, and the increasing number of startups heavily invested in this race. This notion, known as "proof of quantumness",[1] is generally viewed as a major milestone towards unlocking the powers of quantum computing. We can classify existing approaches towards proof of quantumness into three families:

1. There are tasks that are generally believed to be classically intractable, and for which quantum algorithms are known; most notably the factoring and discrete logarithm problems [16]. Constructing a quantum computer that can factor beyond our classical capabilities would constitute a valid proof of quantumness. Alas, in order to implement the factoring algorithm on relevant input sizes one requires fault-tolerant quantum computation, which seems technologically out of reach (see e.g. [8] for recent and highly optimized estimates ranging in the millions of qubits).
2. A different approach, introduced independently by Bremner, Josza and Shepperd [4] and by Aaronson and Arkhipov [1], is to use a quantum device to sample from distributions that are presumed to be hard to sample from classically. The intractability of classically achieving the task has not stood the same test of time as more established problems such as e.g. factoring, but can nonetheless be based on reasonable complexity-theoretic conjectures, at least for the problem of exact sampling. While quantum devices that can sample from these distributions appear to be "right around the corner", the real challenges are in (i) showing hardness of approximate sampling – the quantum device will never be perfect – and (ii) the classical verification: verification for these methods generally requires investing exponential classical computational resources, and can thus only be performed for fairly small input lengths.
3. A new approach was recently proposed in [3]. They propose to use *post-quantum cryptography*, namely to rely on cryptographic assumptions that cannot be broken even by the quantum device. Rather than verifying that the quantum device has the ability to break the assumption, cryptography is used to compel the device to generate a quantum superposition in a way that can be efficiently verified using a secret key. This method is inherently interactive, unlike the previous two, and requires at least four rounds of communication. As a cryptographic building block it uses trapdoor claw-free function families (recall that claw-freeness was originally introduced in the context of digital signatures and constructed based on factoring [9]). In addition to claw freeness, the [3] approach also requires an additional adaptive hardcore bit property which appears to be hard to realize and is currently only known to be achievable based on the Learning with Errors (LWE) assumption [15].

The third approach is compelling in its ability to verify quantumness even of large quantum devices efficiently, but it still requires a large number of quantum operations.

[1] The term "quantum supremacy" is also used in the literature.

Furthermore, the interactive nature of the protocol requires the quantum device to retain a superposition while waiting for the verifier's second message (a single random bit).

In this work we simplify the [3] approach and allow for it to be based on a more diverse set of computational assumptions. This marks a step towards a protocol that can be realistically implemented on an actual quantum device, and can be efficiently verified on a classical computer.

Our Results

We propose to use the *random oracle heuristic* as a tool to reduce the round complexity of the proof of quantumness protocol from [3], making it into a simple one-round message-response protocol. We note that it is unlikely that a similar result can be achieved *in the standard model* without introducing an additional hardness assumption. The reason is that a single-round message-response protocol in the standard model (i.e. without oracles) immediately implies that quantum samplers cannot be efficiently de-quantized (otherwise the protocol will have no soundness). Such a result therefore implies a (weak) separation between the BQP and BPP models. However, the LWE assumption does not appear to imply such a separation, and the current state of the art suggests that it is equally intractable in the quantum and classical settings.[2]

We show that using the random oracle heuristic, it is possible to implement the protocol in a single round while at the same time eliminating the need for an adaptive hard-core bit property, and thus relying on any family of claw-free functions. In particular, we propose a construction of trapdoor claw free functions which is analogous to that of [3] but relies on the Ring-LWE assumption [10, 11]. Ring-LWE based primitives are often regarded as more efficient than their LWE-based counterparts since they involve arithmetic over polynomial rings, which can be done more efficiently than over arbitrary linear spaces. Despite the similarity between LWE and Ring-LWE, proving an adaptive hard-core theorem for the latter appears to be a challenging task. This is since the LWE-based construction uses a so-called lossiness argument that is not known to be replicable in the Ring-LWE setting. We note that we can also instantiate our method using "pre-quantum" cryptography since soundness should hold only with respect to classical adversaries. Using a back-of-the-envelope calculation we estimate that it is possible to execute our protocol using superpositions over $\sim 8\lambda \log^2 \lambda$ qubits, for security parameter λ and the adversary would have advantage negligible in λ.

While we allow the use of trapdoor claw-free families based on arbitrary assumptions, which should allow for better security/efficiency trade-offs, our protocol still requires the quantum device to evaluate the random oracle on a quantum superposition, which could potentially create an additional burden. We point out that current and future heuristic instantiations of the random oracle model using explicit hash functions are assumed to enjoy efficient quantum implementation. Specifically, in evaluating the *post-quantum* security level of cryptographic constructions (e.g. for the NIST competition [14]), security is evaluated in the Quantum Random Oracle model where adversaries are assumed to evaluate hash functions on superpositions as efficiently as they do classically. Granted, this is just a model for an adversary, but it is customary to try to be as realistic as possible and not over-estimate the power of the adversary. We therefore consider the evaluation of the random hash function as a relatively lower-order addition to the cost of performing the quantumness test.

[2] This insight is due to a discussion with Omer Paneth.

Lastly, we compare our method to the most straightforward way to employ a random oracle for the purpose of round reduction, the Fiat-Shamir transform [7]. The basic protocol of [3] contains 4 messages, where the third message is simply a random bit. One can therefore do parallel repetition of the protocol (though the soundness of this transformation needs to be shown),[3] and apply Fiat-Shamir to compress it into challenge-response form. Furthermore, for proofs of quantumness soundness is only required to hold against a classical adversary, so the standard security reduction for Fiat-Shamir should hold. This approach only requires to apply the random oracle to a classical input. However, it still requires the adaptive hard-core bit property and is therefore restricted to the LWE assumption. We believe that our protocol, being of a somewhat modified form compared to prior works, may be useful for future applications.

Our Technique

At a high level, a family of trapdoor claw free functions allows to sample a function $f : \{0,1\} \times \{0,1\}^n \to \{0,1\}^n$ together with a trapdoor. The function has two branches $f(0,\cdot), f(1,\cdot)$ which are both injective, i.e. permutations (this is a simplified description, actual protocols use a relaxed "noisy" notion). It is guaranteed that it is computationally intractable to find a collision ("claw") x_0, x_1 s.t. $f(0,x_0) = f(1,x_1)$, however given the trapdoor it is possible to find for all y the preimages x_0, x_1 s.t. $f(0,x_0) = f(1,x_1) = y$.

The [3] protocol sends a description of f to the quantum device, asks it to apply f on a uniform superposition of inputs and measure the image register, call the value obtained y. The quantum device is then left with a uniform superposition over the two preimages of y: $(0,x_0)$ and $(1,x_1)$. The value y is sent to the verifier who challenges the quantum device to measure the remaining superposition on inputs in either the standard or Hadamard basis. A classical adversary that can answer each query independently must also be able to answer both at the same time, which is ruled out by the adaptive hard core property.

We propose to enable the quantum device to generate a superposition over $(0,x_0,H(0,x_0))$ and $(1,x_1,H(1,x_1))$, where H is a one-bit hash function modeled as a random oracle. This can be done in a straightforward manner, similar to the previous method. The device is then asked to measure the resulting state in the Hadamard basis (always), and send the outcomes obtained to the verifier.[4] Since the device makes a single measurement, there is no need for a challenge from the verifier, which effectively collapses the protocol to two messages. A quick calculation shows that the verifier receives a bit m and vector d s.t. in the case of a honest behavior the equation $m = d \cdot (x_0 \oplus x_1) \oplus H(0,x_0) \oplus H(1,x_1)$ holds. Finally, the verifier uses the trapdoor to recover x_0, x_1 from y and checks that the equation is satisfied. The crux of the security proof is that a classical adversary cannot query the oracle at both $(0,x_0)$ and $(1,x_1)$, otherwise it would have been able to find a claw and break the cryptographic assumption. Therefore at least one value out of $H(0,x_0)$ and $H(1,x_1)$ remains random, and thus the adversary cannot compute m, d that adhere to the required equation with probability greater than $1/2$. The proof thus follows from a simple extraction-style argument. In our main protocol, we use parallel repetition to argue that no prover can succeed with non-negligible probability.

---

[3] Very recently, two concurrent works by Alagic et al. [2] and Chia et al. [6] showed that parallel repetition of Mahadev's protocol indeed achieves negligible soundness error.

[4] In fact we use a slight variant of this protocol, since measuring the H part in Hadamard basis has probability $1/2$ of erasing the information on that bit. Instead we append the H values directly to the phase. This is immaterial for the purpose of this exposition.

Discussion on the "Random Oracle" Heuristic

As discussed above, the Fiat-Shamir heuristic can be used for the quantum supremacy protocol of Brakerski et al. [3]. However, this would mean that the resulting scheme would require stronger assumptions (in particular, it would require noisy TCFs with the adaptive hardcore bit property). Secondly, starting with the work of Canetti et al. [5], many works have shown uninstantiability of the random oracle. These works show certain cryptographic primitives which are secure in the random oracle model, but are broken when instantiated by any concrete hash function. However, these constructions are very contrived, and in particular, do not apply to our protocol.

Efficiency of our Protocol, and Comparison to Previous Approaches

We would like to emphasize that at the current level of maturity of quantum technology, any estimate of "practical advantage" would be educated guesswork at best. The technology for any option is far from being available and it is hard to predict the direction that technology will take, and as a consequence the practical cost of implementing certain operations.

This state of affairs, we believe, highlights the importance of developing multiple approaches to tasks such as proof of quantumness. This way, an assortment of solutions will be ready to accommodate the different directions that technology may lead.

A second point that we wish to highlight before getting into technical calculations, is that our approach allows to use *any* family of trapdoor claw free permutations (and as we point out, for proofs of quantumness even "pre-quantum" candidates will suffice, e.g. if a candidate can be devised based on DDH in EC groups). This means that our back of the envelope calculation only refers to one specific way of using our scheme. Currently, we do not know any candidates for trapdoor claw free permutations based on such "pre-quantum" assumptions.

Our protocol can be executed using a quasi-linear number of qubits and, with the proper choice of candidate for the hash function, has quasi-linear computational complexity.

Comparison with [3]: Since we do not require the hardcore bit property, our input dimension n is smaller by a factor of at least $60\log(\lambda)$. This follows due to Lemma 4.2 in [3]. Also, note that the parameter q must also grow, hence the overall number of qubits required to implement the protocol in [3] is $O(\lambda \log^3(\lambda))$, at least $100\log(\lambda)$ times more. Secondly, since [3] is a four-round protocol, the prover must maintain its quantum state until it receives a challenge from the verifier.

Comparison to discrete log via Shor's algorithm: Let n denote the number of bits required for representing the group elements. The current estimates for the number of qubits required for discrete log are $3n$, while the number of quantum gates required is $0.3n^3$ (see [8]). Similar to Shor's algorithm for factoring/discrete log, our protocol is also a non-interactive one (that is, the verifier sends a challenge, and the prover responds with an answer).

Open Problems

Our work suggests a number of open problems in the context of utilizing random oracles in the regime of classical verification of quantum computation. Most desirably, whether it is possible to use the random oracle in order to eliminate the need for other assumptions, or at least the need for a trapdoor. Obtaining a publicly verifiable protocol is a highly desirable goal. We can also wonder whether our protocol can be used for the purposes of certifying randomness or verifying quantum computation. In the plain model, the adaptation of the proof of quantumness for these purposes was far from trivial and yet the protocol itself is almost identical. Improving the state of the art in certifying randomness and in verifiability using random oracles (or using other methods) is also an interesting open problem.

2 Preliminaries

2.1 Notations

For an integer n we write $[n]$ for the set $\{1,\ldots,n\}$. For any finite set X, let $x \leftarrow \mathcal{X}$ denote a uniformly random element drawn from X. Similarly, for any distribution $\mathcal{D}$, let $x \leftarrow \mathcal{D}$ denote a sample from $\mathcal{D}$. For an element $x \in X$ we write $\mathsf{BitDecomp}(x)$ for an arbitrarily chosen but canonical (depending only on the implicit set X) binary representation of x. For any density function f on domain X, let $\textsc{Supp}(f)$ denote the support of f; that is $\textsc{Supp}(f) = \{x \in X : f(x) > 0\}$.

For density functions f_1, f_2 over the same finite domain X, the Hellinger distance between f_1 and f_2 is

$$H^2(f_1,f_2) = 1 - \sum_{x\in X} \sqrt{f_1(x)f_2(x)}.$$

The total variation distance between f_1 and f_2 is

$$\|f_1 - f_2\|_{\mathrm{TV}} = \frac{1}{2}\sum_{x\in X} |f_1(x) - f_2(x)| \le \sqrt{2H^2(f_1,f_2)}.$$

The following lemma relates the Hellinger distance and the trace distance of superpositions.

▶ **Lemma 1.** *Let X be a finite set and f_1, f_2 two density functions on X. Let*

$$|\psi_1\rangle = \sum_{x\in X} \sqrt{f_1(x)}\,|x\rangle\,, \text{ and } |\psi_2\rangle = \sum_{x\in X} \sqrt{f_2(x)}\,|x\rangle\,.$$

Then

$$\||\psi_1\rangle - |\psi_2\rangle\|_{\mathrm{tr}} \le \sqrt{1-(1-H^2(f_1,f_2))^2}.$$

2.2 Ideal Lattices

In this section, we present some background on ideal lattices, the truncated discrete Gaussian distribution and the Ring Learning with Errors problem. For a positive integer B, modulus q, and dimension n, the truncated discrete Gaussian distribution is a distribution with support $\left\{x \in \mathbb{Z}_q^n : \|x\| \le B\sqrt{n}\right\}$ defined as follows:

$$D_{\mathbb{Z}_q^n,B}(x) = \frac{\exp\left(-\pi\|x\|^2/B^2\right)}{\sum_{z\in\mathbb{Z}_q^n,\|z\|\le B\sqrt{n}} \exp(-\pi|z|^2/B^2)}.$$

The Ring Learning with Errors (RLWE) assumption[11] is parameterized by a ring R, modulus $q \in \mathbb{N}$ and a noise distribution χ. Informally, the assumption states that given many samples of the form $(a, a\cdot s + e)$ where s is fixed for all samples, a is chosen uniformly at random and e is chosen from the error distribution χ for each sample, it is hard to compute s. The formal definition is given below. Here, we restrict ourselves to a special family of *cyclotomic rings*.

▶ **Assumption 1.** *Let n be a power of two, $f_n(X) = X^n + 1$ an irreducible polynomial over $\mathbb{Q}[X]$ and $R_n = \mathbb{Z}[X]/(f_n(X))$. Let $q = \{q_n\}_{n\in\mathbb{N}}$ be a family of moduli, $R_{n,q_n} = R_n/q_nR_n = \mathbb{Z}_{q_n}[X]/(f_n(X))$ the quotient space, and $\chi = \{\chi_n\}_{n\in\mathbb{N}}$ a family of error distributions, where χ_n is a distribution over R_{n,q_n}. For any secret s in R_{n,q_n}, let $\mathcal{O}_s$ denote the oracle that, on*

each query, chooses $a \leftarrow R_{n,q_n}$, $e \leftarrow \chi_n$ *and outputs* $(a, a \cdot s + e \mod q_n)$. *The Ring Learning with Errors assumption* $\mathsf{RLWE}_{R,q,\chi}$, *parameterized by the family of rings* $\{R_n\}_{n=2^k,k\in\mathbb{N}}$, *moduli family* q *and distribution family* χ, *states that for any PPT adversary* $\mathcal{A}$, *there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all security parameters* $n = 2^k, k \in \mathbb{N}$,

$$\Pr\left[s \leftarrow \mathcal{A}^{O_s()}(1^n) : s \leftarrow R_{n,q_n}\right] \leq \mathsf{negl}(n).$$

Given many samples $\{a_i, a_i \cdot s + e_i\}_i$, one can efficiently find s using a *trapdoor* for the public elements $\{a_i\}_i$. There exists a sampling algorithm that can sample $\{a_i\}_i$ together with a trapdoor τ, and an inversion algorithm that uses τ to extract s from the set of evaluations $\{a_i \cdot s + e_i\}_i$. Without the trapdoor, the public elements $\{a_i\}_i$ look uniformly random.

▶ **Theorem 2** (Theorem 5.1 of [13] in the Ring setting). *Let* n, m, q *be such that* n *is a power of* 2, $m = \Omega(\log q)$. *There is an efficient randomized algorithm* GenTrap *that takes as input* $(1^n, 1^m, q)$, *and returns* $\mathbf{a} = (a_i)_i \in R_{n,q}^m$ *and a trapdoor* τ *such that the distribution of* $\mathbf{a}$ *is negligibly (in* n*) close to the uniform distribution over* $R_{n,q}^m$. *Moreover, there is an efficient algorithm* Invert *and a universal constant* C_T *such that the following holds with overwhelming probability over the choice of* $(\mathbf{a}, \tau) \leftarrow \textsc{GenTrap}(1^n, 1^m, q)$*:*

$$\text{for all } s \in R_{n,q}, \mathbf{e} \text{ such that } \|\mathbf{e}\| \leq \frac{q}{C_T\sqrt{n \log q}}, \textsc{Invert}(\mathbf{a}, \tau, \mathbf{a} \cdot s + \mathbf{e}) = s.$$

2.3 Noisy Trapdoor Claw-Free Hash Functions

In this section we introduce the notion of noisy trapdoor claw-free functions (NTCFs). Let $\mathcal{X}, \mathcal{Y}$ be finite sets and $\mathcal{K}$ a set of keys. For each $k \in K$ there should exist two (efficiently computable) injective functions $f_{k,0}, f_{k,1}$ that map $\mathcal{X}$ to $\mathcal{Y}$, together with a trapdoor t_k that allows efficient inversion from $(b, y) \in \{0,1\} \times \mathcal{Y}$ to $f_{k,b}^{-1}(y) \in \mathcal{X} \cup \{\perp\}$. For security, we require that for a randomly chosen key k, no polynomial time adversary can efficiently compute $x_0, x_1 \in \mathcal{X}$ such that $f_{k,0}(x_0) = f_{k,1}(x_1)$ (such a pair (x_0, x_1) is called a *claw*).

Unfortunately, we do not know how to construct such "clean" trapdoor claw-free functions. Hence, as in previous works [3, 12], we will use "noisy" version of the above notion. For each $k \in \mathcal{K}$, there exist two functions $f_{k,0}, f_{k,1}$ that map $\mathcal{X}$ to a distribution over $\mathcal{Y}$.

The following definition is taken directly from [3].

▶ **Definition 3** (NTCF family). *Let* λ *be a security parameter. Let* $\mathcal{X}$ *and* $\mathcal{Y}$ *be finite sets. Let* $\mathcal{K}_\mathcal{F}$ *be a finite set of keys. A family of functions*

$$\mathcal{F} = \left\{f_{k,b} : \mathcal{X} \to \mathcal{D}_\mathcal{Y}\right\}_{k\in\mathcal{K}_\mathcal{F}, b\in\{0,1\}}$$

is called a noisy trapdoor claw free (NTCF) family *if the following conditions hold:*

1. ***Efficient Function Generation.*** *There exists an efficient probabilistic algorithm* $\textsc{Gen}_\mathcal{F}$ *which generates a key* $k \in \mathcal{K}_\mathcal{F}$ *together with a trapdoor* t_k*:*

   $$(k, t_k) \leftarrow \textsc{Gen}_\mathcal{F}(1^\lambda) .$$

2. ***Trapdoor Injective Pair.***
   a. *Trapdoor: There exists an efficient deterministic algorithm* $\textsc{Inv}_\mathcal{F}$ *such that with overwhelming probability over the choice of* $(k, t_k) \leftarrow \textsc{Gen}_\mathcal{F}(1^\lambda)$, *the following holds:*

      $$\text{for all } b \in \{0,1\}, x \in \mathcal{X} \text{ and } y \in \textsc{Supp}(f_{k,b}(x)), \textsc{Inv}_\mathcal{F}(t_k, b, y) = x.$$

b. *Injective pair: For all keys $k \in \mathcal{K}_{\mathcal{F}}$, there exists a perfect matching $\mathcal{R}_k \subseteq \mathcal{X} \times \mathcal{X}$ such that $f_{k,0}(x_0) = f_{k,1}(x_1)$ if and only if $(x_0, x_1) \in \mathcal{R}_k$.*

3. ***Efficient Range Superposition.*** *For all keys $k \in \mathcal{K}_{\mathcal{F}}$ and $b \in \{0,1\}$ there exists a function $f'_{k,b} : \mathcal{X} \to \mathcal{D}_{\mathcal{Y}}$ such that the following hold.*
   a. *For all $(x_0, x_1) \in \mathcal{R}_k$ and $y \in \text{Supp}(f'_{k,b}(x_b))$, $\text{Inv}_{\mathcal{F}}(t_k, b, y) = x_b$ and $\text{Inv}_{\mathcal{F}}(t_k, b \oplus 1, y) = x_{b\oplus 1}$.*
   b. *There exists an efficient deterministic procedure $\text{Chk}_{\mathcal{F}}$ that, on input k, $b \in \{0,1\}$, $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, returns 1 if $y \in \text{Supp}(f'_{k,b}(x))$ and 0 otherwise. Note that $\text{Chk}_{\mathcal{F}}$ is not provided the trapdoor t_k.*
   c. *For every k and $b \in \{0,1\}$,*

$$E_{x \leftarrow_U \mathcal{X}}\left[H^2(f_{k,b}(x),\, f'_{k,b}(x)) \right] \leq 1/50\ .^5$$

Here H^2 is the Hellinger distance. Moreover, there exists an efficient procedure $\text{Samp}_{\mathcal{F}}$ that on input k and $b \in \{0,1\}$ prepares the state

$$\frac{1}{\sqrt{|\mathcal{X}|}} \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} \sqrt{(f'_{k,b}(x))(y)}\, |x\rangle\, |y\rangle\ . \tag{1}$$

4. ***Claw-Free Property.*** *For any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that the following holds:*

$$\Pr\left[(x_0, x_1) \in \mathcal{R}_k : (k, t_k) \leftarrow \text{Gen}_{\mathcal{F}}(1^\lambda), (x_0, x_1) \leftarrow \mathcal{A}(k)\right] \leq \mathsf{negl}(\lambda)$$

3 Proof of Quantumness Protocol

We will now present our protocol. Throughout the protocol, we will ignore dependence on the security parameter when clear from context. Let $\mathcal{F}$ be a NTCF family with domain $\mathcal{X}$, range $\mathcal{Y}$ described by the algorithms $\text{Gen}_{\mathcal{F}}, \text{Inv}_{\mathcal{F}}, \text{Chk}_{\mathcal{F}}, \text{Samp}_{\mathcal{F}}$. Let w denote the length of bit decomposition of elements of $\mathcal{X}$. Finally, let H be a hash function that maps $\mathcal{X}$ to $\{0,1\}$.

Proof of Quantumness Protocol

The protocol is parameterized by a hash function $H : \{0,1\}^n \to \{0,1\}$ (which will be modeled as a random oracle in the security proof).

1. The verifier generates $(k, \kappa) \leftarrow \text{Gen}_{\mathcal{F}}(1^\lambda)$ and sends k to the prover.
2. The prover sends λ tuples $\{(y_i, m_i, d_i)\}_{i \in [\lambda]}$. The verifier initializes $\mathsf{count} = 0$ and performs the following checks:
   a. It checks that all values in $\{y_i\}_i$ are distinct.
   b. It computes $x_{i,b} = \text{Inv}_{\mathcal{F}}(\kappa, b, y_i)$ for each $i \in [\lambda]$, $b \in \{0,1\}$. Next, it checks if $m_i = d_i^T \cdot (\mathsf{BitDecomp}(x_{i,0}) + \mathsf{BitDecomp}(x_{i,1})) + H(x_{i,0}) + H(x_{i,1})$. If this check passes, it increments the value of count by 1.
3. If $\mathsf{count} > 0.75\lambda$, the verifier outputs 1, else it outputs $\perp$.

Figure 1 Protocol for Proof of Quantumness.

▶ **Theorem 4.** *Let $\mathcal{F}$ be a family of NTCF functions satisfying Definition 3. Then Protocol 1 satisfies the following properties:*

- *Completeness: There exists a quantum polynomial-time prover $\mathcal{P}$ and a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ and hash functions H, $\mathcal{P}$ succeeds in the protocol with probability at least $1 - \mathsf{negl}(\lambda)$.*
- *Proof of Quantumness: For any PPT (classical) adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\mathcal{A}$ succeeds in the protocol with probability at most $\mathsf{negl}(\lambda)$ where H is modeled as a random oracle.*

3.1 Completeness

In this section, we show that the honest (quantum) prover is accepted by the verifier.

The honest prover receives NTCF key k. It does the following:

1. It starts with λ copies of the state $|0\rangle|0\rangle|0\rangle|-\rangle$. For each $i \in [\lambda]$, let $|\psi_i\rangle = |0\rangle|0\rangle|0\rangle|-\rangle$. It then applies $\textsc{Samp}_{\mathcal{F}}$ to the first three registers of $|\psi_i\rangle$ for each i, resulting in the state $\left|\psi_i'^{(1)}\right\rangle$, where

$$\left|\psi_1'^{(1)}\right\rangle = \left(\frac{1}{\sqrt{2|\mathcal{X}|}} \sum_{x\in\mathcal{X}, y\in\mathcal{Y}, b\in\{0,1\}} \sqrt{(f'_{k,b}(x))(y)}\,|b\rangle|x\rangle|y\rangle\right)|-\rangle\,. \tag{2}$$

   This quantum state is at distance at most 0.2 from the following quantum state:

$$\left|\psi_i^{(1)}\right\rangle = \left(\frac{1}{\sqrt{2|\mathcal{X}|}} \sum_{x\in\mathcal{X}, y\in\mathcal{Y}, b\in\{0,1\}} \sqrt{(f_{k,b}(x))(y)}\,|b\rangle|x\rangle|y\rangle\right)|-\rangle\,. \tag{3}$$

2. Next, it measures the third register, obtaining measurement $y \in \mathcal{Y}$. Let $x_0, x_1 \in \mathcal{X}$ be the unique elements such that y is in the support of $f_{k,b}(x_b)$. Applying this operation to the state in (3), the resulting state (ignoring the measured register) is

$$\left|\psi_i^{(2)}\right\rangle = \left(\frac{1}{\sqrt{2}}\left(|0\rangle|x_0\rangle + |1\rangle|x_1\rangle\right)\right)|-\rangle\,. \tag{4}$$

3. Let U_H be a unitary that maps $|a\rangle|b\rangle$ to $|a\rangle|b + H(a)\rangle$. The prover applies U_H to the second and third register. On applying this operation to the state in (4), the new state is

$$\left|\psi_i^{(3)}\right\rangle = \frac{1}{2}\left(\sum_{b,b'}(-1)^{b'}|b\rangle|x_b\rangle|b' + H(x_b)\rangle\right). \tag{5}$$

4. The prover then evaluates the function $\mathsf{BitDecomp}$ on the second register. Applying this to (5), the resulting state is

$$\left|\psi_i^{(4)}\right\rangle = \frac{1}{2}\left(\sum_{b,b'}(-1)^{b'}|b\rangle|\mathsf{BitDecomp}(x_b)\rangle|b' + H(x_b)\rangle\right). \tag{6}$$

5. Finally, the prover applies the Hadamard operator to all registers. On applying this to (6), this produces the state (where $h_b = H(x_b)$ and $\overline{x}_b = \mathsf{BitDecomp}(x_b)$)

$$\begin{aligned}\left|\psi_i^{(5)}\right\rangle &= \frac{1}{\sqrt{2^{w+4}}} \sum_{b,b'\in\{0,1\}} \sum_{\substack{m,m'\in\{0,1\},\\ d\in\{0,1\}^w}} (-1)^{m\cdot b + d^T\cdot\overline{x}_b + m'\cdot b' + m'\cdot h_b + b'}|m\rangle|d\rangle|m'\rangle \\ &= \frac{1}{\sqrt{2^{w+2}}} \sum_{m\in\{0,1\}, d\in\{0,1\}^w} |m\rangle|d\rangle|1\rangle\left((-1)^{d^T\cdot\overline{x}_0 + h_0} + (-1)^{m + d^T\cdot x_1 + h_1}\right)\end{aligned} \tag{7}$$

Upon measurement of the state in (7), the output tuple $(m, d, 1)$ satisfies $m = d^T \cdot (\overline{x}_0 + \overline{x}_1) + h_0 + h_1$ (with probability 1). As a result, applying the above operations to $\left|\psi_i'^{(1)}\right\rangle$ results in a tuple (y, m, d) that is accepted with probability at least 0.8. Using a Chernoff bound it is straightforward to argue that there exists a negligible function $\mathsf{negl}(\cdot)$ such that with probability at least $1 - \mathsf{negl}(\lambda)$, at least 3/4 fraction of the tuples in $\{(y_i, m_i, d_i)\}$ pass the verification.

3.2 Proof of Quantumness : Classical Prover's Advantage in the Random Oracle Model

Here, we will show that if the function H is replaced with a random oracle, then any classical algorithm that has non-negligble advantage in Protocol 1 can be used to break the claw-free property of $\mathcal{F}$. Consider the following security experiment which captures the interaction between a (classical) prover and a challenger in the random oracle model; the challenger represents the verifier in the protocol.

Experiment 1

In this experiment, the challenger represents the verifier in Protocol 1 and also responds to the random oracle queries issued by the prover.

1. The challenger (verifier) chooses an NTCF key $(k, \kappa) \leftarrow \textsc{Gen}_{\mathcal{F}}(1^\lambda)$ and sends k to the prover. The prover and challenger have access to a random oracle $H : \{0,1\}^n \to \{0,1\}$.
2. The prover sends $\{(y_i, m_i, d_i)\}_{i\in[\lambda]}$. For each $i \in [\lambda]$, the challenger computes $x_{i,b} \leftarrow \textsc{Inv}_{\mathcal{F}}(\kappa, b, y_i)$ for $b \in \{0,1\}$, queries the random oracle H on $x_{i,0}, x_{i,1}$ and receives $h_{i,0}, h_{i,1}$ respectively. Next, it checks if $m_i = d_i^T \cdot (\mathsf{BitDecomp}(x_{i,0}) + \mathsf{BitDecomp}(x_{i,1})) + h_{i,0} + h_{i,1}$. If at least 0.75λ tuples satisfy the check, it outputs 1, else it outputs $\perp$.

Experiment 2

This experiment is similar to the previous one, except that the challenger implements the random oracle, and does not use the trapdoor for performing the final λ checks.

1. The challenger (verifier) chooses an NTCF key $(k, \kappa) \leftarrow \textsc{Gen}_{\mathcal{F}}(1^\lambda)$ and sends k to the prover. The challenger also implements the random oracle as follows. It maintains a database which is initially empty. On receiving a query x, it checks if there exists a tuple (x, h) in the database. If so, it outputs h, else it chooses a random bit $h \leftarrow \{0,1\}$, adds (x, h) to the database and outputs h.
2. The prover sends $\{(y_i, m_i, d_i)\}_{i\in[\lambda]}$. On receiving this set from the prover, the challenger does not compute the inverses of y_i. Instead, it initializes $\mathsf{count} = 0$, and for each i, it looks for tuples $(x_{i,0}, h_{i,0})$ and $(x_{i,1}, h_{i,1})$ in the table such that $\textsc{Chk}_{\mathcal{F}}(y_i, 0, x_{i,0}) = \textsc{Chk}_{\mathcal{F}}(y_i, 1, x_{i,1}) = 1$. If such $(x_{i,0}, x_{i,1})$ do not exist, then the challenger chooses a random bit r_i and sets $\mathsf{count} = \mathsf{count} + r_i$. Else, it checks if $m_i = d_i^T \cdot (\mathsf{BitDecomp}(x_{i,0}) + \mathsf{BitDecomp}(x_{i,1})) + h_{i,0} + h_{i,1}$. If so, it increments count.
   Finally, it checks if $\mathsf{count} > 0.75\lambda$. If so, it outputs 1, else outputs $\perp$.

Experiment 3

This experiment is identical to the previous one, except that the challenger, after receiving $\{(y_i, m_i, d_i)\}_i$, outputs $\perp$ if for all $i \in [\lambda]$, there does not exist two entries $(x_{i,0}, h_{i,0}), (x_{i,1}, h_{i,1})$ in the database such that $\textsc{Chk}_{\mathcal{F}}(y_i, 0, x_{i,0}) = \textsc{Chk}_{\mathcal{F}}(y_i, 1, x_{i,1}) = 1$.

3.2.1 Analysis

For any classical PPT prover $\mathcal{A}$, let $p_{\mathcal{A}}$ denote the probability that the verifier outputs 1 in Protocol 1 (when H is replaced with a random oracle), and for $w \in \{1,2,3\}$, let $p_{\mathcal{A},w}$ denote the probability that the challenger interacting with $\mathcal{A}$ in Experiment w outputs 1. From the definition of Experiment 1 it follows that $p_{\mathcal{A}} = p_{\mathcal{A},1}$.

▷ Claim 5. For any prover $\mathcal{A}$, $p_{\mathcal{A},1} = p_{\mathcal{A},2}$.

Proof. The main differences between Experiment 1 and Experiment 2 are that the challenger implements the random oracle, and secondly, after receiving $\{(y_i, m_i, d_i)\}_i$, the challenger does not use the trapdoor for checking. Note that in Experiment 1, if either $x_{i,0}$ or $x_{i,1}$ are not queried to the random oracle H, then $H(x_{i,0}) + H(x_{i,1})$ is a uniformly random bit. Moreover, since the y_i values are distinct, if there exist two indices i, j such that both the preimages of y_i and y_j are not queried, then $H(x_{i,0}) + H(x_{i,1})$ is independent of $H(x_{j,0}) + H(x_{j,1})$. As a result, for each index i such that the preimages of y_i are not queried, the value of count is incremented with probability $1/2$.

In Experiment 2, the challenger checks for pairs corresponding to $x_{i,0}$ and $x_{i,1}$ in the database, and if either of them is missing, it increments count with probability $1/2$. As a result, the probability of $\mathsf{count} > 0.75\lambda$ is identical in both experiments. ◁

▷ Claim 6. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any prover $\mathcal{A}$ and any security parameter $\lambda \in \mathbb{N}$, $p_{\mathcal{A},2} \leq p_{\mathcal{A},3} + \mathsf{negl}(\lambda)$.

Proof. The only difference between these two experiments is that the challenger, at the end of the experiment, outputs $\perp$ if for all $i \in [\lambda]$, either $x_{i,0}$ or $x_{i,1}$ has not been queried to the random oracle. The only case in which the challenger outputs 1 in Experiment 2 but outputs $\perp$ in Experiment 3 is when for all $i \in [\lambda]$, either $x_{i,0}$ or $x_{i,1}$ has not been queried, but there exist $t \geq 0.75\lambda$ indices $\{i_1, \ldots, i_t\}$ such that count was incremented. Using Chernoff bounds, we can show that this happens with negligible probability. ◁

▷ Claim 7. Assuming $\mathcal{F}$ is a secure claw-free trapdoor family, for any PPT prover $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $p_{\mathcal{A},3}(\lambda) \leq \mathsf{negl}(\lambda)$.

Proof. Suppose there exists a PPT prover $\mathcal{A}$ and a non-negligible function $\epsilon(\cdot)$ such that for all $\lambda \in \mathbb{N}$, the challenger outputs 1 with probability $\epsilon = \epsilon(\lambda)$ in Experiment 3. This means with probability at least ϵ, there exists an index $i^* \in [\lambda]$ such that $\mathcal{A}$ queries the random oracle on $x_{i^*,0}, x_{i^*,1}$ and finally outputs $\{(y_i, m_i, d_i)\}_i$ such that $\textsc{Chk}_{\mathcal{F}}(y_{i^*}, 0, x_{i^*,0}) = \textsc{Chk}_{\mathcal{F}}(y_{i^*}, 1, x_{i^*,1}) = 1$.

We will construct a reduction algorithm $\mathcal{B}$ that breaks the claw-free property of $\mathcal{F}$ with probability ϵ. The reduction algorithm receives the key k from the NTCF challenger, which it forwards to $\mathcal{A}$. Next, $\mathcal{A}$ makes polynomially many random oracle queries, which are answered by the reduction algorithm by maintaining a database. Eventually, $\mathcal{A}$ sends $\{(y_i, m_i, d_i)\}$. The reduction algorithm checks if there exist tuples $(x_{i^*,0}, h_{i^*,0})$ and $(x_{i^*,1}, h_{i^*,1})$ in its database such that $\textsc{Chk}_{\mathcal{F}}(y_{i^*}, 0, x_{i^*,0}) = \textsc{Chk}_{\mathcal{F}}(y_{i^*}, 1, x_{i^*,1}) = 1$. If so, it sends $(x_{i^*,0}, x_{i^*,1})$ to the NTCF challenger. ◁

Using the above claims, it follows for every classical prover $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $p_{\mathcal{A}} \leq \mathsf{negl}(\lambda)$.

4 Construction of NTCFs based on Ring LWE

Our construction is similar to the one in [3]. Let λ be the security parameter, $n = 2^{\lceil \log \lambda \rceil}$. The following are other parameters chosen by our scheme (we will ignore dependence on security parameter/n):

- Ring $R = \mathbb{Z}[X]/(X^n + 1)$.
- Modulus $q = \mathsf{poly}(n)$, $R_q = R/qR$
- $m = \Omega(\log q)$: determines the dimension of range space
- χ: the noise distribution. In our case, χ is a Discrete Gaussian over $\mathbb{Z}^n$ with parameter B_V.
- B_P : the noise bound for function evaluation. We require the following constraints on B_P:
  - $B_P \geq \Omega(n \cdot m \cdot B_V)$
  - $2B_P\sqrt{n \cdot m} \leq q/(C_T \cdot \sqrt{n \log q})$ for some constant C_T

The domain is $\mathcal{X} = R_q$, and range is $\mathcal{Y} = R_q^m$.

Each function key $k = (\mathbf{a}, \mathbf{a} \cdot s + \mathbf{e})$, where $s \in R_q$, $a_i, e_i \in R_q$ for all $i \in [m]$, $\mathbf{a} = [a_1 \dots a_m]^T$, $\mathbf{e} = [e_1 \dots e_m]^T$. For $b \in \{0,1\}, x \in \mathcal{X}, k = (\mathbf{a}, \mathbf{a} \cdot s + \mathbf{e})$, the density function $f_{k,b}(x)$ is defined as follows:

$$\forall \mathbf{y} \in \mathcal{Y}, (f_{k,b}(x))(\mathbf{y}) = D_{\mathbb{Z}^{n \cdot m}, B_P}(\mathbf{y} - \mathbf{a} \cdot x - b \cdot \mathbf{a} \cdot s), \tag{8}$$

where $\mathbf{y} = [y_1 \dots y_m]^T$, and each $\mathbf{y}_i$ can be represented as an element in $\mathbb{Z}_q^n$ (using the coefficient representation); similarly for $\mathbf{a} \cdot x$ and $\mathbf{a} \cdot s$.

We will now show that each of the properties of NTCFs hold.

1. **Efficient Key Generation**: The key generation algorithm $\textsc{Gen}_{\mathcal{F}}(1^\lambda)$ first chooses $(\mathbf{a}, \tau) \leftarrow \textsc{GenTrap}(1^n, 1^m, q)$, $s \leftarrow R_q$ and $\mathbf{e} \leftarrow \chi^m$. It outputs key $k = (\mathbf{a}, \mathbf{a} \cdot s + \mathbf{e})$, and the trapdoor is $\kappa = (\tau, k, s)$.
2. **Trapdoor Injective Pair**:
   a. *Trapdoor*: For $k = (\mathbf{a}, \mathbf{a} \cdot s + \mathbf{e})$, $x \in \mathcal{X}$ and $b \in \{0,1\}$, the support of $f_{k,b}(x)$ is
   $$\textsc{Supp}(f_{k,b}(x)) = \left\{\mathbf{y} \in \mathcal{Y} : \|\mathbf{y} - \mathbf{a} \cdot x - b \cdot \mathbf{a} \cdot s\| \leq B_P\sqrt{n \cdot m}\right\}$$
   The inversion algorithm $\textsc{Inv}_{\mathcal{F}}$ takes as input the lattice trapdoor τ, $b \in \{0,1\}$, $\mathbf{y} \in \mathcal{Y}$ and outputs $\textsc{Invert}(\tau, \mathbf{a}, \mathbf{y}) - b \cdot s$. From Theorem 2, it follows that with overwhelming probability over the choice of $\mathbf{a}$, for all $\mathbf{y} \in \textsc{Supp}(f_{k,b}(x))$, $\textsc{Invert}(\tau, \mathbf{a}, \mathbf{y}) = x + b \cdot s$. Hence, it follows that $\textsc{Inv}_{\mathcal{F}}(\kappa, b, \mathbf{y}) = x$.
   b. *Injective Pair*: Let $k = (\mathbf{a}, \mathbf{a} \cdot s + \mathbf{e})$. From the construction, it follows that $f_{k,0}(x_0) = f_{k,1}(x_1)$ if and only if $x_1 = x_0 + s$. Hence the set $\mathcal{R}_k = \{(x, x + s) : x \in \mathcal{X}\}$.
3. **Efficient Range Superposition**: The function $f'_{k,0}$ is same as $f_{k,0}$, while $f'_{k,1}$ is defined as follows (recall $k = (\mathbf{a}, \mathbf{a} \cdot s + \mathbf{e})$):
   $$\forall \mathbf{y} \in \mathcal{Y}, (f'_{k,1}(x))(\mathbf{y}) = D_{\mathbb{Z}^{n \cdot m}, B_P}(\mathbf{y} - \mathbf{a} \cdot x - (\mathbf{a} \cdot s + \mathbf{e})) \tag{9}$$
   a. Since $f'_{k,0} = f_{k,0}$, it follows that for all $(x_0, x_1) \in \mathcal{R}_k$ and $\mathbf{y} \in \textsc{Supp}(f'_{k,0}(x_0))$, $\textsc{Inv}_{\mathcal{F}}(\kappa, 0, \mathbf{y}) = x_0$ and $\textsc{Inv}_{\mathcal{F}}(\kappa, 1, \mathbf{y}) = x_1$. We need to show the same for $f'_{k,1}$; that is, for all $(x_0, x_1) \in \mathcal{R}_k$ and $\mathbf{y} \in \textsc{Supp}(f'_{k,1}(x_1))$, $\textsc{Inv}_{\mathcal{F}}(\kappa, 1, \mathbf{y}) = x_1$ and $\textsc{Inv}_{\mathcal{F}}(\kappa, 0, \mathbf{y}) = x_0$. For all $x \in \mathcal{X}$,
   $$\textsc{Supp}(f'_{k,1}(x)) = \left\{\mathbf{y} \in \mathcal{Y} : \|\mathbf{y} - \mathbf{a} \cdot x - \mathbf{a} \cdot s - \mathbf{e}\| \leq B_P\sqrt{n \cdot m}\right\}$$
   Hence for any $\mathbf{y} \in \textsc{Supp}(f'_{k,1}(x))$, $\|\mathbf{y} - \mathbf{a} \cdot x_1 - \mathbf{a} \cdot s\| \leq 2B_P\sqrt{n \cdot m}$; using Theorem 2, we can conclude that $\textsc{Inv}_{\mathcal{F}}(\kappa, 1, \mathbf{y}) = x_1$.

b. The procedure $\text{CHK}_{\mathcal{F}}$ takes as input $\mathbf{y} \in \mathcal{Y}, k = (\mathbf{a}, \mathbf{v}), b \in \{0,1\}, x \in \mathcal{X}$ and checks if $\|\mathbf{y} - \mathbf{a} \cdot x - b \cdot \mathbf{v}\| \leq B_P \sqrt{n \cdot m}$.

c. The definition of $\text{SAMP}_{\mathcal{F}}$ is identical to the one in [3], and the Hellinger distance can be bounded by $1 - e^{-\frac{2\pi m \cdot n \cdot B_V}{B_P}}$.From our setting of parameters, this quantity is at most $1/50$.

4. **Claw-Free Property** Suppose there exists an adversary $\mathcal{A}$ that, on input $k = (\mathbf{a}, \mathbf{a} \cdot s + \mathbf{e})$ can output $(x_0, x_1) \in \mathcal{R}_k$. Then this adversary can be used to break the Ring LWE assumption, since $x_1 - x_0 = s$.

References

1 Scott Aaronson and Alex Arkhipov. The computational complexity of linear optics. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 333–342, 2011.

2 Gorjan Alagic, Andrew M. Childs, Alex B. Grilo, and Shih-Han Hung. Non-interactive classical verification of quantum computation, 2019. `arXiv:1911.08101`.

3 Zvika Brakerski, Paul Christiano, Urmila Mahadev, Umesh V. Vazirani, and Thomas Vidick. A cryptographic test of quantumness and certifiable randomness from a single quantum device. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 320–331, 2018.

4 Michael J Bremner, Richard Jozsa, and Dan J Shepherd. Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 467(2126):459–472, 2010.

5 Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. of the ACM*, 51(4):557–594, 2004.

6 Nai-Hui Chia, Kai-Min Chung, and Takashi Yamakawa. Classical verification of quantum computations with efficient verifier. *ArXiv*, abs/1912.00990, 2019.

7 Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, pages 186–194, 1986.

8 Craig Gidney and Martin Ekerå. How to factor 2048 bit rsa integers in 8 hours using 20 million noisy qubits. *arXiv preprint arXiv:1905.09749*, 2019.

9 Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A "paradoxical" solution to the signature problem. In George Robert Blakley and David Chaum, editors, *Advances in Cryptology*, 1985.

10 Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, pages 1–23, 2010.

11 Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for ring-lwe cryptography. *IACR Cryptology ePrint Archive*, 2013:293, 2013.

12 Urmila Mahadev. Classical verification of quantum computations. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 259–267, 2018.

13 Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, pages 700–718, 2012.

14 NIST. Candidate quantum-resistant cryptographic algorithms publicly available. URL: `https://www.nist.gov/news-events/news/2017/12/candidate-quantum-resistant-cryptographic-algorithms-publicly-available`.

15 Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 84–93, 2005.

16 Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 124–134, 1994.