Zero-Knowledge Proofs and String Commitments Withstanding Quantum Attacks*

Ivan Damgård[1], Serge Fehr[2], and Louis Salvail[1]

[1] BRICS, FICS, Aarhus University, Denmark**
{ivan,salvail}@brics.dk
[2] ACAC***, Department of Computing, Macquarie University, Australia
sfehr@ics.mq.edu.au

Abstract. The concept of zero-knowledge (ZK) has become of fundamental importance in cryptography. However, in a setting where entities are modeled by quantum computers, classical arguments for proving ZK fail to hold since, in the quantum setting, the concept of rewinding is not generally applicable. Moreover, known classical techniques that avoid rewinding have various shortcomings in the quantum setting.

We propose new techniques for building *quantum* zero-knowledge (QZK) protocols, which remain secure even under (active) quantum attacks. We obtain computational QZK proofs and perfect QZK arguments for any NP language in the common reference string model. This is based on a general method converting an important class of classical honest-verifier ZK (HVZK) proofs into QZK proofs. This leads to quite practical protocols if the underlying HVZK proof is efficient. These are the first proof protocols enjoying these properties, in particular the first to achieve perfect QZK.

As part of our construction, we propose a general framework for building unconditionally hiding (trapdoor) string commitment schemes, secure against quantum attacks, as well as concrete instantiations based on specific (believed to be) hard problems. This is of independent interest, as these are the first unconditionally hiding string commitment schemes withstanding quantum attacks.

Finally, we give a partial answer to the question whether QZK is possible in the plain model. We propose a new notion of QZK, *non-oblivious verifier* QZK, which is strictly stronger than honest-verifier QZK but weaker than full QZK, and we show that this notion can be achieved by means of efficient (quantum) protocols.

1 Introduction

Since its introduction by Goldwasser, Micali and Rackoff [16], the concept of *zero-knowledge* (ZK) proof has become a fundamental tool in cryptography. Informally, in a ZK proof of a statement, the verifier learns nothing beyond the validity of the statement. In particular, everything the verifier can do as a result of the interaction with the prover during the ZK proof, the verifier could also do "from scratch", i.e., without interacting with the prover. This is argued by the existence of an efficient *simulator* which produces a simulated transcript of the execution, indistinguishable from a real transcript. ZK protocols exist for any NP language if one-way functions exist [3, 4, 17], also more efficient solutions are known for specific languages like Quadratic-Residuosity [16] or Graph-Isomorphism [17].

From a theoretical point of view, it is natural to ask whether such classical protocols are still secure if cheating players are allowed to run (polynomial time bounded) quantum

* This is the full version of [12].
** BRICS stands for Basic Research in Computer Science (www.brics.dk) and FICS for Foundations in Cryptography and Security, both funded by the Danish Natural Sciences Research Council.
*** Centre for Advanced Computing - Algorithms and Cryptography.

computers. But the question also has some practical relevance: although quantum computers may not be available to the general public in any foreseeable future, even a single large scale quantum computer could be used to attack the security of existing protocols.

To study this question, two issues are important. First, the computational assumption on which the protocol is based must remain true even if the adversary is quantum. This rules out many assumptions such as hardness of factoring or extracting discrete logs [30], but a few candidates still remain, for instance some problems related to lattices or error correcting codes. In general, it is widely believed that quantum one-way functions exist, i.e., functions that are easy to compute classically, but hard to invert, even on a quantum computer.

A second and more difficult question is whether the proof of security remains valid against a quantum adversary. A major problem in this context comes from the fact that in the classical definition of ZK, the simulator is allowed to *rewind* the verifier in order to generate a simulated transcript of the protocol execution. However, if prover and verifier are allowed to run quantum computers, rewinding is not generally applicable, as it was originally pointed out by Van de Graaf [34]. We discuss this in more detail later, but intuitively, the reason is that when a quantum computer must produce a classical output, such as a message to be sent, a (partial) measurement on its state must be done. This causes an irreversible collapse of the state, so that it is not generally possible to reconstruct the original state. Moreover, copying the verifier's state before the measurement is forbidden by the no-cloning theorem. Therefore, protocols that are proven ZK in the classical sense using rewinding of the verifier may not be secure with respect to a quantum verifier. This severe breakdown of the classical concept of ZK in a quantum world is the motivation of this work.

It is well known that rewinding can cause "problems" already in a classical setting. In particular, it has been realized that rewinding the verifier limits the composability of ZK protocols. As a result, techniques have been proposed that avoid rewinding the verifier, for instance the non-black-box ZK technique from [2], or – in the common reference string model – techniques providing concurrent ZK [15, 29, 11], non-interactive ZK [5] or universally-composable (UC) ZK [6, 7, 13] and related models [27]. One might hope that some of these ideas would translate easily to the quantum setting.

However, the non-black box technique from [2] is based on the simulator using the verifier's program and current state to predict its reaction to a given message. Doing so for a quantum verifier will collapse its state when a measurement is done to determine its next message, so it is not clear that this technique will generalize to a quantum setting. The known constructions of UCZK protocols and non-interactive ZK are all based on computational assumptions that are either false in a quantum setting or for which we have no good candidate for concrete instantiations: the most general sufficient assumption is the existence of one-way trapdoor permutations (i.e. as far as we know) but all known candidates are easy to invert on a quantum computer. Regardless of this type of problem, great care has to be taken with the security proof: despite the fact that the simulator in the UC model must not use rewinding, it is **not** true that a security proof in the UC model automatically implies security against quantum adversaries - we discuss this in more details later in the paper. Finally, the technique for concurrent ZK from [11] avoids rewinding the verifier but instead rewinds the prover to prove soundness, leading to similar problems.

Before describing our results, we note that quantum zero-knowledge proof systems were already studied from a complexity theoretic point of view by Watrous in [33]. The proof systems considered there all assume the prover to be computationally unbounded and the zero-knowledge condition is only enforced against honest verifiers. Clearly, these restrictions make those proof systems unsuitable for cryptographic applications. In this paper, we focus on efficient quantum zero-knowledge protocols in a cryptographic setting.

We propose three distinct techniques applicable to an important class of (classical) honest-verifier ZK (HVZK) proofs (in which the verifier is guaranteed to follow the protocol), namely so-called Σ-protocols (3-move public-coin protocols). We convert such protocols into *quantum zero-knowledge* (QZK) proofs, which are ZK (as well as sound) even with respect to (active) quantum attacks. In all cases, the new proof protocol proceeds in three moves like the underlying Σ-protocol, and its overhead in terms of communication is reasonable. To the best of our knowledge, these are the first (practical) zero-knowledge proofs withstanding active quantum attacks.

The first technique assumes the existence of an unconditionally hiding trapdoor string commitment scheme (secure against quantum attacks) and can be proven secure in the common-reference-string (CRS) model. It requires only classical computation and communication and achieves *perfect or statistical* QZK, assuming the underlying Σ-protocol was perfect or statistical HVZK, and is an interactive argument (computationally sound). The communication overhead of the new QZK protocol in comparison with the underlying Σ-protocol is essentially given by communicating and opening one string commitment. The technique directly implies perfect or statistical QZK arguments for NP.

This first approach requires addressing the problem of constructing unconditionally hiding and computationally binding trapdoor string commitment schemes withstanding quantum attacks. This is non-trivial since the classical definition of computational binding cannot be used for a quantum adversary as it was pointed out in [14] with respect to bit commitments and in [10] with respect to string commitments. In fact, it was not even clear how computational binding for a string commitment should be defined. In [10], a computational binding condition was introduced with their application in mind but no concrete instance was proposed.

We propose a new definition of computational binding that is strong enough for our (and other) applications. On the other hand, we propose a generic construction for schemes satisfying our definition based on special-sound Σ-protocols for hard-to-decide languages, and we give examples based on concrete intractability assumptions. Our construction yields the first unconditionally hiding string commitment schemes withstanding quantum attacks, under concrete as well as under general intractability assumptions. Moreover, since our definition implies the one from [10], our schemes can be used to provide secure quantum oblivious transfer.

The second technique assumes the existence of any quantum one-way function and is also secure in the CRS model. It requires classical communication and computation and produces computational QZK interactive proofs for any NP language. It can be efficiently instantiated under more specific complexity assumptions.

The last technique requires no computational assumption and is provably secure in the *plain model* (no CRS). However, it requires quantum computation and communication and does not achieve full QZK but what we call *non-oblivious verifier QZK*. This new notion is weaker than QZK but strictly stronger than honest-verifier QZK (as defined in [33]). Essentially, a non-oblivious verifier may arbitrarily deviate from the protocol but still generates all private and public classical random variables available to the honest verifier according the same distribution. The (quantum) communication complexity of the non-oblivious verifier QZK proof essentially equals the (classical) communication complexity of the underlying Σ-protocol.

The paper is organized as follows. In Sect. 2, we introduce some relevant notations. We also argue why rewinding causes a problem in a quantum setting and why UCZK does not imply QZK. In Sect. 3, we define and construct the unconditionally hiding (trapdoor) commitment schemes used in Sect. 4 for QZK proofs in the common-reference-string model. Finally, the non-oblivious verifier QZK proof in the plain model is presented in Sect. 5.

2 Preliminaries

2.1 Zero-Knowledge Interactive Proofs

The Classical Case: Let $R = \{(x, w)\}$ be a binary relation. Write $L_R = \{x \mid \exists w : (x, w) \in R\}$ for the language defined by R. For $x \in L_R$, any w such that $(x, w) \in R$ is called a *witness (for $x \in L$)*, and we write $W_R(x) = \{w \mid (x, w) \in R\}$ for the set of witnesses for $x \in L$. We assume that the size of the witnesses for $x \in L$ are polynomially bounded by the size of x, and that R is poly-time testable.

An *(interactive) proof* for a language $L = L_R$ is a protocol (P, V) between a probabilistic prover P and a probabilistic poly-time verifier V. Since we are mainly interested in *efficient protocols*, we also require P to be poly-time. P and V have common input x, claimed to be in L by P, and P has additional (private) input $w \in W_R(x)$. As result of the execution, V outputs either `accept` or `reject`, respectively 0 or 1, indicating whether he accepts the proof or not, such that the following two conditions hold. *Completeness*: if $x \in L$ and the protocol is correctly executed, then V outputs `accept` (respectively 0) with probability 1, and *soundness*: if $x \notin L$ then for any (not necessarily poly-time) $\tilde{\mathsf{P}}$, V outputs `accept` with probability at most $\epsilon < 1$, called the *soundness error*. If $\tilde{\mathsf{P}}$ is restricted to be poly-time, then (P, V) is called an *argument* for L (or a *computationally sound* proof).

A *Σ-protocol* for a language $L = L_R$ is a three-move interactive proof (P, V) for L which proceeds as follows: (1) P computes a *first message* a and sends it to V, (2) V chooses a random *challenge* c and sends it to P, and (3) P computes an *answer* z and sends it to V, on which V decides on whether to output `accept` or `reject` (respectively 0 or 1) by applying a (deterministic) predicate verify_x to (a, c, z). Per default, we understand a Σ-protocol to be *unconditionally* sound. Clearly, for a fixed $x \notin L$, the soundness error ϵ of such a Σ-protocol is given by the maximum over all possible first messages a of the fraction of the possible challenges c for a that allow an answer z which is accepted by V. A Σ-protocol is called *special sound* if the soundness-error equals the inverse of the number of possible challenges c, i.e., if for $x \notin L$ any (valid) first message a uniquely defines a challenge c which allows an answer z with $\mathsf{verify}_x(a, c, z) = \texttt{accept}$.

We refer to a Σ-protocol (P, V) for a language L by a triple $(\mathsf{a}, \mathsf{c}, \mathsf{z})$, where we understand a, c and z as the processes of choosing/computing the first message a, the (random) challenge c and the corresponding answer z, respectively, as specified by the protocol (with some input $x \in L$), and we write $a \leftarrow \mathsf{a}$, $c \leftarrow \mathsf{c}$ and $z \leftarrow \mathsf{z}_x(a, c)$, respectively, for the execution of these processes. We stress that when considering a computationally bounded (honest) prover P as we do here the answer z is typically not computed by P as a function of a, c and x (as the notation $z \leftarrow \mathsf{z}_x(a,c)$ might suggest), but rather as a function of the randomness used to generate a, of the challenge c and of a witness $w \in W_R(x)$.

An interactive proof (or argument) is called *zero-knowledge* (ZK) if for every poly-time $\tilde{\mathsf{V}}$ there exists a (expected) poly-time *simulator* S, which takes as input $x \in L$ and outputs a simulated view of $\tilde{\mathsf{V}}$ in the execution of $(\mathsf{P}, \tilde{\mathsf{V}})$ on input x, indistinguishable from the real view. Depending on the flavor of indistinguishability, ZK can be perfect, statistical or computational. An interactive proof is called *honest-verifier* zero-knowledge (HVZK) if the above holds for the given V (rather than any $\tilde{\mathsf{V}}$).

It is known that statistical ZK Σ-protocols only exist for languages $L \in$ co-AM. Most of the well-known Σ-protocols are proof-system for languages that are trivial on a quantum computers. However, some languages like graph isomorphism (i.e. GI) have special sound Σ-protocols and are not known to be trivial on a quantum computer. This is also the case for some recently proposed lattice problems [23]. It is not known whether co-AM can be efficiently recognized by a quantum computer.

The Quantum Case: Quantum interactive proof systems are defined as a simple generalization of their classical counterpart. They were introduced and first studied by Watrous in [31].

Definition 1. *A pair* (P, V) *is said to be a* quantum interactive proof for L with soundness error ϵ *if* $\mathsf{P} = \{\mathsf{P}(x,i)\}_{i=1}^m$ *and* $\mathsf{V} = \{\mathsf{V}(x,i)\}_{i=1}^m$ *are families of quantum circuits such that:*

Circuit Representation: $\mathsf{V}(x,1), \mathsf{P}(x,1), \mathsf{V}(x,2), \mathsf{P}(x,2), \ldots, \mathsf{V}(x,m), \mathsf{P}(x,m)$ *is the circuit representing the interaction between* P *and* V *upon input* x*. Circuits* $\mathsf{V}(x,i)$ *and* $\mathsf{P}(x,i)$ *act upon* V*'s and* P*'s ancilla qubits respectively together with message qubits.* $\mathsf{V}(x,i)$*'s message qubits are provided to* $\mathsf{P}(x,i)$ *(communication from* V *to* P *at round* i*) and* $\mathsf{P}(x,i)$*'s message qubits are provided to* $\mathsf{V}(x,i+1)$ *(communication from* P *to* V *at round* i*).*

Completeness: P *is such that if* $x \in L$ *then* V *accepts* x *with probability* 1*.*

Soundness: *For any* $\tilde{\mathsf{P}}$*, if* $x \notin L$ *then* V *accepts* x *with probability at most* ϵ*. If* $\tilde{\mathsf{P}}$ *is restricted to be of polynomial size then the proof is called an* argument *instead.*

In this paper we adapt the definition of honest-verifier quantum statistical zero-knowledge introduced by Watrous [33] to the case where the verifier is any poly-time quantum algorithm. In [33] and differently than for the classical case, the simulator was given as input the round for which the honest-verifier's view should be generated. The reason is that simulating the honest-verifier's final view does not imply that intermediary views can also be generated. This is a consequence of the no-cloning theorem. When zero-knowledge should hold against any poly-time verifier, the difference between the classical and the quantum cases disappears. The reason being that if for every $\tilde{\mathsf{V}}$ there exists a simulator that generates $\tilde{\mathsf{V}}$'s final view then there certainly exists a simulator S_k that generated $\tilde{\mathsf{V}}_k$'s final view where $\tilde{\mathsf{V}}_k$ behave like $\tilde{\mathsf{V}}$ up to round k before stopping. S_k is a simulator for $\tilde{\mathsf{V}}$'s view immediately after round k.

Let $\mathsf{view}_{\tilde{\mathsf{V}},\mathsf{P}}(x)$ be the reduced state of $\tilde{\mathsf{V}}$'s ancilla and message qubits at the end of the interaction when $(\mathsf{P}, \tilde{\mathsf{V}})$ is run with input x. Similarly to the classical case, quantum statistical zero-knowledge interactive proof systems are defined as follows:

Definition 2. *A (quantum) interactive proof* (P, V) *for a language* L *is called* perfect quantum zero-knowledge (QZK) *if for any poly-time* $\tilde{\mathsf{V}}$*, there exists a poly-time preparable state* σ_x *such that*

$$\|\sigma_x - \mathsf{view}_{\tilde{\mathsf{V}},\mathsf{P}}(x)\|_{tr} = 0\,,$$

where $\|\cdot\|_{tr}$ *denotes the trace-norm distance. If instead* $\|\sigma_x - \mathsf{view}_{\tilde{\mathsf{V}},\mathsf{P}}(x)\|_{tr} \leq \delta(|x|)$ *for a negligible function* δ *then* (P, V) *is called* statistical *QZK; and if* σ_x *is only indistinguishable from* $\mathsf{view}_{\tilde{\mathsf{V}},\mathsf{P}}(x)$ *by any poly-size quantum circuit then* (P, V) *is called* computational QZK.

In the above definition, σ_x corresponds to the output of the poly-time quantum simulator.

2.2 The Problem with Quantum Rewinding

Rewinding a party to a previous state is a common proof technique for showing the security of many different kinds of protocols in the computational model. In general, this technique cannot be applied when the party is modeled by a quantum computer. Originally observed by Van de Graaf [34], this implies that security proofs of many well-established classical protocols do not hold if one party is running a quantum computer even if the underlying assumption under which the security proof holds withstands quantum attacks. As we shall discuss in Sect. 2.3, it also applies to security proofs taking place in models that explicitly prohibit rewinding of the adversary.

Rewinding is in general not possible since taking a snapshot of a quantum memory is tantamount to quantum cloning. Unlike in the classical case, there is no way to copy a quantum memory regardless of what the memory contains. The only generic way to restore a quantum memory requires to re-generate it from scratch. Proceeding that way may not be possible efficiently.

One consequence of the *no quantum rewinding* paradigm is particularly relevant to us. Sequential repetitions of an HVZK Σ-protocol for a language L results in a ZK protocol for L with negligible soundness error. This straightforward construction is not guaranteed to be secure against quantum verifiers as we discuss next.

Let Π_n be n sequential repetitions of a Σ-protocol Π with one-bit challenges. Let S be the HVZK simulator for Π. The simulator S_n for Π_n works as follows:

For $i = 1..n$ do
1. Take a snapshot ρ_i of $\tilde{\mathsf{V}}$'s internal state,
2. Run S to obtain (a_i, c_i, z_i), send a_i to $\tilde{\mathsf{V}}$,
3. Get $\tilde{c}_i$ from $\tilde{\mathsf{V}}$, If $\tilde{c}_i \neq c_i$ Then return $\tilde{\mathsf{V}}$ to state ρ_i and return to Step 2 Else output (a_i, c_i, z_i).

Clearly, challenges of small size make sure that S_n will stop in expected poly-time. Moreover, the final transcript $\{(a_i, c_i, z_i)\}_{i=1}^n$ together with $\tilde{\mathsf{V}}$'s randomness is produced with the same probability distribution than when interaction with the real prover takes place. However, if $\tilde{\mathsf{V}}$ is a quantum machine then the simulator S_n is not a well-defined procedure. Taking a snapshot(i.e. cloning) of $\tilde{\mathsf{V}}$'s internal memory cannot be performed unitarily and independently of ρ_i as in the classical case. It means that for some ρ_i, taking snapshot might not be doable efficiently.

Is it then possible to return $\tilde{\mathsf{V}}$ to a previous state ρ_i without having to take a snapshot? This is not possible even if the challenge space is 1 bit and the initial state of the verifier

ρ_1 can be generated by a constant size quantum circuit. Even though $\tilde{\mathsf{V}}$'s behavior can be modeled by unitary transform during computation and is thus reversible, this is no more the case after $\tilde{c}_i$ is announced. A classical announcement is the outcome of a measurement applied to ρ_i. It maps $\rho_i \mapsto P_{\tilde{c}} U_i \rho_i U_i^\dagger P_{\tilde{c}} / p_{\tilde{c}} = \rho_i(\tilde{c})$ with probability $p_{\tilde{c}} = \mathrm{tr}(P_{\tilde{c}} \rho_i)$ where U_i is unitary and $P_{\tilde{c}} = |\tilde{c}\rangle\langle\tilde{c}|$ is a projector in the computational basis. This mapping is not unitary in general and hence not necessarily reversible. Running U_i backward from state $\rho_i(\tilde{c})$ will only produce an approximation of ρ_i. As the number of rounds increases, quantum rewinding produces states with lower and lower fidelity to the original state.

Rewinding is also often used when proving the computational binding property of commitments as for example the NOVY scheme [25]. The *no quantum rewinding* paradigm also applies in this situation as discussed in [14, 10]. To see why, suppose a string commitment is such that if an adversary can produce two different openings of a commitment then a computational assumption does not hold. This guarantees that a classical committer cannot "change its mind" after having produced a commitment, meaning that there is only one string to which it can open the commitment, since otherwise, by rewinding it, such a committer can be used to produce two different openings of one commitment, which contradicts the computational assumption. If the committer is quantum however, the ability to open a commitment to a string of its choice does not necessarily imply that two different openings can be produced side by side. The problem is similar to the one the simulator is facing in the previous example. Opening a commitment can be seen as the result of a measurement performed by the committer. After one opening has been obtained, rewinding the committer to its state before the opening cannot be done perfectly.

2.3 UCZK Does Not Imply QZK

In [6], Canetti proposes a new framework for defining and proving cryptographic protocols secure: the universal composability (UC) framework. This framework allows to define and prove secure cryptographic protocols as stand-alone protocols, while at the same time guaranteeing security in any application by means of a general composition theorem. The UC security definition essentially requires that the view of any adversary attacking the protocol can be simulated while in fact running an idealized version of the protocol, which essentially consists of a trusted party called *ideal functionality*. The simulation should be indistinguishable for any distinguisher, called *environment*, which may be *on-line*, and provides the inputs and receives the outputs. Furthermore, the UC definition explicitly *prohibits* rewinding of the environment and thus of the adversary (as it may communicate with the environment). This restriction is crucial for the proof of the composition theorem. We refer to [6] for more details.

Since the UC framework forbids rewinding the adversary, it seems that UCZK implies QZK, assuming the underlying computational assumption withstands quantum attacks. This intuition is false in general. The reason being that even though the UC framework does not allow the simulator to rewind the adversary, it is still allowed to use rewinding as a proof-technique in order to show that the simulator produces a "good" simulation. For instance, it is allowed to argue that if an environment can distinguish the simulation from a real protocol execution, then by rewinding the environment together with the adversary one can solve efficiently a problem assumed to be hard. We illustrate this on a concrete example: the

UCZK argument in the common reference string (CRS) model[3] proposed in [18]. Roughly, the construction works as follows. Consider a language $L = L_R$. The CRS consist of two public-keys pk_S and pk_E, the first for a signature scheme S (secure against chosen message attack) and the second for a (semantically secure) encryption scheme E. Furthermore, a one-time signature scheme S^1 is fixed. Now, to prove $x \in L_R$, the prover computes an encryption e of a witness $w \in W_R(x)$ and generates a public-key secret-key pair (pk_{S^1}, sk_{S^1}) for the one-time signature scheme. Then, using a standard witness-indistinguishable interactive proof, he proves that e indeed contains a witness for x *or* that he knows a signature on pk_{S^1} under pk_S. Finally, he signs the transcript of the above proof using the secret-key sk_{S^1} of the one-time signature scheme.

It is proven in [18] that this argument is secure in the UC sense (against a static adversary). In fact it is proven secure in the UC framework with *joint state* [8]. This allows to reuse the CRS but it requires to prove secure (in the UC sense) a *multiple* execution of the protocol (using the same CRS). We briefly recall the security proof. The simulator chooses pk_S and pk_E such that it knows the corresponding secret keys. If for some execution the player acting as verifier is corrupted, then the simulator can simulate the corresponding prover (if not corrupted) without knowing w and without rewinding: he encrypts a default message w' and uses the fact that he can produce a signature on pk_{S^1} under pk_S to do the interactive proof of the or-statement. On the other hand if in some execution the player acting as prover is corrupted and makes the verifier accept the proof for some x, then the simulator (now simulating the corresponding verifier) can extract w from the proof and send it to the ideal functionality without rewinding by decrypting e. This is necessary since otherwise the ideal functionality will not accept the proof (by its specification) and thus the environment can distinguish. It remains to argue that e indeed decrypts to a witness w for x. First note that by the security of the one-time signature scheme, the secret key for the public-key pk_{S^1} used by the corrupted prover must have been generated by the adversary, otherwise the adversary (acting for the corrupted prover) could not have signed the transcript. In particular, pk_{S^1} has not been generated and signed (under pk_S) by the simulator while simulating a prover during some earlier execution. It follows that *rewinding* the adversary's execution of the interactive proof (together with the environment) produces either a (non-interactive) proof that e decrypts to a witness for x or a signature (under pk_S) on pk_{S^1}, which has not been signed previously. The latter, however, contradicts the security of the signature scheme S. It follows e indeed decrypts to a witness w for x.

Note that it is not the simulator who rewinds the adversary, rewinding is only used to show that the simulator works properly. Namely if not, then there exists an algorithm which breaks the signature scheme — by using and rewinding the assumed adversary (and the environment) from the UC framework, which though is completely legitimate in a classical setting. However it is not if the adversary is allowed to be quantum. Therefore, the above UCZK argument is not necessarily QZK.

3 Unconditionally Hiding (Trapdoor) Commitment Schemes

In this section we study and construct classical (trapdoor) commitment schemes secure against quantum attacks. In contrast to quantum commitment schemes, such schemes do

[3] We refer to Sect. 4.1 or the literature for a description of the CRS model.

not require quantum computation (in order to compute, open or verify commitments), but they are guaranteed to remain secure even under quantum attacks. Our construction, which is based on hard-to-decide languages with special-sound Σ-protocols, yields the first unconditionally hiding string commitment schemes withstanding quantum attacks. In Sect. 4, we use these commitments to construct QZK proofs. A further application of our commitment schemes is given in Appendix A, where we show how they give rise to quantumly secure oblivious transfer.

3.1 Defining Security in a Quantum Setting

Informally, by publishing a commitment $C = \mathsf{commit}_{pk}(s, \rho)$ for a random ρ, a *commitment scheme* allows a party to commit to a secret s, such that the commitment C reveals nothing about the secret s (*hiding property*) while on the other hand the committed party can *open* C to s by publishing (s, ρ) *but only to* s (*binding property*).

Formally, a commitment scheme (of the kind we consider) consists of two poly-time algorithms: A key-generation algorithm $\mathcal{G}$ which takes as input the security parameter ℓ and specifies an instance of the scheme by generating a *public-key* pk, and an algorithm commit which allows to compute $C = \mathsf{commit}_{pk}(s, \rho)$ given a public-key pk as well as s and ρ chosen from appropriate finite sets $\mathcal{S}$ and $\mathcal{R}$ (specified by pk). $\mathcal{S}$ is called the *domain* of the commitment scheme. Classically, the hiding property is formalized by the non-existence of a *distinguisher* which is able to distinguish $C = \mathsf{commit}_{pk}(s, \rho)$ from $C = \mathsf{commit}_{pk}(s', \rho')$ with non-negligible advantage, where $s, s' \in \mathcal{S}$ are chosen by the distinguisher and $\rho, \rho' \in \mathcal{R}$ are random. On the other hand, the binding property is formalized by the non-existence of a *forger* able to compute $s, s' \in \mathcal{S}$ and $\rho, \rho' \in \mathcal{R}$ such that $s \neq s'$ but $\mathsf{commit}_{pk}(s, \rho) = \mathsf{commit}_{pk}(s, \rho')$. If the distinguisher respectively the forger is restricted to be poly-time, then the scheme is said to be *computationally* hiding respectively binding, while without restriction on the distinguisher respectively the forger, it is said to be *unconditionally* hiding respectively binding.

In order to define security of such a commitment scheme $(\mathcal{G}, \mathsf{commit})$ in a quantum setting, the (computational or unconditional) hiding property can be adapted in a straightforward manner by allowing the distinguisher to be quantum. The same holds for the *unconditional* binding property, which is equivalent to requiring that every C uniquely defines s such that $C = \mathsf{commit}_{pk}(s, \rho)$ for some ρ. However, adapting the *computational* binding property in a similar manner simply by allowing the forger to be quantum results in a too weak definition. The reason being that in order to prove secure an application of a commitment scheme, which is done by showing that an attacker that breaks the application can be transformed in a black-box manner into a forger that violates the binding property, the attacker typically needs to be rewound, which cannot be justified in a quantum setting by the no-quantum-rewinding paradigm as discussed in Sect. 2.2. The following definition for the computational binding property of a commitment scheme with respect to quantum attacks is strong enough to prove secure applications (as in Sect. 4 and Appendix A) based on the security of the underlying commitment scheme, but it is still weak enough in order to prove the binding property for concrete commitment schemes (see Sect. 3.2 and 3.3).

Let $(\mathcal{G}, \mathsf{commit})$ be a commitment scheme as introduced above, and let $\mathcal{S}$ denote its domain. Informally, we require that it is infeasible to produce a list of commitments and then open (a subset of) them in a certain specified way with a probability significantly greater than expected. We formalize this as follows. Let Q be a predicate of the following form. Q takes three inputs: (1) a non-empty set $A \subseteq \{1, \ldots, N\}$ where N is upper bounded by a polynomial in ℓ, (2) a tuple $\boldsymbol{s}_A = (s_i)_{i \in A}$ with $s_i \in \mathcal{S}$, and (3) an element $u \in \mathcal{U}$ where $\mathcal{U}$ is some finite set; and it outputs $Q(A, \boldsymbol{s}_A, u) \in \{0,1\}$. We do *not* require Q to be efficiently computable. Consider a polynomially bounded quantum forger $\mathcal{F}$ in the following game: $\mathcal{F}$ takes as input pk, generated by $\mathcal{G}$, and announces commitments $C_1, \ldots, C_N$. Then, it is given a random $u \in \mathcal{U}$, and it outputs A, $\boldsymbol{s}_A = (s_i)_{i \in A}$ and $\boldsymbol{\rho}_A = (\rho_i)_{i \in A}$. $\mathcal{F}$ is said to win the game if $Q(A, \boldsymbol{s}_A, u) = 1$ and $C_i = \mathsf{commit}_{pk}(s_i, \rho_i)$ for every $i \in A$. We require that every forger has essentially the same success probability in winning the game as when using an *ideal* (meaning *unconditionally* binding) commitment scheme (where every C_i uniquely defines s_i). In the latter case, the success probability is obviously given by $p_{\text{IDEAL}} = \max_{\boldsymbol{s} \in \mathcal{S}^N} |sat_Q(\boldsymbol{s})|/|\mathcal{U}|$ with $sat_Q(\boldsymbol{s}) = \{u \in \mathcal{U} \mid \exists\, A : Q(A, \boldsymbol{s}_A, u) = 1\}$, where $\boldsymbol{s}_A$ stands for the restriction of $\boldsymbol{s}$ to its coordinates s_i with $i \in A$. In this definition, Q models a condition that must be satisfied by the opened value in order for the opening to be useful for the committer. For each application scenario, such a predicate can be defined.

Definition 3. *A commitment scheme* $(\mathcal{G}, \mathsf{commit})$ *is called* computational Q-binding *if for every predicate Q, every polynomially bounded quantum forger $\mathcal{F}$ wins the above game with probability $p_{\text{REAL}} = p_{\text{IDEAL}} + adv$, where adv, the* advantage *of $\mathcal{F}$, is (negative or) negligible (in ℓ).*

It is not hard to verify that in a classical setting (where $\mathcal{F}$ is allowed to be rewound), the classical computational binding property is equivalent to the above computational Q-binding property. Furthermore, it is rather obvious that the computational Q-binding property for a commitment scheme with domain $\mathcal{S}$ implies the computational Q-binding property for the natural extension of the scheme to the domain $\mathcal{S}^k$ (for any k) by committing componentwise. Note that this desirable preservation of the binding property does not hold for the binding property introduced in [10].

Finally, we define a *trapdoor* commitment scheme[4] as a commitment scheme in the above sense with the following additional property. Besides the public-key pk, the generator $\mathcal{G}$ also outputs a trapdoor τ which allows to break either the hiding or the binding property. Specifically, if the scheme is unconditionally binding, then τ allows to efficiently compute s from $C = \mathsf{commit}_{pk}(s, \rho)$, and if it is unconditionally hiding, then τ allows to efficiently compute commitments C and correctly open them to any s.

3.2 A General Framework

In this section, we propose a general framework for constructing unconditionally hiding and computationally Q-binding (trapdoor) string commitment schemes. For that, consider a language $L = L_R$ and assume that

1. L admits a (statistical) HVZK *special-sound* Σ-protocol $\Pi = (\mathsf{a}, \mathsf{c}, \mathsf{z})$, [5]

[4] Depending on its flavor, a trapdoor commitment scheme is also known as an extractable respectively as an equivocable or a chameleon commitment scheme.

[5] As will become clear, the prover's efficiency in the Σ-protocol does not influence the efficiency of the resulting commitment scheme as far as the committer and the receiver are concerned. An efficient prover is only required if one wants to take advantage of the trapdoor.

2. there exists a poly-time generator $\mathcal{G}_{\mathsf{yes}}$ generating $x \in L$ together with witness $w \in W_R(x)$ (more precisely, $\mathcal{G}_{\mathsf{yes}}$ takes as input security parameter ℓ and outputs $x \in L$ of bit size ℓ and $w \in W_R(x)$), and
3. for all poly-size quantum circuits $\mathcal{D}$ and polynomials $p(\ell) > 0$, if ℓ is large enough then there exists $x_{\mathsf{no}} \notin L$ of bit size ℓ such that for x_{yes} generated by $\mathcal{G}_{\mathsf{yes}}$ (on input ℓ)

$$\big| \Pr\left(\mathcal{D}(x_{\mathsf{yes}}) = \mathtt{yes}\right) - \Pr\left(\mathcal{D}(x_{\mathsf{no}}) = \mathtt{yes}\right) \big| < 1/p(\ell). \tag{1}$$

Note that 3. only requires that for every distinguisher $\mathcal{D}$ it is hard to distinguish a randomly generated yes-instance $x \in L$ from *some* no-instance $x \notin L$, which in particular may depend on $\mathcal{D}$.

Given such L, the construction in Fig. 1 provides an unconditionally hiding trapdoor commitment scheme. We assume that c samples challenge c randomly from $\{0,1\}^t$ for some t.

> $\mathcal{G}$ is given by $\mathcal{G}_{\mathsf{yes}}$, where the generated $x \in L$ is parsed as public key pk and $w \in W_R(x)$ as trapdoor τ. The domain $\mathcal{S}$ is defined to be $\mathcal{S} = \{0,1\}^t$.
>
> commit_{pk}: To commit to $s \in \mathcal{S} = \{0,1\}^t$, use the HVZK simulator for Π to generate (a, c, z). Set $C = (a, s \oplus c)$ to be the commitment for s.
>
> A commitment $C = (a, d)$ is opened to s by announcing the corresponding values c and z, and such an opening is accepted if and only if $s \oplus c = d$ and $\mathsf{verify}_x(a, c, z) = \mathtt{accept}$.

Fig. 1. Trapdoor commitment scheme $(\mathcal{G}, \mathsf{commit})$.

If Π is *special* HVZK, meaning that (a, c, z) can be simulated for a given c, then the commitment scheme can be slightly simplified: (a, c, z) is generated such that $c = s$ and C is simply set to be $C = a$.

Theorem 1. *Under assumption 3., $(\mathcal{G}, \mathsf{commit})$ in Fig. 1 is an unconditionally hiding and computationally Q-binding trapdoor commitment scheme.*

As will become clear from the proof below, if the underlying Σ-protocol Π is *perfect* HVZK, then $(\mathcal{G}, \mathsf{commit})$ is *perfectly* binding in the sense that there exists no distinguisher with *non-zero* advantage, meaning that a commitment C for s is statistically independent of s.

Proof. It is clear that a correct opening is accepted. It is also rather obvious that the scheme is unconditionally hiding: The distribution of (a, c, z) generated by the HVZK simulator is statistically close to the distribution of (a, c, z) generated by the protocol. There, however, c is chosen independently of a. Therefore, a gives essentially no information on c and thus $C = (a, s \oplus c)$ gives essentially no information on s (as $s \oplus c$ acts as a one-time pad). The trapdoor property can be seen as follows. Knowing the trapdoor $\tau = w$, put $C = (a, d)$ where $a \leftarrow \mathsf{a}$ and d is randomly sampled from $\{0,1\}^t$. Given arbitrary $s \in \{0,1\}^t$, compute $c = d \oplus s$ and $z \leftarrow \mathsf{z}_x(a, c)$ using the witness w (and the randomness for the generation of a). It is obvious that (s, c, z) opens C correctly to s.

It remains to show the computational Q-binding property. We show that if there exists a forger $\mathcal{F}$ that can break the Q-binding property of the commitment scheme (without knowing the trapdoor) for some predicate Q according to Definition 3, then there exists a

$\mathcal{D}$: The input is x, either in L or not in L.

1. Invoke $\mathcal{F}$ with public-key $pk = x$ in order to get commitments $C_1, \ldots, C_N$,
2. Pick random $u \in \mathcal{U}$ and announce it to $\mathcal{F}$,
3. $\mathcal{F}$ announces $A \subseteq \{1, \ldots, N\}$ and, for $i \in A$, tries to open C_i to s_i such that $Q(A, \boldsymbol{s}_A, u) = 1$ for $\boldsymbol{s}_A = (s_i)_{i \in A}$,
4. Verify the openings and whether indeed $Q(A, \boldsymbol{s}_A, u) = 1$, if successful then output yes and otherwise no.

Fig. 2. Distinguisher $\mathcal{D}$ for $x \in L$ versus $x \notin L$.

circuit $\mathcal{D}$ that contradicts assumption 3. $\mathcal{D}$ is illustrated in Figure 2 and is quantum if and only if $\mathcal{F}$ is.

Clearly, if x is generated by $\mathcal{G}_{\mathsf{yes}}$ then $pk = x$ is a valid public-key for the commitment scheme with the right distribution and hence $\Pr(\mathcal{D}(x) = \mathtt{yes}) = p_{\text{REAL}} = p_{\text{IDEAL}} + adv$ where adv is $\mathcal{F}$'s advantage. On the other hand, if $x \notin L$, then by the special soundness property of Π, given a there is only one c that allows an answer z such that $\mathsf{verify}_x(a, c, z) = \mathtt{accept}$. Hence, for any C_i there is only one $s_i \in \mathcal{S}$ to which C_i can be successfully opened. Therefore, $\Pr(\mathcal{D}(x) = \mathtt{yes}) \leq p_{\text{IDEAL}}$. If adv is (positive and) non-negligible, then this contradicts 3. □

We would like to point out once more that our definition of the (computational) binding property inherits the following feature. If a commitment scheme with domain $\mathcal{S}$ is computational Q-binding, then its natural extension to a commitment scheme with domain $\mathcal{S}^k$ by committing componentwise (with the same pk) is also computational Q-binding. In particular, any computational Q-binding *bit* commitment scheme gives rise to a computational Q-binding *string* commitment scheme.

3.3 Concrete Instantiations

We propose three concrete languages L which are believed to satisfy requirements 1. to 3. posed in Sect. 3.2 above and which admit HVZK special-sound Σ-protocols. The first is based on a problem from coding theory.

Code-Equivalence Problem L_{CE}: The yes-instances of L_{CE} are pairs of equally-dimensional matrices (G_0, G_1) over a finite field $\mathbb{F}_q$ such that G_0 and G_1 are generator matrices of the same linear code over $\mathbb{F}_q$, up to a permutation of the coordinates. In other words, $(G_0, G_1) \in L_{CE}$ if there exists a permutation matrix P and an invertible matrix S (both of suitable dimension) such that $G_1 = SG_0P$.

It has been shown in [26] that deciding membership for L_{CE} is at least as hard as solving the Graph-Isomorphism problem which is believed to be hard *in the worst case* (even allowing quantum computation), and that there exists an interactive proof for Code-*Non*equivalence (requiring a computationally unbounded prover). We sketch a special-sound perfect HVZK Σ-protocol for Code-Equivalence L_{CE}: The strategy is the same as in the well-known interactive proof for Graph-Isomorphism [17]. The prover computes as first message a random generator matrix H which is code-equivalent to G_0 (and G_1), and answers the random challenge $c \in \{0, 1\}$ by providing P and S such that $H = SG_cP$.

Concerning the generation of yes-instances, it is believed that random self-dual or weakly-self-dual codes give rise to yes-instances which are hard to distinguish from no-instances as required. In combination with Theorem 1, this results in unconditionally (in fact perfectly) hiding and computationally Q-binding trapdoor bit respectively string commitment schemes based on the Code-Equivalence problem. We stress however that this problem is not (yet) very well studied and thus this scheme should be used with caution.

Next, we consider two languages based on lattice problems: the Gap Shortest-Vector problem and the Gap Closest-Vector problem. These problems are so called *promise* problems, in that the no-instances are promised to be "not too close" to the yes-instances.

Gap Shortest-Vector Problem L_{GapSVP_γ}*:* For fixed $\gamma > 1$, yes-instances of L_{GapSVP_γ} are tuples $(\boldsymbol{B}, \delta)$, where $\boldsymbol{B} \subset \mathbb{Z}^m$ is an integer lattice basis and $\delta > 0$ such that there exists a lattice vector $\boldsymbol{v} \neq \boldsymbol{0}$ with $||\boldsymbol{v}|| \leq \delta$. On the other hand, no-instances are guaranteed to satisfy $||\boldsymbol{v}|| > \gamma\delta$. A witness for a yes-instance $(\boldsymbol{B}, \delta)$ is a vector $\boldsymbol{x} \neq \boldsymbol{0}$ such that $||\boldsymbol{B}\boldsymbol{x}|| \leq \delta$ (viewing $\boldsymbol{B}$ as matrix).

Gap Closest-Vector Problem L_{GapCVP_γ}*:* Yes-instances of L_{GapCVP_γ} are triples $(\boldsymbol{B}, \boldsymbol{y}, \delta)$, where $\boldsymbol{B} \subset \mathbb{Z}^m$ is an integer lattice basis, $\boldsymbol{y} \in \mathbb{Z}^m$ and $\delta > 0$ such that there exists a lattice vector $\boldsymbol{v}$ with $||\boldsymbol{v} - \boldsymbol{y}|| \leq \delta$. On the other hand, no-instances are guaranteed to satisfy $||\boldsymbol{v} - \boldsymbol{y}|| > \gamma\delta$. A witness for a yes-instance $(\boldsymbol{B}, \boldsymbol{y}, \delta)$ is a vector $\boldsymbol{x}$ such that $||\boldsymbol{B}\boldsymbol{x} - \boldsymbol{y}|| \leq \delta$ (viewing $\boldsymbol{B}$ as matrix).

Micciancio and Vadhan recently presented special-sound statistical HVZK Σ-protocols for L_{GapSVP_γ} and for L_{GapCVP_γ} for reasonable values of γ's, both with a 1-bit challenge. They also give some proposals of how to generate yes-instances which seem to be hard to distinguish from no-instances. For instance, using results from [1], one can construct lattices such that breaking the computational assumption is at least as hard as solving some lattice problem in the *worst-case*. In combination with Theorem 1, this results in unconditionally hiding and computationally Q-binding trapdoor bit or string commitment schemes based on the Gap Shortest-Vector or Gap Closest-Vector problem. Though also here one should take care: Regev [28] has shown a quantum reduction from $f(n)$-unique-SVP (i.e. unique-SVP with the promise that the shortest vector is shorter by a factor of at least $f(n)$ from all other non-parallel vectors) to the average case subset sum problem where $f(n) = \tilde{\Theta}(n^{2.5})$. Classically, the best known reduction of that type requires $f(n) = \tilde{\Theta}(n^{3.5})$ [21]. This might indicate that lattice problems are easier to solve on a quantum computer.

4 Quantum Zero-Knowledge Proofs

4.1 Common-Reference-String Model

The *common-reference-string* (CRS) model assumes that there is a string σ (honestly) generated according to some distribution and available to all parties from the start of the protocol. In the CRS model, an interactive proof (or argument) is (Q)ZK if there exists a simulator which can simulate the (possibly dishonest) verifier's view of the protocol execution together with a CRS σ having correct joint distribution as in a real execution.

4.2 Efficient QZK Arguments

We show how to convert any HVZK Σ-protocol into a quantum zero-knowledge (QZK) argument. The construction is based on a trapdoor commitment scheme and can be proven secure in the CRS model.

It is actually very simple. P and V simply execute the Σ-protocol, but instead of sending message a in the first move, P sends a *commitment* to a, which he then opens when he sends the answer z to the challenge c in the third move. The zero-knowledge property then follows essentially by observing that the simulator (who knows the trapdoor of the commitment scheme) can cheat in the opening of the commitment. So far, the strategy for the QZK proof is the same as in Damgård's concurrent ZK proof [11]; the proof of soundness however will be different since [11] requires to rewind the prover, which cannot be justified in our case by the no-quantum-rewinding paradigm. In order not to rely on the *special* HVZK property (as introduced and explained in Sect. 3.2), the protocol is slightly more involved than sketched here, though the idea remains.

Let a HVZK Σ-protocol $\Pi = (\mathsf{a}, \mathsf{c}, \mathsf{z})$ for a language $L = L_R$ be given. Let ϵ denote its soundness error. We assume without loss of generality that a and c sample first messages a and challenges c of fixed bit lengths r and t, respectively. Furthermore, let an unconditionally hiding and computationally Q-binding trapdoor commitment scheme $(\mathcal{G}, \mathsf{commit})$ be given (where the knowledge of the trapdoor allows to break the binding property of the scheme). We assume that its domain $\mathcal{S}$ contains $\{0,1\}^{r+t}$. Consider Protocol 1 illustrated in Fig. 3.

Protocol 1: V has input x, claimed to be in L; P has input x and $w \in W_R(x)$.
The CRS is set to be pk where pk is generated by $\mathcal{G}$.

1. P computes $a \leftarrow \mathsf{a}$ and chooses $c_P \leftarrow \mathsf{c}$. Then it commits to the concatenation $a \| c_P$ of a and c_P by $C = \mathsf{commit}_{pk}(a \| c_P, \rho)$, and sends C to V.
2. V chooses $c_V \leftarrow \mathsf{c}$ and sends it to P.
3. P computes $z \leftarrow \mathsf{z}_x(a, c)$ for $c = c_P \oplus c_V$ and sends (a, c_P, ρ) and z to V.
4. V accepts iff $C = \mathsf{commit}(a \| c_P, \rho)$ and $\mathsf{verify}_x(a, c_P \oplus c_V, z) = \mathtt{accept}$.

Fig. 3. QZK proof protocol in the CRS model.

As mentioned above, Protocol 1 can be slightly simplified in case Π is *special* HVZK in that P commits to a (rather than to $a \| c_P$) and computes z with respect to the challenge $c = c_V$ provided by V.

Theorem 2. *Under the assumption that* $(\mathcal{G}, \mathsf{commit})$ *is an unconditionally hiding and computationally Q-binding trapdoor commitment scheme, Protocol 2 is a QZK (quantum) argument for L in the CRS model. Its soundness error is* $\epsilon' = \epsilon + negl$ *where negl is negligible (in the security parameter).*

Concerning the flavor of QZK, Protocol 2 is *computational* QZK if the underlying Σ-protocol Π is computational HVZK, and it is *statistical* QZK provided that Π is statistical or perfect HVZK. In case $(\mathcal{G}, \mathsf{commit})$ is perfectly (rather than unconditionally) hiding, the flavor of QZK of Protocol 2 is exactly given by the flavor of HVZK of Π.

Proof. As mentioned above, the zero-knowledge property is rather straight forward: The simulator generates a public-key for the commitment scheme together with a trapdoor and outputs the public-key as CRS. Then, on input $x \in L$, it generates a commitment C (which he can open to an arbitrary value using the trapdoor) and sends it to $\tilde{\mathsf{V}}$. On receiving c_V from $\tilde{\mathsf{V}}$, the simulator simulates an accepting conversation (a, c, z) for the original Σ-protocol using the HVZK property, it sets $c_P = c \oplus c_V$ and computes ρ such that $C = \mathsf{commit}(a\|c_P, \rho)$ using the trapdoor, and it sends (a, c_P, ρ) and z to $\tilde{\mathsf{V}}$.

For the soundness property, it has to be shown that given a (quantum) prover $\tilde{\mathsf{P}}$, which succeeds in making (honest) V accept the proof for an $x \notin L$ with a probability exceeding ϵ by a non-negligible amount, $\tilde{\mathsf{P}}$ can be used to break the Q-binding property of the commitment scheme for some predicate Q. Fix $x \notin L$. We define Q as follows. $N = 1$, and $\mathcal{U}$ is given by the set of all possible challenges c_V sampled by c. For $s \in \mathcal{S}$ and $u = c_V \in \mathcal{U}$, where s is parsed as $s = a\|c_P$ with $a \in \{0,1\}^r$ and $c_P \in \{0,1\}^t$, we set $Q(\{1\}, s, u) = 1$ if and only if the challenge $c = c_P \oplus c_V$ for the first message a allows an answer z such that $\mathsf{verify}_x(a, c, z) = \mathtt{accept}$. Note that $A = \{1\}$ is the only legitimate choice for A. By construction of Q, making V accept the proof means that $\tilde{\mathsf{P}}$ opens C (correctly) to $a\|c_P$ such that $Q(\{1\}, a\|c_P, c_V) = 1$. Furthermore, $p_{\text{IDEAL}} = \epsilon$. It follows that if $\tilde{\mathsf{P}}$ succeeds in making V accept the proof with probability greater that ϵ by a non-negligible amount, then $\tilde{\mathsf{P}}$ is a forger $\mathcal{F}$ that breaks the Q-binding property of $(\mathcal{G}, \mathsf{commit})$. This completes the proof. □

4.3 QZK Arguments for all of *NP*

Consider a (generic) ZK argument for an *NP*-complete language using (ordinary) unconditionally hiding commitments. For instance, consider the classical interactive proof for Circuit-Satisfiability due to Brassard, Chaum and Crépeau [4]: the prover "scrambles" the wires and the gates' truth tables of the circuit and commits upon it, and he answers the challenge $c = 0$ by opening all commitments and showing that the scrambling is done correctly and the challenge $c = 1$ by opening the (scrambled) wires and rows of the gates' truth tables that are activated by the satisfying input. Following the lines of the proof of Theorem 2 above, it is straightforward to prove that replacing the commitment scheme in this construction by an unconditionally hiding and computationally Q-binding commitment scheme results in a QZK argument in the CRS model for Circuit-Satisfiability, and thus for all languages in *NP*.

4.4 Computational QZK Proofs

We sketch how to construct rather efficient computational QZK proofs for languages that allow (computational) HVZK Σ-protocols based on specific intractability assumptions, as well as computational QZK proofs for all of *NP* based on any quantum one-way function.

Consider any of the languages $L = L_R$ with HVZK Σ-protocol on which the commitment construction from Sect. 3.2 is based, except that we allow the Σ-protocol to be *computational* HVZK. Assume in addition that there is also a generator $\mathcal{G}_{\mathsf{no}}$ that produces no-instances that cannot be distinguished from the yes-instances produced by $\mathcal{G}_{\mathsf{yes}}$.

Then, put a no-instance x_{no} in the reference string. The prover can now prove any statement S that can be proved by an HVZK Σ-protocol Π by using a standard witness-indistinguishable HVZK proof for proving that S is true or $x_{\mathsf{no}} \in L$ [9]. Here, we allow the Σ-protocol Π to be *computational* HVZK, in particular Π might be the Σ-protocol for Circuit-Satisfiability sketched in Sect. 4.3 above but based on an unconditionally binding and computationally hiding commitment scheme (secure against quantum attacks), which can be constructed from any (quantum) one-way function (see below).

This is clearly unconditionally sound, and can be simulated, where the simulator uses a yes-instance x_{yes} in place of x_{no} and uses its witness $w \in W_R(x_{\mathsf{yes}})$ to complete the protocol without rewinding. A distinguisher would have to contradict the HVZK property of one of the underlying Σ-protocols, or the indistinguishability of yes- and no-instances.

This can be instantiated efficiently if we are willing to assume about the coding or lattice problem or some other candidate problem that it also satisfies this stronger version of indistinguishability of yes- and no-instances. But it can also be instantiated in a version that can be be based on any one-way function: First, the (unconditionally binding and computationally hiding) commitment scheme of Naor [24] is also secure against quantum adversaries, and exists if any one-way function exists. So consider the language of pairs (pk, O) where pk is a public-key for the commitment scheme and O is a commitment of 0. This language has a computational HVZK Σ-protocol using generic ZK techniques, driven by Naor's commitments. Furthermore, the set of no-instances (pk, E) where E is a commitment to 1 is easy to generate and hard to distinguish from the yes-instances.

5 Relaxed Honest-Verifier Quantum Proofs

It is a natural question whether QZK proof systems exist without having to rely upon common reference strings. In this section, we answer this question partially. We define a quantum interactive proof system associated to any Σ-protocol. Our scheme is QZK against a relaxed version of honest verifiers that we call non-oblivious. Intuitively, a non-oblivious verifier is a verifier having access to the same classical variables than the honest verifier. We show that any HVZK Σ-protocol can be turned into a non-oblivious verifier QZK proof using quantum communication.

5.1 Quantum Circuits for Σ-Protocols

Assume $L = L_R$ has a classical HVZK Σ-protocol $\Pi = (\mathsf{a}, \mathsf{c}, \mathsf{z})$. We specify unitary transforms $Z_x(a)$, and $T_x(a)$, depending on $a \leftarrow \mathsf{a}$, which implement quantum versions of the computations specified by z and verify. Throughout, we assume without loss of generality that c samples c uniformly from $\{0,1\}^t$ for some t.

The answer $z \leftarrow \mathsf{z}_x(a, c)$ to challenge c when a was announced during the first round can be computed quantumly through some unitary transform $Z_x(a)$ depending upon the initial announcement a. That is, provided quantum registers P and X, we have:

$$Z_x(a) : |c\rangle^P |y\rangle^X \mapsto |c\rangle^P |y \oplus \mathsf{z}_x(a, c)\rangle^X .$$

Similarly, the testing process performed by V can also be executed by a quantum circuit $T_x(a)$ depending on the announcement of a. Transformation $T_x(a)$ stores the output of the verification process in an extra one-qubit register T:

$$T_x(a) : |z\rangle^X |c\rangle^V |t\rangle^T \mapsto |z\rangle^X |c\rangle^V |t \oplus \mathsf{verify}_x(a, c, z)\rangle^T .$$

If $z \leftarrow \mathsf{z}_x(a, c)$ and $\mathsf{verify}_x(a, c, z)$ can be classically computed in polynomial time (given the randomness of the computation of a and a witness $w \in W_R(x)$ for the former), circuits $Z_x(a)$ and $T_x(a)$ can be implemented by poly-size quantum circuits.

5.2 EPR-pairs Based Proofs

The idea behind the protocol is as follows. P chooses $a \leftarrow \mathsf{a}$ and sends the answer to all possible challenges in quantum superposition to V. V then verifies quantumly that all answers in the superposition are correct. In a further step, P convinces V that the state contains the answer to more than one challenge. Since Π is assumed to be special sound, it follows that $x \in L$.

Concretely, P starts by choosing $a \leftarrow \mathsf{a}$ and by preparing t EPR pairs in state:

$$|\Omega_t\rangle^{P,V} = 2^{-t/2} \sum_{c\in\{0,1\}^t} |c\rangle^P |c\rangle^V = 2^{-t/2} \sum_{c\in\{0,1\}^t} |c\rangle^P_\times |c\rangle^V_\times . \tag{2}$$

The two equivalent ways of writing $|\Omega_t\rangle$ shows that it exhibits the same correlation between registers P and V in both the computational and the diagonal bases. This property will be used later in the protocol. Now, P adds an extra register X initially in state $|0\rangle^X$ before applying $Z_x(a)$ upon registers P and X. This results in state,

$$|\psi_a\rangle = 2^{-t/2} \sum_{c\in\{0,1\}^t} Z_x(a)\,|c\rangle^P\,|0\rangle^X \otimes |c\rangle^V = 2^{-t/2} \sum_{c\in\{0,1\}^t} |c\rangle^P\,|z\rangle^X \otimes |c\rangle^V , \tag{3}$$

where every z in the superposition is computed as $z \leftarrow \mathsf{z}_x(a,c)$. P then announces a and sends registers V and X to V allowing him to apply the verification quantum circuit $T_x(a)$ after adding an extra register T initially in state $|0\rangle^T$. That is,

$$\begin{aligned}|\psi_a^T\rangle &= \big(\mathbb{I}_P \otimes T_x(a)\big)|\psi_a\rangle|0\rangle^T = 2^{-t/2} \sum_{c\in\{0,1\}^t} |c\rangle^P \otimes T_x(a)|z\rangle^X|c\rangle^V|0\rangle^T \\ &= 2^{-t/2} \sum_{c\in\{0,1\}^t} |c\rangle^P \otimes |z\rangle^X|c\rangle^V|\mathsf{verify}_x(a,c,z)\rangle^T = |\psi_a\rangle \otimes |0\rangle^T .\end{aligned}$$

V then measures register T in the computational basis and rejects if $|0\rangle^T$ is not observed. Provided P was honest, the test will always be successful by assumption on the original Σ-protocol Π, and the verification process does not affect the state $|\psi_a\rangle$. V then returns register X back to P, who can recover t shared EPR pairs by running $Z_x(a)^\dagger$, the inverse of $Z_x(a)$. Finally, P measures register P in the diagonal basis and announces the outcome to V. V does the same to register V and verifies that the same outcome is obtained. By the properties of EPR pairs (2), it follows that the measurements coincide provided P was honest. A compact description of the protocol is given by Protocol 2 in Fig. 4.

Protocol 2: V has input x, claimed to be in L; P has input x and $w \in W_R(x)$.

1. P computes $a \leftarrow \mathsf{a}$ and prepares the quantum state $|\psi_a\rangle^{P,X,V} = 2^{-t/2}\sum_c |c\rangle^P|z\rangle^X|c\rangle^V$ as in (3) where $z \leftarrow \mathsf{z}_x(a,c)$, and he sends a and the registers X and V to V,
2. V runs the verification circuit $T_x(a)$ and rejects if a non-zero outcome is obtained. If the test was successful then V returns register X to P,
3. P runs $(Z_x(a)^\dagger \otimes \mathbb{I}_V)|\psi_a\rangle = |\Omega_t\rangle^{P,V} \otimes |0\rangle^X$, measures the P register in the diagonal basis and announces the outcome $c_P \in \{0,1\}^t$ to V.
4. V accepts iff register V measured in the diagonal basis produces outcome $c_V = c_P$.

Fig. 4. Non-oblivious verifier QZK proof.

5.3 Soundness

Consider $x \notin L$. We show that in Protocol 2, any prover $\tilde{\mathsf{P}}$ has probability at most 2^{-t} to convince V, given that Π is special sound. Let a be announced by $\tilde{\mathsf{P}}$ at step 1. By the special soundness property of Π, if $\tilde{\mathsf{P}}$ passes the test at step 2. then the state shared between $\tilde{\mathsf{P}}$ and V is of the following form: $|\tilde{\psi}_a\rangle = |\gamma_{a,x}\rangle^{P,X} \otimes |c\rangle^V |0\rangle^T$, where c is the unique challenge that can be answered given the announcement of a. Since after register X has been sent back to $\tilde{\mathsf{P}}$, register V is in pure state, it follows that only one answer is possible when V is measured in the computational basis. That is, $|c\rangle$ is guaranteed to be observed. In other words, $\tilde{\mathsf{P}}$ has no Shannon uncertainty about the outcome obtained when register V is measured in the computational basis. However, V's final test involves a measurement of that same register in the diagonal basis. The following theorem, proven in [22], allows for a simple proof of soundness of our protocol. It gives a tight lower bound on the Shannon uncertainty about the outcome of two measurements in *mutually unbiased* bases applied to any state. Note, two bases B_0 and B_1 of a 2^t-dimensional Hilbert space $\mathcal{H}_{2^t}$ are said to be mutually unbiased if $|\langle v_0|v_1\rangle| = 2^{-t/2}$ for all $|v_0\rangle \in B_0$ and $|v_1\rangle \in B_1$, and it is easy to verify that the computational and the diagonal bases are indeed mutually unbiased.

Theorem ([22]). *Let $|\psi\rangle$ be any quantum state in $\mathcal{H}_{2^t}$ and let B_0 and B_1 be two mutually unbiased bases for $\mathcal{H}_{2^t}$. For $i \in \{0,1\}$, write $p_i(|\psi\rangle)$ for the probability distribution of the observation when measuring $|\psi\rangle$ in basis B_i. Then, $H\big(p_0(|\psi\rangle)\big) + H\big(p_1(|\psi\rangle)\big) \geq t$.*

In the following, let B_0 be the computational basis and let B_1 be the diagonal basis. Since $H\big(p_0(|c\rangle)\big) = 0$, the above theorem implies that $H\big(p_1(|c\rangle)\big) = t$, as t maximizes the entropy. It follows:

Theorem 3. *If $\Pi = (\mathsf{a}, \mathsf{c}, \mathsf{z})$ is a special-sound HVZK Σ-protocol for language $L = L_R$ where c samples in $\{0,1\}^t$, then Protocol 2 is a quantum interactive proof for L with soundness error 2^{-t}.*

It should be mentioned that Π being special sound is not a strict necessary condition for Protocol 2 to be sound. A more careful analysis can handle the case where Π is "not too far away" from special sound. For simplicity, in this paper we only address the case of special sound Σ-protocols.

5.4 Non-Oblivious Verifier Quantum Zero-Knowledge

Classical Σ-protocols with large challenges are not known to be ZK against a dishonest verifier. This is due to the fact that rewinding allows the simulator to succeed only if it has a non-negligible probability to guess the challenge that the verifier will pick. This is true even with respect to verifiers that submit a uniformly distributed challenge $c \in \{0,1\}^t$ and are able to do the verification test as prescribed. To see this, let $\sigma : \{0,1\}^\ell \rightarrow \{0,1\}^\ell$ be a one-way permutation and let us assume for simplicity that $t = \ell$ and a samples a from $\{0,1\}^t$. If $\tilde{\mathsf{V}}$ announces challenge $c = a \oplus \sigma(m)$ for random $m \in \{0,1\}^\ell$ and $a \leftarrow \mathsf{a}$ announced by P as first message, then the simulator must generate (a, c, z, m) since it is part of $\tilde{\mathsf{V}}$'s view. However, the simulator typically can compute a only after having picked c, which means that it has to compute m as $m = \sigma^{-1}(c \oplus a)$. Note that even though $c \oplus a$ is not necessarily uniformly distributed, it seems that the simulator has typically not enough control over the value $c \oplus a$ in order to compute m.

Notice that a verifier $\tilde{\mathsf{V}}$ acting as described above rejects a false statement with the same probability and chooses the challenge c with the same distribution as an honest verifier, yet there is no known efficient simulator for $\tilde{\mathsf{V}}$. In this section we show that Protocol 2 is quantum zero-knowledge provided that $\tilde{\mathsf{V}}$ is non-oblivious of the value c_V needed for the verification at step 4. More generally, we define non-oblivious verifiers the following way:

Definition 4. *A verifier* $\tilde{\mathsf{V}}$ *is said to be* non-oblivious *if it produces the same (public and private) variables as honest* V *according the same distribution.*

As illustrated above, in contrast to an honest verifier a non-oblivious verifier can produce his variables in an arbitrary manner, as long as they are correctly distributed.

In Protocol 2, a non-oblivious verifier $\tilde{\mathsf{V}}$ has access to the string c_V so it can be made available to the simulator. Indeed, this allows to produce a simulation of the interaction between P and $\tilde{\mathsf{V}}$.

Simulator: Input is $x \in L$.

1. Run the honest verifier simulator for Π in order to get triplet (a, c, z), and send a together with the quantum state $|c\rangle|z\rangle$ to $\tilde{\mathsf{V}}$,
2. If $\tilde{\mathsf{V}}$ rejects P then halt, otherwise throw away the state sent by $\tilde{\mathsf{V}}$,
3. Extract c_V using the non-obliviousness of $\tilde{\mathsf{V}}$ and announce $c_P = c_V$.

Fig. 5. Simulator for Protocol 2.

Theorem 4. *Protocol 2 built from a special-sound (statistical/perfect) HVZK Σ-protocol Π is (statistical/perfect) QZK provided* $\tilde{\mathsf{V}}$ *is non-oblivious.*

Proof. Consider the simulator illustrated in Figure 5. Since Π is HVZK it follows that (a, c, z) generated at step 1. has the same probability distribution than selecting $a \leftarrow \mathsf{a}$, $c \leftarrow \mathsf{c}$, and $z \leftarrow \mathsf{z}_x(a, c)$. Because $\tilde{\mathsf{V}}$ does not have access to register P, its view after step 1. in Protocol 2 is $\rho = \sum_{a,c,z} p(a, c, z)|a\rangle|c\rangle|z\rangle\langle z|\langle c|\langle a|$ where $p(a, c, z)$ is the probability distribution induced by a, c, and $\mathsf{verify}_x(a, c)$. The simulator sends $(a, |c\rangle|z\rangle)$ with probability $p(a, c, z)$ which results in the same mixed state ρ. This is a perfect simulation of step 1. The simulation of step 2. is also perfect since during this step only $\tilde{\mathsf{V}}$ sends something back which results in the same state for both the simulation and the protocol since the state prior to this step was simulated perfectly. The simulation for the last step is clearly perfect since $\tilde{\mathsf{V}}$ is non-oblivious. $\square$

A weaker assumption about $\tilde{\mathsf{V}}$'s behavior would be obtained if the only constraint was that $\tilde{\mathsf{V}}$ detects false statements with the same probability as the honest verifier V. Let us say that such a verifier is *verification-enabled.* In general, a verification-enabled verifier $\tilde{\mathsf{V}}$ is not necessarily non-oblivious since in order to verify $\tilde{\mathsf{P}}$'s announcement, c_P does not necessarily have to be determined by $\tilde{\mathsf{V}}$ without P's help. However, it can be shown that for Σ-protocols with challenges of polylogarithmic size, any verification-enabled $\tilde{\mathsf{V}}$ in Protocol 2 is also non-oblivious.

Acknowledgements

The authors are grateful to Claude Crépeau for having introduced the problem to one of us and discussed its relevance. We would also like to thank Jesper Nielsen for enlightening discussions.

References

1. Ajtai, M., *Generating Hard Instances of Lattice Problems*, in 28th Annual Symposium on Theory of Computing (STOC), 1996.
2. Barak, B., *How to Go Beyond the Black-box Simulation Barrier*, in 42th Annual Symposium on Foundations of Computer Science (FOCS), 2001.
3. Brassard, G., and C. Crépeau, *Zero-Knowledge Simulation for Boolean Circuits*, in Advances in Cryptology - CRYPTO 86, Lecture Notes in Computer Science, vol. 263, Springer-Verlag, 1987.
4. Brassard, G., D. Chaum, and C. Crépeau, *Minimum Disclosure Proofs of Knowledge*, JCSS, vol. 37, no. 2, 1988.
5. Blum, M., P. Feldman and S. Micali, *Non-Interactive Zero-Knowledge and Its Applications*, in 20th Annual Symposium on Theory of Computing (STOC), 1988.
6. Canetti, R., *Universally Composable Security: A New Paradigm for Cryptographic Protocols*, in 42th Annual Symposium on Foundations of Computer Science (FOCS), 2001.
7. Canetti, R., and M. Fischlin, *Universally Composable Commitments*, in Advances in Cryptology - CRYPTO 01, Lecture Notes in Computer Science, vol. 2139, Springer-Verlag, 2001.
8. Canetti, R., and T. Rabin, *Universal Composition with Joint State*, in ePrint archive, Report 2002/047, `http://eprint.iacr.org`, 2002.
9. Cramer, R., I. Damgård, and B. Schoenmakers, *Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols*, in Advances in Cryptology - CRYPTO 94, Lecture Notes in Computer Science, vol. 839, Springer-Verlag, 1994.
10. Crépeau, C., P. Dumais D. Mayers and L. Salvail,*Computational Collapse of Quantum State with Application to Oblivious Transfer*, in Advances in Cryptology – TCC 04, Lecture Notes in Computer Science, vol. 2951, Springer-Verlag, 2004.
11. Damgård, I., *Efficient Concurrent Zero-Knowledge in the Auxiliary String Model*, in Advances in Cryptology - EUROCRYPT 00, Lecture Notes in Computer Science, vol. 1807, Springer-Verlag, 2000.
12. Damgård, I.,S. Fehr, and L. Salvail, *Zero-Knowledge Proofs and String Commitments Withstanding Quantum Attacks*, in Advances in Cryptology - CRYPTO 04, Lecture Notes in Computer Science, vol. 3152, Springer-Verlag, 2004.
13. Damgård, I., and J. Nielsen, *Perfect Hiding and Perfect Binding Universally Composable Commitment Schemes with Constant Expansion Factor*, in Advances in Cryptology - CRYPTO 02, Lecture Notes in Computer Science, vol. 2442, Springer-Verlag, 2002.
14. Dumais, P., D. Mayers, and L. Salvail, *Perfectly Concealing Quantum Bit Commitment From Any Quantum One-Way Permutation*, in Advances in Cryptology - EUROCRYPT 00, Lecture Notes in Computer Science, vol. 1807, Springer-Verlag, 2000.
15. Dwork, C., M. Naor, and A. Sahai, *Concurrent Zero-Knowledge*, in 30th Annual Symposium on Theory of Computing (STOC), 1998.
16. Goldwasser, S., S. Micali, and C. Rackoff, *The Knowledge Complexity of Interactive Proof Systems*, in 17th Annual Symposium on Theory of Computing (STOC), 1985.
17. Goldreich, O., S. Micali, and A. Wigderson, *Proofs that Yield Nothing but their Validity, or All Languages in NP Have Zero-Knowledge Proof Systems*, J. ACM., vol. 38, no. 3, 1991.
18. Garay, J., P. MacKenzie, and K. Yang., *Strengthening Zero-Knowledge Protocols using Signatures*, in Advances in Cryptology - EUROCRYPT 03, Lecture Notes in Computer Science, vol. 2656, Springer-Verlag, 2003.
19. Fiat, A., and A. Shamir, *How to Prove Yourself: Practical Solutions to the Identification and Signature Problem*, in Advances in Cryptology - CRYPTO 86, Lecture Notes in Computer Science, vol. 263, Springer-Verlag, 1987.
20. Kitaev, A., and J. Watrous, *Parallelization, Amplification, and Exponential Time Simulation of Quantum Interactive Proof Systems*, in 32nd Annual Symposium on Theory of Computing (STOC), 2000.

21. MICCIANO, D., *Improved Cryptographic Hash Function with Worst-case/Average-case Connection*, in Proc. 34th ACM Symposium on Theory of Computing (STOC), 2002.
22. MAASSEN, H., and J.B.M. UFFINK, *Generalized Entropic Uncertainty Relations*, Phys. Rev. Letters, vol. 60, 1988.
23. MICCIANCIO, D., and S. P. VADHAN, *Statistical Zero-Knowledge Proofs with Efficient Provers: Lattice Problems and More*, in Advances in Cryptology - CRYPTO 03, Lecture Notes in Computer Science, vol. 2729, Springer-Verlag, 2003.
24. NAOR, M., *Bit Commitment Using Pseudorandomness*, Journal of Cryptology, vol. 4, no. 2, 1991.
25. NAOR, M., R. OSTROVSKY, R. VENTKATESAN and M. YOUNG, *Perfect Zero-Knowledge Arguments for NP Using One-Way Permutation*, Journal of Cryptology, vol. 11, no. 2, 1998.
26. PETRANK, E., and R. ROTH, *Is Code Equivalence Easy to Decide?*, in IEEETIT: IEEE Transactions on Information Theory, vol. 43, 1997.
27. PFITZMANN, B., and M. WAIDNER,*Composition and Integrity Preservation of Secure Reactive Systems*, in 7th ACM Conference on Computer and Communications Security, 2000.
28. REGEV, O., *Quantum Computation and Lattice Problems*, in 43rd Annual Symposium on the Foundations of Computer Science (FOCS), 2002.
29. RICHARDSON, R. and J. KILIAN, *On the Concurrent Composition of Zero-Knowledge Proofs*, in Advances in Cryptology - EUROCRYPT 99, Lecture Notes in Computer Science, vol. 1592, Springer-Verlag, 1999.
30. SHOR, P., *Algorithms for Quantum Computation: Discrete Logarithms and Factoring*, in 35th Annual Symposium on Foundations of Computer Science (FOCS), 1994.
31. WATROUS, J., *PSPACE has Constant-Round Quantum Interactive Proof Systems*, in 40th Annual Symposium on Foundations of Computer Science (FOCS), 1999.
32. WATROUS, J., *Succinct Quantum Proofs for Properties of Finite Groups*, Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS), 2000.
33. WATROUS, J., *Limits on the Power of Quantum Statistical Zero-Knowledge*, in 43rd Annual Symposium on the Foundations of Computer Science (FOCS), 2002.
34. VAN DE GRAAF, J., *Towards a Formal Definition of Security for Quantum Protocols*, Ph.D. thesis, Computer Science and Operational Research Department, Université de Montréal, 1997.

A Application of our Commitment Schemes to Oblivious Transfer

In [10], oblivious transfer is shown to be reducible to any (string) commitment scheme with domain $\{0,1\}^{2n}$ satisfying what was introduced as the $\mathcal{G}_m^n$-binding criteria for $\mathcal{G}_m^n$ a class of functions $g : \{0,1\}^{2n} \rightarrow \{0,1\}^m$. The reduction works for any $m \geq f^*(n)$ where $f^*(n)$ is some (specific) function in $O(\log n)$. A $\mathcal{G}_m^n$-binding commitment scheme is such that for all $g \in \mathcal{G}_m^n$ and all polynomial size in n committer $\mathcal{A}$,

$$\sum_{y\in\{0,1\}^m} \sum_{s\in g^{-1}(\{y\})} \Pr\left(\mathrm{Open}_{\mathcal{A}}(s) = \mathtt{ok}\right) < 1 + 1/p(n) \tag{4}$$

for any positive polynomial $p(\cdot)$, where $\Pr\left(\mathrm{Open}_{\mathcal{A}}(s) = \mathtt{ok}\right)$ is the probability that after the committing phase, $\mathcal{A}$ succeeds in opening the commitment to $s \in \{0,1\}^{2n}$. The class of functions $\mathcal{G}_m^n$ contains all $g : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^m$ such that for $x, y \in \{0,1\}^n$, each output bit of $g(x,y)$ is either a bit in x, or in y, or in $x \oplus y$ (see [10] for more details).

In order to show that secure OT can be reduced to the security of Q-binding string commitments, it suffices to show that Q-binding implies $\mathcal{G}_m^n$-binding for some $m \geq f^*(n)$, e.g. for $m = f^*(n)$. We show the contrapositive of that statement. We assume that we have a string commitment scheme that is not $\mathcal{G}_m^n$-binding for $m = f^*(n)$ where n is polynomial in the security parameter ℓ. Let $g \in \mathcal{G}_m^n$ be such that the security criteria expressed in (4) is not satisfied:

$$\sum_{y\in\{0,1\}^m} \sum_{s\in g^{-1}(\{y\})} \Pr\left(\mathrm{Open}_{\mathcal{A}}(s) = \mathtt{ok}\right) \geq 1 + 1/p(n)$$

for some positive polynomial $p(\cdot)$. Define Q as follows: $N = 1, \mathcal{U} = \{0,1\}^m$, and $Q(\{1\}, s, u)$ equals 1 if and only if $g(s) = u$. (Note that $A = \{1\}$ is the only legitimate choice for A.) Then, $|sat_Q(s)| = 1$ for all $s \in \{0,1\}^{2n}$ and therefore $p_{\text{IDEAL}} = 1/2^m$. On the other hand,

$$p_{\text{REAL}} = 2^{-m} \sum_{y \in \{0,1\}^m} \sum_{s \in g^{-1}(\{y\})} \Pr\left(\text{Open}_{\mathcal{A}}(s) = \texttt{ok}\right) \geq 2^{-m}(1 + 1/p(n)),$$

which exceeds p_{IDEAL} by $2^{-m}/p(n)$, which is not negligible since $m \in O(\log n)$. So the commitment scheme is not Q-binding.

Notice that nothing in the above argument depends upon $\mathcal{G}_m^n$ except for the fact that $m \in O(\log n)$. In fact, Q-binding implies $\mathcal{G}_m^n$-binding according (4) for any class of functions $\mathcal{G}_m^n$ provided $m \in O(\log n)$ and n polynomial in ℓ.

Classically, it is unlikely that one can reduce OT to the existence of HVZK Σ-protocols for languages with efficient hard instance generators. The reason being that OT implies secret-key agreement which is unlikely to have its security reduced to such assumption.