Post-Quantum Zero-Knowledge with Space-Bounded Simulation[*]

Prabhanjan Ananth[†]

UCSB

Alex B. Grilo[‡]

Sorbonne Université, CNRS, LIP6

Abstract

The traditional definition of quantum zero-knowledge stipulates that the knowledge gained by any quantum polynomial-time verifier in an interactive protocol can be simulated by a quantum polynomial-time algorithm. One drawback of this definition is that it allows the simulator to consume significantly more computational resources than the verifier. We argue that this drawback renders the existing notion of quantum zero-knowledge not viable for certain settings, especially when dealing with near-term quantum devices.

In this work, we initiate a fine-grained notion of post-quantum zero-knowledge that is more compatible with near-term quantum devices. We introduce the notion of (s, f) space-bounded quantum zero-knowledge. In this new notion, we require that an s-qubit malicious verifier can be simulated by a quantum polynomial-time algorithm that uses at most $f(s)$-qubits, for some function $f(\cdot)$, and no restriction on the amount of the classical memory consumed by either the verifier or the simulator.

We explore this notion and establish both positive and negative results:

- For verifiers with logarithmic quantum space s and (arbitrary) polynomial classical space, we show that (s, f)-space-bounded QZK, for $f(s) = 2s$, can be achieved based on the existence of post-quantum one-way functions. Moreover, our protocol runs in constant rounds.

- For verifiers with super-logarithmic quantum space s, assuming the existence of post-quantum secure one-way functions, we show that (s, f)-space-bounded QZK protocols, with fully black box simulation (classical analogue of black-box simulation) can only be achieved for languages in BQP.

1 Introduction

Zero-knowledge is a foundational notion in cryptography. Invented in the 80s by Goldwasser, Micali and Rackoff [GMR85], this notion has slowly transitioned from being a purely theoretical notion to having applications in practice. Some notable applications of zero-knowledge include secure computation [GMW87] and cryptocurrencies [SCG+14].

A zero-knowledge proof (or argument) system for a language in NP is an interactive protocol between a prover, who receives as input an instance-witness pair (x, w) and a verifier, who receives as input an instance x. The zero-knowledge property states that the verifier, after interacting

---

[†]Email: `prabhanjan@cs.ucsb.edu`

[‡]Email: `alex.bredariol-grilo@lip6.fr`

with the prover, should not gain any information about the witness beyond what is leaked by the statement. That is, in the words of the inventors of zero-knowledge [GMR85], *"... what the verifier sees in the protocol (even if he cheats) should be something which the verifier could have computed himself..."*. Formally, we define the notion of a simulator and we require that the view of the verifier after interacting with the prover should be computationally indistinguishable from the output of the simulator.

When we consider polynomial-time verifiers we typically require the simulator to run in polynomial time. In particular, the storage and the computational power utilized by the simulator can be an arbitrary polynomial, respectively, in the storage and computational power of the verifier. One might worry that this definition is not strong enough: the verifier might gain knowledge from a protocol that would need far more computational resources in order for it to have computed by itself.[1] Nonetheless, this is a reasonable assumption to make in the present day as getting access to more computational resources is cheaper than ever before.

Quantum Zero-Knowledge. The advancement in quantum technology forces us to re-think the design of cryptographic protocols. But even before we build protocols that are secure against quantum adversaries, we need to first focus on formal definitions and appropriately define post-quantum security for existing cryptographic primitives. We focus on the notion of zero-knowledge against malicious quantum verifiers, also referred to as quantum zero-knowledge. One approach to formalize this definition is to modify the definition of classical zero-knowledge as follows: instead of considering (classical) probabilistic polynomial-time adversaries in the traditional zero-knowledge definition, we instead consider quantum polynomial-time (QPT) adversaries. And instead of modeling the auxiliary input to the verifier as a classical string, we model it as a quantum state. Almost all the works [Wat09, BS20, ALP20, ABG$^+$20, ACP21, BKS21, CCY21, LMS21] take this approach of defining quantum zero-knowledge. While this is the most general definition one can think of, this definition does not accommodate the subtleties on the computational power of the quantum verifier that is highlighted by the example below.

Definitional Issues. Let us consider the following language based on integer factorization:

$$\mathcal{L}_{\mathsf{FACTORING}} = \{\langle N, L, U \rangle : \exists \text{ prime } p \text{ that divides } N, \text{ s.t.} L \leq p \leq U\}.$$

Consider the following protocol for $\mathcal{L}_{\mathsf{FACTORING}}$ between a prover P and a verifier V:

1. P sends a prime p that divides N and moreover, $L \leq p \leq U$

2. V checks if: *a)* p is prime; *b)* p divides N; and *c)* $L \leq p \leq U$. If all the checks pass, then V accepts. Otherwise, V rejects.

It is not hard to see that this protocol is complete and sound. Interestingly, according to the existing definition, this protocol satisfies the quantum zero-knowledge property. The simulator works as follows:

1. Using the Shor's algorithm for integer factorization [Sho99], find all the prime factors $p_1, ..., p_\ell$ of N.

---

[1]See the discussion on precise zero-knowledge in Section 1.3.

2. Let i be such that p_i is prime, p_i divides N and $L \leq p_i \leq U$. Send p_i to the verifier.

Notice that when describing the simulator, we did not take into consideration the computational power of the verifier. In particular, let us consider a malicious verifier V' which is a hybrid computer, consisting of a quantum device available today (such as the implementations of low-depth Random Circuit Sampling devices [Aru19], Boson sampling [Zho21], non-universal adiabatic computation [Joh11], etc.) and a classical machine. None of the currently available quantum devices have the capability to factor large prime numbers. If V' participates in the protocol described above then V' could be gaining knowledge, i.e., a non-trivial factor of N, that it could not have been able to compute by itself; despite the protocol being quantum zero-knowledge!

This discrepancy appears because the definition of quantum zero-knowledge allows the simulator to be an arbitrary quantum polynomial-time algorithm, regardless of the computational power of the verifier. For instance, even if the malicious verifier is a classical polynomial-time algorithm, the simulator is still a quantum polynomial-time algorithm.

A more realistic definition should instead consider the resources used by the verifier and require the simulator to run under (roughly) the same constraints. There are many resources we can take into consideration when formulating this definition. In this work, we focus on the resource of quantum memory. More specifically, we focus on the number of qubits utilized by the simulator in relation to the number of qubits used by the verifier.

Constructions of QZK: Prior Work. We first observe that the existing works on post-quantum zero-knowledge propose simulators whose space complexity is a large polynomial overhead in the space complexity of the verifier. One main reason behind this is the fact that all the existing black-box simulation techniques [Wat09, Unr12, CCY21, CMSZ22, CCLY21b, LMS21] first purify the verifier; that is, they convert the next message function of the verifier into a unitary[2] by delaying the measurements. The purification process increases the space complexity, proportional to the number of intermediate measurements.

Even if we ignore the above issue and consider verifiers where the next message functions are implemented as unitaries, the existing simulators still have large polynomial overheads in the space complexity of the verifier. In [Wat09], the reason for this is the fact that the space complexity of the simulator depends on the communication complexity of the protocol which in turn is some function of the security parameter. In [CCY21, CCLY21b, LMS21], the simulator runs the verifier coherently multiple times and thus, the space complexity additionally depends on the number of iterations.

1.1 Our Contributions

We propose a new definition of quantum zero-knowledge, where the amount of quantum memory used by the simulator is closely related to the quantum memory used by the verifier. We also investigate the feasibility of the new notion.

1. New Definition: Space-Bounded QZK. We formulate a new definition of post-quantum zero-knowledge, that we call (s, f)-*space-bounded* QZK, where $s \in \mathbb{N}$ and $f(\cdot)$ is some function. Suppose the malicious quantum polynomial-time (QPT) verifier is a quantum algorithm that uses at most s qubits of quantum memory. Then, we require that the simulator runs in polynomial time

---

[2]Technically, it is converted into a unitary followed by measurement, where the measurement outcome will be the message communicated to the prover.

and use the number of qubits is at most $f(s)$. For example, if f is defined to be $f(x) = 2x$ then (s, f)-space bounded QZK states that the simulator should take up at most twice the number of qubits as the verifier. On the other hand, we don't place any restriction on the classical memory of the simulator. The amount of classical storage of the simulator could be an arbitrary polynomial in the amount of classical storage of the verifier. Indeed, in reality, classical storage is much cheaper than quantum storage and thus, we need to be more precise about modeling the latter.

We consider three different notions of (s, f)-space-bounded QZK:

- *Fully Black Box*: In this notion, the simulator gets oracle access to the verifier. More precisely, each round of the verifier is modeled as a sequence of channels (one per round) and the simulator can make polynomially many queries to each of these channels. This notion of quantum zero-knowledge is a direct analogue of classical zero-knowledge.

  In terms of space complexity, the space undertaken by the simulator would also take into account the space needed to store the private state of the verifier in-between the executions of every two consecutive rounds.

  In this work, we mainly focus on understanding the feasibility of (s, f)-space bounded QZK for different $s(\cdot)$ functions.

- *Black Box*: In this notion, the simulator gets oracle access to the purification of the verifier and its inverse. More precisely, suppose the verifier is represented as a sequence of channels and let $U_1, \ldots, U_k$ be their *canonical* purifications[3]. Then, the simulator gets oracle access to U_i and also oracle access to $U_i^\dagger$, for every $i \in [k]$. Although, in this model, the verifier's code is modified before giving access to the simulator, we still call it black box in order to be consistent with most of the previous works on black box quantum zero-knowledge who adopt this model.

  Note that here, we model the quantum space complexity of the simulator as a function of the quantum space complexity of the original verifier (and not the one obtained after purification). A (canonically) purified verifier can take significantly more space than the underlying non-purified verifier. For example, purifying an s-qubit verifier with ℓ measurements will result in a unitary that consumes at least $s+\ell$ qubits. Thus, (s, f) space-bounded black-box QZK, for any polynomial $f(\cdot)$, is impossible to achieve. On the other hand, prior works [Wat09, ACP21, CCY21, CCLY21b, LMS21] are (s, f)-space bounded black box QZK for super-polynomial $f(s) = 2^{\omega(\log(s))}$[4].

- *Non Black Box*: Finally, one can consider non black box quantum zero knowledge, where the simulator is allowed to arbitrarily depend on the verifier and in particular, could make use of the code of the verifier. This definition resembles the counterpart definition of (classical) non-black box zero-knowledge. Prior works [BS20, BKS21] satisfy (s, f)-space bounded non black box QZK for super-polynomial $f(s) = 2^{\omega(log(s))}$. We leave the investigation on the feasibility of (s, f)-space bounded non black box QZK, when $f(\cdot)$ is a *polynomial*, as an interesting open problem.

---

[3]This means that the purification is computed in a specific manner: that is, by delaying the measurements of the channel.

[4]Notice that in these previous results, the number of qubits used by the simulator is polynomial but it might scale with the *time-complexity* of the verifier and that is why we can only achieve a super-polynomial upper-bound on the number of qubits.

2. Space-Bounded QZK Against Logarithmic Space Verifiers. We first focus on a restricted case case when the malicious QPT verifiers only have access to logarithmically many qubits. In this case, we demonstrate a feasibility result.

Suppose $s = O(\log(\lambda))$. We show that there exist, even constant-round, (s, f)-space-bounded quantum zero-knowledge protocols for NP, where $f(x) = 2x$, based on the assumption of post-quantum one-way functions. In fact, our protocol even achieves fully black-box quantum zero-knowledge.

3. Space-Bounded QZK Against Super-Logarithmic Space Verifiers. We then investigate the case when the malicious verifiers have access to super-logarithmically many qubits. In this case, we present a negative result.

Suppose $s = \omega(\log(\lambda))$. Assuming the existence of post-quantum one-way functions, we show that there do not exist fully black-box (s, f)-space bounded quantum zero-knowledge protocols for languages outside BQP.

Our negative result only applies to fully black-box quantum zero-knowledge and it is an interesting open problem to either prove a negative result or circumvent our negative result using non black box techniques.

1.2 Technical Overview

In Section 1.2.1 and Section 1.2.2, we present an overview of the techniques employed in both the results.

1.2.1 Zero-Knowledge Against Logarithmic Quantum Space Verifiers

Rewinding has been the quintessential technique employed in proving black-box simulation security of cryptographic protocols against probabilistic polynomial-time classical adversaries. A rewinding-based simulator has to store copies of the intermediate states of the verifier so that it can use these copies whenever it rewinds the execution of the verifier to an earlier round. However, adopting rewinding to prove post-quantum security has not been easy; this was first identified by [VDG97]. At a high level, the reason for the difficulty arises since rewinding implicitly requires the ability to store copies of the intermediate states, which the no-cloning theorem [Die82, WZ82]. Thus, most of the recent works on post-quantum zero-knowledge [Wat09, Unr13, ACP21, CCY21, CMSZ22, CCLY21a, LMS21] have proposed various rewinding techniques to perform simulation without having the need to store intermediate copies of the verifier's state.

State Recovery. To rewind the verifier to an earlier round, the most commonly adopted strategy is to invert the operations performed by the verifier in the hope that we can completely recover the earlier states. There are two issues with it:

1. Firstly, this means that the verifier has to be a unitary and thus needs to be *purified*; that is, all the intermediate measurements of the verifier needs to be pushed to the end of each round. As a consequence of the purification process, the simulator could take much larger quantum space than the verifier.

2. Secondly, each round could potentially disturb the verifier's intermediate state in an irreversible way. Thus, we might no longer be able to recover the earlier states.

5

All the recent works on black-box quantum zero-knowledge give up on the first issue. Regarding the second issue, they provide a workaround by showing that still retains some useful properties about the original intermediate state, even if the recovered state is quite different from the original intermediate state. For example, the probability that the verifier does not abort on the recovered state could be at least as much as the probability that the verifier does not abort on the original intermediate state. We will not go into the details of how state recovery is performed in each of the recent works [Wat09, CMSZ22, CCY21, LMS21, CCLY21b] since it is not relevant to our approach.

An observation. Consider an M-qubit quantum state ρ, where $M = O(\log(\lambda))$. Our observation is that a maximally mixed state $\frac{I}{2^M}$ is a good approximation of ρ. That is, suppose there exists a binary positive operator valued measure (POVM) Λ such that $p = \text{Tr}(\Lambda(\rho))$ then it holds that $p = \text{Tr}(\Lambda(\frac{I}{2^M})) \geq \frac{p}{2^M}$. This suggests a new approach to perform rewinding: to rewind the verifier to an earlier round, say i, simply execute the ith round of the verifier on input $\frac{I}{2^M}$. The advantage of this approach is two-fold. Firstly, there is no need to purify the verifier! Secondly, we only need M additional qubits of quantum storage to initialize the maximally mixed state.

Case Study: Four Round Protocol due to Goldreich-Kahan [GK96]. Let us try to use this idea to prove post-quantum security of a zero-knowledge protocol. We will start with the four-round protocol due to Goldreich and Kahan [GK96]. Denote the four rounds in the protocol to be $(\alpha, \beta, \gamma, \delta)$.

Simulating Classical Verifiers: We first recall the strategy to simulate classical adversaries. The classical simulator, running in expected polynomial time, receives as input α, which is nothing but the commitment of verifier's challenges. Let the verifier's state at this point be st.

- *First Step. Execution of Main Thread*: The simulator then sends β, which is nothing but the commitment of 0. There are two things that can happen. Either the verifier aborts, in which case, even the simulator aborts. Or it could happen that the verifier does not abort. In this case, the simulator moves on to the second step.

- *Second Step. Execution of Lookahead Threads*: At this point, the simulator has the opening of α, denoted by γ. The simulator then rewinds the verifier to st, i.e., the state where it just sent α. It uses the information from the opening γ, to compute the commitment β^*. It sends β^* to the verifier. If the verifier sends γ^* then the simulator completes the transcript $(\alpha, \beta^*, \gamma^*, \delta^*)$, and additionally outputs the verifier's final state. If the verifier aborts then the simulator keeps repeating the rewinding process until it wins.

The important thing to remember is that the simulator first runs the main thread to decide whether to continue or not. If it continues, it outputs one of the lookahead threads that did not abort.

Simulating Quantum Verifiers: Suppose we need to simulate quantum verifiers. We extend the above approach to the quantum setting as follows. The quantum simulator receives as input α, which is nothing but the commitment of verifier's challenges. Let the verifier's state at this point be (ρ, st), where ρ is a M-qubit quantum state and st is a classical string.

- The first step is the same as above. The simulator executes the verifier on input the state (ρ, st). Let the resulting state of the verifier be (σ, st').

- If the verifier has not aborted at this point, it moves to the second step. At this point, if the simulator has to rewind, it no longer has a copy of ρ since it was computed upon in the first round. Thus, the simulator uses the maximally mixed state $\frac{I}{2^M}$ in place of ρ whenever it rewinds.

As per our observation earlier, $\frac{I}{2^M}$ serves as a good approximation of ρ. In particular, if p is the probability that the verifier on input ρ outputs a valid third message then the verifier on input $\frac{I}{2^M}$ outputs a valid third message with probability at least $\frac{p}{2^M}$. It can be argued that, as long as M is logarithmic, the simulator in expected polynomial time can recover a valid transcript $(\alpha, \beta^*, \gamma^*, \delta^*)$ from one of the lookahead threads just like the classical simulator. However, this is not sufficient. The quantum simulator should not also output the residual state of the verifier along with the transcript $(\alpha, \beta^*, \gamma^*, \delta^*)$. But note that the residual state of the verifier obtained in the lookahead thread is useless: it is the state obtained by replacing the intermediate state ρ with a maximally mixed state. One option is to output (σ, st'), which is the state obtained in the main thread, along with $(\alpha, \beta^*, \gamma^*, \delta^*)$. However, the state (σ, st') is obtained in the main thread and hence, is inconsistent with the lookahead thread transcript $(\alpha, \beta^*, \gamma^*, \delta^*)$. Thus, we need a different approach where we can simulate in such a way that the joint distribution of the transcript and the verifier's state is computationally indistinguishable from the real world.

Protocol Template. We propose a different protocol and prove their post-quantum security using our key observation. As a starting point, we consider a commitment scheme. For the current discussion, we will consider commitments that are non-interactive and satisfy perfect binding and computationally hiding property. In the technical sections, we relax the requirement to allow the commitment to be interactive and moreover, we only require statistical binding.

We present a simplified version of the protocol template first. Let $\mathcal{L}$ be the NP language associated with the protocol. We denote the prover to be P and the verifier to be V. The prover receives as input instance-witness pair (x, w) and the verifier receives as input x.

1. V sends two commitments of α_0 and α_1.

2. P sends a bit b.

3. V sends the opening of commitment of α_b.

4. P uses a witness-indistinguishable protocol to prove to V that either $x \in \mathcal{L}$ or it knows $\alpha_0 \oplus \alpha_1$.

The above protocol is sound since a malicious prover can only receive one of the two decommitments and thus, will not know $\alpha_0 \oplus \alpha_1$ in order to complete the WI phase. To prove that the above template satisfies post-quantum zero-knowledge, we first need to demonstrate a simulator.

- The simulator receives as input commitments of α_0 and α_1 from the malicious verifier. Let (ρ, st) be the state of the verifier at this point, where ρ is the M-qubit state and st is the classical state.

- *First Step. Execution of Main Thread*: The simulator sends a random bit b. It receives an opening of the commitment of α_b from the verifier. Let the state of the verifier at this point to be (σ, st').

7

- *Second Step. Execution of Lookahead Threads*: The simulator then rewinds the verifier to just after the first round. This means that the simulator restarts the verifier on the state $\left(\frac{I}{2^M}, \mathsf{st}\right)$. The simulators sends a random bit b' to the verifier and it hopes that $b' \neq b$ and the verifier does not abort. The simulator keeps repeating this process until it succeeds.

Once the simulator succeeds, it has both α_0, α_1. It discards all the lookahead transcripts and only retains the main thread transcript. Continue the execution of the protocol with the verifier, where the state of the verifier is (σ, st'). The simulator then convinces the verifier in the WI phase using $\alpha_0 \oplus \alpha_1$.

Note that unlike Goldreich-Kahan, the simulator retains the main thread transcript along with the verifier's residual state, just after the rewinding process, whose joint distribution is computationally indistinguishable from the real world.

While the main approach is sound, implementing the above approach encounters some issues. We mention the issues and how we fix them. The first issue is that it could happen that the verifier doesn't abort with non-negligible probability and also, that the simulator never succeeds when the verifier does not abort. Indeed, the verifier could decide that it will not abort if and only if the bit b sent by the prover is 0. In this case, the simulator will never be able to recover two valid transcripts if the verifier does not abort. We fix this by increasing the challenge space. Instead of requiring the verifier to send two commitments, it sends 2λ commitments of messages $((\alpha_{1,0}, \alpha_{1,1}), \ldots, (\alpha_{\lambda,0}, \alpha_{\lambda,1}))$ and in the third message, it opens commitments of messages $(\alpha_{1,b_1}, \ldots, \alpha_{\lambda,b_\lambda})$, where $b_1, \ldots, b_\lambda$ are the challenge bits sent by the prover in the second message. If the verifier does not abort with non-negligible probability then there exists an index i such that the verifier opens to each value $\alpha_{i,0}, \alpha_{i,1}$ with non-negligible probability. The second issue is on non-malleability. The prover could launch a malleability attack by leveraging the commitments sent by the verifier in the first message to cheat the verifier in the WI phase. A first attempt is to make the prover commit to the witness to be used in the WI phase. But even this is susceptible to the malleability attack. We use a technique explored in a recent work [ALP20], where the prover commits to some randomness $\mathbf{r}^*$ at the very beginning even before it sees any message from the verifier. At a later point in time, when the prover commits to the witness, it is expected to use this randomness $\mathbf{r}^*$. This prevents the mauling attack since if the prover computed its own commitment by mauling the commitment by the verifier, it would not have the knowledge of the randomness contained in its commitment. Once we incorporate these fixes, we have a complete description of our protocol.

1.2.2 Zero-Knowledge Against Super-Logarithmic Quantum Space Verifiers

In Section 1.2.1, we restricted our attention to zero-knowledge against logarithmic quantum space verifiers. We now justify the restriction on the quantum space of the verifiers by showing the following: assuming post-quantum one-way functions, only for languages in BQP, there exist protocols that are space-bounded fully black-box QZK against super-logarithmic quantum space verifiers.

This is shown by first proving that any fully black-box QZK protocol against super-logarithmic quantum space verifiers should have a straight-line simulator. We design a contrived verifier V' that is composed of channels $\Phi_1, \ldots, \Phi_k$, where k is the number of rounds in the protocol and Φ_i is used to compute the ith round query and suppose $M = \omega(\log(\lambda))$. We argue that any simulator simulating this verifier should be of a specific form: it should first make a successful query to Φ_1, then make a successful query to Φ_2 and so on. For any i, a query made by the simulator to Φ_i

is deemed successful[5] if Φ_i does not abort on that query. Note that a straightline simulator could make many unsuccessful queries to any Φ_j (i.e., queries leading to abort answers) in between two successful queries.

Once we prove that the simulator is straightline, we then show that the language $\mathcal{L}$ associated with the protocol has to be in BQP. This step is relatively more standard and follows similar ideas employed in prior works [GO94]. We use the straightline simulator to come up with a malicious prover $\widetilde{P}$ who is able to convince the verifier to accept an instance in $\mathcal{L}$ with probability close to 1; for now, assume that the protocol satisfies perfect completeness. At the same time, when given a NO instance, due to the soundness property, $\widetilde{P}$ can convince the verifier to accept only with negligible probability. Thus, using $\widetilde{P}$ and the honest verifier, we can construct a quantum polynomial time algorithm that decides $\mathcal{L}$.

Forcing Simulator to be Straightline. All that is left is to show that the simulator has to be straightline with respect to the contrived verifier V' that we will design below. Since the simulator has complete control over the intermediate states of the verifier in-between different rounds of the protocol, it seems challenging to prevent the simulator from making multiple successful queries to the same channel Φ_i, for some i, where $(\Phi_1, \ldots, \Phi_k)$ are as defined above. We leverage the principles of quantum information to our advantage. Specifically, we use *subspace states* [AC13], as intermediate private states of the verifier. Subspace states are known to be unclonable and have been instrumental in constructing public-key quantum money [AC13, Zha21].

In more detail, V' is composed of the channels $(\Phi_1, \ldots, \Phi_n)$, where Φ_i works as follows:

- It takes as input a state ρ_i and the ith round message from the prover.

- It checks if ρ_i is indeed the state $|S_{i-1}\rangle\langle S_{i-1}|$, where S_{i-1} is a random $\frac{M}{2}$-dimensional subspace of $\mathbb{Z}_2^M$ and $|S_{i-1}\rangle = \sum_{x \in S_{i-1}} \sqrt{2^{-|S_{i-1}|}} |x\rangle$. The description of the subspace S_{i-1} is hardwired in Φ_i.

- If the check fails, Φ_i aborts.

- Otherwise, V' computes the next message using the honest verifier and outputs this message along with another subspace state $|S_i\rangle$, where S_i is a random $\frac{M}{2}$-dimensional subspace of $\mathbb{Z}_2^M$. The description of S_i is hardwired in Φ_i (and Φ_{i+1}).

Let us look at the different possible queries a simulator can submit to $(\Phi_1, \ldots, \Phi_k)$.

Out-of-order queries: If the simulator tries to query V' out-of-order, for example, submit a query to Φ_{i+1} before querying Φ_i, it would not have the valid subspace state $|S_i\rangle$ and thus, Φ_{i+1} will most likely abort.

Repeated queries: If the simulator queries the same Φ_i twice then we claim that there can only be at most one query that passes the subspace state test. If the simulator makes two successful queries to Φ_i then we argue that it must have two copies of $|S_{i-1}\rangle$, thus violating the unclonability property of subspace states. We first argue this for $i = 1$: this follows from the fact that the simulator has only one copy of $|S_0\rangle$. Assuming that at most only successful query can made to all Φ_j, for $j < i$,

---

[5]In the technical sections, we call such queries non-abort queries (Definition 5.3).

we can then invoke the unclonability of $|S_{i-1}\rangle$ to argue that there can be at most one successful query to Φ_i.

One has to be careful when violating the unclonability property: the simulator can make many unsuccessful queries before making the two successful queries to Φ_i. The reduction that violates the hardness of subspace states needs to be able to discern whether the adversary has submitted a successful query or not. Moreover, the reduction only receives as input a single copy of $|S_{i-1}\rangle$ and in particular, does not receive as input a description of S_{i-1}. Thus, the reduction does not have the capability to distinguish successful versus unsuccessful queries. We overcome this issue by giving the reduction access to a oracle that tests if a given state is $|S_{i-1}\rangle$. It was shown by [AC13] that as long as the subspace has super-logarithmic dimension, unclonability still holds even if the reduction gets access to the membership oracle.

Missing queries: There might exist some i such that the simulator never submits a successful query to Φ_i. By using similar arguments as in the previous cases, we can then argue that it cannot submit a successful query to the last round, i.e., Φ_k. From this, we can come up with a distinguisher that can distinguish the real world and the ideal world.

From the above cases, we can argue that a successful simulator can only submit successful queries *in-order* and moreover, can only submit exactly one successful query for Φ_i, for every $i \in [k]$.

However, there are a couple of issues we need to take into consideration while designing the straight-line simulator to ensure that $\tilde{P}$ is successful. Firstly, we need to modify V' so that it encrypts its internal state with a symmetric-key encryption scheme, whose secret key is also hardcoded in all Φ_i. Otherwise, the straightline simulator would expect the internal state in the clear and $\tilde{P}$ would be unable to provide this information to the simulator. Secondly, we need that the internal state chosen by the simulator on query Φ_i to be exactly the same as the internal state after the query to Φ_{i-1}. To solve this issue, V' uses a digital signature scheme to sign its classical internal state. This forces the simulator to use the internal state chosen by the simulator on query Φ_i to be the internal state after the query to Φ_{i-1}, otherwise we could violate the unforgeability of the digital signature scheme.

1.3 Related Work

Knowledge Tightness and Precise Zero-Knowledge. Analogous questions have been explored in the classical literature. However, the main focus has been on designing zero-knowledge simulators whose runtime is closely related to the runtime of the verifiers.

Goldreich, Micali and Widgerson [GMW86] (also, [Gol01]) explore the notion of knowledge tightness which is the ratio of the running time of the simulator and the running time of the verifier. Some existing protocols [GMW87, GMR85, Blu86] already achieve constant knowledge tightness. Along the lines of knowledge tightness, Pass and Micali [MP06] formalized a notion called precise zero-knowledge where the simulator's runtime is closely related to the runtime of the verifier. Many followup works [PPS+08, CPT12, DG12] study precise zero-knowledge in different settings.

Of relevance is the work by Ning and Du [DG11] who consider a stronger notion of precise zero-knowledge, where the simulator's space complexity also needs to be closely related to the space complexity of the verifier. However, they study this notion only in the context of zero-knowledge against classical verifiers.

Space-bounded verifiers. Another related model considers proof systems where the verifier, even the honest one, has bounded space. Some works [AF93, SPY92] have proposed feasibility results in this model. We remind the reader that in our proof system, we don't limit the amount of classical space a malicious QPT verifier can have.

Post-Quantum Zero-Knowledge. Watrous [Wat09] presented the first construction of zero-knowledge for NP against quantum verifiers. In the past few years, we have seen remarkable progress in understanding the feasibility of QZK. Bitansky and Shmueli [BS20] presented the first construction of quantum zero-knowledge in constant rounds. They prove that their construction satisfies QZK in the non-black box simulation setting. In fact, it was shown by Chia, Chung, Liu and Yamakawa [CCLY21b] that constant round black box quantum zero-knowledge is impossible. This negative result was circumvented by a recent work by Lombardi, Ma and Spooner [LMS21] who achieved black-box constant round QZK in the coherent simulation setting. Relaxations of QZK systems have been considered in some recent works [ALP20, CCY21]. Post-quantum zero-knowledge under composition have been studied [JKMR06, ABG+20, ACP21]. Finally, zero-knowledge for quantum complexity classes such as QMA have also been studied [BJSW16, BG19].

1.4 Future Directions

In this section, we discuss future directions regarding fine-grained notions of quantum zero-knowledge.

Non-black-box quantum space-bounded simulation with superlogarithmic qubits. Our result in Section 5 shows that fully black-box $(\omega(\log \lambda), \cdot)$-space-bounded quantum zero-knowledge is impossible, and as we discussed in Section 3.3, (standard) black-box $(\cdot, p)$-space bounded quantum zero-knowledge is impossible for any polynomial p. We leave as an open problem to explore the feasibility of non-black black-box quantum $(\omega(\log \lambda), p)$-space-bounded simulation for some polynomial p.

New definitions of "NISQ"-safe zero-knowledge protocols. In this work, we initiated the study of classical zero-knowledge protocols, whose security is still guaranteed against *weaker* quantum adversaries.[6] We focus here on the number of qubits allowed in the simulation and we leave as an open question the proposal of other meaningful definitions such as verifiers/simulators that use a fixed non-universal gateset or even a fixed architecture.

Space-bounded/fully black box simulation. Another interesting direction is to explore the new notions of simulation that we introduce in this work in the context of other cryptographic protocols. Specifically, we could explore the feasibility of fully black-box simulation and space-bounded simulation in both classical protocols and also protocols that use quantum resources such as quantum oblivious transfer and quantum secure computation protocols [HSS11, ABG+20, GLSV21, BCKM21].

---

[6]As we discussed, since the adversaries are weaker, we can morally break the zero-knowledge property that have much more power than the verifier.

2 Preliminaries

We denote the security parameter by λ. We assume basic familiarity of cryptographic concepts.

We denote (classical) computational indistiguishability of two distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ by $\mathcal{D}_0 \approx_{c,\varepsilon} \mathcal{D}_1$. In the case when ε is negligible, we drop ε from this notation. We denote the process of an algorithm A being executed on input a sample from a distribution $\mathcal{D}$ by the notation $A(\mathcal{D})$.

2.1 Quantum Preliminaries

Let $\mathcal{H}$ be any finite Hilbert space, and let $L(\mathcal{H}) := \{\mathcal{E} : \mathcal{H} \to \mathcal{H}\}$ be the set of all linear operators from $\mathcal{H}$ to itself (or endomorphism). Quantum states over $\mathcal{H}$ are the positive semidefinite operators in $L(\mathcal{H})$ that have unit trace.

A state over $\mathcal{H} = \mathbb{C}^2$ is called a qubit. For any $n \in \mathbb{N}$, we refer to the quantum states over $\mathcal{H} = (\mathbb{C}^2)^{\otimes n}$ as n-qubit quantum states. To perform a standard basis measurement on a qubit means projecting the qubit into $\{|0\rangle, |1\rangle\}$. A quantum register is a collection of qubits. A classical register is a quantum register that is only able to store qubits in the computational basis.

Quantum Circuits. A unitary quantum circuit is a sequence of unitary operations (unitary gates) acting on a fixed number of qubits. Measurements in the standard basis can be performed at the end of the unitary circuit. A (generalized) quantum circuit is a unitary quantum circuit with 2 additional operations: (1) a gate that adds an ancilla qubit to the system, and (2) a gate that discards (trace-out) a qubit from the system. A non-uniform quantum polynomial-time algorithm (QPT) C consists of a family $\{(C_n, \rho_n)\}_{n \in \mathbb{N}}$, where C_n is an n-input qubit generalized quantum circuit of size $p(n)$ for some polynomial $p(\cdot)$, and ρ_n is a density matrix assigned to a subset of input qubits of C_n. Unless explicitly specified, all the algorithms considered in this work are non-uniform algorithms.

Later, in Section 3.2, we present a more rigorous definition of quantum circuits that carefully describes the interaction between the classical and the quantum memory.

Oracle Access. A QPT algorithm A has oracle access to a circuit Φ, denoted by A^Φ, if the computation proceeds as follows. On input a state ρ, defined on two registers $\mathbf{X}$ and $\mathbf{Y}$, A^Φ first applies a circuit on ρ, then applies Φ on the $\mathbf{Y}$ register, followed by applying another circuit on the result, followed by applying Φ again on the $\mathbf{Y}$ register and so on.

2.2 Computational Indistinguishability

The following definition is due to [Wat09].

Definition 2.1 (Computational Indistinguishability of Quantum States). *Let I be an infinite subset $I \subset \{0,1\}^*$, let $p : \mathbb{N} \to \mathbb{N}$ be a polynomially bounded function, and let ρ_x and σ_x be $p(|x|)$-qubit states. We say that $\{\rho_x\}_{x \in I}$ and $\{\sigma_x\}_{x \in I}$ are **quantum computationally indistinguishable collections of quantum states** if for every QPT $\mathcal{E}$ that outputs a single bit, any polynomially bounded $q : \mathbb{N} \to \mathbb{N}$, and any auxiliary collection of $q(|x|)$-qubits states $\{\nu_x\}_{x \in I}$, and for all (but finitely many) $x \in I$, we have that*

$$|\Pr[\mathcal{E}(\rho_x \otimes \nu_x) = 1] - \Pr[\mathcal{E}(\sigma_x \otimes \nu_x) = 1]| \leq \epsilon(|x|)$$

for some negligible function $\epsilon : \mathbb{N} \to [0,1]$. We use the following notation

$$\rho_x \approx_{\mathcal{Q},\epsilon} \sigma_x$$

and we ignore the ϵ when it is understood that it is a negligible function.

2.3 Interactive Protocols

Languages and Relations. A language $\mathcal{L}$ is a subset of $\{0,1\}^*$. A relation $\mathcal{R}$ is a subset of $\{0,1\}^* \times \{0,1\}^*$. We use the following notation:

- Suppose $\mathcal{R}$ is a relation. We define $\mathcal{R}$ to be *efficiently decidable* if there exists an algorithm A and fixed polynomial p such that $(x,w) \in \mathcal{R}$ if and only if $A(x,w) = 1$ and the running time of A is upper bounded by $p(|x|, |w|)$.

- Suppose $\mathcal{R}$ is an efficiently decidable relation. We say that $\mathcal{R}$ is a NP relation if $\mathcal{L}(\mathcal{R})$ is a NP language, where $\mathcal{L}(\mathcal{R})$ is defined as follows: $x \in \mathcal{L}(R)$ if and only if there exists w such that $(x,w) \in \mathcal{R}$ and $|w| \le p(|x|)$ for some fixed polynomial p.

Interactive Models. We consider (classical) interactive protocols between two parties, a prover P and a verifier V. We only consider interactive protocols corresponding to NP relations. An interactive protocol for an NP relation $\mathcal{R}$ has the following format: a probabilistic polynomial-time prover P takes as input an NP instance x and a witness w while a probabilistic polynomial time verifier V takes as input x. Both the parties exchange some messages with each other. At the end of the protocol, the verifier V outputs either Accept or Reject.

Notation. We use the following notation in the rest of the paper.

- $\langle P, V \rangle$ denotes the interactive protocol between P and V. We denote the $\langle P(y_1), V(y_2) \rangle$ to be (z_1, z_2), where z_1 is the prover's output and z_2 is the verifier's output. Sometimes we omit the prover's output and write this as $z \leftarrow \langle P(y_1), V(y_2) \rangle$ to indicate the output of the verifier to be z.

- $\mathsf{View}_V(\langle P(y_1), V(y_2) \rangle)$ denotes the view of the V in the protocol Π, where y_1 is the input of P and y_2 is the input of V. The view includes the output of V and the transcript of the protocol.

2.3.1 Proof and Argument Systems

We start by recalling the definitions of the completeness and soundness properties of a classical interactive proof system.

Definition 2.2 (Proof System). *Let Π be an interactive protocol between a classical PPT prover P and a classical PPT verifier V. Let $\mathcal{R}(\mathcal{L})$ be the NP relation associated with Π.*

Π is said to be a proof system if it satisfies the completeness and the soundness properties defined below.

- **Completeness**: *For every* $(x, w) \in \mathcal{R}(\mathcal{L})$,

$$\Pr[\mathsf{Accept} \leftarrow \langle P(x,w), V(x) \rangle] \geq 1 - \mathsf{negl}(|x|),$$

for some negligible function negl.

- **Soundness**: *For every prover* P^* *(possibly computationally unbounded), every* $x \notin \mathcal{R}(\mathcal{L})$, *every state* ρ,

$$\Pr\left[\mathsf{Accept} \leftarrow \langle P^*(x,\rho), V(x) \rangle\right] \leq \mathsf{negl}(|x|),$$

for some negligible function negl.

We also consider the notion of argument systems where the prover is restricted to be a QPT algorithm.

Definition 2.3 (Argument System). *Let* Π *be an interactive protocol between a classical PPT prover* P *and a classical PPT verifier* V. *Let* $\mathcal{R}(\mathcal{L})$ *be the NP relation associated with* Π.

Π *is said to be an argument system if it satisfies completeness (as defined in Definition 2.2) and computational soundness (defined below).*

- **Computational Soundness**: *For every QPT prover* P^*, *every* $x \notin \mathcal{R}(\mathcal{L})$, *for every* $\mathrm{poly}(|x|)$-*qubit state* ρ,

$$\Pr\left[\mathsf{Accept} \leftarrow \langle P^*(x,\rho), V(x) \rangle\right] \leq \mathsf{negl}(|x|),$$

for some negligible function negl.

2.3.2 Quantum Witness-Indistinguishable Proofs for NP

For our construction, we use a proof system that satisfies a property called quantum witness indistinguishability. We recall this notion below.

Definition 2.4 (Quantum Witness-Indistinguishability). *An interactive protocol between a (classical) PPT prover* P *and a (classical) PPT verifier* V *for a language* $L \in \mathsf{NP}$ *is said to be a **quantum witness-indistinguishable proof system** if in addition to completeness, unconditional soundness, the following holds:*

- **Quantum Witness-Indistinguishability**: *For every QPT verifier* V^*, *for every* $x \in \mathcal{L}$, w_1, w_2 *such that* $(x, w_1) \in \mathcal{R}(\mathcal{L})$ *and* $(x, w_2) \in \mathcal{R}(\mathcal{L})$, *with* $\mathrm{poly}(|x|)$-*qubit advice* ρ, *the following holds:*

$$\left\{ \mathsf{View}_{\mathsf{Verify}^*}\left(\langle P(x, w_1), V^*(x, \rho) \rangle\right) \right\} \approx_{\mathcal{Q}} \left\{ \mathsf{View}_{\mathsf{Verify}^*}\left(\langle P(x, w_2), V^*(x, \rho) \rangle\right) \right\}$$

Instantiation. By suitably instantiating the constant round WI argument system of Blum [Blu86] with statistically binding commitments (which in turn can be based on post-quantum one-way functions [Nao91]), we achieve a 4 round quantum WI proof system for NP. Moreover, this proof system is a public-coin proof system; that is, the verifier's messages are sampled uniformly at random.

2.4 Statistically Binding and Quantum-Concealing Commitments

A bit commitment protocol is a two-party protocol defined between two parties, a committer (Comm) and a receiver ($\mathsf{Receiver}$).

The protocol consists of two phases:

- Commit phase: in this phase, Comm commits to a bit b and $\mathsf{Receiver}$ does not receive any input.
  In this work, we only consider commitments with two-message commit phases.

- Opening phase: in this phase, Comm reveals b and any relevant randomness used in the commit phase. $\mathsf{Receiver}$ then outputs accept or reject.
  In this work, we consider only canonical opening phases. That is, they only consist of a single message. Moreover, the opening consists of the revealed bit and all the random bits used by Comm.

As a result of considering canonical opening phases, we have the following property, referred to as *message recovery* property: given just the randomness used by the committer, the receiver can recover the bit used in the commit phase.

Properties. We consider a two-message commitment scheme that satisfies the following two properties.

Definition 2.5 (Statistically Binding). *A two-message commitment scheme between a committer (*Comm*) and a receiver (*$\mathsf{Receiver}$*), both running in probabilistic polynomial time, is said to satisfy statistical binding property if the following holds for any adversary $\mathcal{A}$:*

$$\Pr\left[\begin{array}{c} (\mathbf{c}, r_1, x_1, r_2, x_2) \leftarrow \mathcal{A} \\ and \\ \mathsf{Comm}(1^\lambda, \mathbf{r}, x_1; r_1) = \mathsf{Comm}(1^\lambda, \mathbf{r}, x_2; r_2) = \mathbf{c} : \mathbf{r} \leftarrow \mathsf{Receiver}(1^\lambda) \\ and \\ x_1 \neq x_2 \end{array}\right] \leq \mathsf{negl}(\lambda),$$

for some negligible function negl.

Definition 2.6 (Quantum-Concealing). *A commitment scheme Comm is said to be quantum concealing if the following holds. Suppose $\mathcal{A}$ be a non-uniform QPT algorithm and let $\mathbf{r}$ be the message generated by $\mathcal{A}(1^\lambda)$. We require that $\mathcal{A}$ cannot distinguish the two distributions, $\{\mathsf{Comm}(1^\lambda, \mathbf{r}, x_1)\}$ and $\{\mathsf{Comm}(1^\lambda, \mathbf{r}, x_2)\}$, for any two inputs $x_1 \in \{0, 1\}, x_2 \in \{0, 1\}$.*

Instantiation. We can instantiate statistically binding and quantum-concealing commitments from post-quantum one-way functions [Nao91].

Generalizations. We can extend the above definition to commitments of long messages and not just bits. We consider a specific instantiation of commitments of long messages. To commit to a long message, the committer commits to each bit separately. Note that even this instantiation satisfies the message recovery property.

2.5   Query Lower Bounds for Cloning Subset States

In the impossibility result (Section 5), we use the following theorem stated and proved by Aaronson and Christiano [AC13] in the context of public-key quantum money schemes.

Subspace states.   For a subspace $A \subseteq \mathbb{F}_2^n$, for some $n \in \mathbb{N}$, we denote $|A\rangle = \sum_{x \in A} \frac{1}{\sqrt{2^{|A|}}} |x\rangle$. We also denote U_A to be the following unitary. It maps an $n + 1$-qubit state $|A\rangle |y\rangle$ into $|A\rangle |y \oplus 1\rangle$, and for every $|\psi\rangle$ orthogonal to $|A\rangle$, it leaves $|\psi\rangle |y\rangle$ unchanged. Notice that U_A can be efficiently implemented with the description of A.

Space complexity of testing subspace states.   In Section 5, we will need to implement the following projective measurement $\Pi = \{ \underbrace{|A\rangle\langle A|}_{\text{outcome}=0} , \underbrace{I - |A\rangle\langle A|}_{\text{outcome}=1} \}$, where $A \subseteq \mathbb{F}_2^n$ is a subspace of dimension k, with low quantum memory complexity.

We will demonstrate an efficient implementation of Π that takes space $O(n)$ qubits. In order to implement Π, we first describe how to implement an unitary C_A that maps $|0^k\rangle |0^n\rangle \mapsto |0^k\rangle |A\rangle$, given some basis $v_1, \ldots, v_k$ of the subspace A. We denote the first k qubits to be the register $\mathbf{X}$ and the next n qubits to be the register $\mathbf{Y}$. Notice that we can assume without loss of generality, we can assume that v_i and v_j are orthogonal for $i \neq j$.

C_A can be implemented as follows:

1. Apply $H^{\otimes k}$ to the first register $\mathbf{X}$,

2. For each $i = 1, \ldots, k$:

    2.1. On the i-th qubit of $\mathbf{X}$ and i-th qubit of $\mathbf{Y}$, perform the following operation: $|b\rangle |x\rangle \mapsto |b\rangle |b \cdot v_i \oplus x\rangle$

3. For each $i = 1, \ldots, k$:

    3.1. With the i-th qubit of $\mathbf{X}$ and i-th qubit of $\mathbf{Y}$, perform the following operation: $|b\rangle |x\rangle \mapsto |b \oplus \langle x, v_i\rangle\rangle |x\rangle$, where $\langle y, z\rangle$ is the inner product of $y, z \in \mathbb{F}_2^n$.

After step 2, we have the state of the form $\sum_{\mathbf{b} \in \{0,1\}^k} 2^{-k/2} |\mathbf{b}\rangle |\oplus_{1 \leq i \leq k} \mathbf{b}_i v_i\rangle$, and that step 3 uncomputes the first register $\mathbf{X}$ (and here we use the crucial fact that the basis is orthogonal).

In order to finally perform the projective measurement $\{ \underbrace{|A\rangle\langle A|}_{\text{outcome}=0} , \underbrace{I - |A\rangle\langle A|}_{\text{outcome}=1} \}$ on some state ρ, we can then simply prepend the state $|0^k\rangle$ to it, perform the unitary $C_A^\dagger$, trace out $\mathbf{X}$ and then measure the last register in the computational basis. If we get the value 0^n then the outcome of Π is set to be 0, otherwise it is set to be 1.

To see, why this is the correct implementation, notice that $\text{Tr}_{\mathbf{X}} \left(C_A |0^k\rangle |0^n\rangle \langle 0^k| \langle 0^n| C_A^\dagger \right) = \text{Tr}_{\mathbf{X}} (|0^k\rangle\langle 0^k| \otimes |A\rangle\langle A|) = |A\rangle\langle A|$.

We notice that to implement this projective measurement, we need only k extra qubits. Thus, the total memory complexity needed to implement Π is $n + k$ qubits.

Testing subspace states. Throughout this work, we will consider the test $\mathsf{Test}^{U_A}(\rho)$ to check if ρ is a subspace state $|A\rangle$ when we have oracle access to U_A:

1. Query the oracle U_A on input $\rho \otimes |0\rangle\langle 0|$.

2. Measure the last qubit, with output b. Let σ_b be the reduced state on the first register after the measurement.

3. Return $|b\rangle\langle b| \otimes \sigma_b$.

Whenever the first bit is $|1\rangle\langle 1|$, we say that the test passes, otherwise we say that the test fails.

We have the following lemma regarding the resulting state of the verification test.

Lemma 2.7. *The state σ_1 corresponding to the quantum state returned when $\mathsf{Test}^{U_A}(\rho)$ passes is equal to $|A\rangle\langle A|$.*

The proof of this lemma follows directly from the definition of U_A.

Unclonability of subspace states. We state now the result from [AC13] regarding the unclonability of subspace states. For that, let $\mathcal{S}$ be the set of all subspaces of $\mathbb{F}_2^n$ of dimension $\frac{n}{2}$.

Lemma 2.8 (Theorem 25, [AC13]). *Let $A \subseteq \mathbb{F}_2^n$ be a subspace sampled uniformly at random from $\mathcal{S}$. Then given $|A\rangle$, as well as oracle access to U_A, any quantum algorithm C needs $\Omega(\sqrt{\varepsilon}2^{n/4})$ queries to prepare a $2n$-qubit state ρ that projects into $|A\rangle\langle A|^{\otimes 2}$ with probability (over choice of A, C and the projection onto $|A\rangle\langle A|^{\otimes 2}$) at least ε, for all $\varepsilon = \omega(2^{-n/2})$.*

More formally, C needs $\Omega(\sqrt{\varepsilon}2^{n/4})$ queries to prepare a $2n$-qubit state ρ such that,
$$\sum_{A \in \mathcal{S}} \frac{1}{|\mathcal{S}|} \mathrm{Tr}(|A\rangle\langle A|^{\otimes 2} C^{U_A}(|A\rangle\langle A|)) \geq \epsilon.$$

3 Space-bounded Quantum Zero-Knowledge: Definitions

We present the definition of quantum zero knowledge with space bounded simulation. First, we will revisit the definition of zero-knowledge in Section 3.1. Towards defining space-bounded simulation, we present the computational model in Section 3.2. This model will explicitly capture the interaction between the classical memory and the quantum memory. We then define space-bounded quantum zero-knowledge in Section 3.3.

3.1 Defining Quantum Zero-Knowledge

We discuss the different types of quantum zero-knowledge below. We first start with the notion of quantum zero-knowledge, referred to as *black-box* quantum zero-knowledge, considered in the literature.

Black-Box QZK. Consider the following two definitions.

Definition 3.1 (Canonical purification). *Let V be a quantum channel that is implemented with quantum gates, intermediate measurements and trace-out operations. We denote the canonical purification of V as $\mathscr{P}(V)$, which consists of the unitary circuit that performs all the measurements coherently and replaces all the trace-out operations with identity gates.*

Any circuit can be expressed as its purification followed by measurements and trace out operations.

Definition 3.2. *Let V be a party in a k-round interactive protocol. We define $\mathscr{C}(V) = (\Phi_1, ..., \Phi_k)$, where $\Phi_i : \mathcal{M}_i \times \mathcal{V}_i \to \mathcal{M}_{i+1} \times \mathcal{V}_{i+1}$ to be the quantum channel corresponding to the computation performed by V on the ith round of the protocol, where $\mathcal{V}_i$ is the Hilbert space corresponding to the memory of V on round i and $\mathcal{M}_i$ is the Hilbert space corresponding to message sent in round i.*

We denote $\mathscr{P}(V) = (\mathscr{P}(\Phi_1), \ldots, \mathscr{P}(\Phi_k))$, where $\mathscr{C}(V) = (\Phi_1, \ldots, \Phi_k)$ and $\mathscr{P}$ is as defined in Definition 3.1. Similarly, we define $\mathscr{P}(V)^\dagger = \left(\mathscr{P}(\Phi_1)^\dagger, \ldots, \mathscr{P}(\Phi_k)^\dagger\right)$.

Most of the recent works [Wat09, ACP21, CCY21, CCLY21b, LMS21, CCLY21a] consider the following definition of quantum zero-knowledge.

Definition 3.3 (Black-Box Quantum Zero-Knowledge). *A proof (resp., argument) system (P, V) for an NP relation $\mathcal{R}$ is black-box quantum zero-knowledge if there exists a QPT simulator Sim such that for every QPT verifier V^*, for every $(x, w) \in \mathcal{R}$, for every $\mathrm{poly}(|x|)$-qubit state ρ,*

$$\mathsf{View}_{V^*}\left(\langle P(x, w), V^*(x, \rho)\rangle\right) \approx_{\mathcal{Q}, \varepsilon(|x|)} \mathsf{Sim}^{\mathscr{P}(V^*), \mathscr{P}(V^*)^\dagger}(x, \rho),$$

where Sim having oracle access to $\mathscr{P}(V^*)$ and $\mathscr{P}(V)^\dagger$ is denoted by $\mathsf{Sim}^{\mathscr{P}(V^*), \mathscr{P}(V^*)^\dagger}$, $\mathscr{P}(\cdot)$ is as defined in Definitions 3.1 and 3.2 and $\varepsilon(|x|)$ is a negligible function in $|x|$.

Comparison with Classical Black-Box ZK: In the definition of black-box classical zero-knowledge, the simulator has oracle access to the verifier circuit V. However, in Definition 3.3, the simulator does not have direct access to the verifier's circuit. Instead it has oracle access to the purification of the verifier circuit. Not only that, it also has oracle access to the inverse of the purification as well. Thus, we believe that the above definition is not the true quantum analogue of classical black-box zero-knowledge.

Fully Black-Box QZK. We consider a new definition that resembles the black-box definition in the classical setting. We call this definition fully black-box QZK. Just like the classical setting, the simulator in the fully black-box QZK only gets oracle access to the verifier circuit, and not the purified version as considered in Definition 3.3.

We now present our definition of fully black-box zero-knowledge.

Definition 3.4 (Fully Black-Box Quantum Zero-Knowledge). *A proof (resp., argument) system (P, V) for an NP relation $\mathcal{R}$ is fully black-box quantum zero-knowledge if there exists a QPT polynomial-time simulator Sim such that for every QPT malicious verifier V^*, for every $(x, w) \in \mathcal{R}$, for every state ρ,*

$$\mathsf{View}_{V^*}\left(\langle P(x, w), V^*(x, \rho)\rangle\right) \approx_{\mathcal{Q}, \varepsilon(|x|)} \mathsf{Sim}^{\mathscr{C}(V^*)}(x, \rho),$$

where Sim having oracle access to $\mathscr{C}(V^*)$ is denoted by $\mathsf{Sim}^{\mathscr{C}(V^*)}$, $\mathscr{C}(\cdot)$ is as defined in Definition 3.2 and $\varepsilon(|x|)$ is negligible in $|x|$.

Non Black-Box QZK. We can also define the most general definition of QZK where the simulator can depend arbitrary on the code of the verifier. We call this non black box QZK. This definition was considered in a few recent works [BS20, ALP20, BKS21].

Definition 3.5 (Non Black-Box Quantum Zero-Knowledge). *A proof (resp., argument) system (P, V) for an NP relation $\mathcal{R}$ is non black-box quantum zero-knowledge if for every QPT verifier V^*, there exists a QPT polynomial-time simulator Sim, for every $(x, w) \in \mathcal{R}$, for every $\mathrm{poly}(|x|)$-qubit state ρ,*

$$\mathsf{View}_{V^*}\left(\langle P(x, w), V^*(x, \rho)\rangle\right) \approx_{\mathcal{Q}, \varepsilon(|x|)} \mathsf{Sim}(x, \rho),$$

where $\varepsilon(|x|)$ is negligible in $|x|$.

3.2 Computational model

Towards defining space-bounded simulation, we define the model of quantum computation albeit using a different terminology. The new terminology is necessary to carefully model the interaction between the classical memory and the quantum memory since we are explicitly measuring the amount of quantum memory required for any computation.

First, we start with a weaker computational model. The following definition considers quantum circuits that take as input a unitary and a state and output the computation of the unitary on the state.

Definition 3.6 (*M*-qubit programmable quantum computer). *We say that Q is an M-qubit programmable quantum computer, if Q receives as an input an M-qubit quantum state $|\psi\rangle$, the (classical) description of an M-qubit unitary U and it outputs the state $U|\psi\rangle$. We depict such a quantum device in* Figure 1.
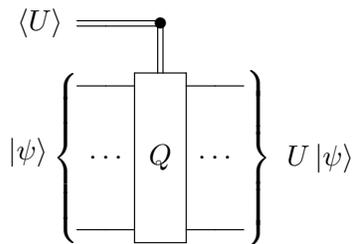
Figure 1: *M*-qubit programmable quantum computer

We now generalize the above model in the following way. After the computation of the unitary, we allow for some classical preprocessing. This is enabled by measuring a subset of the qubits of the state obtained after the unitary computation and the resulting measurement outcomes can be used to choose the function to be applied on the classical memory.

This entire computation is considered to be a block of computation. Later, we will consider the most general model where the computation is defined to a sequence of blocks.

Definition 3.7 (*M*-qubit of unitary quantum computation with classical post-processing). *An M-qubit of unitary quantum computation with classical post-processing is defined by a tuple $(c, \langle U\rangle, \ell, |\psi\rangle, f)$, where $c \in \{0, 1\}^L$ is a classical register, $\langle U\rangle$ is the description of an M-qubit unitary, $\ell \in [M + 1]$ is the number of qubits that will be measured, $|\psi\rangle$ is the input state of the quantum computation and f is a classical function used for post-processing.*

The computation proceeds as follows:

1. *Using an M-qubit programmable quantum computer Q, we run the unitary U on the state $|\psi\rangle$,*

19

2. *We measure the first ℓ qubits of the output, having outcome o and the state on the last $M - \ell$ qubits is $|\phi_o\rangle$,*

3. *We apply a classical function f on $(c, \langle U \rangle, \ell, o)$, resulting in $(c', \langle U' \rangle, \ell')$,*

4. *The first ℓ qubits are reset to $|0\rangle$,*

5. *The output of the computation is $(c', \langle U' \rangle, \ell', |\psi'\rangle)$, where $|\psi'\rangle = |0^\ell\rangle |\phi_o\rangle$.*

We depict such a computation in Figure 2.
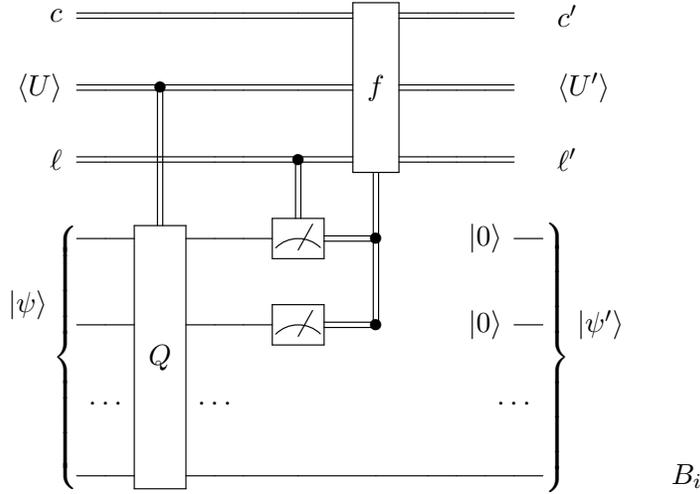
Figure 2: M-qubit of unitary quantum computation with classical post-processing

Our Model. We are now ready to define the model that we consider in this work. As mentioned earlier, the model below comprises of a sequence of blocks of computation, where each block is an M-qubit of unitary computation with classical post-processing.

Definition 3.8 (M-qubit quantum channel). *An M-qubit of quantum computation with classical post-processing is defined by a tuple $(c_0, \langle U_0 \rangle, \ell_0, |\psi_0\rangle, f)$, where $c \in \{0, 1\}^L$ is a classical register, $\langle U_0 \rangle$ is the description of an M-qubit unitary, $\ell_0 \in [M + 1]$ is the number of qubits that will be measured, $|\psi_0\rangle$ is the input state of the quantum computation and for f is a classical function used for post-processing.*

The computation proceeds as follows: We run an M-qubit of unitary quantum computation with classical post-processing (Definition 3.7) defined by $(c_i, \langle U_i \rangle, \ell_i, |\psi_i\rangle, f)$ and we let the output of the computation be $(c_{i+1}, \langle U_{i+1} \rangle, \ell_{i+1}, |\psi_{i+1}\rangle)$. If $\langle U_{i+1} \rangle = \perp$, this process stops and output c', ℓ' and $|\psi'\rangle$.

We depict such a computation in Figure 3.

We consider M-qubit channels that are efficiently implementable.

Definition 3.9 (M-qubit Quantum Polynomial Time Algorithm). *A (non-uniform) quantum polynomial algorithm C is said to be an M-qubit (resp. non-uniform) strict quantum polynomial-time*
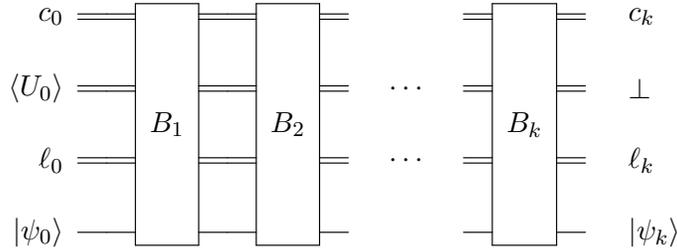
Figure 3: M-qubit quantum computation with classical post-processing. Each block B_i corresponds to the circuit in Figure 2 with the corresponding choices of input and classical post-processing f_i.

algorithm if it is an M-qubit quantum channel and there is a polynomial p such that each of the blocks runs in time at most $p(|x|)$ and after $p(|x|)$ rounds, the procedure stops (i.e., there exists some $k \leq p(|x|)$ such that $\langle U_k \rangle = \perp$).

A (non-uniform) quantum polynomial algorithm C is said to be an M-qubit (resp. non-uniform) expected quantum polynomial-time algorithm if it is an M-qubit quantum channel and there is a polynomial p such that each of the blocks runs in time at most $p(|x|)$ and the expected number of rounds of the procedure is $p(|x|)$ (i.e., $\mathbb{E}[k \mid \langle U_k \rangle = \perp] \leq p(|x|)$, where the expectation is over the measurement outcomes of the procedure).

Oracle Access. In order to discuss space-bounded black-box zero-knowledge, we need to define how the to give oracle access to an M-qubit quantum channel and how the memory of the overall channel is counted.

An M_A-qubit quantum channel A has oracle access to an M_B-qubit quantum channel B, for $M_A \geq M_B$, then A prepares a classical register $\mathbf{B_C}$ and M_B-qubit quantum register $\mathbf{B_Q}$, and then runs the channel B on input registers $\mathbf{B_C}$ and B_Q. A then has access to the output of B, consisting again of a classical register and an M_B-qubit quantum register.

3.3 Space-Bounded Quantum Zero-Knowledge

Equipped with the computational model described in Section 3.2, we state the definition of space-bounded quantum zero-knowledge. Roughly speaking, the definition states that the number of qubits consumed by the simulator should be a fixed polynomial in the number of qubits consumed by the verifier. Ideally, we would like the polynomial to be of low degree; however, we place no such restriction on the definition below.

Definition 3.10 ((s, p)-space-bounded QZK)**.** *Let $\mathcal{X} \in \{$fully black box, black box, non black box$\}$. An $\mathcal{X}$ QZK proof (resp., argument) system (P, V) for a language $\mathcal{L}$ is said to be (s, f)-space-bounded $\mathcal{X}$ QZK proof (resp., argument) system if the following property is satisfied:*

- *Suppose V^* is an ℓ_{V^*}-qubit QPT (Definition 3.9), where $\ell_{V^*} = s(|x|)$. Then, Sim is an ℓ_{Sim}-qubit QPT, where $\ell_{\mathsf{Sim}} \leq f(s(|x|))$.*

In the case of the black box QZK (Definition 3.3), the amount of quantum memory utilized by Sim is compared against the amount of quantum memory utilized by V^* before its purification. Moreover, by definition, the quantum memory of Sim contains the memory of the purified verifier V^*. For

any polynomial $f(\cdot)$, for any s, it is possible to come up with an s-qubit channel[7] (Definition 3.8) such that purifying this channel would require strictly more than $f(s)$ qubits. This means that any black box simulator against a verifier represented as this s-qubit channel would utilize more than $f(s)$ qubits. Hence, (s, f)-space bounded black box QZK, for any polynomial f, is impossible to achieve. On the other hand, for superpolynomial $f(\cdot)$, existing works demonstrate that (s, f)-space bounded QZK exists .

From the above discussion, it follows that the only two meaningful notions of space bounded QZK to consider are fully black box QZK and non black box QZK.

4 Zero-Knowledge against Logarithmic Quantum Space Verifiers

We present the construction of an argument system Π for the NP relation $R(\mathcal{L})$ which is space-bounded fully black-box zero-knowledge against $O(\log(\lambda))$-qubit quantum channels (Definition 3.8).

We use the following tools in our construction.

- Statistically-binding and quantum-concealing commitment protocol (see Section 2.4), denoted by $(\mathsf{Comm}, \mathsf{Receiver}, \mathsf{Open})$.

- Four round quantum witness-indistinguishable proof system Π_{wi} (Definition 2.4). The relation associated with Π_{wi}, denoted by $\mathcal{R}(\mathcal{L}_{\mathsf{wi}})$, is defined as follows:

$$\mathcal{R}(\mathcal{L}_{\mathsf{wi}}) = \left\{ \left(\left(x, \mathbf{c}^*, \mathbf{c}^{**}, \{\alpha_{i,b}\}_{i \in [\lambda], b \in \{0,1\}} \right); (w, \mathbf{r}^*, r, i^*) \right) : (x, w) \in \mathcal{R}(\mathcal{L}) \text{ or} \right.$$

$$\left. \mathsf{Open}(\mathbf{c}^*) = (\mathbf{r}^*, r) \text{ and } \mathsf{Open}(\mathbf{c}^{**}) = ((i^*, \alpha_{i^*,0} \oplus \alpha_{i^*,1}), \mathbf{r}^*) \right\}$$

We present the construction of Π in Figure 4.

Completeness. The completeness of the protocol Π in Figure 4 follows from the completeness of Π_{wi}.

4.1 Proof of Soundness

Proposition 1. *The protocol Π in Figure 4 satisfies soundness.*

Proof. Let P^* be a (non-uniform) malicious prover. Let $x \in \{0,1\}^*$ be such that $x \notin \mathcal{L}$. We claim that $\Pr\left[\mathsf{Accept} \leftarrow \langle P^*(x), V(x) \rangle\right] \leq \mathsf{negl}(\lambda)$.

From the soundness of Π_{wi}, the following holds:

$$\Pr\left[\mathsf{Accept} \leftarrow \langle P^*(x), V(x) \rangle\right] \leq \Pr\left[\mathbf{x}_{\mathsf{wi}} \in \mathcal{L}_{\mathsf{wi}}\right] + \mathsf{negl}(\lambda), \tag{1}$$

where x_{wi} is a random variable that is assigned the WI instance sampled in the execution of $P^*(x)$.

Therefore, to prove soundness, it suffices upper bound the value $\varepsilon := \Pr[\mathbf{x}_{\mathsf{wi}} \in \mathcal{L}_{\mathsf{wi}}]$.

---

[7]For example, consider an s-qubit channel contains $f(s)$ bits of classical memory. After canonical purification, the overall amount of quantum memory would be more than $f(s)$ qubits. Note that we don't rule out other types of purification which critically use the structure of the s-qubit channel to not increase the quantum memory significantly. However, this is not in the spirit of black box QZK and we consider such variants to be closer to non black box QZK.

Input of P: Instance $x \in \mathcal{L}$ along with witness w.

Input of V: Instance $x \in \mathcal{L}$.

Trapdoor Phase:

1. $P \leftrightarrow V$: P and V engage in an execution of the commitment protocol, where P plays the role of the committer and V plays the role of the receiver. P commits to $\mathbf{r}^*$, where $\mathbf{r}^* \xleftarrow{\$} \{0,1\}^\lambda$. Denote the commitment to be $\mathbf{c}^*$. Let the randomness used by P in the commitment protocol be r.

2. $V \leftrightarrow P$: P and V run 2λ parallel executions of the commitment protocol, with V playing the role of the committer and P playing the role of the receiver. We index the executions using the notation (i,b), where $i \in [\lambda]$ and $b \in \{0,1\}$. In the $(i,b)^{th}$ execution, V commits to $\alpha_{i,b}$, where $\alpha_{i,b} \xleftarrow{\$} \{0,1\}^\lambda$. Denote the commitments by $\left(\{\mathbf{c}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right)$.

3. $P \to V$: P sends $\mathbf{b}_1, \ldots, \mathbf{b}_\lambda$, where $\mathbf{b}_i \xleftarrow{\$} \{0,1\}^\lambda$.

4. $V \to P$: V sends the openings of all the commitments $\left(\{\mathbf{c}_{i,\mathbf{b}_i}\}_{i\in[\lambda]}\right)$. Denote the opening of the $(i,\mathbf{b}_i)^{th}$ commitment to be $o_{i,\mathbf{b}_i} = \mathsf{Open}(\mathbf{c}_{i,\mathbf{b}_i})$. If any of the openings are invalid, P rejects. Otherwise, P parses $o_{i,\mathbf{b}_i}$ as $(\alpha_{i,\mathbf{b}_i}, r_{i,\mathbf{b}_i})$.

5. $P \leftrightarrow V$: P and V engage in another execution of the commitment protocol, where P plays the role of the committer and V plays the role of the receiver. P commits to 0, using the randomness $\mathbf{r}^*$. Denote the commitment to be $\mathbf{c}^{**}$.

6. $V \to P$: V sends the rest of the openings of $\left(\{\mathbf{c}_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right)$. That is, it sends $\{o_{i,1-\mathbf{b}_i}\}_{i\in[\lambda]}$, where $o_{i,1-\mathbf{b}_i} = \mathsf{Open}(\mathbf{c}_{i,1-\mathbf{b}_i})$. If any of the openings are invalid, P rejects. Otherwise, P parses $o_{i,1-\mathbf{b}_i}$ as $(\alpha_{i,1-\mathbf{b}_i}, r_{i,1-\mathbf{b}_i})$.

WI Phase: P and V engage in Π_{wi} with the common input being the following:

$$\mathbf{x}_{\mathsf{wi}} = \left(x, \mathbf{c}^*, \mathbf{c}^{**}, \{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right)$$

Additionally, P uses the witness $(w, \bot, \bot, \bot)$.

Figure 4: Description of the protocol Π.

Lemma 4.1. $\varepsilon \leq \mathsf{negl}(\lambda)$.

Proof. Let $\mathbf{x}_{\mathsf{wi}} = \left(x, \mathbf{c}^*, \mathbf{c}^{**}, \{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right)$ be the Π_{wi} instance computed during the execution of $P^*(x)$ and $V(x)$. Define the following event, parameterized by a set of values:

Event $\mathbf{E}\left[\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right]$: Output 1 if there exists an $\mathbf{i} \in [\lambda]$ such that the commitment $\mathbf{c}^{**}$ sent

by P^* is a commitment of $(\alpha_{\mathbf{i},0} \oplus \alpha_{\mathbf{i},1})$.

Consider the following hybrid argument.

$\underline{\mathsf{H}_0}$: This corresponds to the execution of $P^*(x)$ and $V(x)$.

Let $\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}$ be the values committed by V in Step 2. Since $x \notin \mathcal{L}$, the following holds:

$$\Pr\left[\mathbf{E}\left[\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right] = 1 \text{ in } \mathsf{H}_0\right] = \Pr\left[\mathsf{x}_{\mathsf{wi}} \in \mathcal{L}_{\mathsf{wi}}\right] = \varepsilon$$

$\underline{\mathsf{H}_1}$: We modify the description of $V(x)$ as follows. Sample $\alpha_i \xleftarrow{\$} \{0,1\}^\lambda, \alpha_{i,b} \xleftarrow{\$} \{0,1\}^\lambda$ such that $\alpha_i = \alpha_{i,0} \oplus \alpha_{i,1}$.

The two hybrids H_0 and H_1 are identical. Thus, we have the following:

$$\Pr\left[\mathbf{E}\left[\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right] = 1 \text{ in } \mathsf{H}_1\right] = \varepsilon$$

$\underline{\mathsf{H}_{2.(i^*,b^*)}}$ for $i^* \in [2\lambda], b^* \in \{0,1\}$: The verifier behaves as in H_1 except for the following: for every $i \in [\lambda]$, it additionally samples $\beta_{i,0} \xleftarrow{\$} \{0,1\}^\lambda, \beta_{i,1} \xleftarrow{\$} \{0,1\}^\lambda$. Furthermore, for every $i \leq i^*$, $b \leq b^*$, the $(i,b)^{th}$ committed value by V in Step 2 is $\beta_{i,b}$. We emphasize that for every $i \in [\lambda]$, it still samples $\alpha_{i,0} \xleftarrow{\$} \{0,1\}^\lambda, \alpha_{i,1} \xleftarrow{\$} \{0,1\}^\lambda$.

We prove the following two claims.

Claim 1. *Assuming the quantum-concealing property of* $(\mathsf{Comm}, \mathsf{Receiver}, \mathsf{Open})$, *there exists a negligible function* $\nu(\lambda)$ *such that the following holds:*

1. For every $i^* \in [\lambda]$,

$$\left|\Pr\left[\mathbf{E}\left[\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right] \text{ in } \mathsf{H}_{2.(i^*.0)}\right] - \Pr\left[\mathbf{E}\left[\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right] \text{ in } \mathsf{H}_{2.(i^*.1)}\right]\right| \leq \nu(\lambda)$$

2. For every $i^* \in [\lambda-1]$,

$$\left|\Pr\left[\mathbf{E}\left[\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right] \text{ in } \mathsf{H}_{2.(i^*-1.1)}\right] - \Pr\left[\mathbf{E}\left[\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right] \text{ in } \mathsf{H}_{2.(i^*.0)}\right]\right| \leq \nu(\lambda),$$

3.

$$\left|\Pr\left[\mathbf{E}\left[\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right] \text{ in } \mathsf{H}_1\right] - \Pr\left[\mathbf{E}\left[\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right] \text{ in } \mathsf{H}_{2.(1.0)}\right]\right| \leq \nu(\lambda)$$

Proof. We will prove just the first bullet and the other items follow analogously.

Let $\delta_{i.\mathbf{b}} = \Pr\left[\mathbf{E}\left[\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right] \text{ in } \mathsf{H}_{2.(i.\mathbf{b})}\right]$. Suppose the first bullet does not hold. This means that there exits $i^* \in [\lambda]$ and a non-negligible function $\nu(\lambda)$ such that $|\delta_{i^*.0} - \delta_{i^*.1}| \geq \nu(\lambda)$.

We construct a non-uniform reduction $\mathcal{B}$ as follows. It receives as non-uniform advice, $(\mathbf{r}^*, \mathbf{c}^*, \sigma_{\mathsf{adv}})$ generated as follows:

- Run $P^*(x)$ to generate $\mathbf{c}^*$.
- Let $\mathbf{r}^*$ be the value committed in $\mathbf{c}^*$.
- Let σ_{adv} be the state of $P^*(x)$ after the execution of step 1 of Π in Figure 4.

$\mathcal{B}$ proceeds as follows:

- For every $i \in \{0,1\}^\lambda$, it samples $\alpha_{i,0}, \alpha_{i,1}, \beta_{i,0}, \beta_{i,1} \xleftarrow{\$} \{0,1\}^\lambda$,

- It runs $P^*(x)$ starting from step 2 of Π in Figure 4. P^* is initialized with the state σ_{adv}.

- For all the $(i,0)^{th}$ and $(i,1)^{th}$ executions of step 2 of Π, $\mathcal{B}$ plays the role of $V(x)$. That is, in the $(i,b)^{th}$ execution, for $i < i^*$, it commits to $\beta_{i,b}$. In the $(i,b)^{th}$ execution for $i > i^*$, it commits to $\alpha_{i,b}$. Similarly, for $i = i^*$, in the $(i,0)^{th}$ execution, it commits to $\beta_{i,b}$. For the $(i^*,1)^{th}$ execution, it acts as an intermediary between the external challenger, which plays the role of sender in (Comm, Receiver, Open), and the prover $P^*(x)$. The challenger uses the challenge message pair $(\alpha_{i^*,1}, \beta_{i^*,1})$ in the quantum-concealing experiment when interacting with $\mathcal{B}$, who plays the role of the malicious receiver.

- $\mathcal{B}$ continues the execution of $P^*(x)$ by emulating the verifier $V(x)$ until step 5 of Π.

- Let $\mathbf{c}^{**}$ be the commitment sent by $P^*(x)$ in step 5.

- $\mathcal{B}$ recovers the message γ from $\mathbf{c}^{**}$ using $\mathbf{r}^*$. If the recovery fails, it aborts.

- If there exists $i \in [\lambda]$ such that $\gamma = \alpha_{i,0} \oplus \alpha_{i,1}$, $\mathcal{B}$ outputs 1. Otherwise, it outputs 0.

If the challenger uses the message $\alpha_{i^*,1}$ in the execution of the commitment protocol then the probability that $\mathcal{B}$ outputs 1 is precisely $\delta_{i^*,0}$. Similarly, if the challenger uses $\beta_{i^*,1}$ then the probability that $\mathcal{B}$ outputs 1 is precisely $\delta_{i^*,1}$.

Since $|\delta_{i^*,0} - \delta_{i^*,1}| \geq \nu(\lambda)$, where $\nu(\lambda)$ is non-negligible, we have that $\mathcal{B}$ violates the quantum-concealing property of (Comm, Receiver, Open), a contradiction. Thus, $\nu(\lambda)$ is negligible which completes the proof of the claim. $\qquad\square$

As a corollary and from the fact that $\Pr\left[\mathbf{E}\left[\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right] = 1 \text{ in } \mathsf{H}_1\right] = \varepsilon$, we have the following.

Corollary 4.2.
$$\left|\Pr\left[\mathbf{E}\left[\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right] \text{ in } \mathsf{H}_{2.(\lambda.1)}\right]\right| \geq \varepsilon - (2\lambda) \cdot \nu(\lambda)$$

Proof. Follows by applying triangle inequality on Claim 1. $\qquad\square$

Claim 2. $\Pr\left[\mathbf{E}\left[\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}\right] = 1 \text{ in } \mathsf{H}_{2.(\lambda.1)}\right] \leq \frac{\lambda}{2^\lambda}$.

Proof. In $\mathsf{H}_{2.(\lambda.1)}$, $\{\alpha_{i,b}\}_{i\in[\lambda],b\in\{0,1\}}$ is information-theoretically hidden from $P^*(x)$. Thus, for any given $i \in [\lambda]$, the probability that $\alpha_{i,0} \oplus \alpha_{i,1}$ is committed by P^* in bullet 5 of Π is $\frac{1}{2^\lambda}$. By union bound, the proof of the claim follows. $\qquad\square$

From Corollary 4.2 and Claim 2, ε is a negligible function in λ.

$\qquad\square$

From Lemma 4.1 and Equation 1, it follows that $\Pr\left[\mathsf{Accept} \leftarrow \langle P^*(x), V(x)\rangle\right] \leq \mathsf{negl}(\lambda)$.

$\qquad\square$

4.2 Proof of Quantum Zero-Knowledge

Proposition 2. *Let $M = O(\log \lambda)$ and $f(M) = 2M$. The protocol described in Figure 4 is (M, f)-space bounded QZK.*

Proof. Suppose V^* be a malicious QPT (non-uniform) verifier with M-qubit quantum space. We describe a QPT simulator Sim such that the output distribution of Sim is computationally indistinguishable from the output distribution of the real world.

Description of Simulator. Let $\mathbf{R}$ and $\mathbf{X}$ be two M-qubit registers. The register $\mathbf{X}$ is V^*'s private register whereas $\mathbf{R}$ is an auxiliary register maintained by the simulator. The verifier V^*'s advice state is stored in the register $\mathbf{X}$. We describe the simulator Sim in Figure 5.

We need to show that Sim indeed runs in expected QPT. We also need to demonstrate that the output distribution of Sim is computationally indistinguishable from the real world.

We first prove a simple claim. Define Λ be a POVM that on input σ, defined on $\mathbf{X}$, implements steps 3 and 4 of Π in Figure 4 and outputs 1 if and only if V^* does **not** abort. We assume that the classical state is hardcoded in the description of Λ. Define another POVM Γ that on input σ, defined on $\mathbf{R}$, implements 2.2 of Figure 5 and outputs 1 if and only if V^* did **not** abort and $\exists i \in [\lambda]$ such that $\mathbf{b}_i \neq \mathbf{b}_i'$.

Claim 3. *Suppose $p = \mathrm{Tr}(\Lambda(\rho_{\mathsf{res}}))$, where ρ_{res} is as defined in Figure 5. Then, $\mathrm{Tr}(\Gamma(\frac{I}{2^M})) \geq \frac{p}{2^M} \cdot (1 - 2^{-\lambda})$.*

Proof. Firstly, note that $\mathrm{Tr}(\Lambda(\sigma)) \cdot (1 - 2^{-\lambda}) = \mathrm{Tr}(\Gamma(\sigma))$, for every density matrix σ. This follows from the fact that the probability that $\exists i \in [\lambda]$ such that $\mathbf{b}_i \neq \mathbf{b}_i'$ is precisely $(1 - 2^{-\lambda})$.

Now, we estimate $\mathrm{Tr}(\Lambda(\frac{I}{2^M}))$. We have:

$$
\begin{aligned}
\mathrm{Tr}\left(\Lambda\left(\frac{I}{2^M}\right)\right) &= \frac{1}{2^M}\mathrm{Tr}\left(\Lambda\left(I\right)\right) \\
&= \frac{1}{2^M}\mathrm{Tr}\left(\Lambda\left(\rho_{\mathsf{res}} + I - \rho_{\mathsf{res}}\right)\right) \\
&\geq \frac{1}{2^M}\mathrm{Tr}\left(\Lambda\left(\rho_{\mathsf{res}}\right)\right) \quad (\because \; I - \rho_{\mathsf{res}} \succeq 0) \\
&= \frac{p}{2^M}
\end{aligned}
$$

Thus, $\mathrm{Tr}\left(\Gamma(\frac{I}{2^M})\right) = \mathrm{Tr}\left(\Lambda(\frac{I}{2^M})\right) \cdot (1 - 2^{-\lambda}) \geq \frac{p}{2^M} \cdot (1 - 2^{-\lambda})$. $\qquad\square$

Runtime Analysis. Let $q(\lambda)$ be an upper bound on the runtime of all the steps (1 through 6) in Figure 5. Let ρ be the auxiliary state of V^*. We use the notation Λ and Γ as defined earlier. Let $p = \mathrm{Tr}(\Lambda(\rho_{res}))$ and $p' = \mathrm{Tr}(\Gamma(\frac{I}{2^M}))$. From 3, $p' \geq \frac{p}{2^M}(1 - 2^{-\lambda})$. There are three cases.

Case 1: $p = 0$. In this case, Sim aborts after step 1 of Figure 5. Thus, Sim runs in polynomial time.

Case 2: $p' = 1$. In this case, either Sim aborts in step 1 or it runs the loop in step 2 only once. In either of the two cases, Sim runs in polynomial time.

1. By taking the role of the honest prover, run the first four steps in Figure 4. That is, until the simulator receives $\{(\alpha_{i,\mathbf{b}_i}, r_{i,\mathbf{b}_i})\}_{i \in [\lambda]}$. Abort if V^* aborts at any point. We emphasize that V^* is executed on input the register $\mathbf{X}$. Denote ρ_{res} to be the quantum residual state after this step. Denote the classical state to be st.

   1.1. Sim and V^* engage in an execution of the commitment protocol, where Sim commits to $\mathbf{r}^*$, which is sampled as $\mathbf{r}^* \xleftarrow{\$} \{0,1\}^\lambda$. Denote the commitment to be $\mathbf{c}^*$.

   1.2. Sim and V^* run 2λ number of executions of the commitment protocol. Denote the commitments in all the 2λ executions by $(\{\mathbf{c}'_{i,b}\}_{i \in [\lambda], b \in \{0,1\}})$.

   1.3. Sim sends $\mathbf{b}_1, \ldots, \mathbf{b}_\lambda$, where $\mathbf{b}_i \xleftarrow{\$} \{0,1\}^\lambda$, to V^*.

   1.4. Sim receives the openings of all the commitments $(\{\mathbf{c}_{i,\mathbf{b}_i}\}_{i \in [\lambda]})$. Denote the opening of the $(i, \mathbf{b}_i)^{th}$ commitment to be $o_{i,\mathbf{b}_i} = \mathsf{Open}(\mathbf{c}_{i,\mathbf{b}_i})$. If any of the openings are invalid, reject. Otherwise, parse $o_{i,\mathbf{b}_i}$ as $(\alpha_{i,\mathbf{b}_i}, r_{i,\mathbf{b}_i})$.

2. Repeat the following forever:

   2.1. Initialize the register $\mathbf{R}$ with the maximally mixed state $\frac{I}{2^M}$.

   2.2. Run the third and fourth steps of Figure 4, where V^* is executed on the register $\mathbf{R}$. That is, send $\mathbf{b}'_1, \ldots, \mathbf{b}'_\lambda \xleftarrow{\$} \{0,1\}^\lambda$. Receive the openings from V^*, denoted by $\{(\alpha_{i,\mathbf{b}'_i}, r_{i,\mathbf{b}'_i})\}_{i \in [\lambda]}$. Abort the loop if V^* did not abort and $\exists i \in [\lambda]$ such that $\mathbf{b}_i \neq \mathbf{b}'_i$. Otherwise, continue.

3. Discard the register $\mathbf{R}$. From here on, continue (from Step 1) the execution of V^* on the register $\mathbf{X}$. Specifically, run V^* on $(\rho_{\mathsf{res}}, \mathsf{st})$.

4. The simulator commits to $(i^*, \alpha_{i^*,0} \oplus \alpha_{i^*,1})$, where $i^* \in [\lambda]$ is such that $\mathbf{b}_{i^*} \neq \mathbf{b}'_{i*}$. Denote the commitment to be $\mathbf{c}^{**}$ and let $\mathbf{r}^*$ be the randomness used in $\mathbf{c}^{**}$.

5. It runs the last step of the trapdoor phase by emulating the execution of the honest prover.

6. Finally, Sim runs the WI phase with V^*. Let the instance be $\mathbf{x}_{\mathsf{wi}} = \left(x, \mathbf{c}^*, \mathbf{c}^{**}, \{\alpha_{i,b}\}_{i \in [\lambda], b \in \{0,1\}} \right)$, where $\mathbf{c}^*$ be the commitment sent by Sim in the beginning of the protocol. Sim emulates the honest prover except the witness used is $(\perp, \mathbf{r}^*, r, i^*)$, where r is the randomness used in the commitment of $\mathbf{c}^*$.

Figure 5: (Expected) QPT Simulator Sim.

Case 3: $p > 0$ and $p' < 1$.

$$\text{Expected Runtime} \;=\; q(\lambda)\left((1-p)\cdot 1 + p\sum_{i=1}^{\infty} i\cdot(1-p')^{i-1}\cdot p'\right) + 4q(\lambda)$$

$$=\; q(\lambda)\left((1-p)\cdot 1 + \frac{p\cdot p'}{(1-p')}\sum_{i=1}^{\infty} i\cdot(1-p')^{i}\right) + 4q(\lambda)$$

$$=\; q(\lambda)\left((1-p)\cdot 1 + \frac{p\cdot p'}{(1-p')}\cdot\frac{(1-p')}{(p')^2}\right) + 4q(\lambda) \;\left(\because \sum_{i=1}^{\infty} i\cdot c^i = \frac{c}{(1-c)^2} \text{ for } c>0\right)$$

$$=\; q(\lambda)\left((1-p)\cdot 1 + \frac{p}{p'}\right) + 4q(\lambda)$$

$$\leq\; q(\lambda)\left((1-p) + \frac{p}{\frac{p(1-2^{-\lambda})}{2^M}}\right) + 4q(\lambda)$$

$$\leq\; q(\lambda)\left((1-p) + 2^{M+1}\right) + 4q(\lambda)$$

$$=\; \text{poly}(\lambda)$$

The last equality follows from the fact that M is logarithmic in λ.

Indistinguishability of Real and Ideal worlds. Let $(x,w) \in \mathcal{R}(\mathcal{L})$. Let ρ be the auxiliary state of V^*.

Lemma 4.3. $\{\mathsf{View}_{V^*}(\langle P(x,w), V^*(x,\rho)\rangle)\} \approx_{\mathcal{Q}} \{\mathsf{Sim}(x,\rho)\}$.

Proof. Consider the following hybrid argument.

$\underline{\mathsf{H}_0}$: The output distribution of this hybrid is $\mathsf{View}_{V^*}(\langle P(x,w), V^*(x,\rho)\rangle)$.

$\underline{\mathsf{H}_1}$: We define a hybrid simulator that essentially behaves like an honest prover except it keeps rewinding until it finds two accepting transcripts. We emphasize that, just like the honest prover, the hybrid simulator still uses the witness w in the WI phase. Formally, we consider a hybrid simulator $\mathsf{H}_1.\mathsf{Sim}(x,w)$ defined as follows:

1. Execute steps 1 through 3 of Sim in Figure 5.

   1.1. Run the following steps:

      i. Engage in an execution of the commitment protocol with V^*, by committing to $\mathbf{r}^*$, which is sampled as $\mathbf{r}^* \xleftarrow{\$} \{0,1\}^\lambda$. Denote the commitment to be $\mathbf{c}^*$.

      ii. Execute 2λ number of executions of the commitment protocol with V^*, with V^* playing the role of the committer. Denote the commitments in all the 2λ executions by $(\{\mathbf{c}'_{i,b}\}_{i\in[\lambda],b\in\{0,1\}})$.

      iii. Send $\mathbf{b}_1,\ldots,\mathbf{b}_\lambda$, where $\mathbf{b}_i \xleftarrow{\$} \{0,1\}^\lambda$, to V^*.

      iv. Receive the openings of all the commitments $(\{\mathbf{c}_{i,\mathbf{b}_i}\}_{i\in[\lambda]})$. Denote the opening of the $(i,\mathbf{b}_i)^{th}$ commitment to be $o_{i,\mathbf{b}_i} = \mathsf{Open}(\mathbf{c}_{i,\mathbf{b}_i})$. If any of the openings are invalid, reject. Otherwise, parse $o_{i,\mathbf{b}_i}$ as $(\alpha_{i,\mathbf{b}_i}, r_{i,\mathbf{b}_i})$.

1.2. Repeat the following forever:

    i. Initialize the register $\mathbf{R}$ with the maximally mixed state $\frac{I}{2^M}$.

    ii. Run the third and fourth steps of Figure 4, where V^* is executed on the register $\mathbf{R}$. That is, send $\mathbf{b}'_1, \ldots, \mathbf{b}'_\lambda \xleftarrow{\$} \{0,1\}^\lambda$. Receive the openings from V^*, denoted by $\{(\alpha_{i,\mathbf{b}'_i}, r_{i,\mathbf{b}'_i})\}_{i \in [\lambda]}$. Abort the loop if V^* did not abort and $\exists i \in [\lambda]$ such that $\mathbf{b}_i \neq \mathbf{b}'_i$. Otherwise, continue

1.3. Discard the register $\mathbf{R}$. From here on, continue (from Step 1) the execution of V^* on the register $\mathbf{X}$.

2. Execute steps 5 and 6 of Π in Figure 4. Also, execute the WI phase.

2.1. Engage in another execution of the commitment protocol by committing to 0, using the randomness $\mathbf{r}^*$. Denote the commitment to be $\mathbf{c}^{**}$.

2.2. V^* output the rest of the openings of $\left(\{\mathbf{c}_{i,b}\}_{i \in [\lambda], b \in \{0,1\}}\right)$. That is, it sends $\{o_{i,1-\mathbf{b}_i}\}_{i \in [\lambda]}$, where $o_{i,\mathbf{b}_i} = \mathsf{Open}(\mathbf{c}_{i,1-\mathbf{b}_i})$. If any of the openings are invalid, P rejects. Otherwise, P parses $o_{i,1-\mathbf{b}_i}$ as $(\alpha_{i,1-\mathbf{b}_i}, r_{i,1-\mathbf{b}_i})$.

2.3. Engage in Π_{wi} with V^* on the following common input:

$$\mathbf{x}_{\mathsf{wi}} = \left(x, \mathbf{c}^*, \mathbf{c}^{**}, \{\alpha_{i,b}\}_{i \in [\lambda], b \in \{0,1\}}\right)$$

Additionally, use the witness $(w, \perp, \perp, \perp)$ in Π_{wi}.

The output of $\mathsf{H}_1.\mathsf{Sim}(x, w)$ is the view of V^* in H_1.

Consider the following claim.

Claim 4. *The trace distance between H_0 and $\mathsf{H}_1.\mathsf{Sim}(x, w)$ is at most $\mathsf{negl}(\lambda)$.*

Proof. Note that if $\mathsf{H}_1.\mathsf{Sim}(x, w)$ terminates then the density matrices output in H_0 and $\mathsf{H}_1.\mathsf{Sim}(x, w)$ are identically distributed. It suffices to upper bound the probability that $\mathsf{H}_1.\mathsf{Sim}(x, w)$ does not terminate. Let $p = \mathrm{Tr}(\Lambda(\rho))$, where ρ is the advice state of V^*. Let $p' = \mathrm{Tr}(\Gamma(\rho))$. From Claim 3, $p' \geq \frac{p(1-2^{-\lambda})}{2^M}$. There are two cases:

- *Case 1. $p = 0$ or $p' = 1$:* As we argued in the proof of Claim 3, Sim terminates in this case.

- *Case 2. $p \neq 0$ and $p' \neq 1$:* In this case, we claim that the simulator terminates within $\lambda \cdot (p')^{-1}$ iterations with overwhelming probability. We show this below.

$$\Pr\left[\mathsf{Sim} \text{ does not terminate within } \lambda \cdot (p')^{-1} \text{ iterations}\right] \leq (1 - p')^{\lambda \cdot (p')^{-1}} = e^{-\lambda}$$

$\square$

$\underline{\mathsf{H}_2}$: We define another hybrid simulator $\mathsf{H}_2.\mathsf{Sim}$ which behaves like $\mathsf{H}_1.\mathsf{Sim}$, except in the following steps:

1.1.i Engage in an execution of the commitment protocol with V^*, by committing to 0. Denote the commitment to be $\mathbf{c}^*$.

    Note: in bullet 2.1 of H_1, $\mathbf{c}^{**}$ is still a commitment computed using the randomness $\mathbf{r}^*$, where $\mathbf{r}^* \xleftarrow{\$} \{0,1\}^\lambda$.

The rest of the steps is the same as $H_1.Sim$. The output of $H_2.Sim$ is the view of V^* in H_2.

The output distributions of $H_1.Sim$ and $H_2.Sim$ are computationally indistinguishable from the quantum-concealing property of $(Comm, Open, Receiver)$.

$\underline{H_3}$: We define a hybrid simulator $H_3.Sim$ which behaves like $H_2.Sim$, except in the following steps:

2.1 Engage in an execution of the commitment protocol with V^* by committing to $(i^*, \alpha_{i^*,0} \oplus \alpha_{i^*,1})$, where $i^* \in [\lambda]$ is such that $\mathbf{b}_{i^*} \neq \mathbf{b}'_{i^*}$. Denote the commitment to be $\mathbf{c}^{**}$.

The rest of the steps is the same as $H_2.Sim$. The output of $H_3.Sim$ is the view of V^* in H_3.

The output distributions of $H_2.Sim$ and $H_3.Sim$ are computationally indistinguishable from the quantum-concealing property of $(Comm, Open, Receiver)$.

$\underline{H_4}$: We define a hybrid simulator $H_4.Sim$ which behaves like $H_3.Sim$, except in the following steps:

1.1.i Engage in an execution of the commitment protocol with V^*, by committing to $\mathbf{r}^*$, which is sampled as $\mathbf{r}^* \xleftarrow{\$} \{0,1\}^\lambda$. Denote the commitment to be $\mathbf{c}^*$.

The rest of the steps is the same as $H_3.Sim$. The output of $H_4.Sim$ is the view of V^* in H_4.

The output distributions of $H_3.Sim$ and $H_4.Sim$ are computationally indistinguishable from the quantum-concealing property of $(Comm, Open, Receiver)$.

$\underline{H_5}$: The output of this hybrid is the output of $Sim(x, \rho)$.

The only difference between $H_4.Sim$ and Sim is in the WI phase. The computational indistinguishability of H_4 and H_5 follows from the witness indistinguishability property of Π_{wi}. $\qquad \square$

$\hfill \square$

5 $\omega(\log(\lambda))$-Quantum Space Verifiers: Impossibility Result

In the previous section, we showed the existence of quantum zero-knowledge protocols secure against logarithmic quantum space verifiers. We present a complementary result. We show that there does not exist any quantum zero-knowledge protocol with fully black box simulation for languages outside BQP against super-logarithmic quantum space verifiers.

Lemma 5.1. *Let $s(\lambda) = \omega(\log(\lambda))$ and f be any function such that $f(x) > 2x$. Assume the existence of post-quantum one-way functions. Suppose there exists a fully black-box (s, f)-space bounded quantum zero-knowledge protocol for a language $\mathcal{L}$. Then, $\mathcal{L}$ is in BQP.*

Proof. Consider a language $\mathcal{L}$. Suppose there exists a k-round fully black-box (s, f)-space bounded quantum zero-knowledge protocol $\mathcal{L}$ for some $s(\lambda) = \omega(\log(\lambda))$ and for some function f.

We describe a malicious verifier V' below. In addition to subspace states (Section 2.5), we use the following additional tools:

- Symmetric-key encryption with post-quantum CPA (chosen plaintext attack) security, denoted by $(Setup, Enc, Dec)$.

- Post-quantum secure digital signatures satisfying existential unforgeability property, denoted by $(Gen, Sign, Verify)$.

Both of these primitives can be built from post-quantum secure one-way functions [Gol01, NY89].

Description of V'. Consider the following QPT verifier V' such that the quantum memory of V' is $M = \omega(\lambda)$ qubits. Hardwired in its code are k random $\frac{M}{2}$-dimensional subspaces $S_0,...,S_k$, where $S_i \subseteq \mathbb{F}_2^M$, a secret-key sk_{enc} for a symmetric-key encryption scheme, sampled as $sk_{\mathrm{enc}} \leftarrow \mathsf{Setup}(1^\lambda)$, and a secret-key and verification keys $(sk_{\mathrm{sign}}, vk_{\mathrm{sign}})$ for a digital signature scheme, sampled as $(sk_{\mathrm{sign}}, vk_{\mathrm{sign}}) \leftarrow \mathsf{Gen}(1^\lambda)$. V' receives as input M-qubit subspace state $|S_0\rangle$, which is initialized in the register **Y**.

In the first round of the protocol, V' gets as input two registers **X**, **Y** and does the following:

1. Measure the register **X** to obtain α_1.

2. Check if the state held in **Y** is $|S_0\rangle\langle S_0|$. That is, perform the projective measurement $\{ \underbrace{|S_0\rangle\langle S_0|}_{\mathrm{outcome=0}}, \underbrace{I - |S_0\rangle\langle S_0|}_{\mathrm{outcome=1}} \}$ on the register **Y**. This is possible since V' has the description of S_0 hardwired in its code.

3. If the measured outcome is 1, abort.

4. Run the first round of V on α_1 to obtain β_1. Abort if V aborts.

5. Re-initialize the register **X** with $|\beta_1\rangle\langle\beta_1|$.

6. Re-initialize the register **Y** with a new M-qubit subspace state $|S_1\rangle\langle S_1|$ in the memory.

7. Using the hardcoded sk_{enc}, initialize the register **Z** with the $c_1 = \mathsf{Enc}(sk_{\mathrm{enc}}, \gamma_1)$, where γ_1 is the state of V after the answer.

8. Using the hardcoded sk_{sign}, initialize **T** with the signature of c_1, i.e., $\mathsf{Sign}(sk_{\mathrm{sign}}, c_1)$.

9. Output the registers **X**,**Y**,**Z** and **T**.

In the ith round of the protocol, V' gets as input four registers **X**, **Y**, **Z** and **T** and does the following:

1. Measure the register **X** to obtain α_i.

2. Check if the state held in **Y** is $|S_{i-1}\rangle\langle S_{i-1}|$. That is, perform the projective measurement $\{ \underbrace{|S_{i-1}\rangle\langle S_{i-1}|}_{\mathrm{outcome=0}}, \underbrace{I - |S_{i-1}\rangle\langle S_{i-1}|}_{\mathrm{outcome=1}} \}$ on the register **Y**. This is possible since V' has the description of S_i hardwired in its code.

3. If the measured outcome is 1, abort.

4. Using the hardcoded vk_{sign}, check if **T** contains the signature of the value in register **Z**. If the verification does not pass, abort.

5. Using the hardcoded sk_{enc}, decrypt **Z** leading to value γ_{i-1}

6. Run the ith round of V on α_i with internal state γ_{i-1} a to obtain β_i. Abort if V aborts.

7. Re-initialize register **X** with $|\beta_i\rangle\langle\beta_i|$.

8. Re-initialize the register $\mathbf{Y}$ with a new M-qubit subspace state $|S_i\rangle\langle S_i|$ in the memory.

9. Using the hardcoded sk_{enc}, re-initialize the register $\mathbf{Z}$ with the $c_i = \mathsf{Enc}(sk_{\mathrm{enc}}, \gamma_i)$, where γ_i is the state of V after the answer.

10. Using the hardcoded sk_{sign}, initialize $\mathbf{T}$ with the signature of c_i, i.e., , $\mathsf{Sign}(sk_{\mathrm{sign}}, c_i)$.

11. Output the registers $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ and $\mathbf{T}$.

Note that the quantum memory of V' takes M qubits.

Let $\mathscr{C}(V') = (\Phi_1, \cdots, \Phi_k)$ be the quantum channels corresponding to the operations of V' as defined in Section 3.1. Since Π satisfies fully black-box (M, f)-space bounded QZK, there exists a QPT simulator $\mathsf{Sim}^{\mathscr{C}(V')}$, taking space at most $p(M)$ qubits, simulating the interaction of V' with the honest prover P. In particular, $\mathsf{Sim}^{\mathscr{C}(V')}$ does not abort with probability $1 - \mathsf{negl}(\lambda)$, in which case it outputs $|S_k\rangle$.

Helpful definitions. We state some useful definitions below.

Definition 5.2 (State-successful queries). *A query $\rho_{\mathbf{X} \otimes \mathbf{Y} \otimes \mathbf{Z} \otimes \mathbf{T}}$ made by Sim to Φ_i, for some $i \in [k]$, is said to be* state-successful *if $\Phi_i(\rho_{\mathbf{X} \otimes \mathbf{Y} \otimes \mathbf{Z} \otimes \mathbf{T}})$ does not abort on step 3.*

In particular, a state-successful query made by Sim could lead to abort if the other registers $\mathbf{X}, \mathbf{Z}$ and $\mathbf{T}$ are not of a specific form.

Definition 5.3 ((Non-)abort queries). *A query $\rho_{\mathbf{X} \otimes \mathbf{Y} \otimes \mathbf{Z} \otimes \mathbf{T}}$ made by Sim to Φ_i, for some $i \in [k]$, is said to be* non-abort *if $\Phi_i(\rho_{\mathbf{X} \otimes \mathbf{Y} \otimes \mathbf{Z} \otimes \mathbf{T}})$ does not abort. Similarly, we say that it is an abort query if $\Phi_i(\rho_{\mathbf{X} \otimes \mathbf{Y} \otimes \mathbf{Z} \otimes \mathbf{T}})$ aborts.*

Note that a non-abort query is also a state-successful query.

Definition 5.4 (Successful simulation). *We say that Sim is* successful, *if it does not abort.*

Helpful Claim. We now state an important claim regarding the properties of queries made by Sim to $\Phi_1, ..., \Phi_k$.

Claim 5. *Except with $\mathsf{negl}(\lambda)$ probability, the following events do not occur:*

1. *for some i, Sim has a state-successful query to Φ_i without having a non-abort query to Φ_{i-1};*

2. *for some $i \leq j$, Sim has a state-successful query to Φ_i after having a state-successful query to Φ_j.*

3. *for some i, Sim is successful without a non-abort query to Φ_i.*

4. *for some i, Sim has a non-abort query to Φ_i where the input value on register $\mathbf{Z}$ is different from the output value on register $\mathbf{Z}$ of a non-abort query to Φ_{i-1}.*

Main Claim. We defer the proof of Claim 5 to Section 5.1 and we finish now the proof of Lemma 5.1 by showing that $\mathcal{L}$ is in BQP.

Claim 6. $\mathcal{L}$ *is in BQP.*

Proof. We start by showing that there exists a prover $\tilde{P}$ that runs in expected polynomial time and makes the verifier V accept any instance in $\mathcal{L}$ with probability $1 - \mathsf{negl}(\lambda)$.

Intuitively, $\tilde{P}$ will execute Sim and simulate the queries to V' by exchanging messages with V. Henceforth, we will assume that the registers $\mathbf{X}, \mathbf{Z}$ and $\mathbf{T}$ are measured in the computational basis just before making queries to Φ_i: this does not affect the analysis since they are anyway measured in the channel Φ_i.

In more detail, $\tilde{P}$ proceeds as follows: $\tilde{P}$ chooses random subspaces S_i for $0 \leq i \leq k$, samples a secret-key sk_{enc} corresponding to a symmetric-key encryption scheme and samples a secret-key-verification key pair $(sk_{\mathrm{sign}}, vk_{\mathrm{sign}})$ corresponding to a digital signature scheme.

$\tilde{P}$ then runs Sim on input $|S_0\rangle$. For every query of Sim to Φ_i, $i > 1$, $\tilde{P}$ answers the query with abort. For a query made to Φ_1, on input $|\alpha_1\rangle\langle\alpha_1|_{\mathbf{X}} \otimes \rho_{1\mathbf{Y}}$, $\tilde{P}$ first checks if ρ_1 corresponds to $|S_0\rangle$ by measuring ρ_1 with respect to $\{|S_0\rangle\langle S_0|, I - |S_0\rangle\langle S_0|\}$. If the check fails, $\tilde{P}$ responds to the query with an abort and continues the execution of $\tilde{P}$ with the same oracle responses as before. Otherwise, $\tilde{P}$ will do the following:

1. Send the message α_1 to the verifier,

2. Receive the message β_1 from the verifier,

3. Compute $c_1 = \mathsf{Enc}(sk_{\mathrm{enc}}, 0)$ and $t_1 = \mathsf{Sign}(sk_{\mathrm{sign}}, c_1)$

4. Answer the query from Sim with $|\beta_1\rangle\langle\beta_1|_{\mathbf{X}} \otimes |S_1\rangle\langle S_1|_{\mathbf{Y}} \otimes |c_1\rangle\langle c_1|_{\mathbf{Z}} \otimes |t_1\rangle\langle t_1|_{\mathbf{T}}$,

5. Set $r = 2$ and continues executing Sim, simulating the queries to $\Phi_1, ..., \Phi_k$ as described below.

For every query of Sim to Φ_i, for $i \neq r$, $\tilde{P}$ answers the query with abort. For a query made to Φ_r, on input $|\alpha_r\rangle\langle\alpha_r|_{\mathbf{X}} \otimes \rho_{r\mathbf{Y}} \otimes |c_r'\rangle\langle c_r'|_{\mathbf{Z}} \otimes |t_r'\rangle\langle t_r'|_{\mathbf{T}}$, $\tilde{P}$ first checks if ρ_r corresponds to $|S_{r-1}\rangle$ by measuring ρ_r with respect to $\{|S_{r-1}\rangle\langle S_{r-1}|, I - |S_{r-1}\rangle\langle S_{r-1}|\}$. We now consider the following three cases: First, if the check fails, $\tilde{P}$ answers the query with abort and continues the execution of Sim. Secondly, if the check succeeds and $c_{r-1} \neq c_r'$ then $\tilde{P}$ aborts. Finally, if the check succeeds and the signature t_r' is a correct signature for c_r' then $\tilde{P}$ will do the following:

1. Send the message α_r to the verifier,

2. Receive the message β_r from the verifier,

3. Compute $c_r = \mathsf{Enc}(sk_{\mathrm{enc}}, 0)$ and $t_r = \mathsf{Sign}(sk_{\mathrm{sign}}, c_r)$ (using fresh randomness),

4. Answer the query from Sim with $|\beta_r\rangle\langle\beta_r|_{\mathbf{X}} \otimes |S_r\rangle\langle S_r|_{\mathbf{Y}} \otimes |c_r\rangle\langle c_r|_{\mathbf{Z}} \otimes |t_r\rangle\langle t_r|_{\mathbf{T}}$,

5. Increment r by 1, i.e., set $r = r + 1$, and continue executing Sim by answering the queries with the updated value of r.

We now prove that $\tilde{P}$ makes V accept any instance $x \in \mathcal{L}$ with $1 - \mathsf{negl}(\lambda)$ probability. For that we consider the following hybrids:

$\underline{\mathsf{H}_0}$: The execution of Sim with access to $\Phi_1, ..., \Phi_k$ that come from V' defined above.

$\underline{\mathsf{H}_1}$: The execution of a simulator Sim' which has the same behaviour as Sim, except Sim' aborts if any of the events of the items of Claim 5 happen.

From Claim 5, the events happen with probability at most $\mathsf{negl}(\lambda)$ and therefore the output of H_0 and H_1 are statistically close.

$\underline{\mathsf{H}_2}$: In this hybrid, we consider the interaction of Sim' with access to $\Phi_1', ..., \Phi_k'$, which are controlled by a stateful party P'. At the beginning, for all $j > 1$, P' answers Φ_j' with an abort. On the query to Φ_1', P' will store the value of the output registers $\mathbf{Z}$ and $\mathbf{T}$, which we denote by c_1 and t_1, and replace them by $d_1 = \mathsf{Enc}(sk_{\mathrm{enc}}, 0)$ and its signature $\mathsf{Sign}(sk_{\mathrm{sign}}, d_1)$. P' will set $r = 2$ and starts answering the queries as below.

On a query to Φ_j, for $j \neq r$, P' answers with an abort. On a query to Φ_r, P' works as follows. P' will check if the content of input register $\mathbf{Z}$ is equal to d_{r-1}. If it is not, P' answers with abort. Otherwise, P' re-initializes the contents of the input registers $\mathbf{Z}$ and $\mathbf{T}$ with c_{r-1} and t_{r-1}. Then P' execute the original channel of Φ_r. P' will store the value of the output registers $\mathbf{Z}$ and $\mathbf{T}$, which we denote by c_r and t_r, and replace them by $d_r = \mathsf{Enc}(sk_{\mathrm{enc}}, 0)$ and its signature $\mathsf{Sign}(sk_{\mathrm{sign}}, d_r)$. P' increments r and answers the queries as described above with the updated value of r

Given the IND-CPA security of the encryption scheme, the output of H_1 and H_2 are computationally close.

$\underline{\mathsf{H}_3}$: We consider the interaction of Sim' with the prover $\tilde{P}$ of the protocol.

The output of the simulator in H_2 and H_3 are the same, since Sim' does not have access to any value that depends on the internal state of V in H_2, and $\tilde{P}$ simulates all operations of V'.

We show now that the successful output of the simulator in H_3 implies that V accepts in the protocol. The key observation is that, as described above, if the simulator does not abort, any state-successful query to Φ_k is also non-abort. But by construction, this is only possible if V accepts in the protocol. Therefore, since Sim does not abort with probability at least $1 - \mathsf{negl}(\lambda)$, V accepts with the same probability when interacting with $\tilde{P}$ on a yes-instance.

We also notice that for any no-instance x, by the soundness of the protocol, we have that such a $\tilde{P}$ makes V accept with probability at most $\mathsf{negl}(\lambda)$.

Finally, we need to discuss the runtime of $\tilde{P}$. So far, we have that there is a polynomial q such that the expected runtime of $\tilde{P}$ if $q(\lambda)$. However, we can consider a prover $\tilde{P}'$ that terminates the execution of $\tilde{P}$ after $q(\lambda)^2$ steps and in this case, the probability that $\tilde{P}'$ convinces the honest verifier is still a constant.. In this case, both $\tilde{P}'$ and V runs in polynomial-time, which can be simulated by a BQP algorithm, which finishes the proof. $\qquad\square$

$\hfill\square$

5.1 Proof of Claim 5

We prove each item of Claim 5 in a separate lemma below.

Lemma 5.5. *Except with* $\mathsf{negl}(\lambda)$ *probability, for no* i, Sim *has a state-successful query to* Φ_i *without having a non-abort query to* Φ_{i-1};

Proof. We start by proving this claim assuming that Sim runs in strict polynomial time, and then we generalize to the case where Sim runs in expected polynomial time.

Let J be the event where Sim has a state-successful query to Φ_i, for some i, without having a non-abort query to Φ_{i-1}.[8]

Let us define J_i the event where Sim has a successful query to Φ_i without having a non-abort query to Φ_{i-1}. Notice that, $J = \bigvee_{0 \le i \le k} J_i$, and by union bound

$$\Pr[J] \le \sum_{0 \le i \le k} \Pr[J_i], \tag{2}$$

and we follow by showing that for any i

$$\Pr[J_i] = \mathsf{negl}(\lambda), \tag{3}$$

which implies Equation (2) and the proofs this lemma.

We prove Equation (3) by contradiction, so let us assume that there exists some polynomial p such that $\Pr[J_i] > 1/p(\lambda)$. We show that we can construct an adversary $\mathcal{A}_i$ that is able to clone a random subspace state $|A\rangle$ with polynomially many queries to the verification oracle, contradicting Lemma 2.8.

$\mathcal{A}_i$ works as follows: $\mathcal{A}_i$ receives some state $|A\rangle$ and oracle access to U_A (as described in Section 2.5). $\mathcal{A}_i$ keeps $|A\rangle$ aside and simulates Sim with oracle access to the channels $\Phi'_1, ..., \Phi'_k$ that we define now. For $j \notin \{i-1, i\}$, we have $\Phi'_j = \Phi_j$. Whenever Sim queries Φ'_{i-1}, $\mathcal{A}_i$ runs Φ_{i-1} and if the answer is non-abort, then $\mathcal{A}_i$ aborts. For Φ'_i, $\mathcal{A}_i$ uses the oracle U_A to verify the subspace state and if it passes, then $\mathcal{A}_i$ outputs the result of the query and the original state $|A\rangle$. Otherwise $\mathcal{A}_i$ answers the query to Sim with an abort. If Sim terminates before a state-successful query to Φ'_i, $\mathcal{A}_i$ aborts.

We argue now that $\Pr[J_i] > 1/p(\lambda)$ implies that $\mathcal{A}_i$ is able to output two copies of $|A\rangle$ with probability at least $1/p(\lambda)$ with polynomially-many oracle calls to U_A. Notice that whenever $\mathcal{A}_i$ does not abort, it outputs $|A\rangle^{\otimes 2}$ as desired: one of the copies is guaranteed to be $|A\rangle$ since it was received by the challenger in the quantum money scheme whereas the second copy passed verification test using U_A, and thus by Lemma 2.7 it must be on state $|A\rangle$. In this case, we need to prove the probability that $\mathcal{A}_i$ does not abort.

Let us define the following two events:

E_1 : Sim terminates before a successful query to Φ_i;

E_2 : Sim queries Φ_{i-1} with non-abort answer before querying Φ_i

Let also E be the event on which $\mathcal{A}_i$ aborts. By construction we have that $E = E_1 \vee E_2$. Moreover, we have that $E_j \subseteq \overline{J_i}$ for $j \in \{1, 2\}$, and therefore

$$\Pr[E] \le \Pr[\overline{J_i}] = 1 - \Pr[J_i] \le 1 - 1/p(\lambda).$$

---

[8]As defined in the proof of Lemma 5.1, a query is state-successful if it passes the subspace state check of Φ_i.

It follows then that the probability that $\mathcal{A}_i$ does not abort (and outputs $|A\rangle^{\otimes 2}$) is at least $1/p(\lambda)$ with polynomially-many queries to U_A, which is a contradiction to Lemma 2.8 and this finishes the proof of Equation (3).

Finally, to finish the proof, we need to consider the case the expected runtime of Sim $q(\lambda)$, for some polynomial q. Let us suppose that there exists some non-zero polynomial p such that $\Pr[J_i] \geq 1/p(\lambda)$.

In this case, we can consider a simulator Sim′ that executes Sim but aborts if the runtime of Sim is greater than $q(\lambda)p(\lambda)$. Notice that Sim′ aborts with probability at most $\frac{1}{q(\lambda)p(\lambda)}$ and we have that $\Pr[J_i] > 1/p(\lambda)$ holds for Sim′, such that $\Pr[J_i] \geq 1/p(\lambda) - \frac{1}{q(\lambda)p(\lambda)} = \frac{1}{\text{poly}(\lambda)}$, which is a contradiction. □

Lemma 5.6. *Except with* $\mathsf{negl}(\lambda)$ *probability, for no* $i \leq j$, *Sim has a state-successful query to* Φ_i *after having a state-successful query to* Φ_j.

Proof. As in Lemma 5.5, we prove the lemma for strict polynomial-time Sim and the extension to expected polynomial-time Sim follows as before.

We first prove it for $i = j$ and we then generalize it for $i \leq j$. Let B_i be the event where Sim has two state-successful queries to Φ_i. Our goal now is to show that for every polynomial p,

$$\Pr[B_i] < i \cdot \frac{1}{p(\lambda)}, \tag{4}$$

which implies Lemma 5.6 by a union bound on $\Pr\left[\bigvee_{1 \leq i \leq k} B_i\right]$. We prove now Equation (4) by induction on i.

Let us start with the base case of Equation (4), where $i = 1$. Our proof follows by contradiction. Therefore, let us suppose that there exists some polynomial p such that $\Pr[B_1] \geq 1/p(\lambda)$. We show then that we can construct an adversary $\mathcal{A}_1$ that is able to clone a random subspace state $|A\rangle$ with polynomially many queries to the verification oracle, contradicting Lemma 2.8.

$\mathcal{A}_1$ receives some state $|A\rangle$ and oracle access to U_A, and it proceeds as follows: $\mathcal{A}_1$ sets $q = 0$ and simulates Sim with side information $|A\rangle$ and oracle access to $\Phi'_1, ..., \Phi'_k$, where $\Phi'_j = \Phi_j$ for $j > 1$. For Φ'_1, $\mathcal{A}_1$ uses the oracle U_A to verify the quantum state. If the test does not pass, $\mathcal{A}_1$ answers the query with abort. If the test passes and $q = 0$, $\mathcal{A}_1$ stores the output of U_A on some register $\mathbf{M}$, sets $q = 1$ and continues the simulation of Sim. If the test passes and $q = 1$, $\mathcal{A}_1$ outputs the state returned from the second query to U_A, along with the state stored in register $\mathbf{M}$. If Sim terminates, then $\mathcal{A}_1$ aborts.

We argue now that if $\Pr[B_1] \geq 1/p(\lambda)$, then $\mathcal{A}_i$ is able to output two copies of $|A\rangle$ with probability at least $1/p(\lambda)$ with polynomially many queries to U_A. Notice that whenever $\mathcal{A}_i$ does not abort, it outputs $|A\rangle^{\otimes 2}$ as desired, since both both of the states passed the verification test using U_A (by Lemma 2.7). In this case, we need to bound the probability that $\mathcal{A}_i$ does not abort.

Let F be the event when $\mathcal{A}_1$ aborts, which happens if Sim terminates without two state-successful queries to Φ'_i.

Notice that $F \subseteq \overline{B_1}$, and therefore

$$\Pr[F] \leq \Pr[\overline{B_1}] = 1 - \Pr[B_1] \leq 1 - 1/p(\lambda).$$

It follows then that the probability that $\mathcal{A}_1$ does not abort (and outputs $|A\rangle^{\otimes 2}$) is at least $1/p(\lambda)$, which contradicts Lemma 2.8 and therefore we have Equation (4) for $i = 1$.

We assume now that Equation (4) works for $i-1$, and we prove it for i. The proof is similar to the base case, but we present here for completeness. We prove it again by contradiction. Therefore, let us suppose that $\Pr[B_i] \geq i \cdot p(\lambda)$, for some polynomial p. Notice that

$$
\begin{aligned}
i \cdot p(\lambda) &\leq \Pr[B_i] \\
&= \Pr[B_i \wedge \overline{B_{i-1}}] + \Pr[B_i | B_{i-1}] \Pr[B_{i-1}] \\
&< \Pr[B_i \wedge \overline{B_{i-1}}] + (i-1) \cdot p(\lambda),
\end{aligned}
$$

where the inequality holds by the induction hypothesis. Therefore, our assumption implies that

$$
\Pr[B_i \wedge \overline{B_{i-1}}] \geq p(\lambda). \tag{5}
$$

We show then that if Equation (5) holds, then we can construct an adversary $\mathcal{A}_i$ that is able to clone a random subspace state $|A\rangle$ with polynomially many queries to the verification oracle, contradicting Lemma 2.8.

$\mathcal{A}_i$ receives the state $|A\rangle$ and oracle access to U_A, and it proceeds as follows: $\mathcal{A}_i$ sets $q_{i-1} = q_i = 0$, keeps $|A\rangle$ on a register $\mathbf{N}$ aside and simulates Sim on $\Phi_1', ..., \Phi_k'$, where $\Phi_j' = \Phi_j$ for $j \notin \{i-1, i\}$. For Φ_{i-1}', if $q_{i-1} = 1$, $\mathcal{A}_i$ aborts. Otherwise, $\mathcal{A}_i$ simulates Φ_{i-1}, and if it does not abort, $\mathcal{A}_i$ sets $q_{i-1} = 1$ and stores the quantum state $|S_{i-1}\rangle$ returned by Φ_{i-1} and replaces it by the given copy of $|A\rangle$. For Φ_i', $\mathcal{A}_i$ uses the oracle to verify the quantum state. If it does not pass the test, $\mathcal{A}_i$ answers the query to Φ_i' with abort. If the test passes, $\mathcal{A}_i$ has a different behaviour depending on the value of q_i

- If the test passes and $q_i = 0$, then $\mathcal{A}_i$ puts the state returned by the oracle U_A on a register $\mathbf{M}$, replaces it by $|S_{i-1}\rangle$, simulates Φ_i and sets $q_i = 1$.

- If the test passes and $q_i = 1$, $\mathcal{A}_i$ outputs the state returned by the oracle U_A along with the state in register $\mathbf{M}$.

If Sim terminates, then $\mathcal{A}_i$ aborts.

We argue now that Equation (5) implies that $\mathcal{A}_i$ is able to output two copies of $|A\rangle$ with probability at least $1/p(\lambda)$ with polynomially many queries. Notice that whenever $\mathcal{A}_i$ does not abort, it outputs $|A\rangle^{\otimes 2}$ as desired, since both of the states passed the verification test using U_A (by Lemma 2.7). In this case, we need to bound the probability that $\mathcal{A}_i$ does not abort.

Let us define the following events:

F_1 : Sim aborts due to two non-abort queries to Φ_{i-1}';

F_2 : Sim terminates without two queries to Φ_i' that pass the verification of the subset state;

Let also F be the event on which $\mathcal{A}_i$ aborts. By construction we have that $F = F_1 \vee F_2$. Notice that $F_1 \subseteq B_{i-1}$ and that $F_2 \subseteq \overline{B_i}$. It follows from Equation (5) that

$$
\Pr[F] \leq \Pr[B_{i-1} \vee \overline{B_i}] = 1 - \Pr[\overline{B_{i-1}} \wedge B_i] \leq 1 - p(\lambda).
$$

Therefore, the probability $\mathcal{A}_i$ does not abort (and outputs $|A\rangle^{\otimes 2}$) is at least $p(\lambda)$, which contradicts Lemma 2.8 and we have Equation (4).

To finish the proof, we have to prove the lemma for $i < j$. Let $C_{i,j}$ be the event where Sim has a state-successful query to Φ_i after having a state-successful query to Φ_j. By defining J to be the

event where Sim has a state-successful query to Φ_i, for some i, without having a non-abort query to Φ_{i-1}, we have that

$$\Pr[C_{i,j}] = \Pr[C_{i,j}|J]\Pr[J] + \Pr[C_{i,j} \wedge \overline{J}] \leq \mathsf{negl}(\lambda) + \Pr[B_j] = \mathsf{negl}(\lambda),$$

where the first inequality follows from Lemma 5.5 and $B_j \subseteq C_{i,j} \wedge \overline{J}$, since $\overline{J}$ implies state-successful query to Φ_j before a state-successful query to Φ_i and $C_{i,j}$ implies a state-successful query to Φ_j after a state-successful query Φ_i. $\qquad\square$

Lemma 5.7. *Except with $\mathsf{negl}(\lambda)$ probability, for no i, Sim is successful without a non-abort query to Φ_i.*

Proof. As in Lemma 5.5, we prove the lemma for strict polynomial-time Sim and the extension to expected polynomial-time Sim follows as before. Notice that we need prove it for $i = k$, since it will imply, along with Lemma 5.5, the statement for every $1 \leq i < k$.

Let E be the event where Sim is successful and does not have a non-abort query to Φ_k. We show that if there exists a polynomial p such that

$$\Pr[E] > \frac{1}{p(\lambda)}, \tag{6}$$

then we can construct an adversary $\mathcal{A}$ that is able to clone a random subspace state $|A\rangle$ with polynomially-many queries to the verification oracle U_A, contradicting Lemma 2.8.

For that, we will use the zero-knowledge property of the protocol that implies that for every distinguisher, the output of $\mathsf{Sim}^{\mathscr{C}(V')}$ is indistinguishable from the output of V' while interacting with the honest prover. In particular, we will consider V' where $S_{i+1} = A$ and distinguisher $\mathcal{D}$, using an oracle to U_A, checks if the output register $\mathbf{Y}$ of Sim is indeed $|A\rangle$ (which is always the case for V').

$\mathcal{A}$ receives some state $|A\rangle$, and it proceeds as follows: $\mathcal{A}$ puts $|A\rangle$ on register $\mathbf{M}$ aside, and simulates Sim with oracle access to the channels $\Phi'_1, ..., \Phi'_k$, where $\Phi'_j = \Phi_j$ for $j \neq k$. Whenever Sim queries Φ'_k, $\mathcal{A}$ uses the oracle simulates Φ_k up to step 6 (where V' runs the original verifier V). If Φ_k did not abort up to this point, then $\mathcal{A}$ aborts. Otherwise, $\mathcal{A}$ answer this query with abort and continues the execution of Sim. If Sim aborts, then $\mathcal{A}$ also aborts. Finally, if Sim terminates successfully, $\mathcal{A}$ outputs the register Y of Sim's output, along with the state $|A\rangle$ stored in $\mathbf{M}$.

Let F be the event where the output register $\mathbf{Y}$ of Sim projects into $|A\rangle$. In order to $\mathcal{A}$ to output two copies of $|A\rangle$ we need that $E \wedge F$ hold.

By the zero-knowledge property, we have that $\Pr[F] > 1 - \mathsf{negl}(\lambda)$. This fact, along with Equation (6), gives us that

$$\Pr[E \wedge F] \geq \Pr[E] + \Pr[F] - 1 \geq \frac{1}{p(\lambda)} - \mathsf{negl}(\lambda) = \frac{1}{\mathrm{poly}(\lambda)}.$$

Such $\mathcal{A}$ contradicts Lemma 2.8, and therefore Equation (6) is false. $\qquad\square$

Lemma 5.8. *Except with $\mathsf{negl}(\lambda)$ probability, for no i, for some i, Sim has a non-abort query to Φ_i where the input value on register $\mathbf{Z}$ is different to the output value on register $\mathbf{Z}$ of a non-abort query to Φ_{i-1}.*

Proof. As in Lemma 5.5, we prove the lemma for strict polynomial-time Sim and the extension to expected polynomial-time Sim follows as before.

Let D be the event where there exists some i such that Sim has a non-abort query to Φ_i where the input value on register **Z** is different to the output value on register **Z** of a non-abort query to Φ_{i-1}.

We show that if there is a polynomial p such that

$$Pr[D] > \frac{1}{p(\lambda)} \tag{7}$$

then we can construct an adversary $\mathcal{A}$ that breaks the digital signature scheme.

$\mathcal{A}$ receives a verification key vk and has oracle access to the Sign_{sk}. $\mathcal{A}$ initializes an empty list L and proceeds as follows: $\mathcal{A}$ simulates Sim on $\Phi'_1, ..., \Phi'_k$, where the query to Φ'_i behaves as follows. If $i = 1$, $\mathcal{A}$ simulates the behaviour of Φ_1. Let c be the content of the output register **Z**. $\mathcal{A}$ adds c to L and reinitialize the register **T** with $\mathsf{Sign}_{sk}(c)$.

On query to Φ'_i for $i > 1$, let m be the content of the input register **Z** and s be the content of the input register **T**. $\mathcal{A}$ checks if s is a valid signature of m and if m does *not* appear in L. If the check passes, then $\mathcal{A}$ outputs (m, s). Otherwise, $\mathcal{A}$ continues the simulation of Φ_i. Let c be the content of the output register **Z**. $\mathcal{A}$ adds c to L and reinitialize the register **T** with $\mathsf{Sign}_{sk}(c)$.

If Sim terminates, then $\mathcal{A}$ aborts.

We notice that whenever $\mathcal{A}$ does not abort, it outputs a pair of message and signature (m, s) such that m was not and input to a signing query, and therefore $\mathcal{A}$ is able to forge a signature for m.

To finish the proof, we need to bound the abort probability of $\mathcal{A}$. We notice that D is exactly the case when $\mathcal{A}$ does not abort, and by Equation (7), this happens with probability at least $\frac{1}{p(n)}$. $\quad\square$

References

[ABG⁺20] Amit Agarwal, James Bartusek, Vipul Goyal, Dakshita Khurana, and Giulio Malavolta. "Post-Quantum Multi-Party Computation in Constant Rounds". In: *arXiv preprint arXiv:2005.12904* (2020) (cit. on pp. 2, 11).

[AC13] Scott Aaronson and Paul Christiano. "Quantum money from hidden subspaces". In: *Theory Comput.* 9 (2013), pp. 349–401. DOI: 10.4086/toc.2013.v009a009. URL: https://doi.org/10.4086/toc.2013.v009a009 (cit. on pp. 9, 10, 16, 17).

[ACP21] Prabhanjan Ananth, Kai-Min Chung, and Rolando L La Placa. "On the concurrent composition of quantum zero-knowledge". In: *Annual International Cryptology Conference*. Springer. 2021, pp. 346–374 (cit. on pp. 2, 4, 5, 11, 18).

[AF93] Yonatan Aumann and Uriel Feige. "One message proof systems with known space verifiers". In: *Annual International Cryptology Conference*. Springer. 1993, pp. 85–99 (cit. on p. 11).

[ALP20] Prabhanjan Ananth and Rolando L La Placa. "Secure quantum extraction protocols". In: *Theory of Cryptography Conference*. Springer. 2020, pp. 123–152 (cit. on pp. 2, 8, 11, 18).

[Aru19] Frank Arute *et al.* "Quantum Supremacy using a Programmable Superconducting Processor". In: *Nature* 574 (2019) (cit. on p. 3).

[BCKM21]   James Bartusek, Andrea Coladangelo, Dakshita Khurana, and Fermi Ma. "One-Way Functions Imply Secure Computation in a Quantum World". In: *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO*. Ed. by Tal Malkin and Chris Peikert. Vol. 12825. 2021, pp. 467–496 (cit. on p. 11).

[BG19]   Anne Broadbent and Alex B Grilo. "Zero-knowledge for QMA from locally simulatable proofs". In: *arXiv preprint arXiv:1911.07782* (2019) (cit. on p. 11).

[BJSW16]   Anne Broadbent, Zhengfeng Ji, Fang Song, and John Watrous. "Zero-knowledge proof systems for QMA". In: *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2016, pp. 31–40 (cit. on p. 11).

[BKS21]   Nir Bitansky, Michael Kellner, and Omri Shmueli. "Post-quantum resettably-sound zero knowledge". In: *Theory of Cryptography Conference*. Springer. 2021, pp. 62–89 (cit. on pp. 2, 4, 18).

[Blu86]   Manuel Blum. "How to prove a theorem so no one else can claim it". In: *Proceedings of the International Congress of Mathematicians*. Vol. 1. Citeseer. 1986, p. 2 (cit. on pp. 10, 14).

[BS20]   Nir Bitansky and Omri Shmueli. "Post-quantum zero knowledge in constant rounds". In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. 2020, pp. 269–279 (cit. on pp. 2, 4, 11, 18).

[CCLY21a]   Nai-Hui Chia, Kai-Min Chung, Xiao Liang, and Takashi Yamakawa. "Post-quantum simulatable extraction with minimal assumptions: black-box and constant-round". In: *arXiv preprint arXiv:2111.08665* (2021) (cit. on pp. 5, 18).

[CCLY21b]   Nai-Hui Chia, Kai-Min Chung, Qipeng Liu, and Takashi Yamakawa. "On the Impossibility of Post-Quantum Black-Box Zero-Knowledge in Constant Round". In: *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2021, pp. 59–67 (cit. on pp. 3, 4, 6, 11, 18).

[CCY21]   Nai-Hui Chia, Kai-Min Chung, and Takashi Yamakawa. "A black-box approach to post-quantum zero-knowledge in constant rounds". In: *Annual International Cryptology Conference*. Springer. 2021, pp. 315–345 (cit. on pp. 2–6, 11, 18).

[CMSZ22]   Alessandro Chiesa, Fermi Ma, Nicholas Spooner, and Mark Zhandry. "Post-quantum succinct arguments: breaking the quantum rewinding barrier". In: *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2022, pp. 49–58 (cit. on pp. 3, 5, 6).

[CPT12]   Kai-Min Chung, Rafael Pass, and Wei-Lung Dustin Tseng. "The knowledge tightness of parallel zero-knowledge". In: *Theory of Cryptography Conference*. Springer. 2012, pp. 512–529 (cit. on p. 10).

[DG11]   Ning Ding and Dawu Gu. "Precise time and space simulatable zero-knowledge". In: *International Conference on Provable Security*. Springer. 2011, pp. 16–33 (cit. on p. 10).

[DG12]   Ning Ding and Dawu Gu. "On constant-round precise zero-knowledge". In: *International Conference on Information and Communications Security*. Springer. 2012, pp. 178–190 (cit. on p. 10).

[Die82]   DGBJ Dieks. "Communication by EPR devices". In: *Physics Letters A* 92.6 (1982), pp. 271–272 (cit. on p. 5).

[GK96]   Oded Goldreich and Hugo Krawczyk. "On the composition of zero-knowledge proof systems". In: *SIAM Journal on Computing* 25.1 (1996), pp. 169–192 (cit. on p. 6).

[GLSV21]   Alex B. Grilo, Huijia Lin, Fang Song, and Vinod Vaikuntanathan. "Oblivious Transfer Is in MiniQCrypt". In: *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Ed. by Anne Canteaut and François-Xavier Standaert. Vol. 12697. 2021, pp. 531–561 (cit. on p. 11).

[GMR85]   S. Goldwasser, S. Micali, and C. Rackoff. "The knowledge complexity of interactive proof-systems". In: *STOC*. 1985, pp. 291–304 (cit. on pp. 1, 2, 10).

[GMW86]   Oded Goldreich, Silvio Micali, and Avi Wigderson. "Proofs that yield nothing but their validity and a methodology of cryptographic protocol design". In: *Foundations of Computer Science, 1986., 27th Annual Symposium on*. IEEE. 1986, pp. 174–187 (cit. on p. 10).

[GMW87]   Oded Goldreich, Silvio Micali, and Avi Wigderson. "How to play any mental game". In: *STOC*. 1987 (cit. on pp. 1, 10).

[GO94]   Oded Goldreich and Yair Oren. "Definitions and properties of zero-knowledge proof systems". In: *Journal of Cryptology* 7.1 (1994), pp. 1–32 (cit. on p. 9).

[Gol01]   Oded Goldreich. *The Foundations of Cryptography - Volume 1: Basic Techniques*. 2001 (cit. on pp. 10, 30).

[HSS11]   Sean Hallgren, Adam Smith, and Fang Song. "Classical cryptographic protocols in a quantum world". In: *Annual Cryptology Conference*. Springer. 2011, pp. 411–428 (cit. on p. 11).

[JKMR06]   Rahul Jain, Alexandra Kolla, Gatis Midrijanis, and Ben W Reichardt. "On parallel composition of zero-knowledge proofs with black-box quantum simulators". In: *arXiv preprint quant-ph/0607211* (2006) (cit. on p. 11).

[Joh11]   M. Johnson *et al.* "Quantum annealing with manufactured spins". In: *Nature* 473 (2011) (cit. on p. 3).

[LMS21]   Alex Lombardi, Fermi Ma, and Nicholas Spooner. "Post-quantum zero knowledge, revisited (or: How to do quantum rewinding undetectably)". In: *arXiv preprint arXiv:2111.12257* (2021) (cit. on pp. 2–6, 11, 18).

[MP06]   Silvio Micali and Rafael Pass. "Local zero knowledge". In: *Proceedings of the thirty-eighth annual ACM symposium on Theory of Computing*. 2006, pp. 306–315 (cit. on p. 10).

[Nao91]   Moni Naor. "Bit commitment using pseudorandomness". In: *Journal of cryptology* 4.2 (1991), pp. 151–158 (cit. on pp. 14, 15).

[NY89]   Moni Naor and Moti Yung. "Universal one-way hash functions and their cryptographic applications". In: *Proceedings of the twenty-first annual ACM symposium on Theory of computing*. 1989, pp. 33–43 (cit. on p. 30).

[PPS+08]   Omkant Pandey, Rafael Pass, Amit Sahai, Wei-Lung Dustin Tseng, and Muthuramakrishnan Venkitasubramaniam. "Precise concurrent zero knowledge". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2008, pp. 397–414 (cit. on p. 10).

[SCG⁺14] Eli Ben Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. "Zerocash: Decentralized anonymous payments from bitcoin". In: *2014 IEEE symposium on security and privacy*. IEEE. 2014, pp. 459–474 (cit. on p. 1).

[Sho99] Peter W Shor. "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer". In: *SIAM review* 41.2 (1999), pp. 303–332 (cit. on p. 2).

[SPY92] Alfredo De Santis, Giuseppe Persiano, and Moti Yung. "One-message statistical zero-knowledge proofs and space-bounded verifier". In: *International Colloquium on Automata, Languages, and Programming*. Springer. 1992, pp. 28–40 (cit. on p. 11).

[Unr12] Dominique Unruh. "Quantum proofs of knowledge". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2012, pp. 135–152 (cit. on p. 3).

[Unr13] Dominique Unruh. "Everlasting multi-party computation". In: *Annual Cryptology Conference*. Springer. 2013, pp. 380–397 (cit. on p. 5).

[VDG97] Jeroen Van De Graaf. *Towards a formal definition of security for quantum protocols*. Citeseer, 1997 (cit. on p. 5).

[Wat09] John Watrous. "Zero-knowledge against quantum attacks". In: *SIAM Journal on Computing* 39.1 (2009), pp. 25–58 (cit. on pp. 2–6, 11, 12, 18).

[WZ82] William K Wootters and Wojciech H Zurek. "A single quantum cannot be cloned". In: *Nature* 299.5886 (1982), pp. 802–803 (cit. on p. 5).

[Zha21] Mark Zhandry. "Quantum lightning never strikes the same state twice. Or: quantum money from cryptographic assumptions". In: *J. Cryptology* 34.1 (2021), Paper No. 6, 56. ISSN: 0933-2790. DOI: 10.1007/s00145-020-09372-x. URL: https://doi.org/10.1007/s00145-020-09372-x (cit. on p. 9).

[Zho21] Han-Sen Zhong *et al.* "Quantum computational advantage using photons". In: *Science* (2021) (cit. on p. 3).