Black-Box Timed Commitments from Time-Lock Puzzles

Hamza Abusalah[1] and Gennaro Avitabile[1]

IMDEA Software Institute, Madrid, Spain.
`{hamza.abusalah,gennaro.avitabile}@imdea.org`

Abstract. A Timed Commitment (TC) with time parameter t is hiding for time at most t, that is, commitments can be force-opened by any third party within time t. In addition to various cryptographic assumptions, the security of all known TC schemes relies on the sequentiality assumption of repeated squarings in hidden-order groups. The repeated squaring assumption is therefore a security bottleneck.
In this work, we give a black-box construction of TCs from any time-lock puzzle (TLP) by additionally relying on one-way permutations and collision-resistant hashing.
Currently, TLPs are known from (a) the specific repeated squaring assumption, (b) the general (necessary) assumption on the *existence of worst-case non-parallelizing languages* and indistinguishability obfuscation, and (c) any iteratively sequential function and the hardness of the circular small-secret LWE problem. The latter admits a plausibly post-quantum secure instantiation.
Hence, thanks to the generality of our transform, we get i) the first TC whose *timed* security is based on the the existence of non-parallelizing languages and ii) the first TC that is plausibly post-quantum secure.
We first define *quasi publicly-verifiable* TLPs (QPV-TLPs) and construct them from any standard TLP in a black-box manner without relying on any additional assumptions. Then, we devise a black-box commit-and-prove system to transform any QPV-TLPs into a TC.

1 Introduction

Time-lock puzzles (TLPs) introduced by Rivest, Shamir, and Wagner [RSW96] allow one to commit to a message by generating a message-dependent puzzle that ensures the message remains hidden for a certain time. In particular, a TLP consists of a puzzle generation algorithm Gen and a solving algorithm Solve. On input a message m and a time parameter t, Gen generates a puzzle z. On input an honestly-generated puzzle z, Solve retrieves the message m. A TLP must satisfy (1) *correctness:* Solve always recovers the correct message from honestly-generated puzzles (2) *hardness:* the message remains hidden for any massively parallel adversary running in time much less than t and (3) *efficiency:* while Solve runs in time t, Gen must run in time $\mathsf{polylog}(t)$.

In [RSW96], a very efficient construction whose security is based on the repeated squaring assumption is proposed. This assumption states that given a group element g, an integer t, and an RSA modulus N, it is hard to compute $g^{2^t} \mod N$. The puzzle of [RSW96] works as follows: $\mathsf{Gen}(t, m)$ outputs $z = (m + g^{2^t} \mod N, g, N)$, while Solve computes g^{2^t} to extract m. We stress that while correctness of TLPs guarantees solvability for *honestly generated* puzzles, it guarantees nothing for maliciously crafted puzzles. Furthermore, a TLP *per se* does not come with an algorithm that decides whether a TLP is honestly generated or not. For example, given the puzzle of [RSW96], one can retrieve m via t repeated squarings modular N, but cannot efficiently verify whether N is an RSA modulus or not.

A standard commitment scheme is a 2-phase interactive protocol between a sender and a receiver such that at the end of the *commit phase*, the sender is bound to a message m (binding) and the receiver learns nothing about m (hiding). In the *open phase*, the sender reveals m to the receiver, which either accepts or rejects. Boneh and Naor [BN00] introduced timed commitments (TCs) which add a time dimension to standard commitments and overcome the limitations of TLPs. In a TC with time parameter t, the *commit phase* can be *force-opened* in *sequential* time t without executing the efficient open phase. This is useful in scenarios where the sender refuses to run the open phase. As a result, the message m is only guaranteed to remain hidden for time at most t.

A TC must satisfy (standard) binding, *t-hiding*, *publicly-verifiable forced openings*, and *well-formedness*. *t-hiding* requires that a massively parallel (adversarial) receiver running in time much less than t learns nothing about the committed message. *Well-formedness* requires that if the commit phase terminates successfully, the receiver is guaranteed that the commitment can be forced-open in time t via a force-open algorithm. *Publicly-verifiable forced openings* guarantee that the force-open algorithm, in addition to m, outputs a proof π that allows anyone in possession of the transcript of the commit phase to *quickly* verify that m is the committed message. A TC must also be efficient, meaning that the commit phase must take time $\mathsf{polylog}(t)$.

Similar to TLPs hardness, t-hiding of TCs guarantees that the secrecy of m is preserved for a certain amount of time t. The main difference between TLPs and TCs is that, unlike TCs, TLPs fall short of providing public verifiability and well-formedness. That is, TLPs do not provide any means of verifying the correctness of solutions without re-solving the puzzle, and furthermore, puzzles are not guaranteed to be solvable in time t when maliciously generated.

Known TCs and their limitations. Various constructions of TCs [BN00, KLX20, TCLM21, CJ23] have been proposed. Some of them feature useful additional properties such as non-interactive commit phase and CCA security [KLX20, TCLM21, CJ23], or homomorphic properties [TCLM21, CJ23]. Later, we discuss these constructions in the related work section. Nevertheless, they all follow a similar blueprint. To achieve t-hiding, all these constructions rely, in a non-black box way, on the repeated-squaring-based puzzle of [RSW96]. They ensure well-formedness by proving in zero knowledge (ZK), with either a proof system for NP or with one tailored to the specific language of interest, that the TLP was computed correctly. Moreover, to achieve public verifiability, they crucially rely on repeated squaring. For example, a common technique to ensure public verifiability [TCLM21, CJ23, FKPS21, BDD$^+$21, BDD$^+$23] is to exploit proofs of exponentiation (e.g., [Pie19, Wes19]) which allow to efficiently check that a value y is equal to g^{2^t} modulo N without executing t squarings.

Consequently, *all known TC constructions are based on the single repeated squaring assumption.* This limitation is significant as it is known that repeated squaring is not necessary to obtain TLPs. Indeed, Bitansky et al. constructed TLPs based on the existence of worst-case non-parallelizing languages and succinct randomized encodings [BGJ$^+$16]. Non-parallelizing languages are necessary for timed-based primitives, such as TLPs and TCs, and are implied by the repeated squaring assumption [BGJ$^+$16, JMRR21].

1.1 Our Contributions

The state of affairs in which all known TCs rely on the single sufficient-but-not-necessary assumption of repeated squarings in unknown-order groups is rather unsatisfactory and creates a single point of failure. In this work, we diverge from all known designs of TCs and give a *black-box* transformation of *any* TLP into a TC. By black-box we mean that the TLP (and any other underlying cryptographic primitive) is used only as an oracle. There has been a significant amount of research dedicated to obtaining black-box constructions for various cryptographic primitives [IKLP06, CDMW09, PW09, LP12, GLOV12, Kiy14, HV16, KOS18, Kiy20, COS22]. The benefit of the black-box approach is immediate: basing TCs on TLPs in a black box manner widens the class of known constructions of TCs and allows translating advances in TLPs to TCs. Furthermore, black-box constructions have the advantage that their complexity does not depend on the complexity of the implementation of the underlying primitives[1].

More specifically, we make the following contributions:

- We provide a formal definition of timed commitments. Previous works either gave a rather high-level description of such properties [BN00], or were tailored to the non-interactive setting [KLX20, CJ23].
- Along the way, we define and construct quasi publicly verifiable time-lock puzzles (QPV-TLPs), a novel tool we believe could be of independent interest. Roughly, a QPV-TLP is a TLP where the receiver, if the puzzle is well-formed, is able to provide a convincing proof of correctness of the solution which can be quickly verified. We show how to lift any TLP to a QPV-TLP with black-box use of the TLP itself and without relying on any additional assumptions.
- Relying on QPV-TLPs, we construct a black-box TC assuming the existence of TLPs, collision-resistant hash functions, and one-way permutations. Our construction has a five-round commit phase and a non-interactive

[1] To better understand this aspect consider a TC which uses a generic (non-black-box) ZK protocol to prove that the circuit computing a TLP was correctly executed. The complexity of this TC would crucially depend on the number of gates of such circuit.

open phase. Additionally binding, well-formedness, and public verifiability of forced openings hold w.r.t. unbounded adversaries. Our construction does not require any setup.
– By weakening the well-formedness guarantee to computational, we can get a TC assuming the existence of TLPs and one-way functions. Since TLPs imply one-way functions [BGJ$^+$16], this shows that TLPs imply TCs. Thus, instantiating the TLP with the one of [BGJ$^+$16], we get the first TC whose timed security relies on the weakest complexity assumption of the existence of worst-case non-parallelizing languages. This comes at the cost of the strong cryptographic assumption of indistinguishability obfuscation [BGI$^+$01].
– Using the recent TLP of [AMZ24] as a building block in our transform, we get the first post-quantum secure TC: [AMZ24] constructs TLPs[2] whose security relies on any iteratively sequential function f and the hardness of the circular small-secret LWE problem. By instantiating f based on [LM23], their TLP is plausibly post-quantum secure.

1.2 Technical Overview

A straw-man approach to generically construct a TC from a TLP is the following. To achieve well-formedness, the sender proves with a generic ZK proof system that the TLP is correctly computed. To get public verifiability, the receiver uses a succinct non-interactive argument (SNARG) to prove that it solved the TLP correctly. SNARGs succinctness guarantees that verifying the proof is much faster than solving the puzzle. Apart from being non-black-box, this construction has other shortcomings. Indeed, known SNARGs (for P) require setups (e.g., [CJJ22, HJKS22, CGJ$^+$23, Kiy23]) and the only general assumption they are known from is iO [SW14, WW24]. Additionally, SNARGs are only computationally sound, thus constructing a TC following this approach will inherently give computational public verifiability, unlike our construction that achieves it against unbounded adversaries. We propose a different approach that guarantees public verifiability by lifting the TLP to a QPV-TLP, and proves well-formedness in a black-box way.

Publicly verifiable time-lock puzzles. Publicly verifiable time-lock puzzles (PV-TLPs), introduced by Freitag et al. [FKPS21], are an intermediate notion between TLPs and TCs. Unlike TCs, PV-TLPs are not guaranteed to be well-formed. However, after having solved the puzzle, whether the puzzle has a solution m or not, the solver produces a proof allowing anyone to efficiently verify that m is the correct solution or that the puzzle is malformed. PV-TLPs are a good candidate building block to construct TCs as the public verifiability of the TC essentially follows from that of the underlying PV-TLP. Unfortunately,

[2] In the TLP constructions of [AMZ24], if a one-time public-coin setup is allowed, the TLP generation runs in time $\mathsf{polylog}(t)$, otherwise it runs in time $\sqrt{t}$. The (in)efficiency of puzzle generation of the TLP translates to the (in)efficiency of the commit phase of our TC construction.

all known PV-TLPs [FKPS21, BDD$^+$21, BDD$^+$23] are based on the repeated squaring assumption and random oracles.

Quasi publicly verifiable time-lock puzzles. We notice that the property of providing a proof that is convincing even for invalid puzzles is not necessary to construct a TC. Indeed, to satisfy well-formedness the TC has to prove that the underlying TLPs are correctly generated, meaning that the receiver will not have to solve invalid puzzles. Therefore, we introduce the notion of quasi publicly verifiable time-lock puzzles (QPV-TLP), where the receiver is only guaranteed to produce a convincing proof when the puzzle is well-formed. Remarkably, this simple modification allows us to rely exclusively on the existance of TLPs. We point out that QPV-TLPs are different from *one-sided*[3] PV-TLPs, which are also only known from repeated squaring [FKPS21]. Even though they both provide sound proofs only for well-formed puzzles, one-sided PV-TLPs are additionally required to output accepting proofs for malformed puzzles while still remaining sound w.r.t. honest puzzles. We remark that our QPV-TLP is perfectly sound while PV-TLPs are only computationally sound [FKPS21, BDD$^+$21, BDD$^+$23].

Our QPV-TLP. Our QPV-TLP is very straightforward. Given any TLP, to generate a puzzle for a message m and time t, we compute $z = (z_0, z_1)$ where $z_0 := \mathsf{Gen}(t, m; r_0)$ and $z_1 := \mathsf{Gen}(t, r_0; r_1)$ with random coins r_0, r_1. To solve z, we solve z_0 and z_1 in parallel using Solve, and output m as the message, and r_0 as the proof. To verify the correctness of a claimed message, it suffices to check that $z_0 = \mathsf{Gen}(t, m; r_0)$. If a puzzle is correctly generated, the solver is always able obtain the message and convincing proof. Additionally, our QPV-TLP is perfectly sound. This follows from the fact that TLPs are injective, and thus z_0 cannot belong to the support of both $\mathsf{Gen}(t, m_0)$ and $\mathsf{Gen}(t, m_1)$ with $m_0 \neq m_1$.

TLPs as over-extractable commitments. As pointed out in [LPS17], TLPs can be related to the notion of *extractable* commitments [PW09] with *over-extraction.* A commitment scheme is said to be extractable if there exists an efficient extractor that, having black-box access to a malicious sender that successfully terminates the commit phase, extracts the committed value. An extractable commitment suffers from *over-extraction* if the extractor may output an arbitrary value when the commitment is invalid.[4] TLPs can be seen as over-extractable commitments where extraction is carried out in straight-line by brute-forcing the puzzle, but there is no guarantee on the extracted value if the puzzle is malformed.

Goyal et al. [GLOV12] constructed *weakly extractable* commitments which are extractable in a weaker sense meaning that the extraction can fail with probability 1/2 but without over-extraction. To commit to a message m, the

[3] One-sided PV-TLPs are a weaker primitive than PV-TLPs. They are defined and constructed in [FKPS21]. They use a one-sided PV-TLP and a random oracle to get a PV-TLP.

[4] We say that a commitment invalid if it does not have any valid decommitment. Otherwise, the commitment is valid.

sender computes a 2-out-of-2 secret sharing of m and commits to the shares separately using a statistically binding commitment. Then, the receiver challenges the sender to open one of the two commitments. The commit phase terminates successfully if and only if the sender correctly decommits the challenged commitment. To extract the committed value, it suffices to rewind the sender so that it decommits both commitments and to then reconstruct the message starting from the revealed shares. If the sender does not provide a valid decommitment, the extractor outputs $\perp$, denoting that the extraction was not successful.

As a starting point to get well-formedness, we can apply the above idea by replacing the statistically binding commitment with a TLP. When challenged to open a puzzle, the sender provides the message and randomness used to generate the puzzle. The receiver then checks that on input the message and the randomness, the TLP generation algorithm Gen gives the same puzzle. To extract the committed value, it suffices to solve the unopened puzzle. Unlike [GLOV12], our extraction strategy still suffers from over-extraction. Indeed, our extraction proceeds in straight-line and, since there is no efficient way to decide whether a TLP is malformed, it could be possible to extract a garbage value. Nonetheless, the probability of over-extracting is now at most 1/2.

A black-box proof of well-formedness. A natural idea to amplify the success probability of the extraction is parallel repetition. The committer samples λ different 2-out-of-2 secret sharings of m and commits to them in 2λ TLPs. Then, the receiver asks the sender to open one of the two commitments for each of the λ repetitions. However, a cheating sender could use a different message in different repetitions, making a successful commit phase meaningless. We address this issue by proving, using a black-box commit-and-prove system, that the TLPs across the different repetitions all commit to the same message m.

In a black-box commit-and-prove system, a sender commits to a message m so that later it can prove a predicate ϕ over the committed m in ZK with black-box use of cryptographic primitives. Constant-round black-box commit-and-prove systems [GLOV12, KOS18, Kiy20, COS22] can be constructed using the powerful MPC-in-the-head paradigm introduced by Ishai et al. [IKOS07]. Let us first briefly describe how to construct a black-box commit-and-prove system with constant soundness error using a 3-party 2-private MPC protocol (e.g., [GMW87]). The sender first commits to the shares of a 3-out-of-3 secret sharing of $m := m_1 \oplus m_2 \oplus m_3$. Then, to prove that the committed message satisfies a predicate ϕ, the sender runs in its head the MPC protocol computing the functionality $\phi'(m_1, m_2, m_3) := \phi(m_1 \oplus m_2 \oplus m_3)$ and commits to the resulting MPC views. Then, the receiver asks the sender to decommit to the shares m_i, m_j and to the views $\mathsf{view}_i, \mathsf{view}_j$ for random i, j with $i \neq j$. The receiver checks that (1) the decomittment information are correct, (2) view_i and view_j are consistent with each other and the output is 1 in both views, and (3) for $\delta \in \{i, j\}$, m_δ is the input of party δ in view_δ. Constant soundness follows from the binding of the commitment and the perfect correctness of the MPC, while ZK follows from the perfect 2-privacy of the MPC.

To get our TC, we start from the above protocol and modify it as follows: (1) to allow the receiver to force-open the commitment, we commit to the shares of the message with a TLP, (2) we repeat in parallel the above black-box commit-and-prove and define ϕ as the predicate that ensures that the *same* message is committed across *all* repetitions. To define the predicate, we use a clever technique proposed by Khurana et al. [KOS18]. Instead of committing to m, the sender commits to $m||r$, where r is some random value. The receiver replies with a random value α, and the sender sends $\gamma := r\alpha + m$ to the receiver. We set $\phi(m||r)$ as the predicate checking that γ is computed correctly. By the Schwartz-Zippel lemma, if $m||r \neq m'||r'$ then $r\alpha + m \neq r'\alpha + m'$ with overwhelming probability. Therefore, the fact that γ is a global value guarantees consistency of the message committed across the repetitions.

The resulting TC consists of five rounds, the first two of which are dedicated to defining ϕ and committing to (the shares of) $m||r$, while the last three consist of the black-box commit-and-prove showing that $\phi(m||r) = 1$ in all repetitions. Intuitively, well-formedness follows from the fact that an accepting commit phase indicates that the vast majority of the unopened TLPs across the repetitions are well-formed, and that they all open to the same message. Thus, when solving all the unopened puzzles, the receiver would retrieve the committed message in many repetitions. Additionally, the use of a QPV-TLP instead of a regular TLP immediately gives public verifiability of forced openings. The resulting TC achieves t-hiding only in the presence of an honest receiver. Intuitively, this is because the underlying black-box commit-and-prove is only honest-verifier ZK. Nonetheless, we can get full-fledged t-hiding with a minor modification to the protocol following the approach of Goldreich and Kahan [GK96] to lift an honest-verifier ZK proof to a ZK proof. Namely, we make the receiver commit, with a two-round statistically hiding commitment, to all of its challenges in the second round and to reveal them in the fourth round. Notice that this modification does not increase the number of rounds.

Public verifiability flavors and efficiency of forced openings. Our TC achieves a very strong flavor of public verifiability. Namely, given an arbitrary transcript of the commit phase, which can even be generated by a *computationally unbounded* malicious sender without interacting with a receiver, we require that it is infeasible to provide two valid force-open proofs for different messages. As a result, the receiver needs to solve all the puzzles during the force-open phase.

However, we can formulate a weaker but meaningful notion that enables a much more efficient force-open procedure. In particular, instead of an arbitrarily generated transcript of the commit phase, we consider a commitment from an accepting commit phase that the malicious sender run with an honest receiver. The well-formedness proof contained in the transcript guarantees that the vast majority of the repetitions contain the same message and, more importantly, the value γ can be used to check whether the message extracted from a repetition is consistent with the ones contained in the other (unsolved) puzzles. Thus, as soon as the receiver finds a repetition where the extracted message is consistent with γ, it will be sure that m is committed in the vast majority of repetitions. Therefore, the force-open procedure can output $m_i||r_i$ together with the proof of the QPV-TLP solved from the i-th repetition. Due to the perfect correctness of the MPC, it is easy to show that, with overwhelming probability, $\lambda - O(\log^2(\lambda))$ repetitions are correctly computed. Thus, the probability that the receiver samples more than $\log(\lambda)$ repetitions such that $\gamma \neq \alpha r_i + m_i$ or the unsolved QPV-TLP is malformed is negligible.

The above observation is also made in [KOS18], where the extractor of their black-box commit-and-prove stops rewinding the prover as soon as it can extract a message that is consistent with γ from one of the repetitions. Interestingly, this feature of the extraction strategy used in the security proofs of [KOS18] can be exploited to improve the concrete efficiency of our TC. Indeed, in our case the extraction happens in straight-line and it is part of the actual protocol.

TLPs imply TCs. In this work, we have focused on getting a TC whose security guarantees (except t-hiding) hold against unbounded adversaries. Our TC relies on TLPs, collision-resistant hashing, and one-way permutations.

However, by weakening the well-formedness guarantee to computational, it is possible to get a t-hiding TC without relying on statistically hiding commitments. For example, we can use the technique shown in Sect. 4.1.2 of [CLP20] (which is inspired by [Lin13]) to lift honest-verifier ZK arguments to ZK. The idea is to replace the challenge sent by the receiver in the fourth round of our honest-receiver construction by a coin tossing protocol. The result of such coin tossing is used as the receiver's challenge to finalize the remaining execution. The coin tossing protocol is based on the extractable commitment of Sect. 4 of [PW09] which allows the simulator to extract the receiver's share and bias the coin tossing to its advantage. In this coin tossing, the receiver first commits to a random string using the extractable commitment, the sender replies with its random share, and the receiver opens the commitment. The final challenge is computed by xoring the two shares.

The extractable commitment of [PW09] is a black-box construction from statistically binding commitments. By instantiating the statistically binding commitments in the resulting TC as a two-round protocol from one-way functions, we get a TC whose only assumption is the existence of TLPs, since TLPs imply one-way functions [BGJ$^+$16]. As a result, by instantiating the TLP with the one of [BGJ$^+$16] we get the first TC from general assumptions. Namely, the weakest timed assumption of the existence of worst-case non-parallelizing languages, in addition to indistinguishability obfuscation.

This alternative transformation to lift honest-verifier ZK to ZK leads to a constant increase of the number of rounds in the resulting TC. For the sake of clarity, since the main focus of our work is not to minimize the number of rounds, we have here illustrated a simple approach to base TCs solely on TLPs. Nonetheless, it is conceivable that ZK delayed-input black-box commit-and-prove [KOS18, COS22] could be applied in our setting as well, allowing to have TCs solely based on TLPs in five rounds. Indeed, in delayed-input ZK protocols the predicate to be proved can be decided even in the last round of the protocol. Therefore, the ZK delayed-input black-box commit-and-prove can start right away, in parallel with the exchange of messages used to decide the predicate in our TC. The constructions of [KOS18, COS22] are ZK arguments with four rounds, which become five when the underlying statistically binding commitments are instantiated from one-way functions instead of one-way permutations.

A remark on strongly extractable commitments. Strongly extractable commitments (sExtCom) are extractable commitments (ExtCom) that do not suffer from over-extraction [Kiy14]. In an sExtCom, if the extractor outputs $\perp$ it means that the commitment is invalid. We stress that in an sExtCom, a successful commit phase does not guarantee that the commitment is valid, which is required by TCs. Therefore, even though there exist black-box techniques to get an sExtCom from an ExtCom (e.g., [LP12, Kiy14]), they generally do not give a TC when the ExtCom is replaced with a TLP.

1.3 Related Work

Timed commitments. The first (interactive) timed commitment was proposed by Boneh and Naor [BN00]. Their TC relies on a less standard assumption, stronger than repeated squaring, called the generalized Blum-Blum-Shub assumption. Katz et al. [KLX20] introduced and constructed non-interactive TCs. Their TC is also non-malleable under a CCA-like notion. Their TC relies on repeated squaring and uses generic non-interactive zero knowledge (NIZK) to prove well-formedness. It requires a trusted setup and it is inefficient as committing takes time t. Additionally, it does not provide public verifiability of forced openings. Thyagarajan et al. [TCLM21] constructed the first CCA-secure non-interactive TC with transparent setup. Their construction is based on the repeated squaring assumption in class groups and is also linearly homomorphic. Its security is proven in the random oracle model. Chvojka and Jager [CJ23] proposed several CCA-secure non-interactive TCs, which all require a trusted setup. They provided constructions with either linear or multiplicative homomorphisms. All their constructions rely on the repeated squaring assumption.

Publicly verifiable time-lock puzzles. Freitag et al. [FKPS21] introduced publicly verifiable time-lock puzzles (PV-TLPs). Their PV-TLP is non-malleable[5] and it is based on *strong trapdoor* VDFs which are only known from repeated squaring [Pie19]. Additionally, they require the (non-programmable auxiliary input) random oracle model and a mild form of setup. Baum et al. [BDD$^+$21, BDD$^+$23] formalize the security of timed primitives in the UC framework [Can01]. They construct PV-TLPs based on the (programmable) random oracle model and strong trapdoor VDFs. They prove that their UC notion can only be achieved in the (programmable) random oracle model.

[5] We refer the reader to [FKPS21] for a comparison between their non-malleability notions and the CCA-like notion of [KLX20].

2 Preliminaries

Notation. We use $\mathbb{N}$ to denote the set of natural numbers. Let $n \in \mathbb{N}$, we denote the set $\{1, \dots, n\}$ as $[n]$. For a finite set S, we let $s \leftarrow S$ denote the sampling of s uniformly at random from S. Let A be a probabilistic algorithm, by $y \leftarrow A(x)$ we denote that y is the output of A on input x, while by $y = A(x; r)$ we denote that y is the output of A when it is run with input x and random coins r. We model multi-stage algorithms $\mathsf{A} := (\mathsf{A}_0, \mathsf{A}_1, \dots, \mathsf{A}_k)$ as *stateful* algorithms, i.e., A_i receives the *state* of A_{i-1}. However, when the state contains data of interest that we wish to highlight, we give it explicitly and keep the rest of the state implicit. For a tuple of interactive randomized algorithms (S, R), let

- $(y_s, y_r) \leftarrow \langle S(x_s) \leftrightarrow R(x_r)\rangle$ denote the interactive protocol between S and R on *private* inputs x_s and x_r, respectively. Their respective *private* outputs are y_s and y_r.
- $y_r \leftarrow \mathsf{out}_R(\langle S(x_s) \leftrightarrow R(x_r)\rangle)$ denote the *private* output of R in the protocol.
- $\mathsf{trans}(\langle S(x_s) \leftrightarrow R(x_r)\rangle)$ denote the transcript of the protocol, i.e., all the messages S and R exchanged during the protocol.
- $\mathsf{view}_R(\langle S(x_s) \leftrightarrow R(x_r)\rangle)$ denote the view of R in the protocol, i.e., all inputs, incoming and outgoing messages, and random coins of R during the protocol.

2.1 Timed Commitments

A timed commitment (TC) is an interactive protocol between a sender S and a receiver R defined as follows. We define some specific properties of TCs. *Soundness* guarantees that after a valid commit phase the force-open algorithm will give the committed message m and a convincing proof for m, *public verifiability* guarantees that, for any arbitrary commitment, it is unfeasible to provide two accepting force-open proofs w.r.t. different messages, and *t-hiding* guarantees that a massively parallel malicious receiver running in time less than t learns nothing about the committed message. We formalize the TC definition of [BN00] and give different flavors of some of their notions.

Definition 1. *A tuple of PT algorithms* $(\mathsf{S}, \mathsf{R}, \mathsf{FOpen}, \mathsf{FVerify})$ *where* $\mathsf{S} = (\mathsf{S_c}, \mathsf{S_o})$ *and* $\mathsf{R} = (\mathsf{R_c}, \mathsf{R_o})$ *are the (stateful)* sender *and (stateful)* receiver *respectively, is an* (interactive) timed commitment scheme *for message and commitment spaces* $\mathcal{M} = (\mathcal{M}_\lambda)_{\lambda\in\mathbb{N}}, \mathcal{C} = (\mathcal{C}_\lambda)_{\lambda\in\mathbb{N}}$ *such that* $\perp \notin \mathcal{M}_\lambda \cup \mathcal{C}_\lambda$ *if the following holds:*

1. *Commit Phase:* $(\mathrm{d}, \mathrm{c}) \leftarrow \langle \mathsf{S_c}(1^\lambda, t, m) \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle$*: On common input a security* λ *and a time parameter* t*, the receiver* $\mathsf{R_c}$ *runs an interactive protocol with the sender* $\mathsf{S_c}$*, which has a message* $m \in \mathcal{M}_\lambda$ *as additional input.* $\mathsf{R_c}$ *outputs a commitment* $\mathrm{c} \in \mathcal{C}_\lambda \cup \{\perp\}$ *and* $\mathsf{S_c}$ *outputs a decommitment* d*. (If* $\mathrm{c} \neq \perp$*, the commit phase is valid, otherwise, it is invalid.)*
2. *Open Phase:* $m' \leftarrow \mathsf{out}_{\mathsf{R_o}}(\langle \mathsf{S_o}(\mathrm{d}) \leftrightarrow \mathsf{R_o}(\mathrm{c})\rangle)$*: The sender* $\mathsf{S_o}$ *on input a decommitment* d*, and the receiver* $\mathsf{R_o}$ *on input a commitment* c*, run an interactive protocol which results in* $\mathsf{R_o}$ *outputting a message* $m' \in \mathcal{M}_\lambda \cup \{\perp\}$*. (*$m' \neq \perp$ *indicates a valid open phase, otherwise, it is invalid.)*

3. $\underline{(m,\pi) \leftarrow \mathsf{FOpen}(1^\lambda, 1^t, \mathrm{c})}$: *On input a security parameter* λ*, a time parameter* t*, and a commitment* c*, it outputs a message/proof pair* (m,π)*.*
4. $\underline{b := \mathsf{FVerify}(1^\lambda, t, \mathrm{c}, m, \pi)}$: *On input a security parameter* λ*, a time parameter* t*, a commitment* c*, a message* m*, and a proof* π*,* $\mathsf{FVerify}$ *accepts* $(b=1)$ *or rejects* $(b=0)$*.*

We require correctness of honest and forced openings, soundness, public verifiability, t-hiding, and statistical binding:

– *Correctness of honest openings: For every* $\lambda, t \in \mathbb{N}$ *and* $m \in \mathcal{M}_\lambda$*,*

$$\Pr\left[m' = m \,\middle|\, \begin{array}{l}(\mathrm{d},\mathrm{c}) \leftarrow \langle \mathsf{S_c}(1^\lambda, t, m) \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle \\ m' \leftarrow \mathsf{out}_{\mathsf{R_o}}(\langle \mathsf{S_o}(\mathrm{d}) \leftrightarrow \mathsf{R_o}(\mathrm{c})\rangle)\end{array}\right] = 1 \ .$$

– *Correctness of force openings: For every* $\lambda, t \in \mathbb{N}$ *and* $m \in \mathcal{M}_\lambda$*,*

$$\Pr\left[\begin{array}{l}\mathsf{FVerify}(1^\lambda, t, \mathrm{c}, m', \pi) = 1 \wedge \\ m' = m\end{array} \,\middle|\, \begin{array}{l}(\mathrm{d},\mathrm{c}) \leftarrow \langle \mathsf{S_c}(1^\lambda, t, m) \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle \\ (m',\pi) \leftarrow \mathsf{FOpen}(1^\lambda, 1^t, \mathrm{c})\end{array}\right] = 1 \ .$$

– *Efficiency:* FOpen *runs in time* $t\cdot\mathsf{poly}(\lambda)$*, while* $\mathsf{FVerify}$ *as well as* $\mathsf{S_c}, \mathsf{S_o}, \mathsf{R_c}, \mathsf{R_o}$ *run in time* $\mathsf{poly}(\log t, \lambda)$*.*

– *Soundness: For every* $\lambda, t \in \mathbb{N}$ *and every* unbounded *malicious (stateful) sender* $\tilde{\mathsf{S}} := (\tilde{\mathsf{S}}_\mathsf{c}, \tilde{\mathsf{S}}_\mathsf{o})$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\Pr\left[\begin{array}{l}\mathrm{c} \neq \bot \wedge \\ (\mathsf{FVerify}(1^\lambda, t, \mathrm{c}, m, \pi) = 0 \\ \vee\, (m' \neq \bot \wedge m' \neq m))\end{array} \,\middle|\, \begin{array}{l}(\tilde{\mathrm{d}},\mathrm{c}) \leftarrow \langle \tilde{\mathsf{S}}_\mathsf{c} \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle \\ m' \leftarrow \mathsf{out}_{\mathsf{R_o}}(\langle \tilde{\mathsf{S}}_\mathsf{o}(\tilde{\mathrm{d}}) \leftrightarrow \mathsf{R_o}(\mathrm{c})\rangle) \\ (m,\pi) \leftarrow \mathsf{FOpen}(1^\lambda, 1^t, \mathrm{c})\end{array}\right] \leq \mathsf{negl}(\lambda) \ .$$

– *Public verifiability: For every* $\lambda, t \in \mathbb{N}$ *and every* unbounded *adversary* A*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\Pr\left[\begin{array}{l}\mathsf{FVerify}(1^\lambda, t, \mathrm{c}, m_0, \pi_0) = 1 \wedge \\ \mathsf{FVerify}(1^\lambda, t, \mathrm{c}, m_1, \pi_1) = 1 \wedge \\ m_0 \neq m_1\end{array} \,\middle|\, (\mathrm{c}, m_0, \pi_0, m_1, \pi_1) \leftarrow \mathsf{A}\right] \leq \mathsf{negl}(\lambda) \ .$$

– *Statistical binding: For every* $\lambda, t \in \mathbb{N}$ *and every* unbounded *malicious (stateful) sender* $\tilde{\mathsf{S}} := (\tilde{\mathsf{S}}_\mathsf{c}, \tilde{\mathsf{S}}_\mathsf{o}, \tilde{\mathsf{S}}'_\mathsf{o})$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\Pr\left[\begin{array}{l}\mathrm{c} \neq \bot \wedge \\ m, m' \neq \bot \wedge \\ m \neq m'\end{array} \,\middle|\, \begin{array}{l}(\tilde{\mathrm{d}},\mathrm{c}) \leftarrow \langle \tilde{\mathsf{S}}_\mathsf{c} \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle \\ m \leftarrow \mathsf{out}_{\mathsf{R_o}}(\langle \tilde{\mathsf{S}}_\mathsf{o}(\tilde{\mathrm{d}}) \leftrightarrow \mathsf{R_o}(\mathrm{c})\rangle) \\ m' \leftarrow \mathsf{out}_{\mathsf{R_o}}(\langle \tilde{\mathsf{S}}'_\mathsf{o}(\tilde{\mathrm{d}}) \leftrightarrow \mathsf{R_o}(\mathrm{c})\rangle)\end{array}\right] \leq \mathsf{negl}(\lambda) \ .$$

– *t-hiding: A timed commitment is t-hiding with gap* $\epsilon < 1$ *if there exists a polynomial* $\tilde{t}(\cdot)$*, such that for every polynomial* $t(\cdot) \geq \tilde{t}(\cdot)$ *and every polynomial-size distinguisher* $\mathsf{D} := (\mathsf{D}_\lambda)_{\lambda\in\mathbb{N}}$ *of* $\mathsf{depth}(\mathsf{D}_\lambda) \leq t(\lambda)^\epsilon$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for every* $\lambda \in \mathbb{N}$ *and* $m_0, m_1 \in \mathcal{M}_\lambda$*,*

$$\Pr\left[b' = b \,\middle|\, b \leftarrow \{0,1\}; b' \leftarrow \mathsf{out}_{\mathsf{D}_\lambda}(\langle \mathsf{S_c}(1^\lambda, t, m_b) \leftrightarrow \mathsf{D}_\lambda\rangle)\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda) \ .$$

We also define a weaker version of t-hiding called honest-receiver t-hiding, where the receiver is honest but curious, that is, it follows the protocol's specification but tries to distinguish the committed value. We define honest-receiver hiding for public-coin protocols, where the messages sent from the receiver to the sender are sampled uniformly at random.

Definition 2 (Honest-receiver t-hiding). *A timed commitment is* honest-receiver t-hiding *with gap $\epsilon < 1$ if there exists a polynomial $\tilde{t}(\cdot)$, such that for every polynomial $t(\cdot) \geq \tilde{t}(\cdot)$, there exists a polynomial-size simulator* $\mathsf{Sim} := (\mathsf{Sim}_\lambda)_{\lambda \in \mathbb{N}}$ *of* $\mathsf{depth}(\mathsf{Sim}_\lambda) < t(\lambda)^\epsilon$ *such that for every polynomial-size distinguisher* $\mathsf{D} := (\mathsf{D}_\lambda)_{\lambda \in \mathbb{N}}$ *of* $\mathsf{depth}(\mathsf{D}_\lambda) < t(\lambda)^\epsilon$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for every $\lambda \in \mathbb{N}$ and message $m \in \mathcal{M}_\lambda$,*

$$\left|\Pr\left[\mathsf{D}_\lambda(\mathsf{trans}(\langle \mathsf{S_c}(1^\lambda, t, m) \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle)) = 1\right] - \Pr[\mathsf{D}_\lambda(\mathsf{Sim}_\lambda) = 1]\right| \leq \mathsf{negl}(\lambda).$$

Finally, we define *weak public verifiability.* In the *weak public verifiability* game, the commitment is not directly provided by the adversary but it is the result of a commit phase run with an *honest receiver.* Notice that this weak property basically gives the same guarantees of (regular) public verifiability when commitment transcripts come from a trusted source.[6]

Definition 3 (Weak public verifiability). *A timed commitment is* weakly publicly verifiable *if for every $\lambda, t \in \mathbb{N}$ and every* unbounded *malicious (stateful) sender* $\tilde{\mathsf{S}} := (\tilde{\mathsf{S}}_\mathsf{c}, \tilde{\mathsf{S}}_\mathsf{fo})$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that,*

$$\Pr\left[\begin{array}{l|l} \mathsf{FVerify}(1^\lambda, t, \mathrm{c}, m_0, \pi_0) = 1 \wedge & (\tilde{\mathrm{d}}, \mathrm{c}) \leftarrow \langle \tilde{\mathsf{S}}_\mathsf{c} \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle \\ \mathsf{FVerify}(1^\lambda, t, \mathrm{c}, m_1, \pi_1) = 1 \wedge & (\mathrm{c}, m_0, \pi_0, m_1, \pi_1) \leftarrow \tilde{\mathsf{S}}_\mathsf{fo}(\tilde{\mathrm{d}}) \\ \mathrm{c} \neq \perp \wedge m_0 \neq m_1 & \end{array}\right] \leq \mathsf{negl}(\lambda).$$

2.2 Time-lock Puzzles

We recall the definition of time-lock puzzle given in [BGJ+16].

Definition 4 (Time-lock puzzles [BGJ+16]). *A time-lock puzzle* TLP *is a pair of algorithms* $(\mathsf{Gen}, \mathsf{Solve})$ *that works as follows:*

- $z \leftarrow \mathsf{Gen}(1^\lambda, t, s)$ *is a probabilistic algorithm that takes as input a security parameter $\lambda \in \mathbb{N}$, a difficulty parameter $t \in \mathbb{N}$, and a solution $s \in \{0,1\}^\lambda$, and outputs a puzzle z.*
- $s := \mathsf{Solve}(1^\lambda, 1^t, z)$ *is a deterministic algorithm that takes as input the security parameter λ, the difficulty parameter t, and a puzzle z. It outputs a solution s.*

[6] To reduce trust one might require that, for each round, each party signs its outgoing messages, together with all the messages (both incoming and outgoing) exchanged so far. In this setting, the digital signatures act as a certificate to verify that the transcript actually comes from a protocol run by two specific parties. Another technique to reduce trust is to use decentralized TLS oracles [ZMM+20].

We require correctness and hardness.

- *Correctness: For every security parameter λ, difficulty parameter t, and solution $s \in \{0,1\}^\lambda$:*

$$\Pr\left[\mathsf{Solve}(1^\lambda, 1^t, z) = s \;\middle|\; z \leftarrow \mathsf{Gen}(1^\lambda, t, s)\right] = 1$$

- *Efficiency:* Gen *runs in time* $\mathsf{poly}(\log t, \lambda)$ *while* Solve *runs in time* $t \cdot \mathsf{poly}(\lambda)$.
- *Hardness: A time-lock puzzle* $\mathsf{TLP} = (\mathsf{Gen}, \mathsf{Solve})$ *is hard with gap* $\epsilon < 1$ *if there exists a polynomial* $\tilde{t}$*, such that for every polynomial* $t(\cdot) \geq \tilde{t}(\cdot)$*, polynomial-size* $\mathsf{A} := (\mathsf{A})_{\lambda \in \mathbb{N}}$ *of* $\mathsf{depth}(\mathsf{A}) \leq t(\lambda)^\epsilon$*, there exists a negligible function* $\mathsf{negl}(\cdot)$*, such that for every* $\lambda \in \mathbb{N}$*, and every pair of solutions* $s_0, s_1 \in \{0,1\}^\lambda$*,*

$$\Pr\left[b \leftarrow \mathsf{A}(z) \;\middle|\; b \leftarrow \{0,1\}; z \leftarrow \mathsf{Gen}(1^\lambda, t, s_b)\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

2.3 Secure Multi-party Computation

We give a definition of secure multi-party computation protocols based on the ones given by Ishai et al. [IKOS07]. We consider MPC protocols Π_ϕ that realize any deterministic N-party functionality ϕ whose output is a bit (received by all parties), and we assume that every party implicitly takes the functionality ϕ to be computed as input. We define the concept of consistent views as well as the properties that Π_ϕ has to satisfy below.

Definition 5 (Consistent Views). *Let Π_ϕ be an N-party MPC protocol for functionality ϕ. We say* view_i *and* view_j *with $i, j \in [N]$ s.t. $i \neq j$ are* consistent *if the outgoing messages that can be subsumed from* view_i *are identical to the incoming messages contained in* view_j *and vice versa.*

Definition 6 (Perfect Correctness). *Let Π_ϕ be an N-party MPC protocol for functionality ϕ. We say Π_ϕ has* perfect correctness *if for all private inputs to the parties $(x_1, \ldots, x_N)$, the probability that the output of a party in an honest execution of Π_ϕ is different from $\phi(x_1, \ldots, x_N)$ is 0.*

Definition 7 (2-Privacy). *Let Π_ϕ be an N-party MPC protocol for functionality ϕ. We say Π_ϕ has* perfect 2-privacy *if there exists a simulator* $\mathsf{Sim}_{\mathsf{MPC}}$ *such that for any input of the parties $(x_1, \ldots, x_N)$ and corrupted (semi-honest) parties $i, j \in [N]$,* $\mathsf{Sim}_{\mathsf{MPC}}((i,j), (x_i, x_j), \phi(x_1, \ldots, x_n))$ *is identically distributed to the joint view* $(\mathsf{view}_i, \mathsf{view}_j)$.

3 Our Constructions

In Sect. 3.1, we define *quasi publicly verifiable* TLPs (QPV-TLPs) and we give a black-box construction from any standard TLP in Sect. 3.2. In Sect. 3.3, we give a black-box construction of TCs with honest-receiver t-hiding from any QPV-TLPs. In Sect. 3.4, we lift such construction to full t-hiding. This establishes our main result: a black-box timed commitment from any standard TLP. Finally, in Sect. 3.5 we show a version of our TCs that features a more efficient force-open.

3.1 Quasi Publicly Verifiable TLPs

We define the concept of quasi publicly verifiable time-lock puzzle as a time-lock puzzle satisfying some additional properties.

Definition 8. *A quasi publicly verifiable time-lock puzzle is a tuple of algorithms* $(\mathsf{Gen}, \mathsf{Solve}, \mathsf{Verify})$ *that works as follows:*

- $z \leftarrow \mathsf{Gen}(1^\lambda, t, s)$ *is a probabilistic algorithm that on input a security parameter* $\lambda \in \mathbb{N}$, *a difficulty parameter* $t \in \mathbb{N}$ *and a solution* $s \in \{0,1\}^\lambda$, *outputs a puzzle* z.
- $(s, \pi) \leftarrow \mathsf{Solve}(1^\lambda, 1^t, z)$ *is a probabilistic algorithm that takes as input the security parameter* λ, *the difficulty parameter* t, *and a puzzle* z. *It outputs a solution* s *and a proof* π.
- $b := \mathsf{Verify}(1^\lambda, t, z, s, \pi)$ *is a deterministic algorithm that on input the security parameter* λ, *the difficulty parameter* t, *a puzzle* z, *a solution* s, *and a proof* π, *outputs a bit* b *indicating acceptance* $(b = 1)$ *or rejection* $(b = 0)$.

We require the same correctness (considering only the solution s *output by* Solve*), the same hardness, and same efficiency properties of a regular TLP. Additionally, we require efficient verification, completeness, and perfect soundness.*

- *Efficient verification:* Verify *runs in time* $\mathsf{poly}(\log t, \lambda)$.
- *Completeness: For every security parameter* λ, *difficulty parameter* t, *and* $s \in \{0,1\}^\lambda$,

$$\Pr\left[\mathsf{Verify}(1^\lambda, t, z, \mathsf{Solve}(1^\lambda, 1^t, z)) = 1 \;\middle|\; z \leftarrow \mathsf{Gen}(1^\lambda, t, s)\right] = 1.$$

- *Perfect soundness: For all* λ, *difficulty parameter* t, $\nexists\, s_0, s_1 \in \{0,1\}^\lambda$, π_0, π_1, *and* z *such that* $s_0 \neq s_1 \wedge \mathsf{Verify}(1^\lambda, t, z, s_0, \pi_0) = 1 \wedge \mathsf{Verify}(1^\lambda, t, z, s_1, \pi_1) = 1$.

3.2 Quasi Publicly Verifiable TLP from any TLP

In this section, we show how to build a quasi publicly verifiable time-lock puzzle $\mathsf{TLP}^{\mathsf{qpv}} = (\mathsf{Gen}, \mathsf{Solve}, \mathsf{Verify})$ from any time-lock puzzle $\mathsf{TLP} = (\mathsf{TLP.Gen}, \mathsf{TLP.Solve})$. The construction is given in Fig. 1.

Theorem 1. *If* $(\mathsf{TLP.Gen}, \mathsf{TLP.Solve})$ *is a time-lock puzzle, then* $(\mathsf{Gen}, \mathsf{Solve}, \mathsf{Verify})$ *of Fig. 1 is a quasi publicly verifiable time-lock puzzle.*

Remark 1. An alternative construction would be to replace z_0, z_1 in Fig. 1 by $z_0 \leftarrow \mathsf{TLP.Gen}(1^\lambda, t, s||r_1)$ and $z_1 := \mathsf{TLP.Gen}(1^\lambda, t, s; r_1)$. This has the disadvantage that the solution space of z_0 is twice as large as that of z_1. The advantage, however, is that one needs to *only solve one puzzle* (z_0) and recompute z_1.

Proof (of Theorem 1). Due to lack of space, we defer the proof to the full version. □

- $z := (z_0, z_1) \leftarrow \mathsf{Gen}(1^\lambda, t, s)$: On input a security parameter λ, a difficulty parameter t, and a solution $s \in \{0,1\}^\lambda$, do:
  - $r \leftarrow \{0,1\}^\lambda$
  - $z_0 := \mathsf{TLP.Gen}(1^\lambda, t, s; r)$
  - $z_1 \leftarrow \mathsf{TLP.Gen}(1^\lambda, t, r)$.
- $(s, \pi) := \mathsf{Solve}(1^\lambda, 1^t, z)$: On input a puzzle $z = (z_0, z_1)$, do
  - $s := \mathsf{TLP.Solve}(1^\lambda, 1^t, z_0)$
  - $\pi := \mathsf{TLP.Solve}(1^\lambda, 1^t, z_1)$.
- $b := \mathsf{Verify}(1^\lambda, t, z, s, \pi)$: On input a puzzle $z = (z_0, z_1)$, a solution s, and a proof π, set $b = 1$ if and only if $z_0 = \mathsf{TLP.Gen}(1^\lambda, t, s; \pi)$.

Fig. 1: A black-box construction of QPV-TLP from any (standard) TLP.

3.3 Timed Commitment with Honest-Receiver t-hiding

In this section, we give a compiler that starting from a QPV-TLP, a non-interactive statistically binding commitment scheme $\mathsf{SBCom} = (\mathsf{Com}, \mathsf{Dec})$, and a secure MPC protocol Π_ϕ for a functionality ϕ defined below, gives a timed commitment with honest-receiver t-hiding. Π_ϕ computes the following boolean functionality

$$\phi(\alpha, \gamma, x_1, x_2, x_3) := 1 \text{ iff } \gamma = \alpha r + m \text{ where } m||r := x_1 \oplus x_2 \oplus x_3 \wedge |m| = |r| \quad (1)$$

where ϕ, α, γ are known to all parties and x_i is the private input of party i.

All our underlying primitives are used in a black-box way. Furthermore, MPC protocols are information-theoretic constructions, and black-box non-interactive statistically binding commitments can be realized from any one-way permutation [Gol01]. We assume that the messages to be committed to belong to a field $\mathbb{F} \subseteq \{0,1\}^\lambda$. The commit and open phases and the FOpen and $\mathsf{FVerify}$ algorithms are depicted in Fig. 2 and Fig. 3 respectively.

Theorem 2. *Let* $\mathsf{TLP}^{\mathsf{qpv}} = (\mathsf{Gen}, \mathsf{Solve}, \mathsf{Verify})$ *be a quasi publicly verifiable TLP (Def. 8),* $\mathsf{SBCom} = (\mathsf{Com}, \mathsf{Dec})$ *a non-interactive statistically binding and computationally hiding commitment scheme, and* Π_ϕ *a 3-party secure MPC (as defined in Sect. 2.3) for* ϕ *defined in* (1)*, then* $\mathsf{HRTC} = (\mathsf{S} = (\mathsf{S_c}, \mathsf{S_o}), \mathsf{R} = (\mathsf{R_c}, \mathsf{R_o}), \mathsf{FOpen}, \mathsf{FVerify})$ *defined in Fig. 2 and Fig. 3 is an* honest-receiver t-hiding *timed commitment for message space* $\mathcal{M}_\lambda = \mathbb{F} \subseteq \{0,1\}^\lambda$.

We prove Theorem 2 by proving several lemmas, one for each of the properties required in Def. 1. Correctness of honest openings and correctness of force openings are easily verified by inspection.

Lemma 1. HRTC *is efficient if* $\mathsf{TLP}^{\mathsf{qpv}}$ *is a QPV-TLP.*

Proof (of Lemma 1). First, FOpen runs in time $t \cdot \mathsf{poly}(\lambda)$: its running time is dominated by λ parallel invocations of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}$, each of which runs in

Commit Phase: $(\mathsf{dec}, \mathsf{com}) \leftarrow \langle \mathsf{S_c}(1^\lambda, t, m) \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle$

Step 1: $\mathsf{S_c}$ does the following:
1. Sample randomness $r \leftarrow \mathbb{F} \setminus \{0\}$
2. $\forall i \in [\lambda]$, sample $v_{i,1}, v_{i,2} \leftarrow \{0,1\}^{2\lambda}$ and compute $v_{i,3} = v_{i,1} \oplus v_{i,2} \oplus (m||r)$
3. $\forall i \in [\lambda], j \in [3]$, sample randomness $\beta_{i,j}$ and compute $z_{i,j} := \mathsf{TLP^{qpv}.Gen}(1^\lambda, t, v_{i,j}; \beta_{i,j})$
4. **Send** $(z_{i,j})_{i\in[\lambda], j\in[3]}$ to $\mathsf{R_c}$.

Step 2: $\mathsf{R_c}$ samples $\alpha \leftarrow \mathbb{F} \setminus \{0\}$ and **sends** it to $\mathsf{S_c}$.

Step 3: $\mathsf{S_c}$ does the following:
1. Compute $\gamma := r\alpha + m$
2. $\forall i \in [\lambda], j \in [3]$:
   (a) Run Π_ϕ in the head, with public inputs α and γ, and $v_{i,j}$ as the private input to party j in the i-th execution of Π_ϕ. Let $\mathsf{view}_{i,1}, \mathsf{view}_{i,2}, \mathsf{view}_{i,3}$ be the views of the i-th execution
   (b) Compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$
3. **Send** $(c_{i,j})_{i\in[\lambda], j\in[3]}$ and γ to $\mathsf{R_c}$.

Step 4: $\forall i \in [\lambda]$, $\mathsf{R_c}$ samples random $a_i, b_i \in [3]$ with $a_i \neq b_i$ and **sends** $(a_i, b_i)_{i\in[\lambda]}$ to $\mathsf{S_c}$.

Step 5: $\mathsf{S_c}$ **sends** $(v_{i,a_i}, \mathsf{view}_{i,a_i}, \beta_{i,a_i}, d_{i,a_i}, v_{i,b_i}, \mathsf{view}_{i,b_i}, \beta_{i,b_i}, d_{i,b_i})_{i\in[\lambda]}$ to $\mathsf{R_c}$.

Output of the commit phase: $\mathsf{S_c}$ and $\mathsf{R_c}$ output dec and com where
- $\mathsf{S_c}$ sets $\mathsf{dec} = (v_{i,k}, \beta_{i,k})_{i\in[\lambda], k\in[3]\setminus\{a_i,b_i\}}$.
- $\mathsf{R_c}$ performs the following checks $\forall i \in [\lambda], \delta \in \{a_i, b_i\}$:
  1. $\mathsf{Dec}(1^\lambda, c_{i,\delta}, \mathsf{view}_{i,\delta}, d_{i,\delta}) = 1$
  2. $\mathsf{TLP^{qpv}.Gen}(1^\lambda, t, v_{i,\delta}; \beta_{i,\delta}) = z_{i,\delta}$
  3. view_{i,a_i} and view_{i,b_i} are consistent
  4. $(v_{i,\delta}, 1)$ is the input/output pair of P_δ defined by $\mathsf{view}_{i,\delta}$.

  If the checks are successful, $\mathsf{R_c}$ sets

$$\mathsf{com} = (z_{i,1}, z_{i,2}, z_{i,3}, \alpha, \gamma, a_i, b_i, v_{i,a_i}, \beta_{i,a_i}, v_{i,b_i}, \beta_{i,b_i})_{i\in[\lambda]} \tag{2}$$

  otherwise, $\mathsf{com} = \perp$.

Open Phase: $m \leftarrow \mathsf{out}_{\mathsf{R_o}}(\langle \mathsf{S_o}(\mathsf{dec}) \leftrightarrow \mathsf{R_o}(\mathsf{com})\rangle)$

Decommitment: $\mathsf{S_o}$ **sends** dec to $\mathsf{R_o}$.

Output of the open phase: If for more than $\lambda/2$ repetitions[a] $i \in [\lambda]$, there exists a *fixed* $m' \in \mathbb{F}$ s.t. $\mathsf{TLP^{qpv}.Gen}(1^\lambda, t, v_{i,k}; \beta_{i,k}) = z_{i,k}$ where $k \in [3] \setminus \{a_i, b_i\}$ and $v_{i,a_i} \oplus v_{i,b_i} \oplus v_{i,k} = m'||r_i$ for any $r_i \in \{0,1\}^\lambda$, then $\mathsf{R_o}$ outputs $m = m'$. Otherwise, $m = \perp$.

[a] Allowing up to $\lambda/2$ inconsistent positions in the open phase ensures that a (potentially malicious) accepting commit phase corresponds to a valid commitment, i.e., there exists a valid decommitment for it.

Fig. 2: Commit and open phases of HRTC.

```
$(m, \pi) \leftarrow \mathsf{FOpen}(1^\lambda, 1^t, \mathsf{com})$: Perform the following steps:
    1. Initialize the set $I = \emptyset$ and parse $\mathsf{com}$ as in (2)
    2. In parallel, $\forall i \in [\lambda]$ and $k \in [3] \setminus \{a_i, b_i\}$, run $(v_{i,k}, \pi_{i,k}) \leftarrow \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}(1^\lambda, 1^t, z_{i,k})$. If $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Verify}(1^\lambda, t, z_{i,k}, v_{i,k}, \pi_{i,k}) = 1$, then $I = I \cup \{i\}$.
    3. If there exists $J \subseteq I$ s.t. $|J| > \lambda/2$ and $\forall j \in J$ it holds that $v_{j,a_j} \oplus v_{j,b_j} \oplus v_{j,k} = m' || r_j$ where $k \in [3] \setminus \{a_j, b_j\}$, $r_j \in \{0,1\}^\lambda$ arbitrary and $m' \in \mathbb{F}$ fixed, then set $m = m'$ and $\pi := (J, (v_{j,k}, \pi_{j,k}))_{j \in J, k \in [3] \setminus \{a_j, b_j\}}$; otherwise return $m = \pi = \perp$.

$b := \mathsf{FVerify}(1^\lambda, t, \mathsf{com}, m, \pi)$: If either $m = \perp$ or $\pi = \perp$ return 0. Otherwise, parse
  $\mathsf{com}$ as in (2) and $\pi := (J, (v_j^*, \pi_j^*)_{j \in J})$ and set $b = 1$ iff $|J| > \lambda/2$ and $\forall j \in J$:
    1. $z_{j,a_j} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{j,a_j}; \beta_{j,a_j})$
    2. $z_{j,b_j} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{j,b_j}; \beta_{j,b_j})$
    3. $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Verify}(1^\lambda, t, z_{j,k}, v_j^*, \pi_j^*) = 1$ for $k \in [3] \setminus \{a_j, b_j\}$
    4. $v_{j,a_j} \oplus v_{j,b_j} \oplus v_j^* = m || r_j$ for any $r_j \in \{0,1\}^\lambda$.
```

Fig. 3: FOpen and $\mathsf{FVerify}$ algorithms of HRTC.

time $t \cdot \mathsf{poly}(\lambda)$. Second, $\mathsf{FVerify}$ runs in time $\mathsf{poly}(\log t, \lambda)$: its running time is dominated by $O(\lambda)$ invocations of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}$ and $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Verify}$, each of which runs in time $\mathsf{poly}(\log t, \lambda)$. Finally, $\mathsf{S_c}, \mathsf{S_o}, \mathsf{R_c}, \mathsf{R_o}$ all run in time $\mathsf{poly}(\log t, \lambda)$ as all the primitives they invoke either take time $\mathsf{poly}(\lambda)$ (e.g. SBCom and MPC in the head) or $\mathsf{poly}(\log t, \lambda)$ (e.g. $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}$ and $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Verify}$). □

Lemma 2. HRTC *is sound if* $\mathsf{TLP}^{\mathsf{qpv}}$ *is a QPV-TLP,* SBCom *is a non-interactive statistically biding commitment, and* Π_ϕ *is a secure* 3*-party MPC.*

Proof (of Lemma 2). For simplicity, in the following proof, we treat both SBCom and $\mathsf{TLP}^{\mathsf{qpv}}$ as perfectly binding. This means that once the sender commits to a view (with SBCom) or a share (with $\mathsf{TLP}^{\mathsf{qpv}}$), it is impossible for the sender to later decommit to a different value than the one initially committed to. While $\mathsf{TLP}^{\mathsf{qpv}}$ is indeed perfectly binding (a proof of this is given in Lemma 4), SBCom is only statistically binding. However, the statistical binding property of SBCom ensures that each commitment defines a string such that, except with negligible probability, only that specific string can be decommitted to later. Therefore, the soundness error of our construction when using statistically binding commitments is at most negligibly larger than that of using perfectly binding commitments (i.e., this difference accounts for the negligible probability that the sender breaks the binding of one of the commitments).

Let us call a repetition $i \in [\lambda]$ bad if $\phi(\alpha, \gamma, v_{i,1}, v_{i,2}, v_{i,3}) = 0$ or $\exists u \in [3]$ s.t. the time-lock puzzle $z_{i,u}$ is malformed. We argue that conditioned on $\mathsf{com} \neq \perp$, the probability that a repetition is bad at most $\frac{1}{3}$. First, observe that the receiver detects a malformed puzzle with probability at least $\frac{1}{3}$. Indeed, the receiver would ask the sender to provide $v_{i,u}, \beta_{i,u}$ s.t. $z_{i,u} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{i,u}; \beta_{i,u})$

with probability $\frac{1}{3}$ (i.e., see the second verification check performed by $\mathsf{R_c}$ when deciding the output of the commit phase in Fig. 2).

Second, if $\phi(\alpha, \gamma, v_{i,1}, v_{i,2}, v_{i,3}) = 0$, then the receiver will also detect it with probability at least $\frac{1}{3}$. Indeed, if within a repetition $\phi(\alpha, \gamma, v_{i,1}, v_{i,2}, v_{i,3}) = 0$, by the perfect correctness of Π_ϕ, for all choices of $v_{i,1}, v_{i,2}, v_{i,3}$, the outputs of all 3 parties in any honest execution of Π_ϕ must be 0. Therefore, considering the inputs $(v_{i,j})_{j\in[3]}$ and views $(\mathsf{view}_{i,j})_{j\in[3]}$ committed by the sender in Step 1 and Step 3 of the commit phase respectively, either in all the views the output is 0, or there exists two views which are malformed (i.e, they are either inconsistent with each other or have different inputs w.r.t. the ones committed in Step 1). In the former case, the receiver would always reject, while in the latter it would reject with probability at least $\frac{1}{3}$ (i.e., see the third and fourth verification checks performed by $\mathsf{R_c}$ when deciding the output of the commit phase in Fig. 2). Therefore, the probability that, given that $\mathsf{com} \neq \bot$, a repetition $i \in [\lambda]$ is bad is at most $\frac{1}{3}$.

Let us now consider the probability that, conditioned on $\mathsf{com} \neq \bot$, more than $\log^2(\lambda)$ repetitions are bad. Such probability is at most $(\frac{1}{3})^{\log^2(\lambda)} < (\frac{1}{2})^{\log^2(\lambda)} = (\frac{1}{\lambda})^{\log(\lambda)}$, which is negligible.

Let $S \subseteq [\lambda]$ be the set containing the indices of the repetitions that are not bad. For every $i \in S$ where $v_{i,1} \oplus v_{i,2} \oplus v_{i,3} = m_i || r_i$ with $|m_i| = |r_i|$, it holds that $\gamma = \alpha r_i + m_i$. By the Schwartz-Zippel lemma, with overwhelming probability over the choice of α, there must exist (m, r) such that $m_i = m$ and $r_i = r$ for all $i \in S$. Recall that $\forall i \in S$ and $k \in [3]$ s.t. $a_i, b_i \neq k$ we have that $z_{i,k}$ is well-formed. The correctness and the completeness of $\mathsf{TLP^{qpv}}$ guarantee that whenever a puzzle is correctly generated, the solution/proof pair output by $\mathsf{TLP^{qpv}.Solve}$ will be accepting and will contain the committed value. Thus, since $|S| \geq \lambda - \log^2(\lambda)$ except with negligible probability, the force open phase and the open phase will agree on the same message m on more than $\frac{\lambda}{2}$ positions (i.e., $m' = m$) and $\mathsf{FVerify}$ will output 1 on input the message/proof pair returned by FOpen. □

Lemma 3. HRTC *is publicly verifiable if* $\mathsf{TLP^{qpv}}$ *is a QPV-TLP.*

Proof (of Lemma 3). Assume A outputs $\mathsf{com} = (z_{i,1}, z_{i,2}, z_{i,3}, \alpha, \gamma, a_i, b_i, v_{i,a_i}, \beta_{i,a_i}, v_{i,b_i}, \beta_{i,b_i})_{i\in[\lambda]}$ and two proof/message pairs $(m_0, \pi_0 = (v^0_{i,k}, \pi^0_{i,k})_{i\in[\lambda]})$ and $(m_1, \pi_1 = (v^1_{i,k}, \pi^1_{i,k})_{i\in[\lambda]})$ with $m_0 \neq m_1$ and that are accepting w.r.t. com. This means that for more than $\lambda/2$ repetitions $v_{i,a_i} \oplus v_{i,b_i} \oplus v^0_{i,k} = m_0 || r^0_i$ and $v_{i,a_i} \oplus v_{i,b_i} \oplus v^1_{i,k} = m_1 || r^1_i$. Thus, there must be at least one repetition $i \in [\lambda]$ s.t. $v^0_{i,k} \neq v^1_{i,k}$ and $\mathsf{TLP^{qpv}.Verify}(1^\lambda, t, z_{i,k}, v^0_{i,k}, \pi^0_{i,k}) = 1$ and $\mathsf{TLP^{qpv}.Verify}(1^\lambda, t, z_{i,k}, v^1_{i,k}, \pi^1_{i,k}) = 1$, which contradicts the perfect soundness of $\mathsf{TLP^{qpv}}$. □

Lemma 4. HRTC *is statistically binding if* $\mathsf{TLP^{qpv}}$ *is a QPV-TLP.*

Proof (of Lemma 4). Assume that the malicious sender provides two valid decommitments $\mathsf{dec}_0 = (v^0_{i,k}, \beta^0_{i,k})_{i\in[\lambda]}$ and $\mathsf{dec}_1 = (v^1_{i,k}, \beta^1_{i,k})_{i\in[\lambda]}$ to m_0 and m_1, respectively, such that $m_0 \neq m_1$, for the same $\mathsf{com} = (z_{i,1}, z_{i,2}, z_{i,3}, \alpha, \gamma, a_i, b_i, v_{i,a_i},$

$\beta_{i,a_i}, v_{i,b_i}, \beta_{i,b_i})_{i\in[\lambda]}$. This means that for more than $\lambda/2$ repetitions $v_{i,a_i} \oplus v_{i,b_i} \oplus v^0_{i,k} = m_0||r^0_i$ and $v_{i,a_i} \oplus v_{i,b_i} \oplus v^1_{i,k} = m_1||r^1_i$. Thus, there exists at least one repetition $i \in [\lambda]$ such that $v^0_{i,k} \neq v^1_{i,k}$ and $z_{i,k} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v^0_{i,k}; \beta^0_{i,k})$ and $z_{i,k} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v^1_{i,k}; \beta^1_{i,k})$, which contradicts the correctness of $\mathsf{TLP}^{\mathsf{qpv}}$. Indeed, the correctness property of TLPs guarantees that they are injective. For the reader's convenience we report again here the proof of this fact, as we have already shown in the proof of perfect soundness of $\mathsf{TLP}^{\mathsf{qpv}}$ (see Sect. 3.2). Any puzzle z, even if maliciously generated, belongs to the support of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, s)$ for at most one solution s. Assume, for the sake of contradiction, that there exists a z belonging to the support of both $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, s_0)$ and $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, s_1)$ with $s_0 \neq s_1$. Let $s = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}(1^\lambda, 1^t, z)$. If $s \neq s_0$, this contradicts the correctness of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}$ regarding puzzles in the support of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, s_0)$. If $s = s_0$, it contradicts the correctness of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}$ regarding puzzles in the support of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, s_1)$. Hence, for any puzzle z, there exists at most one solution s, and if a solution s exists, then $s = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}(1^\lambda, 1^t, z)$. □

Lemma 5. HRTC *is honest-receiver t-hiding if* SBCom *is a non-interactive computationally hiding commitment,* $\mathsf{TLP}^{\mathsf{qpv}}$ *is a QPV-TLP, and* Π_ϕ *is a secure 3-party MPC.*

Proof (of Lemma 5). The simulator Sim proceeds as follows:

1. Sample random $\alpha, \gamma \leftarrow \mathbb{F} \setminus \{0\}$
2. $\forall i \in [\lambda]$ sample random $a_i, b_i \leftarrow [3]$ with $a_i \neq b_i$
3. $\forall i \in [\lambda], j \in \{a_i, b_i\}, k \in [3] \setminus \{a_i, b_i\}$ sample $v_{i,j} \leftarrow \{0,1\}^{2\lambda}$ and set $v_{i,k}$ to garbage, say $v_{i,k} = 0^{2\lambda}$
4. $\forall (i,j) \in [\lambda] \times [3]$ sample randomness $\beta_{i,j}$ and compute $z_{i,j} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{i,j}; \beta_{i,j})$
5. $\forall i \in [\lambda], k \in [3] \setminus \{a_i, b_i\}$, compute $(\mathsf{view}_{i,a_i}, \mathsf{view}_{i,b_i}) \leftarrow \mathsf{Sim}_{\mathsf{MPC}}((a_i, b_i), (v_{i,a_i}, v_{i,b_i}), 1)$ and set $\mathsf{view}_{i,k} := 0^\ell$ where ℓ is the size of a view
6. $\forall (i,j) \in [\lambda] \times [3]$ compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$
7. Output the protocol's transcript $(z_{i,1}, z_{i,2}, z_{i,3}, \alpha, c_{i,1}, c_{i,2}, c_{i,3}, \gamma, a_i, b_i, v_{i,a_i}, \mathsf{view}_{i,a_i}, \beta_{i,a_i}, d_{i,a_i}, v_{i,b_i}, \mathsf{view}_{i,b_i}, \beta_{i,b_i}, d_{i,b_i})_{i\in[\lambda]}$.

Notice the running time of Sim is essentially independent of t. We prove indistinguishability from a regular transcript via a series of indistinguishable hybrids.

$\mathcal{H}_0$: This is identical to the sender in the commit phase of Fig. 2 except that, instead of receiving $\alpha, (a_i, b_i)_{i\in[\lambda]}$ from the receiver, the sender *itself* samples samples random $\alpha \leftarrow \mathbb{F} \setminus \{0\}$ and for all $i \in [\lambda]$, $a_i, b_i \leftarrow [3]$ with $a_i \neq b_i$. Formally, $\mathcal{H}_0$ is defined as follows:

```
1. Sample random $r, \alpha \leftarrow \mathbb{F} \setminus \{0\}$ and set $\gamma := r\alpha + m$
2. $\forall i \in [\lambda]$ sample random $a_i, b_i \in [3]$ with $a_i \neq b_i$
3. $\forall i \in [\lambda]$ sample $v_{i,1}, v_{i,2} \leftarrow \{0,1\}^{2\lambda}$ and compute
   $v_{i,3} = v_{i,1} \oplus v_{i,2} \oplus (m||r)$
4. $\forall i \in [\lambda], j \in [3]$, sample randomness $\beta_{i,j}$ and compute
   $z_{i,j} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{i,j}; \beta_{i,j})$
```

5. $\forall i \in [\lambda], j \in [3]$, run Π_ϕ in the head, with public inputs α and γ, and $v_{i,j}$ as the private input to party j in the i-th execution of Π_ϕ. Let $\mathsf{view}_{i,1}, \mathsf{view}_{i,2}, \mathsf{view}_{i,3}$ be the views of the i-th execution
6. $\forall i \in [\lambda], j \in [3]$, compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$
7. Output the protocol's transcript: $(z_{i,j}, \alpha, c_{i,j}, \gamma, a_i, b_i, v_{i,a_i}, \mathsf{view}_{i,a_i}, \beta_{i,a_i}, d_{i,a_i}, v_{i,b_i}, \mathsf{view}_{i,b_i}, \beta_{i,b_i}, d_{i,b_i})_{i\in[\lambda], j\in[3]}$

$\mathcal{H}_1$: This is identical to $\mathcal{H}_0$ except that for all $i \in [\lambda]$ and $j \in \{a_i, b_i\}$, the shares $v_{i,j}$ are randomly sampled before drawing the remaining share depending on $m||r$. In more details, Step 3 of $\mathcal{H}_0$ is modified as follows. The remaining steps remain unchanged. Formally, $\mathcal{H}_1$ is defined as follows:

$\vdots$

3. $\forall i \in [\lambda], j \in \{a_i, b_i\}, k \in [3] \setminus \{a_i, b_i\}$, sample $v_{i,j} \leftarrow \{0,1\}^{2\lambda}$ and set $v_{i,k} = v_{i,a_i} \oplus v_{i,b_i} \oplus (m||r)$

$\vdots$

$\mathcal{H}_2^i$: This hybrid is parameterized by $i \in \{0, \ldots, \lambda\}$. We define $\mathcal{H}_2^0 := \mathcal{H}_1$. For $i \geq 1$ $\mathcal{H}_2^i$ be identical to $\mathcal{H}_2^{i-1}$ except that in the i-th repetition of the unopened party in Step 6 of $\mathcal{H}_1$, instead of committing to $\mathsf{view}_{i,k}$ where $k \in [3] \setminus \{a_i, b_i\}$, commit to a garbage value, say 0. Formally, $\forall i \in [\lambda]$, $\mathcal{H}_2^i$ is defined as follows:

$\vdots$

6. $\forall j \in [\lambda]$ and $k \in [3] \setminus \{a_j, b_j\}$:
   - $(c_{j,a_j}, d_{j,a_j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{j,a_j})$
   - $(c_{j,b_j}, d_{j,b_j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{j,b_j})$
   - If $j \leq i$:$(c_{j,k}, d_{j,k}) \leftarrow \mathsf{Com}(1^\lambda, 0^\ell)$ where $\ell := |\mathsf{view}_{j,a_j}| = |\mathsf{view}_{j,b_j}|$, i.e., views are padded and are of size ℓ.
   - If $j > i$: $(c_{j,k}, d_{j,k}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{j,k})$.

$\vdots$

$\mathcal{H}_3^i$: This hybrid is parameterized by $i \in \{0, \ldots, \lambda\}$ and we define $\mathcal{H}_3^0 := \mathcal{H}_2$. For $i \geq 1$, $\mathcal{H}_3^i$ is identical to $\mathcal{H}_3^{i-1}$ except that in i-th repetition where $k \in [3] \setminus \{a_i, b_i\}$, instead of committing with $\mathsf{TLP}^{\mathsf{qpv}}$ to the input $v_{i,k}$ of the unopened party in Step 4 of $\mathcal{H}_2^\lambda$, we commit to a garbage value, say 0. Formally, $\forall i \in [\lambda]$, $\mathcal{H}_3^i$ is defined as follows:

$\vdots$

4. $\forall j \in [\lambda]$ and $k \in [3] \setminus \{a_j, b_j\}$:
   - $z_{j,a_j} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{j,a_j}; \beta_{j,a_j})$
   - $z_{j,b_j} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{j,b_j}; \beta_{j,b_j})$
   - If $j \leq i$: $z_{j,k} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, 0^{2\lambda}; \beta_{j,k})$
   - If $j > i$: $z_{j,k} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{j,k}; \beta_{j,k})$.

$\vdots$

$\mathcal{H}_4$: $\mathcal{H}_4$ is identical to $\mathcal{H}_3^\lambda$ except that in Step 5 of $\mathcal{H}_3^\lambda$ for all $i \in [\lambda]$ we compute the opened views as $(\mathsf{view}_{i,a_i}, \mathsf{view}_{i,b_i}) \leftarrow \mathsf{Sim}_{\mathsf{MPC}}((a_i, b_i), (v_{i,a_i}, v_{i,b_i}), 1)$. Formally, $\mathcal{H}_4$ is defined as follows:

$\vdots$

5. $\forall i \in [\lambda] : (\mathsf{view}_{i,a_i}, \mathsf{view}_{i,b_i}) \leftarrow \mathsf{Sim}_{\mathsf{MPC}}((a_i, b_i), (v_{i,a_i}, v_{i,b_i}), 1)$

$\vdots$

$\mathcal{H}_5$: $\mathcal{H}_5$ is identical to $\mathcal{H}_4$ except that γ in Step 1 is sampled uniformly at random. Formally, $\mathcal{H}_5$ is defined as:

1. Sample random $\alpha, \gamma \leftarrow \mathbb{F} \setminus \{0\}$

$\vdots$

Notice that $\mathcal{H}_0$ coincides with the distribution of the transcript of an honest commit phase and $\mathcal{H}_5$ is identical to Sim. We now introduce the following claims.

Claim 1 *$\mathcal{H}_1$ is perfectly indistinguishable from $\mathcal{H}_0$.*

Claim 2 *If SBCom is computationally hiding, then for every $i \in [\lambda]$, $\mathcal{H}_2^i$ is computationally indistinguishable from $\mathcal{H}_2^{i-1}$.*

Claim 3 *If $\mathsf{TLP}^{\mathsf{qpv}}$ is hard, then for every $i \in [\lambda]$, $\mathcal{H}_3^i$ and $\mathcal{H}_3^{i-1}$ are indistinguishable for every poly-size distinguisher with depth upper-bounded by t^ϵ.*

Claim 4 *If Π_ϕ is perfectly 2-private, then $\mathcal{H}_4$ is perfectly indistinguishable from $\mathcal{H}_3^\lambda$.*

Claim 5 *$\mathcal{H}_5$ is perfectly indistinguishable from $\mathcal{H}_4$.*

Proof (of Claim 1). It follows from the fact that for any $x \in \{0,1\}^{2\lambda}$ and all $i, j, k \in [3]$ s.t. $i \neq j \neq k$ where $v_i, v_j \leftarrow \{0,1\}^{2\lambda}$ and $v_k = v_i \oplus v_j \oplus x$, any pair (v_a, v_b) with $a, b \in [3]$ s.t. $a \neq b$ is uniformly distributed in $\{0,1\}^{2\lambda} \times \{0,1\}^{2\lambda}$. □

Proof (of Claim 2). Assume there exists a poly-sized distinguisher D which is able to distinguish between $\mathcal{H}_2^i$ and $\mathcal{H}_2^{i-1}$ with non-negligible probability, we can use D to build an adversary A that wins the hiding game of SBCom with the same probability. A plays in the hiding game with $m_0 = \mathsf{view}_{i,k}$ and $m_1 = 0^\ell$ and gets back from the challenger a commitment c. A constructs the transcript regularly except that for the i-th repetition it sets $c_{i,k} = c$. It then forwards the transcript to D and outputs whatever D outputs. Notice that if $b = 0$ in the hiding game, then A perfectly simulates $\mathcal{H}_2^{i-1}$, and if $b = 1$ then A perfectly simulates $\mathcal{H}_2^i$. Therefore, A wins the hiding game with the same probability with which D distinguishes between $\mathcal{H}_2^i$ and $\mathcal{H}_2^{i-1}$. □

Proof (of Claim 3). Assume there exists a poly-sized distinguisher D whose depth is bounded by t^ϵ which is able to distinguish between $\mathcal{H}_3^i$ and $\mathcal{H}_3^{i-1}$ with non-negligible probability, we can use D to build an adversary A that breaks the hardness of $\mathsf{TLP}^{\mathsf{qpv}}$ with the same probability. A plays in the hardness game with $s_0 = v_{i,k}$ and $s_1 = 0^{2\lambda}$ and gets back from the challenger a puzzle z. A constructs the transcript regularly except that for the i-th repetition it sets $z_{i,k} = z$. It then forwards the transcript to D and outputs whatever D outputs. Notice that if $b = 0$ in the hardness game, then A perfectly simulates $\mathcal{H}_3^{i-1}$, and if $b = 1$ then A perfectly simulates $\mathcal{H}_3^i$. Notice that constructing the transcript keeps the depth of A bounded since the MPC protocol, the commitment algorithm, and the puzzle generation algorithm all run in polynomial time essentially independent of t. Therefore, A wins the hardness game with the same probability with which D distinguishes between $\mathcal{H}_3^i$ and $\mathcal{H}_3^{i-1}$. □

Proof (of Claim 4). It directly follows from the perfect indistinguishability of a pair of real views and a pair of simulated views of Π_ϕ. □

Proof (of Claim 5). It follows from the fact that for any $m \in \mathbb{F}$ and random $\alpha, r \in \mathbb{F} \setminus \{0\}$ the value $\gamma = r\alpha + m$ is uniformly distributed in $\mathbb{F} \setminus \{0\}$. □

The proof is concluded by observing that it simply follows from the proofs of Claims 1 -5, since all of the hybrids are indistinguishable by a poly-size distinguisher whose depth is bounded by t^ϵ. □

3.4 *t*-Hiding Timed Commitment

To lift our *honest-receiver* t-hiding timed commitment to a t-hiding timed commitment we adopt the approach used by Goldreich and Kahan [GK96] to get a constant-round ZK proof from a constant-round honest-verifier ZK proof. Basically, we have the receiver commit, with a two-round statistically-hiding commitment (SHCom), to all of its random challenges at the beginning of the protocol, and then open these commitments in the subsequent rounds. Two-round statistically-hiding commitments can be constructed from collision-resistant hashing [HM96]. We describe the commit phase of our t-hiding TC in Fig. 4.

The verification of the commit phase and its output, of the open phase, and the $\mathsf{FOpen}, \mathsf{FVerify}$ algorithms are identical to the ones described in Sect. 3.3.

Theorem 3. *Let* $\mathsf{TLP}^{\mathsf{qpv}} = (\mathsf{Gen}, \mathsf{Solve}, \mathsf{Verify})$ *be a QPV-TLP (Def. 8),* $\mathsf{SBCom} = (\mathsf{Com}, \mathsf{Dec})$ *a non-interactive statistically binding and computationally hiding commitment scheme,* SHCom *a two-round statistically hiding and computationally binding commitment scheme, and* Π_ϕ *a* 3*-party secure MPC for functionality* ϕ *defined in* (1)*, then* $\mathsf{TC} = (\mathsf{S} = (\mathsf{S_c}, \mathsf{S_o}), \mathsf{R} = (\mathsf{R_c}, \mathsf{R_o}), \mathsf{FOpen}, \mathsf{FVerify})$*, where* S *and* R *are defined in Fig. 4 and* $\mathsf{FOpen}, \mathsf{FVerify}$ *are defined Fig. 3, is a (fully fledged)* t-hiding *timed commitment scheme (Def. 1).*

We only prove t-hiding and soundness, as the proofs of public verifiability and binding are identical to the construction in Sect. 3.3 – see Lemmas 3 and 4.

Commit Phase: $(\mathsf{dec}, \mathsf{com}) \leftarrow \langle \mathsf{S_c}(1^\lambda, t, m) \leftrightarrow \mathsf{R_c}(1^\lambda, t) \rangle$

Step 1: $\mathsf{S_c}$ does the following:

1. Sample randomness $r \leftarrow \mathbb{F} \setminus \{0\}$
2. $\forall i \in [\lambda]$:
   - (a) Sample $v_{i,1}, v_{i,2} \leftarrow \{0,1\}^{2\lambda}$ and compute $v_{i,3} = v_{i,1} \oplus v_{i,2} \oplus (m||r)$
   - (b) $\forall j \in [3]$, sample randomness $\beta_{i,j}$ and compute $z_{i,j} := \mathsf{TLP^{qpv}.Gen}(1^\lambda, t, v_{i,j}; \beta_{i,j})$
   - (c) Compute the first message $\sigma_i[1]$ of the two-round SHCom
3. **Send** $(z_{i,j})_{i\in[\lambda], j\in[3]}, (\sigma_i[1])_{i\in[\lambda]}$ to $\mathsf{R_c}$.

Step 2: $\mathsf{R_c}$ does the following:

1. $\forall i \in [\lambda]$,
   - (a) Sample random $a_i, b_i \in [3]$ with $a_i \neq b_i$
   - (b) Commit using SHCom to (a_i, b_i) by computing the second message $\sigma_i[2]$ of the two-round SHCom. ($\sigma_i = (\sigma_i[1], \sigma_i[2])$ commits to (a_i, b_i).)
2. Sample random $\alpha \leftarrow \mathbb{F} \setminus \{0\}$
3. **Send** $\alpha, (\sigma_i[2])_{i\in[\lambda]}$ to $\mathsf{S_c}$

Step 3: $\mathsf{S_c}$ does the following:

1. Compute $\gamma = r\alpha + m$.
2. $\forall i \in [\lambda], j \in [3]$:
   - (a) Run Π_ϕ in the head, with public input α and γ, and $v_{i,j}$ as the private input to party j in the i-th execution of Π_ϕ. Let $\mathsf{view}_{i,1}, \mathsf{view}_{i,2}, \mathsf{view}_{i,3}$ be the views of the i-th execution
   - (b) Compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$
3. **Send** $(c_{i,j})_{i\in[\lambda], j\in[3]}$ and γ to $\mathsf{R_c}$.

Step 4: $\forall i \in [\lambda]$, $\mathsf{R_c}$ computes decommitment y_i to (a_i, b_i) w.r.t. σ_i and **sends** $(y_i)_{i\in[\lambda]}$ to $\mathsf{S_c}$.

Step 5: If $\forall i \in [\lambda]$, y_i is a valid decommitment to (a_i, b_i) w.r.t. σ_i, $\mathsf{S_c}$ **sends** $(v_{i,a_i}, \mathsf{view}_{i,a_i}, \beta_{i,a_i}, d_{i,a_i}, v_{i,b_i}, \mathsf{view}_{i,b_i}, \beta_{i,b_i}, d_{i,b_i})_{i\in[\lambda]}$ to $\mathsf{R_c}$. Otherwise, $\mathsf{S_c}$ **aborts**.

The output of the commit phase, i.e., $(\mathsf{dec}, \mathsf{com})$, as well as the open phase are identical to those in Fig. 2.

Fig. 4: Commit phase of TC.

Lemma 6. TC *is t-hiding if* SBCom *is a non-interactive computationally hiding commitment scheme,* SHCom *is a two-round computationally binding commitment scheme,* $\mathsf{TLP^{qpv}}$ *is a quasi publicly verifiable time-lock puzzle, and Π_ϕ is a 3-party secure MPC for functionality ϕ defined in* (1).

Proof (of Lemma 6). Our proof strategy involves constructing a bounded-depth simulator Sim that, with black-box access to a malicious (bounded-depth) receiver $\mathsf{R_c^*}$, is able to simulate its view. Such simulated view is indistinguishable from the one coming from a commit phase with an honest sender for every possible message $m \in \mathcal{M}_\lambda$. Let $\approx$ denote that two distributions are indistinguishable by a polynomial-size distinguisher whose depth is bounded by t^ϵ. We then observe that for all $m_0, m_1 \in \mathcal{M}_\lambda$: $\mathsf{view}_{\mathsf{R}_\mathsf{c}^*}(\langle \mathsf{S}_\mathsf{c}(1^\lambda, t, m_0) \leftrightarrow \mathsf{R}_\mathsf{c}^*(1^\lambda, t)\rangle) \approx \mathsf{Sim}^{\mathsf{R}_\mathsf{c}^*} \approx \mathsf{view}_{\mathsf{R}_\mathsf{c}^*}(\langle \mathsf{S}_\mathsf{c}(1^\lambda, t, m_1) \leftrightarrow \mathsf{R}_\mathsf{c}^*(1^\lambda, t)\rangle)$. Therefore, $\mathsf{view}_{\mathsf{R}_\mathsf{c}^*}(\langle \mathsf{S}_\mathsf{c}(1^\lambda, t, m_0) \leftrightarrow \mathsf{R}_\mathsf{c}^*(1^\lambda)\rangle) \approx \mathsf{view}_{\mathsf{R}_\mathsf{c}^*}(\langle \mathsf{S}_\mathsf{c}(1^\lambda, t, m_1) \leftrightarrow \mathsf{R}_\mathsf{c}^*(1^\lambda)\rangle)$, i.e., the existence of this simulator implies t-hiding. We first describe a *simplified* simulator Sim:

1. $\forall i \in [\lambda]$, compute the first message $\sigma_i[1]$ of SHCom; $\forall i \in [\lambda], j \in [3]$, compute $z_{i,j} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, 0^{2\lambda}; \beta_{i,j})$ for random $\beta_{i,j}$; **send** $(\sigma_i[1])_{i\in[\lambda]}$ and $(z_{i,j})_{i\in[\lambda],j\in[3]}$ to R_c^*.
2. **receive** $\forall i \in [\lambda]$ from R_c^* its statistically hiding commitment $\sigma_i[2]$ to (a_i, b_i), and the value α.
3. $\forall i \in [\lambda], j \in [3]$, compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, 0^\ell)$; sample a random $\gamma \leftarrow \mathbb{F} \setminus \{0\}$; **send** $(c_{i,j})_{i\in[\lambda],j\in[3]}$ and γ to R_c^*.
4. **receive** $\forall i \in [\lambda]$ from R_c^* the decommitments y_i w.r.t. σ_i to the challenge indices (a_i, b_i); **abort** if any of the decommitments is not valid.
5. **Rewind Phase:** repeatedly rewind R_c^* back to Step 3, until R_c^* decommits to $(a'_i, b'_i)_{i\in[\lambda]} = (a_i, b_i)_{i\in[\lambda]}$. In particular, instead of Steps 3 and 4, do the following:
   (a) $\forall i \in [\lambda]$, compute $(\mathsf{view}_{i,a_i}, \mathsf{view}_{i,b_i}) \leftarrow \mathsf{Sim}_{\mathsf{MPC}}((a_i, b_i), (v_{i,a_i}, v_{i,b_i}), 1)$ and $\mathsf{view}_{i,k} := 0^\ell$ for $k \in [3] \setminus \{a_i, b_i\}$; $\forall i \in [\lambda], j \in [3]$, compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$; sample a random $\gamma \leftarrow \mathbb{F} \setminus \{0\}$; **send** γ and $(c_{i,j})_{i\in[\lambda],j\in[3]}$ to R_c.
   (b) **receive** $\forall i \in [\lambda]$ from R_c^* the decommitments y'_i w.r.t. σ_i to the challenge indices (a'_i, b'_i); if any of the decommitments y'_i is not valid, execute again Step 5a (with fresh randomness).
   (c) if $\exists i \in [\lambda]$ s.t. $(a_i, b_i) \neq (a'_i, b'_i)$, output $\mathsf{ambiguous}$. Otherwise, exit rewind phase and proceed to Step 6.
6. **send** $(v_{i,a_i}, \mathsf{view}_{i,a_i}, \beta_{i,a_i}, d_{i,a_i}, v_{i,b_i}, \mathsf{view}_{i,b_i}, \beta_{i,b_i}, d_{i,b_i})_{i\in[\lambda]}$ to R_c^*.

Akin to what happens in the the protocol by Goldreich and Kahan [GK96], the simple simulation strategy of Sim suffers from the problem that Sim may not terminate in expected polynomial time. The issue stems from the fact that SBCom is only computationally hiding. Let p_0 be the probability with which R_c^* correctly decommits $(\sigma_i)_{i\in\lambda}$ when it receives commitments $(c_{i,j})_{i\in\lambda,j\in[3]}$ to $0^{2\lambda}$ (i.e., Step 3). Similarly, let p_1 be probability with which R_c^* correctly decommits $(\sigma_i)_{i\in\lambda}$ when (some of the) $(c_{i,j})_{i\in\lambda,j\in[3]}$ are commitments to the output of $\mathsf{Sim}_{\mathsf{MPC}}$ as in Step 5 of Sim. Although $|p_0 - p_1|$ is negligible due to the computational hiding of SBCom, this (negligible) difference in the behaviour of R_c^* may cause Sim to run in exponential time.

To solve this issue, we can use the same technique of [GK96] to modify the simple simulator above to ensure that it does not run for too long. This technique involves first estimating, via a polynomial number of rewinds, the value of p_0 and using such value to limit the total the number of rewinds. Since this technique identically applies to our setting, we omit its description and refer the reader to [GK96, Lin16] for more details. For the rest of the proof we can ignore such subtlety and refer to the simplified simulator above. Indeed, the modifications introduced by the technique of [GK96] do not involve the simulation strategy itself, but they only take care of the running time of the simulator.

It remains to show that the output of Sim and the view of R_c^* in an interaction with honest S_c are indistinguishable by a polynomial-size distinguisher whose depth is bounded by t^ϵ. We show this via a sequence of indistinguishable hybrids. Let $\mathcal{H}_0$ be the real world interaction between S_c and R_c^* (i.e., Fig. 4).

$\mathcal{H}_1$: $\mathcal{H}_1$ is identical to $\mathcal{H}_0$ except that $\mathcal{H}_1$ behaves like Sim in the sense that it runs the rewind phase in the same way as Sim, and outputs $\mathsf{ambiguous}$ under the same conditions. However, $\mathcal{H}_1$ is provided with m, and it commits (via $\mathsf{TLP}^{\mathsf{qpv}}$) to an honest secret sharing of $m||r$ and commits (via Com) to honest views from the MPC in the head. Formally, $\mathcal{H}_1$ is defined as follows:

1. sample randomness $r \leftarrow \mathbb{F} \setminus \{0\}$; $\forall i \in [\lambda]$, compute the first message $\sigma_i[1]$ of SHCom, sample $v_{i,1}, v_{i,2} \leftarrow \{0,1\}^{2\lambda}$ and compute $v_{i,3} = v_{i,1} \oplus v_{i,2} \oplus (m||r)$; $\forall i \in [\lambda], j \in [3]$, sample random $\beta_{i,j}$ and compute $z_{i,j} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{i,j}; \beta_{i,j})$; **send** $(\sigma_i[1])_{i\in[\lambda]}$ and $(z_{i,j})_{i\in[\lambda],j\in[3]}$ to R_c^*.
2. **receive** $\forall i \in [\lambda]$ from R_c^* its statistically hiding commitment $\sigma_i[2]$ to (a_i, b_i), and the value α.
3. $\forall i \in [\lambda]$, run Π_ϕ in the head, with public inputs α and γ, and $v_{i,j}$ as the private input to party j in the i-th execution of Π_ϕ. Let $\mathsf{view}_{i,1}, \mathsf{view}_{i,2}, \mathsf{view}_{i,3}$ be the views of the i-th execution;
4. $\forall i \in [\lambda], j \in [3]$, compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$; **send** $(c_{i,j})_{i\in[\lambda],j\in[3]}$ and $\gamma := r\alpha + m$ to R_c^*.
5. **receive** $\forall i \in [\lambda]$ from R_c^* the decommitments y_i w.r.t. σ_i to the challenge indices (a_i, b_i); **abort** if any of the decommitments is not valid.
6. **Rewind Phase:** repeatedly rewinds R_c^* back to Step 3:
   (a) $\forall i \in [\lambda]$, run Π_ϕ in the head, with public inputs α and γ, and $v_{i,j}$ as the private input to party j in the i-th execution of Π_ϕ. Let $\mathsf{view}_{i,1}, \mathsf{view}_{i,2}, \mathsf{view}_{i,3}$ be the views of the i-th execution
   (b) $\forall i \in [\lambda], j \in [3]$, compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$
   (c) **send** $(c_{i,j})_{i\in[\lambda],j\in[3]}$ and $\gamma := r\alpha + m$ to R_c^*.
   (d) **receive** $\forall i \in [\lambda]$ from R_c^* the decommitments y'_i w.r.t. σ_i to the challenge indices (a'_i, b'_i); if any of the decommitments y'_i is not valid, execute again Step 6a (with fresh randomness).
   (e) If $\exists i \in [\lambda]$ s.t. $(a_i, b_i) \neq (a'_i, b'_i)$, output $\mathsf{ambiguous}$. Otherwise, exit the rewind phase and proceed Step 7.
7. **send** $(v_{i,a_i}, \mathsf{view}_{i,a_i}, \beta_{i,a_i}, d_{i,a_i}, v_{i,b_i}, \mathsf{view}_{i,b_i}, \beta_{i,b_i}, d_{i,b_i})_{i\in[\lambda]}$ to R_c^*.

$\mathcal{H}_2^i$: This hybrid is parameterized by $i \in \{0, \dots, \lambda\}$. We define $\mathcal{H}_2^0 := \mathcal{H}_1$ and let $\mathcal{H}_2^i$ be identical to $\mathcal{H}_2^{i-1}$ except that in Step 6b, $\mathcal{H}_2^i$ computes $(c_{i,\delta}, d_{i,\delta}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,\delta})$ and $(c_{i,k}, d_{i,k}) \leftarrow \mathsf{Com}(1^\lambda, 0^\ell)$ for $\delta \in \{a_i, b_i\}$ and $k \in [3]$ s.t. $a_i, b_i \neq k$. Formally, $\mathcal{H}_2^i$ is defined as follows:

```
⋮
(6b) $\forall j \in [\lambda]$ and $k \in [3] \setminus \{a_j, b_j\}$:
    • $(c_{j,a_j}, d_{j,a_j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{j,a_j})$
    • $(c_{j,b_j}, d_{j,b_j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{j,b_j})$
    • If $j \leq i$:$(c_{j,k}, d_{j,k}) \leftarrow \mathsf{Com}(1^\lambda, 0^\ell)$
    • If $j > i$: $(c_{j,k}, d_{j,k}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{j,k})$.
⋮
```

$\mathcal{H}_3$: This is identical to $\mathcal{H}_2^\lambda$ except that in Step 6a, the MPC views are simulated. Formally, $\mathcal{H}_3$ is defined as follows:

```
⋮
(6a) $\forall i \in [\lambda]$, compute $(\mathsf{view}_{i,a_i}, \mathsf{view}_{i,b_i}) \leftarrow \mathsf{Sim}_{\mathsf{MPC}}((a_i, b_i), (v_{i,a_i}, v_{i,b_i}), 1)$
     and $\mathsf{view}_{i,k} := 0^\ell$ for $k \in [3] \setminus \{a_i, b_i\}$
⋮
```

$\mathcal{H}_4^i$: This hybrid is parameterized by $i \in \{0, \dots, 3\lambda\}$. We define $\mathcal{H}_4^0 := \mathcal{H}_3$. For $i \geq 1$, $\mathcal{H}_4^i$ is identical to $\mathcal{H}_4^{i-1}$ except that, instead of committing to a share of $m||r$, all the puzzles up until the i-th of the 3λ puzzles sent in Step 1 commit to $0^{2\lambda}$. $\mathcal{H}_4^i$ is formally defined as follows:

```
1. sample randomness $r \leftarrow \mathbb{F} \setminus \{0\}$; $\forall \ell \in [\lambda]$, compute the first message
   $\sigma_\ell[1]$ of $\mathsf{SHCom}$ and sample $v_{\ell,1}, v_{\ell,2} \leftarrow \{0,1\}^{2\lambda}$ and compute $v_{\ell,3} =$
   $v_{\ell,1} \oplus v_{\ell,2} \oplus (m||r)$.
   $\forall k \in [3\lambda]$:
     • $u = \lceil \frac{k}{3} \rceil$, $\delta = (k-1) \mod 3 + 1$, sample randomness $\beta_{u,\delta}$
     • if $k \leq i$, $z_{u,\delta} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, 0^{2\lambda}; \beta_{u,\delta})$
     • else $z_{u,\delta} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{u,\delta}; \beta_{u,\delta})$
   **send** $(\sigma_u[1])_{u \in [\lambda]}$ and $(z_{u,\delta})_{u \in [\lambda], \delta \in [3]}$ to $\mathsf{R}_\mathsf{c}^*$.
⋮
```

$\mathcal{H}_5^i$: This hybrid is parameterized by $i \in \{0, \dots, 3\lambda\}$. We define $\mathcal{H}_5^0 := \mathcal{H}_3$. For $i \geq 1$, $\mathcal{H}_5^i$ is identical to $\mathcal{H}_5^{i-1}$ except that , instead of committing to an MPC view, all the commitments up until the i-th of the 3λ commitments sent in Step 4 commit to $0^{2\lambda}$. $\mathcal{H}_5^i$ is formally defined as follows:

```
⋮
4 $\forall k \in [3\lambda]$:
    • $u = \lceil \frac{k}{3} \rceil$, $\delta = (k-1) \mod 3 + 1$
    • if $k \leq i$, $(c_{u,\delta}, d_{u,\delta}) \leftarrow \mathsf{Com}(1^\lambda, 0^\ell)$;
    • else $(c_{u,\delta}, d_{u,\delta}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{u,\delta})$
  **send** $(c_{u,\delta})_{u \in [\lambda], \delta \in [3]}$ and $\gamma := r\alpha + m$ to $\mathsf{R}_\mathsf{c}^*$.
```

$\vdots$

$\mathcal{H}_6$ This hybrid is identical to $\mathcal{H}_5^{3\lambda}$, but γ is sampled uniformly at random.

Notice that $\mathcal{H}_6$ is identical to Sim. We now introduce the following claims.

Claim 6 *If* SHCom *is computationally binding, then $\mathcal{H}_1$ is statistically indistinguishable from $\mathcal{H}_0$.*

Claim 7 *If* SBCom *is computationally hiding, then for every $i \in [\lambda]$, $\mathcal{H}_2^i$ is computationally indistinguishable from $\mathcal{H}_2^{i-1}$.*

Claim 8 *If Π_ϕ is perfectly 2-private, then $\mathcal{H}_3$ is identically distributed to $\mathcal{H}_2^\lambda$.*

Claim 9 *If* $\mathsf{TLP}^{\mathsf{qpv}}$ *is hard, then for every $i \in [3\lambda]$, $\mathcal{H}_4^i$ and $\mathcal{H}_4^{i-1}$ are indistinguishable for every poly-size distinguisher with depth upper-bounded by t^ϵ.*

Claim 10 *If* SBCom *is computationally hiding, then for every $i \in [3\lambda]$, $\mathcal{H}_5^i$ is computationally indistinguishable from $\mathcal{H}_5^{i-1}$.*

Claim 11 *$\mathcal{H}_6$ is perfectly indistinguishable from $\mathcal{H}_5^{3\lambda}$.*

Proof (of Claim 6). Conditioned on not outputting ambiguous, the output distribution of $\mathcal{H}_1$ is identical to $\mathcal{H}_0$. It remains to show that $\mathcal{H}_1$ outputs ambiguous only with negligible probability. Assuming that there exists an infinite series of inputs that makes $\mathcal{H}_1$ output ambiguous with non-negligible probability, one can easily construct an adversary A for the binding of SHCom. A runs $\mathcal{H}_1$ on such an input and looks for $i \in \lambda$ s.t.$(a_i, b_i) \neq (a_i', b_i')$ and both pairs have a valid decommitment sent by $\mathsf{R_c}^*$ when interacting with $\mathcal{H}_1$. A simply re-uses such decommitments in the binding game. The only subtlety is that $\mathcal{H}_1$ runs in expected polynomial time, whereas A must run in strict polynomial time. Nevertheless, this issue can be addressed by simply truncating $\mathcal{H}_1$ to twice its expected running time. By Markov's inequality, this adjustment decreases the attack's success probability against the binding of SHCom of at most 1/2, which remains non-negligible. □

Proof (of Claim 9). If there exists a polynomial-size distinguisher D_λ, whose depth is bounded by t^ϵ, distinguishing with noticeable probability $\mathcal{H}_4^i$ from $\mathcal{H}_4^{i-1}$, then we can use D to build a bounded-depth adversary A against the hardness property of $\mathsf{TLP}^{\mathsf{qpv}}$. A runs the instructions of $\mathcal{H}_4^{i-1}$ with one change. That is, when A has to compute the i-th puzzle at Step 1 it plays in the hardness game of $\mathsf{TLP}^{\mathsf{qpv}}$ by sending s_0 equal to a share of $m||r$ and $s_1 = 0^{2\lambda}$ and uses the puzzle z that it gets back from the challenger as the i-th puzzle of the simulation. When the simulation concludes, then A invokes D on the output generated by the simulator, and outputs whatever D outputs. Notice that when $b = 0$ in the hardness game, A perfectly simulates $\mathcal{H}_4^{i-1}$, otherwise it perfectly simulates $\mathcal{H}_4^i$. Therefore, A wins the hardness game with the same probability with which D distinguishes between $\mathcal{H}_4^i$ and $\mathcal{H}_4^{i-1}$. The only subtlety is that $\mathcal{H}_4^{i-1}$ runs in expected polynomial time essentially independent of t, whereas A must run in strict polynomial time essentially independent of t. Nevertheless, this issue can be addressed by simply truncating the running time of $\mathcal{H}_4^{i-1}$. □

The proof of Claim 7 and 10 are analogous to the one of 2, with the only difference that we have to truncate the running time of $\mathcal{H}_2^{i-1}$. The proofs of Claim 8 and 11 are identical to the ones of 4 and 5 respectively.

The proof is concluded by observing that it simply follows from the proofs of Claims 6 -11, since all of the hybrids are indistinguishable by a poly-size distinguisher whose depth is bounded by t^{ϵ}. □

Lemma 7. HRTC *is sound if* $\mathsf{TLP}^{\mathsf{qpv}}$ *is a quasi publicly verifiable TLP,* SBCom *is a non-interactive statistically biding commitment scheme,* SHCom *is a two-round statistically hiding commitment scheme, and* Π_{ϕ} *is a* 3*-party secure MPC.*

Proof (of Lemma 7). The only difference with Lemma 2 is the presence of statistically hiding commitments going from the receiver to the malicious sender. Since such commitments statistically convey no information about the receiver's challenges, soundness is argued similarly. □

3.5 A More Efficient Force-Open

By carefully modifying FOpen and FVerify, we can get a much more efficient force-open phase of our TC constructions (Sects. 3.3 and 3.4). In the modified TC scheme, commitments can be force opened by solving $\log(\lambda)$, instead of λ, puzzles. The intuition is that after a valid commit phase, the value γ can be used to check if, after having solved the puzzle associated with a certain repetition $i \in [\lambda]$, the recovered message is the right one. By right we mean that the sender did not cheat in the i-th repetition and thus the recovered message coincides with the one committed in (the vast majority of) the other repetitions.

In a nutshell, FOpen force-opens random repetitions until it finds one where γ is consistent with the recovered message m and the proof π given in output by $\mathsf{TLP}^{\mathsf{qpv}}$.Solve is accepting. Since, with overwhelming probability, there are at most $\log^2(\lambda)$ bad repetitions (i.e., where γ is not consistent with m or π is not accepting) after a valid commit phase (see the proof of Lemma 2), the probability of only finding bad repetitions with more than $\log(\lambda)$ tries is negligible. Whenever FOpen finds a good repetition, it outputs the repetition's index i and the recovered message m, along with the share and the proof given in output by $\mathsf{TLP}^{\mathsf{qpv}}$.Solve. Then, FVerify just verifies that: (1) m coincides with the one reconstructed from the shares of the i-th repetition, (2) m is consistent with γ, and (3) the proof is accepting w.r.t. the unopened puzzle of the i-th repetition.

However, this efficiency improvement comes at the price of only satisfying *weak* public verifiability. Recall that in the weak public verifiability game (Def. 3), the commitment is not directly provided by the adversary but it is the result of a commit phase run with the honest receiver. Intuitively, we can only get weak public verifiability with this modification because the commitment (transcript) com could be simulated by an adversary not interacting with a receiver at all. Therefore, every repetition could possibly contain a different message while still being consistent w.r.t. γ. The modified FOpen and FVerify are in Fig. 5.

```
$(m, \pi) \leftarrow \mathsf{FOpen}(1^\lambda, 1^t, \mathsf{com})$: Perform the following steps:
    1. Set $I := [\lambda]$ and parse $\mathsf{com}$ as in (2).
    2. Sample random $i \leftarrow I$. Let $k \in [3]$ be s.t. $a_i, b_i \neq k$ and compute
       $(v_{i,k}, \pi_{i,k}) \leftarrow \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}(1^\lambda, 1^t, z_{i,k})$
    3. Check if $v = v_{i,a_i} \oplus v_{i,b_i} \oplus v_{i,k}$ can be parsed as $m||r$ so that $\gamma = \alpha r + m$
    4. Check if $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Verify}(1^\lambda, t, z_{i,k}, v_{i,k}, \pi_{i,k}) = 1$
    5. If all the checks are successful output $m$ and proof $\pi = (v_{i,k}, \pi_{i,k}, i, k)$.
       Otherwise, update $I = I \setminus \{i\}$. If $|I| \geq \lambda - \log(\lambda)$, go back to Step 2,
       otherwise output $(\perp, \perp)$.
$b := \mathsf{FVerify}(1^\lambda, t, \mathsf{com}, m, \pi)$: Set $b = 1$ iff all the following checks are successful:
    1. Parse $\mathsf{com}$ as in (2) and $\pi$ as $(v_{i,k}, \pi_{i,k}, i, k)$
    2. Check that $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Verify}(1^\lambda, t, z_{i,k}, v_{i,k}, \pi_{i,k}) = 1$
    3. Parse $v_{i,a_i} \oplus v_{i,b_i} \oplus v_{i,k}$ as $m||r$ with $|m| = |r|$ and check that $\gamma = \alpha r + m$.
```

Fig. 5: Modified FOpen and FVerify algorithms.

Theorem 4. *Let* $\mathsf{TLP}^{\mathsf{qpv}} = (\mathsf{Gen}, \mathsf{Solve}, \mathsf{Verify})$ *be a quasi publicly verifiable TLP (Def. 8),* $\mathsf{SBCom} = (\mathsf{Com}, \mathsf{Dec})$ *a non-interactive statistically binding and computationally hiding commitment scheme,* SHCom *a two-round statistically hiding and computationally binding commitment scheme, and* Π_ϕ *a* 3*-party secure MPC for functionality* ϕ *defined in* (1)*, then the scheme* HRTC *from Sect. 3.3 (respectively* TC *from Sect. 3.4) where the algorithms* FOpen *and* $\mathsf{FVerify}$ *are modified as reported in Fig. 5 is honest-verifier (respectively t-hiding) timed commitment scheme with weak public verifiability.*

To prove the above theorem, we only need to argue soundness and weak public verifiability of the modified schemes. Indeed, we do not need to prove (honest-receiver) t-hiding and binding again as these property only involve the commit and open phases, which are left unaltered.

Lemma 8. $\mathsf{C} \in \{\mathsf{HRTC}, \mathsf{TC}\}$ *where the algorithms* FOpen *and* $\mathsf{FVerify}$ *are modified as reported in Fig. 5 is sound if* $\mathsf{TLP}^{\mathsf{qpv}}$ *is a QPV-TLP,* SBCom *is a non-interactive statistically biding commitment scheme,* Π_ϕ *is a* 3*-party secure MPC.*

Proof (of Lemma 8). Let us call a repetition $i \in [\lambda]$ bad if $\phi(\alpha, \gamma, v_{i,1}, v_{i,2}, v_{i,3}) = 0$ or $\exists\ u \in [3]$ such that the TLP $z_{i,u}$ is malformed. Recall that conditioned on $\mathsf{com} \neq \perp$, the probability that a repetition is bad at most $\frac{1}{3}$ (see the proof of Lemma 2). As a result, if the commit phase is successful, i.e., $\mathsf{com} \neq \perp$, the probability that at least $\log^2(\lambda)$ repetitions are bad is at most $(\frac{1}{3})^{\log^2(\lambda)} < (\frac{1}{2})^{\log^2(\lambda)} = (\frac{1}{\lambda})^{\log(\lambda)}$, which is negligible. Therefore, the probability that all $\log(\lambda)$ samples of FOpen are bad is at most $(\frac{\log^2(\lambda)}{\lambda})^{\log(\lambda)}$, which is negligible[7].

[7] Notice that eliminating a bad repetition from the set of repetitions that may be forced-open only further reduces the probability of encountering a bad repetition. Indeed, $\log^2(\lambda) < \lambda$ and $\frac{n}{k} > \frac{n-i}{k-i}$ for any $n, k > 1, n < k$, and $i \geq 1$.

Let $S \subseteq [\lambda]$ be the set containing the indices of the repetitions that are not bad, from the above discussion it follows that $|S| \geq \lambda - \log^2(\lambda)$ except with negligible probability. For every $i \in S$ where $v_{i,1} \oplus v_{i,2} \oplus v_{i,3} = m_i || r_i$ for $|m_i| = r_i$, it holds that $\gamma = \alpha r_i + m_i$. By the Schwartz-Zippel lemma, with overwhelming probability over the choice of α, there must exist (m, r) such that $m_i = m$ and $r_i = r$ for all $i \in S$. Recall that $\forall i \in S$ and $k \in [3]$ s.t. $a_i, b_i \neq k$ we have that $z_{i,k}$ is well-formed. The correctness and the completeness of $\mathsf{TLP}^{\mathsf{qpv}}$ guarantee that whenever a puzzle is correctly generated, the solution/proof pair output by $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}$ will be accepting and will contain the committed value. Recall that a successful honest open phase outputs the same message m' in more than $\lambda/2$ repetitions. Thus, since $|S| \geq \lambda - \log^2(\lambda)$, except with negligible probability, m' coincides with the message m output by FOpen (recall that a repetition can only be opened in one way since $\mathsf{TLP}^{\mathsf{qpv}}$ is perfectly binding). Additionally, $\mathsf{FVerify}$ outputs 1 on input the message/proof pair returned by FOpen. □

Lemma 9. $\mathsf{C} \in \{\mathsf{HRTC}, \mathsf{TC}\}$ *where the algorithms* FOpen *and* $\mathsf{FVerify}$ *are modified as reported in Fig. 5 is weak publicly verifiable if* $\mathsf{TLP}^{\mathsf{qpv}}$ *is a QPV-TLP.*

Proof (of Lemma 9). Assume A outputs two accepting proof/message pairs $(m, \pi = (v_{i,k}, \pi_{i,k}, i, k))$, and $(m', \pi' = (v_{i',k'}, \pi_{i',k'}, i', k'))$ with $m \neq m'$ for the same $\mathsf{com} = (z_{i,1}, z_{i,2}, z_{i,3}, \alpha, \gamma, a_i, b_i, v_{i,a_i}, \beta_{i,a_i}, v_{i,b_i}, \beta_{i,b_i})_{i \in [\lambda]}$. If $i = i'$, then $m = m'$ since, due to the perfect soundness of $\mathsf{TLP}^{\mathsf{qpv}}$, there exists only one possible share $v_{i,k}$ for which an accepting proof $\pi_{i,k}$ can be provided for the puzzle $z_{i,k}$. Together with v_{i,a_i}, v_{i,b_i}, the share $v_{i,k}$ fixes the message m. Let us assume $i \neq i'$. As already discussed above, due to the perfect soundness of $\mathsf{TLP}^{\mathsf{qpv}}$, within each repetition $\ell \in [\lambda]$ there exists only a possible share $v_{\ell,k}$ for which an accepting proof $\pi_{\ell,k}$ can be provided for the puzzle $z_{\ell,k}$. Hence, the malicious sender is bound to a single $v_{\ell,k}$, which results in a single $v_\ell = v_{\ell,a_\ell} \oplus v_{\ell,b_\ell} \oplus v_{\ell,k}$ within each repetition. Thus at the end of the commit phase, the sender is bound to a single v_ℓ in each repetition. By the Schwartz-Zippel lemma we know that, with overwhelming probability over the choices of α, for every $v_\ell = m_\ell || r_\ell$ such that $\gamma = \alpha r_\ell + m_\ell$ it must be that all m_ℓ are such that $m_\ell = m$. Hence, with overwhelming probability, the two force opening proofs provided by the adversary will point to two repetition indices i and i' which contain $v_i = m || r$ and $v_{i'} = m' || r'$ such that $m = m'$. □

Acknowledgements

We thank Peter Chvojka and Ivan Visconti for insightful comments and discussions. We thank the anonymous reviewers of TCC 2024 for their feedback. This result is part of projects that have received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program under project PICOCRYPT (grant agreement No. 101001283), and from the Spanish Government under projects PRODIGY (TED2021-132464B-I00) and ESPADA (PID2022-142290OB-I00). The last two projects are co-funded by European Union EIE, and NextGenerationEU/PRTR funds.

References

AMZ24. Shweta Agrawal, Giulio Malavolta, and Tianwei Zhang. Time-lock puzzles from lattices. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part III*, volume 14922 of *Lecture Notes in Computer Science*, pages 425–456. Springer, 2024.

BDD$^+$21. Carsten Baum, Bernardo David, Rafael Dowsley, Jesper Buus Nielsen, and Sabine Oechsner. TARDIS: A foundation of time-lock puzzles in UC. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part III*, volume 12698 of *LNCS*, pages 429–459, Zagreb, Croatia, October 17–21, 2021. Springer, Heidelberg, Germany.

BDD$^+$23. Carsten Baum, Bernardo David, Rafael Dowsley, Ravi Kishore, Jesper Buus Nielsen, and Sabine Oechsner. CRAFT: Composable randomness beacons and output-independent abort MPC from time. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 439–470, Atlanta, GA, USA, May 7–10, 2023. Springer, Heidelberg, Germany.

BGI$^+$01. Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 1–18, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany.

BGJ$^+$16. Nir Bitansky, Shafi Goldwasser, Abhishek Jain, Omer Paneth, Vinod Vaikuntanathan, and Brent Waters. Time-lock puzzles from randomized encodings. In Madhu Sudan, editor, *ITCS 2016*, pages 345–356, Cambridge, MA, USA, January 14–16, 2016. ACM.

BN00. Dan Boneh and Moni Naor. Timed commitments. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 236–254, Santa Barbara, CA, USA, August 20–24, 2000. Springer, Heidelberg, Germany.

Can01. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145, Las Vegas, NV, USA, October 14–17, 2001. IEEE Computer Society Press.

CDMW09. Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Simple, black-box constructions of adaptively secure protocols. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 387–402. Springer, Heidelberg, Germany, March 15–17, 2009.

CGJ$^+$23. Arka Rai Choudhuri, Sanjam Garg, Abhishek Jain, Zhengzhong Jin, and Jiaheng Zhang. Correlation intractability and snargs from sub-exponential ddh. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023*, pages 635–668, Cham, 2023. Springer Nature Switzerland.

CJ23. Peter Chvojka and Tibor Jager. Simple, fast, efficient, and tightly-secure non-malleable non-interactive timed commitments. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 500–529, Atlanta, GA, USA, May 7–10, 2023. Springer, Heidelberg, Germany.

CJJ22. Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. SNARGs for $\mathcal{P}$ from LWE. In *62nd FOCS*, pages 68–79, Denver, CO, USA, February 7–10, 2022. IEEE Computer Society Press.

CLP20. Rohit Chatterjee, Xiao Liang, and Omkant Pandey. Improved black-box constructions of composable secure computation. Cryptology ePrint Archive, Report 2020/494, 2020. `https://eprint.iacr.org/2020/494`.

COS22. Michele Ciampi, Emmanuela Orsini, and Luisa Siniscalchi. Four-round black-box non-malleable schemes from one-way permutations. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part II*, volume 13748 of *LNCS*, pages 300–329, Chicago, IL, USA, November 7–10, 2022. Springer, Heidelberg, Germany.

FKPS21. Cody Freitag, Ilan Komargodski, Rafael Pass, and Naomi Sirkin. Non-malleable time-lock puzzles and applications. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part III*, volume 13044 of *LNCS*, pages 447–479, Raleigh, NC, USA, November 8–11, 2021. Springer, Heidelberg, Germany.

GK96. Oded Goldreich and Ariel Kahan. How to construct constant-round zero-knowledge proof systems for NP. *J. Cryptol.*, 9(3):167–190, 1996.

GLOV12. Vipul Goyal, Chen-Kuei Lee, Rafail Ostrovsky, and Ivan Visconti. Constructing non-malleable commitments: A black-box approach. In *53rd FOCS*, pages 51–60, New Brunswick, NJ, USA, October 20–23, 2012. IEEE Computer Society Press.

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, New York City, NY, USA, May 25–27, 1987. ACM Press.

Gol01. Oded Goldreich. *The Foundations of Cryptography - Volume 1: Basic Techniques.* Cambridge University Press, 2001.

HJKS22. James Hulett, Ruta Jawale, Dakshita Khurana, and Akshayaram Srinivasan. SNARGs for P from sub-exponential DDH and QR. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 520–549, Trondheim, Norway, May 30 – June 3, 2022. Springer, Heidelberg, Germany.

HM96. Shai Halevi and Silvio Micali. Practical and provably-secure commitment schemes from collision-free hashing. In Neal Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 201–215, Santa Barbara, CA, USA, August 18–22, 1996. Springer, Heidelberg, Germany.

HV16. Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. On the power of secure two-party computation. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 397–429, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany.

IKLP06. Yuval Ishai, Eyal Kushilevitz, Yehuda Lindell, and Erez Petrank. Black-box constructions for secure computation. In Jon M. Kleinberg, editor, *38th ACM STOC*, pages 99–108, Seattle, WA, USA, May 21–23, 2006. ACM Press.

IKOS07. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In David S. Johnson and Uriel Feige, editors, *39th ACM STOC*, pages 21–30, San Diego, CA, USA, June 11–13, 2007. ACM Press.

JMRR21. Samuel Jaques, Hart Montgomery, Razvan Rosie, and Arnab Roy. Time-release cryptography from minimal circuit assumptions. In Avishek Adhikari, Ralf Küsters, and Bart Preneel, editors, *Progress in Cryptology –*

INDOCRYPT 2021, pages 584–606, Cham, 2021. Springer International Publishing.

Kiy14. Susumu Kiyoshima. Round-efficient black-box construction of composable multi-party computation. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 351–368, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany.

Kiy20. Susumu Kiyoshima. Round-optimal black-box commit-and-prove with succinct communication. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 533–561, Santa Barbara, CA, USA, August 17–21, 2020. Springer, Heidelberg, Germany.

Kiy23. Susumu Kiyoshima. Holographic snargs for p and batch-np from (polynomially hard) learning with errors. In Guy Rothblum and Hoeteck Wee, editors, *Theory of Cryptography*, pages 333–362, Cham, 2023. Springer Nature Switzerland.

KLX20. Jonathan Katz, Julian Loss, and Jiayu Xu. On the security of time-lock puzzles and timed commitments. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part III*, volume 12552 of *LNCS*, pages 390–413, Durham, NC, USA, November 16–19, 2020. Springer, Heidelberg, Germany.

KOS18. Dakshita Khurana, Rafail Ostrovsky, and Akshayaram Srinivasan. Round optimal black-box "commit-and-prove". In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 286–313, Panaji, India, November 11–14, 2018. Springer, Heidelberg, Germany.

Lin13. Yehuda Lindell. A note on constant-round zero-knowledge proofs of knowledge. *Journal of Cryptology*, 26(4):638–654, October 2013.

Lin16. Yehuda Lindell. How to simulate it - a tutorial on the simulation proof technique. Cryptology ePrint Archive, Paper 2016/046, 2016. `https://eprint.iacr.org/2016/046`.

LM23. Russell W. F. Lai and Giulio Malavolta. Lattice-based timed cryptography. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part V*, volume 14085 of *Lecture Notes in Computer Science*, pages 782–804. Springer, 2023.

LP12. Huijia Lin and Rafael Pass. Black-box constructions of composable protocols without set-up. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 461–478, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Heidelberg, Germany.

LPS17. Huijia Lin, Rafael Pass, and Pratik Soni. Two-round and non-interactive concurrent non-malleable commitments from time-lock puzzles. In Chris Umans, editor, *58th FOCS*, pages 576–587, Berkeley, CA, USA, October 15–17, 2017. IEEE Computer Society Press.

Pie19. Krzysztof Pietrzak. Simple verifiable delay functions. In Avrim Blum, editor, *ITCS 2019*, volume 124, pages 60:1–60:15, San Diego, CA, USA, January 10–12, 2019. LIPIcs.

PW09. Rafael Pass and Hoeteck Wee. Black-box constructions of two-party protocols from one-way functions. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 403–418. Springer, Heidelberg, Germany, March 15–17, 2009.

RSW96. R. L. Rivest, A. Shamir, and D. A. Wagner. Time-lock puzzles and timed-release crypto. Technical report, USA, 1996.

SW14. Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *46th ACM STOC*, pages 475–484, New York, NY, USA, May 31 – June 3, 2014. ACM Press.

TCLM21. Sri Aravinda Krishnan Thyagarajan, Guilhem Castagnos, Fabien Laguillaumie, and Giulio Malavolta. Efficient CCA timed commitments in class groups. In Giovanni Vigna and Elaine Shi, editors, *ACM CCS 2021*, pages 2663–2684, Virtual Event, Republic of Korea, November 15–19, 2021. ACM Press.

Wes19. Benjamin Wesolowski. Efficient verifiable delay functions. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 379–407, Darmstadt, Germany, May 19–23, 2019. Springer, Heidelberg, Germany.

WW24. Brent Waters and David J. Wu. Adaptively-sound succinct arguments for NP from indistinguishability obfuscation. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 387–398. ACM, 2024.

ZMM$^+$20. Fan Zhang, Deepak Maram, Harjasleen Malvai, Steven Goldfeder, and Ari Juels. DECO: Liberating web data using decentralized oracles for TLS. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 1919–1938, Virtual Event, USA, November 9–13, 2020. ACM Press.