A Survey on Digital Signatures in Quantum Setting

William Ling Chen, Kamil Doruk Gur, Wenqing Xu

December 2019

1 Introduction

Cryptography, like various other fields in computer science, is bound to change drastically once quantum computers are introduced. Several mathematical problems that are considered "hard" in a classical scheme can be solved in polynomial time in a quantum setting. Any cryptographic algorithm that uses these primitives should be reevaluated in the quantum setting. This includes but is not limited to digital signatures. Digital signature cryptosystems are protocols designed to verify authenticity of messages and ensure that they are not altered during transfer. The introduction of quantum computers forces security assumptions of digital signatures to change, for which an analysis of alternate methods that utilize the same notion of computing is required. Quantum resistant signatures will be an essential part of secure computing, which means that a comprehensive analysis of current algorithms on the matter is crucial and almost a necessity. This paper aims to explore the range of cryptographic techniques that can securely "sign" messages in the context of quantum computers. As an analysis of the means of non-repudiation and authentication, our goal is to understand why traditional signatures are insecure under a non-classical model and discover algorithms that can be used as digital signatures which can resist attacks from a quantum computer.

2 Background

2.1 Classical Signatures Under Quantum Model

Pre-quantum signature algorithms focus primarily on prime factorizations and discrete logarithms and their extensions to elliptic curves. These problems do not have known polynomial algorithms to solve, hence they are used extensively in cryptosystems. The most efficient algorithms cannot go faster than exponential time, which makes these problems suitable bases for digital signatures as by the time the systems are broken, the information carried would be invalid. However, this is not the case in quantum setting as the both prime factorization and discrete logarithm can be solved in polynomial time, which renders the algorithms based on these problems insecure.

Integer(Prime) Factorization The integer(prime) factorization can be described as follows: Given integer n, find $n = pq$ where p and q are both prime integers. The application of the problem as a digital signature was first used by Rivest *et al.* [15] as part of their RSA cryptosystem. A simple representation of the scheme can be described as follows:

- *Key generation*: Find two large prime numbers p and q, and calculate $n = pq$. The operations are now done in $\mathbb{Z}_n$ and n is public. Next, calculate $\lambda(n)$ where λ is the totient function and $\lambda(n) = (p-1)(q-1)$ since both p and q are primes. $\lambda(n)$ is kept secret. For each user in the system, find a random public (verification) key vk such that $gcd(vk, \lambda(n) = 1$ which guarantees vk will have an inverse mod$\lambda(n)$. Then calculate the private (signing) key sk such that$sk \cong vk^{-1}$ mod $\lambda(n)$. Each user will then have their own (vk, sk) pair and vks are public.

- *Signing*: When user A wants to sign a message $m \in \mathbb{Z}_n$, A calculates a digest $H(m)$ using a public random oracle $H : \{0,1\}^* \to \mathbb{Z}_n$. A then calculates the signature $S \cong H(m)^{sk_A}$ mod n.

- *Verification*: If another user B wants to verify that the signature S indeed corresponds to the message m, user gets A's public verification key vk_A and checks the following: $H(m) \cong S^{vk_A}$ mod n. If the congruence holds true then the message is verified.

For an adversary to recover sk_A, $\lambda(n)$ must be known so that it is possible to calculate $sk_A \cong vk_A^{-1} \bmod \lambda(n)$. Since $\lambda(n)$ is private, the only way to calculate $\lambda(n)$ is to find the prime factors of n so that $(p-1)(q-1) = \lambda(n)$. Finding p and q does not have a known sub-exponential algorithm in classical setting, which makes the scheme secure. However, when moved to quantum, we can see the security does not hold anymore; as given n, p and q can be recovered using Shor's algorithm [16] in polynomial time, which reduces the problem into an order finding problem and solves it with a quantum circuit.

Discrete Logarithm The discrete logarithm problem on the other hand is defined as follows: Given n, its primitive root g and a value $a, gcd(a, m) = 1$ find x such that $g^x \cong a \bmod m$. Replacing a with 1, the problem is also known as the *order finding* problem. Application of discrete log into the digital signatures is commonly associated with Elgamal's [7] scheme: which can be simplified as the following:

- *Key Generation*: Find a prime p and a generator g for $\mathbb{Z}_p$. The parameters (p, g) are public to the system. For each user, randomly choose the signing key sk from $[1, p-1)$. Then calculate the verification key $vk = g^s k \bmod p$.

- *Signing*: When user A wants to sign a message $m \in \mathbb{Z}_p$, A calculates a digest $H(m)$ using a public random oracle $H : \{0,1\}^* \to \mathbb{Z}_n$. A then calculates the signature by first randomly choosing a $k \in [2, p-1)$ where $gcd(k, p-1) = 1$ then calculate r and s where $r \cong g^k \bmod p$ and $s \cong (H(m) - sk_A r)k^{-1} \bmod p - 1$. The pair (r, s) is the signature corresponding to m.

- *Verification*: If another user B wants to verify that the signature (r, s) indeed corresponds to the message m, user gets A's public verification key vk_A checks the following: $g^{H(m)} \cong vk_A^r r^s \bmod p$. If the congruence holds true then the message is verified.

For an adversary to recover sk_A, k must be known so that it is possible to calculate $sk_A \cong sk - H(m) \bmod (p-1)$. Since k is private, only way to calculate is through r, which requires access to an efficient way of calculating discrete logarithm. Since there is no sub exponential way with classical computers, the signature is considered secure. However, expanding to the quantum setting the same security won't hold as calculating k can be considered an instance of order finding, which can be solved by Shor's algorithm's [16]'s quantum part.

2.2 Random Oracle in Quantum Setting

One additional property of signatures in classical setting independent from the underlying problem is the use of a random oracle $H : \{0, 1\}^* \rightarrow \mathbb{Z}_n$ to use a digest of the message during signing process. The oracle H is chosen in a way that given a and $H(a)$, it is hard to find $b \neq a$ such that $H(a) = H(b)$. This is also known as the *weak collision resistance* and is the reason why H is used as part of signature schemes as for given pair of message and signature (m, S), it is hard to find m' such that $H(m) = H(m')$ which would result in the same signature S. While the best methods for finding such m' depend on birthday paradox in classical setting, this can easily be solved in quantum setting using Grover's search algorithm [10] which makes message forgery possible.

3 Secure Signatures in Quantum Setting

Secure signatures in quantum setting can be examined under two main categories: quantum and post-quantum signatures. While both of these paradigms assume the existence of a quantum adversary, the schemes themselves are based on different structures and hence different assumptions and security proofs.

3.1 Post-Quantum Signatures

Popular of the two, post-quantum signatures are based on structures that can be implemented in classical setting and assumed to withstand the presence of a quantum adversary. Their security proofs are often based on resistance against known quantum algorithms, hence they are often classified as "quantum resistant". The capability of implementing these schemes in classical computers concentrates the majority of the effort in signatures on this field where National Institute of Standard and Technology (NIST) leads the ongoing research in standardization. [5]

3.1.1 Lamport-Diffie One-Time Signature System

The Lamport-Diffie one-time signature scheme [4] makes use of a hash function H, and its security depends on H being hard to invert. The scheme goes as follows:

Let H be a hash function that generates b output bits.

- *Key Generation*: The signer independently picks at random $2b$ b-bit strings $x_1[0], x_1[1], \dots, x_b[0], x_b[1]$. These strings make up the secret key. The public key $y_1[0], y_1[1], \dots, y_b[0], y_b[1]$ also consists of $2b$ b-bit strings, and it is computed by hashing the strings of the secret key: $y_1[0] = H(x_1[0]), y_1[1] = H(x_1[1]), \dots, y_b[0] = H(x_b[0]), y_b[1] = H(x_b[1])$.

- *Signing*: The signer generates uniformly at random a b-bit string r, and computes $H(r, m)$, resulting in b output bits $h_1, \dots, h_b$. The signature is $(r, x_1[h_1], \dots, x_b[h_b])$.

- *Verification*: The verifier has the message m and the signature $(r, z_1, \dots, z_b)$ and computes $H(r, m)$, resulting in b output bits $h_1, \dots, h_b$. The verifier then checks whether $y_1[h_1] = H(z_1), \dots, y_b[h_b] = H(z_b)$. The message is accepted only when each of the equalities holds.

If an attacker A wants to deceive the verifier, A must be able to efficiently invert H, and this is infeasible even with quantum algorithms.

Note that the signature scheme described above only allows one message to be signed. In addition, this scheme requires each individual character to have its own signature, which is an inefficient method of signing. There are ways, however, of extending the scheme to allow multiple signed messages.

3.1.2 Proof-based Signatures

Quantum signatures can take many different forms, but one of the most common signature types is the non-interactive, zero-knowledge proof. A cryptosystem based on a proof system involves two entities, a Prover and Verifier. The Prover uses a message to generate some proof, then sends it to the verifier, who checks the proof and receives a verdict. This allows the sender to sign and encrypt a message, and the receiver to decrypt and ensure authenticity.

The proof can be considered non-interactive if the prover and verifier do not

send messages to each other. A proof is zero-knowledge if the prover can show the verifier their knowledge of the message without giving specific information. Thus, a non-interactive, zero-knowledge proof can be viewed as Alice sending an encrypted public message which Bob can then decrypt and ensure that it came from Alice without knowing Alice's unique signature. One example of a zero-knowledge, non-interactive proof is a Σ-protocol, which is a public-coin verifier that sends 3 messages.
The Fiat-Shamir signature scheme [12, 13] is one of the most widely studied of these types of post-quantum cryptosystems based on proof systems. Fiat-Shamir is a transform of a Σ-protocol using a random oracle.

Definition 1 (Security Parameter Relationship) *For every statistical security parameter λ, there is a set of relationships*

$$\mathcal{R}_\lambda = \{(x, w) : x \in L_\lambda, w \in W(x)\} \tag{1}$$

where L_λ is a language in NP, and $W(x)$ is a set of witnesses for proving statement x. In other words, there is a polynomial time and poly(λ) algorithm that decides if $(x, w) \in \mathcal{R}_\lambda$.

Definition 2 (Σ-Protocol)
A Σ protocol for $\mathcal{R}_\lambda$ consists of a prover $\mathcal{P}$ and a verifier $\mathcal{V}$, both polynomial time algorithms.

- *$\mathcal{P}$ is given (x, w), and outputs (a, st), where a is a commitment sent by a state st.*
- *$\mathcal{V}$ is given (x, a) generates a challenge $c \sim \{0, 1\}^\lambda$ uniformly at random.*
- *$\mathcal{P}$, given (x, w, st, c), generates a response r.*
- *$\mathcal{V}$ is given r, and using (x, a, c), determines if the proof (a, c, r) is valid.*

Definition 3 (Fiat Shamir Transformation) *The Fiat Shamir transform replaces the verifier's challenge with a hash function $c \in \mathcal{H}(a)$. Therefore, the prover algorithm $\mathcal{P}$ can generate this interaction themselves,*

As a result, we can turn a Fiat Shamir transform on any Σ-protocol into a non-interactive proof in the quantum random oracle model (QROM). By reprogramming the random oracle, we can turn a quantum prover that attacks the Fiat Shamir transform into a quantum prover that attacks the underlying Σ-protocol. To accomplish this, suppose we have a dishonest prover $\mathcal{A}$, producing a proof $\pi = (a, z)$ for a statement x. We can obtain an interactive dishonest prover for the Σ-protocol that extracts a from $\mathcal{A}$ and sends it to verifier $\mathcal{V}$. The verifier then sends back challenge c, which prover $\mathcal{A}$ can then use to reprogram the random oracle. We can make it so that the output z of $\mathcal{A}$ will be a correct reply to c with a probability not much smaller than the probability that $\mathcal{A}$ samples the proof π in the QROM.
We see that using a quantum computer, a Σ-protocol is not secure in the QROM model. This is because we can simulate the distribution of all $c \sim \{0,1\}^{\lambda}$ using a superposition of all states. But, we see that the Fiat Shamir transform can be described as quantum resistant because it is hard to simulate all outputs of a particular hash function $\mathcal{H}(a)$. Thus, the Fiat Shamir transforms a proof-based communication scheme into a post-quantum resistant cryptosystem.

3.1.3 Lattice-Based Signature Systems

As part of the post-quantum cryptography, lattice-based digital signatures include algorithms based on mathematical structures called lattices. A lattice Λ with the dimension m and rank k can be described as an additive subgroup of $\mathbb{R}_m$ whose basis consists of k linearly independent vectors. Ajtai [1] first showed that a cryptosystem based on a lattice problem has the complexity of the underlying cryptosystem, which made lattices as a viable alternative to standard public-key algorithms. With additional problems defined [2], signature schemes based on lattice problems are widely seen as the successor of the traditional digital signature schemes as there are no known quantum algorithms that can solve hard lattice problems in polynomial time and the problems can equivalently transformed into other problems defined on matrices and rings, which also allows easiness of implementation in traditional computers. This implementation easiness can also seen in NIST Standardization Effort, as a significant portion of the round 2 candidates are based on lattices. [5]. Important examples include but are not limited to CRYSTALS-Dilithium [6], Falcon [8] and qTESLA [3].

3.2 Quantum Signatures

As name suggests, quantum signatures consist of schemes that are defined on actual quantum circuits. This limits the variety of algorithms compared to post-quantum ones, however from an information theory perspective, quantum signatures are often impossible to forge or repudiate hence called "quantum secure". However, due to the setting, these schemes also come with several issues which would have been trivial problems in classical setting.

3.2.1 Gottesman - Chuang Signatures

Proposed in [9], Gottesman- Chuang signatures are considered one of the widely known quantum signatures although the scheme itself is a naive one time signature. Gottesman-Chuang signature can be seen as the extension of Lamport-Diffie signatures into the quantum setting, where instead of hashes, a construct called *quantum one-way functions* are used which also means the signature is created bit by bit. Use of one-way functions is a common occurrence in any type of digital signatures; however quantum one way functions differ from its traditional counterparts since what constitutes non-feasible operation in classical one way functions does not carry over to quantum setting. These functions are limited in terms of utility and the extent of one-way properties are different compared to classical setting. However, Gottesman-Chuang signature also contains 3 different types of verification result, which is something unusual for a classical signature (where the possible verification results are either accept or reject).

Quantum One-Way Functions For a function f, the *one-way property* can be described as for element x in the domain of f, it is easy to calculate $f(x) = y$ but computationally hard to invert the function and calculate $f^{-1}(y)$. One-way functions are extensively used in digital signatures in classical setting which makes the use of an extended definition of one-way functions for an algorithm that is the extension of classical algorithm logical. However, due to the setting, the quantum one-way function we will be discussing is limited in terms of capability and bound by number of recipients in order to preserve non-inversibility.

In essence, quantum one-way functions in this context are defined as follows: Giving a classical bit string k, generate a quantum state $|f_k\rangle$ such that $k \mapsto |f_k\rangle$. This function manipulates two different properties of quantum

systems, which helps it to act as a one-way function. Unlike in classical one, bits in quantum setting can be in superposition of 0 and 1, which means any n-qubit state has exponential number of coefficients. For all k bit strings of length L i.e. $k \in \{0,1\}^L$ and n-qubit states $|f_k\rangle$, this property allows $L >> n$ since $\langle f_k| |f'_k\rangle \leq \delta$ (almost orthogonal) for $k \neq k'$. Conversely, this helps the function not to be targeted by search algorithms.
While it is easy to calculate and verify $k \mapsto |f_k\rangle$, to preserve the irreversibility outside knowing k, additional restrictions must be met. Unlike a classical one-way function, it's possible to reverse a mapping in quantum setting, given enough copies of the same state $|f_k\rangle$. To prevent such case, the number of recipients and lengths of both classical strings and resulting quantum states must be bounded, for which the second property of quantum states is manipulated. According to Holevo's theorem [11,14], for a given n-qubit state the most number of classical bits of information that can be deduced from the state is n-bits. Given T copies of n-qubit $|f_k\rangle$, the most number of bits that can be deduced about k will become Tn. If $L - Tn >> 1$, the probability of guessing k will remain small, preserving the one-way property.
However, the auxillary requirements to calculate and verify the quantum one-way function are not as trivial as its classic counterparts. An equality test for classical strings k and k' is required, which is complicated considering the states $|f_k\rangle$ and $|f'_k\rangle$ should be preserved after this check. This can be done using the SWAP-test where starting with $|+\rangle$ as the ancilla qubit, states $|f_k\rangle$ and $|f'_k\rangle$ are put through a controlled Fredkin gate (where ancilla qubit is acting as the control) followed by a Hadamard gate and measurement on the ancilla qubit. If the measurement is $|0\rangle$ then the test is passed and $k = k'$ since this would only happen if $|f_k\rangle = |f'_k\rangle$. However, there is still a chance of failing if $|f_k\rangle \neq |f'_k\rangle$ as $|f_k\rangle$ and $|f'_k\rangle$ are not completely orthogonal.
Another task to be accomplished is given an arbitrary state $|\psi\rangle$ how can you verify that it is indeed the output of the function i.e. given k how to check if $|f_k\rangle = |\psi\rangle$. Assuming a black-box mapping for the quantum one-way function, this is done through inverting mapping. Since k is known, it is possible to treat the mapping as $|k\rangle |0^{\otimes n}\rangle \mapsto |k\rangle |f_k\rangle$. Hence to check $|\psi\rangle$, the state can be put into the reverse mapping and verified if $|0^{\otimes n}\rangle$ is measured.

Algorithm The signature scheme based on the quantum one-way function then differs from a classical counterpart by three different aspects. Unlike a traditional signature scheme, Gottesman-Chuang signatures have three veri-

fication results: 1-ACC, 0-ACC, and REJ. 1-ACC and 0-ACC both refer to the case with authentic signature, the difference being former indicates that the result can be transferrable to another party whereas the latter lacks this property. REJ on the other hand refers to the case where the signature is not authentic and cannot be verified. Second aspect is the fact that the signature security requires only to be true in high probability. Finally, as part of the quantum setting, verification keys for the signatures are quantum states rather than bit strings.
As part of the setup, all parties in communication agrees upon number of key pairs M, and security thresholds c_1 and c_2.It is also assumed that everyone in the party knows how to implement the mapping $k \mapsto |f_k\rangle$. Based on these assumptions the signature scheme is as follows:

- *Key Generation:* Alice generates classical L-bit string pairs $\{k_0^i, k_1^i\}$ for $1 \leq i \leq M$. Next, Alice calculates the states $\{f_{k_0}^i, f_{k_1}^i\}$ for each i again using the quantum one-way function. The initial classical bit-strings are Alice's signing keys whereas the quantum states are the verification key.

- *Signing:* To sign the bit b, Alice prepares $(b, k_b^1, k_b^2, ...k_b^M)$ and sends it to every party in the communication. The tuple is the signature.

- *Verification:* To verify Alice's signature, each receiving party checks the mapping $k_b^i \mapsto |f_{k_b}^i\rangle$ and records the number of incorrect mappings j. Based on j the verification result is determined:

  - $j \leq c_1 M \longrightarrow$ 1-ACC
  - $j \geq c_2 M \longrightarrow$ REJ
  - $c_1 M < j < c_2 M \longrightarrow$ 0-ACC

Like Lamport-Diffie signatures, this scheme is for one time only, meaning after each bit signing every key, used or not used, must be discarded.

Security Security argument for the signature algorithm considers two different types of attacks: Forgery and repudiation. Forgery considers the case where an outsider tries to recover Alice's classical bit strings and create non-authentic signatures. Repudiation on the other hand considers the case where Alice tries to make recipients disagree on the validity of the same signature.

Security against forgery is straightforward based on the properties of quantum one-way functions. Assuming that the malicious party has access to all T copies of the same signature, by Holevo's theorem the most number of classical bits that it can reveal about each k_b^i is Tn. This means for each key, there are $L - Tn$ bits that have to be guessed by the malicious party. Considering there are M keys, the correct number of guesses is $2^{-(L-Tn)}(2M)$, meaning with high probability it will fail. Furthermore, if it tries to change the key based on failed attempt, each recipient would see a majority of the public keys fail.

Before addressing the repudiation security, the problem of key distribution must be handled. Key distribution itself is a significant problem in classical setting as repudiation is possible if two different recipients have access to two different keys. This combined with the lack of a broadcast channel in quantum setting makes the key distribution a non-trivial problem. Like its classical counterpart, one way to handle this issue is for Alice to submit sets of public keys to a trusted third party where the trusted third party applies swap tests to verify the validity of keys sent to recipients. If the test fails Alice's repudiation attempts are uncovered. However, Alice can still cheat by submitting a symmetric state which would not be affected by this test. In this scenario, Alice cannot determine which of the recipients gets a valid key. Since M is large, it is with low probability that $j < c_1 M$ in one recipient and $j > c_2 M$ in another one making it resistant against a repudiation attempt.

Having a trusted third party may not be an option for most of the time which means for a better proof a distributed verification method, where each recipient receives multiple set of keys to verify both their copies and their peers is required. To simplify the security argument, the disagreement between only 2 recipients can be considered. For recipients Bob and Charlie, and their incorrect mappings j_B and j_C repudiation case translates to achieving $|j_B - j_C| > (c_2 - c_1)M$. To show with high probability, this cannot be the case, authors in [9] consider a global state $|\psi\rangle$ representing what Alice could have submitted for public keys as superposition of two different terms: Terms that pass the swap test but leaves recipients in agreement and terms that fail the swap test. Representing the global state as a combination of the both kept and tested keys of both recipients follows to show that the global state passing the swap test has a low probability which results Alice's chances of repudiating being unlikely.

4 Open Problems and Conclusion

There are several open problems that we could identify in both post-quantum and quantum signatures:

- *Security of Post-Quantum Signatures:* Are the quantum resistant digital signature algorithms actually secure in quantum setting? Or does the resistance of the algorithms come from the fact that there are no known ways yet? Security proofs of post-quantum signatures often use the latter, which makes the actual security questionable.

- *Security Parameters in Post-Quantum Signatures:* Are the security parameters in signature algorithms as "tight" as possible? If yes, how do they affect the practicality and applicability of the signature algorithms?

- Practicality of Quantum Signatures: Compared to their post-quantum counterparts, quantum signatures are rare and limited in current capability, which raises the question "Is it actually practical to prefer quantum signatures for their security?". The Gottesman-Chuang signature cycle is valid for one bit only and requirement of discarding each available key poses a question in terms of practicality.

Digital signatures, being one of the key concepts in cryptography, are bound to change with the proper introduction of quantum computers. The security of prevalant classical signatures does not carry over to quantum setting due to possibility of forgery utilizing quantum algorithms. As a solution, both quantum and post-quantum signatures propose different alternatives utilizing both classical and quantum primitives. However, both have their own advantages and setbacks with specific problems, preventing a full transition to quantum-secure digital signatures completely for the time being.

References

[1] Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108. ACM, 1996.

[2] Miklós Ajtai. Generating hard instances of the short basis problem. In *International Colloquium on Automata, Languages, and Programming*, pages 1–9. Springer, 1999.

[3] Erdem Alkim, Paulo SLM Barreto, Nina Bindel, Patrick Longa, and Jefferson E Ricardini. The lattice-based digital signature scheme qtesla.

[4] Daniel Bernstein, Johannes Buchmann, and Erik Dahmen. *Post-Quantum Cryptography*. Springer, 2009.

[5] Computer Security Division, Information Technology Laboratory, National Institute of Standards, Technology, and Department of Commerce. Pqc standardization process: Second round candidate announcement, Jan 2019.

[6] Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 238–268, 2018.

[7] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469–472, 1985.

[8] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. Falcon: Fast-fourier lattice-based compact signatures over ntru.

[9] Daniel Gottesman and Isaac Chuang. Quantum digital signatures. *arXiv preprint quant-ph/0105032*, 2001.

[10] Lov K Grover. A fast quantum mechanical algorithm for database search. *arXiv preprint quant-ph/9605043*, 1996.

[11] Alexander Semenovich Holevo. Bounds for the quantity of information transmitted by a quantum communication channel. *Problemy Peredachi Informatsii*, 9(3):3–11, 1973.

[12] Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of fiat-shamir signatures in the quantum random-oracle

model. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 552–586. Springer, 2018.

[13] Qipeng Liu and Mark Zhandry. Revisiting post-quantum fiat-shamir.

[14] Michael A Nielsen and Isaac L Chuang. Quantum computation and quantum information.

[15] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[16] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.