Asynchronous Multi-Party Quantum Computation*

Vipul Goyal[1], Chen-Da Liu-Zhang[2]**, Justin Raizes[3], and João Ribeiro[4]

[1] vipul@cmu.edu, Carnegie Mellon University and NTT Research
[2] chen-da.liuzhang@ntt-research.com, NTT Research
[3] jraizes@cs.cmu.edu, Carnegie Mellon University
[4] jlourenc@cs.cmu.edu, Carnegie Mellon University

Abstract. Multi-party quantum computation (MPQC) allows a set of parties to securely compute a quantum circuit over private quantum data. Current MPQC protocols rely on the fact that the network is synchronous, i.e., messages sent are guaranteed to be delivered within a known fixed delay upper bound, and unfortunately completely break down even when only a single message arrives late.

Motivated by real-world networks, the seminal work of Ben-Or, Canetti and Goldreich (STOC'93) initiated the study of multi-party computation for classical circuits over *asynchronous* networks, where the network delay can be arbitrary. In this work, we begin the study of asynchronous multi-party quantum computation (AMPQC) protocols, where the circuit to compute is quantum.

Our results completely characterize the optimal achievable corruption threshold: we present an n-party AMPQC protocol secure up to $t < n/4$ corruptions, and an impossibility result when $t \geq n/4$ parties are corrupted. Remarkably, this characterization differs from the analogous classical setting, where the optimal corruption threshold is $t < n/3$.

1 Introduction

Secure multi-party computation (MPC) allows a set of parties to compute a function of their private inputs, in such a way that the parties' inputs remain as secret as possible, even in the presence of an adversary corrupting a subset of the parties.

The problem of MPC has been studied mostly in classical setting, where the function to evaluate as well as the adversary are classical [Yao82, GMW87, BGW88, CCD88, RB89]. However, with the recent advances on quantum computing technology, it has become increasingly relevant to consider quantum functionalities and quantum adversaries. This motivated an important line of works on multi-party quantum computation (MPQC) protocols [CGS02, BCG+06, Unr10, DNS12, DGJ+20, BCKM21, ACC+21a].

Current MPQC protocols operate in the so-called *synchronous* network model, where parties have access to synchronized clocks and there is an upper bound on the network communication delay Δ. Although this model is theoretically interesting, it fails to capture real-world networks such as the Internet, which is inherently asynchronous. In fact, assuming a synchronous network is arguably worse in the quantum world, given the difficulties in maintaining quantum states coherently. This has catastrophic consequences, since the security of synchronous protocols is often completely compromised as soon as even one message is delayed by more than Δ time.

In contrast, protocols in the *asynchronous* network model do not rely on any timing assumptions, and messages sent can be arbitrarily (and adversarially) delayed. While asynchronous MPC protocols in the classical setting have been known since a few decades, no protocol has been proposed in the quantum setting. An inherent difficulty in such protocols is that one cannot distinguish between a dishonest party not sending a message, or an honest party that sent a message that was delayed by the adversary. As a result, parties have to make progress in the protocol after receiving messages from $n-t$ parties. This also implies that in this setting it is impossible to consider the inputs of all honest parties – the inputs of up

* Research supported in part by the NSF award 1916939, DARPA SIEVE program, a gift from Ripple, a DoE NETL award, a JP Morgan Faculty Fellowship, a PNC center for financial services innovation award, and a Cylab seed funding award.

** Work partially done while the author was at Carnegie Mellon University.

to t (potentially honest) parties may be ignored. Asynchronous protocols impose even further restrictions in the quantum setting. In particular, since states cannot be duplicated, the no-cloning theorem rules out *quantum broadcast*, which is a crucial tool for enabling classical asynchronous protocols.

In the classical setting, the foundational works of Ben-Or, Canetti and Goldreich [BCG93] and Ben-Or, Kelmer and Rabin [BKR94] showed that the optimal achievable corruption tolerance in the asynchronous model is $t < n/3$. In this work, we initiate the study of the quantum counterpart, namely asynchronous multi-party quantum computation (AMPQC) protocols:

Is it possible to achieve AMPQC? If so, what is the optimal achievable corruption threshold?

We completely resolve this question by providing the first AMPQC protocol secure up to $t < n/4$ corruptions, and showing a lower bound that tolerating $t \geq n/4$ corruptions is impossible.

Theorem 1 (AMPQC Feasibility). *There exists an information-theoretic asynchronous multiparty quantum computation protocol for n parties which is secure against up to any $t < \frac{n}{4}$ corruptions.*

Theorem 2 (AMPQC Impossibility). *No asynchronous multiparty quantum computation protocol exists for n parties which tolerates any corruption threshold $t \geq \frac{n}{4}$.*

1.1 Related Work

Classical Asynchronous MPC. The seminal works of Ben-Or, Kelmen, and Rabin [BKR94], and Ben-Or, Canetti, and Goldreich [BCG93] showed that the optimal achievable corruption tolerance for asynchronous classical MPC is $t < n/3$ even with setup, in both the computational and information-theoretic settings, and $t < n/4$ when requiring perfect security.

Since then, a huge amount of work has been devoted to improving the communication complexity. In the information-theoretic setting with optimal resilience, Patra, Choudury, and Pandu Rangan [PCR10] achieved $\mathcal{O}(n^5\kappa)$ bits per multiplication, where κ is the security parameter, and recently Choudhury [Cho20] further improved this result to $\mathcal{O}(n^4\kappa)$ bits per multiplication. Going to the sub-optimal resilience $t < n/4$, several works achieve linear communication [SR00, PSR02, CHP13, PCR15].

In the cryptographic setting, most protocols make use of some form of homomorphic encryption. The works by Hirt, Nielsen, and Przydatek [HNP05, HNP08] make use of additive homomorphic threshold encryption, with the protocol in [HNP08] communicating $\mathcal{O}(n^2\kappa)$ bits per multiplication, and the work by Chopard, Hirt, and Liu-Zhang [CHL21] achieves adaptive security with the same communication. The work by Choudhury and Patra [CP15] achieves $\mathcal{O}(n\kappa)$ per multiplication using somewhat-homomorphic encryption, and several other works [Coh16, LLM+20, BKLL20] make use of fully-homomorphic encryption to achieve communication complexity independent of the circuit size.

Multi-Party Quantum Computation. All known MPQC protocols assume a synchronous network and have a negligible error probability. Crépeau, Gottesman, and Smith [CGS02] introduced the first MPQC protocol with guaranteed output delivery up to $t < n/6$ corruptions. This was improved by Ben-Or, Crépeau, Gottesman, Hassidim, and Smith [BCG+06], who achieved the optimal resilience $t < n/2$. In the dishonest majority setting, where up to $n - 1$ parties may be corrupted, Dulek, Grilo, Jeffery, Majenz, and Schaffner [DGJ+20] gave the first MPQC protocol achieving security with abort, building upon the work by Dupuis, Nielsen and Salvail that achieved two-party secure computation. Very recently, in a followup work, Alon, Chung, Chung, Huang, Lee, and Shen [ACC+21a] designed a protocol with identifiable abort. Another recent work by Bartusek, Coladangelo, Khurana, and Ma [BCKM21] achieves a constant round MPQC protocol in the dishonest majority setting.
Our work achieves guaranteed output delivery up to $t < n/4$ corruptions under an asynchronous network and incurs negligible error. It is left as an open question whether one can achieve MPQC with perfect security (meaning 0 error probability), even for synchronous networks.

Post-Quantum Computation. A line of works has focused on the problem of *post-quantum* computation which considers classical computation, but an adversary with quantum capabilities.

As noted in [ABG+21], many of the results in the classical setting [BGW88, CCD88, CLOS02, IPS08] can be proven post-quantum secure, provided they are instantiated using primitives that are plausibly quantum secure. Damgård and Lunemann [DL09] introduced a two-party coin-flipping protocol, and

Lunemann and Nielsen [LN11] and Hallgren, Smith and Song [HSS11] introduced general two-party computation protocols secure against quantum adversaries. Bitansky and Shmueli [BS20] gave a constant-round two-party coin-flipping protocol, with full simulation of one party. Finally, very recently, Agarwal, Bartusek, Goyal, Khurana and Malavolta [ABG+21] introduced a constant-round post-quantum multi-party computation protocol in the plain model.

2 Technical Overview

We give an overview of the main techniques used in the paper.

2.1 Feasibility Result

We provide a protocol secure up to $t < n/4$ corruptions under an asynchronous network. Our starting point is the work by Ben-Or, Crépeau, Gottesman, Hassidim, and Smith [BCG+06], which achieves optimal resilience up to $t < n/2$ when the network is synchronous.

Their protocol follows the traditional sharing-based paradigm for multi-party computation [BGW88]. Parties distribute their private inputs with a so-called verifiable quantum secret sharing scheme (VQSS). Parties then evaluate the circuit in a gate-by-gate fashion on the encoded inputs. In the end, parties end up with encodings of the outputs of the circuit, which can each be jointly decoded towards the corresponding party. Technically, the main challenge comes from the design of a VQSS which is in some sense compatible with quantum operations. For this, the authors elegantly design a VQSS which relies on several building blocks, including a special quantum authentication scheme and a so-called *weak* quantum secret-sharing scheme (WQSS), which is similar to VQSS, except that a dishonest dealer can choose not to reconstruct the shared secret (i.e. reconstruct $\perp$). The design allows to push the complexity of performing quantum operations over encoded data to performing classical operations on the authentication keys.

Roughly speaking, the VQSS scheme is composed of three parts: First, the dealer shares $2\kappa + 2$ quantum $|0\rangle$-states among the parties, and each share is then re-distributed using a WQSS scheme. Second, parties jointly run a checking phase to verify that they indeed hold a sharing of $|0\rangle$ (except with error negligible in κ). This is accomplished with a so-called *zero-purity test*, which can be run using a classical trusted third party (TTP). The zero-purity test requires each party to do some local computation before measuring 2κ of the shared states and sending the measurement results to the TTP, which combines them. This consumes 2κ of the shared states. Finally, with the remaining two sharings of $|0\rangle$, the parties jointly create sharings of an EPR pair. The first half of the EPR pair is decoded to the dealer, who can then distribute the secret to the parties using quantum teleportation.

Their scheme relies on a synchronous network, where every message sent by an honest party is delivered within a known delay upper bound, and unfortunately it completely breaks down as soon as even one message gets delayed. In fact, an honest dealer could *appear* corrupted because his messages did not arrive in time. The recipients will then accuse him during the checking phase, resulting in the sharing protocol being unsuccessful.

Challenges in the Asynchronous Setting. In order to make this work in the asynchronous setting, the main obstacle is to design a mechanism that allows parties to learn when they jointly have enough information to uniquely determine the shared secret. When the network is synchronous, this is straightforward, given that parties proceed synchronously and the secret is guaranteed to be shared after a certain number of rounds. In asynchrony, some parties might have gotten a lot of messages while others might still be waiting. Note that if a party P_i didn't receive a message from another party P_j, P_i cannot ask for a missing message, since the response can also be delayed.[5]

In the classical setting, asynchronous VSS is typically solved by reaching agreement on a core-set of parties of size at least $n - t$ who received correct shares. The reasoning is that if $n - 2t > t$, then the core-set contains at least $t + 1$ honest parties, whose shares should uniquely define the secret. The parties in the core-set will be the only ones that contribute their shares in the reconstruction step. One usually also requires a way to tell during the reconstruction phase whether a received share is correct (this is usually achieved using authentication tools such as information checking or signatures, see, e.g.,

[5] In fact, it is more complicated than that: In many synchronous protocols, if P_i doesn't receive a message from P_j, P_j is deemed corrupted and the protocol can for example reveal P_j's secret state.

[BKR94, PCPR09]), which allows parties to correctly reconstruct the secret with high resilience. Three challenges arise when trying to achieve this, which we discuss below.

Using a Classical TTP. First, in order to agree on a core-set, we make use of an *asynchronous* trusted third party (TTP) that can perform classical computations. Perhaps surprisingly, this does not follow from standard classical asynchronous MPC in a black-box way. This is because asynchronous MPC protocols have a very concrete interaction pattern where the TTP waits for the values of any $n - t$ parties (which the adversary can choose by scheduling messages), and outputs a function of these values. However, the interaction that we need with the TTP is more general: in particular, the TTP can wait for the values of designated parties to arrive before computing the function. For example, in the broadcast functionality a designated party sends a message to the TTP, and then the TTP forwards that message to all other parties.

In order to deal with more general functionalities, we first formalize a *generalized* secure function evaluation protocol, which not only takes into account the function f to be evaluated, but also a monotone predicate $Q : \mathcal{P} \to \{0, 1\}$ (i.e., if $T \subseteq T'$ and $Q(T) = 1$, then $Q(T') = 1$) that indicates which sets of parties' inputs may be included into the computation. [6] Using generalized secure function evaluation protocols, one can realize the classical TTP using standard techniques, where each evaluated function takes into account the internal state of the TTP, which is jointly maintained by the parties.

We then show how to modify current existing classical asynchronous protocols for secure function evaluation to achieve generalized secure function evaluation. Current information-theoretic protocols for secure function evaluation follow the traditional sharing-based paradigm, and distribute the inputs as follows: Parties use an asynchronous verifiable secret sharing (AVSS) scheme. Then, since the network is asynchronous, some sharings terminate earlier than others, and therefore parties need to agree on when to proceed to the computation phase. For that, parties run a *core-set agreement* protocol [BKR94, BCG93] to agree on a core-set of parties of size at least $n - t$ whose inputs are taken into the computation (all other parties' inputs are ignored). In order to take the inputs into account according to a predicate Q, one can proceed as follows: run n Byzantine Agreement (BA) protocols $\mathsf{BA}_1, \ldots, \mathsf{BA}_n$, one for each party. Every time the AVSS from party P_j terminates, P_i inputs 1 to BA_j. Every time BA_j outputs 1, it adds P_j to his local set T^i. Party P_i then waits until the set of parties T^i satisfies $Q(T^i) = 1$. If so, P_i inputs 0 to all remaining BAs, and waits for all BAs to terminate before proceeding to the computation phase. Due to agreement of BA, all honest parties agree on the same set of parties $\mathsf{CoreSet}$. Moreover, all honest parties eventually receive all the inputs from parties in $\mathsf{CoreSet}$, due to properties of the AVSS. Finally, since Q is monotone, it follows that the final set of inputs taken into account for the computation satisfies Q (as it contains at least the set T^i). For more details, see Section 7.

Reconstruction and Corruption Thresholds. Second, in contrast to the classical setting, a quantum secret sharing scheme cannot have reconstruction threshold $t+1$, because the no-cloning theorem enforces that the reconstruction threshold must always be at least $\lfloor n/2 \rfloor + 1$ [CGL99]. Due to the asynchronous nature of the network, the reconstructor must be able to perform the reconstruction process with shares from only $n - t$ parties, since the protocol must succeed even if the t adversarial parties refuse to participate. However, in order to uniquely define the secret, $\lfloor n/2 \rfloor + 1$ of those shares must be from honest parties because of the reconstruction threshold. Combining these observations with the fact that t of the provided shares can be from corrupted parties imposes the requirement that $n - 2t \geq \lfloor n/2 \rfloor + 1$, or $t \leq n/4$. Note that in the classical setting, setting the reconstruction threshold to $\lfloor n/3 \rfloor + 1$ allows $t \leq n/3$. We later expand this intuition into a proof of impossibility for $t \geq n/4$.

Robust Reconstruction. Finally, even for $t < n/4$ and assuming a classical TTP, it is not clear how to robustly reconstruct a secret (even in the strictly weaker setting of WQSS).

To see this, let us say that one follows the classical approach of defining a core-set of $n - t$ parties who received correct shares and should contribute during the reconstruction phase. Among these, given that t parties may be corrupted, parties cannot expect to receive all and must reconstruct already from $n - 2t$ shares. However, the $n - 2t$ received shares may contain t corrupted shares. Given that one needs at least $\lfloor n/2 \rfloor + 1$ shares to reconstruct the secret, one can only safely reconstruct if $t < \frac{n}{6}$. In this case, there are $n - 2t - |\mathsf{errors}| = n - 3t > n/2$ honest shares, which is enough to uniquely define the secret. However, a

[6] In a traditional asynchronous protocol, Q evaluates to 1 for any set of size at least $n - t$.

higher corruption threshold might result in the honest shares *not* uniquely defining the secret, in which case the corrupted parties might be able to force a different reconstruction value. Therefore achieving the optimal threshold of $t < \frac{n}{4}$ requires additional ideas.

In the classical setting, this issue is addressed by letting each share be signed by every party during the sharing phase. This ensures that during the reconstruction the adversary cannot send a corrupted share (with correct signatures) corresponding to a different secret. Unfortunately, there is no easy way to achieve this in the quantum setting (digitally signing quantum data is even impossible! [BCG+02, AGM21]). To overcome this barrier we introduce a novel primitive called asynchronous weak quantum secret-sharing scheme (AWQSS) *with weak termination*, which does not guarantee that the reconstruction procedure outputs a value (even when the sharing was successful) in the dishonest-dealer case.[7] We then show how this weaker primitive is enough to achieve AVQSS.

The starting point for the AWQSS with weak termination is the synchronous protocol from Ben-Or et. al. [BCG+06]. The dealer verifiably distributes authenticated shares of two $|0\rangle$ states, where the classical TTP holds the classical authentication key. With the help of the TTP, the parties test the two sets of shares to ensure they are actually shares of $|0\rangle$ states. Once assured, the parties entangle the two shared qubits to create a shared EPR pair and send half to the dealer. Finally, the dealer uses the EPR half it receives to teleport its state. Since the protocol is required to progress with only $n-t$ active participants, the $|0\rangle$ test must occur as soon as any $n-t$ parties have provided their test measurements.

Contrary to the synchronous WQSS case, the requirement that progression is necessary with only $n-t$ parties opens up problems in aligning the cases when the dealer is honest and when it is corrupt. Specifically, it becomes possible for t honest parties to receive inconsistent shares (or not receive them at all) without disrupting the sharing protocol. Then, depending on the behavior of the corrupted parties during reconstruction, the reconstructor may only receive $n-2t$ shares. Since the reconstructor cannot tell whether the remaining shares are simply delayed on the network or withheld by corrupt parties, it must decide whether to attempt to reconstruct the secret or output $\perp$ based only on these $n-2t > \lfloor n/2 \rfloor + 1$ shares. However, even though $n-2t$ is at least the threshold of $\lfloor n/2 \rfloor + 1$ for $t < n/4$, it is never safe to attempt to reconstruct the secret, even if the received shares are consistent. This is because if the dealer is corrupt, then they know the authentication keys and so the other corrupt parties may provide up to t arbitrary shares in the reconstruction process, potentially changing the reconstructed value. To make matters worse, the same situation can occur with an honest dealer, but reconstruction must always succeed in this case!

Our insight here is to build a late checking mechanism into the protocol which allows parties *outside of the core-set* to contribute. These parties did not participate in the initial share-checking, but will still hold valid shares in the case of an honest dealer. This is in contrast to current classical asynchronous protocol techniques, where only parties inside the core-set contribute their shares in the reconstruction step.

Honest parties who receive their shares after the $|0\rangle$ test has already occurred can send their portion of the test measurements[8] to the TTP and receive back the result of performing the $|0\rangle$ test on their measurement and $n-t-1$ of the original $n-t$ test measurements used in the main $|0\rangle$ test. This TTP behavior makes use of the expanded classical TTP functionality discussed above in order to allow a single party to interact with the TTP. With an honest dealer, the t honest parties who are delayed will have their shares confirmed by the TTP *after* the main check has already occurred. This allows $n-t$ shares to be provided to the reconstructor, which is sufficient to complete reconstruction safely. It is important to note that late checking does not change the case for a corrupted dealer, since the delayed honest parties might never receive shares in the first place.

AVQSS avoids the weak termination problem by creating a two-level sharing of $|0\rangle$. Creating a two-level sharings of $|0\rangle$ intuitively means that the quantum state $|0\rangle$ is shared using an AWQSS scheme, and each level-1 share is again shared using an AWQSS scheme. The sharing terminates once at least $n-t$ parties hold correct shares, as checked by another $|0\rangle$ test. These $n-t$ parties uniquely define the secret. Moreover, during the reconstruction, corrupted parties cannot send corrupted level-1 shares, since each of these shares are distributed among all honest parties. They can, however, withhold their shares. But

[7] In ordinary weak secret-sharing, as is used in prior work, the reconstructor is allowed to output $\perp$, but must terminate at some point. With weak termination, the reconstructor may even fail to output $\perp$ when the dealer is dishonest.

[8] The test is defined for all parties, although it does not require all of them to participate.

since the $n-t$ parties contain at least $n-2t \geq \lfloor n/2 \rfloor + 1$ honest parties, these are enough to reconstruct the secret.

2.2 Impossibility Result

As we saw above, we require the corruption threshold to be $t < n/4$ in our protocol. Interestingly, we show that this corruption threshold is optimal. This is in contrast to the classical setting, where the optimal threshold for asynchronous computation is $t < n/3$ [BKR94].

Formally, we prove that AVQSS is impossible for $t \geq n/4$. The ideas in our proof generalize naturally to a secure function evaluation protocol for all-to-all AVQSS, where all the parties end up with shares from each of the $n-t$ parties in the core-set. We provide a high level idea of the proof below for $n = 4$ and $t = 1$. By standard arguments, this implies an impossibility for the general case $t \geq n/4$.

Consider the existence of an AVQSS protocol with four parties D, P_2, P_3, P_4, where D also acts as a recipient, and secure up to $t = 1$ corruption. We show that this implies an "approximate" quantum erasure-correcting code (QECC) of length 4 and *approximately* correcting 2 erasures, in the sense that the decoded quantum state is close in trace distance to the true input state. We prove that it is impossible to construct such codes. The idea is that there can be one corrupted party, and, because the protocol succeeds under an asynchronous network, it must succeed even when a potentially honest party is locked out of the protocol. Formally showing this intuition requires carefully designing a scenario-based argument.

A bit more formally, consider a secret x, and further consider without loss of generality the set $\{P_3, P_4\}$ (the argument holds for any set of size 2). We will show that the internal state of $\{P_3, P_4\}$ fully determines x (up to a small error).

Consider a first scenario where all parties are honest in the execution of the sharing phase, but P_2's messages are delayed. Since the protocol is secure even when P_2 crashes from the start, all parties successfully terminate holding a share. In this case, since the dealer is honest, the reconstructed value is x.

In the second scenario, P_2 does not receive any information from a corrupted dealer D in addition to having its messages delayed. The dealer D otherwise behaves as in an honest execution with input x with respect to parties $\{P_3, P_4\}$ (by internally emulating P_2). In this case, the view of $\{P_3, P_4\}$ is exactly the same as in the first scenario. Furthermore, note that there is an adversarial strategy so that the reconstructed value is x: simply let D also participate in the reconstruction protocol honestly while delaying all messages from P_2, as this is the same execution as in the first scenario. In turn, this means that the committed value is x, and, *regardless* of the adversarial strategy, the reconstructed value must be x.

However, since P_2 did not receive any message from D, his internal state cannot provide any information that is unknown to $\{P_3, P_4\}$. Since AQVSS requires the views of the honest parties to define the secret, the internal state of $\{P_3, P_4\}$ must fully determine the secret x.

This implies the existence of an approximate QECC with length 4 that is resilient to two erasures, since the secret can be recovered from any two shares.

3 Preliminaries

3.1 Notation

We denote the security parameter by κ and write $\mathtt{neg}(\kappa)$ for any function that is negligible in κ, i.e., $\mathtt{neg}(\kappa)$ decays faster than κ^{-c} for any constant $c > 0$ as κ grows. We also write $\mathsf{poly}(\kappa)$ for a function that grows polynomially with κ. We write $k \leftarrow K$ to mean that k is sampled uniformly at random from the set K. The finite field of order q is denoted by $\mathbb{F}_q$. The conjugate transpose of a matrix U is denoted by $U^\dagger$.

3.2 Concepts from Quantum Computation

We assume familiarity with basic concepts from quantum computation, such as pure and mixed states, density matrices, Clifford and Toffoli gates, entanglement, measurements, and quantum teleportation. We

refer the reader to the book of Nielsen and Chuang [NC10] for an overview of these concepts. We denote quantum registers and gates by uppercase roman letters. The distinction will be clear from context. Density matrices are denoted by lowercase greek letters such as ρ and σ, and we sometimes write ρ_M for the state associated with register M. Furthermore, we write U_M for a unitary transformation U to denote that U is applied to the contents of register M.

Trace Distance. We make use of the notion of trace distance between states.

Definition 1 (Trace distance). *The* trace distance *between two mixed states with associated density matrices ρ and σ, denoted by $D(\rho, \sigma)$, is given by*

$$D(\rho, \sigma) = \frac{1}{2}\|\rho - \sigma\|_1,$$

where $\|\rho\|_1 = \mathrm{Tr}[\sqrt{\rho^\dagger \rho}]$ is the trace norm*.*

The trace distance is a distance and has the following useful interpretation: If $D(\rho, \sigma) \leq \varepsilon$, then any POVM applied to states with density matrices ρ and σ yields classical measurement outcome distributions, say $(p_1, \ldots, p_m)$ and $(q_1, \ldots, q_m)$, which are ε-close in statistical distance, i.e., $\frac{1}{2}\sum_{i=1}^{m} |p_i - q_i| \leq \varepsilon$.

Generalized Quantum Gates. We work with basis states indexed by elements of a finite field $\mathbb{F}_p$ and apply several quantum operations to such states. We describe them next. We consider the generalized Pauli gates X and Z over $\mathbb{F}_p$, which act as $X|\alpha\rangle = |\alpha + 1\rangle$, where the sum is over $\mathbb{F}_p$, and $Z|\alpha\rangle = \omega_p^\alpha |\alpha\rangle$, where $\omega_p = e^{2\pi i/p}$. Moreover, we will use the controlled SUM gate (generalizing the CNOT gate) over $\mathbb{F}_p$ acting as $\mathrm{SUM}|\alpha, \beta\rangle = |\alpha, \alpha + \beta\rangle$, and the γ-Fourier gate F_γ over $\mathbb{F}_p$ acting as

$$F_\gamma |\alpha\rangle = p^{-1/2} \sum_{\beta \in \mathbb{F}_p} \omega_p^{\gamma\alpha\beta} |\beta\rangle .$$

When $\gamma = 1$ we simply write $F = F_1$.

3.3 Quantum Secret Sharing

Quantum secret sharing [HBB99] allows a dealer to share a secret quantum state so that authorized subsets can recover the secret, but unauthorized sets cannot gain information about the secret. In our schemes, we use an extension of Shamir's secret sharing scheme to the quantum setting, first described by Cleve, Gottesman, and Lo [CGL99]. We begin by defining threshold quantum secret sharing schemes.

Definition 2 (Threshold quantum secret sharing scheme). *A pair of maps* $(\mathsf{Share}, \mathsf{Reconstruct})$ *is an* (n, t)-quantum threshold secret sharing (QTSS) scheme *from a message register M to share registers $S_1, S_2, \ldots, S_n$ if for every message state ρ_M the conditions below hold:*

1. *Correctness: If $T \subseteq [n]$ satisfies $|T| \geq t$, then*

$$\mathsf{Reconstruct}(\mathsf{Share}(\rho_M)_T, T) = \rho_M$$

   for every message ρ_M, where $\mathsf{Share}(\rho_M)_T$ denotes the subsystem associated to registers $(S_i)_{i \in T}$;
2. *Privacy: If $T \subseteq [n]$ satisfies $|T| < t$, then there is a density matrix ρ_T such that the reduced density matrix of $\mathsf{Share}(\rho_M)_T$ is ρ_T for every message ρ_M. In other words, $\mathsf{Share}(\rho_M)_T$ gives no information about ρ_M.*

Note that the no-cloning theorem implies that we must always have $t > n/2$ in a QTSS scheme [CGL99].

Quantum Shamir Secret Sharing. Suppose we wish to share a state over $\mathcal{H}$ among $m = 2r + 1$ parties for some parameter r. Fix a prime $p \in (m, 2m)$ and consider m distinct non-zero evaluation points $\alpha_1, \ldots, \alpha_m \in \mathbb{F}_p$. Furthermore, identify a basis of $\mathcal{H}$ with elements of $\mathbb{F}_p$. Then, we define Share to behave on basis states $|\beta\rangle$ for $\beta \in \mathbb{F}_p$ as

$$\mathsf{Share}(|\beta\rangle \otimes |0\rangle^{\otimes m-1}) = p^{-r/2} \sum_{f \in \mathbb{F}_p[x] : \deg(f) \leq r, f(0) = \beta} |f(\alpha_1), \ldots, f(\alpha_m)\rangle,$$

where the sum ranges over all polynomials f of degree at most r over $\mathbb{F}_p$ such that $f(0) = \beta$ and the i-th party receives the i-th register S_i associated to α_i. Given T and $\mathsf{Share}(\rho_M)_T$, the reconstruction procedure $\mathsf{Reconstruct}$ performs Lagrange interpolation based on the evaluation points $(\alpha_i)_{i\in T}$ to recover the message. Cleve, Gottesman, and Lo [CGL99] showed that this sharing procedure induces an efficient $(2r+1, r+1)$-QTSS scheme. We can then obtain an $(n, r+1)$-QTSS scheme for arbitrary n such that $r+1 \leq n \leq 2r+1$ simply by discarding the last $2r+1-n$ shares.

3.4 Quantum Authentication Schemes

Quantum authentication schemes were first introduced in [BCG$^+$02]. Intuitively, a quantum authentication scheme encodes a quantum state with the help of a classical key so that operations performed on the authenticated state by an adversary who does not know the key can be detected.

We consider a keyed scheme $(\mathsf{Auth}_k, \mathsf{Dec}_k)$ with key space K. Given $k \leftarrow K$, the authentication procedure Auth_k maps the contents of a message register M to a ciphertext register C. The decoding procedure Dec_k then maps the contents of C, which may have been modified by the adversary, to a message register M and a flag register F over a space with orthogonal basis accept/reject states $|\mathsf{acc}\rangle$ and $|\mathsf{rej}\rangle$. The recipient can measure the contents of the flag register to check whether the message was modified. We require that for every key $k \in K$ and message ρ_M it holds that

$$\mathsf{Dec}_k(\mathsf{Auth}_k(\rho_M)) = \rho_M \otimes |\mathsf{acc}\rangle\langle\mathsf{acc}|,$$

i.e., if the message is not modified then decoding succeeds and returns the message. On the other hand, intuitively, we require that if the message is modified, then Dec_k either accepts and returns the original message ρ_M, or rejects and returns a placeholder state. More formally, we present a composable simulation-based security definition, as discussed in [BCG$^+$06, DNS12, BW16, HLM16]. Consider an additional side information register R held by the adversary which may be entangled with the message; we see the contents of registers M and R as a state ρ_{MR}.

Real-World Process. In the real-world, we sample $k \leftarrow K$. The adversary $\mathcal{A}$ has access to C and R and applies a unitary transformation U_{CR} to these registers. The output of this process (over registers M, F, and R) is

$$\mathtt{REAL}_{\mathcal{A}}(\rho_{MR}) = \frac{1}{|K|}\sum_{k\in K}(\mathsf{Dec}_k \otimes I_R)(U_{CR}(\mathsf{Auth}_k \otimes I_R)(\rho_{MR})U_{CR}^\dagger).$$

Ideal-World Process. In the ideal-world, the simulator $\mathcal{S}$ has access to R and a flag bit flag. If it sets $\mathsf{flag} = 0$, then the channel outputs $\Omega_M \otimes |\mathsf{rej}\rangle\langle\mathsf{rej}|$, where Ω_M is a placeholder state. Else, if flag is set to 1, the message is left unchanged and F is set to $|\mathsf{acc}\rangle$. Moreover, the simulator is allowed to update R based on the value of flag. We denote the overall output of this process by $\mathtt{IDEAL}_{\mathcal{S}}(\rho_{MR})$.

Security Notion. We say that a scheme $(\mathsf{Auth}_k, \mathsf{Dec}_k)$ is an *ε-quantum authentication (ε-QA) scheme* if for any adversary $\mathcal{A}$ there exists a simulator $\mathcal{S}$ such that

$$D(\mathtt{REAL}_{\mathcal{A}}(\rho_{MR}), \mathtt{IDEAL}_{\mathcal{S}}(\rho_{MR})) \leq \varepsilon \tag{1}$$

for all states ρ_{MR}.

Polynomial-Based Authentication Scheme. We will be using the efficient polynomial-based authentication scheme from [BCG$^+$06] and based on [AB97]. This scheme has several properties which will be quite useful. First, the authentications keys (k, x) are classical, which allows them to be managed by the classical TTP. This advantage is especially important because their scheme allows the application of Clifford gates and measurements to authenticated states where the keys are held by the TTP and the quantum registers are held by other parties. In the case of multi-qubit gates, states authenticated under the same k (but potentially different x) can be operated on jointly. Second, the scheme remains secure in a setting where the adversary has access to several states authenticated with the same k but *independently sampled x's*. This will later allow parties to authenticate multiple states which can all be operated on together.

Lemma 1 ([BCG+06, ABEM17, HLM16]). *The scheme* $(\mathsf{Auth}_{x,k}, \mathsf{Dec}_{x,k})$ *is a* $\mathtt{neg}(\kappa)$*-QA scheme when* $m = \kappa$*.*

Lemma 2 ([BCG+06]). *The scheme* $(\mathsf{Auth}_{x,k}, \mathsf{Dec}_{x,k})$ *is* $\mathtt{neg}(\kappa)$*-secure in the s-parallel authentication setting when* $m = \kappa$ *and* $s \leq \mathsf{poly}(\kappa)$*.*

For completeness, we discuss the construction and properties in more detail in Appendix A.

4 Model of Multiparty Computation

We consider a set of n parties $\mathcal{P} = \{P_1, \ldots, P_n\}$. We extend the standard classical asynchronous model for multiparty computation to the quantum setting, where parties have quantum states as inputs and wish to apply a quantum circuit to their states.

We consider quantum circuits which use only Clifford and Toffoli gates, since these form a universal quantum gate set [Sho96].

4.1 Communication and Adversarial Models

Parties have access to point-to-point private classical and quantum channels. For simplicity, we consider a *static* computationally unbounded adversary who is allowed to corrupt any t parties at the start of the protocol, who may deviate arbitrarily from the protocol. But we believe our protocols should also achieve adaptive security.

Our network model is *asynchronous*. This means that there may be an arbitrary (finite) delay between the time a message is sent and the time it is delivered. The adversary controls the scheduling of the messages, subject only to the constraint that every message sent must eventually be delivered. Since we assume private channels, the adversary has no information about the contents of each message besides the identity of the sender and receiver.

4.2 Multi-Party Asynchronous Quantum Computation

We adapt the security definition to our setting, following the works on asynchronous classical MPC by Ben-Or, Canetti, and Goldreich [BCG93] and synchronous MPQC by Dulek, Grilo, Jeffery, Majenz, and Schaffner [DGJ+20]. The security definition introduces a real and an ideal world, and intuitively guarantees that any attack that happens in the real-world can be efficiently reproduced in the ideal-world. Similar to previous works on MPQC (see e.g. [DNS12, DGJ+20, ACC+21b]), we chose to provide a security definition that is simple but stand-alone. However, note that our simulators are black-box straight-line (they do not access the code of the adversary or rewind the adversary), so our overall protocol might as well satisfy UC security.

Real-World Process. In the real-world, protocol Π is executed with the set of parties $\mathcal{P}$, adversary $\mathcal{A}$ and environment $\mathcal{Z}$. All entities receive the security parameter κ. Parties execute the protocol and the adversary $\mathcal{A}$ corrupts up to t of the parties at the onset of the computation (corrupted parties behave arbitrarily). The environment $\mathcal{E}$ learns the identities of the corrupted parties and interacts with the adversary and the parties. In particular it chooses for each party P_i the input R_i, and it learns all the outputs. In the end, the environment outputs a bit b. We denote by $\mathtt{REAL}_{\Pi,\mathcal{A},\mathcal{E}}$ the output from the environment $\mathcal{E}$ when interacting with protocol Π and adversary $\mathcal{A}$.

Ideal-World Process. In the ideal-world, the environment interacts with the ideal adversary (a.k.a. the simulator) $\mathcal{S}$ and the parties as described below. We denote by $\mathtt{IDEAL}_{C,\mathcal{S},\mathcal{E}}$ the output from the environment $\mathcal{E}$ when interacting in the ideal process with circuit C. We formally describe the ideal process below.

Functionality Ideal world

The parties $P_1, \ldots, P_n$ hold quantum registers $R_1, \ldots, R_n$. There is also an input ancillary quantum register R_{ancilla} with k qubits, initialized to $|0\rangle^{\otimes k}$. Let C be a quantum circuit with W_{in} input and W_{out} output wires. Consider a partition of the input/output wires into the parties' registers and an ancilla register as $[W_{\mathsf{in}}] = W_{\mathsf{in},1} \sqcup \cdots \sqcup W_{\mathsf{in},n} \sqcup W_{\mathsf{in},\mathsf{ancilla}}$ and $[W_{\mathsf{out}}] = W_{\mathsf{out},1} \sqcup \cdots \sqcup W_{\mathsf{out},n} \sqcup W_{\mathsf{out},\mathsf{ancilla}}$.

1: The adversary can replace the inputs of corrupted parties by any (possibly entangled) quantum state of his choice.
2: The adversary chooses a set of parties $S \subseteq [n]$ of size $|S| \geq n - t$.
3: The adversary sends the contents of the registers R_i for $i \in S$ to the TTP.
4: The TTP feeds the registers $R'_1, \ldots, R'_n, R_{\mathsf{ancilla}}$ to the appropriate subset of wires of W_{in} in the quantum circuit C, where the contents of R_i are written into R'_i if $i \in S$ and R'_i contains an unentangled $|0\rangle^{\otimes k_i}$ state otherwise.
5: The TTP runs the quantum circuit and sends the content of the i-th output register fed by wires in $W_{\mathsf{out},i}$ to party P_i.

Security Notion. The security notion states that the real and ideal worlds are indistinguishable for any quantum polynomial-time (QPT) environment. More concretely, for any QPT adversary $\mathcal{A}$ in the real-world, there must be a QPT simulator $\mathcal{S}$ such that no QPT environment $\mathcal{E}$ can distinguish between both worlds.

Definition 3. *Protocol Π t-*securely computes circuit C *if for any quantum polynomial-time (QPT) adversary $\mathcal{A}$ corrupting up to t parties, there is a QPT adversary $\mathcal{S}$ such that for every environment $\mathcal{E}$:*

$$|\Pr[\mathtt{IDEAL}_{C,\mathcal{S},\mathcal{E}} = 1] - \Pr[\mathtt{REAL}_{\Pi,\mathcal{A},\mathcal{E}} = 1]| \leq \mathtt{neg}(\kappa)$$

4.3 Classical Trusted Third Party

We make use of a classical trusted third party in our protocol which is stateful and reactive. Upon receiving any message from any party, it parses the message type and responds according to the message contents accordingly. For ease of exposition, we present the reactions of the TTP modularly alongside the corresponding sub-protocols. When the TTP would be initialized with the same information at multiple steps in a protocol, it should be understood that it is initialized once with this information and saves it as internal state.

Realizing a classical asynchronous TTP does not follow directly from protocols for secure function evaluation, since the evaluated function takes into account the inputs of *any* $n - t$ parties. In some cases, the interaction with the TTP can include instructions that allow the TTP to wait for inputs from designated parties. In Section 7, we show how to achieve such a classical asynchronous TTP.

5 Protocols

5.1 Verified Quantum State Authentication

The verified authentication of zeros protocol allows each honest party to receive an authenticated $|0\rangle$ state from some dealer and to be certain that all other honest parties will eventually receive a $|0\rangle$ state authenticated under the same key. These $|0\rangle$ states can then be transformed into an arbitrary authenticated state using quantum teleportation. Later, this will act as a way to allow honest parties to prove to the classical TTP that they provided measurements of some state received from the dealer.

At a high level, the dealer prepares and authenticates many $|0\rangle$ states. They send the classical authentication keys to the TTP, then send each party some of the $|0\rangle$ states. With the help of the TTP, each party tests the states it received using the zero purity testing protocol from [BCG+06]. If the test passes, the party keeps one of the $|0\rangle$ states and forwards the rest to the other parties. If it fails, the party waits to receive forwarded states, which it then tests again.

Functionality Verified Authentication of Zeros Classical TTP

Initialization step. The TTP receives keys $(k, \vec{x})$ from the dealer.

Execution. Upon receiving the message $(\mathsf{VAZCheck}, m, s, f)$ from any party P_i, where m is a set of measurements, s is a starting index, and f is a finishing index, the TTP does the following:

1: Finish the zero purity test on states s through $f-1$ using the measurements m. This involves temporarily updating $\vec{x}$. Let the result be result.
2: Send $(\mathsf{VAZCheck}, s, f, \mathsf{result})$ to P_i.

Upon receiving the message $(\mathsf{VAZAgree}, 1)$ from $n-t$ distinct parties, send the message $(\mathsf{VAZAgree}, \mathsf{Accept})$ to all parties.

Protocol Verified Authentication of Zeros

Initialization step. Let $s = \mathsf{poly}(\kappa)$. The dealer D chooses one random key k and a random $\vec{x}$ consisting of $n^2(r+2s)$ substrings. They create states $\mathsf{Auth}_{(k,x_j)}(|0\rangle)$ for each $j = 1, \ldots, n^2(r+2s)$ and send each player $n(r+2s)$ of them, then send the keys $(k, \vec{x})$ to the TTP.

Protocol Execution.

1: Upon receiving a message containing $n(r+2s)$ states, P_i performs the zero purity testing protocol from [BCG+06] on them, getting measurements m. This spends $2s$ of the states.
2: P_i sends $(\mathsf{VAZCheck}, m, in(r+2s), (i+1)(n)(r+2s))$ to the TTP.
3: P_i receives the TTP's decision bit indicating whether the zero purity testing protocol should accept or reject.
4: **if** The TTP's decision bit indicates Accept **then**
5: Send $r+2s$ authenticated $|0\rangle$ states to each other player.
6: Send $(\mathsf{VAZAgree}, 1)$ to the TTP.
7: Upon receiving $(\mathsf{VAZAgree}, \mathsf{Accept})$ from the TTP, output the remaining r states.
8: **else**
9: **for** each received set of states from a player P_j **do**
10: P_i performs the zero purity testing protocol on the received states, getting measurements m.
11: P_i sends $(\mathsf{VAZCheck}, m, (in+j)(r+2s), (in+j+1)(r+2s))$ to the TTP.
12: P_i receives the TTP's decision bit indicating whether the zero purity testing protocol should accept or reject.
13: **if** the TTP's decision bit indicates accept **then**
14: Upon receiving $(\mathsf{VAZAgree}, \mathsf{Accept})$ from the TTP, output the remaining states received from P_j.
15: **end if**
16: **end for**
17: **end if**

The following lemma states some important properties of the proposed authentication protocol.

Lemma 3. *With probability at least $1 - \mathtt{neg}(\kappa)$, the following properties hold in an execution of the Verified Authentication of Zeros protocol:*

- *Each honest party which outputs holds r states with trace distance $\mathtt{neg}(\kappa)$ to $|0\rangle$ states authenticated under the keys $(k, \vec{x})$ held by the classical TTP.*
- *If any honest party outputs, then all honest parties do so.*
- *If the dealer is honest then all honest parties output.*

Moreover, these properties continue to hold even when composed in parallel with another Authentication of Zeros protocol which shares the same dealer key k (and independent $\vec{x}$).

Proof. The first part of the lemma statement follows directly from the soundness of the $|0\rangle$ purity test which the TTP carries out using the keys $(k, \vec{x})$ (Lemma 17).

If some honest party outputs, then it received $(\mathsf{VAZAgree}, \mathsf{Accept})$ from the TTP, so all honest parties eventually receive this from the TTP. Furthermore, since the TTP sent this message, it received the message $(\mathsf{VAZAgree}, 1)$ from $n-t$ parties. Since $2t < n$, at least one of these must have been an honest party, say P_i. The party P_i therefore must have sent $r+2s$ authenticated $|0\rangle$ states which passed the

purity test to every other party, so every other party will either use its own states or the states which it received from P_i.

If the dealer is honest, then all honest parties pass the zero purity test and send $(\mathsf{VAZAgree}, 1)$ to the TTP. This yields $n - t$ messages, so the TTP sends $(\mathsf{VAZAgree}, \mathsf{Accept})$ to all parties.

Parallel composition while sharing the key k follows from a similar argument to above, except we rely on the following properties continuing to hold even in the presence of a parallel repetition of the respective protocol sharing the key k: the soundness of the zero purity test, the security of the ideal authentication scheme, and the security of the classical TTP.

□

Protocol Authenticated Teleportation TTP

Initialization step. The TTP receives keys $(k, \vec{x})$ from the dealer.

Execution.

1: Upon receiving measurement results from the dealer, the TTP applies the measurement results to the keys $\vec{x}$ to complete the teleportation.
2: The TTP informs all parties that the teleportation is complete.

Protocol Authenticated Teleportation

We describe the protocol from the point of view of party P_i. The dealer D aims to send the contents of a register R_D to the receiver R, where at the end of the protocol the sent state is authenticated.
Initialization Step. D sends at least 2 authenticated $|0\rangle$ states to R using the Verified Authentication of Zeros protocol, where the TTP receives the authentication keys $(k, \vec{x})$.
Receiver Execution

1: R constructs an EPR pair using two of the authenticated $|0\rangle$ states.
2: R sends one half of the EPR pair to D.
3: R terminates when the TTP responds.

Dealer Execution Upon receiving an EPR pair half, D uses it to teleport the contents of R_D, sending the measurement results to the TTP.

5.2 Asynchronous Weak Quantum Secret Sharing with Weak Termination

The next building block for asynchronous verifiable quantum secret sharing is asynchronous weak quantum secret sharing (AWQSS), which is described by a pair of protocols $(\mathsf{Share}, \mathsf{Reconstruct})$. In the first protocol, Share, a designated party called the dealer D distributes a private input s among the set of parties. In the second protocol, Reconstruct, the parties jointly participate and a designated receiver R obtains the secret. In weak secret sharing, R is also allowed to output $\perp$ when the dealer is dishonest. The protocol we describe uses a trusted party which performs classical computation.

Definition 4 (Asynchronous Weak Quantum Secret Sharing with Weak Termination). *Consider a pair of protocols* $(\mathsf{Share}, \mathsf{Reconstruct})$ *for n parties, where a designated party D, called the* dealer*, has a private input quantum state s for* Share*, and each honest party that completes* Share *subsequently invokes* Reconstruct *with its local output of* Share*, and a designated* reconstructor *R outputs a quantum state upon terminating. We say that* $(\mathsf{Share}, \mathsf{Reconstruct})$ *is a* (t, ε)-secure asynchronous weak quantum secret sharing scheme *if the following holds with probability at least $1 - \varepsilon$ whenever up to t parties are corrupted:*

- *Termination:*
  - *If D is honest, then every party eventually terminates* Share*. Moreover, R outputs a state at the end of* Reconstruct*.*
  - *If some honest party terminates* Share*, then all honest parties eventually terminate* Share*.*
- *Privacy: If D is honest, the view of the adversary is independent of s.*
- *Correctness: If D is honest, then if R outputs a state at the end of* Reconstruct*, the state is s.*

– *Commitment: Even if D is corrupted, if all honest parties terminate* Share*, their joint views defines a unique state s', such that if R outputs a state after* Reconstruct*, the state has* $\mathtt{neg}(\kappa)$ *trace distance to either s' or $\perp$.*

Note that if D is corrupted, an honest reconstructor R is not guaranteed to obtain an output from the protocol Reconstruct. We emphasize that this occurs only at the level of weak secret sharing and does not propagate to verifiable secret sharing.

Functionality AWQSS Classical TTP

Initialization step. Let $s = \mathsf{poly}(\kappa)$. The TTP receives keys $(k, \vec{x})$ from the dealer D. It sets $S = \varnothing$ and $\mathsf{result} = \perp$.

Sharing Execution Upon receiving $(\mathsf{ShareCheck}, m)$ from P_i, where m is a set of measurements and P_i has not sent a ShareCheck message before, the TTP does the following:

Verify the authentication code on m using $(k, \vec{x})$. If it fails, discard this message. Otherwise, continue:
if $|S| < n - t$ **then**
  $S \leftarrow S \cup \{i\}$
  if $|S| = n - t$ **then**
    Perform the zero purity test using m and set store the result in result.
    Send $(\mathsf{ShareCheck}, \mathsf{result})$ to each P_j for $j \in S$.
  end if
else
  Perform the purity test $2s$ times using m.
  Send $(\mathsf{ShareCheck}, \mathsf{acc})$ to P_i if this test accepts and result = acc. Otherwise send $(\mathsf{ShareCheck}, \mathsf{rej})$ to P_i.
end if

Reconstruction Execution The TTP sends $(\mathsf{Reconstruct}, k, \vec{x})$ to the reconstructor R.

Protocol AWQSS

We describe the protocol from the point of view of party P_i, then the final sharing from the point of view of the dealer D. The dealer D aims to share the contents of a register R_D with the other players.

Initialization step. Let $s = \mathsf{poly}(\kappa)$. The dealer D chooses a random key k_D. It generates $2s + 2$ $|0\rangle$ states for $s = \mathsf{poly}(\kappa)$ and shares them using quantum Shamir secret sharing with threshold $\lfloor \frac{n}{2} \rfloor + 1$ and n shares. For each encoded $|0\rangle$ state, send the i-th share to P_i using the key k_D and freshly random x in the Verified Authentication protocol. Let $\vec{x}$ be the vector containing each x which is used. D sends the keys $(k, \vec{x})$ to the TTP.

Sharing Execution.

Let $R_1, \ldots, R_{2s+2}$ be the registers containing the (authenticated) shares which P_i received from D.
Perform the zero purity test measurements transversally, storing the results in m.
P_i sends $(\mathsf{ShareCheck}, m)$ to the TTP. Implicitly, the TTP finishes the zero purity test.
P_i waits to receive $(\mathsf{ShareCheck}, \mathsf{result})$ from the TTP.
if result is acc **then**
  P_i generates an EPR pair share using the two remaining $|0\rangle$ shares in R_0, R_1.
  Send the share in R_0 to D.
else
  Abort
end if

Teleportation

D waits until it receives $\frac{n}{2}$ correctly authenticated shares from the parties, then uses them to reconstruct the EPR pair half (this is possible since it knows the keys and can update them according to the EPR construction, which is deterministic).
D uses the reconstructed EPR pair half to teleport the contents of R_D, sending the measurement results to the TTP.

3: The TTP informs all parties that the sharing is over.

Reconstruct

1: All parties send their shares to the reconstructor R.
2: R waits to receive the classical keys classical keys $(k, \vec{x})$ from the TTP.
3: R checks the authentication on each received share and discards any which do not authenticate correctly.
4: R waits until $n - t$ correctly authenticated shares are received.
5: R removes the authentication on these shares.
6: **if** these shares all lie on the same polynomial superposition **then**
7: R reconstructs and outputs the state using an arbitrary $\lfloor \frac{n}{2} \rfloor + 1$ of them.
8: **else**
9: R outputs $\perp$.
10: **end if**

Lemma 4. *Protocol AWQSS is a $(t, \mathtt{neg}(\kappa))$-secure asynchronous weak quantum secret sharing scheme for all $t < \frac{n}{4}$. Furthermore, it maintains these properties even when composed in parallel with another AWQSS instance using the same keys k_D (but independent $\vec{x}_D$).*

Proof. We begin by considering the case where the dealer D is honest. To see that every party eventually completes Share, note that every honest party eventually receives an authenticated share of $|0\rangle$ from D by Lemma 3 and D's honesty, so D passes the zero purity test. Therefore, it can reconstruct its half of the EPR pair and perform the teleportation as soon as it receives more than $n/2$ correct shares. The dealer D can test whether a share is correct by checking its authentication code, so it can wait until it has received the requisite number of correct shares. By the correctness of quantum teleportation and quantum Shamir secret sharing, the honest parties hold $n-t$ shares of the state s. During reconstruction, an honest reconstructor R receives the keys $(k, \vec{x})$ from the TTP. Therefore, with probability at least $1 - \mathtt{neg}(\kappa)$, the shares R uses to reconstruct each have $\mathtt{neg}(\kappa)$ trace distance to D's original messages. At least $\lfloor \frac{n}{2} \rfloor + 1$ shares are submitted by honest parties, so R can reconstruct s successfully. The fact that the view of the adversary is independent of the shared state s follows directly from no-cloning and the fact that it holds fewer than t shares.

Now suppose that the dealer D is dishonest and consider the case where an honest reconstructor R would output $s' \neq \perp$ after Reconstruct. Since R output a value, it must have received $n - t$ shares. Note that every honest party who submits a share to R has confirmed, via the TTP, that their share was originally a correct share of $|0\rangle$, in the sense that with probability $1 - \mathtt{neg}(\kappa)$ their share is part of the same Shamir secret sharing as the original $n-t$ shares which the TTP checked and that the $n-t$ shares which the TTP checked decoded to a state with $\mathtt{neg}(\kappa)$ trace distance to $|0\rangle$. The correctness of the share persists across the teleportation. Therefore the joint views of all honest parties after outputting a value during Share define a unique value s'' . It remains to be shown that $s' = s''$ except with probability $\mathtt{neg}(\kappa)$. As before, R receives the keys $(k, \vec{x})$ from the TTP. Observe that R receives $n - 2t \geq \lfloor \frac{n}{2} \rfloor + 1$ shares from honest parties, and by Lemma 3 all are correctly authenticated except with probability $\mathtt{neg}(\kappa)$. This is a majority of the shares, which uniquely determines the reconstructed value. Therefore either $s' = s''$ or $s' = \perp$.

To show Share termination, note that if some honest party P_i terminates Share, then P_i must have received the following: an authenticated share from D, a zero purity test result from the TTP, and a message from the TTP that the teleportation procedure has finished. Since P_i received an authenticated share from D, by Lemma 3 so do all honest parties except with probability $\mathtt{neg}(\kappa)$. Therefore every honest party sends zero purity test measurements to the TTP. The TTP returns the zero purity test results to every party which sends measurements, since otherwise it would have received fewer than $n-t$ measurements total and not sent a result to P_i. Finally, all honest parties receive the message from the TTP that the teleportation is over. All other steps can be done without waiting.

Composition in parallel with itself using the same dealer key k_D follows from a similar argument as above, except we rely on the following properties holding even when composed in a parallel repetition of the respective protocol which shares the key k: security of the classical TTP, security of the authentication scheme, and the properties of the Verified Authentication of Zeros protocol (see Lemma 3). □

5.3 Asynchronous Verifiable Quantum Secret Sharing

An asynchronous verifiable quantum secret sharing (AVQSS) scheme is described by a pair of protocols $(\mathsf{Share}, \mathsf{Reconstruct})$. In the first protocol, Share, a designated party called the dealer D distributes a private input s among the set of parties. In the second protocol, Reconstruct, the parties jointly participate and a designated receiver R obtains the secret.

Definition 5 (Asynchronous verifiable quantum secret sharing). *Let* $(\mathsf{Share}, \mathsf{Reconstruct})$ *be a pair of protocols for n parties, where a designated party D, called the* dealer*, has a private input quantum state s for* Share*, and each honest party that completes* Share *subsequently invokes* Reconstruct *with its local output of* Share*, and a designated* receiver *R outputs a quantum state upon terminating. We say that* $(\mathsf{Share}, \mathsf{Reconstruct})$ *is a* (t, ε)-secure asynchronous verifiable quantum secret sharing scheme *if the following holds with probability at least $1 - \varepsilon$ whenever up to t parties are corrupted:*

- *Termination:*
  - *If D is honest, then every party eventually terminates* Share*.*
  - *If some honest party terminates* Share*, then all honest parties eventually terminate* Share*.*
  - *If all honest parties terminate* Share *and start* Reconstruct*, then (an honest) R eventually outputs a quantum state.*
- *Privacy: If D is honest, the view of the adversary is independent of s.*
- *Correctness: If D is honest, then if R outputs a state at the end of* Reconstruct*, the state is s.*
- *Commitment: Even if D is corrupted, if all honest parties terminate* Share*, their joint views defines a unique state s', such that if R outputs a state after* Reconstruct*, the state has* $\mathtt{neg}(\kappa)$ *trace distance to s'.*

Protocol Description. The main idea of the protocol is to proceed in two levels. In the first level, the secret is shared using AWQSS, and then in the second level each share is shared again with AWQSS. This allows the sharing scheme to be such that corrupted parties cannot arbitrarily contribute wrong shares during the reconstruction step; they can only refuse to contribute shares. In contrast to the protocol for AWQSS, we will define a fixed set of size at least $n - t$ parties that will contribute shares during the reconstruction. Within this set there are at least $n - 2t > n/2$ honest parties, and, because corrupted parties cannot contribute wrong shares, the reconstructor R can safely recover the secret.

Functionality AVQSS Classical TTP

Initialization step. The TTP receives keys $(k, \vec{x})$ from the dealer D and keys $(k_i, \vec{x}_i)$ from each party P_i. It sets $S = \varnothing$, sets $\mathsf{counts}[i] = 0$ for $i = 1, \ldots, n$, and sets $S_{\mathsf{purity}} = \varnothing$.

WSS Level-One Execution Upon receiving $(\mathsf{WSS1Complete}, j)$ from a player P_i for the first time, the TTP does the following:

if $|S| < n - t$ and $j \notin S$ **then**
  $\mathsf{counts}[j] \leftarrow \mathsf{counts}[j] + 1$
  if $\mathsf{counts}[j] = t + 1$ **then**
    $S \leftarrow S \cup \{j\}$
    if $|S| = n - t$ **then**
      Send $(\mathsf{WSS1}, S)$ to all parties.
    end if
  end if
end if

Zero Purity Test Upon receiving $(\mathsf{AQVSSShareCheck}, m)$ from P_i, where m is a set of measurements and P_i has not sent a AQVSSShareCheck message before, the TTP does the following:

Verify the authentication codes on m using the classical authentication keys. If it fails, discard this message. Otherwise, continue:
if $|S_{\mathsf{purity}}| < n - t$ **then**
  $S_{\mathsf{purity}} \leftarrow S_{\mathsf{purity}} \cup \{i\}$
  if $|S_{\mathsf{purity}}| = n - t$ **then**
    Perform the zero purity test, storing the result in result.

6: Send (AQVSSShareCheck, result) to each P_j for $j \in S$.
7: **end if**
8: **else**
9: Perform the purity test s times in the computational and Fourier bases using m.
10: Send (AQVSSShareCheck, acc) to P_i if this test accepts and if result = acc. Otherwise send the message (AQVSSShareCheck, rej) to P_i.
11: **end if**

Teleportation and Sharing Termination Upon receiving teleportation measurements from the dealer D, the TTP does the following:

1: Applies the measurements to complete the teleportation by transforming the classical authentication keys.
2: Remove the dealer's level-one AWQSS authentication keys by modifying the level-two AWQSS authentication keys.
3: Inform all parties that the sharing is over.

Reconstruction The TTP sets $S_{\mathsf{recon}} = \varnothing$, then participates in R's VSS execution to share half an EPR pair.

Upon receiving (VSSReconstruct, m_i) from party P_i, where m_i are measurements, the TTP does the following:

1: Update the keys $\vec{x}_i$ according to the transversally-applied teleportation circuit.
2: Attempt to reconstruct the level-one shares of the teleportation measurements using m_i (which consists of the level-two shares of the teleportation measurements).
3: **for** Each new successful reconstruction of P_j's level-two sharing **do**
4: Set $S_{\mathsf{recon}} \leftarrow S_{\mathsf{recon}} \cup \{j\}$
5: Save the reconstructed level-one share (which P_j shared as its level-two value).
6: **end for**
7: **if** $|S_{\mathsf{recon}}| \geq \frac{n}{2}$ **then**
8: Use the reconstructed level-one shares in S_{recon} to reconstruct the measurement result of the teleportation circuit.
9: Send the reconstructed teleportation measurement result to R.
10: **end if**

Protocol AVQSS

We describe the protocol from the point of view of party P_i. The dealer D aims to share the contents of a register R_D with the other players.

Initialization step. D chooses a random key k_D. It generates $2s + 2$ $|0\rangle$ states for $s = \mathsf{poly}(\kappa)$ and encodes them using quantum Shamir secret sharing with threshold $\lfloor \frac{n}{2} \rfloor + 1$ and n shares. For each encoded $|0\rangle$ state, send the i-th share to P_i using the key k_D and freshly random x in the Verified Authentication protocol. Let $\vec{x}$ be the vector containing each x which is used. The dealer D sends the keys $(k_D, \vec{x})$ to the TTP.

Sharing Execution.

1: Let $R_1, \ldots, R_{2s+2}$ be the registers containing the (authenticated) shares which P_i received from D. Share each state using AWQSS and key k^i. Let $R_{i,j}$ denote the register containing the j-th share.
2: For each sharing that terminated from party P_j, send (WSS1Complete, j) to the TTP.
3: Upon receiving (WSS1, S) from the TTP, wait until all the sharings in S terminate.
4: Perform the measurements for the zero purity test on the shares from parties in S. Store the measurements in m.
5: Send (AVQSSShareCheck, m) to the TTP.
6: Wait to receive (AQVSSShareCheck, result) from the TTP.
7: **if** result is acc **then**
8: Generate an EPR pair share using the two remaining shares. Send the share of the first half of the EPR pair to the dealer.
9: **else**
10: Abort
11: **end if**

Teleportation

1: D waits until it receives $\lfloor \frac{n}{2} \rfloor + 1$ correctly authenticated shares from the parties, then uses them to reconstruct the EPR pair half (this is possible since it knows the keys and can update them according to the EPR construction, which is deterministic).
2: D uses the reconstructed EPR pair half to teleport the contents of R_D, sending the measurement results to the TTP.
3: The TTP informs all parties that the sharing is over.

Reconstruct

1: The reconstructor R shares half of an EPR pair using AVQSS, where the other parties use the same key as before.
2: Each party P_i transversally teleports the state shared by D to R using R's shared EPR pair. They send the (authenticated) measurements to the TTP.
3: R receives the teleportation measurements from the TTP and applies them to finish the teleportation.

Lemma 5. *Protocol AVQSS is a $(t, \mathtt{neg}(\kappa))$-secure asynchronous verifiable quantum secret sharing scheme for all $t < \frac{n}{4}$. Furthermore, it maintains these properties even when composed in parallel with another AVQSS instance using the same keys k_D and k_i for each party P_i (but independent $\vec{x}_D$, $\vec{x}_i$).*

Proof. First, note that the AWQSS scheme satisfies termination, privacy, correctness, and commitment with all but $\mathtt{neg}(\kappa)$ probability by Lemma 4 since $t < \frac{n}{4}$. By union bound, these properties hold for all of the AWQSS instances with all but $\mathtt{neg}(\kappa)$ probability, so in the following we condition on these properties holding for all instances.

Furthermore, these properties hold even if AWQSS is composed in parallel with another AWQSS instance using the same dealer key k_D, by Lemma 4. A similar statement applies for the zero purity test and Verified Authentication of Zeros protocol (see Lemma 3). Additionally, the classical TTP and (ideal) authentication codes compose in parallel with themselves.

We begin by considering the case where the dealer D is honest and show that all honest parties eventually terminate Share. We first prove that the TTP will eventually send a level-two AWQSS sharing session identifier $(\mathsf{WSS1}, S)$ to all parties, then show that all honest parties terminate every sharing session in S. This is sufficient to show termination since every honest party which terminates all sharing sessions in S will participate in the $|0\rangle$ purity test. This implies that the TTP will receive at least $n - t$ measurements for this test and will send the results to all parties. At this point, all honest parties terminate.

To see that the TTP will eventually send a level-two sharing session identifier $(\mathsf{WSS1}, S)$ to all parties, first observe that every honest party terminates the initial AWQSS sharing phase by the properties of AWQSS since D is honest. Then, in the second level of sharing every honest party terminates the AWQSS sharing phase for each honest level-two dealer (of which there are $n - t$, since all completed the level-one sharing). Therefore, the TTP receives confirmation of termination of the level-two sharings from every honest party (total $n - t$) for every honest level-two sharing (total $n - t$). This is sufficient to trigger the thresholds for the TTP to decide and send S. Finally, to show honest termination of every session in S, note that the TTP receives more confirmations of termination $(t+1)$ for each session in S than there are corrupted parties (t). Therefore, for each session in S some honest party terminated that session, and so by the properties of AWQSS all honest parties eventually terminate that session.

Privacy with an honest dealer follows naturally from the privacy of the underlying AWQSS. The honest parties all share the level-one AWQSS shares they receive from the dealer in their level-two AWQSS. Let P_i be such an honest party, who receives share sh_i from the original dealer in the level-one AWQSS. By the privacy guarantee of the level-two AWQSS, the adversary obtains no information about sh_i. Hence, the privacy of the overall AVQSS scheme follows from the privacy of the level-one AWQSS.

Correctness with an honest dealer follows from commitment with a corrupt dealer, which we prove later, as well as correctness of the underlying AWQSS. Since the honest parties share the level-one shares as their level-two secrets, the joint view of the honest parties defines the unique reconstruction value guaranteed by commitment to be the original secret shared by the honest dealer.

Now consider the case of a corrupt dealer. To show sharing termination, observe that if some honest party terminates Share, then it received both a zero purity test result and a set S from the TTP. Both of these are eventually received by all honest parties, since the TTP sends them to all parties. The only other place an honest party might hang is while waiting for all level-two AWQSS in S to complete. Since the TTP constructs S to contain level-two AWQSS sessions which more than t parties have reported

termination on, for each session in S it holds that some honest party must have terminated that session. Therefore, all other honest parties will terminate each session in S as well.

To show reconstruction termination with a corrupt dealer, first recall that if some honest party starts Reconstruct, it must have terminated Share, and so all honest parties will eventually terminate Share. If R is honest, then all honest parties terminate the AVQSS step in reconstruction where R shares half an EPR pair. All honest parties will perform the teleportation step, so the TTP will receive enough authenticated measurements to complete the teleportation.

Finally, we show commitment regardless of whether or not the dealer is corrupted. To do this, we show that the TTP will successfully reconstruct precisely the teleportation measurement which would have been made if the teleportation circuit was not evaluated transversally. By the commitment property of the underlying AWQSS protocol, it holds that for each level-two sharing by P_i there is a unique reconstruction state s'_i such that the TTP either reconstructs s'_i or fails to reconstruct P_i's level-two sharing.[9] Furthermore, since the zero purity test accepted, these values s'_i must be consistent shares of the same secret teleportation measurement s'_D except with probability $\mathtt{neg}(\kappa)$ (recall that the zero purity test establishes that these encode a state within small trace distance of $|0\rangle$, which are then transformed together by transversal operations). The measurement translates the guarantee of low trace given by the zero purity test into a small failure probability. In this case, there is a unique measurement s'_D which the TTP can reconstruct. By the correctness of the level-two AWQSS sharings, the TTP will at a minimum be able to reconstruct all level-two secrets dealt by honest parties, which corresponds to the level-one AWQSS shares they hold. Since the collection of level-one AWQSS shares held by honest parties uniquely determines the dealer's secret teleportation measurement s'_D, the TTP can therefore successfully reconstruct s'_D using only the reconstructed level-two secrets dealt by honest parties, except with $\mathtt{neg}(\kappa)$ probability. The low trace distance guarantee again translates into a small failure probability since the shares are measured. The successful reconstruction by R then follows from the correctness of the teleportation circuit. □

5.4 Asynchronous Toffoli Gate Computation

As discussed in Section 3.4 and Appendix A.1, the secret-sharing and authentication schemes allow for transversal evaluation of Clifford operations on the secret. In order to allow Toffoli operations, which form a universal gate set together with Clifford operations, the parties will also share a set of Toffoli states. Toffoli gates can be performed using Clifford operations and an ancillary Toffoli state [BCG$^+$06, Appendix F]. The shared Toffoli state generation protocol constructs one or more (for sake of exposition, we describe the single version) Toffoli states which are shared amongst the parties.

At a high level, the shared Toffoli state generation protocol requires each player to send a set of Toffoli states. These sets are then tested for polynomial closeness to Toffoli states using quantum tomography and cut-and-choose techniques. This results in a set of states which are polynomially close to a set of Toffoli states with respect to trace distance. Finally, one of the sets which passes the test is further purified to achieve an exponentially good Toffoli state using techniques from fault-tolerant quantum computation [ABO97].

Functionality Toffoli Sharing TTP

Let $d = \mathsf{poly}(\kappa)$ and let $m = 3d + 1$.
Initialization. Set $S_{i,\mathsf{cac}} = \varnothing$ and $S_{i,\mathsf{tom}} = \varnothing$ for $i = 1, \ldots, n$, set $\mathsf{indices}_i = \varnothing$, and set set $\mathsf{result} = \perp$.

Cut-and-Choose. Upon receiving the message (Cut-and-Choose, j) from party P_i, do the following:

1: $S_{i,\mathsf{cac}} \leftarrow S_{i,\mathsf{cac}} \cup \{i\}$
2: **if** $\mathsf{result} = \perp$ and $S_{i,\mathsf{cac}}| = n - t$ **then**
3: Sample random indices for testing all but m of the Toffoli states and store them in $\mathsf{indices}_j$.
4: Send (Cut-and-Choose, $j, \mathsf{indices}_j$) to all parties.
5: **end if**

[9] After performing the teleportation circuit, the teleportation measurement and the shared value are identical and are classical.

Tomography. Upon receiving the message $(\mathsf{Tomography}, j, m_{i,j})$ from party P_i, do the following:

if $\mathsf{indices}_j \neq \varnothing$ **then**
  $S_{i,\mathsf{tom}} \leftarrow S_{i,\mathsf{tom}} \cup \{i\}$
  if $\mathsf{result} = \perp$ and $|S_{i,\mathsf{tom}}| = \lfloor \frac{n}{2} \rfloor + 1$ **then**
    Reconstruct the tomography measurements for the states sent by P_j and selected in $\mathsf{indices}_j$.
    if the tomography test for Toffoli states in $\mathsf{indices}_j$ is accepting **then**
      Set $\mathsf{result} = j$.
      Send $(\mathsf{ToffoliIndex}, \mathsf{result})$ to all parties.
    end if
  end if
end if

Protocol Toffoli Sharing

Let $d = \mathsf{poly}(\kappa)$ and let $m = 3d + 1$. We describe the protocol from the point of view of party P_i.
Execution.

Prepare and share the following states using AVQSS: $\mathsf{poly}(d)$ Toffoli states for a sufficiently large polynomial and $3m$ ancillas initialized to $|0\rangle$.
for Each set of AVQSS (associated with a party P_j) which P_i terminated **do**
  Send (Cut-and-Choose, j) to the TTP.
end for
for Each message (Cut-and-Choose, $j, \mathsf{indices}_j$) received from the TTP **do** Run a state tomography on the states from P_j indicated by $\mathsf{indices}_j$ to test if they are indeed Toffoli states. Let $m_{i,j}$ be the resulting set of measurements.
  Send $(\mathsf{Tomography}, j, m_{i,j})$ to the TTP.
end for
Receive a message $(\mathsf{ToffoliIndex}, j)$ from the TTP. The rest of the steps refer to the states shared by P_j.
Using the shared ancillas, encode three shared $|0\rangle$ states using quantum Shamir secret sharing with m shares and reconstruction threshold $2d + 1$.
for $k = 1, \ldots, m$ **do**
  Use the k-th remaining shared Toffoli state to apply a Toffoli operation on the k-th coordinates of the three encoded $|0\rangle$ states.
end for
Decode the resulting encoded Toffoli state. This may require correcting errors in the encoding.

Lemma 6. *For all $t < \frac{n}{4}$, with all but negligible probability every honest party terminates the Toffoli Sharing protocol and holds a share of a state with $\mathtt{neg}(\kappa)$ trace distance from a Toffoli State.*

Proof. We condition on the soundness of the tomography test and the AVQSS properties holding, which both occur with all but negligible probability.

To see termination, observe that all honest parties will eventually terminate the AVQSS instances dealt by honest parties. Therefore the classical TTP will eventually receive enough $(n - t > \lfloor \frac{n}{2} \rfloor + 1)$ shares of tomography measurements for a set of states polynomially close to Toffoli states to select an index. All other operations are local.

It remains to show that every honest party holds a share of a state which is close to a Toffoli state in trace distance. Observe that the quantum Shamir secret sharing used can correct up to $\frac{d}{2}$ errors since it tolerates up to d erasures. The fact that there are less than $\frac{d}{2}$ errors follows from a straightforward cut-and-choose argument. This requires that the tested states are independent of the cut-and-choose indices. Observe that the cut-and-choose indices for testing a party P_j's set of sharings are chosen *after* the TTP receives $n - t$ messages confirming termination of P_j's set of sharings. This is sufficient for AVQSS commitment to occur, since it must hold even if only the parties sending termination confirmation messages participate in reconstruction. Observe that AVQSS commitment occurs for each dealer before the cut-and-choose indices for that dealer are chosen. □

This protocol can be easily extended to create multiple Toffoli states by increasing the initial number of states shared by each party.

5.5 Asynchronous Multiparty Quantum Computation

We combine the previous tools together to construct an asynchronous multiparty quantum computation (AMQPC) protocol. At a high level, the parties first construct a set of secret-shared Toffoli states which will later allow Toffoli gates to be performed on secret-shared states. They provide their inputs via AVQSS and use the classical TTP to aid in deciding a core-set of inputs. Using the shared Toffoli states, they transversally evaluate the circuit on the selected inputs. Finally, they reconstruct the states on the output wire(s) for each party.

Functionality AMPQC Classical TTP

Initialization step. The TTP receives keys $(k_i, \vec{x}_i)$ from each party P_i. It sets $S = \varnothing$ and sets $\mathsf{counts}[i] = 0$ for $i = 1, \ldots, n$.

Core-Set Agreement Upon receiving a new message (Core-Set, j) from a player P_i, the TTP does the following:

if $|S| < n - t$ and $j \notin S$ **then**
  $\mathsf{counts}[j] \leftarrow \mathsf{counts}[j] + 1$
  if $\mathsf{counts}[j] = t + 1$ **then**
    $S \leftarrow S \cup \{j\}$
    if $|S| = n - t$ **then**
      Send (Core-Set, S) to all parties.
    end if
  end if
end if

Output For each participant P_j, participate in the reconstruction of P_j's output wire with P_j as the reconstructor.

Protocol AMPQC

Initialization Each party P_i sends its authentication keys $(k_i, \vec{x}_i)$ to the TTP. **Execution**

1. P_i participates in the generation of shared Toffoli states protocol.
2. P_i shares its input using VSS and simultaneously acts as a receiver in the other VSS instances. In the other VSS instances, P_i uses the same authentication key k_i as it sent to the TTP. The other part of the authentication key for each instance is according to $\vec{x}_i$ and is distinct for each VSS instance.
3. For each sharing that terminated from party P_j, send (Core-Set, j) to the TTP.
4. Upon receiving (Core-Set, S) from the TTP, wait until all sharings in S terminate.
5. Evaluate the circuit C transversally on the shares of inputs from S with the help of the TTP.
6. **for** Each party P_j **do**
7. Participate in the reconstruction of P_j's output wire, where P_j is the reconstructor.
8. **end for**
9. P_i outputs the result of reconstructing P_i's output wire.

Theorem 3. *Protocol AMPQC t-securely computes any circuit C for all $t < \frac{n}{4}$.*

Proof. The simulator honestly participates in the protocol, using input $|0\rangle$ for each honest party, until the classical TTP[10] sends the message (Core-Set, S) to all parties. At this point, send S to the ideal functionality. Additionally, use $t + 1$ of the shares held by honest parties to reconstruct the corrupted parties' inputs, then send them to the ideal functionality. Participate honestly in the computation of the circuit C and in the VQSS reconstructions where honest parties act as reconstructors. For the VQSS reconstructions where corrupted parties act as reconstructors, the simulator forces the outputs prescribed by the ideal functionality. Let $|y_j\rangle$ be the ouput specified for party P_j by the ideal functionality. To force P_j to reconstruct $|y_j\rangle$, the simulator first participates honestly in the VQSS of P_j's EPR half. It locally

[10] Recall the the protocol uses a classical TTP. This is distinct from the *quantum* TTP specified in the ideal functionality.

splits $|y_j\rangle$ using a VQSS where only honest parties participate, then instructs the honest parties to teleport that state instead of teleporting P_j's actual output wire. It intercepts the teleportation measurement sent to P_j by the classical TTP and replaces it with a teleportation measurement computed using only the honest parties' measurements (who are teleporting $|y_j\rangle$.

Indistinguishability of the view produced by the above simulator follows from the following sequence of hybrids, where hybrid 0 is the real world.

1. The first hybrid simulator "forces" the output defined by the honest shares by using only the honest shares to reconstruct the teleportation result in the output phase. It intercepts and replaces the classical TTP's teleportation result message. Indistinguishability from the real world follows from the commitment property of the VQSS, since the reconstructed output is the same regardless of which set of shares is used to reconstruct.
2. The next hybrid simulator forces the output prescribed by the ideal functionality. It reconstructs the adversary's input using the honest parties' shares and sends this to the ideal functionality, receiving back an output state for each adversarial party. To complete the transcript, it shares these states locally amongst the (simulated) honest parties, which replaces the honest shares of the adversary's output wires, and proceeds as in the previous transcript. Indistinguishability follows from the secrecy of the VQSS scheme.
3. The final simulator replaces the honest inputs with $|0\rangle$ states. Indistinguishability follows from the secrecy of the VQSS scheme. □

6 Impossibility Result for Asynchronous Verifiable Quantum Secret Sharing

In this section, we prove the following impossibility result, which shows that our AVQSS protocol from Section 5.3 is optimal with respect to the corruption threshold when the error is sub-constant.

Theorem 4. *There exists a constant $\varepsilon^\star > 0$ such that there does not exist a $(t, \varepsilon^\star)$-asynchronous verifiable quantum secret sharing scheme among n parties for any corruption threshold $t \geq n/4$.*

The main intuition behind Theorem 4 is as follows: In a protocol with four parties, not only can the corrupted party introduce a faulty share, but the adversarial asynchronicity of the network may also cause an additional honest party to be locked out of the protocol. This means that reconstruction must proceed with access to only two out of four shares. However, this is not possible in the quantum setting, even approximately.

We formally prove the theorem statement in a sequence of lemmas. Let D be the dealer and $P_1, \ldots, P_4$ be four parties. Towards a contradiction, assume that there is an asynchronous verifiable quantum secret sharing scheme $(\mathsf{Share}, \mathsf{Reconstruct})$ for dealer D and four parties, where the dealer and at most one of the parties may be corrupted.[11] We show that $(\mathsf{Share}, \mathsf{Reconstruct})$ yields an *approximate quantum erasure-correcting code* (QECC) of length $n = 4$ correcting 2 erasures, which we show cannot exist at the end of this section. We now state the definition of approximate QECCs.

Definition 6 (Approximate quantum erasure-correcting code). *Consider a pair of maps $(\mathsf{Enc}, \mathsf{Dec})$ operating as follows: The encoder Enc maps an arbitrary qubit $|\psi\rangle \in \mathbb{C}^2$ to a possibly mixed state $\mathsf{Enc}(|\psi\rangle)$ over $\bigotimes_{i=1}^n \mathcal{H}_i$. Let S_T denote the subsystem of $\mathsf{Enc}(|\psi\rangle)$ restricted to $\bigotimes_{i\in T} \mathcal{H}_i$. Then, the decoder Dec on input S_T and T outputs a possibly mixed state $\mathsf{Dec}(S_T, T)$ over $\mathbb{C}^2$ with associated density matrix ρ_T. We say $(\mathsf{Enc}, \mathsf{Dec})$ is an ε-*approximate (n, t)-quantum erasure-correcting code *(QECC) if for every subset $T \subseteq [n]$ of size $|T| \geq t$ and every qubit $|\psi\rangle$ it holds that*

$$D(|\psi\rangle\langle\psi|, \rho_T) \leq \varepsilon.$$

We are now ready to state our first main lemma.

Lemma 7. *Every $(1, \varepsilon)$-asynchronous verifiable quantum secret sharing scheme among 4 parties is also an ε-approximate $(4, 2)$-QECC.*

[11] We make the argument for an external dealer, but the argument also holds when the dealer is one of the four parties.

To prove Lemma 7, consider the set $\{P_3, P_4\}$ (the argument holds for any set of size 2). We show that this set is authorized. Consider the following scenarios:

Scenario 1. Parties execute the sharing phase of the protocol honestly where the dealer D has input s, except P_2 who crashes from the start.

Scenario 2. Parties execute the sharing phase of the protocol honestly where the dealer D has input s, but all messages from P_2 to any other party are delayed.

Scenario 3. Similar to Scenario 2, all messages from P_2 to any other party are delayed, but in addition, the dealer D is corrupted and does not send any message to P_2. Instead, the dealer D internally emulates an execution with input s and also internally emulates P_2. All messages between D, P_1, P_3, P_4 are delivered.

Scenario 4. Similar to the above Scenario 3, but P_1 is also corrupted and does not send any message to P_2. Concretely, the corrupted D and corrupted P_1 jointly emulate party P_2, and do not send any message to the real P_2. However, all messages between D, P_1, P_3, P_4 are delivered.

Lemma 8. *All honest parties terminate the sharing phase in Scenario* 4*.*

Proof. From Scenario 1, we know that since the dealer D is honest, it is guaranteed that all the remaining parties P_1, P_3, P_4 terminate the sharing phase. Given that in Scenario 2 the view of the parties P_1, P_3, P_4 is identical to that of Scenario 1, they terminate the sharing phase as well. Moreover, since there is an honest party that terminates the sharing phase, P_2 terminates as well. In Scenario 3, the view of the parties P_1, P_3, P_4 is identical to that of Scenario 2, and by the same argument as above, all honest parties terminate the sharing phase. Finally, in Scenario 4, the view of the parties P_3 and P_4 is identical to that of Scenario 3, and therefore all honest parties terminate the sharing phase. □

Lemma 9. *The views of parties P_3 and P_4 after the Sharing phase uniquely define the secret s in Scenario* 4*.*

Proof. By the previous lemma, all honest parties terminate the sharing protocol in Scenario 4. Moreover, by termination of AVSS, if all honest parties start the reconstruction protocol, the output reconstructed is close in trace distance to some fixed value s'. Since the dealer and P_1 did not send any message to P_2, his state can be emulated among the parties P_3 and P_4. Therefore, this defines a decoding map mapping the states of P_3 and P_4 to a state that is close in trace distance to s'. Moreover, $s' = s$, since there is an adversarial strategy which lets to the reconstructed value to be s: D and P_1 act honestly in the reconstruction, while delaying messages from P_2. This is because it corresponds exactly to a reconstruction from Scenario 1, where the dealer is honest and the reconstructed value is close in trace distance to s, yielding the desired statement. □

Combining Lemmas 8 and 9 yields Lemma 7. We now prove that ε-approximate $(4, 2)$-QECCs as presented in Definition 6 do not exist for small enough ε.

Lemma 10. *There is no* 0.01*-approximate* $(4, 2)$*-QECC.*

Proof. This statement follows from an approximate version of the no-cloning theorem. We provide a full proof for completeness.

Suppose we have a 0.01-approximate $(4, 2)$-QECC $(\mathsf{Enc}, \mathsf{Dec})$. Consider the mixed states

$$S_1 = \begin{cases} |0\rangle, & \text{with probability } 1/2, \\ |1\rangle, & \text{with probability } 1/2, \end{cases}$$

and

$$S_2 = \begin{cases} |+\rangle, & \text{with probability } 1/2, \\ |-\rangle, & \text{with probability } 1/2, \end{cases}$$

where $\{|+\rangle, |-\rangle\}$ is the Fourier basis. We wish to argue that we cannot distinguish between S_1 and S_2. To this end, we make use of the following version of the Holevo-Helstrom theorem.

Lemma 11 (Holevo-Helstrom theorem [O'D15, Theorem 3.4]). *In general, the best success probability to discriminate between two mixed states with associated density matrices ρ_1 and ρ_2 is given by $\frac{1}{2} + \frac{1}{2}D(\rho_1, \rho_2)$.*

Since S_1 and S_2 have the same associated density matrix $\rho = \frac{1}{2}|0\rangle\langle 0| + \frac{1}{2}|1\rangle\langle 1|$, Lemma 11 implies that the advantage of distinguishing between S_1 and S_2 is 0. On the other hand, we will show that access to $(\mathsf{Enc}, \mathsf{Dec})$ allows us to distinguish between S_1 and S_2 with positive advantage, leading to a contradiction. Consider a mixed state S which is either S_1 or S_2 with probability $1/2$, and let $S' = \mathsf{Enc}(S)$. We can then apply the decoder to subsystems $\{1,2\}$ and $\{3,4\}$ of S' to obtain states $\widetilde{S}_{1,2} = \mathsf{Dec}(S'_{1,2}, \{1,2\})$ and $\widetilde{S}_{3,4} = \mathsf{Dec}(S'_{3,4}, \{3,4\})$. By the properties of this coding scheme, conditioned on the event that S is the pure state $|\psi\rangle$ we have that

$$D(|\psi\rangle\langle\psi|, \widetilde{S}_{1,2}), D(|\psi\rangle\langle\psi|, \widetilde{S}_{3,4}) \leq 0.01. \tag{2}$$

We now measure $\widetilde{S}_{1,2}$ and $\widetilde{S}_{3,4}$ in the computational basis, leading to measurement outcomes $(M_{1,2}, M_{3,4}) \in \{0,1\}^2$. Then, we have

$$\Pr[(M_{1,2}, M_{3,4}) = (0,0) \text{ or } (M_{1,2}, M_{3,4}) = (1,1) | S = S_1] \geq 0.99^2 > 0.98.$$

This holds since the probability of outcome 0 when measuring $\widetilde{S}_{1,2}$ in the computational basis is at least $1 - D(|0\rangle\langle 0|, \widetilde{S}_{1,2}) \geq 0.99$ when $|\psi\rangle = |0\rangle$ by (2) and the properties of the trace distance, and likewise for $\widetilde{S}_{3,4}$ in place of $\widetilde{S}_{1,2}$ and also outcome 1 in place of 0. On the other hand, we have

$$\Pr[(M_{1,2}, M_{3,4}) = (0,0) \text{ or } (M_{1,2}, M_{3,4}) = (1,1) | S = S_2] \leq 2 \cdot 0.51^2 < 0.53.$$

This follows by a union bound and the fact that the probability of seeing outcome 0 when measuring $\widetilde{S}_{1,2}$ in the computational basis is at most $1/2 + D(|+\rangle\langle +|, \widetilde{S}_{1,2}) \leq 0.51$ when $|\psi\rangle = |+\rangle$ by (2) and the properties of the trace distance, and likewise for $\widetilde{S}_{3,4}$ in place of $\widetilde{S}_{1,2}$ and also outcome 1 in place of 0. Consequently, we can distinguish between the events $S = S_1$ and $S = S_2$ with probability strictly larger than $1/2$ using the measurements $(M_{1,2}, M_{3,4})$, as desired. □

Combining Lemmas 7 and 10 yields an impossibility of AVQSS with respect to $n = 4$ parties and $t = 1$ corruptions.

Lemma 12. *There exists a constant $\varepsilon^\star > 0$ such that there does not exist a $(1, \varepsilon^\star)$-asynchronous verifiable quantum secret sharing scheme among 4 parties.*

Theorem 4, i.e., the generalization to arbitrary $t \geq n/4$ then follows from standard emulation arguments by observing that any AVQSS for $t \geq n/4$ implies an AVQSS among 4 parties (denote them $Q_1, \ldots, Q_4$) and 1 corruption. To see this, simply partition the n parties (from the n-party AVQSS) into 4 sets $S1, \ldots, S4$, each of size (at most) t. In order to construct an AVQSS protocol among the four parties, simply let each party Q_i emulate the set S_i. The resulting protocol is a protocol among $Q_1, \ldots, Q_4$ and tolerating 1 corruption, since the n-party protocol tolerates any corrupted set S_i.

Finally, we remark that Theorem 4 combined with the analysis of our AVQSS scheme in Section 5.3 implies that it is also impossible to construct an AWQSS scheme with negligible error when the corruption threshold t satisfies $t \geq n/4$. It suffices to observe that if we instantiate our AVQSS scheme with a low-error AWQSS scheme withstanding some corruption threshold t, then the proof of Lemma 5 directly yields a low-error AVQSS scheme withstanding any corruption threshold

$$t' \leq \min(t, \lceil n/2 \rceil - 1).$$

Since Theorem 4 implies that we must have $t' < n/4$, it must also be the case that $t < n/4$. This leads to the following result.

Corollary 1. *There exists a constant $\varepsilon^\star > 0$ such that there does not exist a $(t, \varepsilon^\star)$-asynchronous weak quantum secret sharing scheme among n parties for any corruption threshold $t \geq n/4$.*

7 Classical TTP in the Asynchronous Setting

In this section we show how to realize a classical TTP, a.k.a. a reactive functionality, in the asynchronous model with eventual delivery of messages. To the best of our knowledge, previous works considered constructing reactive functionalities in the synchronous setting [KMTZ13] or in the asynchronous setting without guaranteed delivery [CLOS02].

Description of the Reactive Functionality. A reactive computation can be specified as an ordered sequence of secure function evaluations which can maintain a joint state. The state used to evaluate any function is passed on to the subsequent functions. In contrast to the synchronous model where without loss of generality one can assume that all parties give input and the functionality receives all inputs, in the asynchronous model it is important to consider functionalities that perform a certain evaluation without having received all the inputs. In order to model the set of parties' inputs that should be taken into account for the computation, we include a predicate $Q : \mathcal{P} \to \{0,1\}$ on the set of parties.[12] In typical secure function evaluation protocols, the predicate evaluates to 1 if and only if the set of parties has a certain size (e.g., at least $n - t$, where t is the corruption threshold).

More concretely, the computation is described as a vector of pairs function-predicate

$$\mathbf{g} = ((f_1, Q_1), \ldots, (f_m, Q_m)),$$

where each function f_λ takes as input a vector of values $\{0,1\}^* \cup \{\perp\}$, a uniform random value r from some domain R and a state vector $\mathbf{S}_\lambda$ containing the inputs and randomness used to evaluate the functions $f_1, \ldots, f_{\lambda-1}$. Then, f_λ outputs a vector of values.

We describe the reactive functionality $\mathcal{F}_{\mathsf{reactive}}$. It has parameters the vector $\mathbf{g}$ and the set of parties $\mathcal{P}$. For each function f_λ, the functionality can receive an input $x_{i,\lambda}$ at any point in time, as long as the function was not evaluated. The function f_λ is evaluated as soon as the set of parties T_λ who provided input satisfies the predicate $Q_\lambda(T_\lambda) = 1$. A detailed description can be found below.

Functionality $\mathcal{F}_{\mathsf{reactive}}$

Parameters. Vector $\mathbf{g} = ((f_1, Q_1), \ldots, (f_m, Q_m))$ and party set $\mathcal{P}$ of size n.

Initialization. Local inputs $x_{1,1}, \ldots, x_{n,m}$ and outputs $y_{1,1}, \ldots, y_{n,m}$, initialized to $\perp$. Initial vector state $\mathbf{S}_0 = (\perp, \ldots, \perp)$. Initial sets of parties $T_1, \ldots, T_m = \varnothing$.
We denote $\mathbf{x}_\lambda = (x_{1,\lambda}, \ldots, x_{n,\lambda})$ and $\mathbf{y}_\lambda = (y_{1,\lambda}, \ldots, y_{n,\lambda})$.

1: Upon receiving input $(\mathsf{input}, \lambda, v)$ from party $P_i \in \mathcal{P}$ and $\lambda \in \{1, \ldots, m\}$, if $\mathbf{y}_\lambda$ has not been set, then set $x_{i,\lambda} = v$, add P_i to T_λ and output $(\mathsf{input}, P_i, \lambda)$ to the adversary.
2: **if** $Q_\lambda(T_\lambda) = 1$ **then**
3: Choose a random value $r_\lambda \leftarrow_\$ R$
4: $\mathbf{y}_\lambda = f_\lambda(x_\lambda, \mathbf{S}_{\lambda-1}, r_\lambda)$.
5: Set $\mathbf{S}_\lambda = (\mathbf{x}_1, \ldots, \mathbf{x}_\lambda, r_1, \ldots, r_\lambda)$.
6: **end if**
7: Upon receiving $(\mathsf{output}, \lambda)$ from party $P_i \in \mathcal{P}$ and $\lambda \in \{1, \ldots, m\}$, if $\mathbf{y}_\lambda$ has been set, output $y_{i,\lambda}$ to P_i.

In order to formalize security, we can make use of a composable framework that models asynchronous protocols with eventual delivery [CGHZ16, LLM+20].

Definition 7. *A protocol Π secure computes the function vector $\mathbf{g}$ among parties in $\mathcal{P}$ if it UC-realizes functionality $\mathcal{F}_{\mathsf{reactive}}$ with parameters $\mathbf{g}$ and $\mathcal{P}$.*

Realizing $\mathcal{F}_{\mathsf{reactive}}$. We sketch how one can adapt previous protocols (e.g., [BKR94, PCR10]) for secure function evaluation to realize $\mathcal{F}_{\mathsf{reactive}}$. Such protocols achieve information-theoretic (and post-quantum) UC security and tolerate up to $t < n/3$ corruptions.

These protocols follow the traditional sharing-based paradigm: parties secret-share their inputs, proceed in a gate-by-gate fashion to compute shares of the output wires from shares of the input wires, and

[12] For simplicity, we assume that the predicate only depends on the set of parties, as this will be enough for us. However, one can consider more general predicates.

reconstruct the outputs towards the corresponding parties. We are mainly interested in the input phase. There, parties distribute their inputs using an asynchronous verifiable secret sharing (AVSS) scheme. Because the network is asynchronous, some sharings terminate earlier than others, and therefore parties need to agree on when to proceed to the computation phase. For that, parties run a *core-set agreement* [BKR94, BCG93] protocol to agree on a core-set of parties of size at least $n - t$ whose inputs are taken into the computation (all other parties' inputs are ignored). That is, the predicate Q that is considered evaluates to 1 when the core-set has size at least $n - t$ parties.

In order to realize $\mathcal{F}_{\mathsf{reactive}}$, we need to address two aspects. The first is to design a core-set agreement for a general *monotone* predicate Q [13]. The second is to keep track of the internal state of $\mathcal{F}_{\mathsf{reactive}}$.

Both aspects are addressed by changing the input stage as follows. At the execution of (f_λ, Q_λ), each party P_i has shares of the internal state of $\mathcal{F}_{\mathsf{reactive}}$ (initially default shares of $\mathbf{S}_0$). Upon receiving an input v, P_i uses AVSS to distribute his input. Then, in the core-set agreement, n Byzantine agreement protocols $\mathsf{BA}_1, \ldots, \mathsf{BA}_n$ are run, one for each party. Every time the AVSS from party P_j terminates, P_i inputs 1 to BA_j. Every time BA_j outputs 1, it adds P_j to T^i_λ. P_i then waits until the set of parties T^i_λ satisfies $Q_\lambda(T^i_\lambda) = 1$. If so, P_i inputs 0 to all remaining BAs and waits for all BAs to terminate, before proceeding to the computation phase.

Since Q_λ is monotone, the set of inputs taken into account for the computation satisfies Q_λ (as it contains at least the set T^i_λ). This leads to the desired lemma:

Lemma 13. *There is a post-quantum secure protocol that securely computes* $\mathbf{g} = ((f_1, Q_1), \ldots, (f_m, Q_m))$*, where* $Q_1, \ldots, Q_m$ *are monotone predicates, among parties in* $\mathcal{P}$*,* $|\mathcal{P}| = n$*, for any* $t < n/3$ *corruptions.*

References

[AB97] D. Aharonov and M. Ben-Or. Fault-tolerant quantum computation with constant error. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '97, page 176–188, New York, NY, USA, 1997. Association for Computing Machinery.

[ABEM17] Dorit Aharonov, Michael Ben-Or, Elad Eban, and Urmila Mahadev. Interactive proofs for quantum computations. *arXiv e-prints*, page arXiv:1704.04487, April 2017.

[ABG+21] Amit Agarwal, James Bartusek, Vipul Goyal, Dakshita Khurana, and Giulio Malavolta. Post-quantum multi-party computation. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021*, pages 435–464, Cham, 2021. Springer International Publishing.

[ABO97] D. Aharonov and M. Ben-Or. Fault-tolerant quantum computation with constant error. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '97, page 176–188, New York, NY, USA, 1997. Association for Computing Machinery.

[ACC+21a] Bar Alon, Hao Chung, Kai-Min Chung, Mi-Ying Huang, Yi Lee, and Yu-Ching Shen. Round efficient secure multiparty quantum computation with identifiable abort. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021*, pages 436–466, Cham, 2021. Springer International Publishing.

[ACC+21b] Bar Alon, Hao Chung, Kai-Min Chung, Mi-Ying Huang, Yi Lee, and Yu-Ching Shen. Round efficient secure multiparty quantum computation with identifiable abort. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 436–466, Virtual Event, August 2021. Springer, Heidelberg.

[AGM21] Gorjan Alagic, Tommaso Gagliardoni, and Christian Majenz. Can you sign a quantum state? *Quantum*, 5:603, 2021.

[BCG93] Michael Ben-Or, Ran Canetti, and Oded Goldreich. Asynchronous secure computation. In *25th ACM STOC*, pages 52–61. ACM Press, May 1993.

[BCG+02] Howard Barnum, Claude Crépeau, Daniel Gottesman, Adam Smith, and Alain Tapp. Authentication of quantum messages. In *43rd FOCS*, pages 449–458. IEEE Computer Society Press, November 2002.

[BCG+06] Michael Ben-Or, Claude Crépeau, Daniel Gottesman, Avinatan Hassidim, and Adam Smith. Secure multiparty quantum computation with (only) a strict honest majority. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 249–260, 2006. Full version at `http://crypto.cs.mcgill.ca/~crepeau/PDF/BCGHS06.pdf`.

[BCKM21] James Bartusek, Andrea Coladangelo, Dakshita Khurana, and Fermi Ma. On the round complexity of secure quantum computation. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021*, pages 406–435, Cham, 2021. Springer International Publishing.

[13] If $T \subseteq T'$ and $Q(T) = 1$, then $Q(T') = 1$.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *20th ACM STOC*, pages 1–10. ACM Press, May 1988.

[BKLL20] Erica Blum, Jonathan Katz, Chen-Da Liu-Zhang, and Julian Loss. Asynchronous byzantine agreement with subquadratic communication. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 353–380. Springer, Heidelberg, November 2020.

[BKR94] Michael Ben-Or, Boaz Kelmer, and Tal Rabin. Asynchronous secure computations with optimal resilience (extended abstract). In Jim Anderson and Sam Toueg, editors, *13th ACM PODC*, pages 183–192. ACM, August 1994.

[BS20] Nir Bitansky and Omri Shmueli. Post-quantum zero knowledge in constant rounds. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *52nd ACM STOC*, pages 269–279. ACM Press, June 2020.

[BW16] Anne Broadbent and Evelyn Wainewright. Efficient simulation for quantum message authentication. In Anderson C. A. Nascimento and Paulo S. L. M. Barreto, editors, *Information Theoretic Security - 9th International Conference, ICITS 2016, Tacoma, WA, USA, August 9-12, 2016, Revised Selected Papers*, volume 10015 of *Lecture Notes in Computer Science*, pages 72–91, 2016.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *20th ACM STOC*, pages 11–19. ACM Press, May 1988.

[CGHZ16] Sandro Coretti, Juan A. Garay, Martin Hirt, and Vassilis Zikas. Constant-round asynchronous multi-party computation based on one-way functions. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 998–1021. Springer, Heidelberg, December 2016.

[CGL99] Richard Cleve, Daniel Gottesman, and Hoi-Kwong Lo. How to share a quantum secret. *Phys. Rev. Lett.*, 83:648–651, Jul 1999.

[CGS02] Claude Crépeau, Daniel Gottesman, and Adam Smith. Secure multi-party quantum computation. In *34th ACM STOC*, pages 643–652. ACM Press, May 2002.

[CHL21] Annick Chopard, Martin Hirt, and Chen-Da Liu-Zhang. On communication-efficient asynchronous MPC with adaptive security. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part II*, volume 13043 of *LNCS*, pages 35–65. Springer, Heidelberg, November 2021.

[Cho20] Ashish Choudhury. Optimally-resilient unconditionally-secure asynchronous multi-party computation revisited. Cryptology ePrint Archive, Report 2020/906, 2020. `https://eprint.iacr.org/2020/906`.

[CHP13] Ashish Choudhury, Martin Hirt, and Arpita Patra. Asynchronous multiparty computation with linear communication complexity. In *Proceedings of the 27th International Symposium on Distributed Computing - Volume 8205*, DISC 2013, page 388–402, Berlin, Heidelberg, 2013. Springer-Verlag.

[CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *34th ACM STOC*, pages 494–503. ACM Press, May 2002.

[Coh16] Ran Cohen. Asynchronous secure multiparty computation in constant time. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part II*, volume 9615 of *LNCS*, pages 183–207. Springer, Heidelberg, March 2016.

[CP15] Ashish Choudhury and Arpita Patra. Optimally resilient asynchronous MPC with linear communication complexity. In *Proceedings of the 2015 International Conference on Distributed Computing and Networking*, ICDCN '15, New York, NY, USA, 2015. Association for Computing Machinery.

[DGJ+20] Yfke Dulek, Alex B. Grilo, Stacey Jeffery, Christian Majenz, and Christian Schaffner. Secure multi-party quantum computation with a dishonest majority. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 729–758. Springer, Heidelberg, May 2020.

[DL09] Ivan Damgård and Carolin Lunemann. Quantum-secure coin-flipping and applications. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 52–69. Springer, Heidelberg, December 2009.

[DNS12] Frédéric Dupuis, Jesper Buus Nielsen, and Louis Salvail. Actively secure two-party evaluation of any quantum operation. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 794–811. Springer, Heidelberg, August 2012.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

[HBB99] Mark Hillery, Vladimír Bužek, and André Berthiaume. Quantum secret sharing. *Physical Review A*, 59(3):1829, 1999.

[HLM16] Patrick Hayden, Debbie W. Leung, and Dominic Mayers. The universal composable security of quantum message authentication with key recyling. *arXiv e-prints*, page arXiv:1610.09434, October 2016. Presented at QCRYPT 2011.

[HNP05] Martin Hirt, Jesper Buus Nielsen, and Bartosz Przydatek. Cryptographic asynchronous multi-party computation with optimal resilience (extended abstract). In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 322–340. Springer, Heidelberg, May 2005.

[HNP08] Martin Hirt, Jesper Buus Nielsen, and Bartosz Przydatek. Asynchronous multi-party computation with quadratic communication. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP 2008, Part II*, volume 5126 of *LNCS*, pages 473–485. Springer, Heidelberg, July 2008.

[HSS11] Sean Hallgren, Adam Smith, and Fang Song. Classical cryptographic protocols in a quantum world. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 411–428. Springer, Heidelberg, August 2011.

[IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 572–591. Springer, Heidelberg, August 2008.

[KMTZ13] Jonathan Katz, Ueli Maurer, Björn Tackmann, and Vassilis Zikas. Universally composable synchronous computation. In Amit Sahai, editor, *TCC 2013*, volume 7785 of *LNCS*, pages 477–498. Springer, Heidelberg, March 2013.

[LLM+20] Chen-Da Liu-Zhang, Julian Loss, Ueli Maurer, Tal Moran, and Daniel Tschudi. MPC with synchronous security and asynchronous responsiveness. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 92–119. Springer, Heidelberg, December 2020.

[LN11] Carolin Lunemann and Jesper Buus Nielsen. Fully simulatable quantum-secure coin-flipping and applications. In Abderrahmane Nitaj and David Pointcheval, editors, *AFRICACRYPT 11*, volume 6737 of *LNCS*, pages 21–40. Springer, Heidelberg, July 2011.

[NC10] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.

[O'D15] Ryan O'Donnell. Lecture notes on quantum computation. Lecture 17: Discriminating quantum states. `https://www.cs.cmu.edu/~odonnell/quantum15/lecture17.pdf`, November 2015. Last accessed February 15, 2022.

[PCPR09] Arpita Patra, Ashish Choudhary, and Chandrasekharan Pandu Rangan. Simple and efficient asynchronous byzantine agreement with optimal resilience. In *Proceedings of the 28th ACM symposium on Principles of distributed computing*, pages 92–101, 2009.

[PCR10] Arpita Patra, Ashish Choudhary, and C. Pandu Rangan. Efficient statistical asynchronous verifiable secret sharing with optimal resilience. In Kaoru Kurosawa, editor, *ICITS 09*, volume 5973 of *LNCS*, pages 74–92. Springer, Heidelberg, December 2010.

[PCR15] Arpita Patra, Ashish Choudhury, and C. Pandu Rangan. Efficient asynchronous verifiable secret sharing and multiparty computation. *Journal of Cryptology*, 28(1):49–109, January 2015.

[PSR02] B. Prabhu, K. Srinathan, and C. Pandu Rangan. Asynchronous unconditionally secure computation: An efficiency improvement. In Alfred Menezes and Palash Sarkar, editors, *INDOCRYPT 2002*, volume 2551 of *LNCS*, pages 93–107. Springer, Heidelberg, December 2002.

[RB89] Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In *21st ACM STOC*, pages 73–85. ACM Press, May 1989.

[Sho96] Peter W. Shor. Fault-tolerant quantum computation. In *37th FOCS*, pages 56–65. IEEE Computer Society Press, October 1996.

[SR00] K. Srinathan and C. Pandu Rangan. Efficient asynchronous secure multiparty distributed computation. In Bimal K. Roy and Eiji Okamoto, editors, *INDOCRYPT 2000*, volume 1977 of *LNCS*, pages 117–129. Springer, Heidelberg, December 2000.

[Unr10] Dominique Unruh. Universally composable quantum multi-party computation. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 486–505. Springer, Heidelberg, May / June 2010.

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd FOCS*, pages 80–91. IEEE Computer Society Press, November 1982.

Appendix

A Polynomial-Based Authentication Scheme

We exploit the efficient polynomial-based authentication scheme from [BCG+06] and based on [AB97]. Fix parameters r and $m = 2r + 1$. Choose a prime $p \in (m, 2m)$ and m distinct evaluation points $\alpha_1, \ldots, \alpha_m \in \mathbb{F}_p$. Given a classical key (x, k) with $k \in \{-1, 1\}^m$ and $x = (x_{1,1}, x_{1,2}, \ldots, x_{m,1}, x_{m,2}) \in (\mathbb{F}_p^2)^m$, we authenticate a basis state $|\beta\rangle$ with $\beta \in \mathbb{F}_p$ as

$$\mathsf{Auth}_{x,k}(|\beta\rangle \otimes |0\rangle^{\otimes m-1}) = E_x\left(p^{-r/2} \sum_{f \in \mathbb{F}_p[x]:\deg(f)\leq r, f(0)=\beta} |k_1 \cdot f(\alpha_1), \ldots, k_m \cdot f(\alpha_m)\rangle\right), \quad (3)$$

where the sum is over all polynomials f over $\mathbb{F}_p$ of degree at most r satisfying $f(0) = \beta$. The operator E_x encrypts the i-th system corresponding to α_i by applying $X^{x_{i,1}} Z^{x_{i,2}}$, where X and Z are generalized Pauli gates over $\mathbb{F}_p$. The decoding procedure $\mathsf{Dec}_{x,k}$ first removes the encryption E_x, then removes the sign changes caused by k, and finally performs Lagrange interpolation to recover the message. Intuitively, this scheme is secure because an adversary must apply an attack which affects at least r qupits, otherwise the attack is detected by the error-correction properties of the code. However, since the attacker does not know the key, its attack will yield consistent evaluations of a polynomial of degree at most r only with exponentially small probability. The following lemma formalizes the security properties of the polynomial code.

Lemma 14 ([BCG+06, ABEM17, HLM16]). *The scheme* $(\mathsf{Auth}_{x,k}, \mathsf{Dec}_{x,k})$ *is a* $\mathsf{neg}(\kappa)$*-QA scheme when* $m = \kappa$.

As in [BCG+06], our protocol will require that the polynomial authentication scheme remains secure in a setting where the adversary has access to several states authenticated with the same k but encrypted under *independently sampled x's*. More precisely, consider an extension of the real-world process detailed above where the register M is replaced by s registers $M_1, \ldots, M_s$ and the contents of register M_i are authenticated using $\mathsf{Auth}_{x^{(i)},k}$ (as in (3)) for $i \in [s]$ with $k \leftarrow \{-1, 1\}^m$ and $x^{(1)}, \ldots, x^{(s)}$ sampled independently and uniformly from $(\mathbb{F}_p^2)^m$. Then, a unitary transformation $U_{C_1 \ldots C_s R}$ is applied to the authenticated states and the side information. Likewise, the ideal-world process is augmented by having s flags $\mathsf{flag}_1, \ldots, \mathsf{flag}_s$ behaving as before. Then, we say $(\mathsf{Auth}_{x,k}, \mathsf{Dec}_{x,k})$ is *ε-secure in the s-parallel authentication setting* if, analogously to (1), these two experiments are ε-close in trace distance. The following lemma is implicit in [BCG+06].

Lemma 15 ([BCG+06]). *The scheme* $(\mathsf{Auth}_{x,k}, \mathsf{Dec}_{x,k})$ *is* $\mathsf{neg}(\kappa)$*-secure in the s-parallel authentication setting when* $m = \kappa$ *and* $s \leq \mathsf{poly}(\kappa)$.

Lemma 16 ([BCG+06]). *The scheme* $(\mathsf{Auth}_{x,k}, \mathsf{Dec}_{x,k})$ *is* $\mathsf{neg}(\kappa)$*-secure in the s-parallel authentication setting when* $m = \kappa$ *and* $s \leq \mathsf{poly}(\kappa)$.

A.1 Operations on Authenticated States

For completeness, we discuss how to apply the generalized gates described in Section 3.2 to states which have been authenticated via the polynomial code from Section 3.4 by applying transversal operations on the different components of the authenticated state and relying on the classical TTP. This has already been shown in [BCG+06] (see also [ABEM17, Section 2.5.1 and Appendix D]).

Write $x = (x_1, x_2)$ with $x_i = (x_{i,1}, \ldots, x_{i,m})$ for $i = 1, 2$. With respect to the X gate, note that

$$\mathsf{Auth}_{x,k}(|\beta\rangle) = \mathsf{Auth}_{x',k}(|\beta + 1\rangle),$$

where $x' = (x'_1, x_2)$ and $x'_{1,j} = x_{1,j} + 1$ for $j \in [m]$. Therefore, we can apply the X gate to $|\beta\rangle$ simply by having the TTP update x to x'. To apply the controlled SUM gate to two quantum states authenticated with keys (x, k) and (y, k) respectively, let $\widetilde{\mathrm{SUM}}$ denote the transversal operation on $2m$ registers consisting of applying SUM on registers i and $m + i$ for $i \in [m]$. Then, we have

$$\widetilde{\mathrm{SUM}}(\mathsf{Auth}_{x,k}(|\alpha\rangle) \otimes \mathsf{Auth}_{y,k}(|\beta\rangle)) = \mathsf{Auth}_{x',k}(|\alpha\rangle) \otimes \mathsf{Auth}_{y',k}(|\alpha + \beta\rangle),$$

where $x' = (x_1, x_2 - y_2)$ and $y' = (y_1 + x_1, y_2)$. Therefore, we may apply SUM to $|\alpha, \beta\rangle$ indirectly by applying SUM gates transversally on the $2m$ registers and having the TTP update the keys. This works so long as registers i and $m+i$ are held by the same party. Measurements in the computational basis may be applied directly on the authenticated state, yielding a string $(x_{1,1} + k_1 f(\alpha_1), \ldots, x_{1,m} + k_m f(\alpha_m))$ with f a polynomial of degree at most r, from which $f(0)$ can be recovered given knowledge of (x, k).

In order to apply the Fourier and Z gates, we introduce the notion of *interpolation coefficients*: Given distinct evaluation points $\alpha_1, \ldots, \alpha_m \in \mathbb{F}_p$, there exist (efficiently computable) coefficients $c_1, \ldots, c_m \in \mathbb{F}_p$ such that

$$\sum_{i=1}^{m} c_i f(\alpha_i) = f(0)$$

for all polynomials f over $\mathbb{F}_p$ of degree at most $m-1$. With respect to the Fourier gate F, consider the transversal operation $\widetilde{F} = F_{c_1} \otimes \cdots \otimes F_{c_m}$. Then, we have

$$\widetilde{F}\mathsf{Auth}_{x,k}(|\beta\rangle) = p^{-1/2} \sum_{\gamma \in \mathbb{F}_p} \omega_p^{\gamma \cdot \beta} \mathsf{Auth}_{x',k}(|\gamma\rangle),$$

where $x' = (x_2, x_1)$. Therefore, we may apply F by transversally applying $\widetilde{F}$ and having the TTP update x to x'. In order to apply the Pauli Z gate, it suffices to observe that

$$\mathsf{Auth}_{x,k}(|\beta\rangle) = \omega_p^{\beta} \mathsf{Auth}_{x',k}(|\beta\rangle),$$

where $x' = (x_1, x_2')$ with $x'_{2,i} = x_{2,i} + k_i \cdot c_i$. Therefore, it is enough to have the TTP update x as described.

Finally, note that in all operations above no information is leaked about the keys.

B Zero Purity Test

At several points in our protocols, we need to ensure that a set of received states are all properly authenticated $|0\rangle$ states using the quantum authentication scheme from Section 3.4. Ben-Or et. al. [BCG+06, Appendix C] introduce a simple zero purity test which allows us to achieve this with low error. Our protocols use their test in a black-box manner, which only requires local operations and interaction between one party and the classical TTP which holds the authentication keys.

For completeness, we formally define the guarantees provided by their protocol below. These are similar to those of quantum authentication in Section 3.4. Let $R = (R_1, \ldots, R_w)$ be a tuple of registers where each R_i holds a qupit, and denote the state held by these registers by ρ_R. Then, the zero purity test $\mathsf{ZPT}(\rho_R, s)$, where s is some parameter, is a protocol between a party P holding ρ_R and a classical TTP holding the authentication keys (x, k). The output of the protocol to P, which we denote by $\mathtt{ZPTout}(\rho_R, s)$, is placed into registers $R' = (R_1, \ldots, R_{w-2s})$ and a flag register flag F with basis states $\{|\mathsf{acc}\rangle, |\mathsf{rej}\rangle\}$. The remaining registers are destroyed. We require the following guarantees of such a protocol:

1. **Correctness:** If $\rho_R = \mathsf{Auth}_{x,k}(|0\rangle)^{\otimes w}$, then

$$\mathtt{ZPTout}(\rho_R, s) = \mathsf{Auth}_{x,k}(|0\rangle)^{\otimes(w-2s)} \otimes |\mathsf{acc}\rangle,$$

   i.e., the protocol succeeds and P holds correctly authenticated states, and the classical TTP holds the correct keys (x, k) for these states.
2. **Security:** Consider an ideal-world channel where a simulator $\mathcal{S}$ has access to a flag bit flag. If $\mathcal{S}$ sets $\mathsf{flag} = 0$, then the channel outputs $\Omega_{R'} \otimes |\mathsf{rej}\rangle\langle\mathsf{rej}|$, where $\Omega_{R'}$ is a placeholder state. Else, if $\mathsf{flag} = 1$, the channel outputs $\mathsf{Auth}_{x,k}(|0\rangle)^{\otimes(w-2s)} \otimes |\mathsf{acc}\rangle$. Denote the output of this process by $\mathtt{IDEALZPT}_{\mathcal{S}}$. Then, the protocol ZPT achieves *ε-security* if for any input state ρ_R there is a simulator $\mathcal{S}$ such that

$$D(\mathtt{ZPTout}(\rho_R, s), \mathtt{IDEALZPT}_{\mathcal{S}}) \leq \varepsilon.$$

The following lemma is proved and used in [BCG+06].

Lemma 17 ([BCG+06, Lemma C.1, adapted]). *There is an efficient zero purity test $\mathsf{ZPT}(\rho_R, s)$ between a party holding ρ_R and a classical TTP holding the authentication keys (x, k) that achieves $O(p^{-s})$-security. In particular, it achieves $\mathtt{neg}(\kappa)$-security when $s = \kappa$.*